ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

SEMESTER PROJECT

BACHELOR IN MATHEMATICS

Random graph models and inference methods

Author:

Thomas MOLINIER

Supervisors:

Anda SKEJA

Prof. Sofia OLHEDE

Contents

Preliminaries

This section consits of definitions of notions used throughout the project.

Definition 0.0.1 (Graph, vertex, edge). A graph is as a pair $G = (V, E)$, where V denotes the vertex set and $E \subseteq \{xy | (x, y) \in V^2\}$ denotes the edge set.

Considering a graph $G = (V, E)$, we define :

Definition 0.0.2 (Order and size). The *order* and *size* of G are respectively $|V|$ and $|E|$.

Definition 0.0.3 (Adjacent vertices). For any pair $u, v \in V$ we say that u, v are *adjacent* if $uv \in E$.

Definition 0.0.4 (Adjacent edges). For any $u, v, w \in V$ such that $uv = e, vw = f \in E$, we say that e and f are *adjacent*.

Definition 0.0.5 (Degree). For any $u \in V$ the *degree of* u corresponds to the number of its neighbours. Formally, $deg(v) = |\{u \in V | vu \in E\}|$

Definition 0.0.6 (Incidency). For any pair $u, v \in V$ such that $uv = e \in E$, we say that v and u are *incident* to e.

The following definitions are necessary to understand the Configuration Model which is defined in Part 2.2.

Definition 0.0.7 (Degree sequence). Let $G = (V, E)$, be an order p graph. A sequence $d_1, ..., d_p \in \mathbb{N}$ is called *a degree sequence for* G if there exists a labelling $v_1, ..., v_p$ of V such that $deg(v_i) = d_i \ \forall i = 1, ..., p$.

Definition 0.0.8 (Graphical sequence). A sequence of p non-negative integers is called a *graphical sequence* (or graphic sequence or graph sequence) if there exists an order p graph that admits such a degree sequence.

Definition 0.0.9 (Adjacency matrix). Let $G = (V, E)$ be an order n simple graph. The *Adjacency matrix* A is an n x n matrix defined as follows :

$$A_{ij} = \begin{cases} 1, & \text{if } v_i v_j \in E \\ 0, & \text{otherwise} \end{cases}$$

This definition extends naturally to weighted graphs with $A_{ij} = w_{ij}$, the weight of the edge $v_i v_j$.

Definition 0.0.10 (Laplacian matrix). Let $G = (V, E)$ be an order n simple graph. The *Laplacian matrix* L is an n x n matrix defined as follows :

$$L_{ij} := \begin{cases} \deg(v_i), & \text{if } i = j \\ -1, & \text{if } i \neq j \text{ and } v_i \text{ is adjacent to } v_j \\ 0, & \text{otherwise} \end{cases}$$

This definition extends naturally to weighted graphs with $L = D - A$, with A the adjacency matrix for a weighted graph and D the degree matrix containing the degree of the vertices on the diagonal and 0 elsewhere.

Definition 0.0.11 (Likelihood function). Let x_i, $i = 1, ..., n$ be observations of n random variables drawn from a probability distribution with joint density f_P belonging to a parametric family $\{f(.;P)|P \in \Psi\}$ with Ψ the parameter space, of finite dimension. The likelihood function is defined as the evaluation of the joint density at the observation

$$L(P) := f(x_1, ..., x_n; P)$$

Definition 0.0.12 (Graph labeling). Given a graph $G = (V, E)$, a labeling is a function

$$f : V \longrightarrow \{1, ..., |V|\}.$$

A graph equipped with such a function is called a **labeled graph**, otherwise it is called **unlabeled**.

1 Introduction

"A graph is worth a thousand words"

Frank Harary, mathematician

Graphs, also known as networks, have a long history and have gained significant attention since their initial appearance in Leonhard Euler's Seven Bridges of Königsberg [15]. These graphical representations of complex data have become increasingly prevalent in data-related fields due to the exponential growth of population and advancements in technology. Notably, colossal networks like the World Wide Web, which encompasses billions of indexed pages, have emerged as prominent examples. The prevalence of large networks has led to the development of Network Science, an interdisciplinary field utilizing graph theory, probability, statistics, and information theory. Nowadays, large graphs are ubiquitous in the real-world, and consequently they are at the core of various active fields of study, such as computer science, electrical engineering[7, 34], finance, biology[18], archaeology[36], economics, public health[38], sociology, neuroscience [6]and many more. Papers that are considered to be among founding papers of the networks science are known to be [42, 4] and [17], as they have first raised awareness on complex networks, which is an intense area of study [25]. Random graph models are essential tools in the field of network science, providing a framework for studying and understanding complex networks. These models involve probability distributions over graphs and serve as crucial null models for analyzing real-world networks. Random graph models play a significant role in diverse fields where complex networks need to be modeled, including social networks, biological networks, communication networks, and the internet. They allow researchers to simulate and analyze network structures, study various network properties, and investigate the emergence of patterns and behaviors in real-world networks. By comparing real networks to the predictions of random graph models, researchers can identify unique features, detect deviations from randomness, and gain insights into the underlying mechanisms shaping complex systems. Through the development and analysis of random graph models, researchers aim to uncover fundamental principles governing network formation, dynamics, and functionality, ultimately contributing to advancements in various scientific disciplines and practical applications. Among the most popular random graph models we have the Erdős-Rényi Model [29], the Configuration Model [10], the Stochastic Block Model [19] and its extensions the Degree Corrected Stochastic Block Model [21] and Mixed Membership Stochastic Block Model, and the family of Exponential Random Graph Models [9]. First, in section 2 we will motivate and define all of these, and generate some observations of graphs following the Erdős-Rényi Model 2.1, the Configuration Model 2.2, and the Stochastic Block Model 2.3using MATLAB. Then in section 3 we will take a look into an object formalizing the notion of graph limits, called graphons, and how to link these objects to the previously seen models. Finally, in section 4 we shall investigate the question : Given a network realization from a random graph model, how can we recover its parameters? . We will look into various algorithm and methods, such as spectral methods 4.1.1, modularity maximization 4.1.5, maximum likelihood estimation 4.1.6, and network histogram4.2.2 that have been developed trying to answer this. In particular, in 4.1.2 and 4.1.3 we will apply an algorithm detailed in 4.1.1 called the spectral clustering algorithm, and discuss bounds regarding its ability to perform well in 4.1.4. We conclude our project with section 5 by summarizing what we have done.

2 Random Graph Models

2.1 Erdős–Rényi Model

Erdős–Rényi (ER) model is the first and simplest random graph model. Despite its simplicity, it has served to be a foundational model which has been studied extensively[29]. To define it in an intuitive fashion, consider a set of vertices. To build an Erdős–Rényi graph from this set, pick any pair of vertices from it. The probability that there exists and edge between the two nodes is fixed, and the same for every such pair. Below we provide two different related definitions of the Erdős–Rényi random graph model:

Definition 2.1.1 (ER G(n, p)). Let G be a graph on n vertices, $\mathrm{A} \in \mathbb{R}^{n\times n}$ its adjacency matrix, and $p \in [0,1]$. Then G is said to be an be ER G(n,p) random graph if :

$$\mathrm{A}_{ij} \overset{iid}{\sim} \mathrm{Bernoulli}(p) \tag{1}$$

for every $i, j \in 1,...,n$ distinct, and $\mathrm{A}_{ii} = 0$.

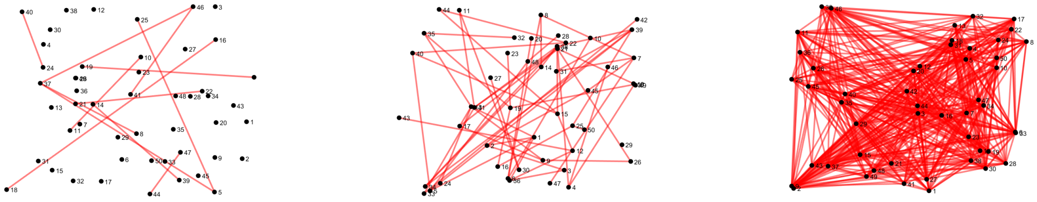
Definition 2.1.2 (ER G(n, m)). Let G = (V, E) be a labelled graph on n vertices, and m be any integer in $\{0, 1, ..., \binom{n}{2}\}$. Then G is said to be an ER G(n, m) random graph if $|E| = m$. Notice that as in this definition the vertices are considered to be labeled, two graphs that are nothing else than a relabeling one of each other are considered to be distinct.

Even if those two models may seem different at first sight, the following lemma provides their asymptotic equality :

Lemma 2.1. If $n^2 p \longrightarrow +\infty$, then G$(n, p)$ behaves similarly as G$(n, \binom{n}{2}p)$.

PROOF : Let G = (V, E) ∈ G(n, p). Then it is clear that $\mathbb{E}(|E|) = \binom{n}{2}p$, thus if this quantity tends to infinity then by the Law of Large Numbers every graph in G(n, p) will almost surely have approximately this many edges. Consequently, a reasonable heuristic is that G(n, p) and G$(n, \binom{n}{2}p)$ behave similarly provided that $n^2 p \longrightarrow +\infty$. □

Below we illustrate some graphs generated from the ER model, with different probabilities (the legends G(x,y) refer to 2.1.1) :



(a) G(50, 0.01)

(b) G(50, 0.05)

(c) G(50, 0.5)

Figure 1: ER random graphs on 50 vertices generated using MATLAB

The simplicity of the ER model is unfortunately also its main disadvantage [33], as it makes it too homogeneous to be realistic. For instance, it is frequent to see some clusters in a lot of social networks, whereas ER graphs tend to have low clustering due to the fact that the same randomization is applied to each edge appearance, without taking into consideration the belonging of the vertices to a community, making them more likely to be linked by an edge. Also, in real life, it is often the case that some vertices are more important than others, so those tend to have more connections with the rest of the graph, i.e a higher degree. This motivates us to consider another model: the Configuration model.

2.2 Configuration Model

The Configuration Model [10], allows one to generate a random graph based on a degree sequence *s* (see 0.0.7). Provided the feasibility of the degree sequence, we can follow a simple algorithm to get a graph G for which *s* is graphical (see 0.0.8). Now, to get any possible random graph admitting *s* as its graphical sequence, one can simply cut all the edges in G, so that every node still retains its degree by the number of half-edges (or stubs) emanating from it. The result will be an even number of half-edges. To create a new graph with the same degree, one simply needs to randomly pair all of the half-edges, in order to create the new edges in the graph. Figure 2 illustrates this procedure. Notice that this way of proceeding can lead to multi-edges (as in figure 3a), or self-loops (as in figure 3b).

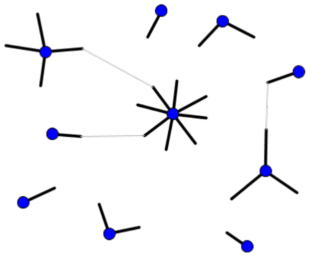
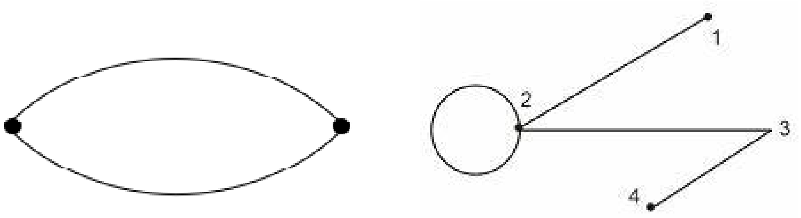


Figure 2

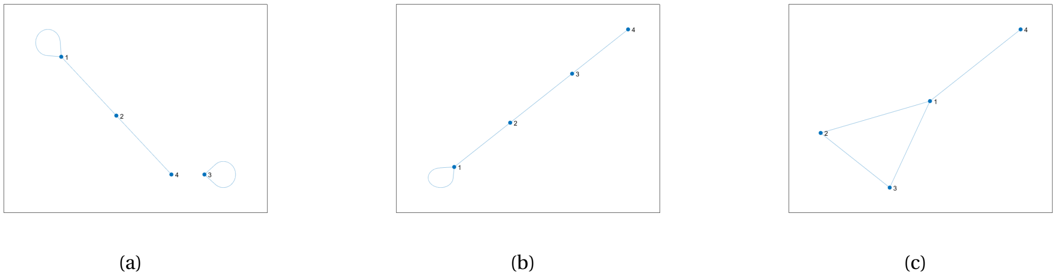


(a) Multi-edge

(b) A graph with a self loop on vertex 2

Figure 3

Because sometimes an image is worth a thousand words, we illustrate the possible outcome of Configuration Model with 4 nodes and degree sequence s=3,2,2,1 in figure 4 below.



(a) (b) (c)

Figure 4: Random graphs based on the CM on 4 vertices and degree sequence (3,2,2,1)

We see that in figure 4a and figure 4b we get self-loops. We don't get any multiple edges, as MATLAB gets automatically rid of them.

To end this subsection on the configuration model, we give the following algorithm which defines its generation :

Algorithm 1

```
Input : $m \in \mathbb{N}$, and $s = k_1, ..., k_m$ a nonincreasing sequence of nonnegative integers.
Check that $s$ is graphical thanks to Havel-Hakimi theorem
Represent each vertex by point with as many stubs on it as its degree
Let S be the set of stubs
while S is non empty do
    take a pair $s_1, s_2 \in$ S of distinct stubs and link them
    $S = S \setminus \{s_1, s_2\}$
end while
Output: A graph G such that $s$ is graphical for it.
```

2.3 Stochastic Block Model (SBM)

Stochastic blockmodel's first formulation goes back to 1983 [19], and is a model for networks characterized by block structure. It is part of the bigger category of Latent Space models [1]. By block structure, we mean that the nodes of the network are partitioned into subgroups called blocks (or communities) and that the distribution of the ties between nodes is dependent on the blocks to which the nodes belong. We

[1] Latent Space models are models in which a latent variable, that we are not able to see, explains some of the graph's behavior. In the case of SBM, each vertex $v_1, ..., v_n$ has an associated latent variable $u_1, ..., u_n$ which belongs to the sets of communities. Finding the community of appurtenance of a given vertex can be an interesting challenge.

may distinguish two types of SBM : the assortative, with higher probability of connection inside comunities than between, and dissortative, with higher probability of connection between communities than inside.

Let us give an example of a situation where the SBM may be appropriate. Consider a set of inhabitants of two cities (let us say Saint Germain au Mont d'Or and Prizren), and let us be interested in the friendship between two people. It should be obvious that as Saint Germain au Mont d'Or and Prizren are far far far away (1 888 km precisely), two people are way more likely to be friends if they are in the same city.

Thus it can be relevant to assign for every pair a probability p_1 to be friends if they are both Saint Germinois, a probability p_2 to be friends if they are both from Prizren and a probability q if one leaves in Saint Germain au Mont d'Or and the other in Prizren; with $p_1, p_2 >> q$ as two people are less likely to be friends if they are not in the same city. Note that this is an example of the assortative case.

Note that in 2.1.1 we explicated the law followed by the adjacency matrix of an ER $G(n, p)$ random graph. We can do the same for the SBM, to give a rigorous definition :

Definition 2.3.1 (k-term stochastic block model). Let $c \in \mathbb{R}^n$ be taking values in $\{1, ..., k\}$. The SBM for a $n \times n$ array A is specified in terms of connectivity matrix $P^{k\times k}$ as

$$A_{ij}|c \overset{iid}{\sim} \text{Bernoulli}(P_{c_i c_j})$$

for every $i, j \in 1, ..., n$ distinct, and $A_{ii} = 0$.

In words, the stochastic block model takes the following parameters:

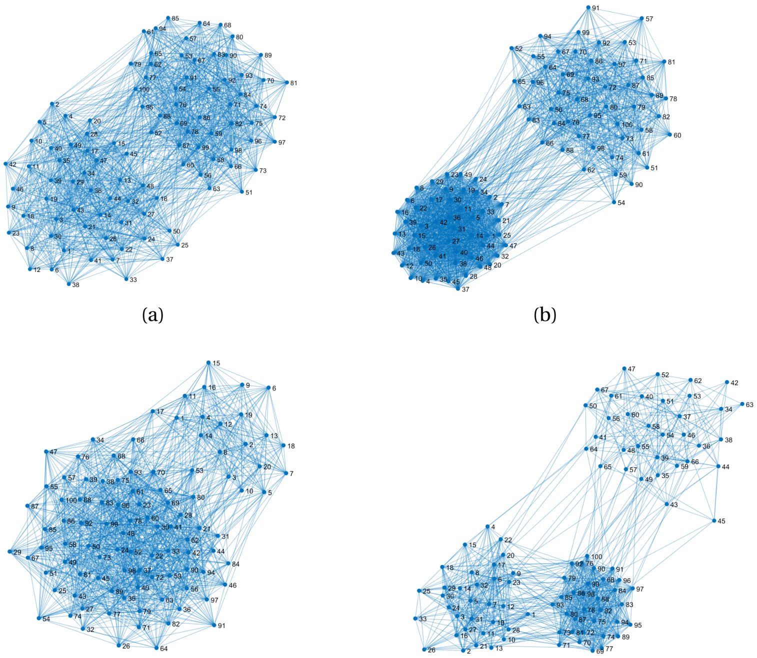
- The number n of vertices;
- a partition of the vertex set $\{1, ..., n\}$ into disjoint subsets $C_1, ..., C_k$ called *communities*
- a symmetric $k \times k$ matrix P of edges probabilities.

The edge set is then sampled at random as follows: any two vertices $u \in C_i$ and $v \in C_j$ are connected by an edge with probability P_{ij}. Going back to our introductory exampl the connectivity matrix P for it would be given as :

$$\begin{pmatrix} p_1 & q \\ q & p_2 \end{pmatrix}$$

with two communities C_{p_1} and C_{p_2}, a partition of the habitants with respect to their city (Saint Germain au Mont d'Or or Prizren).

Below, we add an illustration which is a result of SBM simulation in MATLAB with different P and C for $n = 100$.



(a)

(b)

(c)

(d)

Figure 5: Random graphs based on different SBM on 100 vertices generated using MATLAB

Note that all four pictures, we have 100 vertices but the separation into communities and associated probability matrix vary. Let P_i for $i \in \{a, b, c, d\}$ be the matrix associated with the corresponding picture, then we get :

$$P_a = \begin{pmatrix} 0.5 & 0.1 \\ 0.1 & 0.5 \end{pmatrix}, P_b = \begin{pmatrix} 1 & 0.05 \\ 0.05 & 0.5 \end{pmatrix}$$

$$P_c = \begin{pmatrix} 0.8 & 0.1 \\ 0.1 & 0.3 \end{pmatrix}, P_d = \begin{pmatrix} 0.5 & 0.01 & 0.1 \\ 0.01 & 0.3 & 0.05 \\ 0.1 & 0.05 & 0.08 \end{pmatrix}$$

Also, we see that in figures 5a and 5b, the population is divided into two equal communities of 50 individuals, whereas for figure 5c the first community (first with respect to the columns/rows order in P_c) counts 20 individuals and the second 80. Lastly, in figure 5d we have three communities counting respectively 33,33 and 34 individuals.

Figure 5a represents the basic situation, where two communities equal in size are `strongly` linked inside and `weakly` linked outside communities. Figure 5b is based on Figure 5a, but we doubled the probability associated with the first community (this can be seen on the plot as the bottom left cloud is way denser than the top right one). Also, the two clouds seem less linked; which is indeed the case as we divided the probability associated with an edge between the two communities by 2.

Figure 5c represents two communities this time not equal in size (this could apply to our Prizren/Saint Germain au Mont d'Or example, as Prizren is way bigger), and figure 5d a slightly more complex situation; with three communities C_1, C_2, C_3 linked as follow: C_3 is linked to C_1 $(p = 0.1)$ and to C_2 $(p = 0.05)$, but C_2 and C_1 are almost not linked at all (only a few edges on the graph, $p = 0.01$). Each community has its own distinct inside probability, respectively $0.5, 0.3, 0.8$.

2.4 Degree Corrected Stochastic Block Model (DCSBM)

Despite its utility, SBM with a fixed k is often not sufficient to model real-world graphs. In particular, due to the model assumption, all nodes within the same community in an SBM are exchangeable and hence have the same degree distribution. In comparison, nodes in real-world networks often exhibit a degree heterogeneity even when they belong to the same community. One way to accommodate degree heterogeneity is to introduce a set of degree-correction parameters $\{\theta_i : i = 1, ..., n\}$, one for each node, which can be interpreted as the popularity or importance of a node in the network. In probabilistic terms, θ_i controls the expected degree of vertex i. This motivates the definition of the DCSBM [21]:

Definition 2.4.1 (DCSBM). Under the same formalism as in DEFINITION 13, a graph G on $\{1,...,n\}$ is said to be built on the SBM if its adjacency matrix $A \in \mathbb{R}^{n\times n}$ verifies:

$$A_{ij} \overset{iid}{\sim} \text{Bernoulli}(\theta_i \theta_j P_{C(i)C(j)}) \tag{2}$$

In the example of the study of [21] of the "karate club" network [45], a club which have split into two different parts after internal problems, the DCSBM has reveal itself to be more performing, in the sense of community detection, than the simple SBM.

2.5 Mixed Membership Block Model (MMSBM)

A different perspective of considering degree heterogeneity, is taking into account the fact that some nodes may belong to several communities in the mean time. For instance, if we are interested in studying interactions between people based on their belonging in communities, it rarely happens that people belong to one and only one community. Even if we restrain our interest in a specific type of community, multi-belonging may occur. As an example, let us imagine that we are looking into relations of friendship in a high school based on hobbies of teenagers. Some may be into sports, music, and theatre together. Even if we restrain ourselves to only sports, some may practice several. This motivates an extension of the SBM which would allow individuals to belong to several communities in the meantime. That is where the MMSBM comes into play. It has emerged in the literature independently in various statistical applications settings like population genetics [32], text analysis [5] or survey data [14]. We gives its definition for directed graphs, as the one for undirected derives quite easily from it

Definition 2.5.1 (MMSBM). Let us use the same parameters as in Definition 2.3.1. Then a directed graph $G = (V, E)$ is said to be built on the MMSBM when :

- For every vertex $v \in V$:

  - we have an associated r dimensional mixed membership vector $\vec{\pi}_v \sim$ Dirilchet($\vec{\alpha}$), where r is the number of communities and $\vec{\alpha}$ a latent variable.

- For every pair of nodes $(u, v) \in V \times V$:
  - we have membership indicator for the initiator, $\vec{z}_{u \to v} \sim$ Multinomial($\vec{\pi}_u$)
  - we have membership indicator for the initiator, $\vec{z}_{v \to u} \sim$ Multinomial($\vec{\pi}_v$)
  - A determination of their interaction $A_{uv} \sim$ Bernouilli($\vec{z}_{u \to v}^{\top} P \vec{z}_{v \to u}$)

2.6 Exponential random graph models (ERGM)

ERGM [9] is a generative statistical network model whose ultimate goal is to present a subset of networks with particular characteristics as a statistical distribution. We call those characteristics statistics, or configurations. Often they are numbers of repeated subgraphs (such as complete graphs, cycles, ...) across the studied graphs. ERGMs are useful in statistical analysis of social networks, or knowledge graphs [2]. We can define the model as follows :

Definition 2.6.1 (ERGM). Let $\mathscr{Y}$ be the set of all possible graphs on a set of n vertices. A graph $y \in \mathscr{Y}$ is said to be built on the ERGM if it follows :

$$P(Y = y|\vec{\theta}) = \frac{exp(\vec{\theta}^{\top} s(\vec{y}))}{c(\vec{\theta})}$$

where $s(y)$ is a given vector of sufficient statistics depending of the observed graph, $\vec{\theta}$ is a vector of model parameters and $c(\vec{\theta}) = \sum_{y \in \mathscr{Y}} exp(\vec{\theta}^{\top} s(\vec{y}))$ is a normalising constant.

3 Graph limits/Graphons

Graphs are fundamentally discrete objects but some problems necessiate using sequences of graphs instead of a single graph. For instance, if we want to minimise the 4-cycle densities among graphs with

[2] An instance of a popular knowledge graph is the Google Knowledge Graph. This is a knowledge base from which Google serves relevant information in an infobox beside its search results. This allows the user to see the answer in a glance, as an instant answer. The data is generated automatically from a variety of sources, covering places, people, businesses, and more.

edge-density $p \in (0,1)$, we can prove that that the 4-cycle density is $\geq p^4$ and approached by a sequence of graph. However, there is no explicit graph that touches this bound, even if one can exhibit a sequence that approaches it. Rather than solving this problem in a sequential way, we would like to have a single analytic object that depicts the solution. This motivates the definition of Graph limits, or Graphons.

Definition 3.0.1 (Graphon). A *graphon* (short for graph function) is a symmetric measurable map given as

$$W : [0,1]^2 \to [0,1]$$

This object is of particular importance when in studying dense graphs. Graphons arise via two things. Firstly, they arise quite naturally as a limit of some sequence of random graphs. The link between graphons and dense graphs resides in two observations: the RGM defined by graphons generate almost surely dense graphs, and, by the *regularity lemma*, graphons capture the structure of arbitrary large dense graphs. Before stating the *regularity lemma*, or also called *Szemerédi's regularity lemma* we shall know what is the edge distribution between some parts that behave 'almost randomly' real meaning. The notion between the idea of 'almost random' is called ϵ-regularity. Let us begin with the definition of edge density.

Definition 3.0.2 (edge density). Let X, Y be disjoint subsets of the set of vertices of our graph. We define the edge density of the pair (X, Y) as:

$$d(X,Y) := \frac{|E(X,Y)|}{|X||Y|}$$

where $E(X, Y)$ stands for the set of edges that have one end vertex in X, and the other one in Y.

A pair (X, Y) as in the above definition is ϵ-regular if, whenever large subsets of each of X and Y are taken, the edge density of the pair of subsets is not too different from the edge density of the pair of parts. Let us be more formal:

Definition 3.0.3 (ϵ-regularity). Take a pair of vertex sets X and Y, and $\epsilon > 0$: . Then the pair is called $\epsilon - regular$, if $\forall A \subseteq X$, $\forall B \subseteq Y$ that satisfy $|A| \geq \epsilon|X|$, $|B| \geq \epsilon|Y|$, the following holds :

$$|d(X,Y) - d(A,B)| \leq \epsilon$$

Considering the above-mentioned definition, we may want to define an ϵ-regular partition as one where each pair of parts is ϵ-regular. However, some graphs, like for instance half graphs, require lots of pairs of partitions (but small relatively to the total number of pairs) to be irregular. So it makes more sense to define ϵ-regular partitions to be such as most pairs of parts (but not all) are ϵ-regular.

Definition 3.0.4 (ϵ-regular partitions). We say that a partition of V into k sets $P = \{V_1, ..., V_k\}$ is an ϵ-regular partition whenever:

$$\sum_{(V_i,V_j) \text{ not } \epsilon\text{-regular}} |V_i||V_j| \leq \varepsilon|V(G)|^2$$

Now we have enough definitions to give the lemma, but let us motivate it before. Theoretical reasons coming from this lemma strongly support the fact that graphons can be very well approximated by blocks. The lemma suggests that a large enough graph will almost behaves like it had been drawn from a SBM with k communities. But we may face the problem of k being too large, and in order to encounter this problems we shall follow regularizing strategies to infer a good enough approximation with a reasonable k [44].

Lemma 3.1 (Szemerédi's Regularity Lemma). $\forall \epsilon > 0$ and $\forall m \in \mathbb{N}$, $\exists M \in \mathbb{N}$ such that if $G = (V, E)$ is a graph with $M \leq |V|$, $\exists k \in \mathbb{N}$ verifying $m \leq k \leq M$ and an ϵ-regular partition of the vertex set of G into k sets.

Going back to graphons, they also arise as the fundamental defining objects of exchangeable random graph models, and exchangeability is the natural thing to consider when working with unlabelled graphs. We will need the following definitions :

Definition 3.0.5 (Exchangeable random variables). Let $\{X_i\}$, $1 \leq i < \infty$, be a sequence of binary random variables. It is *exchangeable* if

$$P(X_1 = e_1, ..., X_n = e_n) = P(X_1 = e_{\sigma(1)}, ..., X_n = e_{\sigma(n)})$$

for all $n \in \mathbb{N}$, all permutations $\sigma \in S_n$ and all $e_i \in \{0, 1\}$.

Definition 3.0.6 (Exchangeable random array). Matrix or array of random variables whose distribution is invariant under permutation of row and column indices.

Definition 3.0.7 (Separately/jointly exchangeable random variables). Let $\{X_{ij}\}$, $1 \leq i, j < \infty$, be binary random variables. They are *separately exchangeable* if

$$P(X_{ij} = e_{ij}, 1 \leq i, j \leq n) = P(X_{ij} = e_{\sigma(i)\tau(j)}, 1 \leq i, j \leq n)$$

for all $n \in \mathbb{N}$, all permutations $\sigma, \tau \in S_n$ and all $e_{ij} \in \{0,1\}$. They are *jointly exchangeable* if the above holds in the special case $\tau = \sigma$.

If we follow the two following steps, a graphon can define an exchangeable random graph model :

- each vertex j of the graph has an associated independent random value $u_j \sim U(0,1)$
- The probability of belonging of each edge (i, j) to the graph is $W(u_i, u_j)$. In other words

$$A_{ij}|W, u_i, u_j \overset{iid}{\sim} Ber(W(u_i, u_j))$$

Now let us give an important result, proved independently by Aldous and Hoover in 1981 and 1979, respectively.

Theorem 1 (Aldous-Hoover).— *Let* $A \in \{0,1\}^{n\times n}$ *be a jointly exchangeable random array and* $\xi \in [0,1]^n$ *have independent* $U(0,1)$ *entries. Then there is a function* $f(\xi_i, \xi_j; \alpha)$ *such that*

$$P(A_{ij} = 1|\xi, \alpha) = f(\xi_i, \xi_j; \alpha)$$

The usefulness of this result reside in its ability to link the concept of exancheability to network modelling. A clear example will be given in 4.2.2.

3.1 Representation of various random graphs models via graphons

3.1.1 Erdős–Rényi Model

This is the simplest example of a graphon, where W is defined as :

$$W(x,y) = p, \quad \forall (x,y) \in [0,1]^2 \quad \text{and} \quad p \in [0,1] \tag{3}$$

3.1.2 Stochastic Block Model

Intuitively, we will have a piecewise constant graphon on the block diagonal of the unit square, and 0 outside this diagonal. Rigorously, we divide $[0,1]^2$ into $k \times k$ blocks, and define W as :

$$W(x,y) = p_{lm}, \tag{4}$$

whenever $(x,y) \in (l,m)^{th}$ block, with $p_{lm} \in [0,1]$, and being 0 whenever $l \neq m$.

3.1.3 Configuration Model

The corresponding graphon is, for $(x,y) \in [0,1]^2$:

$$W(x,y) = g(x)g(y)$$

where g is a function from $[0,1]$ from $[0,1]$.

4 Inference

4.1 Inference of SBM parameters

4.1.1 Spectral methods

In this subsection, we will take a closer look into the adjacency and Laplacian matrices of the graphs we will study, in particular their eigenvalues as they are one of the most useful algebraic properties of the graph. The goal is to extract some interesting properties regarding the structure of the graphs. The field that deals with their study is called Spectral Graph Theory, and we may start with one of the most fundamental definitions of the field :

Definition 4.1.1 (Spectrum of a graph). Let A be the adjacency matrix of an undirected graph G on n vertices. Let us denote by $\lambda_1 \geq ... \geq \lambda_n$ the n real eigenvalues (it can be proved that they are real). Then these eigenvalues associated with their multiplicities compose the spectrum of G.

As said previously, the eigenvalues, and therefore consequently the spectrum of a graph, contains a lot of information about it. By looking at the spectrum, one can tell the key properties of the graph such as connectivity, regularity, number of connected components, ect. We give several such information contained in the spectrum in the lemma below :

Lemma 4.1. Consider any undirected graph G with adjacency matrix A. Then we have the following properties :

- If G is d-regular, then $\lambda_1 = d$ and $|\lambda_i| \leq d$ for $i = 2, ..., n$.
- G is connected iff $\lambda_2 < d$, i.e., the eigenvalue d has multiplicity 1. Moreover, the number of connected components of G equals the multiplicity of eigenvalue d.
- If G is connected, then G is bipartite iff $\lambda_n = -d$, with d the largest eigenvalue.

Now the natural question arising is how to use the spectrum in real-life problems. It appears that in the field of data science, one of the most popular clustering algorithms, the spectral clustering algorithm, makes an interesting use of it. Before going deeper into this, we shall first define what clustering is, although we are aware that it is ill defined [22]:

Definition 4.1.2 (Clustering). The task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense) to each other than to those in other groups (clusters).

Now let us give more context. Suppose we have a set of n data points $(x_1, ..., x_n)$ and a relation of similarity between each pair (x_i, x_j) of points quantified by some $s_{ij} \geq 0$, and we wish to divide our set into k

clusters such that the elements inside of each cluster are as similar as possible. A clever thing to do is to represent our data by a *similarity graph* where each node is associated with a point, and each edge is determined by the similarity s_{ij} of its extremities, with existence of the edge if and only if it is bigger than a certain threshold, and if it exists it is weighted by s_{ij}. [3] So our clustering problem appears quite clear : we want to partition our newly built graph into clusters such that the weights inside clusters are maximal, and outside minimal.

With this formality applied to our problem, let us introduce the intuitive idea of the spectral clustering algorithm. It aims to select some relevant eigenvectors of the Laplacian matrix of the similarity graph, in order to apply a clustering method to them. A careful selection of the eigenvectors can consequently lead to a substantial reduction of the dimension in which we are working, and thus the computations.

A widely used clustering method is the *k-means*, where k is the required number of clusters. Intuitively, we want to find k groups of data points, containing all of our points, such that each of the group is made of the closest points from its centroid. We give a rigorous defintion below :

Definition 4.1.3 (k-means). Given a set of n d-dimensional data points $(x_1,...,x_n)$, we want to find a partition $\mathrm{S} = \{\mathrm{S}_1,...,\mathrm{S}_k\}$ such that it minimizes the following function :

$$\underset{\mathrm{S}}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{x \in \mathrm{S}_i} \left\| x - \mu_i \right\|^2$$

where μ_i is the centroid rigorously defined as :

$$\mu_i = \frac{1}{|\mathrm{S}_i|} \sum_{x \in \mathrm{S}_i} x$$

To obtain a satisfactory partition, we may follow the simple following algorithm :

Now that we know the idea behind the spectral clustering algorithm, and have all the necessary tools to make it work we give the pseudocode of the algorithm, from [40] :

Spectral clustering has shown to be a very efficient machine learning method, and gained more popularity in the field after [35] provided a novel solution to the perceptual grouping problem in vision. It is also

[3] Note that an often used form of similarity matrix is $s_{ij} = e^{-(dist_{ij})^2}$, with $dist$ a distance to choose

Algorithm 2 k-means

```
Input : $n$ data points $(x_1, ..., x_n)$, and a positive integer $k$.
Start with $k$ new points $(y_1, ..., y_k)$.
Build $k$ clusters $S_i$ around the associated $y_i$ by adding $x_j$ to the cluster $S_i$ whenever $y_i$ is the closest
  $y$ from $x_j$.
Set $y_i = \frac{1}{|S_i|} \sum_{x \in S_i} x$
Repeat until convergence
Output: A partition S that minimizes the k-means function.
```

Algorithm 3 Spectral clustering

```
Input : Similarity matrix $S \in \mathbb{R}^{n \times n}$, number k of clusters to construct.
Construct a similarity graph as described above. Let A be its weighted adjacency matrix.
Compute the Laplacian L of the graph, and $u_1, ..., u_k$ its first $k$ eigenvectors.
Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1, ..., u_k$ as columns.
For $i = 1, ..., n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i-th row of U.
Cluster the points $(y_i) i = 1, ..., n$ in $\mathbb{R}^k$ with the k-means algorithm into clusters $C_1, ..., C_k$.
Output: Clusters $A_1, ..., A_k$ with $A_i = \{j | y_j \in C_i\}$.
```

worth noticing the work of [43], [3] and [27]. This is a recent topic as many of the relevant publications about it are dated after 2000 (see [39],[3] and [40]). Consistency of the method has been investigated in [41], in which they proved that the eigenvectors of the Laplacian, and proved that, under standard assumptions, they converge to eigenfunctions of some limit operator.

4.1.2 Application of the spectral clustering algorithm for data points

In what this section we shall apply the spectral clustering algorithm to a set of 190 randomly generated 2-dimensional data points divided in four clusters of various sizes. We will follow the steps below :

- First generate data points randomly already in clusters, and plot them such that the clusters are visible.
- Compute the distance between each pair of data points.
- Compute the similarity matrix using the distances via the formula given in footnote 3.
- Plot the similarity graph using a certain threshold to connect to points [4].

[4] There is another way to do it. We could have linked every pair of points by a weighted edge representing their distance, and then choose a certain threshold (for the weight) to decide the clusters in which the points would have been. However, this is computationally far worse as we would need to work with a bigger (in fact complete) graph. That is why we did not choose this way.

- Find the number of connected components k of the similarity graph, and apply the spectral clustering algorithm with this k for the k-means.

Let us begin with the first step. We generate four sets of random points : two noisy circles, a small circle of 50 points inside a big one of 100 points; and two clusters outside of the circles of 20 points, with x and y coordinates following Gaussian of means respectively -6 and 6 and standard deviation 0.5. This gives us Figure 6.

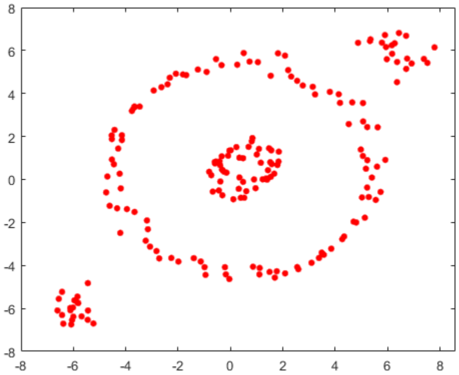


Figure 6

Next, we computed the distances and the similarity matrix, and generated the similarity graphs by using a similarity threshold of 0.2, meaning that two vertices are connected in the similarity graph if and only if their associated points x_i, x_j have a similarity coefficient s_{ij} greater than 0.2. Notice that this makes sense as a coefficient s_{ij} of S is inversely proportional with $dist(x_i, x_j)$. Hereafter you can find the similarity graph obtained in Figure 7.

Finally, we applied the spectral clustering algorithm via the MATLAB function *spectralcluster*, using an unnormalized Laplacian and the k-means method where $k = 5$ is the number of connected components of the similarity graph above, that was found using the *conncomp* MATLAB function. We obtained Figure 8 , where each clusters is represented by its color.

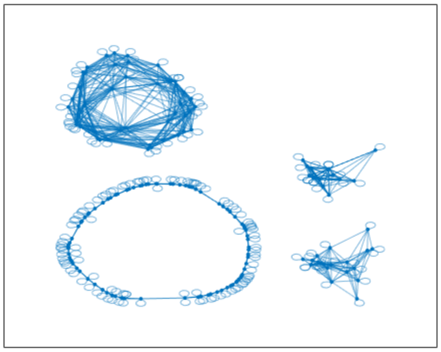


Figure 7

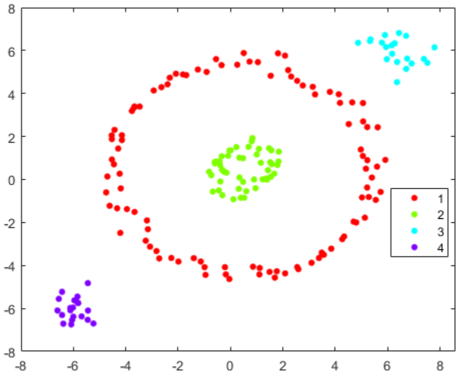


Figure 8

Notice that above we specified that the Laplacian is *unnormalized.* This is because another way to apply the spectral clustering algorithm is through the *normalized* Laplacian. There are two normalized Laplacians:

- the symmetric normalized used in [27]

$$\mathrm{L}^{\mathrm{norm}} := \mathrm{I} - \mathrm{D}^{-1/2}\mathrm{A}\mathrm{D}^{-1/2}.$$

- the random walk (or left) normalized Laplacian used in [35]

$$\mathrm{L}^{\mathrm{rw}} := \mathrm{D}^{-1}\mathrm{L} = \mathrm{I} - \mathrm{D}^{-1}\mathrm{A}.$$

4.1.3 Application of the spectral clustering algorithm for a SBM

In this subsection we generate some graphs via a SBM with 150 nodes separated in 3 clusters of 50 nodes, and see what happens when we change some entries of the probability matrix P. Through the four following examples, we will give $k = 3$ as the number of clusters for the spectral clustering algorithm, and the matrix will look like this :

$$\mathrm{P}_x = \begin{pmatrix} 0.7 & x & 0.01 \\ x & 0.7 & 0.01 \\ 0.01 & 0.01 & 0.7 \end{pmatrix}$$

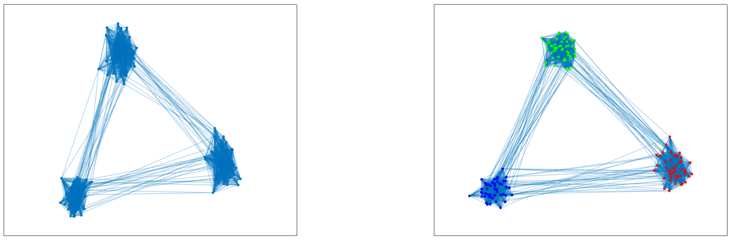
where x is going to vary.

Starting with $x = 0.01$ gives us the following plots shown in Figure 9.

We see that the spectral clustering algorithm is perfectly able to detect the clusters. Multiplying the variable by 30 still gives us a good partition, see Figure 10.

Even if the two communities on the left are more linked than in the first example, the edge forest between them is not dense enough to alter the spectral clustering algorithm. But when x becomes greater than 0.4, our results indicate a bad partitioning. For instance, the following has been obtained with $x = 0.45$, see Figure 11.

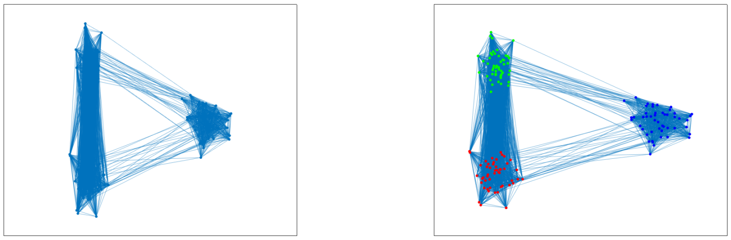
We clearly see an asymmetry between the green and the red community, where the first one seems to



(a) Graph

(b) Partition found

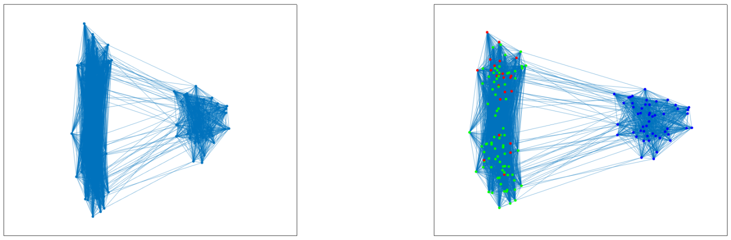
Figure 9: Spectral clustering algorithm applied on a graph following the SBM based on $P_{0.01}$



(a) Graph

(b) Partition found

Figure 10: Spectral clustering algorithm applied on a graph following the SBM based on $P_{0.3}$



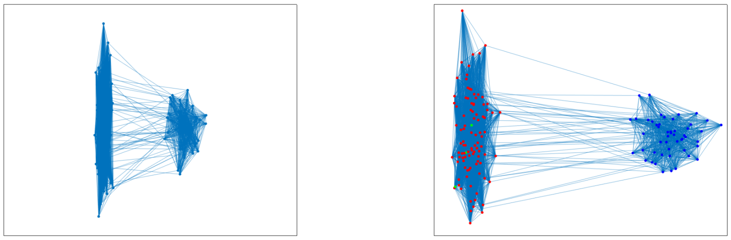
(a) Graph

(b) Partition found

Figure 11: Spectral clustering algorithm applied on a graph following the SBM based on $P_{0.45}$

take over, whereas according to our SBM the two should be equals in number. This is because the two left blocks are too strongly linked to each other.

Finally, when x is very close to the probability value inside clusters, this leads to worse detection, as in the Figure 12 below.



(a) Graph

(b) Partition found

Figure 12: Spectral clustering algorithm applied on a graph following the SBM based on $P_{0.6}$

From this we can clearly see that the algorithm wants to partition our set of nodes into $k = 2$ clusters, instead of the $k = 3$ we fed him. In the eyes of the algorithm, the third community is artificial. This is not surprising, as the high value of x makes the cluster detection very difficult.

What happens in this toy example can find justifications in works that have been conducted about the thresholds for the parameters regarding exact recovery of the Stochastic Block Model. This leads us to the following subsubsection.

4.1.4 Thresholds

The above experiment raises questions about bounds for exact recovery of the clusters in a SBM. In the case of a SBM with two equally-sized communities with same internal probability p, one can find such bounds in the literature. This problem has attracted a lot of attention since 1990s, especially providing lower bounds of $|p-q|$ for exact recovery, where p and q are respectively the internal and external probabilities of edge. For instance, [13] used the min-cut algorithm via degrees to show that $p-q = \Omega(1)$ ensure recovery. The same result has been shown in [37] using the EM algorithm, and more recently in [31] using a spectral method. The result $p - q = \Omega(n^{-1/6+\epsilon})$ has been shown in [20] via Metropolis algorithm, and

$p - q = \Omega(n^{-1/2+\epsilon})$ result has been shown in [12] using the augmentation algorithm.

However [1] obtained a better bound that is tight. Their result is expressed in the two following theorems, in which we let $p = \alpha\frac{log(n)}{n}$ and $q = \beta\frac{log(n)}{n}$, and $\beta < \alpha$.

Theorem 2.— *Let* $0 \leq \beta < \alpha$. *If* $\frac{\alpha+\beta}{2} - \sqrt{\alpha\beta} < 1$, *or equivalently, if either* $\alpha + \beta < 2$ *or* $(\alpha - \beta)^2 < 4(\alpha + \beta) - 4$ *and* $\alpha + \beta \geq 2$, *then ML fails in recovering the communities with probability bounded away from zero.*

Theorem 3.— *If* $\frac{\alpha+\beta}{2} - \sqrt{\alpha\beta} > 1$, *i.e., if* $\alpha+\beta > 2$ *and* $(\alpha-\beta)^2 > 4(\alpha+\beta)-4$ *and* $\alpha+\beta \geq 2$, *then the maximum likelihood estimator exactly recovers the communities (up to a global flip), with high probability.*

In the latter the term with high probability refers to the probability going to 1 as n goes to infinity.

[1] gives an efficient semidefinite programming ML relaxation-based algorithm and proves its ability to recover the communities near the bound. Furthermore, their numerical experiments indicate that the algorithm could attain it. Another efficient algorithm which touches the bound is obtained by means of partial recovery and local improvement procedure.

4.1.5 Modularity maximization

Clusters and community structure appear naturally in real-life networks [26], and their detection can be primordial. For instance, a disease that appears in a very connected social cluster may travel very fast inside of it, but may never expand to other clusters depending on how isolated the infected one is.

Modularity is a graph clustering index. This means that this is a measure quantifying the strength of division of a graph into potential clusters. A high modularity is associated with strong (in the sense of numerous for simple graphs, and heavy for weighted graphs) connections between the nodes inside clusters and weak connections between nodes situated in different clusters. Thus it can be a powerful tool to detect community structure in a network. Nevertheless, it admits limitations, such as its inability to detect small communities [16] [23].

Roughly speaking, we can define it as : given a partition of the vertices, the proportion of the edges that fall inside the given clusters minus the expected proportion if the edges were distributed at random. For undirected and unweighted graphs, this value belongs to $[-1/2, 1]$ [8]. We give a rigorous definition :

Definition 4.1.4 (Modularity). Let G be an order n and size m graph, with adjacency matrix A, a size n degree vector d, and c the n-sized vector of community membership, i.e. c_i determines the community of v_i. Then, we have the following formula for the modularity from [11], a generalization of the case of two communities in [26]:

$$Q = \frac{1}{2m} \sum_{vw \in V \times V} \left[A_{vw} - \frac{d_v d_w}{2m} \right] \delta(c_v, c_w),$$

where δ is the Kronecker function that returns 1 if $c_v = c_w$ and 0 otherwise.

Let us explain a little bit how this formula was derived. As said previously, we shall compute the number of edges between a given pair of vertices minus the expected number. The actual number is A_{vw}, and the expected number is $\frac{d_v d_w}{2m-1}$. We give a probabilistic justification of this as a lemma :

Lemma 4.2. The expected number of edges between two vertices v and w is $\frac{d_v d_w}{2m-1}$.

PROOF : Consider the k_v stubs incident at v, and associate to each of them an indicator variable for $i = 1, ..., k_v$ defined as

$$I_i^{(v,w)} = \begin{cases} 1 & \text{if the i-th stubs connect } v \text{ to } w \\ 0 & \text{else} \end{cases}$$

Then as the ith stub of v can be connected to every other $2m-1$ stubs, but only the k_w of w connecte the two vertices, we get

$$E[I_i^{(v,w)}] = P(I_i^{(v,w)} = 1) = \frac{k_w}{2m-1}$$

So the total number of edges T_{vw} between v and w is nothing more than the sum of the $I_i^{(v,w)}$ and thus we finally get the following result :

$$E[T_{vw}] = E\left[\sum_{i=1}^{k_v} I_i^{(v,w)} \right] = \sum_{i=1}^{k_v} E[I_i^{(v,w)}] = \sum_{i=1}^{k_v} \frac{k_w}{2m-1} = \frac{k_v k_w}{2m-1},$$

which concludes the proof. □

Going back to the definition, we see a sum of $A_{vw} - \frac{d_v d_w}{2m}$, and not $A_{vw} - \frac{d_v d_w}{2m-1}$. This is because one can make this approximation when m large, which is often the case with the graphs we consider in this paper, without losing important information. Finally, a term $A_{vw} - \frac{d_v d_w}{2m}$ is counted in the sum if and only if both v and w belong to the same cluster in the given partition, hence the factor $\delta(c_v, c_w)$. The $\frac{1}{(2m)}$ is a normalizing constant.

4.1.6 Maximum Likelihood Estimation (MLE)

Given an observation of a graph drawn from SBM, a powerful tool to help us recovering the latent blocks is the Maximum Likelihood Estimation. We recall its definition from the preliminaries (see 0.0.11). According to Definition 2.3.1, for A, the adjacency matrix of our network, we get the following likelihood function. Notice that this expression is derived using independence of the A_{ij} condditioned on c

$$L(P; c) = \prod_{i<j} P_{c_i c_j}^{A_{ij}} (1 - P_{c_i c_j}^{(1-A_{ij})})$$

and taking the natural logarithm of this expression yields the log-likelihood function

$$l(P; c) := \sum_{i<j} A_{ij} log(P_{c_i c_j}) + (1 - A_{ij}) log(1 - P_{c_i c_j})$$

that we can re-writte as

$$l(P; c) = \sum_{a \leq b} h_{ab} \bar{A}_{ab}(c) log(P_{ab}) + h_{ab}(1 - \bar{A}_{ab}(z)) log(1 - P_{ab}) \tag{5}$$

with

$$h_{ab} := \sum_{i<j} \mathbb{1}_{\hat{z}_i}(a) \mathbb{1}_{\hat{z}_j}(b)$$

and

$$\bar{A}_{ab}(c) := \frac{\sum_{i<j} A_{ij} \mathbb{1}_{\hat{c}_i}(a) \mathbb{1}_{\hat{c}_j}(b)}{\sum_{i<j} \mathbb{1}_{\hat{c}_i}(a) \mathbb{1}_{\hat{c}_j}(b)} = \frac{\sum_{i<j} A_{ij} \mathbb{1}_{\hat{c}_i}(a) \mathbb{1}_{\hat{c}_j}(b)}{h_{ab}}$$

Let us study the first and second derivatives of l with respect to P

$$\frac{\partial l(P; c)}{\partial P_{ab}} = h_{ab} \Big(\frac{\bar{A}_{ab}(c)}{P_{ab}} - \frac{1 - \bar{A}_{ab}(c)}{1 - P_{ab}} \Big) \tag{6}$$

$$\frac{\partial^2 l(\mathrm{P};c)}{\partial \mathrm{P}_{ab}{}^2} = -h_{ab}\Big(\frac{\bar{\mathrm{A}}_{ab}(c)}{\mathrm{P}_{ab}^2} - \frac{1-\bar{\mathrm{A}}_{ab}(c)}{(1-\mathrm{P}_{ab})^2}\Big) < 0 \tag{7}$$

Thanks to equation (7), we see that for c fixed, l admits a maximum. We know that this maximum is attained when the first derivative is set to 0, and as

$$\frac{\partial l(\mathrm{P};c)}{\partial \mathrm{P}_{ab}}\Big|_{\mathrm{P}=\bar{\mathrm{A}}(c)} = 0$$

we obtain $\hat{\mathrm{P}}_{ab} = \bar{\mathrm{A}}_{ab}(c)$. Notice that when the indices are i, j they range from 1 to n, but when the indices are a, b they range from 1 to k. Then the profile likelihood on c is

$$l(c) = l(\hat{\mathrm{P}};c) = \sum_{i<j} \mathrm{A}_{ij} log(\hat{\mathrm{P}}_{c_i c_j}) + (1-\mathrm{A}_{ij}) log(1-\hat{\mathrm{P}}_{c_i c_j}) \tag{8}$$

One would obtain $\hat{c}$ as a result of equation (8)'s maximization, but unfortunately this is computationally infeasible.

4.1.7 Advantages/disadvantages of the above-mentioned inference methods

In this subsubsection, we summarise some advantages and disadvantages of modularity maximization methods, MLE and spectral clustering algorithm.

Let us start with modularity maximization based methods, which have two big drawbacks: they can not detect small communities [16, 23], are not statistically consistent, and find communities in their own null model which is the Configuration Model. Consequently, they fail to obtain statistically significant community structure in empirical networks, but some solutions to this problem have been recommended in [30].

On the other hand, methods based on Maximum Likelihood Estimation neither suffer the statistical issue nor the resolution issue faced by modularity, but in the worst case it is computationally intractable for large n.

To cover up the spectral clustering algorithm, we may recall that in part 4.1.2 we evoke *normalized* and *unnormalized* Laplacian in order to apply the algorithm. Surprisingly, it has been shown in [24] that convergence of the algorithm towards the true partition in the unnormalized case is difficult to handle, and necessitates additional conditions that are not always met. However, [24] also shows that under standard assumptions, normalized spectral clustering always converges.

4.2 Graphon estimation

4.2.1 Non-parametric graphon estimation

When faced with a complex network, the first wish is to know more about its underlying structure. In this section, we will summarize the non parametric approach to graphon estimation by [44]. The authors do so by using maximum likelihood to obtain such an estimator $\hat{\mathrm{W}}$. As usual, n and k will be the number of nodes and the number of communities respectively. A grouping of n nodes in k communities can be represented by a vector $h \in \{2, ..., n\}^k$ with the property that $\sum_{i=1}^{k} h_i = n$. Then it makes sense to see $\frac{h}{n}$ as the probability mass function of a random variable determined by the following cumulative distribution function H :

$$\mathrm{H}(x) = \frac{1}{n} \sum_{i=1}^{\lfloor x \rfloor} h_i$$

for $0 \leq x \leq k$, so that $\mathrm{H}(x) \in \{0, \frac{h_1}{n}, \frac{h_1+h_2}{n}, ..., 1\}$, authors of [44] denote its generalized inverse H^{-1} as follows

$$\mathrm{H}^{-1}(y) = \inf_{y \in [0,k]} \{\mathrm{H}(y) \geq y\}, 0 < y \leq 1,$$

so that $\mathrm{H}^{-1}(y) \in \{1, ..., k\}$.

One must notice that an important difficulty inherent to this task is that the relation between the ordering of A and the ordered random sample of the $u_j \sim \mathrm{U}[0,1]$, which is hidden. To overcome this, $\hat{\mathrm{W}}$ must be defined by re-indexing the columns and row of A via some permutation σ of $\{1, ..., n\}$ before grouping them following H. Following this idea, the authors define

$$\bar{\mathrm{A}}_{ab} = \frac{1}{h_a(h_b - {}_b)} \sum_{j=n\mathrm{H}(b-1)+1}^{n\mathrm{H}(b)} \sum_{i=n\mathrm{H}(a-1)+1}^{n\mathrm{H}(a)} \mathrm{A}_{\sigma(i)\sigma(j)},$$

for a, b ranging from 1 to k. Then they get this expression [44]

$$\hat{W}(x,y;h) := \frac{1}{\hat{\rho}_n}\bar{A}_{H^{-1}(x)H^{-1}(y)}$$

for $0 < x, y < 1$. The $\hat{\rho}_n$ is the same as in 4.2.2.
It has been shown in [44] that under this framework, if W is Holder continuous, and $\hat{W}$ is fitted by block-model maximum profile likelihood estimation, then $\hat{W}$ will be consistent, provided that $\rho_n = \omega(\frac{log(n)^3}{n})$.

4.2.2 Network histogram

In this section, we will discuss results of [28] which show that if the network we are considering is unlabelled, the stochastic block model is a universal way to approximate an exchangeable network's underlying structure. SBM reduces the number of parameters from $\binom{n}{2}$ to $\binom{k}{2}$, with $k << n$, but as discussed previously, a common problem is that when n grows large, it does not remain reasonable to suppose that understanding the interactions between those fixed k blocks is enough to have a sufficient overview of the network. That is why we went through different extensions of the SBM, such that the DCSBM and MMSBM. However, it has been shown in [28] that the most natural way to capture more information is to simply add blocks to the model, so that k grows with n, and authors discussed the choice of optimal bandwidth. This leads to the **network histogram** [28], a nonparametric statistical approximation obtained by fitting a SBM to a network. Doing this does not require to assume that the network has been generate via a stochastic block model.
Given a network of n nodes defined by its adjacency matrix A, we can model it hierarchically by appealing to the Aldous-Hoover Theorem 1. We do so by using the three following elements :

- a normalized graphon $W(x,y)$, i.e with the particularity that $\int\int_{(0,1)^2} W(x,y)dxdy = 1$.
- n independent random variables associated to each vertex $j : u_j \sim U(0,1)$.
- for n, a deterministic scaling constant $\rho_n = \frac{\mathbb{E}(\sum_{i<j} A_{ij})}{\begin{pmatrix} n \\ 2 \end{pmatrix}} > 0$.

For each n, our simple stochastic network model is then

$$A_{ij}|u_i, u_j \overset{iid}{\sim} Ber(\rho_n W(u_i, u_j)).$$

Thus it follows that,

$$\mathbb{P}(A_{ij} = 1) = \mathbb{E}(A_{ij}) = \rho_n \int\int_{(0,1)^2} W(x, y)dxdy = \rho_n,$$

and so the estimate for ρ_n is the following

$$\hat{\rho}_n = \frac{\sum_{i<j} A_{ij}}{\binom{n}{2}}. \tag{9}$$

To summarize the network completely it remains to estimate W. To do so, as previously mentioned, authors fit a SBM to the network realisation. It has pre-specified bandwidth denoted as h. Let us write $n = kh+r$ where r is the remainder in the Euclidean division of n by h, and k is the dividend. We also need a community membership vector c of length n, with each of this component being an integer ranging from 1 to k. We want c to represent an arrangement of the n nodes into k communities : $k-1$ of size h, and one last of size $h+r$. Thus the authors define the set of possible c as

$$C_k = \{c \in \{1, ..., k\}^n \text{such that } h \text{ components belong to } \{1, ..., k-1\} \text{ and } h+r \text{ equal } k\}$$

Then, they estimate c from A by the method of maximum likelihood [28]

$$\hat{c} = \underset{c \in C_k}{\text{argmin}} \sum_{i<j} (A_{ij} log(\bar{A}_{c_i c_j}) + (1 - A_{ij}) log(1 - \bar{A}_{c_i c_j})).$$

Where $\bar{A}_{ab}$ is called the bin height, representing the proportion of edges present between community a and community b. Thus a, b range from 1 to k. The rigorous definition is

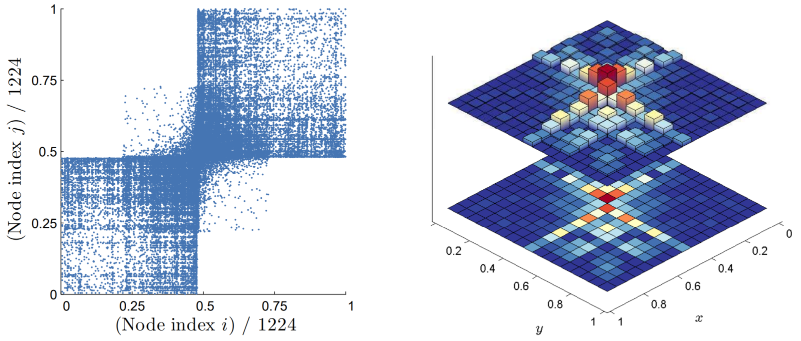
$$\bar{A}_{ab} := \frac{\sum_{i<j} A_{ij} \mathbb{1}_{\hat{c}_i}(a) \mathbb{1}_{\hat{c}_j}(b)}{\sum_{i<j} \mathbb{1}_{\hat{c}_i}(a) \mathbb{1}_{\hat{c}_j}(b)}. \tag{10}$$

Finally, combining equations (10) and (9) they obtain the following estimator for W:

$$\hat{\mathrm{W}}(x,y;h) := \hat{\rho}_n^{+}\bar{\mathrm{A}}_{\min(\lceil nx/h\rceil,k)\min(\lceil ny/h\rceil,k)},\ 0 < x, y < 1$$

where h is the predetermined bandwidth, and $0 < x, y < 1$. $\hat{\rho}_n^{+}$ is the generalized inverse.

Below, we add figures from [28] that show an explicit transformation from an adjacency matrix to a network histogram for making things more visual to the reader [28].



(a) Adjacency matrix

(b) Network histogram

Figure 13: Adjacency matrix of the political weblog data & and associated network histogram [28]

In this example, the network (here given in adjacency matrix form) represents the political weblog of [2], where each of the 1224 nodes represents a blog, and an edge exists between two nodes if at least one of the corresponding blogs' front page links to the other.The x and y axis are normalized so they range from 0 to 1. To obtain 13b from 13a, [28] divided the $n = 1224$ blogs into $k = 17$ communities of size $h = 72$. This yields a division of the unit square into $17 \times 17 = 289$ squares-bins. Each bin is determined by the two segments of the unit segment (each segment represents a community), and thus represent the interaction between the two communities given by the segments. The height of each histogram bin is determined by the number of edges present between the two communities. For visualisation purposes, the height in 13b is the squared root of $\hat{\mathrm{W}}$. The detailed results of [28] show the universality of SBM when it comes to approximating an exchangeable graph.

5 Conclusion

In conclusion, we delved into a variety of notable random graph models, and defined them rigorously. We also generated realisations of some random graphs models in sections 2.1, 2.3 and 2.2. We then formalized the notion of graph limits or graphons, and matched the previously seen random graph models to its graphon. Afterwards, we discussed inference methods and detailed some of the methods and algorithms aiming to recovering hidden information from observations of networks, such as the spectral clustering algorithm, maximum likelihood estimation method or network histogram. In particular, we applied the spectral clustering algorithm to some MATLAB-generated data in section 4.1.2 and 4.1.3. Finally, we looked in graphon estimation. Particularly, we summarized a nonparametric approach to graphon estimation by [44] and discussed the results of [28] that show stochastic blockmodel to be a universal way of approximating exchangeable graphs.

6 Acknowledgments

In this section, I would like to thank warmly Anda Skeja who supervised this project, for her pedagogy, for giving a coherent structure to the project, for letting taking my time sometimes to understand concepts that were at first glance not intuitive for me, for freeing time to meet every week, and for bearing my (terrible) french accent. I also would like to thank her for revising my motivation letter and my CV, thing that she was not obligated to do, but did anyway out of sheer kindness. To finish, I would like to thanks Sofia Olhede for letting me do this project that was originally reserved for master's students, even though I am in bachelor.

References

[1] Emmanuel Abbe, Afonso S Bandeira, and Georgina Hall. “Exact recovery in the stochastic block model”. In: *IEEE Transactions on information theory* 62.1 (2015), pp. 471–487.

[2] Lada A Adamic and Natalie Glance. “The political blogosphere and the 2004 US election: divided they blog”. In: *Proceedings of the 3rd international workshop on Link discovery*. 2005, pp. 36–43.

[3] Francis Bach and Michael Jordan. “Learning spectral clustering”. In: *Advances in neural information processing systems* 16 (2003).

[4] Albert-László Barabási and Réka Albert. “Emergence of scaling in random networks”. In: *science* 286.5439 (1999), pp. 509–512.

[5] Kobus Barnard et al. “Matching words and pictures”. In: *The Journal of Machine Learning Research* 3 (2003), pp. 1107–1135.

[6] Danielle S Bassett and Olaf Sporns. “Network neuroscience”. In: *Nature neuroscience* 20.3 (2017), pp. 353–364.

[7] Annette Borchers and Tomas Pieler. “Programming pluripotent precursor cells derived from Xenopus embryos to generate specific tissues and organs”. In: *Genes* 1.3 (2010), pp. 413–426.

[8] Ulrik Brandes et al. “On modularity clustering”. In: *IEEE transactions on knowledge and data engineering* 20.2 (2007), pp. 172–188.

[9] Sourav Chatterjee and Persi Diaconis. “Estimating and understanding exponential random graph models”. In: (2013).

[10] Fan Chung and Linyuan Lu. “The average distance in a random graph with given expected degrees”. In: *Internet Mathematics* 1.1 (2004), pp. 91–113.

[11] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. “Finding community structure in very large networks”. In: *Physical review E* 70.6 (2004), p. 066111.

[12] Anne Condon and Richard M Karp. “Algorithms for graph partitioning on the planted partition model”. In: *Random Structures & Algorithms* 18.2 (2001), pp. 116–140.

[13] Martin E. Dyer and Alan M. Frieze. “The solution of some random NP-hard problems in polynomial expected time”. In: *Journal of Algorithms* 10.4 (1989), pp. 451–489.

[14] Elena Aleksandrovna Erosheva. “Grade of membership and latent structure models with application to disability survey data”. PhD thesis. Carnegie Mellon University, 2002.

[15] Leonhard Euler. “Solutio problematis ad geometriam situs pertinentis”. In: *Commentarii academiae scientiarum Petropolitanae* (1741), pp. 128–140.

[16] Santo Fortunato and Marc Barthelemy. “Resolution limit in community detection”. In: *Proceedings of the national academy of sciences* 104.1 (2007), pp. 36–41.

[17] Michelle Girvan and Mark EJ Newman. “Community structure in social and biological networks”. In: *Proceedings of the national academy of sciences* 99.12 (2002), pp. 7821–7826.

[18] Iman Habibi, Effat S Emamian, and Ali Abdi. “Quantitative analysis of intracellular communication and signaling errors in signaling networks”. In: *BMC systems biology* 8 (2014), pp. 1–16.

[19] Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. “Stochastic blockmodels: First steps”. In: *Social networks* 5.2 (1983), pp. 109–137.

[20] Mark Jerrum and Gregory B Sorkin. “The Metropolis algorithm for graph bisection”. In: *Discrete Applied Mathematics* 82.1-3 (1998), pp. 155–175.

[21] Brian Karrer and Mark EJ Newman. “Stochastic blockmodels and community structure in networks”. In: *Physical review E* 83.1 (2011), p. 016107.

[22] Jon Kleinberg. “An impossibility theorem for clustering”. In: *Advances in neural information processing systems* 15 (2002).

[23] Jussi M Kumpula et al. “Limited resolution in complex network community detection with Potts model approach”. In: *The European Physical Journal B* 56 (2007), pp. 41–45.

[24] Ulrike Luxburg, Olivier Bousquet, and Mikhail Belkin. “Limits of spectral clustering”. In: *Advances in neural information processing systems* 17 (2004).

[25] Roland Molontay and Marcell Nagy. “Two decades of network science: as seen through the co-authorship network of network scientists”. In: *Proceedings of the 2019 IEEE/ACM international conference on advances in social networks analysis and mining*. 2019, pp. 578–583.

[26] Mark EJ Newman. “Modularity and community structure in networks”. In: *Proceedings of the national academy of sciences* 103.23 (2006), pp. 8577–8582.

[27] Andrew Ng, Michael Jordan, and Yair Weiss. “On spectral clustering: Analysis and an algorithm”. In: *Advances in neural information processing systems* 14 (2001).

[28] Sofia C Olhede and Patrick J Wolfe. “Network histograms and universality of blockmodel approximation”. In: *Proceedings of the National Academy of Sciences* 111.41 (2014), pp. 14722–14727.

[29] Erdős Paul and Rényi Alfréd. “On random graphs I”. In: *Publicationes Mathematicae (Debrecen)* 6 (1959), pp. 290–297.

[30] Tiago P Peixoto. “Descriptive vs. inferential community detection in networks: pitfalls, myths, and half-truths”. In: *arXiv preprint arXiv:2112.00183* (2021).

[31] Karl Rohe, Sourav Chatterjee, and Bin Yu. “Spectral clustering and the high-dimensional stochastic blockmodel”. In: (2011).

[32] Noah A Rosenberg et al. “Genetic structure of human populations”. In: *science* 298.5602 (2002), pp. 2381–2385.

[33] Abbas Ali Saberi. “Recent advances in percolation theory and its applications”. In: *Physics Reports* 578 (2015), pp. 1–32.

[34] Mahmoud Saleh et al. “Optimal microgrids placement in electric distribution systems using complex network framework”. In: *2017 IEEE 6th International Conference on Renewable Energy Research and Applications (ICRERA)*. IEEE. 2017, pp. 1036–1040.

[35] Jianbo Shi and Jitendra Malik. “Normalized cuts and image segmentation”. In: *IEEE Transactions on pattern analysis and machine intelligence* 22.8 (2000), pp. 888–905.

[36] Søren M Sindbæk. “Networks and nodal points: the emergence of towns in early Viking Age Scandinavia”. In: *antiquity* 81.311 (2007), pp. 119–132.

[37] Tom AB Snijders and Krzysztof Nowicki. “Estimation and prediction for stochastic blockmodels for graphs with latent block structure”. In: *Journal of classification* 14.1 (1997), pp. 75–100.

[38] Danielle M Varda et al. “Social network methodology in the study of disasters: Issues and insights prompted by post-Katrina research”. In: *Population research and policy review* 28 (2009), pp. 11–29.

[39] Deepak Verma and Marina Meila. “A comparison of spectral clustering algorithms”. In: *University of Washington Tech Rep UWCSE030501* 1 (2003), pp. 1–18.

[40] Ulrike Von Luxburg. “A tutorial on spectral clustering”. In: *Statistics and computing* 17 (2007), pp. 395–416.

[41] Ulrike Von Luxburg, Mikhail Belkin, and Olivier Bousquet. “Consistency of spectral clustering”. In: *The Annals of Statistics* (2008), pp. 555–586.

[42] Duncan J Watts and Steven H Strogatz. “Collective dynamics of ‘small-world’networks”. In: *nature* 393.6684 (1998), pp. 440–442.

[43] Scott White and Padhraic Smyth. “A spectral clustering approach to finding communities in graphs”. In: *Proceedings of the 2005 SIAM international conference on data mining*. SIAM. 2005, pp. 274–285.

[44] Patrick J Wolfe and Sofia C Olhede. “Nonparametric graphon estimation”. In: *arXiv preprint arXiv:1309.5936* (2013).

[45] Wayne W Zachary. “An information flow model for conflict and fission in small groups”. In: *Journal of anthropological research* 33.4 (1977), pp. 452–473.