UNIVERSITY OF TORINO

DIPARTIMENTO DI MATEMATICA GIUSEPPE PEANO
DIPARTIMENTO ESOMAS
DIPARTIMENTO DI INFORMATICA

SCUOLA DI SCIENZE DELLA NATURA

M.Sc. in Stochastics and Data Science

Final dissertation

Community Detection in Stochastic Block Models with Unknown Number of Communities: a Bayesian Nonparametric Approach

Supervisor: Prof. Pierpaolo De Blasi

Candidate: Michele Chini
ID number: 899694

ACADEMIC YEAR 2019/2020

Ringraziamenti

Questa tesi è dedicata a tutte le persone che si sentono semplicemente normali, sempre fuori posto, mai arrivate e pronte a cambiare strada. Perchè forse, la strada giusta è quella che stiamo già percorrendo.

Desidero ringraziare il Prof. Pierpaolo De Blasi. Oltre che a supportarmi e chiarire i miei numerosi dubbi con disponibilità e gentilezza fuori dal comune, durante questo periodo di lavoro mi ha trasmesso la passione e l'entusiasmo affinchè questa tesi potesse predere forma giorno dopo giorno.

Ringrazio *Gli Eletti*, la mia seconda famiglia in cui mi sento a casa anche a distanza di chilometri. In particolare ringrazio Alessandro, Alessio, e Daniele per il loro instancabile supporto che mi hanno fornito in questi tre anni pieni di sfide e di ostacoli.

Ringrazio Vito e Giorgia. Grazie per essere sempre al mio fianco malgrado la distanza che ci separa.

Ringrazio Gabriele, se c'è un aspetto positivo di questa pandemia è stato il riallacciare i rapporti con te. Le risate con te sono e rimarranno impareggiabili. Dai banchi del liceo fin qua. Ed oltre.

Infine, il ringraziamento più grande va alla mia famiglia. A Mamma e Papà per avermi supportato in tutti questi anni ed ai miei fratelli, Francesco ed Emanuele che mi hanno sempre aiutato in questo percorso universitario. Un ringraziamento particolare va a Francesco, grazie per avermi sopportato durante questo difficile anno di pandemia e grazie per le tue preziose lezioni di analisi funzionale.

Un ringraziamento speciale, infine, a tutte le persone che per motivi di spazio non ho potuto menzionare e che mi sono state vicine in questi anni.

Contents

1. Introduction

Nowadays, data available in the form of networks are increasingly common in modern applications ranging from brain remote activity, protein interactions, web applications and social networks. Accordingly, in recent years there has been an explosion of activities in the statistical analysis of networks. Among various methodological and theoretical developments, the problem of community detection has received widespread attention. Indeed, the large-scale structure of empirically observed networks such as social, biological, and technological networks, is often complex and difficult to comprehend and thus, community detection represents one of the most effective tools for reducing this complexity to a level where network topology can be more easily understood and interpreted.

Community detection consists in the division of the nodes of a network into densely connected groups with only sparse between-group connections. Various algorithms have been developed for doing community detection and a fundamental limitation of most of these is that they must know in advance the number of groups in which to divide the observations. Tipically this information is not available leading thus to a loss of utility of such approaches.

Different strategies to address this problem have been proposed on the algorthmic side, among which the best known is perhaps the Louvain algorithm which is a method for choosing both the number of communities and the community division itself. Nevertheless, it suffers from being only heuristically motivated and there are instances where it is known to give incorrect results. These issues have led to a growing interest in model-based solutions which rely on generative statistical models. In particular, those consist in methods of statistical inference in which one defines a model for networks with community structure and then fits that model to observed network data. The parameters of the the fit tell us about the community structure in much the same way that the fit of a straight line through a set of data points can tell us about their slope. The most commonly employed model in this context is the stochastic blockmodel (SBM) owing also to its unique balance between simplicity and flexibility.

In SBMs the probability of an edge between two nodes only depend on their cluster memberships allowing an efficient inference on both communities and block probabilities which characterize assortative, disassortative, core-periphery or weak community patterns. These desirable properties have motivated extensive theoretical studies and various generalizations of the original SBM, most of them aiming at addressing its fundamental limitation. Classical SBM formulations, indeed, are based on a pre–specified number of communities making them not suitable in real contexts. In this work we

committed to deepen this issue focusing on the Bayesian nonparametric answer, adopting the mixture of finite mixtures version of SBM developed by Geng et al. (2019).

In Chapter 2 we present the essential statistical tools we used to tackle the problem. We give a brief introduction to the motivation behind the Bayesian nonparametric approach and we introduce the fundamental concepts on which it is based, such as the Dirichlet process, its properties and the Gibbs Type priors.

In Chapter 3 we discuss community detection. We start with a brief overview of the different approaches used in literature, which can be divided in two main classes: the algorithmic and the model-based one. Firstly we focus on the most widely used method among those of the first class, the Louvain algorithm and then, for the model-based class, we talk about community detection in stochastic blockmodel. We treat both the case where the number of communities is known and the case in which the number of cummunities is not known, introducing for this one the Mixture of Finite Mixtures formulation of the Stochastic BlockModel (MFM-SBM model). We discuss the Gibbs sampler for the MFM-SBM model developed by Geng et al. (2019) and we describe the choices we made in the implementation on the software R.

In Chapter 4 we present the empirical studies we conducted. In particular, we decided to develop a simulation study and to test the MFM-SBM algorithm on a benchmark real network: the dolphin social network. In both cases we tested the MFM-SBM model with two different priors for the number of communities: the zero-truncated-Poisson prior and the Gnedin prior. The latter, indeed, allows to write in an exact way, and hence more conveniently, the coefficient $V_n(\cdot)$ used in the Gibbs sampler. In the simulation study, for each distribution, we tested five different values for the relative parameters in order to investigate the sensitivity of the MFM-SBM algorithm to parameter variation. We set up eight test scenarios in which we varied the size of the network, the number of communities and the type of community structure present in the network: vague and prominent. Despite the overall performance was very similar, we noticed a small advantage of the Gnedin prior over the zero-truncated-Poisson in terms of execution time. This is due to the fact that, as mentioned above, the calculation of $V_n(\cdot)$ is more efficient. Regarding the results gotten in the dolphin social network data, we noted how a non-informative parameterization for the Gnedin prior allowed an improvement upon the zero-truncated-Poisson in terms of community structure estimation. However, the findings with both priors proved to be satisfactory especially when compared to those obtained with the Louvain algorithm.

Finally, in Chapter 5 we summarise our work and we suggest some possible future directions.

2. Bayesian Nonparametrics

2.1. Motivation and Theoretical framework

"Why adopt the nonparametrics Bayesian approach for inference?
The answer lies in the simultaneous preference for nonparametrics modeling and desire to follow a Bayesian procedure. Nonparametrics models can allow one to avoid the arbitrary and possibly unverifiable assumptions inherent in parametric models. Bayesian procedures may be desirable because of the conceptual simplicity of the Bayesian paradigm, philosophical reasons"(Ghosal and van der Vaart, 2017).

In statistics the term parametric is used to indicate the complete identifiability of every member of the distributions family used to model the data by means a finite-dimensional parameter, let say $\theta \in \Theta$, where $dim(\theta) < \infty$. This allows for example to make inference about a particular problem focusing not on the distribution itself but on the characterizing parameter. Despite its mathematical and conceptual convenience, takling a statistical problem by means a parametric framework might be limiting and misleading (Müller and Mitra, 2013). Traditional parametric models, indeed, using a fixed and finite number of parameters can suffer from over- or under-fitting when there is a mismatch between the complexity of the model (often expressed in terms of the number of parameters) and the amount of data available.

Following Teh (2010) and Müller and Mitra (2013), suppose we observe $y_1, \ldots, y_n$ with $y_i \sim F$ independent and identical draws from an unknown distribution F. A Bayesian would approach this problem by placing a prior over F and then computing the posterior over F given data. Traditionally, the prior over distributions is represented by a parametric family of distribution in which a finite-dimensional parameter uniquely identifies every member. Hence inference can be done on the parameter rather than on the distribution itself. However, as pointed out before, constraining distributions to lie within a parametric family limits the scope and the type of inference that can be made.

Let us think for example that F represents a mixture distribution. A natural question would be how many classes should be used in the mixture model in order to make inference. A strategy with the lowest constraint on the prior beliefs, regarding the data generating mechanism, should be preferred and a Bayesian nonparametrics approach (BNP) represents a valid choice. In particular a BNP approach consists in fiting a single model that can adapt its complexity to the data (Gershman and Blei, 2012; Ghosal and van der Vaart, 2017) using a prior over distributions with wide support,

typically the support being the space of all distributions (Teh, 2010; Müller and Mitra, 2013). Since the number of parameters required to characterise each distribution is infinite, in such a case the unknown quantities are infinite dimensional from which follows the nonparametrics terminology. Therefore, in a nonparametrics model there are still parameters but there are infinitely many. Specifying infinitely many parameters is not feasible and hence a higher level of parameters are introduced which control hierarchically the law of the infinitely-many lower-level parameters. In such a large space over which we make inference, it is important that posterior computations remain tractable.

2.1.1. Formal Framework

BNP methods take their name from the definition of nonparametrics as not describable by a finite number of parameters and they provide for placing a prior distribution directly on the law of the data. However, this comes with some technical complications. We refer the reader to to chapters two and three of Ghosal and van der Vaart (2017) for a throrough discussion. Here we confine ourselves to a strictly required coverage.

Assume that the sample space $(\mathfrak{X}, \mathscr{X})$ is a Polish space, and consider priors on the collection $\mathfrak{M} = \mathfrak{M}(\mathfrak{X})$ of all probability measures on $(\mathfrak{X}, \mathscr{X})$. A prior Π on $\mathfrak{M}$ can be viewed as the law of a random measure P (a map from some probability space, let us say $(\Omega, \mathscr{F}, \mathbb{P})$ into $\mathfrak{M}$) and can be identified with the collection of *random probabilities* $P(A)$ of sets $A \in \mathscr{X}$.

$(P(A) : A \in \mathscr{X})$ is thus a stochastic process on the underlying probability space. Following Ghosal and van der Vaart (2017) $\mathscr{M}$ is set equal to the smallest $\sigma-$field that makes all maps $M \mapsto M(A)$ from $\mathfrak{M}$ to $\mathbb{R}$ measurable and are considered priors Π that are measures on $(\mathfrak{M}, \mathscr{M})$. With respect to the $\sigma-$field $\mathscr{M}$ on the parameter set $\mathfrak{M}$, P is a Random Probability Measure (RPM) which satifies the following:

- $P \mapsto P(A)$ is $\mathscr{M}$-measurable for every $A \in \mathscr{X}$,
- $A \mapsto P(A)$ is a probability measure for every realization of P.

Note that this mathematically justifies speaking of "drawing a measure P from the prior Π and next sampling observations X from P". The fact that $\mathscr{M}$ is generated by all maps $M \mapsto M(A)$, for $A \in \mathscr{X}$, implies that a map $P : (\Omega, \mathscr{U}, P) \to (\mathfrak{M}, \mathscr{M})$ defined on some probability space is measurable precisely if the induced measure $P(A)$ of every set $A \in \mathscr{X}$ is a random variable. Thus, as far as measurability goes, a random probability measure can be identified with a random element $(P(A) : A \in \mathscr{X})$ in the product space $[0, 1]^{\mathscr{X}}$.

It should be noted that a measurability structure linked to the weak topology (or topology of convergence in distribution) has been considered. However, other topologies are possibile and leads to different measurability structures. Ultimately, RPMs can be constructed in several ways. We will consider only a few. We refer to Ghosal and

van der Vaart (2017) for an overview. We instead investigate a specific RPM which is a cornerstone in BNP: the Dirichlet process.

Before introducing the Dirichlet process, we remember that proving the existence of a RPM is typically done through *De Finetti's* theorem. De Finetti's representation theorem is of profound importance in statistics as it provides a formal and coherent mathematical foundation for all Bayesian statistical theory.

Theorem 2.1. *A sequence $(X_n)_{n\geq 1}, X_i \in \mathfrak{X}$ is exchangeable if and only if there exists a random probability measure P on $(\mathfrak{X}, \mathscr{X})$ such that, conditional on $P = p$, $X_i \overset{iid}{\sim} p$. Furthermore, as $n \to \infty$, the empirical distribution $\frac{1}{n}\sum_{i=1}^{n} \delta_{X_i}$ conveges weakly on a set of $\mathbb{P}$ probability one to a RPM with the same law of P.*

Let us make some comments on this theorem.

(a) The statement about the sequence being conditionally *iid* can be written

$$\mathbb{P}(X_1 \in A_1, \ldots, X_n \in A_n | P = p) = \prod_{i=1}^{n} p(A_i), \tag{2.1}$$

so exchangeability is equivalent to conditional independence and identity in distribution, given the realization of P. For brevity, we usually write $X_i|P \overset{iid}{\sim} P$

(b) Marginalising out p in (2.1), we obtain

$$\mathbb{P}(X_1 \in A_1, \ldots, X_n \in A_n) = \int_{\mathfrak{M}} \prod_{i=1}^{n} p(A_i)\Pi(dp) \tag{2.2}$$

hence the theorem states that the law of an exchangeable sequence is a mixture of laws of iid sequences. The mixing measure Π is called *de Finetti measure* of the sequence $(X_n)_{n\geq 1}$. Equivalently, this can be expressed in the so called *hierchical form* by writing

$$\begin{aligned} P &\sim \Pi \\ X_i|P &\overset{iid}{\sim} P. \end{aligned} \tag{2.3}$$

2.2. Dirichlet Process Prior

Introduced by Ferguson (1973), the *"Dirichlet process is the normal distribution of Bayesian nonparametrics. It is the default prior on spaces of probability measures, and a building block for priors on other structures"* (Ghosal and van der Vaart, 2017).

We use the same notation indroduced in section 2.2 to state formally what a Diriclet process is.

2.2.1. Finite-Dimensional Dirichlet Distribution

The Dirichlet distribution extends the Beta distribution to K-dimensional vectors. Define the $(K-1)$-dimensional simplex

$$\Delta_k = \left\{ z \in [0,1]^K : \sum_{i=1}^{K} z_i = 1 \right\}.$$

Let $\alpha = (\alpha_1, \ldots, \alpha_K) \in \mathbb{R}_+^K$ and denote $|\alpha| = \sum_{i=1}^K \alpha_i$. The random vector $\mathbf{p} = (p_1, \ldots, p_K) \in \Delta_K$ is said to have Dirichlet distribution with parameters $(\alpha_1, \ldots, \alpha_K)$, denoted

$$\mathbf{p} \sim Dir(\alpha)$$

if it has density with respect to the Lebesgue measure on Δ_K given by

$$\pi(\mathbf{p}) = \frac{\Gamma(|\alpha|)}{\prod_{i=1}^K \Gamma(\alpha_i)} p_1^{\alpha_1 - 1} \cdots p_K^{\alpha_K - 1} \mathbb{1}(\mathbf{p} \in \Delta_K)$$

where $\Gamma(b), b > 0$ is the Gamma function such that

$$\Gamma(b) = \int_0^\infty x^{b-1} e^{-x} dx, \qquad \Gamma(b) = (b-1)\Gamma(b-1)$$

and for $n \in \mathbb{N}$, $\Gamma(n) = (n-1)!$.

Subcases

- if any $\alpha_i = 0$, set the corresponding $p_i = 0$ and consider the density on the lower dimensional set
- if $\alpha_i = c$ for all i, the density is symmetric
- if $K = 2$, it reduces to

  $$\mathbf{p} \sim \frac{\Gamma(\alpha_1 + \alpha_2)}{\Gamma(\alpha_1)\Gamma(\alpha_2)} p_1^{\alpha_1 - 1} p_2^{\alpha_2 - 1} \mathbb{1}(\mathbf{p} \in \Delta_2)$$

  hence $p_1 \sim$ Beta(α_1, α_2) and $p_2 = 1 - p_1$.

Prior Moments

We have the following:

$$\begin{aligned}
\mathbb{E}(p_j) &= \int_{\Delta_K} p_j \frac{\Gamma(|\alpha|)}{\prod_{i=1}^K \Gamma(\alpha_i)} p_1^{\alpha_1 - 1} \cdots p_K^{\alpha_K - 1} d\mathbf{p} \\
&= \frac{\Gamma(|\alpha|)}{\prod_{i=1}^K \Gamma(\alpha_i)} \int_{\Delta_K} p_1^{\alpha_1 - 1} \cdots p_j^{\alpha_j + 1 - 1} \cdots p_K^{\alpha_K - 1} d\mathbf{p} \\
&= \frac{\Gamma(|\alpha|)}{\prod_{i=1}^K \Gamma(\alpha_i)} \frac{\Gamma(\alpha_j + 1) \prod_{i \neq j} \Gamma(\alpha_i)}{\Gamma(|\alpha| + 1)} = \frac{\alpha_j}{|\alpha|}
\end{aligned}$$

Furthermore

$$\mathbb{V}ar(p_j) = \frac{\alpha_j(|\alpha| - \alpha_j)}{|\alpha|^2(|\alpha|+1)}, \qquad \mathbb{C}ov(p_i, p_j) = \frac{-\alpha_i\alpha_j}{|\alpha|^2(|\alpha|+1)}$$

where the negative covariance has the interpretation that if p_i increases, given the constraint $\sum_{i=1}^K p_i = 1$, the other p_j on average will decrease.

Construction from Normalization

Since the Dirichlet distribution is the extension of the Beta distribution it has a construction related to the property of the Beta distribution written as the ratio of two Gamma distributions. In particular denoting by $Y_j \overset{ind}{\sim} Ga(\alpha_j, 1)$, $\alpha_j > 0$ for $j = 1, \dots, K$ and defining

$$p_j = \frac{Y_j}{\sum_{i=1}^K Y_i} \qquad \forall j$$

then $\mathbf{p}$ is independent of $\sum_{i=1}^K Y_i$ and $\mathbf{p} \sim Dir(\alpha)$.

Marginals

The marginal distributions are lower dimensional Dirichlet distributions. If $\mathbf{p} \sim Dir(\alpha)$, then for any partition $I_1, \dots, I_M$ of $\{1, \dots, K\}$ we have

$$\left(\sum_{i\in I_1} p_i, \dots, \sum_{i\in I_M} p_i\right) \sim Dir(\beta_1, \dots, \beta_M), \qquad \beta_j = \sum_{i\in I_j} \alpha_i$$

In particular, if $I_1 = i$ and $I_2 = \{1, \dots, K\}\backslash\{i\}$, the previous implies

$$p_i \sim Beta\left(\alpha_i, \sum_{j\neq i} \alpha_i\right).$$

Conjugacy

Let $\mathbf{p} \sim Dir(\alpha)$ and $(X_n)_{n\geq 1}$ be categorical observations, such that

$$\mathbb{P}(X_i = j|\mathbf{p}) = p_j.$$

If n_j observations in $X^{(n)}$ are equal to j, for $j = 1, \dots, K$, Bayes Theorem yields

$$\begin{aligned}\pi(\mathbf{p}|x^{(n)}) &\propto \pi(\mathbf{p}) \prod_{j=1}^K \underbrace{p_j^{\alpha_1-1}\cdots p_K^{\alpha_K-1}}_{\text{from prior}} \underbrace{p_1^{n_1}\cdots p_K^{n_K}}_{\text{from likelihood}} \\ &= p_1^{\alpha_1+n_1-1}\cdots p_K^{\alpha_K+n_K-1}\end{aligned}$$

which is again the kernel of a Dirichlet, so

$$[\mathbf{p}|X^{(n)}] \sim Dir\left(\alpha + \sum_{i=1}^n \delta_{X_i}\right).$$

The posterior adds unit masses to the parameters of the types observed in the data.

Predictive Structure

The predictive structure is given by the so called *Pólya urn scheme with K colours* which is a sampling scheme with reinforcement:

- an urn initially contains $\alpha_j > 0$ balls of colour j, for $j = 1, \dots K$. The first draw has law
$$\mathbb{P}(X_1 = j) = \frac{\alpha_j}{|\alpha|}$$
- at each draw, we return the ball to the urn toghether with another of the same colour. The law of the second draw thus is
$$\mathbb{P}(X_2 = j|X_1) = \frac{\alpha_j + \delta_{X_1}(\{j\})}{|\alpha| + 1}$$
where the probability of drawing colour X_1 has now increased.
- then
$$\mathbb{P}(X_3 = j|X_1, X_2) = \frac{\alpha_j + \delta_{X_1}(\{j\}) + \delta_{X_2}(\{j\})}{|\alpha| + 2}$$
- and so on ...

The Pólya urn scheme generates an exchangeable sequence. Indeed, denoting by α_{x_i} to indicate α_j when $x_i = j$, we have

$$\mathbb{P}(X_1 = i, X_2 = j) = \frac{\alpha_i}{|\alpha|} \cdot \frac{\alpha_j}{|\alpha| + 1} = \frac{\alpha_j}{|\alpha|} \cdot \frac{\alpha_i}{|\alpha| + 1} = \mathbb{P}(X_1 = j, X_2 = i.)$$

2.2.2. Dirichlet Process

A RPM P defined on $(\mathfrak{X}, \mathscr{X})$ is said to possess a Dirichlet process distribution, denoted with DP(α), with base measure α if for every finite measurable partition $A_1, \dots, A_k$ of $\mathfrak{X}$,

$$(P(A_1), \dots, P(A_k)) \sim Dir(k; \alpha(A_1), \dots, \alpha(A_k)). \tag{2.4}$$

In this definition, $Dir(k; \alpha(A_1), \dots, \alpha(A_k))$ and α denoted respectively the Dirichlet finite dimensional distribution with dimensionality equal to k and a given non null finite measure on $(\mathfrak{X}, \mathscr{X})$.

We write $|\alpha| = \alpha(\mathfrak{X})$ for its total mass and $\overline{\alpha} = \alpha/|\alpha|$ for the probability measure (denoted also with $P_0 \in \mathfrak{M}(\mathfrak{X})$) obtained by normalizing α, respectively, and use the notations $P \sim$ DP(α) and $P \sim$ DP($|\alpha|, \overline{\alpha}$) interchangeably to say that P has a Dirichlet process distribution with base measure α.

Definition 2.4 is based on the notion of finite-dimensional distributions of a stochastic process, seen as laws of its projections. A Dirichlet process is then a stochastic process indexed by sets (instead of time) whose finite-dimensional projections have Dirichlet distribution.

The existence of the Dirichlet process is not obvious and we refer to Ghosal and van der Vaart (2017) for the proof. Rather, we state some useful properties of the process which are interconnected with the finite-dimensional Dirichlet distribution.

Prior Moments

If $P \sim \mathsf{DP}(\alpha)$, then

$$\mathbb{E}(P) = \overline{\alpha} = P_0.$$

This means that for any $A \in \mathscr{X}$, $\mathbb{E}[P(A)] = P_0(A)$, i.e. the expected value of the random probability assigned to set A is $P_0(A)$. Being the expected value of P and provides the center of the prior $\mathsf{DP}(\alpha)$ on $\mathfrak{M}(\mathfrak{X})$, P_0 is thus interpreted as prior guess or centering distribution. Furthermore, we have

$$\mathbb{V}ar[P(A)] = \frac{\overline{\alpha}(A)\overline{\alpha}(A^c)}{\alpha(\mathfrak{X})+1} = \frac{P_0(A)(1-P_0(A))}{|\alpha|+1}$$

which shows a clear similarity to the case of the finite-dimensional Dirichlet distribution. Since $|\alpha|$ controls the variability of P around the prior guess P_0 we can interpret it as a precision parameter, which reflects the degree of confidence in the prior guess. Finally, for $A, B \in \mathscr{X}$

$$\mathbb{C}ov[P(A), P(B)] = \frac{\overline{\alpha}(A \bigcap B) - \overline{\alpha}(A)\overline{\alpha}(B)}{\alpha(\mathfrak{X})+1} = \frac{P_0(A \bigcap B) - P_0(A)P_0(B)}{|\alpha|+1},$$

which, again, exhibit a close relationship to the finite-dimensional Dirichlet distribution. The intuition behind this formula is that when A, B have little to no overlap, so $P_0(A \bigcap B)$ is small, if $P(A)$ increases then $P(B)$ will tend to decrease, given the total probability mass is constrained to 1, determining a negative covariance. When the overlap is big, an increase of one will tend to increase the other as well.

Conjugacy

One of the most remarkable properties of the Dirichlet process prior is that the posterior distribution is again a Dirichlet process. Let $X_i|P \overset{iid}{\sim} P$, with $P \sim \mathsf{DP}(\alpha)$. Then

$$[P|X^{(n)}] \sim \mathsf{DP}\left(\alpha + \sum_{i=1}^{n} \delta_{X_i}\right).$$

The posterior RPM, given the data, is still a Dirichlet process, with updated parameter. The latter is obtained by adding unit masses to the measure α in correspondence of the observed locations.

Posterior Mean

The prior guess at the unknown distribution is the normalised parameter measure, or centering distribution as seen above. After observing $X^{(n)}$, the updated guess is given by the posterior mean (i.e. the expectation of the posterior distribution) and this is given by the normalised parameter measure after the update.

Define $:= \alpha + \sum_{i=1}^{n} X_i$, whose total mass is

$$\alpha_n(\mathfrak{X}) = \alpha(\mathfrak{X}) + \sum_{i=1}^{n} \delta_{X_i}(\mathfrak{X}) = |\alpha| + n.$$

Then the posterior mean of P is

$$\begin{aligned}\mathbb{E}[P|X^{(n)}] = \overline{\alpha_n} = \frac{\alpha_n}{\alpha_n(\mathfrak{X})} &= \frac{|\alpha|P_0 + \sum_{i=1}^{n} \delta_{X_i}}{|\alpha| + n} \\ &= \frac{|\alpha|}{|\alpha| + n} P_0 + \frac{n}{|\alpha| + n} P_n, \quad P_n = \frac{1}{n} \sum_{i=1}^{n} \delta_{X_i}\end{aligned}$$

yielding a convex combination (or mixture) of the prior guess P_0 and the empirical measure P_n.

Discreteness

A realization from the Dirichlet process is discrete with probability one, also when the base measure is absolutely continuous. We refer to Ghosal and van der Vaart (2017) for the proof and for a detailed discussion about possible drawbacks of this property.

Predictive Distribution

The joint distribution of a sequence $\mathbf{X} = X_1, X_2, \ldots$ generated from a Dirichlet process, has a complicated structure, but can be conveniently described by its sequence of predictive distributions: the laws of $X_1, (X_2|X_1), (X_3|X_1, X_2)$, etc.

Because $\mathbb{P}(X_1 \in A) = \mathbb{E}[\mathbb{P}(X_1 \in A|P)] = \mathbb{E}[P(A)] = \overline{\alpha}(A)$, the marginal distribution of X_1 is $\overline{\alpha}$.

Since $X_2|(P, X_1) \sim P$ and $P|X_1 \sim DP(\alpha + \delta_{X_1})$, we can apply the same reasoning again, but now conditionally given X_1, to see that $X_2|X_1$ follows the normalization of $\alpha + \delta_{X_1}$. This is a mixture of α and δ_{X_1} with weights $|\alpha|/(|\alpha| + 1)$ and $1/(|\alpha| + 1)$, respectively.

Repeating this argument, using that $P|X_1, \ldots, X_{i-1} \sim DP(\alpha + \sum_{j=1}^{i-1} \delta_{X_j})$, we find that the distribution of a further observation given the data $X^{(n)}$ - symbolized by $\mathbb{P}(X_{n+1} \in A|X^{(n)})$ - is represented by the so called *generalised Pólya urn scheme* with parameter α indroduced by Blackwell and MacQueen (1973). It has the following expression

$$
\begin{aligned}
X_1 &\sim P_0 \\
[X_2|X_1] &= \begin{cases} Z_2 \sim P_0 & \text{w.p.} \quad |\alpha|/(|\alpha|+1) \\ X_1 & \text{w.p.} \quad 1/(|\alpha|+1) \end{cases} \\
&\vdots \\
[X_{n+1}|X^{(n)}] &= \begin{cases} Z_{n+1} \sim P_0 & \text{w.p.} \quad |\alpha|/(|\alpha|+n) \\ X_i & \text{w.p.} \quad 1/(|\alpha|+n), \qquad i = 1, \ldots, n \end{cases}
\end{aligned}
\tag{2.5}
$$

So, X_{n+1} equals one of the previous observations with probability $n/(|\alpha|+n)$ and is a new draw from P_0 otherwise. Furthermore, this is an extension of the Pólya urn with K colours seen for the finite-dimensional Dirichlet distribution with an almost identical interpretation.

Let us consider balls which can carry a continuum $\mathfrak{X}$ of "colors". Suppose that, initially, the "number of balls" is $M = |\alpha|$, which may be any positive number, and the colors are distributed according to $\overline{\alpha}$. We draw a ball from the collection, observe its color X_1, and return it to the urn along with an additional ball of the same color. The total number of balls is now $M+1$, and the colors are distributed according to $(M\overline{\alpha} + \delta_{X_1})/(M+1)$. We draw a ball from this updated urn, observe its color X_2, and return it to the urn along with an additional ball of the same color. The probability of picking up the ball that was added after the first draw is $1/(M+1)$, in which case $X_2 = X_1$; otherwise, with probability $M/(M+1)$, we make a fresh draw from the original urn. This process continues indefinitely, leading to the conditional distributions in (2.5).

The Exchangeable Partition Probability Function

The *Exchangeable Partition Probability Function* (EPPF) represents the distribution of the random partition by grouping an exchangeable sequence by distinct types.

A random partition P_n is said to be exchangeable if its distribution is invariant under permutations of $[n] := \{1, \ldots, n\}$; $\mathsf{P}_n := (\mathsf{P}_n)_{n\geq 1}$ is an exchangeable partition of $\mathbb{N}$ if P_n is exchangeable for every n. Clearly, $\mathsf{P}_{\mathbf{X}}$ is exchangeable if $\mathbf{X}$ is exchangeable (Bloem-Reddy, 2018). So, a partition P_n of $\{1, \ldots, n\}$ is said to be exchangeable is its law is a symmetric function of the group sizes only. For a partition $I_1, \ldots, I_k$ of $\{1, \ldots, n\}$ this reads

$$
\mathbb{P}(\mathsf{P}_n = \{I_1, \ldots, I_k\}) = p(n_1, \ldots, n_k), \qquad n_j = \mathsf{card}(I_j)
$$

In the Dirichlet process case, since the sequence of observations is exchangeable we expect the resulting partition to be exchangeable as well. In fact its EPPF can be derived from the predictive structure with a direct computation obtaining

$$
p(n_1, \ldots, n_k) = \frac{|\alpha|^k}{|\alpha|_{(n)}} \prod_{i=1}^{k} \Gamma(n_i) = \frac{|\alpha|^k}{|\alpha|_{(n)}} \prod_{i=1}^{k} (n_i - 1)! \tag{2.6}
$$

where

$$|\alpha|_{(n)} := \frac{\Gamma(|\alpha| + n)}{\Gamma(|\alpha|)} = |\alpha|(|\alpha| + 1) \cdots (|\alpha| + n - 1)$$

is the so called Pochhammer symbol.

2.2.3. Construction of the Dirichlet Process and the Chinese Restaurant Process

Dirichlet process can be constructed in several ways, each of which can lead to particular properties and offer additional characterizations. We do not cover all of them but rather we focus on the stick-breaking representation. We dicuss also the recovering of a sample from a DP through the generalised Pólya urn scheme. For a more in-depth reading, please refer to Ghosal and van der Vaart (2017).

Stick Breaking Representation

Following MacEachern (2016), Sethuraman (1994) provided the *stick breaking* construction of a RPM P with Dirichlet process distribution. Under his construction, the distribution P is built from two independent sequences of draws. One sequence gives the locations of atoms of mass in the discrete mixture. The second sequence determines the amount of mass associated with the atoms. Specifically, the sequences of draws are

$$\begin{aligned} \theta_i &\overset{ind}{\sim} P_0, && i = 1, 2, \ldots, \\ V_i &\overset{ind}{\sim} Beta(1, M), && i = 1, 2, \ldots. \end{aligned} \tag{2.7}$$

The V_i are used to construct a set of point masses which sum to 1 by the rule $W_i = V_i \prod_{j<i}(1 - V_j)$. The resulting countable, discrete distribution is constructed as

$$P := \sum_{i=1}^{\infty} W_i \delta_{\theta_i} \quad \text{where} \quad P \sim DP(\alpha).$$

This construction directly targets P as a countable discrete distribution, much as Blackwell and MacQueen's construction via an urn scheme as seen above. In order to see formally why this construction leads actually to a Dirichlet process please refer to Ghosal and van der Vaart (2017).

Getting a Sample from a DP through Generalized Pólya Urn Scheme

The stick breaking representation allows us to construct a RPM with Dirichlet process distribution while the generalized Pólya urn scheme representation allows to get a sample $X_1, X_2, \ldots$ generated from a Dirichlet process.

A Pólya sequence with parameter α is a sequence of random variables $X_1, X_2, \ldots$ whose joint distribution satisfies

$$X_1 \sim \overline{\alpha}. \qquad X_{n+1}|X_1, \ldots, X_n \sim \frac{\alpha + \sum_{i=1}^{n} \delta_{X_i}}{|\alpha| + 1}, \qquad n \geq 1.$$

In the previous section, a random sample from a distribution generated from a Dirichlet process with base measure α is shown to be a Pólya sequence. Since such a sequence can be constructed without reference to the Dirichlet process prior, here we want to reverse the construction and obtain the Dirichlet process from the Pólya urn sequence invoking de Finetti's theorem.

As shown in section 2.4.1, a Pólya urn scheme $X_1, X_2, \ldots$ is exchangeable therefore, by de Finetti's theorem, there exists a random probability measure P such that $X_i|P \overset{iid}{\sim} P, i = 1, 2, \ldots$. Namely, the law of P is the $DP(\alpha)-$process. In fact, given the existence of the Dirichlet process, we can refer to its *predictive distribution* to see that the Dirichlet process generates a Pólya sequence. Because the random measure given by de Finetti's theorem is unique, the result is proven. For other proofs and further details see Ghosal and van der Vaart (2017).

The sampling scheme induced by the Pólya urn scheme on the space of partitions is known through the metaphor called *Chinese Restaurant Process* (CRP). The interpretation is the following:

- the first customer arrives at the restaurant and sits at table 1 (with probability one)
- the second customer arrives at the restaurant and
  1. w.p. $1/(|\alpha| + 1)$ sits at the same table as customer 1
  2. w.p. $|\alpha|/(|\alpha| + 1)$ sits at a new table

  $\vdots$
- the $(n+1)-$th customer arrives at the restaurant and
  1. w.p. $n_j/(|\alpha| + n)$ sits at table j, where n_j represents the current number of table occupants
  2. w.p. $|\alpha|/(|\alpha| + n)$ sits at a new table.

The seating scheme determines a random partition of the integers into groups or clusters. For this reasons the number T of distinct values in $X^{(n)}$ or distinct tables in the induced partition of $\{1, \ldots, n\}$ is sometimes referred to as the number of clusters. Notice that, from now on, letters K and T are used to denote random variables as opposed to particular values (k and t).

Properties of T

If $\alpha = P_0$ with P_0 is non atomic, we have

$$\mathbb{E}[T] \approx |\alpha| \cdot log(n)$$

It can be shown in fact that

$$\frac{T}{log(n)} \overset{a.s.}{\to} |\alpha|$$

so T grows as $log(n)$. Although new distinct values are increasingly rare, their number diverges to infinity. For fixed n,

- a smaller $|\alpha|$ induces fewer big clusters
- a larger $|\alpha|$ induces a larger number of small clusters
- $\mathbb{E}[T] \approx \mathbb{V}ar[T]$
- $(T - \mathbb{E}(T))/\sqrt{\mathbb{V}ar(T)} \xrightarrow{d} N(0,1)$

Furthermore, denoting by $s = (s_1, s_2, \ldots, s_t)$ the sizes of the $T = t$ distinct tables in $X^{(n)}$ we have:

$$p_{DP(s)} \propto \prod_{j=1}^{t} s_j^{-1} \tag{2.8}$$

From (2.8) it can be seen that CRP assigns large probabilities to tables with relatively smaller size. For further details please refer to Ghosal and van der Vaart (2017); Korwar and Hollander (1973); Miller and Harrison (2018).

2.3. Gibbs -Type Priors

The Dirichlet process introduced before belongs to the family of discrete nonparametric priors which are basic building blocks for hierchical mixture models that are typically used for density estimation, clustering but also in more complex dependent structures. In particular, the Dirichlet process belongs to a large class of priors given by the *Gibbs-Type priors*.

2.3.1. Species Sampling Models

A Gibbs-type RPM P belongs to the more general subclass of discrete random probability measures called *species sampling models*. Any random probability measure associated to a discrete prior can be represented as

$$P = \sum_{j=1}^{\infty} P_j \delta_{Z_j} \tag{2.9}$$

where δ_c stands for the unit point mass concentrated at c, $(P_j)_{j\geq 1}$ is a sequence of non-negative random variables such that $\sum_{j\geq 1} p_j = 1$, almost surely, and $(Z_j)_{j\geq 1}$ is a sequence of $\mathbb{X}$-valued random variables. Species sampling models assume $(P_j)_{j\geq 1}$ and $(Z_j)_{j\geq 1}$ to be independent and also the Z_j's to be i.i.d. from a diffuse probability measure $P^\star$ on $\mathbb{X}$ (i.e. $\mathbb{P}(Z_i \neq Z_j) = 1$ for any $i \neq j$).

The terminology *species sampling model* is due to the fact that (2.9) can be seen as a tool for describing the structure of a population made of different types or species with certain proportions, which are modeled through (2.9) as random proportions P_j. Therefore, one can equivalently use the Z_i's or the positive integers $\{1, 2, \ldots\}$ to label different species or types that can be sampled.

2.3.2. Gibbs -Type RPMs

Typically Gibbs-type RPMs are defined through their EPPF. As seen in Section 2.2.2 for the Dirichlet Process, an EPPF

$$p_{n,t}(n) := p(n_1, \dots, n_t)$$

is a symmetric function that defines the law of the partition of $\{1, \dots, n\}$ in t groups of size $n_1, \dots, n_t$ respectively. The function $p_{n,t}(n)$ must satisfy the following consistency under marginalization

$$p_{n,t}(n) = p_{n+1,t+1}(n, 1) + \sum_{j=1}^{t} p_{n+1,t}(n + e_j), \;\; n + e_j = (n_1, \dots, n_j + 1, \dots, n_t)$$

where $p_{n+1,t+1}(n, 1)$ and $p_{n+1,t}(n + e_j)$ are the laws of the partition induced by $n + 1$ items, where the $(n + 1)$-st, respectively, is a singleton or is in group j. The previous expression gives the law of the partition on n items, obtained by marginalising the $(n + 1)$-st item out of the law of $n + 1$ items. Accordingly, a RPM is said to be of Gibbs-type if it has EPPF of the form

$$p_{n,k}(n) = V_n(t) \prod_{j=1}^{t} (1 - \sigma)_{(n_j - 1)}, \qquad \sigma < 1, \tag{2.10}$$

where $\sum_j n_j = n$ and the set of non-negative weights $\{V_n(t) : n \geq 1, 1 \leq t \leq n\}$ satisfies the forward recursive equation

$$V_n(t) = (n - \sigma t) V_{n+1}(t) + V_{n+1}(t + 1) \tag{2.11}$$

for any $t = 1, \dots, n$ and $n \geq 1$, with $V_1(1) = 1$.

Notice that, letting $\sigma = 0$ and $V_n(t) = \frac{|\alpha|^t}{|\alpha|_{(n)}}$ we obtain the EPPF of the Dirichlet process (2.6). For a more in-depth reading please refer to De Blasi et al. (2013a,b).

Given (2.10), the probability of obtaining a new distinct observation conditional on a sample $X_1, \dots, X_n$ such that $T = t$ is

$$\mathbb{P}(X_{n+1} = \text{“new”} \,|\, X_1, \dots, X_n) = \frac{V_{n+1}(t + 1)}{V_n(t)} = f(n, t, \Theta).$$

Therefore, (2.10) implies that the induced predictive distributions are

$$\mathbb{P}(X_{n+1} \in \cdot \,|\, X_1, \dots, X_n) = \frac{V_{n+1}(t + 1)}{V_n(t)} P^\star + \frac{V_{n+1}(t)}{V_n(t)} \sum_{i=1}^{t} (n_i - \sigma) \delta_{X_i^\star}, \tag{2.12}$$

where $X_1^\star, \dots, X_t^\star$ denotes the $t \leq n$ distinct observations and $n_1, \dots, n_t$ their frequencies such that $\sum_{i=1}^{t} n_i = n$.

The predictive distribution is therefore a linear convex combination of the prior guess $P^\star$ at the shape of P and of the weighted empirical distribution $\widehat{P}_n = (n - t\sigma)^{-1} \sum_{i=1}^{t} (n_i - \sigma)\delta_{X_i^\star}$. This is a generalization of the Pólya urn scheme of which the Dirichlet process is a special case. It is worth mentioning also that σ plays a role in weighting the empirical measure.

In our work we focused on Gibbs-type priors characterized by (2.10) with $\sigma < 0$. In such a situation we obtain Gibbs-type priors which are finite dimensional models with *random dimensionality*. In practice we have a RPM of the type

$$\mathsf{P} := \sum_{j=1}^{K} P_j \delta_{Z_j}, \quad K \sim p_K(\cdot). \tag{2.13}$$

In order to define (2.13) we need to specify a distribution for the weights (or frequencies) P_j, $j = 1, \ldots, K$. In literature we can find two different ways of specification.

1. The first one consists in assigning a *symmetric Dirichlet distribution*:

$$(P_1, \ldots, P_K) \sim Dir(\gamma, \ldots, \gamma), \tag{2.14}$$

which leads to:

$$\mathbb{E}(P_1) = \mathbb{E}(P_2) = \ldots = \mathbb{E}(P_K) = \frac{1}{K}.$$

Notice that we can obtain (2.14) also through the *stick breaking construction*. Indeed if we generate:

$$\begin{aligned} V_1 &\sim Beta(\gamma, (K-1)\gamma) \\ V_2 &\sim Beta(\gamma, (K-2)\gamma) \\ &\vdots \\ V_K &\sim Beta(\gamma, 0) = 1 \end{aligned}$$

and we apply the stick breaking construction we get a vector of weights $(P_1, \ldots, P_K)$ distributed as (2.14).

2. The second way involves to assign a distribution to the P_j's via the *size-biased ordering* of the P_j's, denoted by $(\widetilde{P}_1, \ldots \widetilde{P}_K)$, in order to obtain:

$$\mathbb{E}(\widetilde{P}_1) > \mathbb{E}(\widetilde{P}_2) > \ldots > \mathbb{E}(\widetilde{P}_K). \tag{2.15}$$

In this case, the only way to give a distribution to $(\widetilde{P}_1, \ldots \widetilde{P}_K)$ is via a stick breaking construction. In particular, generating:

$$\begin{aligned} \widetilde{V}_1 &\sim Beta(1+\gamma, (K-1)\gamma) \\ \widetilde{V}_2 &\sim Beta(1+\gamma, (K-2)\gamma) \\ &\vdots \\ \widetilde{V}_K &\sim Beta(1+\gamma, 0) = 1 \end{aligned}$$

and applying the stick breaking construction we get $(\widetilde{P}_1, \ldots \widetilde{P}_K)$ satisfying (2.15).

Summarizing, the Gibbs-type priors with $\sigma < 0$ are species sampling models with a *random finite* number of atoms with weights obtainable through the stick breaking construction. For a given K they can be described in an equivalent way depending on the ordering of the weights. If they are assumed identical distributed then we can use formulation 1 otherwise we should use formulation 2. However, regardless of the way they are described we have the following relationship:

$$\sum_{j=1}^{K} P_j \delta_{Z_j} \stackrel{d}{=} \sum_{j=1}^{K} \widetilde{P}_j \delta_{Z_j}, \tag{2.16}$$

where in the r.h.s. of (2.16) we did not permute the Z_j's since we are working with species sampling model and hence $Z_j \perp \widetilde{P}_j$.

2.4. BNP Mixture Models

RPMs are often used as *mixing measure* in mixture models. Following De Blasi et al. (2013a), let us consider for example the univariate density estimation case and let $f(\cdot|\cdot)$ denote a kernel defined on $\mathbb{R} \times \mathbb{X}$ taking values in $\mathbb{R}^+$ such that $\int_{\mathbb{R}} f(y|x)dy = 1$, for any x in $\mathbb{X}$. Hence, $f(\cdot|x)$ defines a density function on $\mathbb{R}$, for any x. The observations are then from a sequence $(Y_n)_{n\geq 1}$ of real–valued random variables such that

$$\begin{aligned} Y_i|X_i &\stackrel{ind}{\sim} f(\cdot|X_i) && i = 1, \ldots, n \\ X_i|P &\stackrel{iid}{\sim} P && i = 1, \ldots, n \\ P &\sim \Pi. \end{aligned} \tag{2.17}$$

Model (2.17) can be written also as the mixture obtained integrating the latent variable out:

$$f_P(y) = \int_{\mathbb{X}} f(y|x)P(dx) \tag{2.18}$$

Interestingly, apart from density estimation, model (2.17) serves also clustering purposes. In fact, here $(X_n)_{n\geq 1}$ is a sequence of latent exchangeable random elements and the unobserved number K of distinct values among $X_1, \ldots, X_n$ is the number of clusters into which the observations $Y_1, \ldots, Y_n$ can be grouped. Actually, modeling data according to a discrete prior Π implies that a sample $(X_1, \ldots, X_n)$ has ties with positive probability. So, recalling the notation introduced before, we denote by $X_1^\star, \ldots, X_t^\star$ the $t \leq n$ distinct observations and $n_1, \ldots, n_t$ their frequencies such that $\sum_{i=1}^{t} n_i = n$. As pointed out by De Blasi et al. (2013b), in choosing a specific predictive structure the key quantity to consider is the probability of obtaining a new distinct observation

$$\mathbb{P}(X_{n+1} = \text{“new”}\,|X_1, \ldots, X_n). \tag{2.19}$$

Let us denote by Θ a finite-dimensional parameter possibly entering the specification of P in (2.17). In general, one has $\mathbb{P}(X_{n+1} = \text{“new”}\,|X_1, \ldots, X_n) = f(n, k, n_1, \ldots, n_t, \Theta)$,

which means that the probability of obtaining a new observation depends on the sample size n, the number of distinct values t, their frequencies $(n_1, \ldots, n_t)$ and the parameter Θ. We will see that, depending on the particular choice for Π we can modify the dependencies of (2.19).

2.4.1. Dirichlet Process Mixture Models

If on (2.17) we associate a Dirichlet process with base measure $\alpha = |\alpha| \cdot \overline{\alpha}$, we obtain a Dirichlet process mixture model defined as

$$\begin{aligned} Y_i|X_i &\overset{ind}{\sim} f(\cdot|X_i) && i = 1, \ldots, n \\ X_i|P &\overset{iid}{\sim} P && i = 1, \ldots, n \\ P &\sim \mathsf{DP}(\alpha). \end{aligned} \tag{2.20}$$

Alternatively, we can integrate the latent variables out and write the model as the mixture

$$f_P(y) = \int_{\mathbb{X}} f(y|x)P(dx), \quad P \sim \mathsf{DP}(\alpha) \tag{2.21}$$

Under this model $\mathbb{P}(X_{n+1} = \text{“new”}|X_1, \ldots, X_n) = f(n, \Theta)$ meaning that the probability of obtaining a new distint observation does not depend on $(n_1, \ldots, n_t)$. Actually, it can be proved that using a Dirichlet process in (2.17) is the only way to make (2.19) not dependent on the relative frequencies of each distinct observation.

It should be noted that, despite it has be proved to perform well in several applied contexts, it seems too restrictive to let the probability of generating new values depend only on n and θ. Therefore, it would be reasonable to make (2.19) dependent explicitly also on the number of distinct observed values, since it summarizes the heterogeneity in the sample, leading to $\mathbb{P}(X_{n+1} = \text{“new”}|X_1, \ldots, X_n) = f(n, t, \Theta)$. It has been shown that this can be achieved if and only if P is of *Gibbs-type*. One might also want to make (2.19) dependent on even the observed frequencies $n_1, \ldots, n_t$ using all the information conveyed by the data. In this situation, however, some practical and conceptual problems arise (De Blasi et al., 2013a) making thus the use of a Gibbs-type prior P the best compromise between flexibility and tractability.

2.4.2. Mixture of Finite Mixtures Models

If we associate a Gybbs-type RPM with $\sigma < 0$ to the mixing measure Π in (2.18) we obtain a *finite dimensional mixture model* meaning that, given K (K is still random), the mixing measure Π has finite dimension. Leading to:

$$f_P(y) = \int_{\mathbb{X}} f(y|x)P(dx), \quad \text{where} \quad P(dx)|K = \sum_{j=1}^{K} \pi_j \delta_{Z_j}(dx), \tag{2.22}$$

where $Z_j \sim P^\star$, $K \sim p_K$ and $(\pi_1, \ldots, \pi_k)|K \sim Dir(\gamma, \ldots, \gamma)$.

Now, P is a Gibbs-type RPM written as $P = \sum_{j=1}^{K} \pi_j \delta_{Z_j}$. If we introduce an auxiliary integer-valued sequence, $(\phi_n)_{n\geq 1}$, such that $\mathbb{P}(\phi_n = j|\pi) = \pi_j$ for any n and j, model (2.17) corresponds to assume that $X_i = Z_{\phi_i}$. Hence the X_n's can be interpreted as the *observed species labels* since, due to the diffuse nature of $P^\star$, any two data points X_i and X_j for $i \neq j$ differ if and only if ϕ_i and ϕ_j do. So, we can express model (2.22) also as:

$$\begin{aligned} K &\sim p_K, && \text{where } p_K \text{ is a p.m.f. on } \{1, 2, \ldots\} \\ (\pi_1, \ldots, \pi_k) &\sim Dir(\gamma, \ldots, \gamma), && \text{given } K = k \\ \phi_1, \ldots, \phi_n &\overset{iid}{\sim} \pi, && \text{given } \pi \\ Z_1, \ldots, Z_k &\overset{iid}{\sim} P^\star, && \text{given } K = k \\ Y_i &\overset{ind}{\sim} f(\cdot|Z_{\phi_i}) && i = 1, \ldots, n. \end{aligned} \tag{2.23}$$

Model (2.23) is reffered by Miller and Harrison (2018) as a *Mixture of Finite Mixtures* (MFM) model.

Pólya Urn Scheme

Pitman (1996) considered a general class of urn schemes, or restaurant processes, corresponding to exchangeable partition probability functions (EPPFs). The following scheme for model (2.23) falls into this general class.

1. Initialize with a single table consisting of element 1 alone: $C_i = \{\{1\}\}$,
2. For $n = 2, 3, \ldots,$ place element n in
   (a) an existing table $c \in C_{n-1}$ with probability $\propto |c| + \gamma$
   (b) a new table with probability $\propto \frac{V_n(t+1)}{V_n(t)}\gamma$ where $t = |C_{n-1}|$.

$V_n(t)$ here is a coefficient of partition distribution that need to be computed defined as:

$$V_n(t) = \sum_{k=1}^{\infty} \frac{k_{(t)}}{(\gamma k)^{(n)}} p(k), \tag{2.24}$$

where $k_{(t)} = k(k-1)\ldots(k-t+1)$, and $(\gamma k)^{(n)} = \gamma k(\gamma k+1)\ldots(\gamma k+n-1)$. (By convention, $x^{(0)} = 1$ and $x_{(0)} = 1$). The numbers $V_n(t)$ (equation (2.23)) satisfy the recursion:

$$V_{n+1}(t+1) = V_n(t)/\gamma - (n/\gamma + t)V_{n+1}(t) \tag{2.25}$$

for any $0 \leq t \leq n$ and $\gamma > 0$. Note that (2.25) is a special case of (2.11).

Also in this case, as for the CRP, the seating scheme described above determines a random partition of the integers $\{1, \ldots, n\}$ into groups or clusters. Let denote by $X^{(n)}$ the sample of dimension n we got though Pólya urn scheme associated to model

(2.23). Denoting by t and $s = (s_1, \ldots, s_t)$ respectively the number of distinct values (communities) in $X^{(n)}$ and the communities sizes we have:

$$p_{MFM}(s) \propto \prod_{j=1}^{t} s_j^{\gamma-1}. \tag{2.26}$$

Miller and Harrison (2018) proved the relationships between K and T in model (2.23) which is given by:

$$p(T = t|K = k) = \frac{k_{(t)}}{(\gamma k)^{(n)}} \sum_{C:|C|=t} \prod_{c \in C} \gamma^{(|c|)}, \tag{2.27}$$

$$p(K = k|T = t) = \frac{1}{V_n(t)} \frac{k_{(t)}}{(\gamma k)^{(n)}} p_K(k), \tag{2.28}$$

where in equation (2.27), the sum is over partitions C of $\{1, \ldots, n\}$ such that $|C| = t$. Note that $p(t|k)$ and $p(k|t)$ depend on n. Furthermore, the conditional independence relations hold:

$$C \perp K \mid T, \tag{2.29}$$

$$Y_{1:n} \perp K \mid T. \tag{2.30}$$

The formula for $p(k|t)$ is required for doing inference about the number of components K based on posterior samples of C. Indeed:

$$\begin{aligned} p(k|y_{1:n}) &= \sum_{t=1}^{\infty} p(k|t, y_{1:n}) p(t|y_{1:n}) \\ &= \sum_{t=1}^{n} p(k|t) p(t|y_{1:n}), \end{aligned} \tag{2.31}$$

by relation (2.30) and the fact that t cannot exceed n. Using this and the formula for $p(k|t)$ given by equation (2.31), it is simple to transform our estimate of the posterior on t into an estimate of the posterior on k.

Finally, Miller and Harrison (2018) showed that the number of clusters $T = |C|$ behaves very similarly to the number of components K when n is large under the posterior. In particular they proved the following

Theorem 2.2. *Let $y_1, y_2, \ldots \sim f(\cdot)$ of model (2.23) and $k \in \{1, 2, \ldots\}$.*
If $p_K(1), \ldots, p_K(k) > 0$ then:

$$|p(T = k|y_{1:n}) - p(K = k|y_{1:n})| \to 0$$

as $n \to \infty$.

3. Community detection in Stochastic Block Models

In this chapter we discuss stochastic block models and their use in community detection, depending on whether the number of communities is known or not.

3.1. Community Detection: an Overview

3.1.1. Motivation

Networks, or graphs, consist of *vertices* and *edges*. An edge typically connects a pair of vertices. Networks occur in an huge variety of context. Facebook, for example, is a large social network, where more than one billion people are connected via virtual acquaintanceships. Other examples come from biology, physics, economics, engineering, computer science, ecology, marketing, social and political science, etc (Fortunato and Hric, 2016; Newman, 2018).

Most networks of interest display *community structure*, i.e., their vertices are organized into groups, called *communities, clusters*, or *modules*. Communities could represent proteins with similar function in protein-protein interaction networks, groups of friends in social networks, websites on similar topics on the Web graph, and so on.

Identifying communities may offer insight on how the network is organised. It allows us to focus on regions having some degree of autonomy within the graph. It helps to classify the vertices, based on their role with respect to the communities they belong to.

Following Fortunato and Hric (2016) and Rosvall et al. (2019), community detection is an ill-defined problem though. There is no universal definition of the object that one should be looking for. Consequently, there are no clear-cut guidelines on how to assess the performance of different algorithms and how to compare them with each other. On the one hand, such ambiguity leaves a lot of freedom to propose diverse approaches to the problem, which often depend on the specific research question and the particular system at study. On the other hand, it has introduced a lot of noise into the field, slowing down the progress.

3.1.2. What Are Communities?

A precise definition of what constitutes a community in networks has remained elusive (Rosvall et al., 2019). Traditional definitions of community rely on counting edges (internal, external) in various way, see Fortunato and Hric (2016) for a brief rewiev. However, what one should be really focusing on is the *probability* that vertices share edges with a subgraph. The existence of communities, indeed, implies that vertices interact more strongly with the other members of their community than they do with vertices of the other communities. Consequently, there is a preferential linking pattern between vertices of the same group. This is the reason why edge densities end up being higher within communities than between them. We can formulate that by saying that vertices of the same community have a higher probability to form edges with their partners than with the other vertices. Nevertheless, computing the edge probability, starting from an adjacency matrix, is still an ill-defined problem unless one has a model stating how edges are formed. One can make many hypotheses on the process of edge formation. For instance, if we take social networks, we can assume that the probability that two individuals know each other is a decreasing function of their geographical distance, on average (Liben-Nowell et al., 2005). Each set of assumptions thus defines a model and the most famous model of networks with group structure is the *Stochastic Block Model* (SBM).

For the sake of completeness, it is worth mentioning that other definitions of community exitst. For example, alternative definitions are based on the interplay between network topology and dynamics allowing the preservation of more complex features of the network and its communities, which typically get lost when one uses network model solely based on edge probabilities, like SBMs. The drawback, however, is that calculations become more involved and lenghty (Fortunato and Hric, 2016).

Despite a precise definition of what constitutes a community in networks has remained elusive, it might be natural ask which definition of community should be used. There is not a simple answer to this question and often an answer is not even necessary since most techniques to detect communities in networks do not require a precise definition of community, despite defining clusters beforehand is a useful starting point that allows one to check the reliability of the final results. Finally, it should be said that popular techniques are based on similar ideas of communities. What makes the difference, thus, is the way clusters are sought (Fortunato and Hric, 2016).

3.1.3. Approaches to Community Detection

Various research areas view community detection from different perspectives, illustrated by the lack of a consistent terminology: *"network clustering"*, *"graph partitioning"*, *"community"*, *"random vertex colors"*, *"block"* or *"module detection"* all carry slightly different connotations. This jargon barrier creates confusion, as readers and authors have different preconceptions and intuitive notions are not made explicit and consequently, as pointed out by Rosvall et al. (2019), community detection should not

be considered as a well-defined problem, but rather as an umbrella term with many facets. These facets emerge from different goals and motivations for what it is about the network that we want to understand or achieve, and lead to different perspectives on how to formulate the problem of community detection. It is critically important to be aware of these underlying motivations when selecting and comparing community detection methods.

Following Legramanti et al. (2020), the relevance of such topic and the interdisciplinary nature of network science have motivated a collective effort by various disciplines towards the development of methods for community detection, ranging from algorithmic strategies (Newman and Girvan, 2004; Newman, 2006; Girvan and Newman, 2002; Leskovec et al., 2010) to model-based solutions (Nowicki and Snijders, 2001; Holland et al., 1983; Airoldi et al., 2008; Lee and Wilkinson, 2019).

Despite being widely used in practice, most algorithmic approaches lack uncertainty quantification and can only detect communities characterized by dense within-block connectivity and sparser connections between different blocks (Fortunato and Hric, 2016). Moreover, other disadvantages include that different initial configurations may lead to different results even under the same algorithm, and that different results give vastly different results (Lee and Wilkinson, 2019). These issues have motivated a growing interest in model–based solutions which rely on generative statistical models. This choice allows coherent uncertainty quantification, model selection and hypothesis testing, while accounting for more general connectivity patterns, where nodes in the same community are not necessarily more densely connected, but simply share the same connectivity behavior, which may even characterize core–periphery, disassortative or weak community patterns (Legramanti et al., 2020; Rosvall et al., 2019). Among the generative models for learning communities in network data, SBM is perhaps the most widely implemented and well–established formulation, owing also to its unique balance between simplicity and flexibility (Legramanti et al., 2020).

A more in-depth review of the most used methods with their relative pros and cons can be found in Fortunato and Hric (2016) and Lee and Wilkinson (2019).

3.2. Community Detection trough Modularity Optimization: the Louvain Method

We present the main idea behind one of the most used methods in community detection belonging to the algorithmic class: the Louvain algorithm, developed by Blondel et al. (2008).

3.2.1. Terminology and Notation

To introduce the terminology, we consider a graph $\mathscr{G} = (\mathscr{N}, \mathscr{E})$, where $\mathscr{N}$ is the node set of size $n := |\mathscr{N}|$, and $\mathscr{E}$ is the edge list of size $M := |\mathscr{E}|$. We call a pair

of nodes a *dyad*, and consider the existence or absence of an edge for the dyad (i, j), through the use of the $n \times n$ adjacency matrix, denoted by $\mathbf{Y}$, which is another useful representation of the graph. If $\mathscr{G}$ is undirected, then $\mathbf{Y}_{ij} = \mathbf{Y}_{ji} = 1$ if i and j have an edge between them, 0 otherwise, where $\mathbf{M}_{ab}$ represents the $(a, b) - th$ element of a generic matrix $\mathbf{M}$. By construction, under an undirected graph, $\mathbf{Y}$ is symmetric along the major diagonal. If $\mathscr{G}$ is directed, $\mathbf{Y}_{ij} = 1$ (0) represents an edge (non-edge) from i to j, and is independent of $\mathbf{Y}_{ij}$, for $\mathbf{Y}$ need not be symmetric. Hereinafter, we consider only undirected networks $\mathscr{G}$ without self-loops so that $\mathbf{Y}_{ii} = 0$ and $\mathbf{Y}_{ij} = \mathbf{Y}_{ji}$ as this is quite often assumed in earlier works (Lee and Wilkinson, 2019; Geng et al., 2019; Legramanti et al., 2020; Riolo et al., 2017; Newman and Reinert, 2016).

Let us denote by $\overline{z}_i$ the community of node i, which is an integer $\overline{z}_i = 1, \ldots, k$ where k indicates the number of communities in the network. We introduce also the *modularity* measure of a network. Modularity measures the strength of division of a network into communities and is defined as follows:

$$Q = \frac{1}{2n} \sum_{ij} \left(\mathbf{Y}_{ij} - \frac{k_i k_j}{2n} \right) \delta_{\overline{z}_i \overline{z}_j},$$

where k_i denotes the number of edges attached to node i and $\delta(\cdot)$ denotes the Kronecker delta. Basically, it measures the fraction of edges that fall within a given community minus the expected fraction if edges were distributed at random. The value of the modularity for unweighted and undirected graphs is strictly less than 1 and takes positive values if there are more edges between nodes of the same type than we would expect by random chance. It can also take negative values if there are fewer such edges than we would expect by chance. So, networks with high modularity have dense connections between the nodes within communities but sparse connections between nodes in different communities.

3.2.2. Model Formulation

This method is an heuristic algorithm for approximately maximizing modularity over divisions of a network into any number of communities. The algorithm has become a popular choice for many applications because of its speed and because it is included in a number of standard software packages for network analysis (Newman, 2018), such as in the R package `igraph`.

The Louvain algorithm is an *agglomerative* algorithm, which works by taking single nodes and joining them into groups, then joining groups with other groups, and so forth, in an effort to find the configuration with highest modularity. Initially, each node is placed in a separate group on its own.

In order to maximize Q efficiently, the Louvain Method has two phases that are repeated iteratively. Following Newman (2018), in the first phase, one goes through each node in turn and moves that node to another group chosen such that the modularity of the complete system is increased by the largest amount. If no move increases

the modularity then the node stays where it is. To make things faster, one only ever considers moving a node into a group that contains at least one of its neighbors. When all nodes have been considered and potentially moved, one repeats the process, and continues to do so until there are no more moves that increase the modularity. This ends the first round of the algorithm.

In the second phase, one carries out the same procedure again, but now instead of moving nodes, one moves whole groups. That is, one treats the groups found on the previous round as the units of the algorithm and moves them, in their entirety, from group to group in an effort to increase the modularity, stopping when no further increase is possible. And so the algorithm proceeds, through as many rounds as are necessary until one reaches a configuration where there are no moves at all that will increase the modularity. This final configuration is then taken as the community division of the network.

This algorithm works well in practice and is widely use due to its speed, which is its primary advantage over other algorithms.

3.3. Stochastic Block Models

3.3.1. Model Formulation

A stochastic blockmodel is a generative class of models for blocks, groups, or communities in networks. SBMs fall in the general class of random graph models and have a long tradition of study in the social sciences and computer science (Karrer and Newman, 2011). Typically they are used to model the phenomenon of *assortative mixing* - observed particularly in social networks - whereby nodes that share some feature or attribute in common are more likely to be connected. For example, people of the same age, income, nationality, race or educational level are typically more likely to be friends than people who differ on these things (Newman, 2018; Lee and Wilkinson, 2019). SBM is thus a model for data obtained from a network characterized by block structure, where by block structure is meant that the n nodes of the network are partitioned into $K = k$ groups or blocks - numbered from 1 to k - which might represent languages, age brackets, ethnicities or any other variables of interest. The first SBM formulation comes from Fienberg and Wasserman (1981); Holland et al. (1983) who extended the concept of blockmodelling to a stochastic version (Snijders and Nowicki, 1997).

The basic idea of the standard SBM is that the neighborhood relations of each node only depend on the probabilities given by the group memberships. Roughly speaking, the nodes are clustered in a way so that the neighbors of nodes in a group have a similar neighborhood pattern as well. The standard SBM is based on a generative model, which enables the user to generate other network instances from a given structure or allows the prediction of missing edges. Moreover, the SBM is capable to describe any kind of group structure (Funke and Becker, 2019). We first take a look at the

generative model included in the SBM and then we describe the reverse process of SBM inference from a given network.

Generative Model

To explain the generative model of the SBM, we assume all model parameters are given. For the standard SBM in its most simple formulation the model parameters are the group structure $\overline{\mathbf{z}} = (\overline{z}_1, \ldots, \overline{z}_n)^\top \in \{1, \ldots, k\}^n$, which assigns each node to a single group $\overline{z}_r$, and the edge probabilities matrix $\mathbf{Q}$. To create one realization of these parameters, for each node pairing their respective group assignment and the corresponding edge probability is retrieved. Then, the edge probability is evaluated and depending on the result edges are added to the network. For example an edge between a node i in group r and a node j in group s is created with probability

$$P(i \leftrightarrow j) = P(r \leftrightarrow s) = \mathbf{Q}_{rs}.$$

In the description of the generative process, a Bernoulli distributions is used to decide whether an edge is created or not. Some of the early works are based on these model (Holland et al., 1983; Snijders and Nowicki, 1997). Therefore, an adjacency matrix $\mathbf{Y} = (\mathbf{Y}_{ij}) \in \{0, 1\}^{n \times n}$ of a network with n nodes generated through this model (i.e. placing undirected edges at random between node pairs with dependent probability *only* on the groups nodes belong to) should have sub–diagonal entries $\mathbf{Y}_{ij}$, $i = 2, \ldots, n$, $j = 1, \ldots, i-1$ conditionally independent Bernoulli random variables with probabilities depending only on the community memberships of the involved nodes i and j. For this reason, vertices that belong to the same block in a stochastic block model are said to be *stochastically equivalent* (Wasserman and Anderson, 1987; Snijders and Nowicki, 1997).

Denoting by θ_{rs} the probability of an edge between two nodes, one of which is in group r and the other in group s, we have:

$$\mathbf{Y}_{ij} | \overline{\mathbf{z}}, \mathbf{Q}, K \sim Bernoulli(\theta_{ij}), \qquad \theta_{ij} = \mathbf{Q}_{\overline{z}_i \overline{z}_j}, \qquad 1 \leq i < j \leq n. \tag{3.1}$$

where $\mathbf{Q} = (\mathbf{Q}_{rs}) \in [0, 1]^{K \times K}$ with $\mathbf{Q}_{rs} = \mathbf{Q}_{sr}$. It is clear as, under this model, if the diagonal elements of the matrix $\mathbf{Q}$ are larger than the off-diagonal ones, then edges will be more likely inside groups than between them and we will get a network with traditional *assortative mixing*. The model can generate *disassortative mixing* too, if we make the diagonal elements smaller than the off-diagonal ones, although it is less often used this way (Newman, 2018; Geng et al., 2019).

In the formulation of Snijders and Nowicki (1997); Nowicki and Snijders (2001) we need in addition a stochastic vector $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_K)$, where π_h is the probability of a node belonging to group h. With this, the likelihood of the model is given by (see Eq (2) in Snijders and Nowicki (1997)):

$$p(\mathbf{Y}, \overline{\mathbf{z}} | \mathbf{Q}, \boldsymbol{\pi}) = \pi_1^{n_1} \cdots \pi_K^{n_K} \prod_{r \leq s} \mathbf{Q}_{rs}^{e_{rs}} (1 - \mathbf{Q}_{rs})^{n_{rs} - e_{rs}} \tag{3.2}$$

where n_r is the number of nodes in group r, i.e. $n_r = \sum_{i=1}^{n} \mathbb{1}(\overline{z}_i = r)$, and

$$n_{rs} = \begin{cases} n_r n_s & \text{if } r \neq s \\ \frac{n_r(n_r+1)}{2} & \text{if } r = s \end{cases} \tag{3.3}$$

is the number of possible edges between group r and group s. Further, we denote

$$e_{rs} = \frac{1}{1+\delta_{rs}} \sum_{i \in r;\, j \in s} \mathbf{Y}_{ij} \tag{3.4}$$

as the number of edges between group r and s and the Kronecker delta $\delta_{rs} = 1$ for $r = s$ and 0 for $r \neq s$.

With this formulation, one can directly see the general structure of SBM. The likelihood in (3.2) splits into a first part representing the likelihood of node partition and a second part corresponding to the likelihood of all edges. The last part of this equation not only takes existing edges into account, but also non existing edges.

As underlined by Newman and Reinert (2016), this definition of the model specifies its behavior in the *"forward"* direction, for the generation of random artificial networks, but the main interest is in its use in the reverse direction for inference, where we hypotize that an observed network was generated using the model and then estimate by looking at the network which parameter values must have been used in the generation. In fact, the main inference task of all SBM variants is to recover both the partition $\overline{\mathbf{z}}$ of the nodes into K groups and the matrix $\mathbf{Q}$ given the adjacency matrix $\mathbf{Y}$. Without going into too much detail in the formulation of this model, as it is beyond the primary interest of this work, following Legramanti et al. (2020) we can write the likelihood for the adjacency matrix $\mathbf{Y}$ given $\overline{\mathbf{z}}$ and $\mathbf{Q}$ as follows

$$\begin{aligned} p(\mathbf{Y}|\overline{\mathbf{z}}, \mathbf{Q}) &= \prod_{i=2}^{n} \prod_{j=1}^{i-1} \theta_{\overline{z}_i \overline{z}_j}^{\mathbf{Y}_{ij}} (1 - \theta_{\overline{z}_i \overline{z}_j})^{1-\mathbf{Y}_{ij}} \\ &= \prod_{r=1}^{K} \prod_{s=1}^{r} \theta_{rs}^{e_{rs}} (1 - \theta_{rs})^{n_{rs} - e_{rs}}, \end{aligned} \tag{3.5}$$

where e_{rs} and $n_{rs} - e_{rs}$ denote the number of edges and non-edges between communities r and s respectively.

Equation (3.5) is a stochastic block model with independent edges in which $\overline{\mathbf{z}}$ and $\mathbf{Q}$ have the role of parameters that are supposed to be derived from $\mathbf{Y}$. For the statistical application of block models, whether the blocks given by $\overline{\mathbf{z}}$ are known or not is an important distinction, since the latter situation is much more complicated, and sometimes called *a posteriori blockmodel* (Snijders and Nowicki, 1997). A possible approach to *posteriori blockmodelling* is through maximization of the likelihood function (3.3) by means numerical methods even though they are not practical unless n is rather small. An excellent alternative, practical also for larger graphs, its offered by a Bayesian approach (Snijders and Nowicki, 1997).

3.3.2. Bayesian Community Detection

In classical SBMs $K = k$, so the number of communities is assumed to be known and finite. Thus, in a Bayesian setting a common specification of the SBM can be completed by assigning independent priors to $\bar{\mathbf{z}}$ and $\mathbf{Q}$. We use $p(\bar{\mathbf{z}}, \mathbf{Q}) = p(\bar{\mathbf{z}})p(\mathbf{Q})$ to denote the joint prior on $\bar{\mathbf{z}}$ and $\mathbf{Q}$ (Geng et al., 2019). The recovery of the block structure is then based on the *posterior predictive distribution*, which is the conditional distribution of $\bar{\mathbf{z}}$ given the data $\mathbf{Y}$.

Bayesian SBM Specification

Following Legramanti et al. (2020), in classical SBMs independent $Beta(a, b)$ priors for θ_{rs} are assumed, leading thus to the following join density for $\mathbf{Q}$:

$$p(\mathbf{Q}) = \prod_{r=1}^{k} \prod_{s=1}^{r} \frac{\theta_{rs}^{a-1}(1-\theta_{rs})^{b-1}}{\mathbf{B}(a,b)} \tag{3.6}$$

where $\mathbf{B}(\cdot,\cdot)$ is the Beta function. Expression (3.6) can be used to quantify prior uncertainty in the block probabilities. However, the main goal in SBMs is to provide inference on communities, hence $\mathbf{Q}$ is usually treated as a nuisance parameter and marginalized out in (3.2) via beta-binomial conjugacy, obtaining

$$p(\mathbf{Y}|\bar{\mathbf{z}}) = \prod_{r=1}^{k} \prod_{s=1}^{r} \frac{\mathbf{B}(a + e_{rs}, b + n_{rs} - e_{rs})}{\mathbf{B}(a,b)}. \tag{3.7}$$

Equation (3.7) provides a simple likelihood common to several extensions of SBMs, which instead differ in the choice of the probabilistic mechanism underlying the assignments $\bar{\mathbf{z}}$.

In classical SBM a Dirichlet–multinomial distribution on $\bar{\mathbf{z}}$ is chosen, obtained by marginalizing the vector of community assignment probabilities $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_k) \sim Dir(\boldsymbol{\gamma})$ out of the likelihood for $\bar{\mathbf{z}}$ assuming $p(\bar{z}_i = k^\star|\pi) = \pi_{k^\star}$, for $k^\star = 1, \ldots, k$, $i = 1, \ldots, n$.

Relabeling Issue

So far we have considered *labeled* clusters, identified by $\bar{\mathbf{z}}$. This means that a vector $\bar{\mathbf{z}}$ and its relabelings are regarded as distinct objects, even though they define the same partition. Throughout the rest of the work we will rely on a generic $\mathbf{z}$ to denote all relabelings of $\bar{\mathbf{z}}$ that lead to the same partition. Note that (3.7) is invariant under relabeling and hence, $p(\mathbf{Y}|\mathbf{z}) = p(\mathbf{Y}|\bar{\mathbf{z}})$ (Legramanti et al., 2020).

Under those assumption we have then the folloving Bayesian SBM specification

$$\begin{aligned}
&\boldsymbol{\pi} \sim Dir(\overbrace{\gamma, \dots \gamma}^{k}) = Dir(\boldsymbol{\gamma});\\
&\mathbf{Q}_{rs} \overset{ind}{\sim} Beta(a,b) \quad r,s = 1,\dots,k;\\
&p(z_i = h|\boldsymbol{\pi}) = \pi_h, \quad i = 1,\dots,n;\ \ h = 1,\dots,k;\\
&\mathbf{Y}_{ij}|\mathbf{z},\mathbf{Q} \overset{ind}{\sim} Bernoulli(\theta_{ij}), \quad \theta_{ij} = \mathbf{Q}_{z_i z_j}, \quad 1 \le i < j \le n.
\end{aligned} \tag{3.8}$$

Estimation and Prediction of the Community Structure

From the specification of the model in (3.8) we can use Gibbs sampling to approximate the posterior distribution of both $\boldsymbol{\pi}$ and $\mathbf{Q}$ in order to predict the block structure of the network. Gibbs sampling is an iterative simulation scheme that operates by repeatedly drawing in turn each of a set of unknown random variables or vectors, each conditionally on the values of all of the other random variables (Nowicki and Snijders, 2001). This scheme can be applied to $\boldsymbol{\pi}, \mathbf{Q}$ and $\mathbf{z}$ where $(\boldsymbol{\pi}, \mathbf{Q})$ can be treated as a single random vector with prior density $p(\boldsymbol{\pi}, \mathbf{Q}) = p(\boldsymbol{\pi})p(\mathbf{Q})$.

Given current values $\mathbf{z}^{(p)}, \mathbf{Q}^{(p)}$ and $\boldsymbol{\pi}^{(p)}$ the next values $\mathbf{z}^{(p+1)}, \mathbf{Q}^{(p+1)}$ and $\boldsymbol{\pi}^{(p+1)}$ are determined as follows:

(a) $\mathbf{Q}^{(p+1)}, \boldsymbol{\pi}^{(p+1)}$ is drawn from the posterior distribution of $(\mathbf{Q}, \boldsymbol{\pi})$ given the complete data $(\mathbf{z}^{(p)}, \mathbf{Y})$.

(b) For each value $i = 1, \dots, n$ in turn $z_i^{(p+1)}$ is drawn from the conditional distribution of z_i given the values $\mathbf{Q}^{(p+1)}, \boldsymbol{\pi}^{(p+1)}$, $\mathbf{Y}$, $z_{h_\star}^{(p+1)}$ for $h_\star = 1, \dots, i-1$ and $z_{h_\star}^{(p)}$ for $h_\star = i+1, \dots, n$.

Because step (a) does not depend on the current value of $(\boldsymbol{\pi}, \mathbf{Q})$, only the initial value $\mathbf{z}^{(1)}$ is important as a starting condition. It follows from the general theory of Gibbs sampling that for this iteration scheme, irrespective of the starting values, the distribution of $(\boldsymbol{\pi}^{(p)}, \mathbf{Q}^{(p)})$ converges to the posterior distribution, given the observed data $\mathbf{Y}$, and the distribution of $\mathbf{z}^{(p)}$ converges to the posterior predictive distribution.

Model (3.8) uses for both parameters a Dirichlet prior distribution allowing thus to use well-known results on the Bayesian analysis of multinomially distributed data to derive the posterior distribution of $(\mathbf{Q}, \boldsymbol{\pi})$ used in step (a) of the Gibbs sampler. For step (b), instead, the conditional distribution of z_i given $\mathbf{Q}, \boldsymbol{\pi}, \mathbf{Y}$ and $z_1, \dots, z_{i-1}, z_{i+1}, \dots, z_n$ can be derived from the conditional joint distribution of $(\mathbf{Y}, \mathbf{z})$ in (3.2).

For a more exhaustive discussion on the quantities involved in the Gibbs sampler used in this scenario please refer to Snijders and Nowicki (1997); Nowicki and Snijders (2001).

3.3.3. Limitations and Possible Extensions

The model described above has some limitations, among which it is worth mentioning the lack of flexibility to generate networks with structure even moderately similar to that found in most empirical network data, meaning that a posteriori fits to such data often give poor results (Karrer and Newman, 2011). Therefore, fitting a simple stochastic block model to the structure of a complex network is likely to miss important features of the data and can in some cases give radically incorrect answers. A possible solution to this problem comes from the *degree-corrected stochastic block model* (Karrer and Newman, 2011; Riolo et al., 2017; Newman and Reinert, 2016).

In practice, in this slightly different formulation, not just a single edge between any pair of nodes i, j it is placed but a Poisson distributed number with mean ω_{rs}, or half that number when $i = j$. In this variant of the model the generated network may contain both multiedges and self edges, which many real networks do not include. But in typical situations the edge probabilities are so small that both multiedges and self-edges occur with very low frequency, and the model is virtually identical to the original (Bernoulli) formulation given above.

Another fundamental shortcoming of the classical SBM is that it requires to know in advance how many communities a network contains. This information usually is not known *a priori*, meaning that it should be estimated in some way from the data (Legramanti et al., 2020; Riolo et al., 2017; Geng et al., 2019).

3.4. Stochastic Block Models with Unknown Number of Communities K

In this section, we consider a Bayesian formulation of a SBM with standard conjugate Dirichlet-multinomial prior on the community assignments and Beta priors on the edge probabilities indroduced in (3.8). The *"only"* difference here is that K is unknown.

To address this problem, often, a frequentist estimate of K is first determined throug a suitable criterion (Geng et al., 2019) with a subsequent Bayesian model fitted with the estimated number of components. In a fully Bayesian framework, a prior distribution is assigned on the number of communities which can be either updated at each iteration of an MCMC algorithm or can be marginalised out obtaining thus a collapsed MCMC.

3.4.1. Bayesian Community Detection via Dirichlet Process

A natural choice of a prior distribution on $(z_1, z_2, \ldots, z_n)$ that allows automatic inference on the number of communities K is the Dirichlet process. As seen in section 2.2.3, we can obtain a sample $(z_1, z_2, \ldots, z_n)$ from a DP through the CRP (Neal, 2000). Namely, we have: $z_1 = 1$ and z_i, for $i = 2, \ldots, n$ are defined through the following conditional distribution:

$$p(z_i = c|z_1, \ldots, z_{i-1}) \sim \begin{cases} |c| & \text{at an existing community labeled } c \\ \alpha & \text{if } c \text{ is a new community.} \end{cases} \tag{3.9}$$

This prior for $\{z_i\}$ can also be defined through a stochastic process where at any positive-integer time n, the value of the process is a partition C_n of the set $\{1, 2, 3, \ldots, n\}$, whose probability distribution is determined as follows. At time $n+1$ the element $n+1$ is either (i) added to one of the blocks of the partition C_n, where each block is chosen with probability $|c|/(n+1)$ where $|c|$ is the size of the block, or (ii) added to the partition C_n as a new singleton block, with probability $1/(n+1)$. The distribution of z_i is then given by the stick-breaking formulation of a Dirichlet process:

$$z_i \sim \sum_{h=1}^{\infty} \pi_h \delta_h, \quad \pi_h = v_h \prod_{l<h} (1 - v_l), \quad v_h \sim Beta(1, \alpha). \tag{3.10}$$

As seen in section 2.2.3, CRP assigns large probabilities to communities with relatively smaller size. Indeed, under (3.9) the probability of block-sizes $s = (s_1, s_2, \ldots, s_t)$ of a partition C_n is

$$p_{DP(s)} \propto \prod_{j=1}^{t} s_j^{-1}. \tag{3.11}$$

It has been empirically observed that CRPs often have the tendency to create tiny extraneous clusters and recently it has been established that CRPs lead to inconsistent estimation of the number of clusters in a fairly general setting even when the sample size grows to infinity (Miller and Harrison, 2018).

A possible solution to this issue consists in replacing the DP with a Gibbs-type prior with $\sigma < 0$ as distribution for the z_i's.

3.4.2. Bayesian Community Detection via Dirichlet Process via Mixture of Finite Mixtures

If we replace the DP with a Gibbs-type prior with $\sigma < 0$ as distribution for the z_i's we obtain model (2.23) of section 2.4.2 which, as seen before, admits a Pólya urn scheme akin to CRP allowing us to develop an efficient MCMC algorithm. Moreover, this choice alleviates the drawback of CRP by automatic model-based pruning of the tiny extraneous clusters leading to consistent estimate of the number of communities (Miller and Harrison, 2018). As mentioned in section 2.4.2, if we compare the probability of a new cluster in the Pólya urn scheme induced by (2.23) with respect to CRP, we can see that it is slowed down by the factor $V_n(|C_{n-1}| + 1)/V_n(|C_{n-1}|)$ preventing the creation of many small communities. Following Geng et al. (2019), an alternative way to understand this is by looking at the probability of block-sizes $s = (s_1, s_2, \ldots, s_t)$ of

a partition C_n with $t = |C_n|$ under Gibbs-type prior with $\sigma < 0$. As seen in section 2.4.2, the probability of the cluster-sizes $(s_1, \ldots, s_t)$ arising from the Pólya urn scheme for model (2.23) is given by:

$$p_{MFM}(s) \propto \prod_{j=1}^{t} s_j^{\gamma-1}. \tag{3.12}$$

From (3.11) and (3.12), it is easy to see that MFM assigns comparatively smaller probability to clusters with small sizes. The parameter $\gamma > 0$ controls the relative size of the clusters; small γ favors lower entropy π's, while large γ favors higher entropy π's.

Adapting model (2.23) to the SBM setting, the model and prior can be expressed hierarchically as

$$\begin{aligned}
&K \sim p(\cdot), \quad \text{where } p(\cdot) \text{ is a p.m.f. on } \{1, 2, \ldots, \}; \\
&\mathbf{Q}_{rs} \overset{ind}{\sim} Beta(a, b) \quad r, s = 1, \ldots, k; \\
&p(z_i = h|\boldsymbol{\pi}) = \pi_h, \quad i = 1, \ldots, n; \;\; h = 1, \ldots, k; \\
&\boldsymbol{\pi} \sim Dir(\underbrace{\gamma, \ldots \gamma}_{k}); \\
&\mathbf{Y}_{ij}|\mathbf{z}, \mathbf{Q} \overset{ind}{\sim} Bernoulli(\theta_{ij}), \quad \theta_{ij} = \mathbf{Q}_{z_i z_j}, \quad 1 \leq i < j \leq n.
\end{aligned} \tag{3.13}$$

The hierarchical model defined in (3.13) is referred as MFM-SBM. While MFM-SBM admits a CRP representation, an important distinction from infinite mixture models hinges on the fact that for any given prior predictive realization, one draws a value of K and as n grows the nodes are distributed into the K communities. On the other hand, the number of communities keeps growing with n for the infinite mixture models (Geng et al., 2019; Miller and Harrison, 2018).

As in the case with known K, the main goal is to sample from the posterior distribution of the unknown parameters (here represented by K, $\mathbf{z} = (z_1, \ldots, z_n) \in \{1, \ldots, K\}^n$ and $\mathbf{Q} = (\mathbf{Q}_{rs}) \in [0, 1]^{K \times K}$) and a Gibbs sampler is typically the best strategy to achieve it. Geng et al. (2019) developed a *collapsed* Gibbs sampler marginalizing over the number of communities. This produces a more efficient sampler avoiding resorting to complicated reversible jump MCMC algorithms or allocation samplers. We discuss more about their Gibbs samper in the next section.

3.5. Collapsed Gibbs Sampler for MFM-SBM

In our work we used the collapsed Gibbs-sampler developed by Geng et al. (2019). It is a direct extension of the algorithm presented in Miller and Harrison (2018) which is an adaptation to MFMs of algorithm 2 presented in Neal (2000) for DPMs. In this sampler the number K is marginalized out in order to avoid complicated reversible

jump MCMC algorithms or even allocation samplers. Therefore, this collapsed Gibbs-sampler efficiently cycles through the full conditional distribution of $\mathbf{Q}$ and $z_i|z_{-i}$ for $i = 1, 2, \ldots, n$, where $z_{-i} = z \setminus \{z_i\}$.

3.5.1. Pseudo Code

We implemented the collapsed Gibbs-sampler based on the peudocode of Algorithm 1 that can be found in Geng et al. (2019).

Algorithm 1: Collapsed Gibbs sampler for MFM-SBM. Pseudo-code

Result: return posterior sample from $\mathbf{Q}$ and $\mathbf{z}$ given $\mathbf{Y}$.

Initialize $t = t.start$, $\mathbf{z}= (z_1, \ldots, z_n)$ and $\mathbf{Q}= (\mathbf{Q}_{rs})$.

for *iter in 1 to M* **do**

Update $\mathbf{Q}= (\mathbf{Q}_{rs})$ conditional on $\mathbf{z}$ in a closed form as

$$\mathbb{P}(\mathbf{Q}_{rs}|\mathbf{Y}) \sim Beta(e_{rs} + a, n_{rs} - e_{rs} + b) \tag{3.14}$$

Where e_{rs} is computed as (3.4), and n_{rs} is computed as (3.3), for $r = 1, \ldots t; s = 1, \ldots, t$. Here t is the number of clusters formed by current $\mathbf{z}$.

for *i in 1 to n* **do**

Update $\mathbf{z}= (z_1, \ldots, z_n)$ conditional on $\mathbf{Q} = \mathbf{Q}_{rs}$ using the closed form expression for $\mathbb{P}(z_i = c|z_{-i}, \mathbf{Y}, \mathbf{Q})$:

$$\propto \begin{cases} [|c| + \gamma][\prod_{j \neq i} \mathbf{Q}_{cz_j}^{\mathbf{Y}_{ij}}(1 - \mathbf{Q}_{cz_j})^{(1-\mathbf{Y}_{ij})}], & \text{at an existing table } c \\ \frac{V_n(|C_{-i}|+1)}{V_n(|C_{-i}|)}\gamma m(\mathbf{Y}_i) & \text{if } c \text{ is a new table} \end{cases} \tag{3.15}$$

Where C_{-i} denotes the partition obtained by removing z_i and

$$m(\mathbf{Y}_i) = \prod_{t^\star=1}^{|C_{-i}|} [\mathbf{B}(a, b)]^{-1} \mathbf{B}\left[\sum_{j \in c_{t^\star}, j \neq i} \mathbf{Y}_{ji} + a, |c_{t^\star}| - \sum_{j \in c_{t^\star}, j \neq i} \mathbf{Y}_{ji} + b\right] \tag{3.16}$$

where $j \in c_{t^\star}$ is referred to the units that belong to the $t^\star - th$ table in the partition C_{-i} composed by $|C_{-i}|$ tables.

end

end

As we can see from Algorithm 1, we start by initializing the number $t = t.start$ and then we randomly allocate the observartions into $t = t.start$ communities in the vector $\mathbf{z}_{update}$. The initialization of $\mathbf{Q}$ is not really necessary (in our implementation we set all the entrances equal to NA) since once we set $\mathbf{z}_{update}$ we adjust $\mathbf{Q}$ by sampling from (3.14) of Algorithm 1. What is really important, concerning the matrix $\mathbf{Q}$, is how

it is updated in the second `for` cycle. In order to fully understand why this passage is crucial it is better first to understand what happens inside the second `for` cycle of Algorithm 1.

In this part of the code we update the vector $\mathbf{z}_{update}$ conditional on $\mathbf{Q}$ by relying on probability (3.15). Looking at (3.15) we can see the adaptation of algorithm 2 of Neal (2000) to the SBM-MFM model. For each observation $i \in \{1, \dots, n\}$, the probability of being associated to an existing table c is proportional to the cardinality of that table plus the parameter of the Dirichlet distribution, $|c| + \gamma$, times the likelihood $F(\mathbf{Y}_{ij}, \mathbf{Q}_{cz_i})$, while, the probability of being associated to a new table is proportional to $\gamma V_n(t_- + 1)/V_n(t_-)$ times $m(\mathbf{Y}_i)$, where $t_- = |C_{-i}|$ and $V_n(t_-)$ is the same of (2.24), i.e. $V_n(t_-) = \sum_{k=1}^{\infty} \frac{k_{(t_-)}}{(\gamma k)^{(n)}} p_K(k)$.

The only differences with respect to Algorithm 2 of Neal (2000) consist that, for each $i \in \{1, \dots, n\}$, $|c| + \gamma$ is replaced by $|c|$ and $\gamma V_n(t_- + 1)/V_n(t_-)$ is replaced by α (i.e. the concentration parameter of the DP process). As pointed out by Miller and Harrison (2018), the differences between the MFM and DPM versions of the algorithm are precisely the same as the differences between their respective urn/restaurant process.

Regarding $m(\mathbf{Y}_i)$, it represents the sample information of the $i-th$ observation in relation with all the other observations. In fact, under (3.13), if we define

$$
\begin{aligned}
m(\mathbf{Y}_i) &:= \prod_{t^\star=1}^{|C_{-i}|} \int_Z \prod_{j \in c_{t^\star}} f(\mathbf{Y}_{ij}|Z_j) P^\star(dZ) \\
&\quad \text{by assumptions of (3.11)} \\
&= \prod_{t^\star=1}^{|C_{-i}|} \int_0^1 \prod_{j \in c_{t^\star}} \frac{\mathbf{Q}_{c_{t^\star}c_i}^{\mathbf{Y}_{ij}} (1 - \mathbf{Q}_{c_{t^\star}c_i})^{(1-\mathbf{Y}_{ij})} \mathbf{Q}_{c_{t^\star}c_i}^{a-1} (1 - \mathbf{Q}_{c_{t^\star}c_i})^{b-1}}{\text{Beta}(a,b)} d\mathbf{Q}_{c_{t^\star}c_i} \\
&\quad \text{setting } \mathbf{Q}_{c_{t^\star}c_i} = \theta_{t^\star i} \\
&= \prod_{t^\star=1}^{|C_{-i}|} \frac{1}{\text{Beta}(a,b)} \int_0^1 \underbrace{\theta_{t^\star i}^{a+\sum_{j \in c_{t^\star}; j \neq i} \mathbf{Y}_{ij} - 1} (1 - \theta_{t^\star i})^{b+\sum_{j \in c_{t^\star}; j \neq i}(1-\mathbf{Y}_{ij})-1}}_{\text{Kernel of Beta distribution}} d\theta_{t^\star i} \\
&= \prod_{t^\star=1}^{|C_{-i}|} \frac{\text{Beta}(a + \sum_{j \in c_{t^\star}; j \neq i} \mathbf{Y}_{ij}; |c_{t^\star}| - \sum_{j \in c_{t^\star};\ j \neq i} \mathbf{Y}_{ij} + b)}{\text{Beta}(a,b)} \\
&= (3.16).
\end{aligned}
$$

In the above equation we started by defining $m(\mathbf{Y}_i)$ as the product of marginal likelihoods parametrized as in the mixture model introduced in (2.23). In particular, under MFM-SBM (3.13), the parameter $\mathbf{Q}_{rs}$ and the associated prior distribution $Beta(a, b)$ (for every $s, r = 1, \dots, K$) have respectively the same role of the Z_i's and $P^\star$ in model (2.23).

Having explained the nature of probability (3.13), we now proceed to explain all possible scenarios in the reallocation of each observation i in $\mathbf{z}_{update}$. Let us suppose we computed (3.13) for a general observation i, then, we must sample from the vector

made by all the natural numbers $\{1, \ldots, |C_{-i}|, |C_{-i}| + 1\}$ matching the labels of all the "tables" of the partition obtained exluding the $i - th$ observation plus a new one. The possible outcomes of the reallocation are then the following:

1. Observation i is associated to its previous table, say c_i or to an existing one with the addition that $|c_i| \neq 1$. In this case there is nothing to do since the dimensionality of the matrix $\mathbf{Q}$ does not change and all table labels remain the same.

2. Having $|c_i| \neq 1$, observation i is associated to a new table. In this case there is an increment of the number of tables and thus it is necessary to adapt matrix $\mathbf{Q}$. In particular we need to compute the probabilities of connection between the existing tables (communities) and the new one and the probability of connection inside the new table. In order to achieve this we use (3.14) of Algorithm 1 in which we fix $r = t$.

3. Having $|c_i| = 1$, observation i is associated to an existing table. In this case there is a decrement of the number of tables and thus it is necessary to adapt matrix $\mathbf{Q}$. In particular we need to remove the column (and row) corresponding to the old c_i and perform a relabeling operation so that no superfluous labels of empty tables are present.

4. Having $|c_i| = 1$, observation i is associated to a new table. In this case the number of tables remains the same but nevertheless it is necessary to adapt matrix $\mathbf{Q}$. In particular we need to compute the probabilities of connection between the existing tables (communities) and the new one and the probability of connection inside the new table by means (3.14) of Algorithm 1 in which we fix $r = t$ as in point 2. Finally it is necessary also a relabeling operation, like in point 3.

It is clear now why the update of matrix $\mathbf{Q}$ happens also in the second `for` cycle and not only in the first one as it might seem from a first look at the pseudo-code of Algorithm 1. Finally, it should be noted that the prior distribution for K, $p_K(k)$, only comes into play when calculating the probability of a new table, more precisely when determining $V_n(t_-)$ through (2.24).

3.5.2. The Coefficient $V_n(t)$

Computation

In order to use succesfully Algorithm 1, it is necessary to compute $V_n(t)$ defined as (2.24). Despite $V_n(t)$ satisfies the recursion (2.25) we cannot use it in order to precompute values of $V_n(t)$ for n of medium size. This is due to the fact that equation (2.25) requires always to compute $V_n(0)$ as a base case of the recursion. This is unfeasible for $n > 170$ since equation (2.24) presents at the denomiator an ascending

factorial given by $(\gamma k)^{(n)}$, meaning that, in the base scenario, when both k and γ are equal to 1, $(\gamma k)^{(n)} = n!$. In the other possible cases we have $(\gamma k)^{(n)} > n!$. Since a normal PC with 64 bit cannot compute directly a factorial greater than 170, it is clear how it is not feasible to use the recursive formula (2.25) to precompute the coefficients $V_n(t)$ for medium/large networks. It might be tempting to apply $log(\cdot)$ to formula (2.25), however the resulting expression would be not tractable from an analytical point of view since we would end up with a logarithm of a sum of elements that we would like to separate.

So, we decided to find a way to apply the $log(\cdot)$ to the expression of $V_n(t)$ in order to get the logarithm of the ascending factorial which is more tractable than the normal one. However the logarithmic transformation is not enough since if we apply $log(\cdot)$ directly to (2.24) we end up with the same problem of before (i.e. the logarithm of a sum). What we further need is to rewrite the expression for $log(V_n(t))$ using the following property for a finite sum, say $S_N = \sum_{k=1}^{N} t_k$:

$$\begin{aligned} log(S_N) &= log\left(\sum_{k=1}^{N} t_k\right) \\ &= log\left(\sum_{k=1}^{N} e^{log(t_k)}\right) \\ &\quad \text{setting } m := \max_{k=1,\dots,N}(log(t_k)) \\ &= m + log\left(e^{log(t_1)-m} + \dots + e^{log(t_N)-m}\right). \end{aligned} \tag{3.17}$$

Proof. By properties of logarithms

- N = 1

$$\begin{aligned} m &= log(t_1); \\ log(S_1) &= log(t_1) = m + 0 \end{aligned} \tag{3.18}$$

- N = 2

$$\begin{aligned} m &= max(log(t_1), log(t_2)); \\ log(S_2) &= log(t_1 + t_2) \\ &= log(e^{log(t_1)} + e^{log(t_2)}) \\ &= log(e^m(e^{log(t_1)-m} + e^{log(t_2)-m})) \\ &= m + log(e^{log(t_1)-m} + e^{log(t_2)-m}) \end{aligned} \tag{3.19}$$

⋮

- General N

$$
\begin{aligned}
m &= \max_{k=1,\dots,N} (log(t_k))\,; \\
log(S_N) &= log\left(\sum_{k=1}^{N} t_k\right) \\
&= log\left(\sum_{k=1}^{N} e^{log(t_k)}\right) \\
&= log\left(\sum_{k=1}^{N} (e^{m}(e^{log(t_k)-m})\right) \\
&= m + log\left(e^{log(t_1)-m} + \ldots + e^{log(t_N)-m}\right).
\end{aligned}
\tag{3.20}
$$

□

Formally, for any $t = \{1, 2, \ldots\}$, $V_n(t)$ is defined as an infinite sum (and thus $log(V_n(t))$ as the log of an infinite sum), but it can be numerically approximated by setting a threshold τ so that all successive terms smaller than that treshold are excluded from the sum, turning it thus into a finite sum. This is possible since the series for $V_n(t)$ (equation (2.24)) always converges to a finite value when $1 \leq t \leq n$ (Miller and Harrison, 2018) which it is always the case since in our case we compute $V_n(t_-)$ where $t_- = |C_{-i}|$. So, what we have is the following:

$$
\begin{aligned}
log(V_n(t)) &\cong log\left(\sum_{k=1}^{N_\tau} \frac{k_{(t)}}{(\gamma k)^{(n)}} p_K(k)\right) \\
&= log\left(\sum_{k=1}^{N_\tau} e^{log\left(\frac{k_{(t)}}{(\gamma k)^{(n)}} p_K(k)\right)}\right) \\
m &:= \max_{k=1,\dots,N_\tau} \left(log\left(\frac{k_{(t)}}{(\gamma k)^{(n)}} p_K(k)\right)\right) \\
&= log\left(\sum_{k=1}^{N_\tau} e^{m} e^{log\left(\frac{k_{(t)}}{(\gamma k)^{(n)}} p_K(k)\right) - m}\right) \\
&= m + log\left(\sum_{k=1}^{N_\tau} e^{log\left(\frac{k_{(t)}}{(\gamma k)^{(n)}} p_K(k)\right) - m}\right).
\end{aligned}
\tag{3.21}
$$

Moreover we have the following relation:

$$
log\left(\frac{k_{(t)}}{(\gamma k)^{(n)}} p_K(k)\right) = log\left(\frac{\Gamma(k+1)}{\Gamma(k-t+1)} \frac{\Gamma(k\gamma)}{\Gamma(k\gamma+n)} p_K(k)\right),
$$

so, in R we can use the function `lgamma` which is quite efficient. Notice that we apply the logarithm for then compute $e^{log(V_n(t))-log(V_n(t_-))}$ in (3.15) of Algorithm 1.

Algorithm 2: Iterative computation of $log(V_n(t))$

Result: compute in a iterative way $log(V_n(t))$ untill the next term of the sum is less than a treshold τ.

Initialize t, n, the treshold τ and set $b = \infty$, $a = -\infty$,$S = 0$;

while $e^b > \tau$ **do**

$k = t$

$b = log\left(\frac{k_{(t)}}{(\gamma k)^{(n)}} p_K(k)\right)$

$m = \max(a, b)$

$S = m + log(e^{b-m} + e^{a-m})$

$a = S$

$k = k + 1$

end

return S

Since N_τ is not known a priori we should write (3.21) in a iterative way as reported in Algorithm 2.

It should be noted that in Algorithm 2 we start from $k = t$ since $k_{(t)} = 0$ when k is a positive integer less than t and thus the first terms of $V_n(t)$ (equation (2.24)), for $k < t$ are equal to zero.

Closed Form Expression of $V_n(t)$

Setting $\gamma = 1$ in model (2.23), and therefore in model (3.13), Gnedin et al. (2010) has discovered an example of a distribution on K, $p_K(k)$, for which $V_n(t)$ has a closed-form expression for every $t = \{1, 2, \ldots\}$. More precisely

$$\mathbb{P}(K = k) = \frac{\gamma_{gn}(1 - \gamma_{gn})_{k-1}}{k!}, \;\; k = 1, 2, \ldots \tag{3.22}$$

and

$$V_n(t) = \frac{(t-1)!(1 - \gamma_{gn})_{t-1}(\gamma_{gn})_{n-t}}{(n-1)!(1 + \gamma_{gn})_{n-1}}, \tag{3.23}$$

for $\gamma_{gn} \in (0, 1)$.

We refer to (3.23) as *Gnedin distribution* that can be used as prior for K and from a "coding" perspective it is more convenient to work with since it allows to avoid the approximation of $V_n(t)$ through Algorithm 2. As reported in Chapter 4, this closed form that can be obtained for $V_n(t)$ allows to get a time improvement in certain situations compared to its approximate version. Alternatively, a common choice in literature on the distribution to put on K, i.e. $p_K(k)$, is given by a *zero-truncated-Poisson* distribution with parameter $\lambda > 0$. In contrast to the zero-truncated-Poisson distribution, the Gnedin distribution has an infinite expected value.

Asymptotics of $V_n(t)$

Miller and Harrison (2018) proved that, for any $t \in \{1, 2, \ldots\}$, if $p_K(t) > 0$, then, as

$n \to \infty$

$$V_n(t) \sim \frac{t_{(t)}}{(\gamma t)^{(n)}} p_K(t) \sim \frac{t!}{n!} \frac{\Gamma(\gamma t)}{n^{\gamma t - 1}} p_K(t) \tag{3.24}$$

where, $a_n \sim b_n$ indicates $a_n / b_n \to 1$ as $n \to \infty$.

In particular, $V_n(t)$ has a simple interpretation, asymptotically, it behaves like the $k = t$ term in the series expressed in (2.24). For the proof we refer to the supplementary material of Miller and Harrison (2018).

3.5.3. Posterior Inference

Bayesian Point Estimation of z

While algorithmic methods return a single estimated partition, MFM-SBMs provide the whole posterior distribution over the space of partitions. To fully exploit such a posterior, we followed Legramanti et al. (2020) who adapted the decision–theoretic approach of Wade et al. (2018) to the community detection setting. In this way, they summarized posterior distributions on partitions leveraging the variation of information (VI) metric, which quantifies distances between two clusterings by comparing their individual and joint entropies. More formally, the VI is defined as

$$\begin{aligned} &\mathsf{VI}(\mathbf{z}, \mathbf{z}^\star) = \mathsf{H}(\mathbf{z}) + \mathsf{H}(\mathbf{z}^\star) - 2\mathsf{I}(\mathbf{z}, \mathbf{z}^\star) \\ &= -\sum_{i=1}^{t} \frac{n_{i+}}{n} log_2\left(\frac{n_{i+}}{n}\right) - \sum_{j=1}^{t^\star} \frac{n_{+j}}{n} log_2\left(\frac{n_{+j}}{n}\right) - 2\sum_{i=1}^{t}\sum_{j=1}^{t^\star} \frac{n_{ij}}{n} log_2\left(\frac{n_{ij} n}{n_{i+} n_{+j}}\right), \end{aligned} \tag{3.25}$$

where $n_{ij} = |C_i \cap C_j^\star|$, which is the cardinality of the intersection between C_i, the set of data point indices in cluster i under $\mathbf{z}$, and $C_j^\star$, the set of data point indices in cluster j under $\mathbf{z}^\star$, for $i = 1, \ldots, t$ and $j = 1, \ldots, t^\star$ with t and $t^\star$ representing the number of clusters in $\mathbf{z}$ and $\mathbf{z}^\star$, respectively. Finally, $n_{i+} = \sum_j n_{ij}$ and $n_{+j} = \sum_i n_{ij}$.

The first two terms of (3.25) represent the entropy of the two clusterings, which measures the uncertainty in bits of the cluster allocation of an unknown randomly chosen data point given a particular clustering of the data points. The last term is the mutual information between the two clusterings and measures the reduction in the uncertainty of the cluster allocation of a data point in $\mathbf{z}$ when we are told its cluster allocation in $\mathbf{z}^\star$ (Wade et al., 2018). The VI ranges from 0 to $log_2(n)$.

Intuitively, VI measures the amount of information in two clusterings relative to the information shared between them, thus providing a metric that decreases to 0 as the overlap between two partitions grows; see Wade et al. (2018) for a discussion of the key properties of VI that facilitate uncertainty quantification on partitions. Under this framework, a formal Bayesian point estimate for $\mathbf{z}$ is the partition with the lowest posterior averaged VI distance from the other clusterings, namely:

$$\begin{aligned}\hat{\mathbf{z}} &= \arg\min_{\mathbf{z}^\star} \mathbb{E}_{\mathbf{z}}[\mathsf{VI}(\mathbf{z}, \mathbf{z}^\star)|\mathbf{Y}] \\ &= \arg\min_{\mathbf{z}^\star} \sum_{l=1}^{n} log\left(\sum_{l'=1}^{n} \mathbb{1}(z^\star_{l'} = z^\star_l)\right) - 2\sum_{l=1}^{n} \mathbb{E}\left[log\left(\sum_{l'=1}^{n} \mathbb{1}(z_{l'} = z_l, z^\star_{l'} = z^\star_l)\right)|\mathbf{Y}\right]\end{aligned} \tag{3.26}$$

where the expectation is taken with respect to $\mathbf{z}$.

For a given $\mathbf{z}^\star$, the second term in (3.26) can be approximated based on the MCMC output, and evaluating this term is of order $O(Mn^2)$ (where M is the number of MCMC samples). This may be computationally demanding if the number of MCMC samples is large and if (3.26) must be evaluated for a large number of $\mathbf{z}^\star$. Alternatively, as shown from Wade et al. (2018) one can use Jensen's inequality, swapping the $log(\cdot)$ and expectation, to obtain a lower bound on the expected loss which is computationally more efficient to evaluate:

$$\arg\min_{\mathbf{z}^\star} \sum_{l=1}^{n} log\left(\sum_{l'=1}^{n} \mathbb{1}(z^\star_{l'} = z^\star_n)\right) - 2\sum_{l=1}^{n} log\left(\sum_{l'=1}^{n} \mathbb{P}(z_{l'} = z_l|\mathbf{Y})\mathbb{1}(z^\star_{l'} = z^\star_l)\right) \tag{3.27}$$

Minimization of (3.27) only depends on the posterior through the posterior similarity matrix (i.e. a matrix containing the posterior probabilities that the observations i and j are in the same cluster), which can be pre-computed based on the MCMC output. In this case, computational complexity for a given $\mathbf{z}^\star$ is reduced to $O(n^2)$.

Due to the huge dimensions of the partition space, computing the lower bound in (3.27) for every possible $\mathbf{z}^\star$ is practically impossible. Hence, even for moderate n, the optimization is typically carried out through a greedy algorithm (Wade et al., 2018), as in the R package `mcclust.ext`.

Measure the Accuracy of Clustering

In simulation studies, once obtained the Bayesian point estimate of $\mathbf{z}$, say $\hat{\mathbf{z}}$, typically one would like to compare the accuracy of the estimate with the true configuration, say $\mathbf{z}_0$. To do so, we can use the VI defined in (3.25) or alternatively the Rand index (Geng et al., 2019). In particular, the Rand index is used to measure the accuracy of clustering. Given two partition of $\{1, 2, \ldots, n\}$, C_1 and C_2 induced respectively from $\{Z_1, \ldots Z_n\}$ and $\{Z'_1, \ldots, Z'_n\}$, let a, b, c and d, respectively denote the numberof pairs of elements of $\{1, 2, \ldots, n\}$ that are (a) in the same set in C_1 and a same set in C_2, (b) in different sets in C_1 and different sets in C_2, (c) in a same stet in C_1 but in different sets in C_2, and (d) in different sets in C_1 and a same set in C_2. The Rand index (RI) is then defined as

$$\mathsf{RI} = \frac{a+b}{a+b+c+d} = \frac{a+b}{\binom{n}{2}}. \tag{3.28}$$

As we can deduce from (3.28), $0 \leq \text{RI} \leq 1$ with a higher value indicating a better agreement between the two partitions. In particular, RI $= 1$ indicates C_1 and C_2 are identical regardless of possible different labelling of the nodes.

Estimation of K from the Posterior

In the collapsed Gibbs sampler of Algorithm 1 k is marginalized out and therefore is not possible to obtain samples from the posterior distribution of k. What is possible to obtain, instead, is the number of observed clusters, $T = |C|$ (here intended as a random variable) which should not be confused with the number of components K. Namely, the posterior on $T = |C|$ is easily estimated from posterior samples of C induced by $\mathbf{z}$'s. In practice, we choose the mode of the posterior distribution of $|\mathbf{z}|$ (i.e. the number of unique values, or occupied components, in $(z_1, \ldots, z_n)$).

We could use also the results proved by Miller and Harrison (2018) presented in section 2.4.2 in order to transform the posterior estimate of T in the posterior estimate of K. Geng et al. (2019), however, decided to estimate K based only on the posterior distribution of $|\mathbf{z}|$ which is asymptotically justified by Theorem 2.1 of section 2.4.2.

4. Illustrations

In this chapter we present the simulation studies and the test on the dolphin social network data we performed in order to get an empirical validation of the methods introduced in the previous chapter.

For the rest of the chapter, in accordance to Geng et al. (2019), for the MFM-SBM model defined in (3.13) we fixed $a = b = 1$ and $\gamma = 1$ respectively for the Beta prior and the simmetric Dirichlet distribution.

All R code developed for both the graphs and the algorithms can be found in https://github.com/Mich9534/ThesisSDS.

4.1. Simulation Studies

In this section we investigate the performance of the MFM-SBM approach from a variety of angles focusing mostly on the difference between the zero-truncated-Poisson prior and the Gnedin prior for K. At first, we outline the framework of the data-generating process used throughout this section.

4.1.1. Data Generating Process

We followed Geng et al. (2019) in the implementation of the data-generating process.

Step 1: Fix the number of nodes n and the true number of communities K.

Step 2: Generate the true clustering configuration $\mathbf{z}_0 = (z_{01}, \ldots, z_{0n})$ with $z_{0i} \in \{1, \ldots, K\}$. To this end, we fix the respective community sizes $n_{01}, \ldots, n_{0K}$, and without loss of generality, let $z_{0i} = l$ for all $i = \sum_{j<l} n_{0j} + 1, \ldots, \sum_{j<l} n_{0j} + n_{0l}$ and $l = 1, \ldots, K$. We consider both balanced (i.e., $n_{0l} \sim \lfloor n/K \rfloor$ for all l) and unbalanced networks.

Step 3: Construct the matrix $\mathbf{Q}$ in (3.1). Depending on how it is defined we can generate different network structures. For example if we set $\mathbf{Q}_{rs} = q + (p-q)\mathbb{1}(r = s)$, so that all diagonal entries of $\mathbf{Q}$ are p and all off-diagonal entries are q, with $p > q$, we get a classical *community structure* (see Figure 4.1, top panel). If, on the other hand, we allow some entries outside the main diagonal of $\mathbf{Q}$ to be greater than those on the major diagonal, we can obtain the so called *core-pheripery structure* (see Figure 4.1, bottom panel). Clearly, the more similar the values of p and q are, the weaker the

community structure.

Step 4: Generate the edges $\mathbf{Y}_{ij} \sim Bernoulli(\mathbf{Q}_{z_{0i}z_{0j}})$ independently for $1 \leq i < j \leq n$.

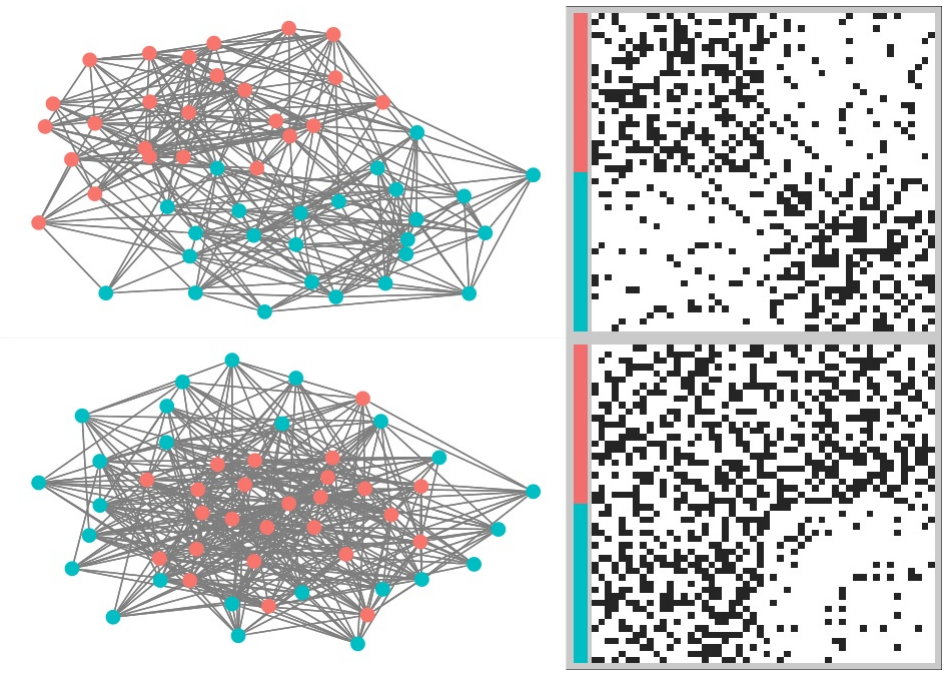


Figure 4.1.: Top panel refers to the network with community structure, bottom panel to the network with core periphery structure. Left: networks. Right: adjacency matrices of the networks. Both are balanced networks. Colors on the side correspond to the true communities. (For both: $K = 2, n = 50, p = 0.4$, $q = 0.1$).

4.1.2. Posterior Estimate for K

Before presenting the results we obtained in the empirical tests, we would like to point out that to estimate K from each MCMC we adopted the strategy of Geng et al. (2019) (i.e. through the posterior distribution of $|\mathbf{z}|$). Nevertheless, in this part we want to show with a small simulation study how the method proposed by Miller and Harrison (2018) to derive an estimate of K via formula (2.31) is the best choice when the size of the network is small.

Setting $K = 4, p = 0.5, q = 0.1$, we generated three different balanced network with $n = \{50, 100, 150\}$ and we then applied for each of them the MFM-SBM algorithm with a zero-truncated-Poisson prior for K with $\lambda = 1$. We ran the sampler for 300 iterations and we set a burn-in of 100. As we can see from Figure 4.2 we got $\hat{K} = K$ for $n = 100$ and $n = 150$, while for $n = 50$ we obtained an identical maximal probability for both $k = 3$ and $k = 4$. However, this is not surprising, since with $n = 50$ it is very

difficult to distinguish the true number of communities, especially with a decent value for K as in this case.

In order to have a comparison with the strategy used by Geng et al. (2019), we tried to estimate K in these three situations using the mode of the posterior distribution of $|\mathbf{z}|$. As we expected, looking at Figure 4.3, we obtained the true K only in the third scenario, where $n = 150$. This is not surprising since, as shown by Miller and Harrison (2018) in Theorem 2.1, the approach of Geng et al. (2019) is correct in estimating K when n is large. We have chosen $K = 4$ and compared relatively small values of n in order to emphasise this fact. When the numerosity of the network is low, it is always better to use (3.31) for the estimation of K.

In the simulations of the following sections, we tried different values for n and K in order to have a further confirmation of this point. In fact, the method used by Geng et al. (2019) showed unsatisfactory results when n was too small compared to K.

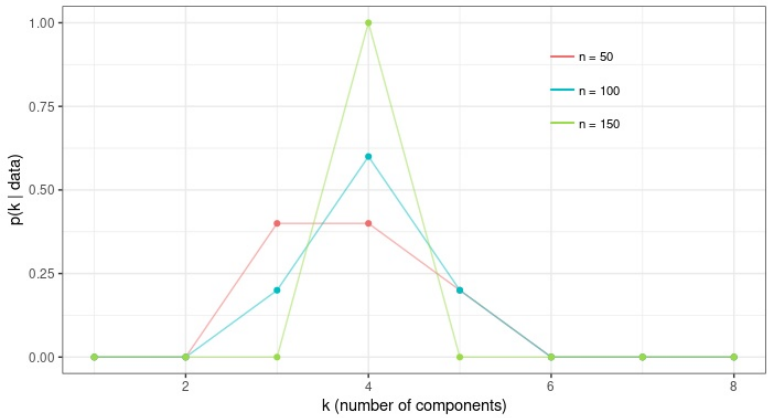


Figure 4.2.: posterior probability for $K = k$ varying n. For all scenarios: $K = 4, p = 0.5$, $q = 0.1$.

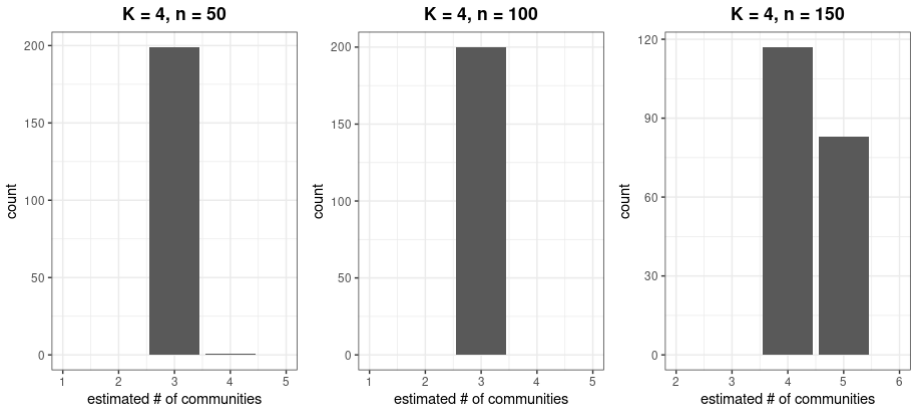


Figure 4.3.: posterior distribution of $|\mathbf{z}|$. For all scenarios: $K = 4, p = 0.5$, $q = 0.1$.

4.1.3. Estimation Performance

We now study the accuracy of the MFM-SBM model in terms of estimating the number of communities as well as the community memberships. In this section we compare model (3.13) with a standard zero-truncated-Poisson prior distribution on the number of communities K parametrized with $\lambda = 1$ and the Louvain algorithm introduced in section 3.2. We choose to compare the MFM-SBM model with the Louvain method (or also called hierachical modularity measure HMM) according to Geng et al. (2019) which showed that this method has the best overall performance among other available methods in the R package `igraph`.

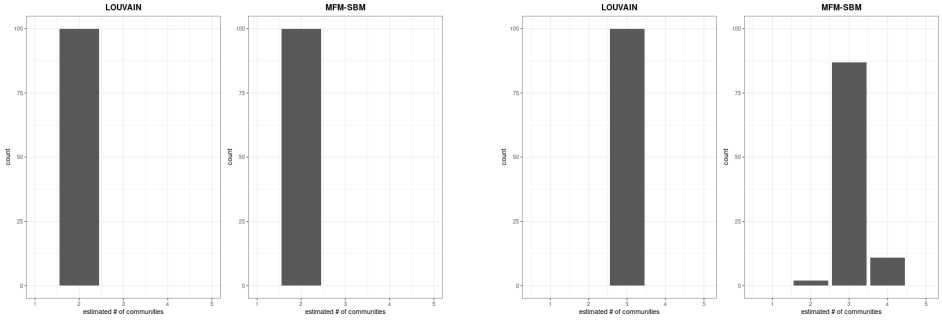
We considered balanced networks with 100 nodes and different choices of K and p, where we fixed $q = 0.1$. We generated 100 independent dataset using the data generating process described in 4.1.1 and compared the two different approaches based on the proportion of time the true K is recovered among the 100 replicates. For MFM-SBM we used random initializations into nine communities to run 10 MCMC chains in parallel for 600 iterations with a burn-in of 300 each, and took majority voting among the posterior modes of $|\mathbf{z}|$ from each chain to arrive at a final point estimate. The summaries from the 100 replicates are provided in Figure 4.4, and in Table 4.1 reported below.

From Table 4.1 we can see that, when the community structure in the network is *prominent* $(p = 0.5)$, both methods have high accuracy. However, the situation is dramatically different when the the block structure is *vague* ($p = 0.24$ for $K = 2$ and $p = 0.3$ for $K = 3$) where MFM-SBM comprehensively outperforms the Louvain algorithm.

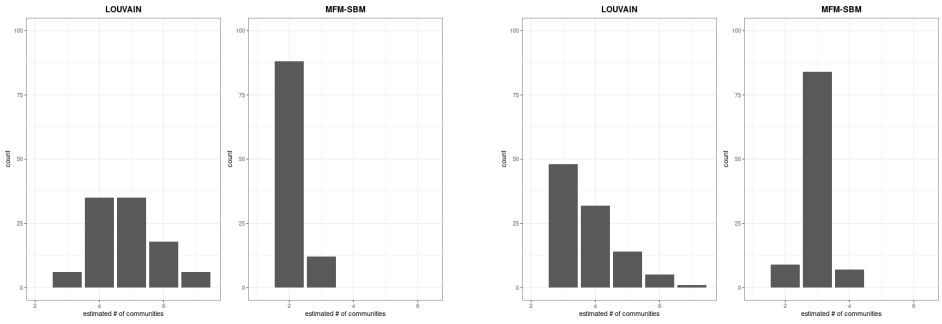
We compared also the estimation performance in recovering the true community memberships using both the Rand Index and the Variation Information as discrepancy measures. For MFM-SBM, inference on the clustering configuration has been obtained using the decision theoretic approach of Legramanti et al. (2020) through (3.24). From Table 4.1 we can see that essentially MFM-SBM provides more accurate estimation of the community memberships.

Table 4.1.: The value outside the parenthesis denotes the proportion of correct estimation of the number of clusters out of 100 replicates. The value inside the parenthesis denotes the average Rand Index (RI) value and the average Variation Information (VI) value between $\hat{\mathbf{z}}$ and $\mathbf{z}_0$.

(K, p)	MFM-SBM	LOUVAIN
$K = 2, p = 0.50$	1.00 (RI: 1.00, VI: 0.00)	1.00 (RI: 1.00, VI: 0.00)
$K = 2, p = 0.24$	0.88 (RI: 0.92, VI: 0.29)	0.00 (RI: 0.66, VI: 1.41)
$K = 3, p = 0.50$	0.87 (RI: 0.99, VI: 0.01)	1.00 (RI: 1.00, VI: 0.00)
$K = 3, p = 0.30$	0.84 (RI: 0.92, VI: 0.38)	0.48 (RI: 0.86, VI: 0.82)



(a) $K = 2, p = 0.5$ (b) $K = 3, p = 0.5$



(c) $K = 2, p = 0.24$ (d) $K = 3, p = 0.3$

Figure 4.4.: Balanced network with 100 nodes. Histograms of estimated number of communities across 100 replicas. The top panel is the case when the community structure in the network is prominent $(p = 0.5)$; the lower panel is for a wague structure. For each couple of histograms: from left to right Louvain algorithm, MFM-SBM method.

4.1.4. Comparison between Poisson and Gnedin Priors for K

In this part we investigate the difference in terms of performance of the MFM-SBM model with different priors on K parametrized in different ways on different situations (i.e. with different type of networks, probabilities of linkage between nodes of the same community etc.). In particular we choose to test a Gnedin prior parametrized with $\gamma_{gn} = \{0.1, 0.3, 0.5, 0.7, 0.9\}$ and a zero-truncated-Poisson prior parametrized with $\lambda = \{1, 3, 5, 7, 9\}$.

Regarding the situations in which we tested the different MFM-SBMs, it is clear that there were many parameters to play with. For example: the number of nodes of the network, the type of network (i.e. with balanced or unbalanced community structure), the probabilities p and q, the number of iterations of the Gibbs sampler, the relative burn-in, and finally the starting configuration of the algorithm. Therefore, for a matter of time, we decided to vary only the parameters that seemed most relevant to us and we fixed all the others at reasonable values. Accordingly, before presenting the results we obtained we describe how we set the simulation study.

Setting and Methodology

Since we are dealing with a simulation study, following the same approach of Geng et al. (2019), we decided to implement a procedure capable of taking into account the randomness of the generating data process. For this reason, for each particular setting of the two algorithms, the simulation scheme consists in generating M adjacency matrices and in applying the algorithms with that particular setting to them. This allowed us to get an idea of the fluctuation of the results due to the random nature of $\mathbf{Y}$.

The quantities involved in this simulation study are then the following: the network size *n*; the true number of communities K; the initial configuration of the community structure and its relative dimensionality, respectively $\mathbf{z}_{start}$ and K_{start}, where $|\mathbf{z}_{start}| = K_{start}$; the number of iterations of the Gibbs sampler, M and the relative burn-in, BI; the connection probability between nodes of the same group, p; the connection probability between nodes of different groups, q; the number of time we created a new adjacency matrix $\mathbf{Y}$ for each particular setting of the model, N; the parameter for the zero-truncated-Poisson prior, λ; the parameter for the Gnedin prior, γ_{gn} and finally the parameters for the Beta prior on the elements of the matrix $\mathbf{Q}$ and for the simmetric Dirichlet distriburion, respectively (a, b) and γ that we fixed to 1 as discussed at the beginning of the chapter.

We choose to test only few different scenarios by fixing most of the aforementioned quantities to specific values which can be found in Table 4.2.

Table 4.2.: Scenarios setting for the simulation study

Scenario	*n*	*K*	*N*	*M*	*BI*	*Type*	K_{start}	*p*	*q*
1	100	2	100	600	300	Unbalanced	9	0.5	0.1
2	100	2	100	600	300	Unbalanced	9	0.24	0.1
3	100	5	100	600	300	Unbalanced	9	0.5	0.1
4	100	5	100	600	300	Unbalanced	9	0.24	0.1
5	200	2	100	600	300	Unbalanced	9	0.5	0.1
6	200	2	100	600	300	Unbalanced	9	0.24	0.1
7	200	5	100	600	300	Unbalanced	9	0.5	0.1
8	200	5	100	600	300	Unbalanced	9	0.24	0.1

Regarding the initialization of the community structure, $\mathbf{z}_{start}$, for both $n = 100$ and $n = 200$ we randomly allocated the observation in K_{start} communities and we used that particular $\mathbf{z}_{start}$ for scenarios 1-4 and scenarios 5-8 respectively.

As it can be seen from Table 4.2 we choose to generate only unbalanced adjacency matrices $\mathbf{Y}$, this choice is motivated by the fact that we believe that unbalanced networks are much more common than balanced ones in real situation. Regarding the number of iterations of the Gibbs sampler and the burn-in, we set them at 600 and

300 respectively, approximately twice the number chosen by Geng et al. (2019) for their simulation study. We fixed also the starting number of communities to nine and the connection probability between nodes belonging to different communities to 0.1, in accordance to Geng et al. (2019). What we did vary instead was the size of the network, the number of communities and the probability of connection between nodes belonging to the same community. Namely, we tested $n \in \{100, 200\}$, $K \in \{2, 5\}$ and $p \in \{0.5, 0.24\}$.

Regarding the priors for K, as mentioned before, we decided to test $\lambda \in \{1, 3, 5, 7, 9\}$ and $\gamma_{gn} \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ respectively for the zero-truncated-Poisson distribution (see Figure 4.5) and for the Gnedin distribution (see Figure 4.6). For both distributions we decided to test only realistic alternative values for the parameters with respect to the true generative model of the network, which in our case was with $K = 2$ or $K = 5$. For example, in this situation, testing the algorithm with a zero-truncated-Poisson prior with $\lambda = 20$ was clearly not a realistic setting.

The simulation scheme then is presented in Algorithm 3.

Algorithm 3: Simulation scheme. Pseudo-code

Result: Compare the performance of MFM-SBM model with different priors on K for a given scenario of Table 4.2.

Initialize n, N, M, K, K_{start}, p, q, a, b, γ, BI;

Set $\mathbf{z}_{start}$ with K_{start} communities;

for *iter in 1 to N* **do**

  Create $\mathbf{Y}$ and $\mathbf{z}_0$ according to the parameter initialization and the data generating process of 4.1.1;

  for *i in* $\{1, 3, 5, 7, 9\}$ **do**

    Run the Gibbs sampler of Algorithm 1 starting from $\mathbf{z}_{start}$ and take M samples where $V_n(\cdot)$ is computed using Algorithm 2 with $\lambda = i$;

    Save the execution time and discard the burn-in $= BI$;

    Save $\mathbb{P}(|\mathbf{z}| = K|\mathbf{Y})$;

    Save $\hat{K}$ where is obtained as the mode of the posterior of $|\mathbf{z}|$;

    Compute through (3.26) $\hat{\mathbf{z}}$ and save RI$(\hat{\mathbf{z}}, \mathbf{z}_0)$, VI$(\hat{\mathbf{z}}, \mathbf{z}_0)$;

  end

  for *j in* $\{0.1, 0.3, 0.5, 0.7, 0.9\}$ **do**

    Run the Gibbs sampler of Algorithm 1 starting from $\mathbf{z}_{start}$ and take M samples where $V_n(\cdot)$ is computed using the closed form expression (3.23) with $\gamma_{gn} = j$;

    Save the execution time and discard the burn-in $= BI$;

    Save $\mathbb{P}(|\mathbf{z}| = K|\mathbf{Y})$;

    Save $\hat{K}$ where is obtained as the mode of the posterior of $|\mathbf{z}|$;

    Compute through (3.26) $\hat{\mathbf{z}}$ and save RI$(\hat{\mathbf{z}}, \mathbf{z}_0)$, VI$(\hat{\mathbf{z}}, \mathbf{z}_0)$;

  end

end

As we can see from Algorithm 3, we first generate an adjacency matrix $\mathbf{Y}$ with the respective community structure $\mathbf{z}_0$ and then we test the two algorithms with the different parametrization always on that same adjacency matrix. Moreover, for a given scenario we run the Gibbs sampler always from the same starting condition $\mathbf{z}_{start}$ in order to minimise the bias due to different initializations of the algorithm. In this simulation study we investigated: (i) the estimation of K, $\hat{K}$, obtained as the mode of the posterior distribution of $|\mathbf{z}|$, (ii) the execution time of the two algorithms, since the one with Gnedin prior uses the exact formula for $V_n(t)$ avoiding thus all the approximation procedure of Algorithm 2, (iii) the posterior probability that $|\mathbf{z}| = K$, $\mathbb{P}(\hat{K} = K|\mathbf{Y})$, computed as the average of time in which $|\mathbf{z}| = K$ in the sample, and finally (iv) we computed the point estimation of the community structure, $\hat{\mathbf{z}}$ through (3.24) and we computed the Rand Index and the Variation Information between $\hat{\mathbf{z}}$ and $\mathbf{z}_0$.

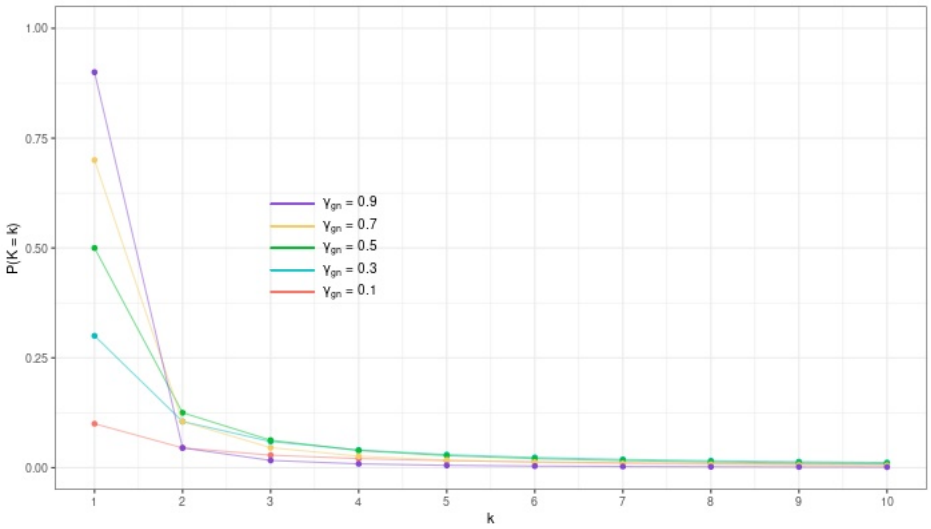


Figure 4.5.: Gnedin distribution parametrized with $\gamma_{gn} \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$.

4.1.5. Results

In this part we only report a summary of the measures we computed. For every MFM-SBM model tested on any scenario of Table 4.2 we get, for each measure of interest, a N-dimensional vector of results. Each entrance corresponding to the result obtained on a particular $\mathbf{Y}$ and $\mathbf{z}_0$. In particular, we denote by $\hat{K}_N$, Time_N, $\mathbb{P}(\hat{K} = K|\mathbf{Y})_N$, $\text{RI}(\hat{\mathbf{z}}, \mathbf{z}_0)_N$ and $\text{VI}(\hat{\mathbf{z}}, \mathbf{z}_0)_N$, respectively the vector containing, for each generation of the data, the estimation of K, the execution time of the MFM-SBM algorithm, $\mathbb{P}(\hat{K} = K|\mathbf{Y})$, the Rand Index computed between the point estimation $\hat{\mathbf{z}}$ and $\mathbf{z}_0$ and the Variation Information computed again between the point estimation $\hat{\mathbf{z}}$ and $\mathbf{z}_0$.

For a complete overview of the results obtained we refer to Appendix A, while here we show Table 4.3 and Table 4.4 which contain for each scenario and each parametrization

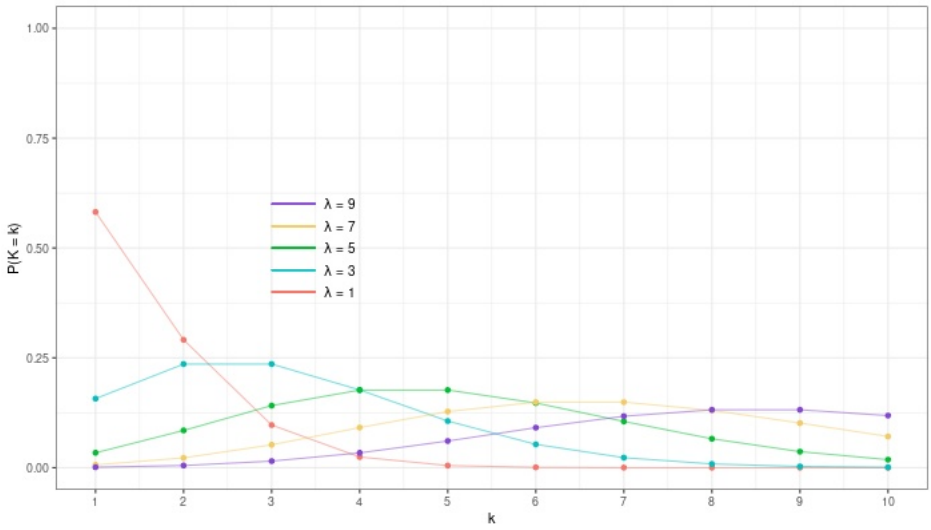


Figure 4.6.: 0-truncated-Poisson distribution parametrized with $\lambda \in \{1, 3, 5, 7, 9\}$.

of the two algorithms: (a) the mode of $\hat{K}_N$, (b) the mean of Time_N (c) the mean of $\mathbb{P}(\hat{K} = K|\mathbf{Y})_N$, (d) the mean of $\text{RI}(\hat{\mathbf{z}}, \mathbf{z}_0)_N$ and (e) the mean of $\text{VI}(\hat{\mathbf{z}}, \mathbf{z}_0)_N$.

Specifically, in Table 4.3 and Table 4.4 are shown respectively the scenarios 1-4 in which $n = 100$ and the scenarios 5-8 where $n = 200$. From the results we have obtained we can see some interesting behaviour of the two algorithms.

Impact of the Size of the Network in Recovering K and z

The first thing that stands out is how an increase in K implies the need to have a larger network size to allow the algorithms to correctly derive the number of communities. Indeed, if we look at Scenarios 3 and 4 of Table 4.3, where $K = 5$, we can see clearly a drop of performance of the two algorithms in estimating K with respect to Scenarios 1 and 2, where instead $K = 2$. This is mentioned also in Geng et al. (2019) in which they say that in order to allow the MFM-SBM model to detect correctly the number of communities when $K > 2$, it is necessary to have an appropriate size of network $(n > 100)$. The difference between the scenarios in which $K = 2$ and $K = 5$ in terms of performance in recovering the true K by the two algorithms, drops significantly if the network size, n, is increased from 100 to 200 as we can see in Table 4.4 (expecially between Scenarios 5-6 and Scenario 7, where the community structure is prominent, i.e. $p = 0.5$).

Note, however, that an incorrect estimate of K does not imply a disastrous estimate of the community structure of the network when it has a *prominent* community structure (i.e. $p = 0.5$). See for example Scenario 3, where the algorithms systematically misestimate K but still manage to identify a network configuration $\hat{\mathbf{z}}$ that is fairly close to $\mathbf{z}_0$ both in terms of Rand Index and Variation Information. On the other hand, with a *vague* community structure (i.e. $p = 0.24$), we have a general drop in the performance of the two algorithms, although the drop is much more significant when

Table 4.3.: simulation results for scenarios 1-4 of Table 4.2.

Scenario	Measure	Gnedin Prior					Poisson Prior				
		γ_{gn} :	.1	.3	.5	.7	.9	λ :	1	3	5	7	9
Scenario 1	$\hat{K}$		2	2	2	2	2		2	2	2	2	2
	Time (secs)		16.37	16.80	16.64	17.08	16.91		15.79	16.74	17.64	18.92	19.79
	$\mathbb{P}(\hat{K} = K\|\mathbf{Y})$		0.93	0.86	0.88	0.85	0.87		0.84	0.73	0.58	0.52	0.44
	RI$(\hat{\mathbf{z}}, \mathbf{z}_0)$		1.00	1.00	1.00	0.99	1.00		1.00	1.00	1.00	1.00	1.00
	VI$(\hat{\mathbf{z}}, \mathbf{z}_0)$		0.01	0.01	0.00	0.02	0.01		0.00	0.00	0.01	0.01	0.01
Scenario 2	$\hat{K}$		2	2	2	2	2		2	2	2	2	2
	Time (secs)		16.48	16.99	17.50	17.52	17.43		16.96	17.44	18.81	19.40	19.99
	$\mathbb{P}(\hat{K} = K\|\mathbf{Y})$		0.88	0.86	0.89	0.80	0.82		0.79	0.64	0.54	0.43	0.38
	RI$(\hat{\mathbf{z}}, \mathbf{z}_0)$		0.93	0.93	0.93	0.92	0.93		0.92	0.93	0.92	0.92	0.92
	VI$(\hat{\mathbf{z}}, \mathbf{z}_0)$		0.26	0.26	0.26	0.28	0.26		0.28	0.26	0.29	0.28	0.29
Scenario 3	$\hat{K}$		4	4	4	4	4		4	4	4	4	4
	Time (secs)		24.07	25.49	26.42	27.05	28.48		26.26	26.24	25.89	25.50	26.58
	$\mathbb{P}(\hat{K} = K\|\mathbf{Y})$		0.30	0.27	0.24	0.26	0.26		0.20	0.27	0.30	0.31	0.33
	RI$(\hat{\mathbf{z}}, \mathbf{z}_0)$		0.94	0.94	0.94	0.94	0.94		0.93	0.94	0.94	0.94	0.95
	VI$(\hat{\mathbf{z}}, \mathbf{z}_0)$		0.28	0.30	0.31	0.30	0.29		0.33	0.29	0.29	0.30	0.27
Scenario 4	$\hat{K}$		2	2	2	2	2		2	2	2	2	2
	Time (secs)		16.21	16.27	16.77	16.91	17.19		16.31	17.37	18.87	19.68	20.13
	$\mathbb{P}(\hat{K} = K\|\mathbf{Y})$		0	0	0	0	0		0.00	0.01	0.01	0.02	0.04
	RI$(\hat{\mathbf{z}}, \mathbf{z}_0)$		0.36	0.35	0.36	0.35	0.35		0.36	0.35	0.35	0.36	0.36
	VI$(\hat{\mathbf{z}}, \mathbf{z}_0)$		1.51	1.50	1.50	1.51	1.52		1.52	1.51	1.52	1.53	1.53

Table 4.4.: simulation results for scenarios 5-8 of Table 4.2.

Scenario	Measure	Gnedin Prior					Poisson Prior				
	γ_{gn} :	.1	.3	.5	.7	.9	λ : 1	3	5	7	9
Scenario 5	$\hat{K}$	2	2	2	2	2	2	2	2	2	2
	Time (secs)	36.72	35.29	41.10	36.67	36.82	40.96	36.41	36.12	35.62	30.56
	$\mathbb{P}(\hat{K} = K\|\mathbf{Y})$	0.89	0.89	0.88	0.93	0.89	0.90	0.83	0.76	0.73	0.67
	RI$(\hat{\mathbf{z}}, \mathbf{z}_0)$	0.9982	0.9978	0.9998	0.9996	0.9963	0.9967	0.9996	0.9968	0.9966	0.9995
	VI$(\hat{\mathbf{z}}, \mathbf{z}_0)$	0.0046	0.0070	0.0007	0.0016	0.0097	0.0081	0.0015	0.0092	0.0095	0.0026
Scenario 6	$\hat{K}$	2	2	2	2	2	2	2	2	2	2
	Time (secs)	35.08	37.56	34.30	33.07	33.28	33.64	32.66	34.12	29.84	30.59
	$\mathbb{P}(\hat{K} = K\|\mathbf{Y})$	0.89	0.89	0.82	0.86	0.81	0.84	0.80	0.75	0.73	0.66
	RI$(\hat{\mathbf{z}}, \mathbf{z}_0)$	0.9910	0.9914	0.9898	0.9914	0.9823	0.9919	0.9895	0.9910	0.9902	0.9881
	VI$(\hat{\mathbf{z}}, \mathbf{z}_0)$	0.0432	0.0441	0.0437	0.0426	0.0672	0.0415	0.0469	0.0450	0.0488	0.0519
Scenario 7	$\hat{K}$	5	5	5	5	5	4	5	5	5	5
	Time (min)	1.75	1.72	1.72	1.76	1.71	1.67	1.70	1.70	1.75	1.76
	$\mathbb{P}(\hat{K} = K\|\mathbf{Y})$	0.55	0.47	0.47	0.48	0.47	0.44	0.52	0.49	0.49	0.53
	RI$(\hat{\mathbf{z}}, \mathbf{z}_0)$	0.9755	0.9731	0.9680	0.9749	0.9675	0.9711	0.9746	0.9712	0.9737	0.9745
	VI$(\hat{\mathbf{z}}, \mathbf{z}_0)$	0.1133	0.1267	0.1407	0.1148	0.1431	0.1319	0.1196	0.1309	0.1189	0.1167
Scenario 8	$\hat{K}$	2	2	2	2	2	2	2	2	2	2
	Time (secs)	20.80	21.92	25.75	18.06	18.58	24.93	22.19	18.60	17.05	15.39
	$\mathbb{P}(\hat{K} = K\|\mathbf{Y})$	0.03	0.02	0.02	0.03	0.01	0.01	0.01	0.04	0.03	0.04
	RI$(\hat{\mathbf{z}}, \mathbf{z}_0)$	0.6982	0.6961	0.6943	0.6940	0.6979	0.6922	0.6934	0.6970	0.6969	0.6991
	VI$(\hat{\mathbf{z}}, \mathbf{z}_0)$	1.1936	1.1973	1.1931	1.1921	1.1976	1.1906	1.1864	1.2031	1.1968	1.1929

n is low and K is high. See in particular Scenarios 4 and 8. The drop of performance respectively to their prominent counterparts (Scenarios 3 and 7) is remarkable, both in terms of $\hat{K}$ and $\hat{\mathbf{z}}$.

Execution Time of the Algorithms

Regarding the execution time of the two algorithms we can see that the MFM-SBM model with a Gnedin prior on K is always slightly faster than the MFM-SBM model with a zero-truncated-Poisson prior. We can also see how the different nature of the two priors affects the execution time. In fact, if we look at each scenario, we can see how the time of the algorithm with Gnedin prior does not suffer major changes depending on the parameterization. The same cannot be said for the algorithm with zero-truncated-Poisson prior where, if $K = 2$, as λ increases there is an increase in the execution time of the algorithm and vice versa if $K = 5$. This is due to the effect the parameterization has on the shape of the two tested distributions as we can see in Figure 4.5 and Figure 4.6.

Changing values for γ_{gn} in the Gnedin distribution, indeed, has the effect of shifting mass mainly to $k = 1$ and $k = 2$ without significantly varying the mass present on subsequent values for k, whereas, changing the value of λ for the zero-truncated-Poisson distribution has the effect of changing both the mean and the variance of the distribution. For this reason, the probability mass that is moved when λ changes is much greater than that moved in the Gnedin distribution when γ_{gn} changes.

This has an influence on the computation of the coefficient $V_n(t)$ for the MFM-SBM model with zero-truncated-Poisson prior on K. In fact, moving more mass towards higher values of k results in higher values of $p_K(k)$ and thus, in general, more iterations of Algorithm 2 for approximating $V_n(t)$ are needed. As pointed out by Miller and Harrison (2018), the convergence of the infinite series for $V_n(t)$ (Equation (2.24)) converges at least as rapidly as the series $\sum_{k=1}^{\infty} p_K(k)$ converges to 1. Therefore, shifting probability mass to higher values for k has the possible effect of slowing down the convergence speed of $V_n(t)$. This automaticaly has side-effects in approximating $V_n(t)$ to a designated level of precision. Namely, more iterations could be required in order to find the next element of the sum smaller than the fixed treshold.

With the MFM-SBM model with a Gnedin prior on K we do not have this problem, since we do not use Algorithm 2 for approximating $V_n(t)$ but rahter, we use the exact formula (3.23).

Finally, regardless of the type of prior used, we can see that both an increase in K and n causes a significant increase in the execution time required by the algorithm. In fact, an increment of n translates in a longer second `for` cycle of Algorithm 1, while an increment of K results in a bigger matrix $\mathbf{Q}$ and hence the MFM-SBM model spends more time updating matrix $\mathbf{Q}$ through (3.14) in the first `for` cycle of Algorithm 1.

Differences Between the Two Algorithms in Deriving the Community Structure

The discussion made for the running time of the two algorithms might suggest that the difference that the parameterizations have on the two distributions might also have affected the estimation of the community structure. In reality, this did not happen. In scenarios where it was clear that convergence of the gibbs sampler was achieved (i.e. when n and K were reasonably balanced) the point estimate of the community structure, $\hat{\mathbf{z}}$ did not suffer in terms of consistence relative to $\mathbf{z}_0$ from the variation of the values assigned to the parameter for both priors. This can be seen from the values of both the Rand Index and Variation Information between $\hat{\mathbf{z}}$ and $\mathbf{z}_0$. This is also reflected in the fact that there is no substantial difference between the two algorithms in terms of the goodness of the estimate of $\mathbf{z}$, $\hat{\mathbf{z}}$.

In borderline scenarios with respect to the values chosen for n and K (expecially scenarios 4 and 8) all the MFM-SBM models have returned inconsistent estimates $\hat{\mathbf{z}}$ with respect to $\mathbf{z}_0$, all with the same degree of inconsistency. This could also be due to a greater number of iterations being required for the Gibbs-sampler to converge.

4.2. Community Detection in Dolphin Social Network Data

4.2.1. Dataset and Methodology

In this section we tested the MFM-SBM model on a benchmarck dataset largely used in literature: the Dolphin social network dataset. This network is obtained from a study on a community of 62 bottlenose dolphins over a period of seven years from 1994 to 2001. The aim was to describe the social organisation of a population of bottlenose dolphins living in a fjord in New Zealand, a deep coastal environment, geographically isolated, located at the southern extreme of the species' range. The nodes in the network represent the dolphin, and ties between nodes represent associations between dolphin pairs occurring more often than by random chance. The term association refers to the number of times the two dolphins have been observed together. If this number appeared to be greater than a randomly obtainable number, then an association was assigned between the two (see Lusseau et al. (2003) for more details). A reference clustering of this undirected network with 62 nodes, that we denote by $\mathbf{z}_0$, is shown in Figure 4.7 (refer to Lusseau and Newman (2004)). Despite this network has several sub-communities, as pointed out by Geng et al. (2019), based on gender, age and other demographic characteristics, we are interested in recovering the principal division into two communities as indicated by the light blue and coral vertices of Figure 4.7 just from the adjacency matrix.

We applied the MFM-SBM model (3.13) with both zero-truncated-Poisson prior and Gnedin prior on K parametrized respectively with $\lambda = 1$ and $\gamma_{gn} = 0.1$. We

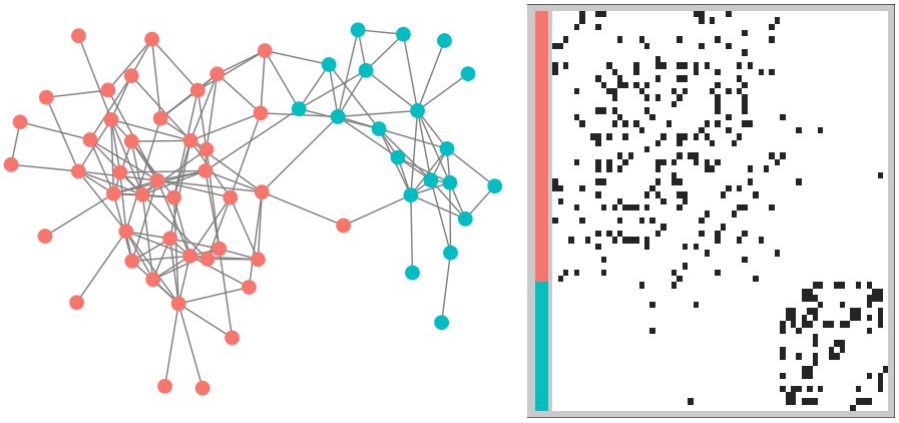


Figure 4.7.: Reference configuration for the dolphin social network data. Left panel: vertex color indicates community membership. Right panel: adjacency matrix of the network. Colors on the side correspond to the true communities.

choose a conservative parametrization for both priors. We ran 20000 MCMC iteration leaving out a burn-in of 10000, initialized at randomly generated configuration with nine clusters. The elements of the probability matrix $\mathbf{Q}$ were assigned independent $Beta(1,1)$ priors while the γ's of the Dirichlet distribution were set equal to 1.

4.2.2. Poisson Prior

For the zero-truncated-Poisson prior with $\lambda = 1$ we can see from Figure 4.8 that the MFM-SBM does not fully recover the correct community configuration and fails into recover the number of K. However, $\hat{K} = 3$ (where K is estimated with the mode of the posterior distribution of $|\mathbf{z}|$) which is not far from the real $K = 2$. Looking at Figure 4.8 we can see how the algorithm separates the nodes which presents no more than three edges with other nodes typically classified in a different community. It is quite evident then how the algorithm in this case classifies as a separate cluster the core-periphery component of this network. However this community detection is not correct which is also confirmed from the Rand Index and Variation Information obtained with respect to the $\hat{\mathbf{z}}$ of (3.24) and the reference configuration, $\mathbf{z}_0$. We get $\mathrm{VI}(\hat{\mathbf{z}}, \mathbf{z}_0) = 0.73$ and $\mathrm{RI}(\hat{\mathbf{z}}, \mathbf{z}_0) = 0.74$ which are respectively quite far from zero and from one.

4.2.3. Gnedin Prior

Using the MFM-SBM model with a Gnedin prior with $\gamma_{gn} = 0.1$ we can see from Figure 4.9 that the situation is different than the case with the Poisson prior. The algorithm indeed recovers correctly the number $K = 2$ and we get a community configuration close from that of reference. In particular we get $\mathrm{VI}(\hat{\mathbf{z}}, \mathbf{z}_0) = 0.14$ and $\mathrm{RI}(\hat{\mathbf{z}}, \mathbf{z}_0) = 0.97$ which are reasonably close to zero and to one respectively.

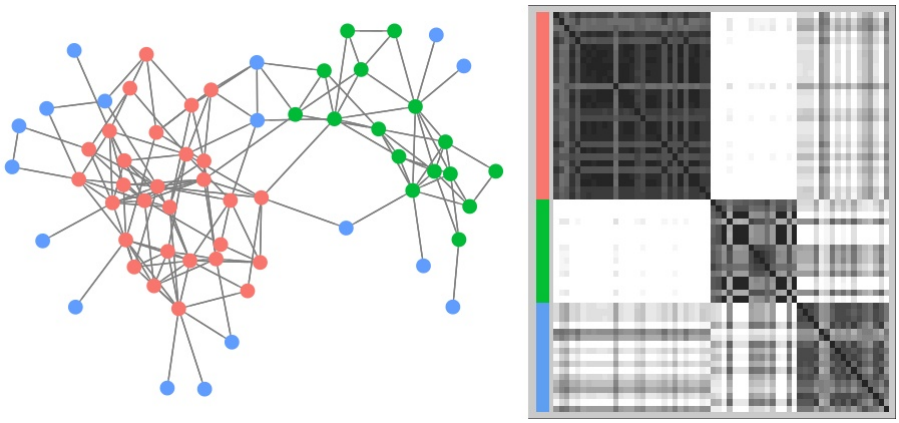


Figure 4.8.: estimated configuration for the dolphin network usign MFM-SBM with zero-truncated-Poisson prior with $\lambda = 1$. Left panel: vertex color indicates community membership. Right panel: heatmap of the similarity matrix $\hat{\mathbf{Q}}$ of the estimated configuration $\hat{\mathbf{z}}$. Colors on the side correspond to the estimated communities.

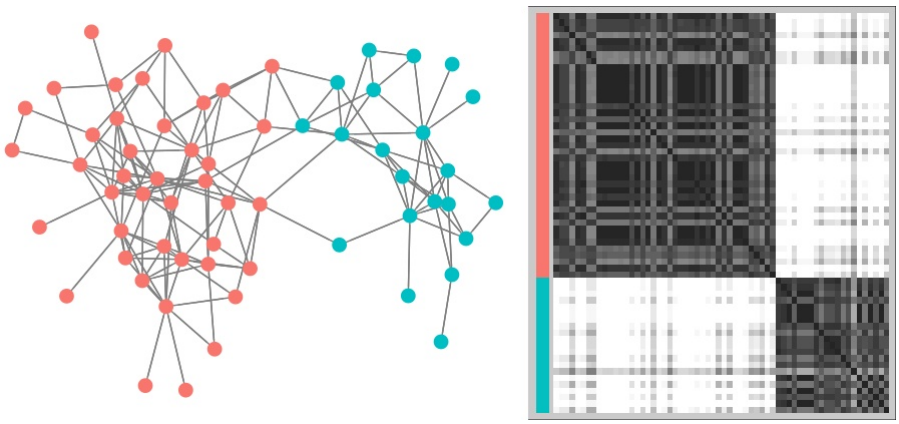


Figure 4.9.: estimated configuration for the dolphin network usign MFM-SBM with Gnedin prior with $\gamma_{gn} = 0.1$. Left panel: vertex color indicates community membership. Right panel: heatmap of the similarity matrix $\hat{\mathbf{Q}}$ of the estimated configuration $\hat{\mathbf{z}}$. Colors on the side correspond to the estimated communities.

4.3. Discussion

With a real world dataset we saw a more evident difference of the same method with the two different priors on the number $K, p_K(\cdot)$, than in the simulation study. The dolphin social network dataset indeed is not obtained from a SBM generative model and thus presents a community structure that might be more complicated to detect as shown in the zero-truncated-Poisson scenario. Also in this case, as in the simulation study, the key difference between the two MFM-SBM models is in the different shape of the prior choosen for K.

As we mentioned before, we choose for both priors a reasonable parametrization. In the case of the zero-truncated-Poisson prior we set the lower possible value for λ, putting all the mass of the distribution around values between 1 and 5 while, setting $\gamma_{gn} = 0.1$ for the Gnedin distribution we choose to set a non informative prior on K in order to affect the posterior distribution as little as possible. We initialized the SBM-MFM algorithm with nine clusters accordingly to Geng et al. (2019), so it was important that the prior did not put too much mass around high values of k. If we look at the distribuition of both priors, we can see that the Gnedin distribution has heavier tails than the zero-truncated-Poisson distribution, and this is the reason why the Gnedin distribution has infinite expectation, while the zero-truncated-Poisson distribution has finite expectation. However, the difference in probability mass in the tails is minimal compared to the difference of the two distributions in small values of k. In particular, looking at the light blue line in Figure 4.10 we can see that, despite $\lambda = 1$ we still have a resonable amount of probability mass on $k = 3$. On the other hand, with a Gnedin distribution with $\gamma_{gn} = 0.1$ we have a much flatter distribution than the Poisson distribution, with still a little more mass in the initial values for k but not comparable to that of the zero-truncated-Poisson. This allows the algorithm to more effectively identify the correct number of k, having much less mass in $k = 3$ than the Poisson distribution.

To support this thesis, we tried to run the MFM-SBM algorithm with a Gnedin prior parameterised with $\gamma_{gn} = 0.5$ (see Figure 4.10, green line) obtaining the community detection showed in Figure 4.11. This configuration is now more similar to the one obtained with the zero-truncated-Poisson prior with the addition of a fourth small community leading thus to $\hat{K} = 4$ (where, again $\hat{K}$ is computed through the mode of the posterior distribution of $|\mathbf{z}|$). In fact, if we look at Figure 4.10 we can see a higher probability on $K = 4$ for the Gnedin distribution with $\gamma_{gn} = 0.3$ than the other two tested distributions. In this scenario we obtained VI $= 0.76$ and RI $= 0.69$ computed with respect to $\hat{\mathbf{z}}$ (obtained through (3.24)) and the reference configuration, $\mathbf{z}_0$.

On the basis of what we have seen with the dolphin social network data, it should be noted how a conscious choice of the prior and of its parameters can make a difference in terms of algorithm performance in the identification of communities in a network of medium size. The results we achieved were all more or less satisfactory, considering the real nature of the network. None of them turned out to be completely wrong,

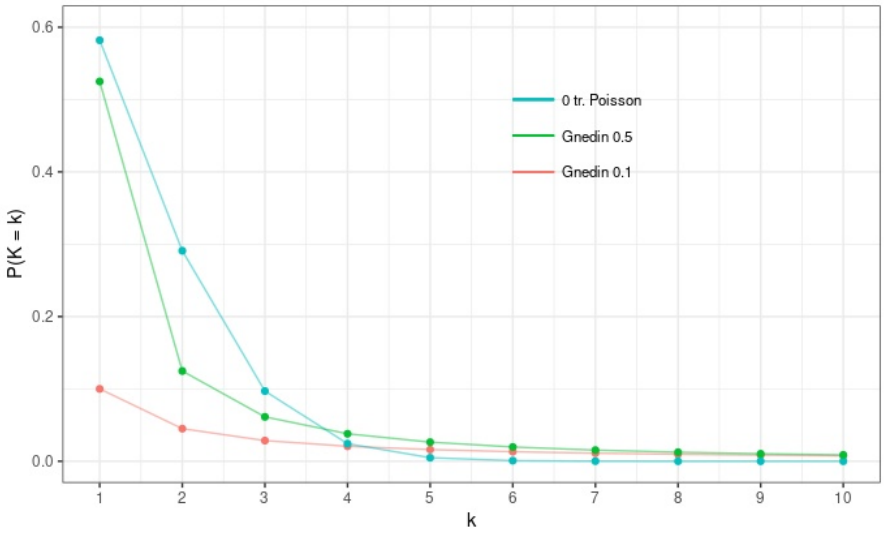


Figure 4.10.: in coral: Gnedin distribution with $\gamma_{gn} = 0.1$. In green: Gnedin distribution with $\gamma_{gn} = 0.5$. In light blue: zero-truncated-Poisson distribution with $\lambda = 1$.

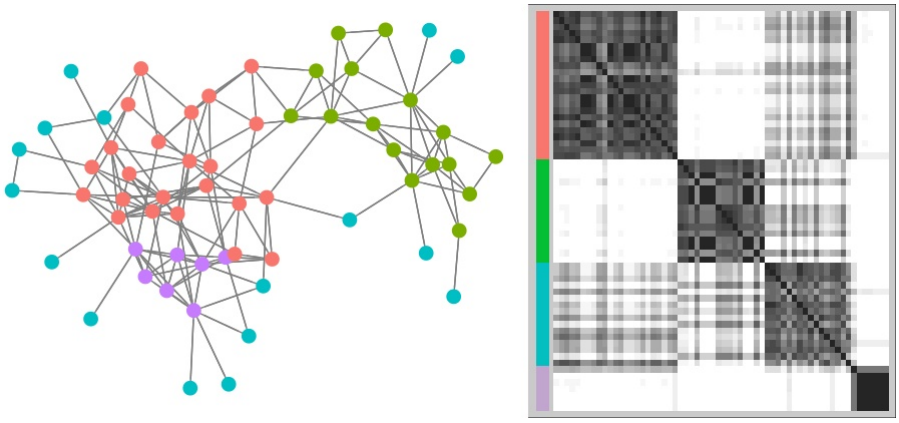


Figure 4.11.: estimated configuration for the dolphin network usign MFM-SBM with Gnedin prior with $\gamma_{gn} = 0.5$. Left panel: vertex color indicates community membership. Right panel: heatmap of the similarity matrix $\hat{\mathbf{Q}}$ of the estimated configuration $\hat{\mathbf{z}}$. Colors on the side correspond to the estimated communities.

and this is also confirmed by the values we obtained for both the Rand Index and the Variation Information. All results indeed have proved to be better than the ones achieved under the Louvain algorithm. As we can see from Figure 4.12, the Louvain method estimates K with $\hat{K} = 5$ and the community structure it achieves, $\hat{\mathbf{z}}$, is the worst we obtained. Indeed $\mathrm{RI}(\hat{\mathbf{z}}, \mathbf{z}_0) = 0.64$ and $\mathrm{VI}(\hat{\mathbf{z}}, \mathbf{z}_0) = 1.06$.

However, we observed some differences obtained trough MFM-SBM with the two different priors. It should be noted that the dolphin social network has few observations. Therefore, the prior distribution chosen has an influence on the result. The study we conducted showed that a non-informative prior on K should be preferred so that the data can "speak for itself". To this end, the zero-truncated-Poisson distribution does not allow to get a totally flat shape on its domain by a specific setting for λ as, on the other hand, the Gnedin priority allows. So, if it is used as a prior, for each λ value it gives precise information to the model which will be incorporated into the posterior. This was also evident from the fact that, by parameterizing the Gnedin Prior in an informative manner, we obtained a worsening of the results.

Finally, it must be said that, in analysing the performance of the two algorithms, we decided to double the number of iterations of the Gibbs sampler and consequently also of the burn-in compared to that indicated by Geng et al. (2019) in section 6, in order to ensure that the Gibbs-sampler converged. In spite of this decision, we observed the aforementioned differences which in our opinion should deserve a special attention and a further exploration in future studies.

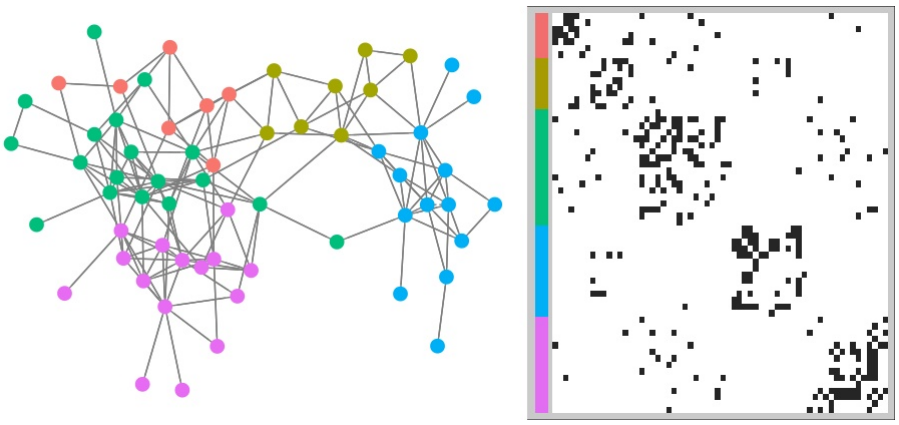


Figure 4.12.: estimated configuration for the dolphin network usign the Louvain algorithm. Left panel: vertex color indicates community membership. Right panel: heatmap of the adjacency matrix ordered according to the configuration $\hat{\mathbf{z}}$. Colors on the side correspond to the estimated communities.

5. Conclusions

5.1. Summary

Nowadays, networks are increasingly common compelling the development of methods to study their complexity. A relevant research field is *community detection* which is useful to extract the underlying structure from complex networks and to understand their patterns. In literature we can find two macro areas where methods used to tackle such a problem are divided: *the algorithmic* and the *model-based* one. A common problem with many methods of both classes is that they require in advance the number of communities to split the units into, making them unattractive for practical purposes. Among the algorithmic methods, i.e. not based on statistical models, the most widely used to estimate both the number of communities and the communities structure is the *Louvain algorithm*. This is a fast algorithm but, like most of them, has limitations on the type of communities it can find and on its theoretical foundations. These issues have motivated a growing interest in *model-based* solutions which rely on generative statistical models. Among the generative models for learning communities in network data, *the stochastic block model (SBM)* is arguably the most widely implemented and well-established formulation, owing also to its unique balance between simplicity and flexibility. However, classical SBM formulation is based on a fixed and pre-specified number of communities which has led to the flourishing of various generalizations thereof.

Through our research we investigated the *mixture of finite mixtures* version of the stochastic blockmodel (*MFM-SBM model*) introduced by Geng et al. (2019). This model is based on a mixture of finite mixtures model which admits a *Pólya urn scheme* similar to CRP. We implemented in R the collapsed Gibbs sampler formulated by Geng et al. (2019) and we studied its behaviour. MFM-SBM model turned out to perform better than the Louvain algorithm both in estimating the community structure and the number of communities itself. We carried out also a simulation study where we compared two versions of the MFM-SBM model, respectively one with a zero-truncated-Poisson prior on the parameter K and the other one with a Gnedin prior on K, where K denotes the number of communities. We wanted to see if there were noteworthy differences in terms of performance of the two algorithms, especially due to the possibility to write exactly the coefficient $V_n(t)$ in Algorithm 1 under Gnedin Prior. The overall performance of the two algorithms was comparable with the only difference that the MFM-SBM model with zero-truncated-Poisson prior was more sensible in terms of execution time to different parametrization. Indeed as opposed to the MFM-

SBM model with Gnedin prior, we saw a difference in the time spent as the parameter of the zero-truncated-Poisson prior value changed. The reason lies on the different role of the parameter on the two distributions, indeed, changing the value of the parameter of the zero-truncated-Poisson prior has the effect of moving more probability mass compared to the Gendin distribution.

In the simulation study we tested also various generative models. We created networks with different sizes, different number of communities and also with different connection probabilities for a total of eight different scenarios content in Table 4.2. We decided to test different values only for the main features of the generative model fixing the others to default values. As expected, we noticed an increment of the execution time of the MFM-SBM model (with both priors) as the size of the network increased, while we got confirmation that with a medium size of the network (i.e. $n \sim 100$) and $K > 2$, the algorithm fails to recover the true number of communities from the posterior distribution of $|\mathbf{z}|$, where $|\mathbf{z}|$ refers to the cardinality of the vector containing the community association for each node. This supports what we mentioned in section 4.1.2 where we showed that, for an insufficiently large size of the network compared to the number of communities, it should be preferred the approach of Miller and Harrison (2018) in estimating K. An incorrect estimation of K, however, was not always associated to a poor community structure estimation, $\hat{\mathbf{z}}$. Indeed, when the community structure in the network was *prominent* (i.e. $p = 0.5$) we got always a $\hat{\mathbf{z}}$ not too far from the true community structure $\mathbf{z}_0$. Among those tested, the most relevant factor was the size of the network, n. Indeed, in a network with $n = 100$ we observed an evident sensitivity of $\hat{\mathbf{z}}$ to different values for the true number of communities K and for the community structure's nature. On the other hand, setting $n = 200$ we obtained satisfactory and robust results with respect to the different values for each factor mentioned above. The only case where the results proved to be inaccurate with $n = 200$ was with $K = 5$ and $p = 0.24$ suggesting the necessity of increasing n in order to achieve adequate results.

Finally, we evaluated the MFM-SBM model on the *dolphin social network* data, typically used in literature as a benchmark network where is recognized to have two main communities. Compared to the Louvain algorithm, the MFM-SBM method proved to perform better either with a zero-truncated-Poisson prior or the Gnedin prior. Regarding the two different priors for K used in the MFM-SBM on the dolphin social network we noted that using a Gnedin distribution with a non informative parametrization allowed to get an improvement that the zero-truncated-Poisson prior.

Summarizing, through our work we studied the MFM-SBM model applied to community detection showing its convenience respect to classical algorithmic methods such as the Louvain algorithm. Regarding the comparison of the two prior tested on K, respectively the zero-truncated-Poisson distribution and the Gnedin distribution, we did not notice any significant difference in terms of $\hat{K}$ and $\hat{\mathbf{z}}$. However, we saw a difference on a real network where the Gnedin distribution seemed to work better. According to us, this is due to the different effect the parameters have on the corre-

sponding distribution as disccussed in section 4.3. Nevertheless, despite our findings we emphasize the necessity to investigate this possible difference further.

5.2. Future Directions

The simulation study presented in this thesis can be developed in several directions. We tested eight different scenarios that we considered representative in showing how n, K and p could affect the performance of the MFM-SBM model. Clearly, these eight scenarios do not cover all possible situations and in the future would be interesting to assess more accurately the impact of an excessively small n with respect to K on the effectiveness of the algorithm through further simulations. Furthermore, it would be useful to get a clearer idea of the sensitivity of the MFM-SBM method to the variation of the parameters of the priors chosen for K by testing more values for the parameters. We tested only a small range of values in accordance to the scenarios considered. It should be mentioned also that the MFM-SBM model does not only present a prior on K, but as seen in equation (3.13) presents two more on $\boldsymbol{\pi}$ and $\mathbf{Q}_{rs}$ for $r, s = 1, \ldots, K$ respectively. Consequently, it would be useful in the future to comprehend the behaviour of MFM-SBM model when varying the different values associated to the different parameters of each prior. In our simulation study we referred to the standard values used in the literature.

In our work we generated only unbalanced networks as they are more similar to reality. In the future, for a more comprehensive simulation study, it would be interesting also to see how much would be lost in terms of performance between a network with a balanced or unbalanced community structure. As far as the type of network is concerned, we tested networks with *community structure*, therefore, another possible direction would be to investigate the behaviour of this method on networks with different structures, e.g. with the aforementioned *core-periphery structure*. Another possible idea for future development would be to add to the MFM-SBM model the possibility of using covariates that could be given for each node in the network in order to increase its effectiveness. Finally, the MFM-SBM model we used is based on the homogeneous standard SBM. Better performing variations of this model, such as the degree corrected SBM, can be found in the literature. Hence, another potential focus could be on investigating the performance between the "standard" MFM-SBM model used in this work and the "degree corrected" MFM-SBM model based on a degree corrected SBM.

In conclusion, community detection is a very broad and expanding subject with many areas for development. Here we focused on a Baysian nonparametric approach in order to perform community detection with unknown K limiting ourselves to investigate a few number of aspects that we found interesting. During this process we realized how much more there is still to understand and investigate on this topic by means of future research.

A. Simulation Results

Scenario 1
K = 2, p = 0.5, n = 100

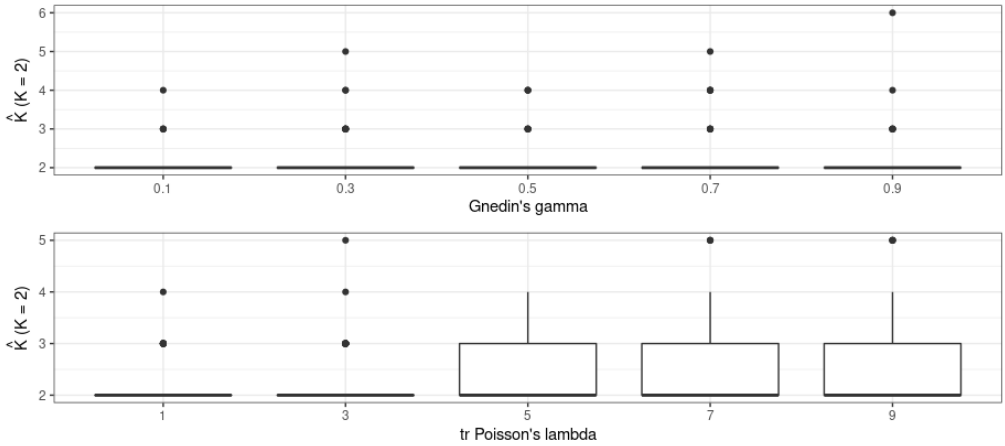


Figure A.1.: boxplots of $\hat{K}_N$ obtained for all the different parameter values of the two algorithms under scenario 1.

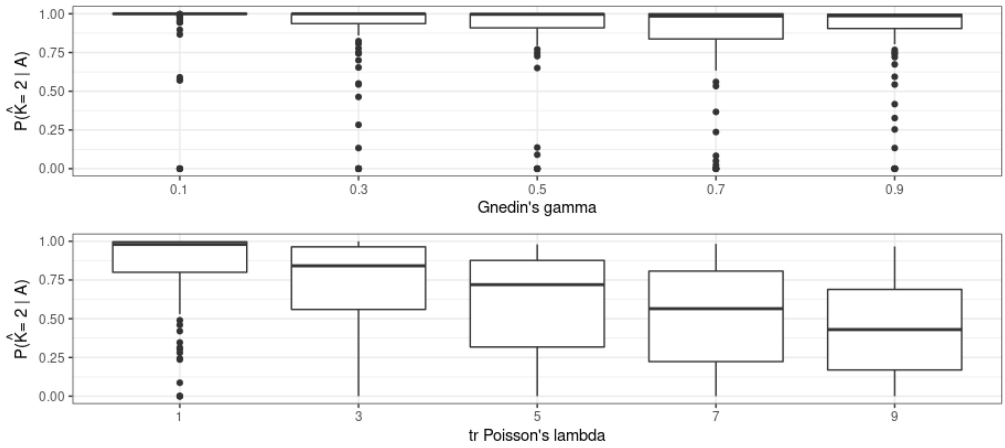


Figure A.2.: boxplots of $\mathbb{P}(\hat{K} = K|\mathbf{Y})_N$ obtained for all the different parameter values of the two algorithms under scenario 1.

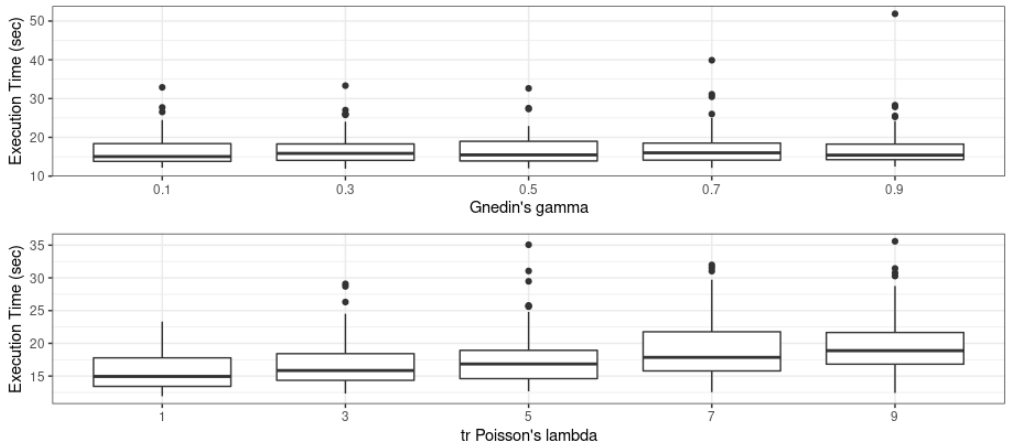


Figure A.3.: boxplots of the execution time, Time_N, obtained for all the different parameter values of the two algorithms under scenario 1.

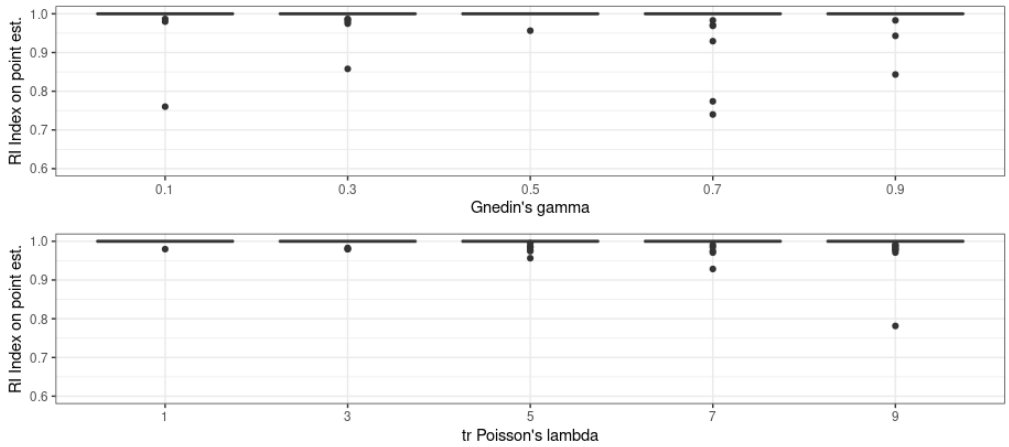


Figure A.4.: boxplots of $\text{RI}(\hat{\mathbf{z}}, \mathbf{z}_0)_N$ obtained for all the different parameter values of the two algorithms under scenario 1.

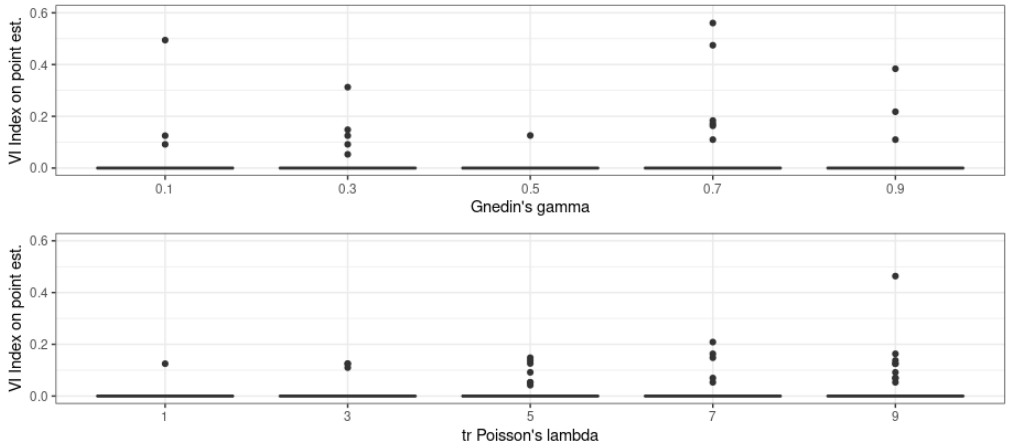


Figure A.5.: boxplots of $\text{VI}(\hat{\mathbf{z}}, \mathbf{z}_0)_N$ obtained for all the different parameter values of the two algorithms under scenario 1.

Scenario 2
K = 2, p = 0.24, n = 100

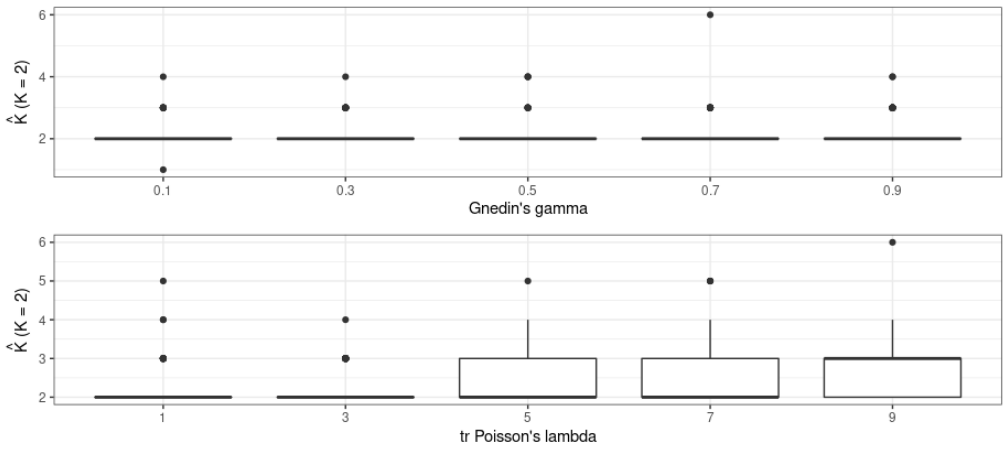


Figure A.6.: boxplots of $\hat{K}_N$ obtained for all the different parameter values of the two algorithms under scenario 2.

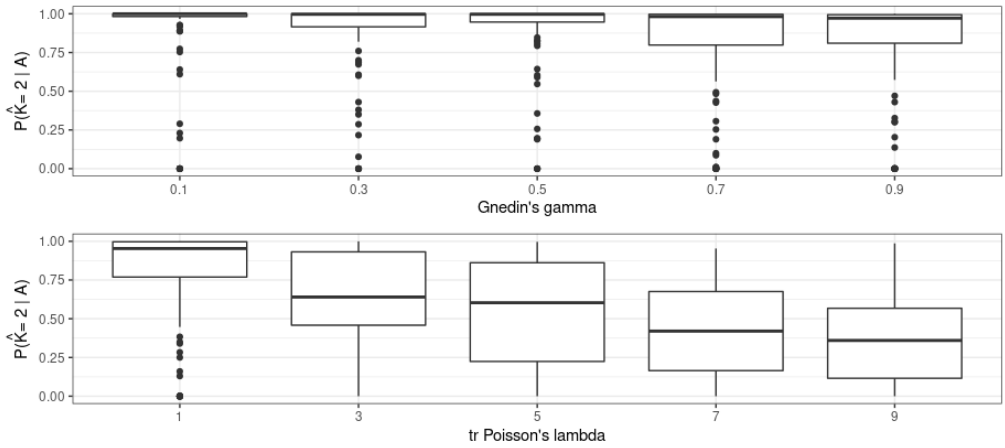


Figure A.7.: boxplots of $\mathbb{P}(\hat{K} = K|\mathbf{Y})_N$ obtained for all the different parameter values of the two algorithms under scenario 2.

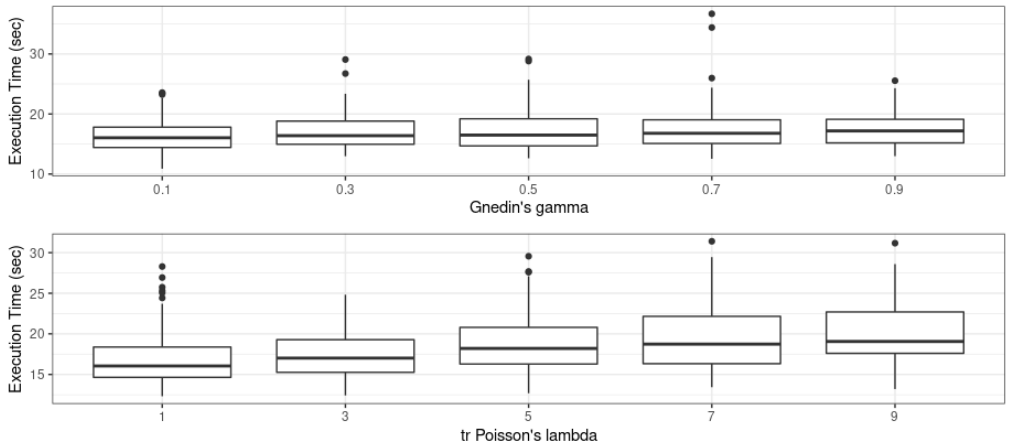


Figure A.8.: boxplots of the execution time, Time_N, obtained for all the different parameter values of the two algorithms under scenario 2.

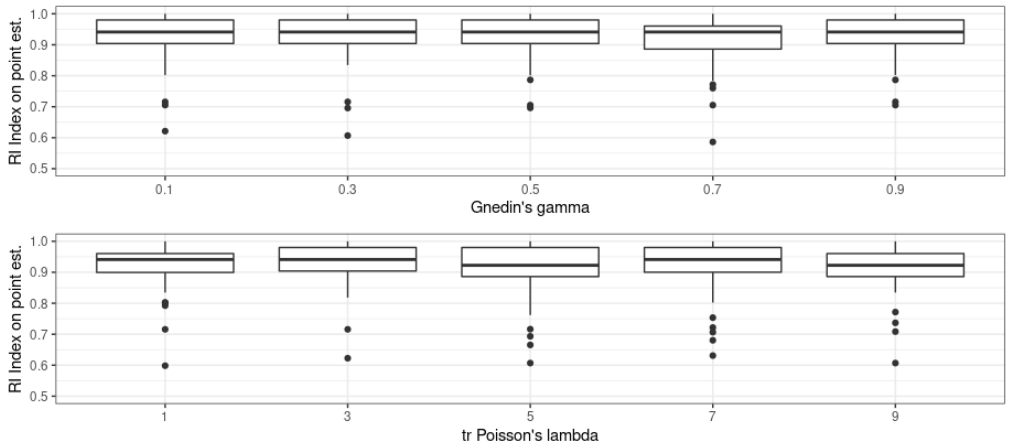


Figure A.9.: boxplots of $\text{RI}(\hat{\mathbf{z}}, \mathbf{z}_0)_N$ obtained for all the different parameter values of the two algorithms under scenario 2.

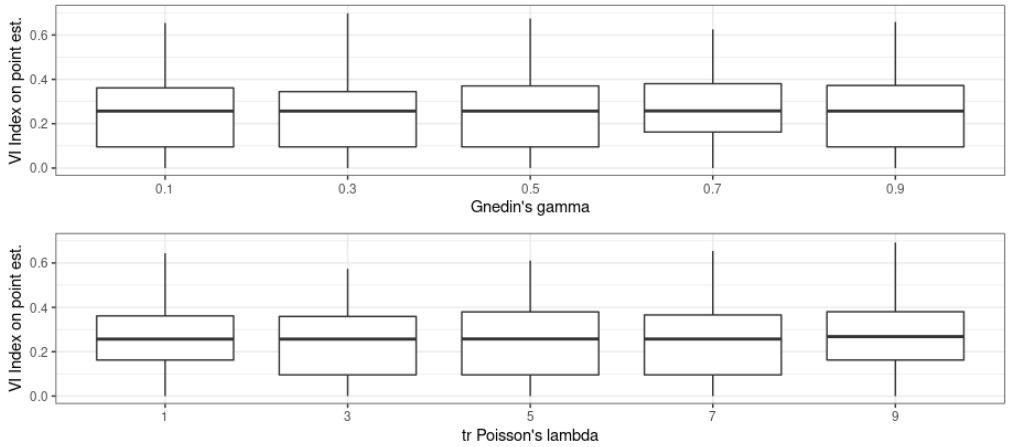


Figure A.10.: boxplots of $\text{VI}(\hat{\mathbf{z}}, \mathbf{z}_0)_N$ obtained for all the different parameter values of the two algorithms under scenario 2.

Scenario 3
K = 5, p = 0.5, n = 100

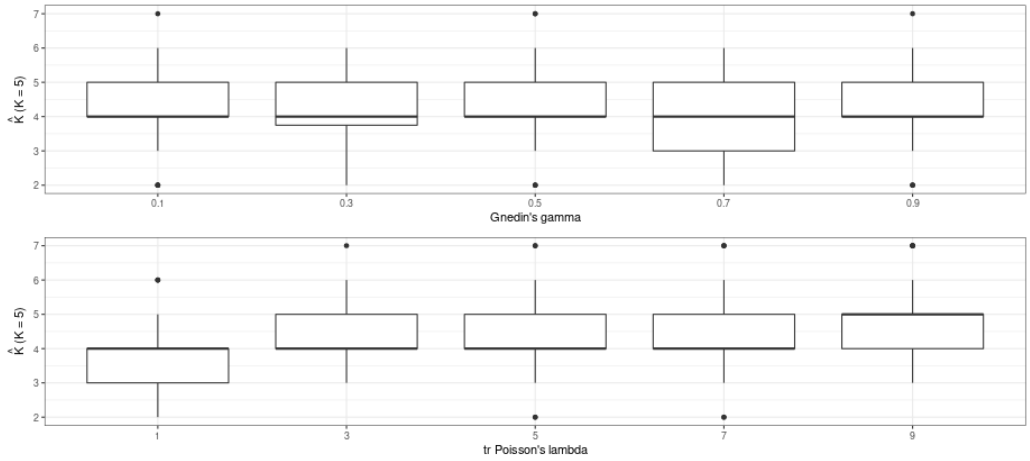


Figure A.11.: boxplots of $\hat{K}_N$ obtained for all the different parameter values of the two algorithms under scenario 3.

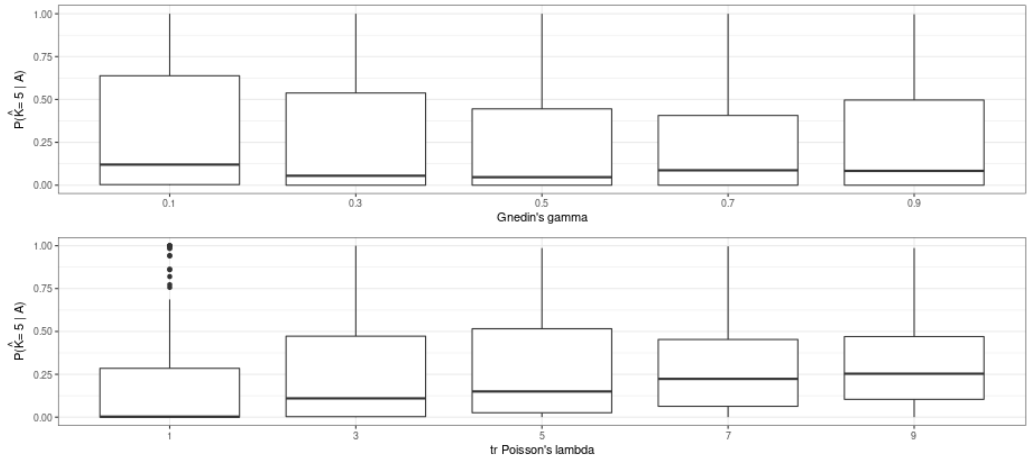


Figure A.12.: boxplots of $\mathbb{P}(\hat{K} = K|\mathbf{Y})_N$ obtained for all the different parameter values of the two algorithms under scenario 3.

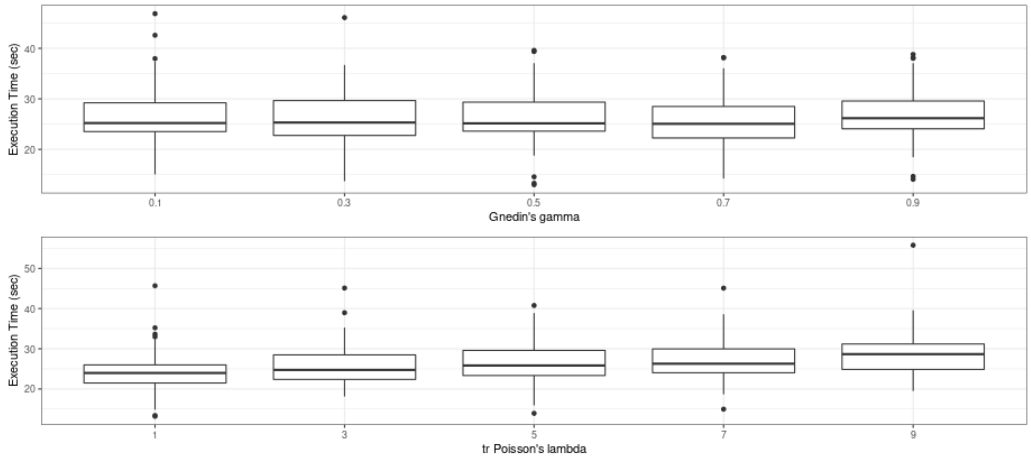


Figure A.13.: boxplots of the execution time, Time_N, obtained for all the different parameter values of the two algorithms under scenario 3.

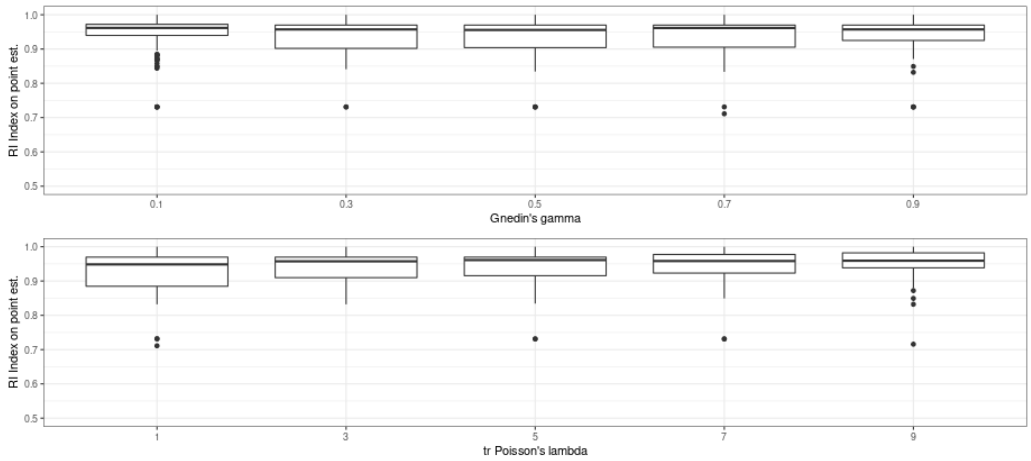


Figure A.14.: boxplots of $\text{RI}(\hat{\mathbf{z}}, \mathbf{z}_0)_N$ obtained for all the different parameter values of the two algorithms under scenario 3.

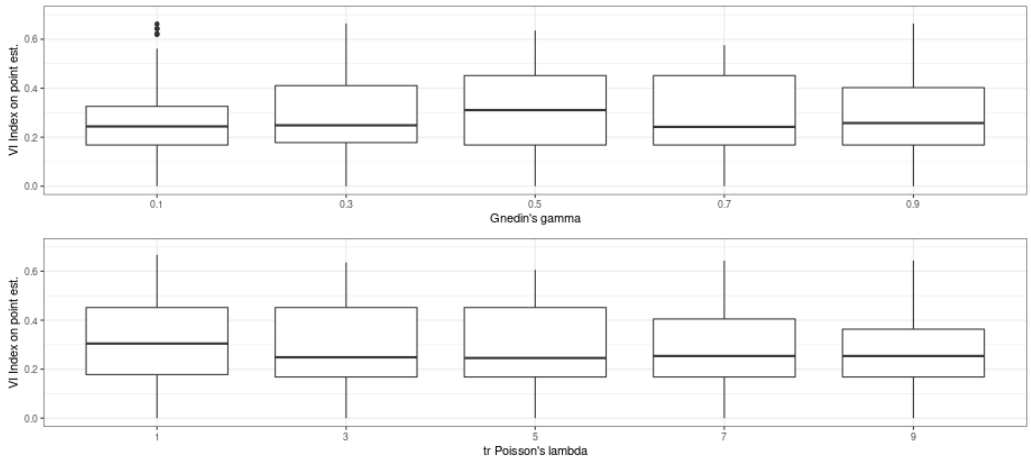


Figure A.15.: boxplots of $\text{VI}(\hat{\mathbf{z}}, \mathbf{z}_0)_N$ obtained for all the different parameter values of the two algorithms under scenario 3.

Scenario 4
K = 5, p = 0.24, n = 100

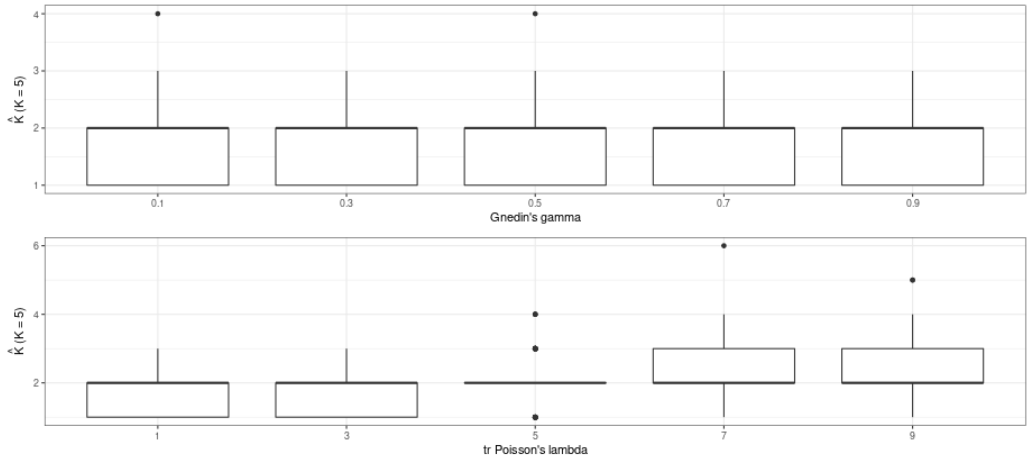


Figure A.16.: boxplots of $\hat{K}_N$ obtained for all the different parameter values of the two algorithms under scenario 4.

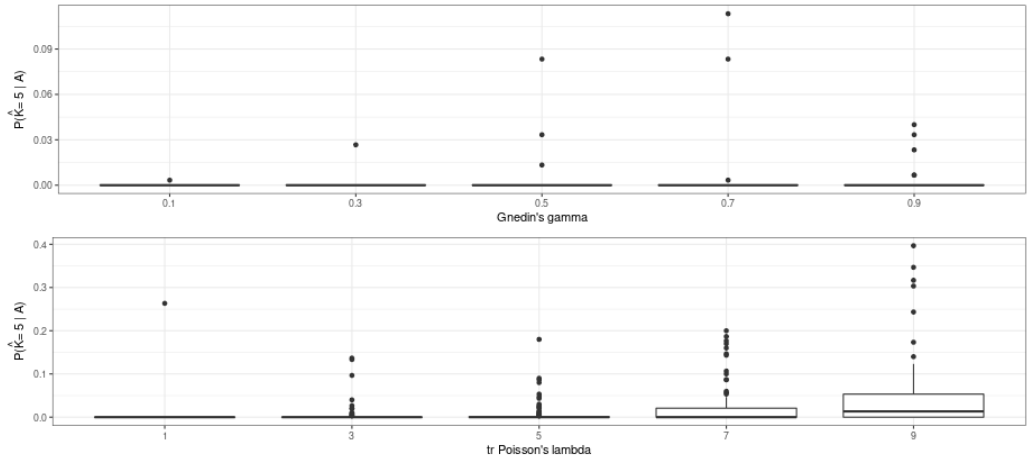


Figure A.17.: boxplots of $\mathbb{P}(\hat{K} = K|\mathbf{Y})_N$ obtained for all the different parameter values of the two algorithms under scenario 4.

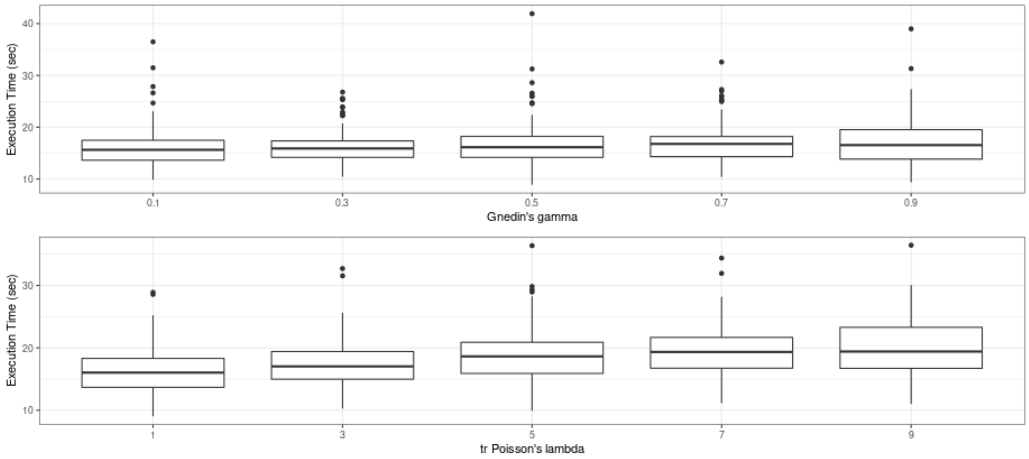


Figure A.18.: boxplots of the execution time, Time_N, obtained for all the different parameter values of the two algorithms under scenario 4.

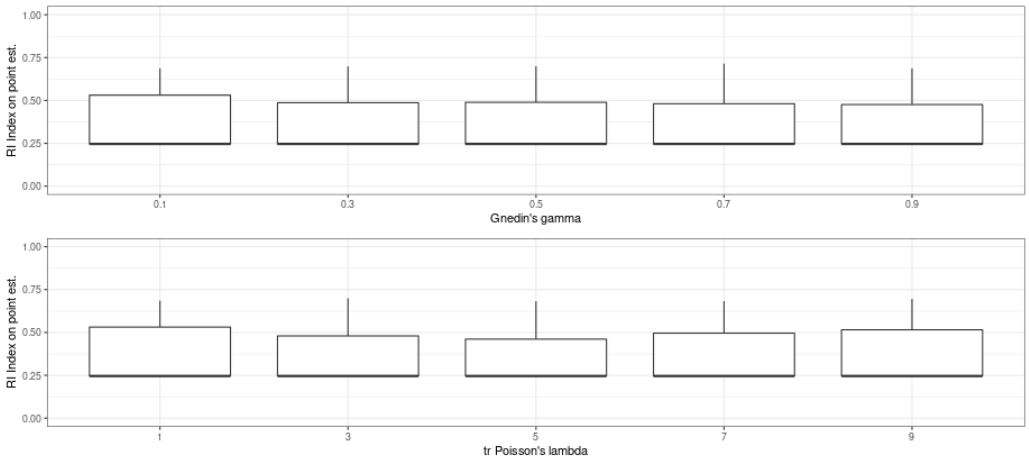


Figure A.19.: boxplots of $\text{RI}(\hat{\mathbf{z}}, \mathbf{z}_0)_N$ obtained for all the different parameter values of the two algorithms under scenario 4.

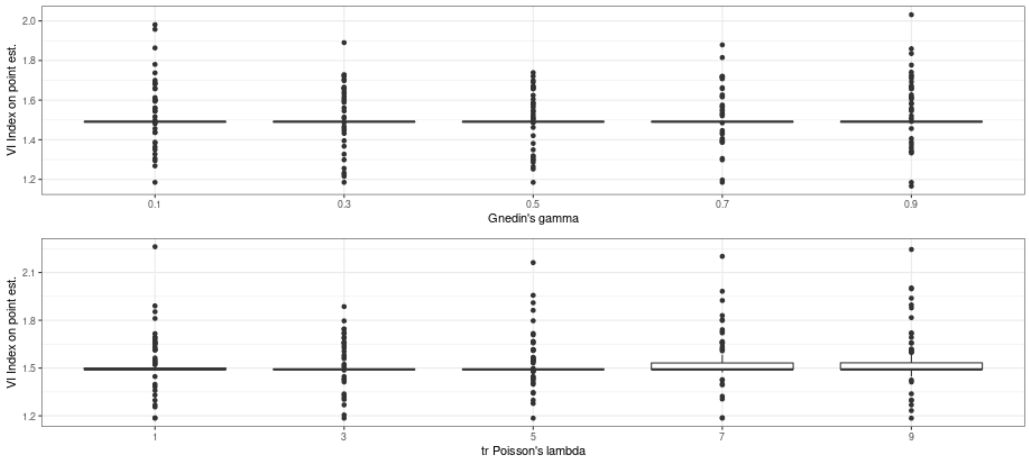


Figure A.20.: boxplots of $\text{VI}(\hat{\mathbf{z}}, \mathbf{z}_0)_N$ obtained for all the different parameter values of the two algorithms under scenario 4.

Scenario 5
K = 2, p = 0.5, n = 200

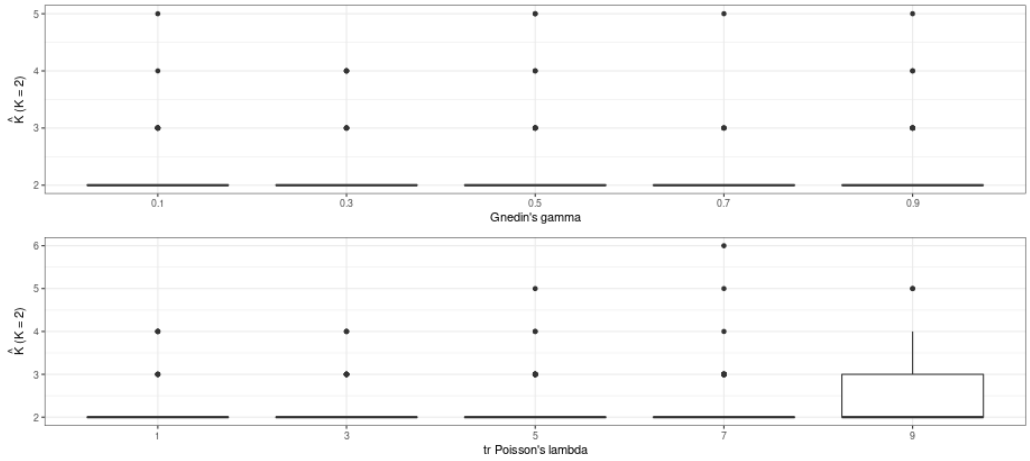


Figure A.21.: boxplots of $\hat{K}_N$ obtained for all the different parameter values of the two algorithms under scenario 5.

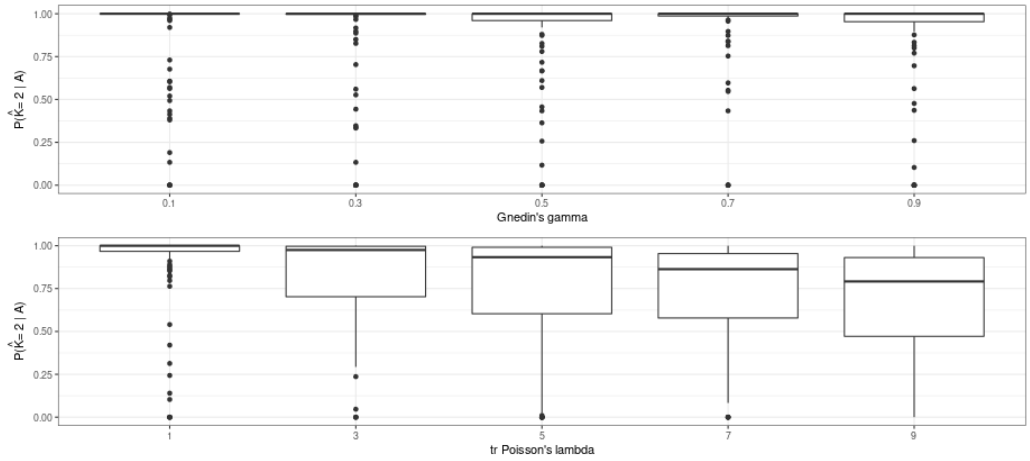


Figure A.22.: boxplots of $\mathbb{P}(\hat{K} = K|\mathbf{Y})_N$ obtained for all the different parameter values of the two algorithms under scenario 5.

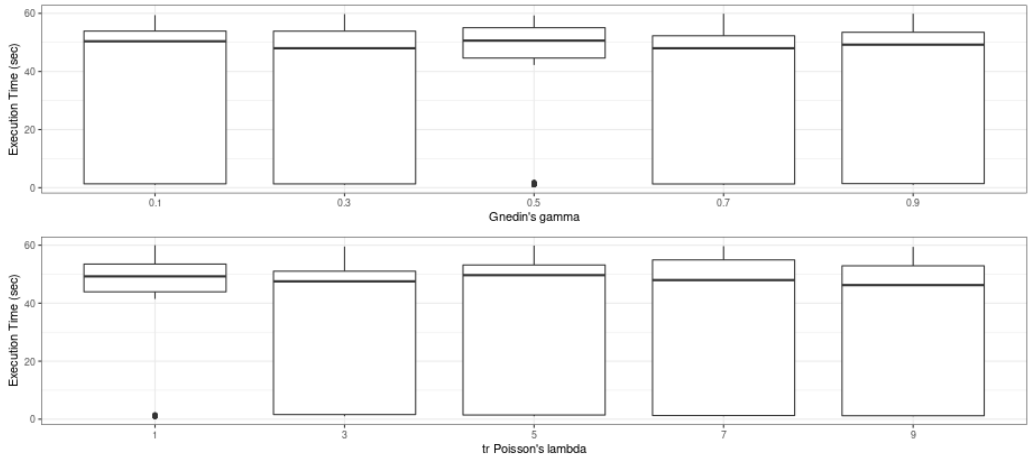


Figure A.23.: boxplots of the execution time, Time_N, obtained for all the different parameter values of the two algorithms under scenario 5.

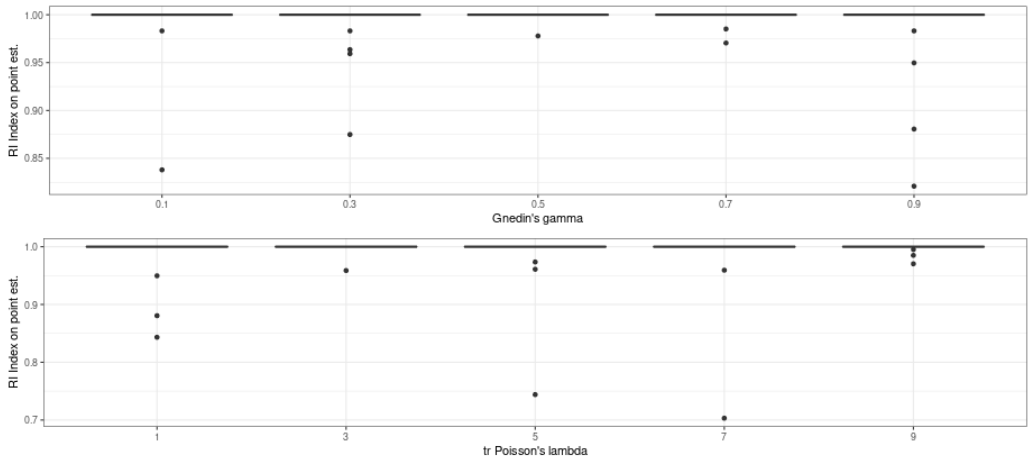


Figure A.24.: boxplots of $\text{RI}(\hat{\mathbf{z}}, \mathbf{z}_0)_N$ obtained for all the different parameter values of the two algorithms under scenario 5.

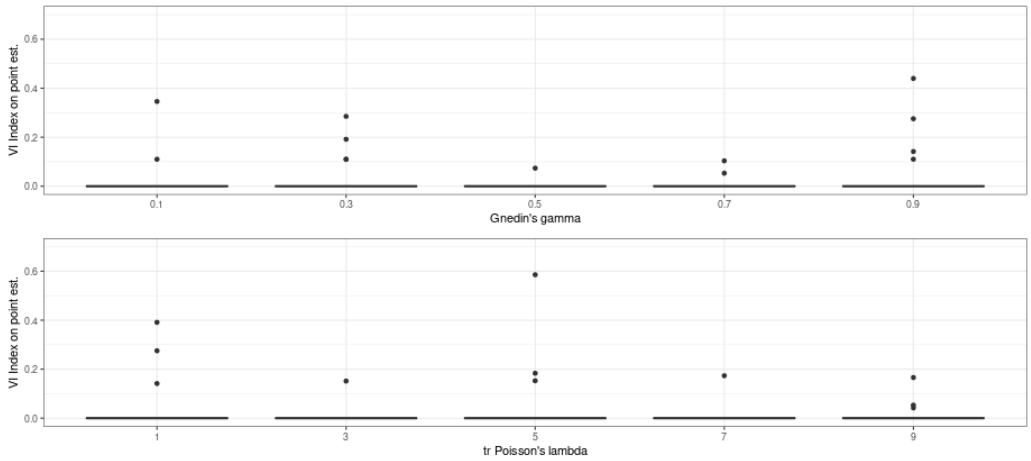


Figure A.25.: boxplot of $\text{VI}(\hat{\mathbf{z}}, \mathbf{z}_0)_N$ obtained for all the different parameter values of the two algorithms under scenario 5.

Scenario 6
K = 2, p = 0.24, n = 200

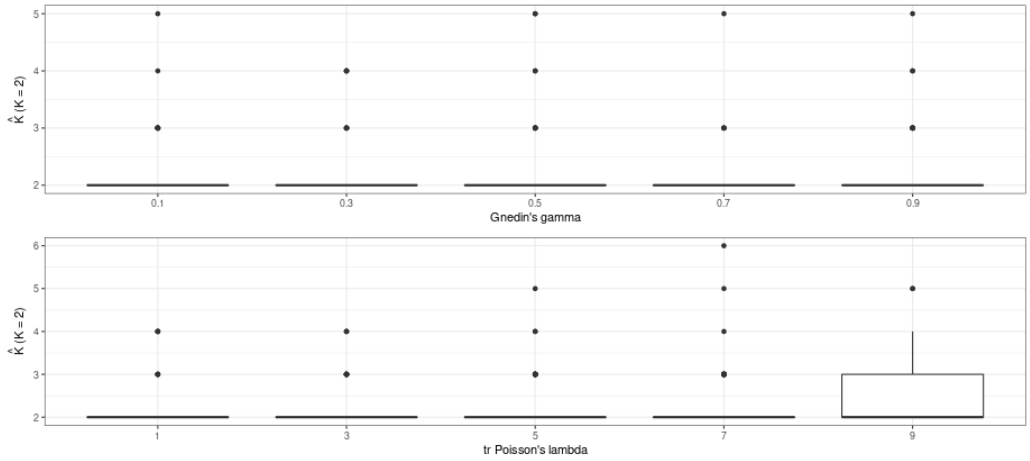


Figure A.26.: boxplots of $\hat{K}_N$ obtained for all the different parameter values of the two algorithms under scenario 6.

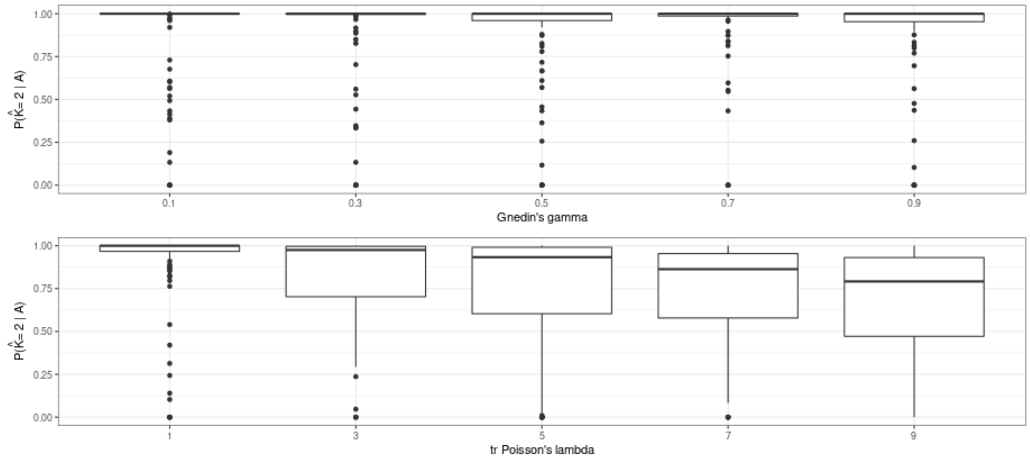


Figure A.27.: boxplots of $\mathbb{P}(\hat{K} = K|\mathbf{Y})_N$ obtained for all the different parameter values of the two algorithms under scenario 6.

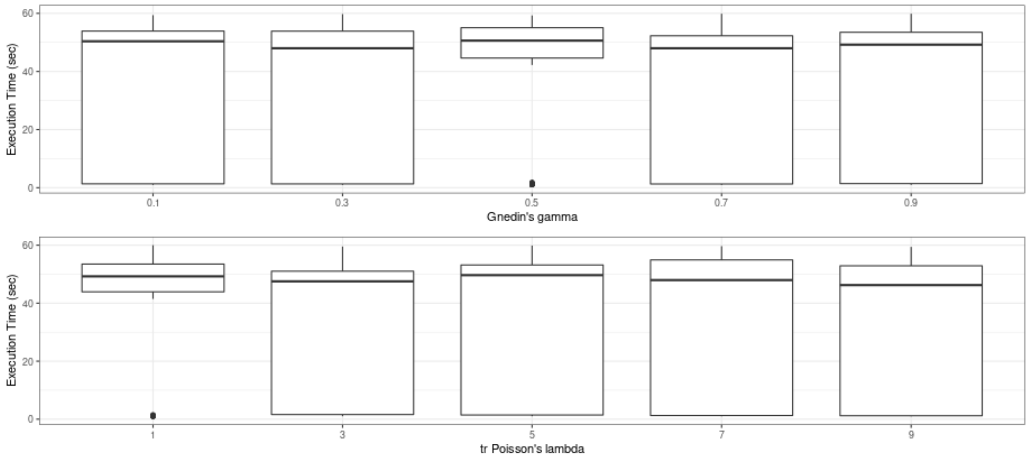


Figure A.28.: boxplots of the execution time, Time_N, obtained for all the different parameter values of the two algorithms under scenario 6.

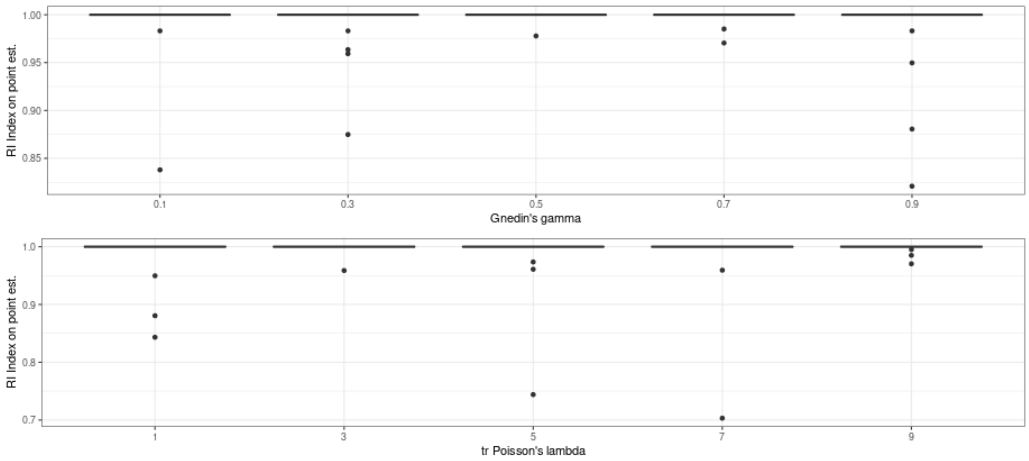


Figure A.29.: boxplots of $\text{RI}(\hat{\mathbf{z}}, \mathbf{z}_0)_N$ obtained for all the different parameter values of the two algorithms under scenario 6.

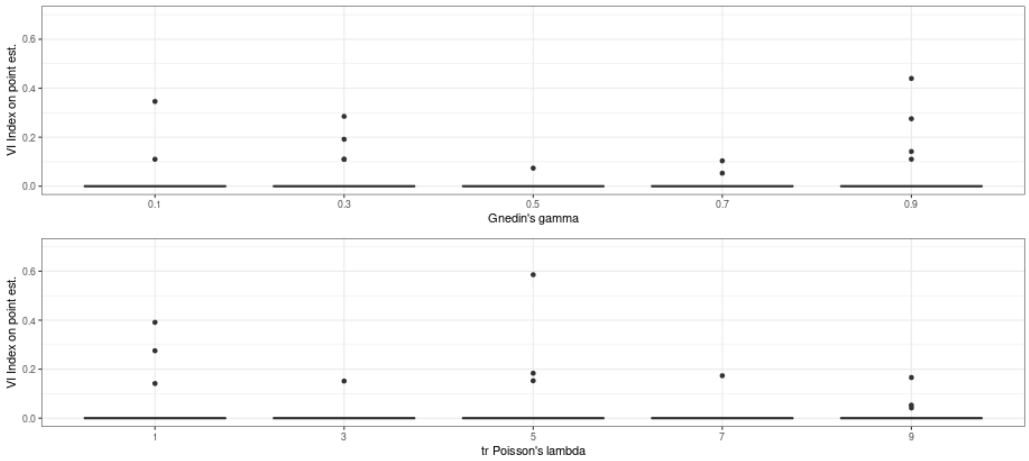


Figure A.30.: boxplots of $\text{VI}(\hat{\mathbf{z}}, \mathbf{z}_0)_N$ obtained for all the different parameter values of the two algorithms under scenario 6.

Scenario 7
K = 5, p = 0.5, n = 200

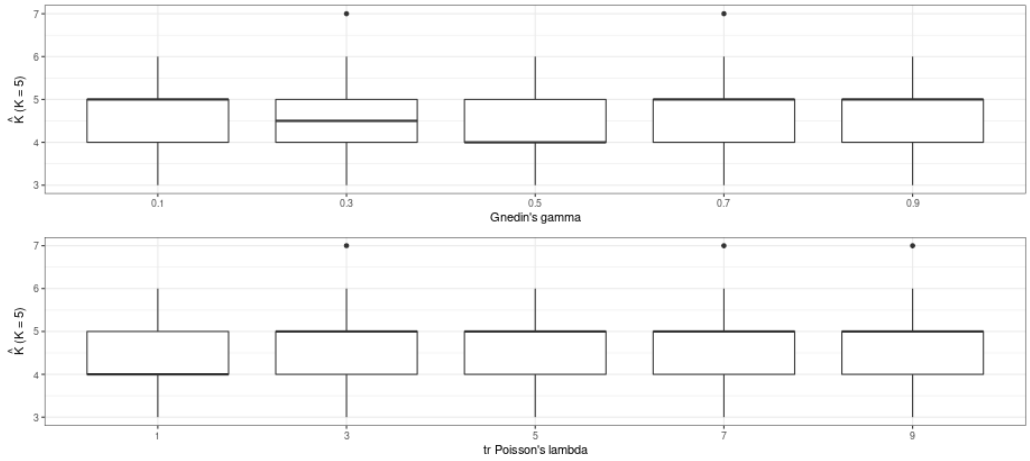


Figure A.31.: boxplots of $\hat{K}_N$ obtained for all the different parameter values of the two algorithms under scenario 7.

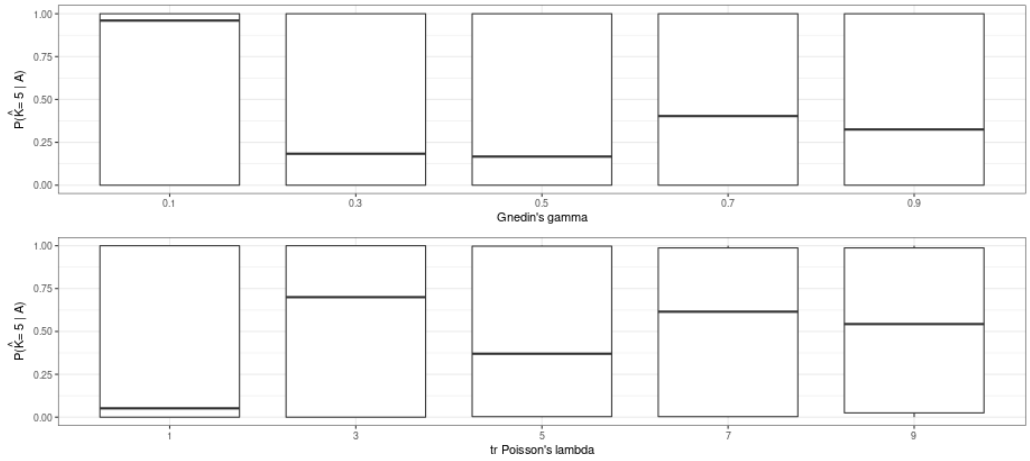


Figure A.32.: boxplots of $\mathbb{P}(\hat{K} = K|\mathbf{Y})_N$ obtained for all the different parameter values of the two algorithms under scenario 7.

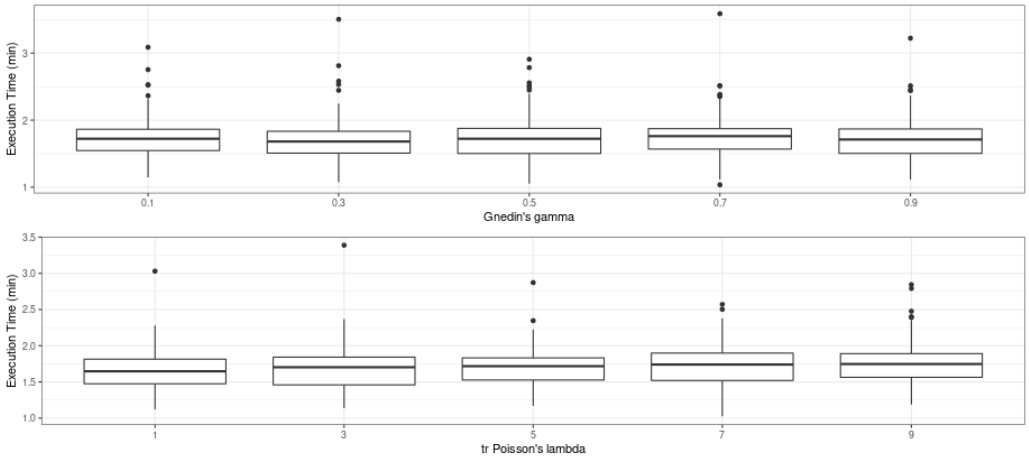


Figure A.33.: boxplots of the execution time, Time_N, obtained for all the different parameter values of the two algorithms under scenario 7.

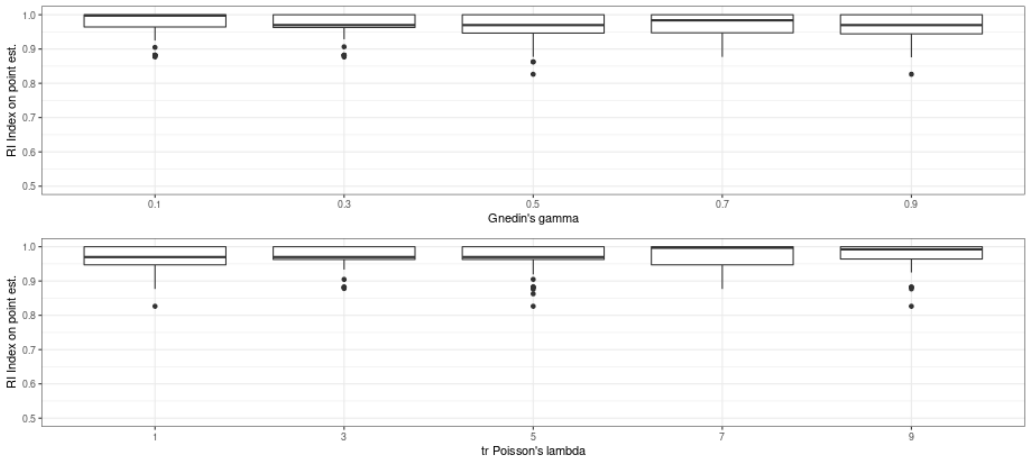


Figure A.34.: boxplots of $\text{RI}(\hat{\mathbf{z}}, \mathbf{z}_0)_N$ obtained for all the different parameter values of the two algorithms under scenario 7.

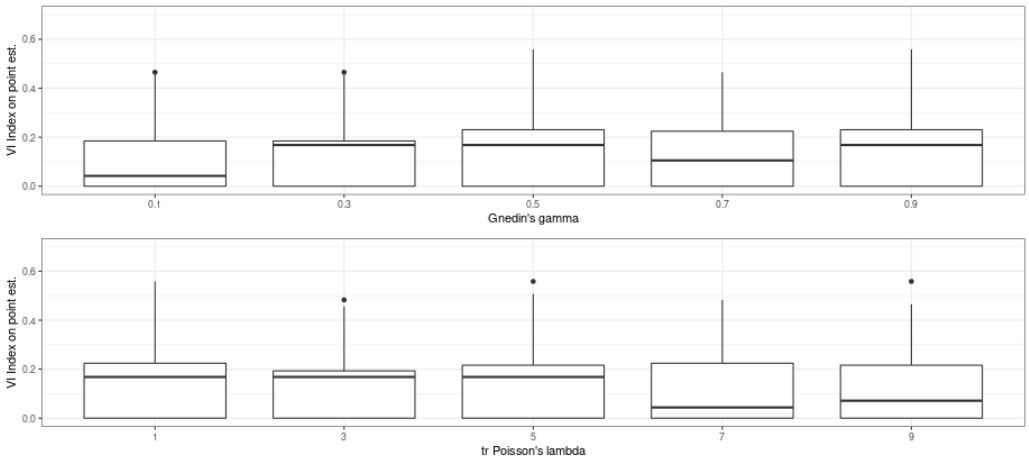


Figure A.35.: boxplots of $\text{VI}(\hat{\mathbf{z}}, \mathbf{z}_0)_N$ obtained for all the different parameter values of the two algorithms under scenario 7.

Scenario 8
K = 5, p = 0.24, n = 200

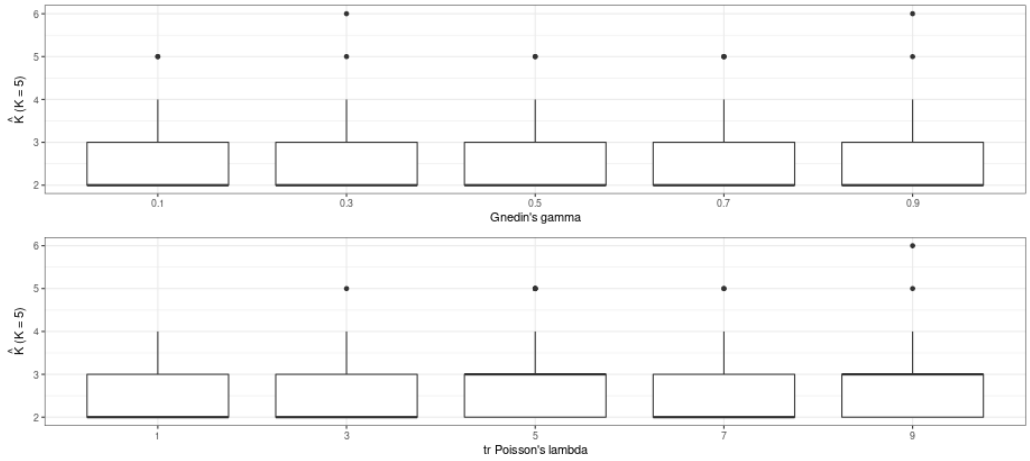


Figure A.36.: boxplots of $\hat{K}_N$ obtained for all the different parameter values of the two algorithms under scenario 8.

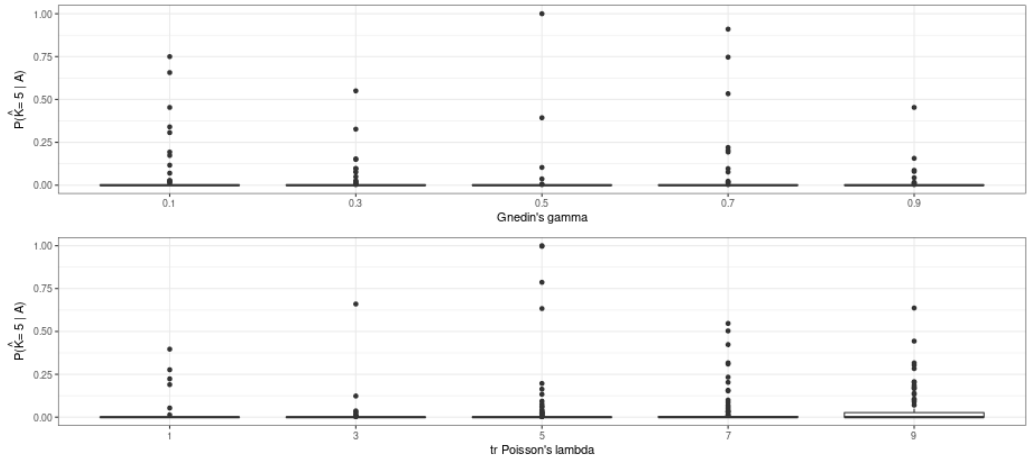


Figure A.37.: boxplots of $\mathbb{P}(\hat{K} = K|\mathbf{Y})_N$ obtained for all the different parameter values of the two algorithms under scenario 8.

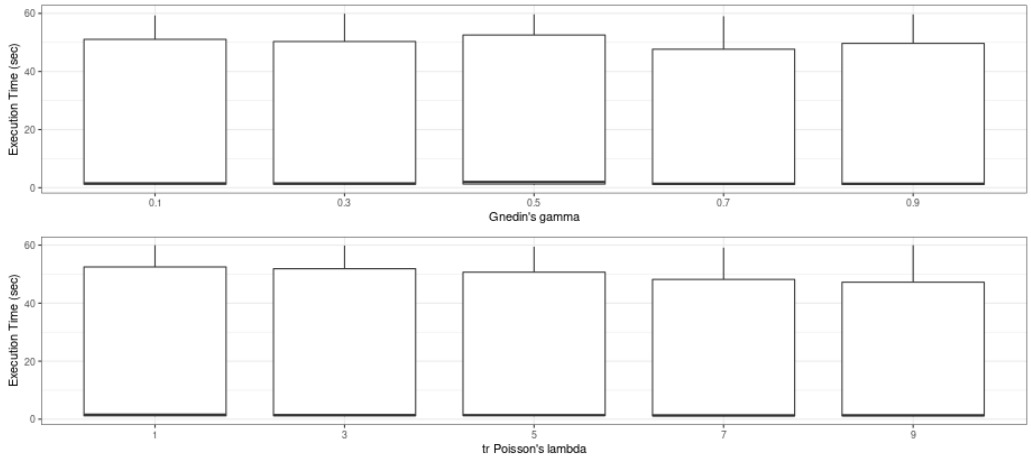


Figure A.38.: boxplots of the execution time, Time_N, obtained for all the different parameter values of the two algorithms under scenario 8.

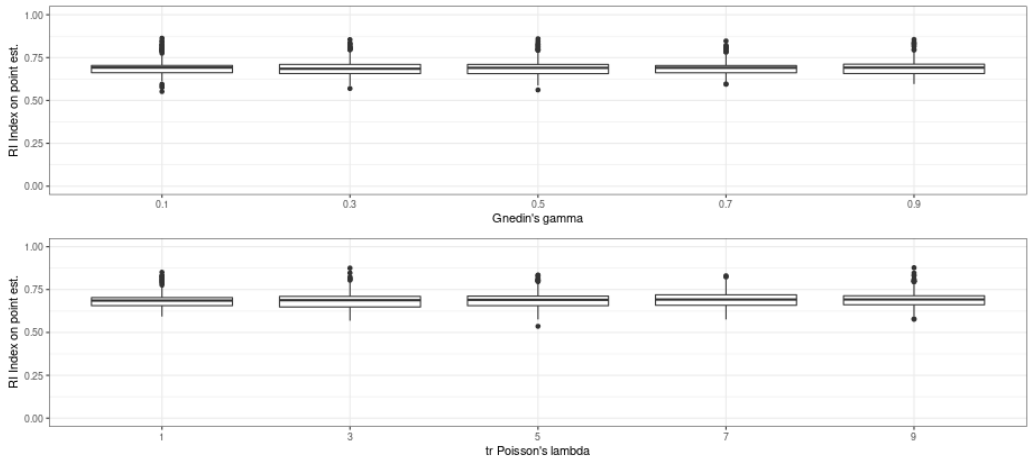


Figure A.39.: boxplots of $\text{RI}(\hat{\mathbf{z}}, \mathbf{z}_0)_N$ obtained for all the different parameter values of the two algorithms under scenario 8.

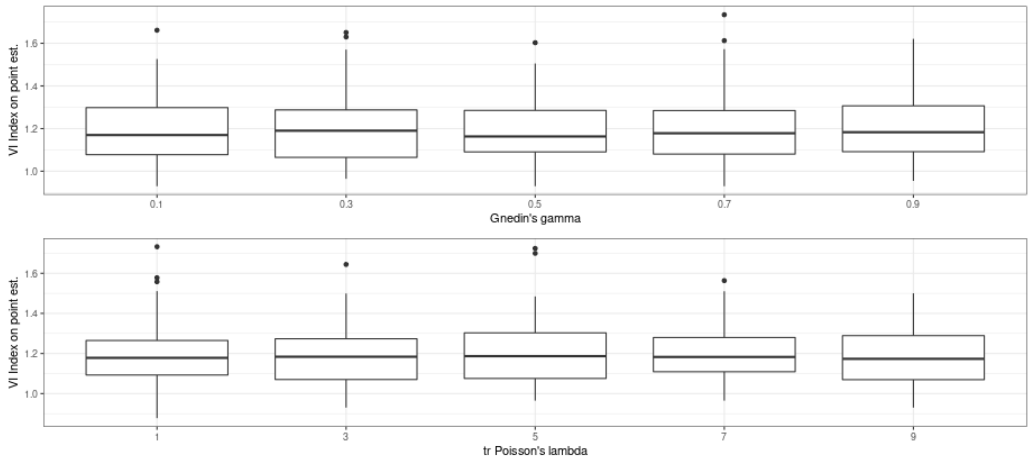


Figure A.40.: boxplots of $\text{VI}(\hat{\mathbf{z}}, \mathbf{z}_0)_N$ obtained for all the different parameter values of the two algorithms under scenario 8.

Bibliography

Airoldi, E. M., Blei, D. M., Fienberg, S. E., and Xing, E. P. (2008). Mixed membership stochastic blockmodels. *Journal of machine learning research*.

Blackwell, D. and MacQueen, J. B. (1973). Ferguson Distributions Via Polya Urn Schemes. *The Annals of Statistics*, 1(2):353 – 355.

Bloem-Reddy, B. (2018). Exchangeable random partitions and random discrete probability measures: a brief tour guided by the dirichlet process.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008.

De Blasi, P., Favaro, S., Lijoi, A., Mena, R. H., Prünster, I., and Ruggiero, M. (2013a). Are gibbs-type priors the most natural generalization of the dirichlet process? *IEEE transactions on pattern analysis and machine intelligence*, 37(2):212–229.

De Blasi, P., Lijoi, A., and Prünster, I. (2013b). An asymptotic analysis of a class of discrete nonparametric priors. *Statistica Sinica*, 23(3):1299–1321.

Ferguson, T. S. (1973). A Bayesian Analysis of Some Nonparametric Problems. *The Annals of Statistics*, 1(2):209 – 230.

Fienberg, S. E. and Wasserman, S. S. (1981). Categorical data analysis of single sociometric relations. *Sociological methodology*, 12:156–192.

Fortunato, S. and Hric, D. (2016). Community detection in networks: A user guide. *Physics Reports*, 659:1–44. Community detection in networks: A user guide.

Funke, T. and Becker, T. (2019). Stochastic block models: A comparison of variants and inference methods. *PLOS ONE*, 14(4):1–40.

Geng, J., Bhattacharya, A., and Pati, D. (2019). Probabilistic community detection with unknown number of communities. *Journal of the American Statistical Association*, 114(526):893–905.

Gershman, S. J. and Blei, D. M. (2012). A tutorial on bayesian nonparametric models. *Journal of Mathematical Psychology*, 56(1):1–12.

Ghosal, S. and van der Vaart, A. (2017). *Fundamentals of Nonparametric Bayesian Inference*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press.

Girvan, M. and Newman, M. E. (2002). Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826.

Gnedin, A. et al. (2010). A species sampling model with finitely many types. *Electronic Communications in Probability*, 15:79–88.

Holland, P. W., Laskey, K. B., and Leinhardt, S. (1983). Stochastic blockmodels: First steps. *Social Networks*, 5(2):109–137.

Karrer, B. and Newman, M. E. (2011). Stochastic blockmodels and community structure in networks. *Physical review E*, 83(1):016107.

Korwar, R. M. and Hollander, M. (1973). Contributions to the Theory of Dirichlet Processes. *The Annals of Probability*, 1(4):705 – 711.

Lee, C. and Wilkinson, D. J. (2019). A review of stochastic block models and extensions for graph clustering. *Applied Network Science*, 4(1):1–50.

Legramanti, S., Rigon, T., Durante, D., and Dunson, D. B. (2020). Extended stochastic block models. *arXiv preprint arXiv:2007.08569*.

Leskovec, J., Lang, K. J., and Mahoney, M. (2010). Empirical comparison of algorithms for network community detection. In *Proceedings of the 19th international conference on World wide web*, pages 631–640.

Liben-Nowell, D., Novak, J., Kumar, R., Raghavan, P., and Tomkins, A. (2005). Geographic routing in social networks. *Proceedings of the National Academy of Sciences*, 102(33):11623–11628.

Lusseau, D. and Newman, M. E. (2004). Identifying the role that animals play in their social networks. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 271(suppl_6):S477–S481.

Lusseau, D., Schneider, K., Boisseau, O. J., Haase, P., Slooten, E., and Dawson, S. M. (2003). The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations. *Behavioral Ecology and Sociobiology*, 54(4):396–405.

MacEachern, S. N. (2016). Nonparametric bayesian methods: a gentle introduction and overview. *Communications for Statistical Applications and Methods*, 23(6):445–446.

Miller, J. W. and Harrison, M. T. (2018). Mixture models with a prior on the number of components. *Journal of the American Statistical Association*, 113(521):340–356. PMID: 29983475.

Müller, P. and Mitra, R. (2013). Bayesian Nonparametric Inference – Why and How. *Bayesian Analysis*, 8(2):269 – 302.

Neal, R. M. (2000). Markov chain sampling methods for dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9(2):249–265.

Newman, M. (2018). *Networks*. Oxford university press.

Newman, M. E. (2006). Modularity and community structure in networks. *Proceedings of the national academy of sciences*, 103(23):8577–8582.

Newman, M. E. and Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113.

Newman, M. E. J. and Reinert, G. (2016). Estimating the number of communities in a network. *Phys. Rev. Lett.*, 117:078301.

Nowicki, K. and Snijders, T. A. B. (2001). Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 96(455):1077–1087.

Pitman, J. (1996). Some developments of the blackwell-macqueen urn scheme. *Lecture Notes-Monograph Series*, pages 245–267.

Riolo, M. A., Cantwell, G. T., Reinert, G., and Newman, M. E. J. (2017). Efficient method for estimating the number of communities in a network. *Phys. Rev. E*, 96:032310.

Rosvall, M., Delvenne, J.-C., Schaub, M. T., and Lambiotte, R. (2019). Different approaches to community detection. *Advances in network clustering and blockmodeling*, pages 105–119.

Sethuraman, J. (1994). A constructive definition of the dirichlet prior. *Statistica Sinica*, 4:639–650.

Snijders, T. A. and Nowicki, K. (1997). Estimation and prediction for stochastic blockmodels for graphs with latent block structure. *Journal of classification*, 14(1):75–100.

Teh, Y. W. (2010). Dirichlet process.

Wade, S., Ghahramani, Z., et al. (2018). Bayesian cluster analysis: Point estimation and credible balls (with discussion). *Bayesian Analysis*, 13(2):559–626.

Wasserman, S. and Anderson, C. (1987). Stochastic a posteriori blockmodels: Construction and assessment. *Social networks*, 9(1):1–36.