Community Detection Problem

Eric Sheng

University of Warwick

Supervised by Dr. Charilaos Efthymiou

Year of Study: 3rd Year

Major: Discrete Mathematics

13 June 2025

WARWICK
THE UNIVERSITY OF WARWICK

Abstract

The community detection problem has gained significant attention across various scientific fields due to its profound impact on numerous real-world applications. This project studies the community detection problem under the stochastic block model, with a particular focus on the hard regime of the Kesten-Stigum bound conjecture. Given the inherent hardness of this regime, we introduce a special setting called the partial seed set, which consists of a set of vertices whose community membership is known beforehand. We propose a greedy recovery algorithm that solves the reconstruction task in expectation in the hard regime under the partial seed set setting. The effectiveness of the algorithm is evaluated through extensive experiments and rigorous mathematical analysis. The proposed method also holds considerable practical relevance, especially in real-world scenarios where certain vertex memberships are known a priori, such as political leaders in a political ideological network.

Our findings contribute to the development of efficient and accurate community detection methods that leverage available information to uncover the underlying community structure in complex networks, especially when the community structure is weak.

Keywords: *community detection, graph clustering, graph partition, algorithms, graph theory, network science*

Acknowledgements

I would like to thank my supervisor, Dr. Charilaos Efthymiou, without whom this project would not have been possible. I had never even heard of this topic at the beginning; it was Charis who guided me through this journey. Our weekly meetings, where he distilled relevant knowledge, answered my queries, and pointed out some of my more naive ideas, significantly reduced my learning curve and spared me many potential missteps. Furthermore, his guidance sparked my interest in statistical physics. Beyond academics, Charis also shared his personal experiences to help me navigate various life challenges throughout this academic year, which I found immensely helpful.

Contents

List of Figures

List of Algorithms

List of Symbols

Planted Assignment	σ
Set of Communities	Ω
Communities Set under σ	$\Omega_i = \{v \in V : \sigma_v = i\}$
Computed Assignment	σ'
Communities Set under σ'	$\Omega'_i = \{v \in V : \sigma'_v = i\}$
Stochastic Block Model	SBM
Symmetric SBM	SSBM
With High Probability	w.h.p
the Kesten-Stigum (KS) threshold	SNR$= 1$
Random assignment (Random guess)	σ_{random}
Greedy Assignment	σ_{greedy}

CHAPTER 1

Introduction

Community detection, or graph clustering, can be informally described as the process of partitioning the vertices of a given graph into non-overlapping groups that are more densely connected internally. Since most network exhibits a community structure [For10], this problem has a significant impact on numerous real-world applications. For example, in social network, to divide individuals into groups according to the frequency of their interactions [MW03], or in protein-protein interaction networks, to identify clusters of proteins that perform identical functions within a cell [CY06]. Therefore, this problem has sparked huge interest across different scientific communities, such as network science, social science, bioinformatics, machine learning and statistical physics, since the 1980s. Despite the development of a remarkable variety of models and algorithms, the challenge remains unsolved, as it is a $\mathcal{NP}$-hard problem in general [Sch07].

To address the community detection problem, a rigorous mathematical framework is essential. Although various effective models exist, such as *Markov* *random field* and *factor graph model*. However, this report adopts a probabilistic generative model, called *stochastic block model (SBM)*, because it is arguably the most representative one due to its canonicality. Moreover, the SBM exhibits a phase transition phenomenon, known as the Kesten-Stigum Bound Conjecture, where community detection is impossible when the Signal-to-Noise Ratio (SNR) is less than one (hard regime), while efficient algorithms exist for SNR greater than one (easy regime) [Abb23] [Jin+21].

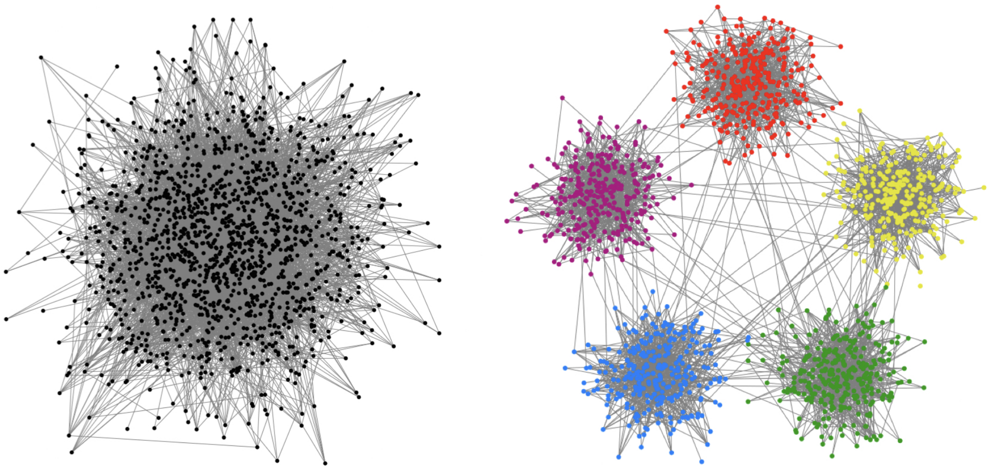


Figure 1.0.1: The left graph is drawn from an SBM with 1000 vertices, 5 built-in communities, vertices inside the same community are connected with probability 1/50, and vertices across different communities connected with probability 1/1000. This figure is cited from [Abb18]. The goal of community detection is to recover the community structure as shown in the right graph from the left graph up to some level of accuracy.

This report will stick closely with the Kesten-Stigum Bound Conjecture, and its structure will be organised based on the SNR value. We will first introduce an existing solution known as the Spectral Algorithm for the easy regime (SNR > 1). We will then delve into the more challenging hard regime (SNR ≤ 1).

The goal of this project is to solve the community detection problem in the hard regime with the aid of additional information, called partial seed set. We propose a greedy recovery algorithm under this setting and discuss some improvements to this algorithm tailored to specific objective conditions, finally extending it to the case of multiple communities.

Before proceeding, we will first formally define the problem and introduce some important concepts that will be used throughout this report.

1.1 Formal Definition

1.1.1 Key Concepts

How can we measure the concept of community? While traditional definitions relied on counting the number of internal and external edges, the modern perspective emphasises the probability of vertices sharing edges within a subgraph [FH16]. The existence of communities suggests that vertices have stronger interactions with members of their own community compared to other vertices belonging to different communities. As a result, vertices within the same community are more likely to form connections with each other than with vertices outside their community. Therefore, this gives rise to a natural definition of community.

Definition 1.1.1 (Community). For a graph $G = (V, E)$, we say $\mathcal{C} \subseteq V$ is a community if for $\forall v \in \mathcal{C}$, we have $\mathbb{P}\left(\{v, u\} \in E\right) > \mathbb{P}\left(\{v, u'\} \in E\right)$ for $\forall u \in \mathcal{C}$ and $u' \in V \setminus \mathcal{C}$.

Remark. *In this report, the terms 'community' and 'group' are used interchangeably.*

Definition 1.1.2 (Stochastic Block Model). Consider a graph $G = (V, E)$ generated by the Stochastic Block Model $SBM(n, k, \mathcal{P})$, where n is the number of vertices, k denotes the number of communities, and $\mathcal{P}$ is a $k \times k$ probability matrix. Under this model, the graph G possesses a built-in community structure, let Ω denote the set of communities, then each vertex v is assigned to a community $\sigma_v \in \Omega$, where σ is called the planted assignment. For any pair of vertices $u, v \in V$, the probability that there is a connection between u and v is specified by $\mathcal{P}_{\sigma_u,\sigma_v}$, independent of other pairs of vertices. We also define the community sets as $\Omega_i := \{v \in [n] : \sigma_v = i\}$ for $i \in \Omega$.

Remark. *In simple terms, each pair of vertices is connected with probability that only depends on their communities.*

This report focuses on the case where communities have equal size, i.e., $|\Omega_i| = |\Omega_j|$ for $\forall i, j \in \Omega$. Therefore, we will often assume this without specifying it explicitly.

We also introduce the symmetric SBM:

Definition 1.1.3 (Symmetric SBM). A symmetric SBM SSBM is a SBM where the σ_v are chosen independently and uniformly, and for $\forall u, v \in V$

$$\mathbb{P}\left(\{u, v\} \in E\right) = \begin{cases} p_{\text{in}} & \text{if } \sigma_u = \sigma_v \\ p_{\text{out}} & \text{if } \sigma_u \neq \sigma_v \end{cases} \tag{1.1}$$

i.e., the probability matrix $\mathcal{P}$ takes the value p_{in} on the diagonal and p_{out} off the diagonal. Therefore we can use $SSBM(n, k, p_{in}, p_{out})$ to represent $SBM(n, k, \mathcal{P})$

Remark. *if $p_{in} = p_{out}$, then it reduces to the Erdős-Rényi random graph.*

In this report, we focus on the symmetric case with $p_{in} > p_{out}$, which is called *homophily* or *assortativity*. On the other hand, the case where $p_{in} < p_{out}$ is called *disassortative.*

Let c_{in} be the expected degree connected inside, and c_{out} be the expected degree connected outside. Since each group has expected size $\frac{n}{k}$, then each vertex has $p_{in} \cdot \frac{n}{k} = \frac{c_{in}}{k}$ neighbours inside of its group in expectation, similarly $\frac{c_{out}}{k}$ neighbours in each of the other groups. Hence, we have

$$c = \frac{c_{in} + (k-1)c_{out}}{k} \tag{1.2}$$

1.1.2 Detection and Reconstruction

Now, given a graph G, how can we compute its planted assignment σ? or can we even confirm whether it possesses a community structure or not. This brings us two primary tasks in this field, *detection* and *reconstruction* [Abb18][Moo17].

Definition 1.1.4 (Detection). Consider a graph G that is randomly drawn with equal probability from either the Erdős-Rényi random graph model or a SBM model with the same expected degree. The detection task aims to determine, with an asymptotic probability of $\frac{1}{2} + \varepsilon$ for some $\varepsilon > 0$, which of the two ensembles the graph G originated from.

For the reconstruction task, also referred to as weak recovery, the goal is to recover the planted assignment σ up to some level of accuracy. There are several reasonable definitions, we now introduce two of them, which turn out to be equivalent for the purpose of this project.

Before moving forward, it is essential to first introduce a measurement that will be used to evaluate the performance of the reconstruction task.

Definition 1.1.5 (Agreement). The agreement between two community vectors $\sigma, \sigma' \in \Omega^k$ is determined by maximising the number of common components between σ and any relabelling of σ'. Mathematically,

$$A(\sigma, \sigma') = \max_{\pi \in S_k} \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}(\sigma_i = \pi(\sigma'_i)) \tag{1.3}$$

where S_k represents the set of all permutations on Ω.

Let Ω'_i denote the communities set computed by σ', i.e., $\Omega'_i = \{v \in [n] : \sigma'_v = i\}$. In the paper of [Moo17] [Dec+11], reconstruction is defined as follows:

Definition 1.1.6 (Reconstruction). Let (G, σ) be drawn from a $SSBM(n, k, p_{in}, p_{out})$, the reconstruction is to compute a mapping $\sigma' : [n] \to \Omega$ with $|\Omega'_i| = |\Omega'_j|$ for any two communities $i, j \in \Omega$ (communities of equal size) such that $A(\sigma, \sigma') > 1/k$ with high probability, in this case, we say reconstruction is solved under this $SSBM$.

Remark. *if we assign each vertex v to one of the communities independently and uniformly at random (we call this random guess), then by law of large numbers, we have $A(\sigma, \sigma') \to 1/k$ [Moo17]. So this definition essentially asks whether we can compute a σ' that performs strictly better than a random guess.*

Remark. *If σ' simply places all vertices into a single community, then $A(\sigma, \sigma') = 1/k$, which does not solve the reconstruction.*

While the definition of reconstruction from [Abb18] is as below:

Definition 1.1.7 (Reconstruction). Reconstruction is solved in $SSBM(n, k, p_{in}, p_{out})$ if for (G, σ) drawn from this $SSBM$, there exists $\varepsilon > 0$, $i, j \in \Omega$ and an algorithm that takes G as an input and outputs a partition of $[n]$ into two sets (S, S^c) such that

$$\mathbb{P}\left\{\left|\frac{|S\cap\Omega_i|}{|\Omega_i|}-\frac{|S\cap\Omega_j|}{|\Omega_j|}\right|\geq\varepsilon\right\}=1-o(1) \tag{1.4}$$

where we recall that $\Omega_i = \{v \in [n] : \sigma_v = i\}$.

Remark. *Put simply, an algorithm is considered to solve the reconstruction problem if it can partition the vertices of the graph into two sets in such a way that vertices belonging to different communities have distinct probabilities of being assigned to one of the sets.*

Remark. *In some papers, the terms reconstruction, weak recovery, and detection are used interchangeably. However, in this report, we treat them as separate concepts to maintain clarity and avoid confusion.*

We now demonstrate that, for the purposes of this report, the two definitions of reconstruction presented above are equivalent.

Claim 1.1.1. *For symmetric SBM with equal size of communities, Definition 1.1.6 and 1.1.7 are equivalent.*

Proof. Let (G,σ) be drawn from $SSBM(n,k,p_{in},p_{out})$, denote $\Omega'_i = \{v \in [n] : \sigma'_v = i\}$. So $|\Omega_i| = |\Omega'_i| = n/k$ for all $i \in \Omega$.
$'' \Rightarrow ''$: There exists an assignment σ' such that $A(\sigma,\sigma') > 1/k$ $w.h.p$ $\Rightarrow$ $\mathbb{P}\{A(\sigma,\sigma') > 1/k\} = 1 - o(1) \Rightarrow \exists\varepsilon > 0$ such that $\mathbb{P}\{A(\sigma,\sigma') \geq 1/k+\varepsilon\} = 1 - o(1)$, so there exits a permutation π such that $\sum_{v\in[n]} \mathbb{1}(\sigma_v = \pi(\sigma'_v)) \geq n/k + n\varepsilon$ with probability $\geq 1 - o(1)$. Let $\phi_v = \pi(\sigma'_v)$ for all v, then there must exist a community j (w.h.p) such that $\sum_{v\in\Omega'_j} \mathbb{1}(\sigma_v,\phi_v) \geq n/k^2 + n\varepsilon/k$, as

otherwise, we would have

$$\begin{aligned}\sum_{v\in[n]} \mathbb{1}(\sigma_v = \phi_v) &= \sum_{\{v\in\Omega_i \,:\, i\in\Omega\}} \mathbb{1}(\sigma_v = \phi_v) \\ &< k\cdot(n/k^2 + n\varepsilon/k) \\ &= n/k + n\varepsilon, \quad \text{contradiction}\end{aligned}$$

Now, let $S = \Omega_j'$, we have $|\Omega_j \cap S| \geq n/k^2 + n\varepsilon/k \Rightarrow \exists$ a community i such that $|\Omega_i \cap S| < n/k^2$, as otherwise,

$$\begin{aligned}|S| &= |S \setminus (S\cap\Omega_j)| + |S\cap\Omega_j| \\ &= \sum_{\{i\in\Omega: i\neq j\}} |S\cap\Omega_i| + |S\cap\Omega_j| \\ &\geq n/k^2\cdot(k-1) + n/k^2 + n\varepsilon/k \\ &> n/k, \quad \text{contradiction}\end{aligned}$$

Since all these happen with probability at least $1-o(1)$, let $0 < \delta \leq \left|\frac{|S\cap\Omega_i|}{|\Omega_i|} - \frac{|S\cap\Omega_j|}{|\Omega_j|}\right|$, then $\mathbb{P}\left\{\left|\frac{|S\cap\Omega_i|}{|\Omega_i|} - \frac{|S\cap\Omega_j|}{|\Omega_j|}\right| \geq \delta\right\} = 1-o(1)$. Therefore, the equation 1.4 is satisfied.

$'' \Leftarrow ''$: We induct on the number of communities $k \geq 2$.
For $k = 2, V = \Omega_1 \cup \Omega_2$, and $|\Omega_1 \cap S| - |\Omega_2 \cap S| \geq n\varepsilon/k$ with probability $1-o(1)$, let $S = \Omega_1'$, so we have

$$\begin{aligned}|\Omega_1\cap\Omega_1'| + |\Omega_2\cap\Omega_2'| &= |\Omega_1\cap\Omega_1'| + |\Omega_2\cap(V\setminus\Omega_1')| \\ &= |\Omega_1\cap\Omega_1'| + |\Omega_2| - |\Omega_2\cap\Omega_1'| \\ &\geq n\varepsilon/k + n/k \quad \text{with probability } 1-o(1)\end{aligned}$$

Therefore, we obtain an assignment σ' with $\Omega'_i = \{v : \sigma'_v = i\}$ such that

$$\begin{aligned}\sum_V \mathbb{1}(\sigma_v = \sigma'_v) &= \sum_{v\in\Omega_1} \mathbb{1}(\sigma_v = \sigma'_v) + \sum_{v\in\Omega_2} \mathbb{1}(\sigma_v = \sigma'_v)\\ &= |\Omega_1 \cap \Omega'_1| + |\Omega_2| - |\Omega_2 \cap \Omega'_1|\\ &\geq n\varepsilon/k + n/k \quad \text{with probability } 1 - o(1)\end{aligned}$$

Hence, $\mathbb{P}\{A(\sigma,\sigma') \geq 1/k + \varepsilon/k\} = 1 - o(1) \Rightarrow A(\sigma,\sigma') > 1/k$ *w.h.p*

Now, consider $V = \cup_{\ell=1}^k \Omega_\ell$, $\exists i, j \in \Omega$ such that $|\Omega_i \cap S| - |\Omega_j \cap S| \geq n\varepsilon_1/k$ with probability $1 - o(1)$. Let $S = \Omega'_i \Rightarrow |\Omega_i \cap \Omega'_i| - |\Omega_j \cap \Omega'_i| \geq n\varepsilon_1/k \Rightarrow |\Omega_i \cap \Omega'_i| \geq n\varepsilon_1/k$. Let $\varphi : S \to i$ denote this map. Now, consider $V' = V \setminus \Omega_i$, by inductive hypothesis, there exists an assignment ψ such that $\frac{1}{n-n/k}\sum_{V'} \mathbb{1}(\sigma_v = \psi_v) > \frac{1}{k-1} \Rightarrow \exists \varepsilon_2 > 0$, such that $\frac{1}{n-n/k}\sum_{V'} \mathbb{1}(\sigma_v = \psi_v) \geq \frac{1}{k-1} + \varepsilon_2 \Rightarrow \sum_{V'} \mathbb{1}(\sigma_v = \psi_v) \geq \frac{n-n/k}{k-1} + \varepsilon_2(n - n/k) = \frac{n}{k} + \varepsilon_2(n - n/k)$. Now define assignment $\sigma' : V \to \Omega$, given by $\sigma'_v = \begin{cases} \varphi_v & \text{if } v \in \Omega'_i \\ \psi_v & \text{if } v \in V' \end{cases}$, then

$$\begin{aligned}\sum_V \mathbb{1}(\sigma_v = \sigma'_v) &= \sum_{V'} \mathbb{1}(\sigma_v = \psi_v) + \sum_{\Omega_i} \mathbb{1}(\sigma_v = \varphi_v)\\ &= \sum_{V'} \mathbb{1}(\sigma_v = \psi_v) + |\Omega_i \cap \Omega'_i|\\ &= \sum_{V'} \mathbb{1}(\sigma_v = \psi_v) + \sum_{\Omega'_i} \mathbb{1}(\sigma_v = \varphi_v)\\ &\geq \frac{n}{k} + \varepsilon_2(n - n/k) + n\varepsilon_1/k \quad w.h.p\end{aligned}$$

Let $\delta = \frac{\varepsilon_2(n-n/k)+n\varepsilon_1/k}{n}$, then we have $\sum_V \mathbb{1}(\sigma_v = \sigma'_v) \geq n/k + n\delta \quad w.h.p \Rightarrow A(\sigma,\sigma') > 1/k \quad w.h.p.$ □

1.1.3 Kesten-Stigum Bound Conjecture

Pierre Curie observed that iron has a phase transition at a critical temperature, where its magnetisation abruptly drops to zero due to an equal proportion of atoms pointing in opposite directions. This occurs when the system is excessively hot and noisy, or equivalently, if the interactions between neighbouring atoms are too weak, the correlations between atoms diminish exponentially with distance, and the proportion of atoms pointing in each direction approaches to $1/2$ as $n \to \infty$ [Moo17]. Similarly, in community detection under the SBM, researchers conjectured that a comparable phase transition phenomenon takes place. Specifically, if the community structure is too weak, the fraction of correctly labelled vertices will converge to 1/k, no better than random guessing, as the number of vertices increases.

We now turn our attention to this pivotal conjecture in this field, which is also the central focus of our report. This conjecture was first presented in [Dec+11], and formally stated in [Abb18] as follows:

Conjecture 1 (Kesten-Stigum Bound). Let (G, σ) be drawn from a symmetric SBM with n vertices, k communities, probability p_{in} of connecting vertices within the same community, and probability p_{out} of connecting vertices across different communities. Define SNR $= \frac{(c_{in}-c_{out})^2}{k(c_{in}+(k-1)c_{out})}$. Then,

1. For any $k \geq 2$, if SNR > 1 (above the Kesten-Stigum (KS) threshold), reconstruction is possibly solvable in polynomial time.

2. If $k \geq 5$[1] and SNR < 1, reconstruction is possibly solvable from an information-theoretical perspective (not necessarily in polynomial time).

[1] $k \geq 4$ if we don't require $p_{in} > p_{out}$.

Let's review some progress made regarding the Kesten-Stigum Bound conjecture. For the first statement in the conjecture, the positive part was initially proven in [Mas14] and [MNS14]:

Theorem 1.1.1. *For $k = 2$ and SNR > 1, reconstruction can be solved efficiently.*

Remark. *This theorem was later proved in [BLM15] using a spectral algorithm based on non-backtracking matrix, which we shall introduce in section 2.2.*

For the negative part of the first statement, it was shown in [MNS15] that:

Theorem 1.1.2. *For $k = 2$ and SNR ≤ 1, reconstruction is not solvable.*

Remark. *This theorem was proved by a reduction to the broadcasting problem on Gaton-Watson tree with Poisson offspring.*

Finally, the case $k > 2$ of the fist statement and the Statement 2 of this conjecture was proved in [AS16a] and [AS16b]:

Theorem 1.1.3. *(part 1 is presented in [AS16a], part 2 is presented in [AS16b])*

- *For $k \geq 2$ and SNR > 1, reconstruction is efficiently solvable with approximate acyclic belief propagation algorithm.*
- *For $k \geq 4$, reconstruction is information-theoretically solvable for some SNR < 1 with the typicality sampling algorithm.*

Remark. *Therefore, we have two thresholds, one is the computational threshold, above which we have efficient algorithm that solves the reconstruction problem. the other one is the information-theoretic threshold, above it we can solve the reconstruction but not necessarily in polynomial time, below it we simply have no sufficient information to solve reconstruction. The two threshold coincides at SNR $= 1$ for the case of $k = 2$, but according to theorem 1.1.3, there is a gap between the computational threshold and the information-theoretic threshold as visualised in figure 1.3.1.*

1.2 Objectives

Inspired by the Kesten-Stigum Bound conjecture and our Claim 1.1.1, this project aims to design an algorithm that performs strictly better than the random guess for reconstruction in the hard regime, where SNR ≤ 1. However, according to Theorem 1.1.2, this is infeasible. Therefore, some additional information is needed in order to perform the reconstruction task in the hard regime. The objectives are categorised as follows:

Primary Objectives:

- Identify a reasonable setting for the additional information.
- Design an algorithm that strictly surpass the random guess in expectation in the hard regime (SNR ≤ 1) on $SSBM(n, 2, p_{in}, p_{out})$ with 2 communities of equal size, assisted by this extra information.
- Implement the algorithm and conduct a comprehensive empirical analysis to assess its effectiveness.

Following the convention in this field, after an in-depth exploration of the two-community case, we will consider extending our approach to scenarios with multiple communities ($k \geq 5$). Therefore,

Secondary Objectives:

- Conduct a rigorous mathematical analysis to evaluate the effectiveness of the proposed algorithm.
- Generalise our algorithm to the case of five communities and evaluate its effectiveness against random guessing on $SSBM(n, 5, p_{in}, p_{out})$ with five equally-sized communities in the hard regime.

1.3 Related Work

Community detection has been a topic of extensive research since the 1980s. Over the years, numerous models and algorithms have been developed, which can be broadly categorised into three main groups [SPK22]: Modurlarity-based, Statistical inference and Traditional algorithms. Modularity-based algorithms aim to maximise the modularity function by identifying a suitable community structure through various heuristic approaches [NG04] [New06]. Statistical inference algorithms include methods such as random walks and belief propagation random walk method and belief propagation algorithm[Moo17] [Abb18]. Traditional algorithms include hierarchical clustering [For10], Girvan-Newman algorithms [New04], spectral clustering [Lux07] [WLJ20], and graph-partitioning-based algorithms [Kar96].

After outlining some notable algorithms in this field, it's crucial to assess their effectiveness. Several benchmarks are devised for graphs with known community structure. Among these, SBM is arguably the most elegant and widely used model for the community detection problem [Abb18] [FH16]. It was invented in [HLL83], it's also known as the planted partition model in theoretical computer science [DF89]. Beyond its application in graph clustering, the SBM also provides a rich framework for investigating a range of fundamental problems in machine learning, computer science, and statistics due to its inherent phase transition properties [Abb18] [AS15].

The Kesten-Stigum Bound conjecture describes a phrase transition phenomenon (see figure 1.3.1), where a desired property becomes almost impossible to achieve below a certain threshold but suddenly becomes highly likely to occur once the threshold is exceeded. Prominent examples include the connectivity of Erdős-Rényi random graph and Shannon's coding theorem [Sha48]. Phrase

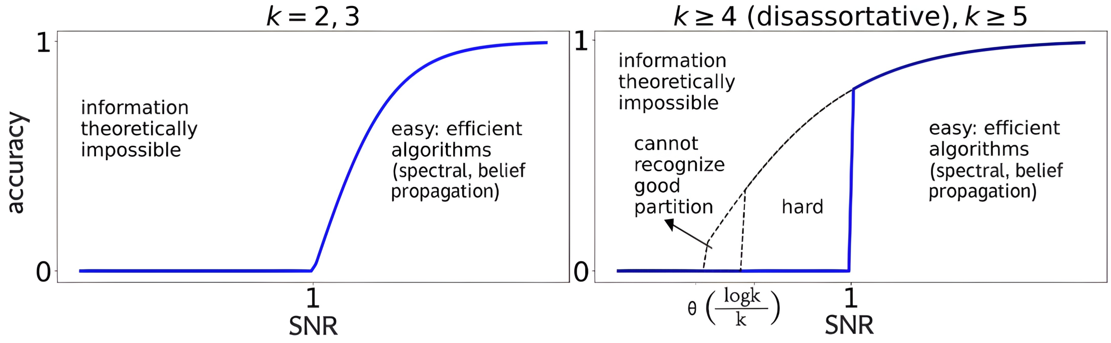


Figure 1.3.1: Phase transition for the Kesten-Stigum Bound conjecture. There is a remarkable surge at SNR = 1, where the reconstruction problem shifts from being impossible to achieve efficiently to being efficiently solvable. For $k = 2$ and 3, information-theoretic and computational threshold are both at $SNR = 1$, but for $k \geq 5$ on disassortative case, the information-theoretic threshold is $\theta(logk/k)$ while computation threshold is 1 [Moo17].

transition phenomenon has played a crucial role in the development of various algorithms, often serving as a guiding principle [Abb18].

Spectral algorithms are arguably the most widely used approaches in community detection, mainly due to its efficiency and mathematical elegance [Nd11]. Belief propagation and standard spectral algorithms, which include those based on adjacency and Laplacian matrices, are both effective for reconstruction when the graph is sufficiently dense and above the Kesten-Stigum (KS) threshold [NN12]. However, when the graph is sparse, only belief propagation can achieve the KS threshold [Dec+11], while standard spectral algorithms fail a significant distance from the KS threshold [Zha+12]. To address this issue, a spectral algorithm based on the non-backtracking matrix was introduced in [Krz+13], which can be viewed as a linearisation of the belief propagation algorithm[AS16c]. the non-backtracking matrix-based spectral algorithm has been proven to succeed down to the KS threshold in both sparse and dense graphs [BLM15], bridging the gap between spectral methods and statistical inference in terms of effectiveness for community detection. Furthermore, the non-backtracking matrix-based spectral algorithm is considered one of the most efficient and elegant approaches to the community detection problem [Krz+13] [Abb18].

This review doesn't encompass the full spectrum of the existing solutions to community detection. Due to the vast amount of literature on this subject, it is challenging to provide an exhaustive account of all the techniques. For a more thorough exploration, I recommend referring to the papers in [Abb18] [For10] [FH16] [Dal21].

1.4 Overview

The rest of this report is structured as follows:

- Chapter 2: This chapter focuses on the easy regime, where SNR > 1. We first discuss the standard spectral algorithm and its limitations. Then, we present the spectral algorithm based on the non-backtracking matrix and highlight its advantages.

- Chapter 3: This chapter focuses on the hard regime, where SNR ≤ 1. We introduce the partial seed set setting and propose a greedy recovery algorithm designed for this setting and discuss some improvements concerning the performance and efficiency of this algorithm. We then provide a mathematical analysis of this algorithm and present experimental results for the two-community case.

- Chapter 4: This chapter investigates the performance of combining the spectral algorithm and our greedy recovery algorithm for the case of two communities and SNR > 1. We explore the potential benefits and limitations of this hybrid approach

- Chapter 5: This chapter generalises our greedy recovery algorithm to the case of five communities.
- Chapter 6: This chapter summaries the work done in this project, including the main contributions, limitations, and discuss potential future work.
- Chapter 7: This chapter discusses project management and methodology.

1.5 Preliminaries

Definition 1.5.1 (Adjacency matrix)**.** Given $G = (V, E)$, the adjacency matrix $\mathbf{A}$ for G is a $n \times n$ matrix such that

$$\mathbf{A}_{i,j} = \begin{cases} 1 & \text{if } \{i, j\} \in E \\ 0 & \text{otherwise} \end{cases}$$

Definition 1.5.2 (Laplacian matrix)**.** Given a simple $G = (V, E)$, the Laplacian matrix $\mathbf{L}$ for G is a $n \times n$ matrix such that

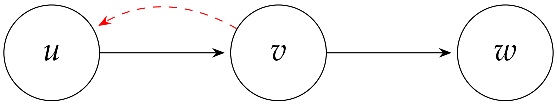
$$\mathbf{L}_{i,j} = \begin{cases} dge(v_i) & \text{if } i = j \\ -1 & \text{if } i \neq j \text{ and } \{i, j\} \in E \\ 0 & \text{otherwise} \end{cases}$$

Equivalently, $\mathbf{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{A}$ is the adjacency matrix and $\mathbf{D}$ is the degree matrix (a diagonal matrix with $\mathbf{D}_{i,i} = deg(i)$).

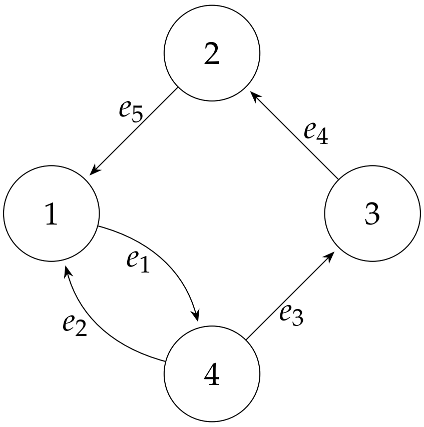
Definition 1.5.3 (Non-backtracking matrix)**.** Given a directed $G = (V, E)$, the non-backtracking matrix $\mathbf{B}$ for G is a $|E| \times |E|$ matrix such that

$$\mathbf{B}_{(u,v)(\ell,k)} = \begin{cases} 1 & \text{if } v = \ell \text{ and } u \neq k \\ 0 & \text{otherwise} \end{cases}$$

Remark. *This matrix corresponds to a non-backtracking walk, which is a walk that does not repeat a vertex within 2 steps. For example, $u \to v \to w$ is a valid non-backtracking walk, but $u \to v \to u$ is not.*



Example. *Given the following graph,*



its corresponding non-backtracking matrix **B** *is*

$$\mathbf{B} = \begin{array}{c} \\ \left(\begin{array}{ccccc} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{array}\right) \end{array} \hspace{-2em} \begin{array}{ccccc|c} e_1 & e_2 & e_3 & e_4 & e_5 & \\ \end{array}$$

$$\mathbf{B} = \begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & \\ 0 & 0 & 1 & 0 & 0 & e_1 \\ 0 & 0 & 0 & 0 & 0 & e_2 \\ 0 & 0 & 0 & 1 & 0 & e_3 \\ 0 & 0 & 0 & 0 & 1 & e_4 \\ 1 & 0 & 0 & 0 & 0 & e_5 \end{array}$$

CHAPTER 2

The Spectral Algorithms

In this chapter, we focus on the the case SNR > 1. Specifically, we revisit the standard spectral algorithms and non-backtracking matrix based spectral algorithm which is used to prove theorem 1.1.1.

Spectral methods typically refer to the approach of partitioning the nodes into groups by using the spectral properties of the graph matrices [Dal21]. The eigenvalue spectrum of various graph matrices, such as the adjacency matrix, the Laplacian, and the non-backtracking matrix, usually consists of a compact bulk of closely spaced eigenvalues, along with some outlying eigenvalues separated from the bulk. The eigenvectors corresponding to these outliers often provide insights into the large-scale structure of the network, including its community structure [FH16]. Therefore, it enables us to detect the communities in a graph by utilising the the eigenvectors corresponding to these outlying eigenvalues.

2.1 Standard Spectral Algorithm

The standard spectral algorithms usually refer to spectral algorithms based on adjacency or Laplacian matrix.

It has been demonstrated in [NN12] that if SNR > 1, for a given graph with built-in community structure, the eigenvector corresponding to the second largest eigenvalue of its adjacency matrix is correlated with the true community structure when the graph is sufficient dense. Therefore, the spectral algorithm based on the adjacency matrix can compute community labels by simply labelling vertices according to the sign of the second eigenvector for the case of two communities ($k = 2$). This approach can be generalised to the case of more than two communities ($k > 2$) by applying heuristic techniques such as the k-means algorithm.

However, this correlation diminishes in sparse graphs (i.e., c is constant while $n \to \infty$), as detailed in [Krz+13]. In sparse graphs, eigenvectors corresponding to eigenvalues outside the bulk may correlate with high-degree vertices rather than the community structure. Similarly, community-related eigenvalue could be merged into the bulk, swamping the community-correlated eigenvector with uninformative eigenvectors. As a consequence, identifying the community-correlated eigenvector becomes challenging or even impossible. The same issue arises when using the Laplacian matrix.

One potential remedy is to remove the high-degree vertices, but this approach discards a significant amount of information and can cause the graph to break apart into disconnected components [Abb18]. Therefore, an alternative method is required to mitigate the influence of high-degree vertices while preserving the community structure information.

2.2 Spectral Algorithm based on Non-backtracking Matrix

The non-backtracking matrix, as defined in definition 1.5.3, is introduced to address this issue. Due to its non-backtracking nature, i.e., a vertex cannot be revisited within two steps, the non-backtracking matrix is less sensitive to high-degree vertices compared to the adjacency or Laplacian matrices.
Let μ_i denote the *i-th* largest eigenvalue of the non-backtracking matrix, λ_i denote its corresponding eigenvector. it was shown in [Krz+13] that the non-backtracking matrix has the following spectrum properties:

Theorem 2.2.1. *Let G be drawn from* $SSBM(n, 2, p_{in}, p_{out})$*. If* $(c_{in} - c_{out})/2 > \sqrt{c}$*, then, with high probability, the two largest eigenvalues of* $\mathbf{B}$ *are real and satisfy*

- $\mu_1 \to c$
- $\mu_2 \to (c_{in} - c_{out})/2$
- $|\mu_{i_{>2}}| \leq \sqrt{c}$*, i.e. the radius of the bulk of* $\mathbf{B}$*'s spectrum is confined in* $\sqrt{c}$

Remark. *the constraint* $(c_{in} - c_{out})/2 > \sqrt{c}$ *is equivalent to* $SNR > 1$.

Krzakala et al. [Krz+13] point out that the eigenvector λ_2 is strongly correlated with the true community structure. Moreover, according to Theorem 2.2.1, μ_2, the second largest eigenvalue of the non-backtracking matrix $\mathbf{B}$, is well-separated from the bulk of $\mathbf{B}$'s spectrum when SNR > 1, regardless of whether the graph is sparse or dense. Figure 2.2.1 illustrates an example of the spectrum of $\mathbf{B}$, showcasing this separation.

Therefore, the spectral algorithm based on non-backtracking matrix validates Theorem 1.1.1. The algorithmic procedure of this spectral algorithm is summarized in Algorithm 1.

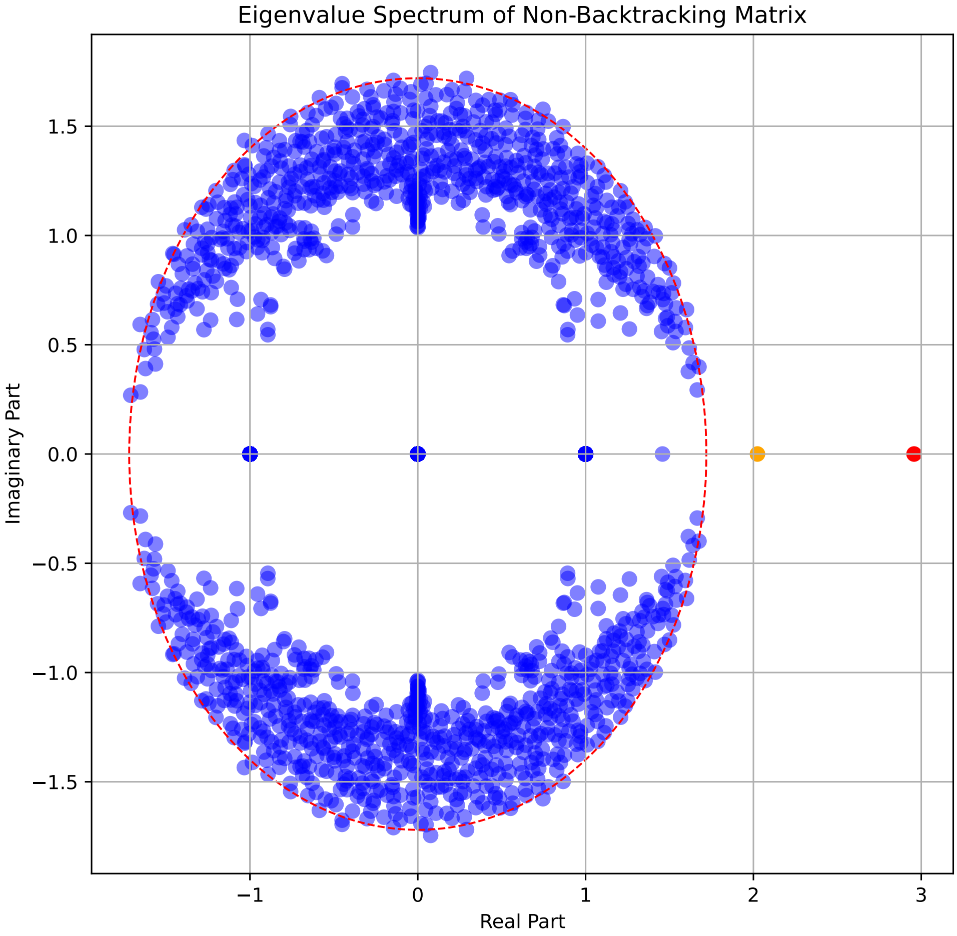


Figure 2.2.1: The spectrum of the non-backtracking matrix of a graph drawn from *SSBM* with parameters $n = 10^3$, $k = 2$, $c_{in} = 5$, $c_{out} = 1$. In this scenario, SNR $= 4/3$, and we can observe that μ_2 is well-separated from bulk of spectrum. Moreover, $\mu_1 \to 3 = (c_{in} + c_{out})/2 = c$, $\mu_2 \to 2 = (c_{in} - c_{out})/2$, and the radius of the bulk is $\sqrt{c} = \sqrt{3}$, These observations are in alignment with Theorem 2.2.1.

Algorithm 1: Spectral Algorithm based on Non-Backtracking Matrix

```
Input: Graph $G = (V, E)$ drawn from $SSBM(n, 2, p_{in}, p_{out})$
Output: Computed communities assignment $\sigma'$
begin
    Convert $G$ into a directed graph;
    $\mathbf{B} \leftarrow$ non-backtracking matrix of $G$;
    $\xi \leftarrow$ eigenvector corresponding to the second largest eigenvalue;
    for $v \in V$ do
        if $\sum_{\{(u,v) \in E:\ \text{head is } v\}} \xi_{(u,v)} > 0$ then
            $\sigma'_v \leftarrow$ Community A;
        end
        else
            $\sigma'_v \leftarrow$ Community B;
        end
    end
    return $\sigma'$
end
```

Figure 2.2.2 presents the experimental results of Algorithm 1. These results are consistent with those reported in [Krz+13].

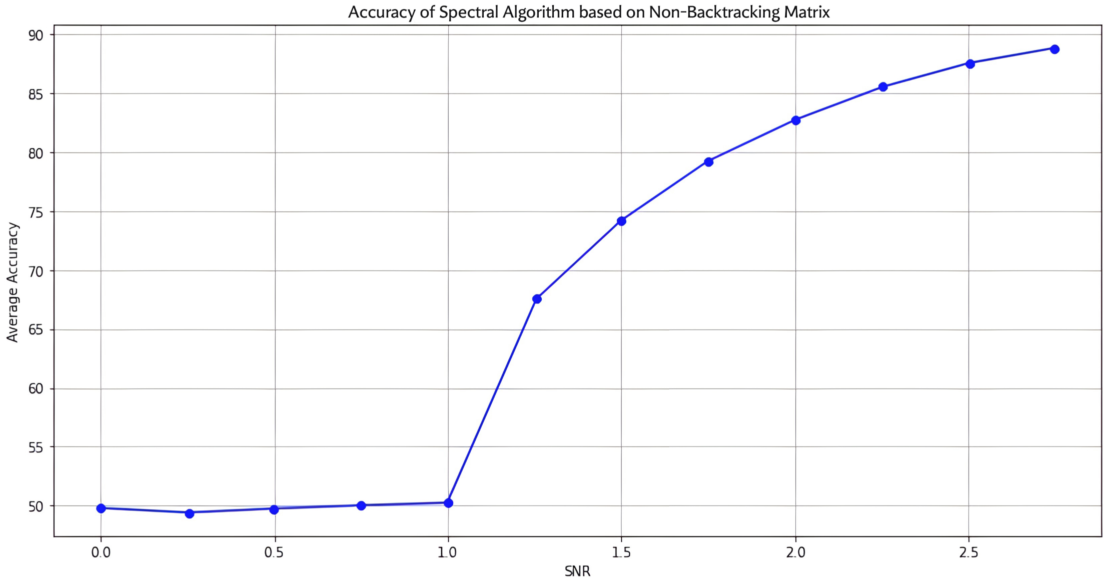


Figure 2.2.2: The accuracy of spectral algorithm based on non-backtracking matrix across different SNR values. This experiment is averaged over 10 instances with $n = 10^4$ and $c = 3$.

From Figure 2.2.2, we observe that when SNR > 1, the spectral algorithm based on the non-backtracking matrix noticeably outperforms random guessing, which has an expected accuracy of 50% across all SNR values.

CHAPTER 3

Partial Seed Set

Having discussed the easy regime (when SNR > 1), we now shift our focus to the hard regime (SNR ≤ 1), which is the primary focus of this project. We aim to achieve the reconstruction efficiently when SNR ≤ 1 for the case of $k = 2$ and then possibly generalise it to $k \geq 5$. However, by theorem 1.1.2, some additional assistance is required.

As discussed in the introductory chapter, one of the main applications of community detection is to recover social circles in a given social network. In many cases, we have certain prior knowledge about a small subset of group members. These could be political leaders and their core supporters in a network based on political ideologies, or influencers and their most engaged followers in a celebrity fandom network. Therefore, a realistic setting for the extra information is that we are given a subset of vertices for which we know their membership, our goal is then to recover the labels of the remaining vertices.

3.1 Problem Definition

Definition 3.1.1 (Partial Seed Set)**.** Let (G, σ) be drawn from $SBM(n, k, \mathcal{P})$, the partial seed set R is a subset of vertices such that σ_v is known for $\forall\ v \in R$, and $|R| = \alpha|V|$ for some $0 < \alpha < 1$.

As outlined in the objectives chapter, given (G, σ) drawn from $SSBM(n, k, \mathcal{P})$ with SNR ≤ 1 and a partial seed set R with $|R| = \alpha|V|$ for some $0 < \alpha < 1$, we aim to compute an assignment $\sigma' : [n] \to \Omega$ efficiently that satisfies $\mathbb{E}(A(\sigma, \sigma')) > \mathbb{E}(A(\sigma, \sigma_{random})) = \frac{1-\alpha}{k} + \alpha$, where σ_{random} is the random guess.

3.2 Greedy Recovery Algorithm

Since each vertex is more likely to connect with a vertex that belongs to the same community, if we assign the community label based on the most common community amongst a vertex's neighbours, then we have the hope to beat the random guess. Specially, for each $v \in V \setminus R$, we assign it the most common label observed among its neighbours whose community labels are already known. This approach allows us to incrementally expand the set R in a greedy fashion, continuing until no further assignments can be made. If any unlabelled vertices remain, we then assign them to one of the communities randomly. The steps of this algorithm are summarised in Algorithm 2.

Algorithm 2: Greedy Recovery Algorithm

```
Input: Graph $G = (V, E)$ drawn from $SSBM(n, k, \mathcal{P})$, and partial seed
       set $R$, which is a dictionary where keys are vertices and values
       are their corresponding community labels
Output: Computed communities assignment $\sigma'$
Function Greedy_Recovery($G, R$):
    if $|R| = |V|$ then
        return $R$;
    end
    for $v \in V \setminus R$ do
        if $N(v) \cap R \neq \emptyset$ then
            $R[v] \leftarrow$ the majority community label of $v$'s neighbours;
        end
    end
    repeat
        Greedy_Recovery($G, R$)                    ▷ greedily expand $R$
    until no new assignment can be made;
    if $\exists\ v \notin R$ then
        $R[v] \leftarrow \sigma_{random}(v)$;
    end
    $\sigma'(v) \leftarrow R[v]$ for $\forall\ v \in V$;
    return $\sigma'$;
```

Therefore, the computed assignment σ' consists of two subroutines: one is the greedy procedure, which we denote as σ_{greedy}, and the other is the random guess σ_{random}. Symbolically, let Γ denote the set of vertices handled by the greedy procedures, then

$$\sigma'(v) = \begin{cases} \sigma(v) & \text{if } v \in R \\ \sigma_{greedy}(v) & \text{if } v \in \Gamma \\ \sigma_{random}(v) & \text{if } v \in V \setminus (\Gamma \cup R) \end{cases} \tag{3.1}$$

Algorithm 2 has a running time of $\mathcal{O}(|V|^2|E|)$. This is because each recursive call to Greedy Recovery (G, R) costs $\mathcal{O}(|V||E|)$ and increase the size of R by at least 1, so there are at most n recursive calls. Therefore, Algorithm 2 is clearly a polynomial time algorithm.

For sparse graph, particularly when $|E| << |V|$, we can improve the running time by only checking the neighbours of vertices in R during each iteration, instead of the entire vertex set. This optimisation leads to Algorithm 3, which has a running time of $\mathcal{O}(|E|^2|V|)$.

Algorithm 3: Faster Greedy Recovery Algorithm

```
Input: Graph $G = (V, E)$ and $R$
Output: $\sigma'$
Function Faster_Greedy_Recovery($G, R$):
    if $|R| = |V|$ then
        return $R$
    end
    $Q \leftarrow$ empty queue
    for $v \in R$ do
        Enqueue $v$ into $Q$
    end
    while $Q$ is not empty do
        $v \leftarrow Q$.dequeue()
        for $u \in N(v) \setminus R$ do
            $R[u] \leftarrow$ the majority community label of $u$'s neighbours
            Enqueue $u$ into $Q$
        end
    end
    if $\exists\ v \notin R$ then
        $R[v] \leftarrow \sigma_{random}(v)$
    end
    $\sigma'(v) \leftarrow R[v]$ for $\forall\ v \in V$
    return $\sigma'$
```

3.3 Analysis

We adopt the same notations as those introduced in Equation 3.1.

Lemma 3.3.1. *Let (G, σ) be drawn from $SSBM(n, k, p_{in}, p_{out})$, with a partial seed set R such that $|R| = \alpha \cdot n$ for some $\alpha \in (0, 1)$. Let σ' be the assignment computed by our greedy recovery algorithm. If for each vertex $v \in \Gamma$, $\mathbb{P}(\sigma(v) = \sigma_{greedy}(v)) > \frac{1}{k}$, then $\mathbb{E}(A(\sigma, \sigma')) > \frac{1-\alpha}{k} + \alpha$.*

Proof. We know for vertex $v \in V \setminus (\Gamma \cup R)$, $\mathbb{P}(\sigma_v = \sigma'_v) = \mathbb{P}(\sigma(v) = \sigma_{random}(v)) = 1/k$, and for each $v \in R$, $\mathbb{P}(\sigma_v = \sigma'_v) = \mathbb{P}(\sigma(v) = \sigma(v)) = 1$. If $\mathbb{P}(\sigma_v = \sigma'_v) > 1/k$ for $v \in \Gamma$, then $\mathbb{E}(\sum_V \mathbb{1}(\sigma_v = \sigma'_v)) = \sum_V \mathbb{E}(\mathbb{1}(\sigma_v = \sigma'_v)) = \sum_V \mathbb{P}(\sigma_v = \sigma'_v) = \sum_\Gamma \mathbb{P}(\sigma(v) = \sigma_{greedy}(v)) + \sum_{V \setminus (\Gamma \cup R)} \mathbb{P}(\sigma(v) = \sigma_{random}(v)) + \sum_R \mathbb{P}(\sigma_v = \sigma_v)$ $> |\Gamma|\frac{1}{k} + |V \setminus (\Gamma \cup R)|\frac{1}{k} + |R| = \frac{1}{k}(n - \alpha n) + \alpha n$, so $\mathbb{E}(A(\sigma, \sigma')) > \frac{1-\alpha}{k} + \alpha$. □

For $v \in \Gamma$, let R_v denote the set of neighbours of v whose assignments have already been determined at the time our greedy algorithm evaluates vertex v. Let $\Re_v$ denote the set of vertices in R_v that belong to the most common community among v's neighbours when the greedy algorithm examines v. So, $\Re_v \subseteq N(v) \cap R_v$ and $\sigma'_v = \sigma'_u$ for any $u \in \Re_v$

Claim 3.3.2. *Let (G, σ) be drawn from $SSBM(n, k, p_{in}, p_{out})$, and σ' be the assignment computed by our greedy recovery algorithm. If $SNR > 0$ (i.e., $p_{in} > p_{out}$)[1] and $|\Re_v| > max(0,\ 1 - \frac{log(|\Re_v|)}{log(1-1/k)})$, then $\mathbb{P}(\sigma_v = \sigma'_v) > \frac{1}{k}$ for all $v \in \Gamma$.*

Proof. Suppose $(v_1, v_2, \ldots, v_{|\Gamma|})$ is the sequence of vertices examined by σ_{greedy}. we induct on the index of the vertices in this sequence. Let e_{out} denote the edge

[1] As we only care about the associativity case.

with endpoints in different communities.

For v_1,

$$\begin{aligned}
\mathbb{P}(\sigma_{v_1} \neq \sigma'_{v_1}) &= \mathbb{P}(\sigma_{v_1} \neq \sigma'_{v_1} | \sigma_u = \sigma'_u \text{ for any u in } \Re_{v_1}) \\
&\leq \mathbb{P}(\{v_1, u\} \text{ is } e_{out} \text{ for any } u \in \Re_{v_1} \mid \{v_1, u\} \in E \text{ for any } u \in \Re_{v_1}) \\
&= (\frac{p_{out}}{p_{in} + p_{out}})^{|\Re_{v_1}|} \\
&< (1 - \frac{1}{k})^{|\Re_{v_1}|} \qquad \text{as } (\frac{p_{out}}{p_{in} + p_{out}}) < \frac{1}{2} \leq (1 - \frac{1}{k}) \\
&< (1 - \frac{1}{k})
\end{aligned}$$

Consider some $v_{i>1}$ in this sequence of vertices. The situation now becomes more subtle, as this time $\Re_{v_i}$ could contain some falsely labelled vertices. So we need to use the number of mislabelled vertices in $\Re_{v_i}$ to partition the sample space of $\sigma_{v_i} \neq \sigma'_{v_i}$. Specially, let A := the event that $\sigma_{v_i} \neq \sigma'_{v_i}$, B_j := the even that j vertices in $\Re_{v_i}$ are labelled incorrectly.

$$\begin{aligned}
\mathbb{P}(\sigma_{v_i} \neq \sigma'_{v_i}) &= \sum_{j=0}^{|\Re_{v_i}|} \mathbb{P}(\sigma_{v_i} \neq \sigma'_{v_i} \cap j \text{ vertices in } \Re_{v_i} \text{ labelled incorrectly}) \\
&= \sum_{j=0}^{|\Re_{v_i}|} \mathbb{P}(A|B_j)\mathbb{P}(B_j)
\end{aligned}$$

Now,

$$\begin{aligned}
\mathbb{P}(A|B_j) &\leq \mathbb{P}((|\Re_{v_i}| - j) \text{ edges are } \ e_{out} \text{ among all edges between } \ v_i \text{ and } \Re_{v_i}) \\
&= (\frac{p_{out}}{p_{in} + p_{out}})^{|\Re_{v_i}| - j} \\
&< (1 - \frac{1}{k})^{|\Re_{v_i}| - j}
\end{aligned}$$

Also by inductive hypothesis, $\mathbb{P}(B_j) < (1 - \frac{1}{k})^j$, and since $|\Re_v| > 1 - \frac{log(|\Re_v|)}{log(1-1/k)}$, we have $|\Re_v|(1-\frac{1}{k})^{|\Re_v|} < 1-\frac{1}{k}$.

Hence,

$$\begin{aligned}
\mathbb{P}(\sigma_{v_1} \neq \sigma'_{v_1}) &= \sum_{j=0}^{|\Re_{v_i}|} \mathbb{P}(A|B_j)\mathbb{P}(B_j) \\
&< \sum_{j=0}^{|\Re_{v_i}|} (1-\frac{1}{k})^j \cdot (1-\frac{1}{k})^{|\Re_{v_i}|-j} \\
&= |\Re_{v_i}|(1-\frac{1}{k})^{|\Re_{v_i}|} \\
&< 1-\frac{1}{k}
\end{aligned}$$

Therefore, we have shown that $\mathbb{P}(\sigma_v \neq \sigma'_v) < 1-1/k$ for $v \in \Gamma \Rightarrow \mathbb{P}(\sigma_v = \sigma'_v) > 1/k$ for $v \in \Gamma$. □

Corollary 3.3.3. *Let (G,σ) be drawn from $SSBM(n,k,p_{in},p_{out})$, $|R| = \alpha \cdot n$ for some $\alpha \in (0,1)$, and σ' be the assignment computed by our greedy recovery algorithm. If $SNR > 0$ and $|\Re_v| > max(0,\ 1-\frac{log(|\Re_v|)}{log(1-1/k)})$, then $\mathbb{E}(A(\sigma,\sigma')) > \frac{1-\alpha}{k}+\alpha$, i.e., our greedy recovery algorithm is strictly better than random guess in expectation.*

Proof. By Lemma 3.3.1 and Claim 3.3.2. □

Remark. *for $k=2$, $|\Re_v| > 2$ would be sufficient.*

3.4 Experimental Results for k = 2 in Hard Regime

In this section, we delve into the empirical results of the greedy recovery algorithm for the case of 2 communities.

3.4.1 Spare Graph

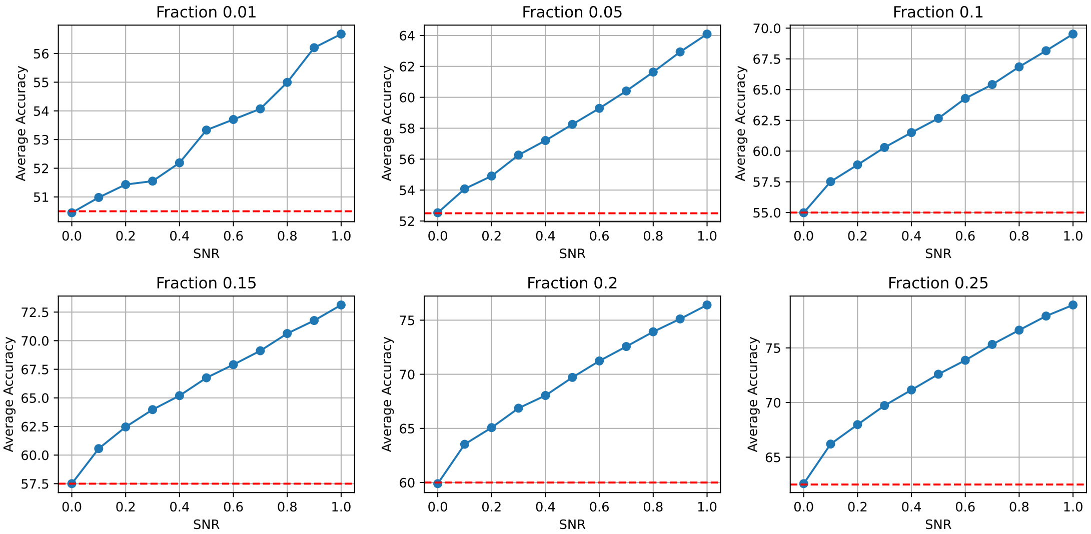


Figure 3.4.1: The blue line represents the accuracy of greedy recovery algorithm across different fraction values α (recall α, defined in definition 3.1.1, represents the fraction of the total number of vertices in V that are contained in partial seed set) with SNR ranging from 0 to 1. The red dashed line represents the accuracy of random guess($= \frac{1-\alpha}{2} + \alpha$) for a given fraction value α. For instance, the first plot is the accuracy of greedy recovery algorithm and the random guess when $|R| = 0.01 \cdot |V|$. This experiment results[2]are averaged over 10 instances with $n = 10^5$ and $c = 3$.

From figure 3.4.1, we observe that when SNR > 0, our greedy recovery algorithm performs noticeably better than the random guess for all the tested fraction values α.

[2] The parameters chosen for the sparse graph are in alignment with those used in [Krz+13].

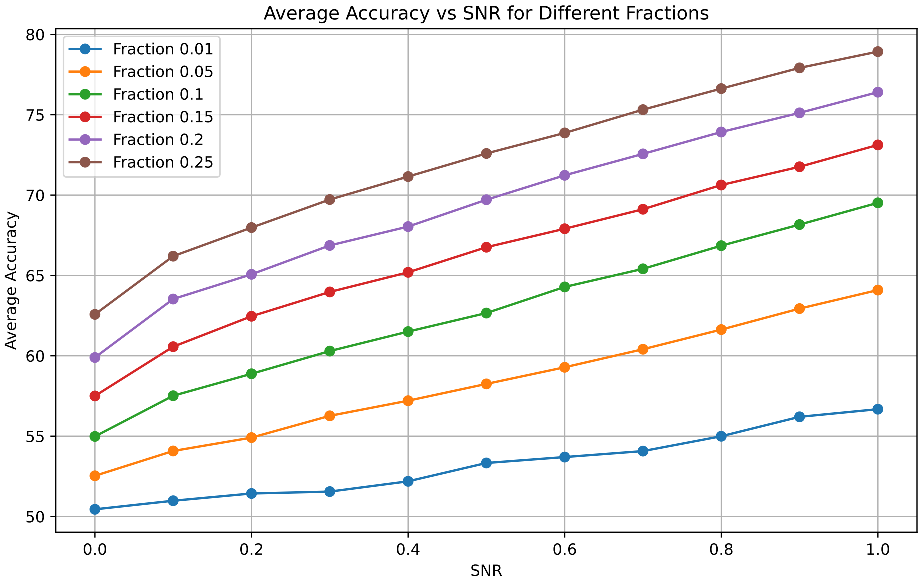


Figure 3.4.2: This plot combines the results from Figure 3.4.1 into a single compact visualisation.

.

From figure 3.4.2, we observe that the accuracy of the greedy recovery algorithm improves as the SNR increases, indicating that our algorithm performs better when the community structure becomes more apparent, which aligns with our expectations.

3.4.2 Dense Graph

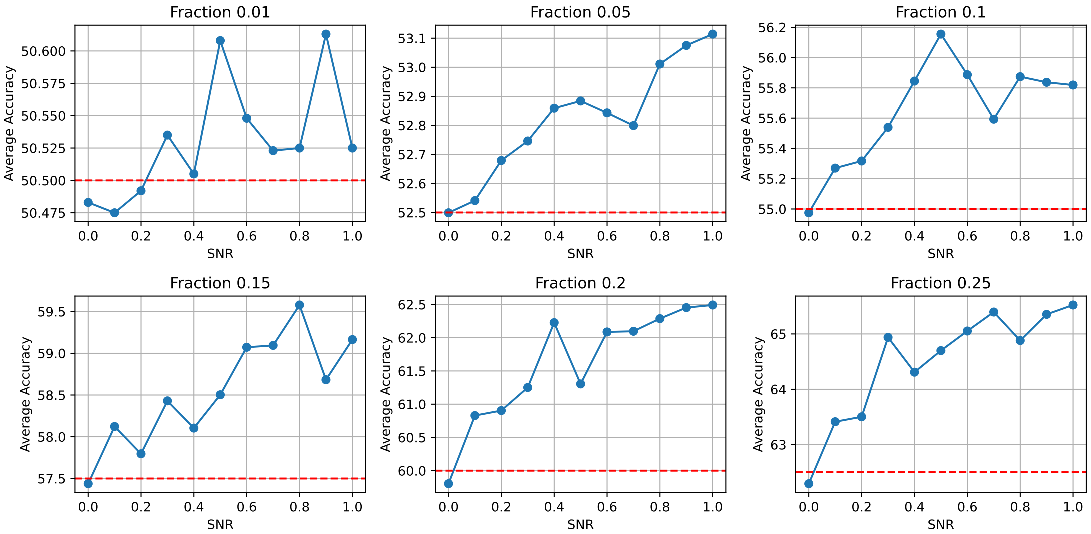


Figure 3.4.3: The blue line represents the accuracy of greedy recovery algorithm across different values of α(=fraction) for SNR ranging from 0 to 1. The red dashed line represents the accuracy of random guess(= $\frac{1-\alpha}{2} + \alpha$) for a given fraction value α. The experiment results[3]are averaged over 10 instances with $n = 10^4$ and $c = 200$.

This time, the pattern observed in figure 3.4.3 is more fluctuating, and the situation is more subtle. Our greedy recovery algorithm generally surpasses the random guess, but its superiority is not consistent, particularly when fraction value and SNR are small.

This is possibly due to the fact that the set R is small in the early stage, so the greedy algorithm is assigning labels based on only a small fraction of each vertex's neighbours, which may cause our greedy algorithm to underperform compared to random guessing. Moreover, the detrimental effects of these incorrect assignments could be magnified in later stages, leading to fluctuations in the accuracy of our greedy algorithm.

[3] The parameters chosen for the dense graph are in alignment with those used in [Dal21].

3.4.3 Heuristic Improvement

One potential remedy to the issue encountered in the dense graph is to delay the assignment of a vertex until we have sufficient information about it. Specially, for each vertex, we determine its label only when a certain fraction, say h, of its neighbours is known to us, i.e., $|R \cap N(v)| > h$. This criterion ensures that our greedy recovery algorithm expands the set R by prioritising vertices for which we have a higher confidence in their labelling. As R grows, we expect the algorithm to accumulate more information about the vertices that were skipped in earlier stages. Consequently, this algorithm should have a higher chance of computing a correct assignment when revisiting these vertices compared to the original approach. Therefore, it is reasonable to expect the modified algorithm to outperform the original greedy recovery algorithm. The steps for this heuristic greedy recovery algorithm are outlined in Algorithm 4.

Algorithm 4: Heuristic Greedy Recovery Algorithm

```
Input: $G, R$ and the heuristic factor $h$
Output: Computed communities assignment $\sigma'$
Function Heuristic_Greedy_Recovery($G, R, h$):
    if $|R| = |V|$ then
        return $R$;
    end
    for $v \in V \setminus R$ do
        if $N(v) \cap R > h$ then
            $R[v] \leftarrow$ the majority community label of $v$'s neighbours;
        end
    end
    repeat
        Heuristic_Greedy_Recovery($G, R, h$)              ▷ greedily grow $R$
    until no new assignment can be made in this heuristic way;
    $\sigma' \leftarrow$ Greedy_Recovery(G, R)
    return $\sigma'$;
```

The following figures present experimental results demonstrating the performance of the heuristic improvement for dense graphs.

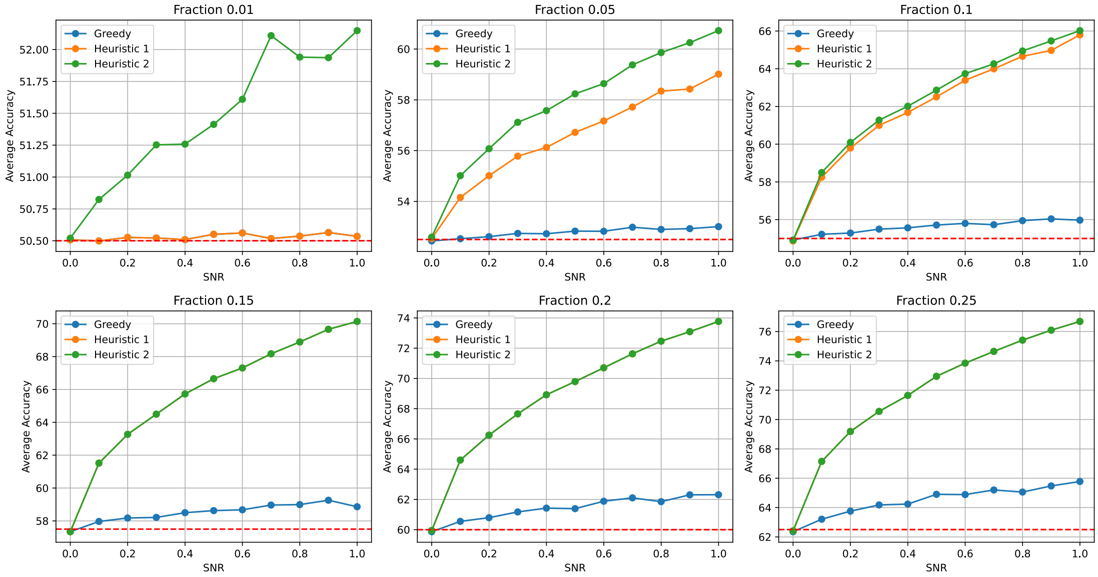


Figure 3.4.4: As usual, the red dashed line represents the accuracy of random guess, while blue line represents the accuracy of greedy recovery algorithm. The orange and green lines represent the accuracy of the greedy recovery algorithm with heuristic factors $h = \sqrt{c}$ and $h = \log c$, respectively. The experiment results are averaged over 10 instances with $n = 10^4$ and $c = 200$.

It is noteworthy that in Figure 3.4.4, when the fraction value is 0.01, the accuracy of Heuristic 1 is identical to the accuracy of the greedy recovery algorithm, causing the blue line to be covered by the orange line in plot 1. As the fraction value increases, the accuracy of Heuristic 1 converges to the accuracy of Heuristic 2. Therefore, in the plots for fraction value = 0.15, 0.2 and 0.25, the orange line is covered by the green line.

Moreover, we can observe that the greedy recovery algorithm with heuristic factor $h = \log c$ improves the accuracy of the original greedy algorithm by up to 12% at SNR = 1 in the plot of fraction value = 0.2. Furthermore, this enhanced algorithm consistently outperforms the random guess across all the selected fraction values when SNR > 0.

CHAPTER 4

Combined Algorithm

Now, we would like to take our discussion one step further. After exploring the spectral algorithm and the greedy recovery algorithm, a natural question[1] arises: can we achieve higher accuracy in the easy regime than the existing solution (i.e., the spectral algorithm) by combining the greedy recovery algorithm with the spectral algorithm? Specifically, if we utilise the assignment computed by the spectral algorithm as the partial seed set for our greedy recovery algorithm, can we improve upon the accuracy of the spectral algorithm?

4.1 Experimental Results for k = 2 in Easy Regime

We randomly select $fraction \times n$ vertices from the assignment computed by spectral algorithm to serve as the partial seed set for the greedy recovery algorithm. The result is shown in figure 4.1.1.

[1] Although this is not our objective, it is tempting enough to explore this direction.

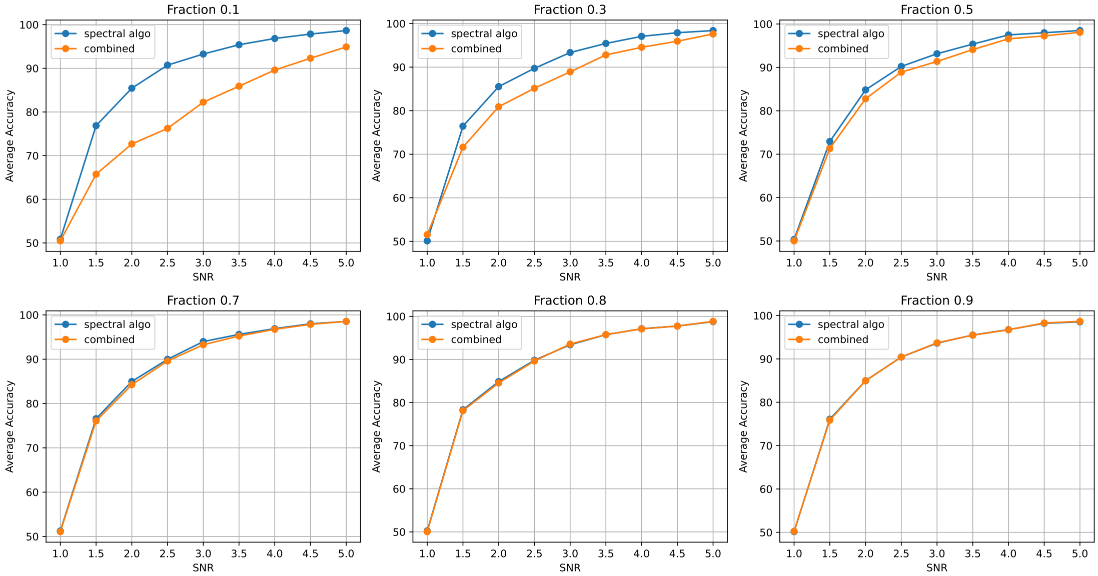


Figure 4.1.1: The blue line represents the accuracy of spectral algorithm, while orange line represents the accuracy of combined algorithm. The experiment results are averaged over 10 instances with $n = 10^4$, averge degree $c = 6$[2] and SNR ranging from 1 to 5.

From this figure, we can observe that the accuracy of the combined algorithm converges to the accuracy of the spectral algorithm as the fraction value increases. Unfortunately, it does not outperform the spectral algorithm. Nevertheless, this finding still inspires a promising direction for future research. If we investigate the assignment computed by the spectral algorithm by utilising some spectral properties of the given graph, can we identify the vertex assignments that are more likely to be correct? By doing so, we may be able to outperform the spectral algorithm in the easy regime.

[2] In order to test a wide range of SNR values, I have to increase the value of c, as SNR ≤ 3 for c = 3.

CHAPTER 5

Extending Greedy Recovery to Multiple Communities

After a thorough exploration of the two-community scenario, we now turn our attention to assessing the performance of the greedy recovery algorithm in the context of five-community case.

5.1 Experimental Results for k = 5 in Hard Regime

From Figure 5.1.1, we can observe that our greedy algorithm noticeably outperforms random guessing, even for the case of $k = 5$ communities.
However, our Claim 3.3.2 does not guarantee the effectiveness of our greedy recovery algorithm for sparse graphs when $k = 5$, as it requires $|\Re_v| > 12$ for the claim to be valid under the five-community case.

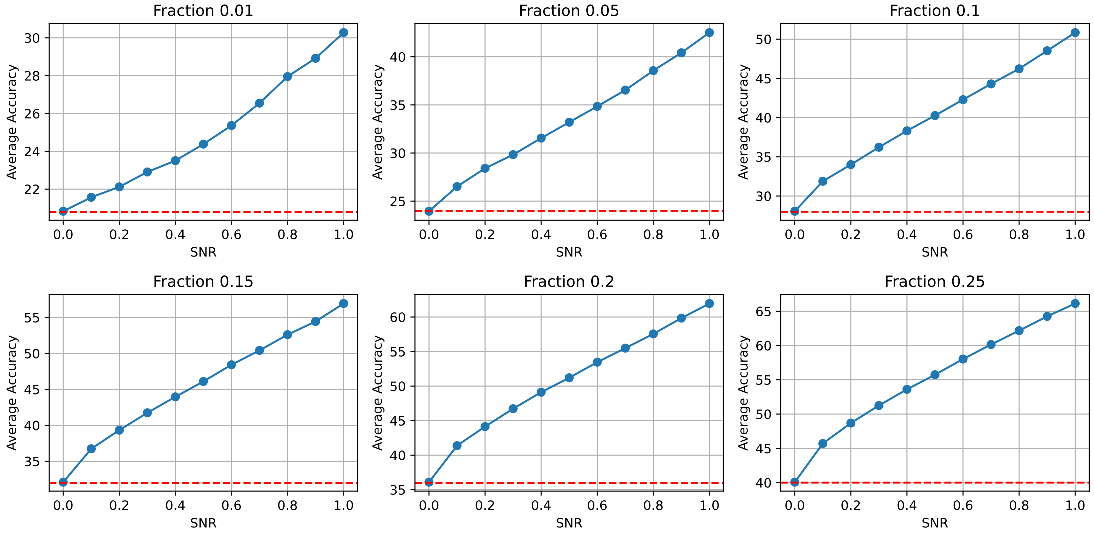


Figure 5.1.1: As in previous figures, the blue line and red dashed lines represent the accuracy of the greedy recovery algorithm and random guess, respectively. The experiment results are averaged over 10 instances with $n = 10^5$ and $c = 3$.

CHAPTER 6

Conclusion

In summary, this research project aims to solve the reconstruction task in expectation in the hard regime of the Kesten-Stigum bound under the setting of a partial seed set. We first introduced some important notations that laid the foundation for subsequent discussion. Then, we discussed an existing approach, the spectral algorithm based on the non-backtracking matrix, for the reconstruction task in the easy regime. Next, we stepped into the hard regime with the partial seed set setting and proposed a greedy recovery algorithm that performed considerably better than random guessing. We further improved the algorithm's accuracy with some heuristics in dense graphs. Finally, we generalised our algorithm to the case of 5 communities in the hard regime, and its superiority over random guessing remained consistent.

Overall, the objectives of this report have been fulfilled, and the effectiveness of the proposed algorithms was substantiated through a series of empirical and mathematical analyses.

6.1 Contributions

The main contributions of this project can be summarised as follows:

- Proved that the two definitions of reconstruction are equivalent for the purposes of this study.
- Established a realistic setting known as the partial seed set.
- Proposed a greedy recovery algorithm for the partial seed set setting that performs noticeably better than the random guess when $0 < \text{SNR} \leq 1$ for both $k = 2$ and $k = 5$.
- Improved the efficiency of the greedy recovery algorithm in sparse graph
- Improved the effectiveness of the greedy recovery algorithm in dense graph.
- Implemented all proposed algorithms and conducted comprehensive experiments to validate their performance.
- Provided a mathematical analysis of the effectiveness of the proposed algorithm.

6.2 Limitations

While this project has achieved all objectives, it is important to acknowledge certain limitations that influenced the scope and depth of this research. Due to my limited background in statistical physics and information theory, I have not explored this topic in depth, particularly the theoretical aspect of the community detection problem and the phase transition phenomenon. Moreover, there are also some limitations to my empirical and mathematical analysis, specifically:

- The bound in the proof of Claim 3.3.2 is not tight enough:

  1. the bound for $\frac{p_{out}}{p_{in}+p_{out}}$ is not tight, if we can link it with SNR value, we might be able to obtain a tighter bound for this formula.
  2. Currently, we only focus on $\Re_v$, if we also consider vertices in R_v, we might get a tighter bound for $\mathbb{P}(\sigma_v \neq \sigma'_v)$.

- Convert it to w.h.p form: Although this project focus on the algorithm performance in expectation, it's always interesting to ask if we can get the desired outcome with high probability (w.h.p.). I attempted to convert the result in Claim 3.3.2 as follows: we need $\mathbb{P}(\sum_V \mathbb{1}(\sigma_v = \sigma'_v) > \frac{n}{k}) = 1 - o(1)$, let $\mathcal{A}$ denote the event that there are at least $n - \frac{n}{k}$ vertices labelled incorrectly, then this is equivalent to saying $\mathbb{P}(\mathcal{A}) \leq \frac{1}{n^\alpha}$ for some $\alpha > 0$. And $\mathbb{P}(\mathcal{A}) \leq \binom{n}{n-\frac{n}{k}}\mathbb{P}(\sigma_v \neq \sigma'_v)^{n-\frac{n}{k}}$, but $\binom{n}{n-\frac{n}{k}} = \binom{n}{O(n)}$, which grows exponentially, given the upper bound we deduced for $\mathbb{P}(\sigma_v \neq \sigma'_v)$, this does not seem to align well with the w.h.p. form. So, we either need to deduce a tighter bound for event $\mathcal{A}$ and $\mathbb{P}(\sigma_v \neq \sigma'_v)$, or we may need to change our analysis framework, which probably requires some more advanced probabilistic tools.

- A more sophisticated analysis is needed: The points mentioned above suggest a need for a deeper analytical approach, potentially involving knowledge in random processes, percolation theory, and statistical mechanics, which is beyond my current background. Nevertheless, this work has sparked my interest in statistical physics, and I plan to study more courses in these areas in the future.

- Implementation on larger instances: If we implement the algorithm on graphs with a sufficiently large number of vertices that differentiates $\log(n)$ and then randomly pick $\log(n)$ vertices as our partial seed set, we may observe some interesting phenomena. Unfortunately, the parameters we used were $n = 10000$ and $n = 50000$, but $\log(10000) \approx 9$ and $\log(50000) \approx 11$. So there is not much difference. In order to obtain meaningful results, we might need to run the algorithm on impractically large graphs, which is infeasible given the limited time.

6.3 Future Work

Our work suggests several promising avenues for further research:

- Improve the accuracy of the combined algorithm to surpass the spectral algorithm: In this report, we randomly chose the assignments from the spectral algorithm to form our partial seed set. However, if we can find a way to select the assignments from the spectral algorithm that are more likely to have a correct assignment, would we be able to outperform the spectral algorithm in the easy regime?
- Use more insightful heuristic strategy: Our heuristic plan for algorithm 4 is based on the number of a vertex's neighbours whose membership is known and has improved accuracy by up to 12%. If we use some more clever heuristic, which might involve studying the graph structure, are we able to improve the accuracy further?
- A more elaborate analysis: Use knowledge from statistical physics and information theory to provide a more insightful analysis of the theoretical aspect of this topic.

- Exploring phase transition: If we conduct experiments on larger instances, particularly when the size of the partial seed set is $\mathcal{O}(\log n)$, can we observe any phase transitions occurring with respect to SNR, accuracy, and the size of the partial seed set? For example, in the hard regime, is there a phase transition for the algorithm's accuracy and the size of the partial seed set, where the algorithm's accuracy exhibits a notable jump when the size of the partial seed set is above a certain threshold?

CHAPTER 7

Project Management

7.1 Methodology

The methodology of this project can be divided into two parts, Research Methodology and Test Methodology.

Research Methodology:
This project originally aimed to study the community detection problem with the help of group theory. This choice was motivated by two factors: firstly, group theory is a powerful tool for analysing structural properties, making it a natural extension to graph theory; secondly, I am more familiar with abstract algebra. However, there are very few papers in this field, and most of them either focus on very small graphs or Cayley graphs, which are graphs constructed from groups. Both of these approaches do not fit this project very well. As a result, I had to shift my focus to the mainstream research in the field of community detection. After extensive research, I realised that there are numerous

variants of the community detection problem, each with different graph models, benchmarks, and consequently, different types of approaches. My supervisor and I decided to focus on community detection under the stochastic block model (SBM) because it proposes a phase transition phenomenon. However, as discussed in the previous chapter, the easy regime (SNR > 1) has been solved. Therefore, we had to focus our attention on the hard regime, but its hardness is supported by Theorem 1.1.2. Hence, we needed some extra information, and the reasoning for the choice of extra information is discussed in Chapter 3. Since we only have a local information, and one suitable algorithm dealing with local information is the greedy algorithm, if our approach is local optimal, then we have the hope that it might not be very distant from the globally optimal solution. The reasoning for the details of greedy algorithm is discussed in section 3.2.

Test Methodology:

The evaluation of the greedy recovery algorithm is based on the following functions:

- *generate_SBM(n, c_in, c_out, communities_size):* This function generates a graph with 2 built-in communities, where the inner-cluster probability is c_{in}/n and the across-cluster probability is c_{out}/n. It also returns the true community assignment σ.
- *generate_SBM_improved(n, c_in, c_out, communities_size):* This function generates a graph with multiple communities and the true assignment σ. To speed up the *generate_SBM* function when generating a graph with multiple communities, I utilise the function *nx.stochastic_block_model(sizes, p, nodelist=None, seed=None, directed=False, selfloops=False, sparse=True)* from

`https://networkx.org/documentation/stable/reference/generated/networkx.generators.community.stochastic_block_model.html`

The evaluation of the spectral algorithm is based on the following function:

- *non_backtracking_matrix(G):* This function constructs a non-backtracking matrix based on the graph G.

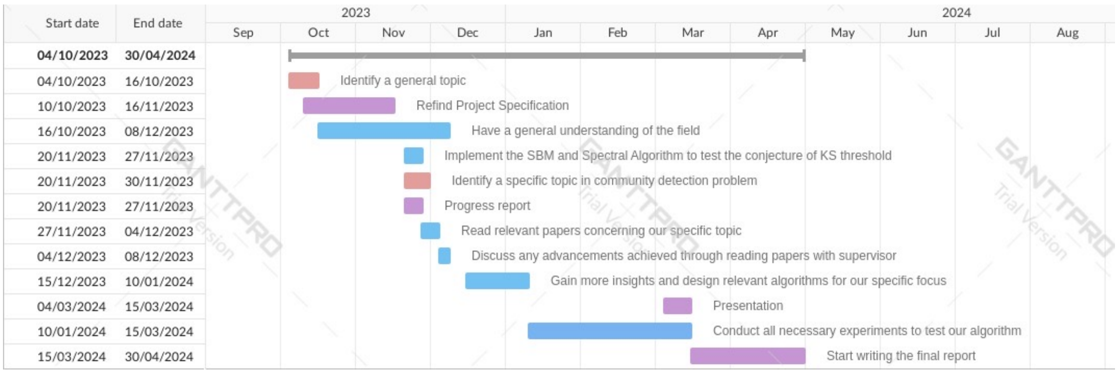
All implementations of the above functions are listed in the Appendices.

7.2 Software Used

- **Jupyter Notebook:** All implementations in this project are coded using Jupyter Notebook.
- **Overleaf:** This report is written in $\LaTeX$ on Overleaf.
- **GitHub:** GitHub is used for version control of both the implementations and the final report.
- **Chatgpt:** Chatgpt is used for proofreading and polishing some pieces of the final report.

7.3 Time Management

Term 1 was primarily dedicated to reading papers to gain an overview of the subject. However, I fell slightly behind schedule during this term due to two main reasons. Firstly, papers in this area generally required more time to digest than I had anticipated. Secondly, I was exploring potential topics, such



Start date	End date	
04/10/2023	**30/04/2024**	
04/10/2023	16/10/2023	Identify a general topic
10/10/2023	16/11/2023	Refind Project Specification
16/10/2023	08/12/2023	Have a general understanding of the field
20/11/2023	27/11/2023	Implement the SBM and Spectral Algorithm to test the conjecture of KS threshold
20/11/2023	30/11/2023	Identify a specific topic in community detection problem
20/11/2023	27/11/2023	Progress report
27/11/2023	04/12/2023	Read relevant papers concerning our specific topic
04/12/2023	08/12/2023	Discuss any advancements achieved through reading papers with supervisor
15/12/2023	10/01/2024	Gain more insights and design relevant algorithms for our specific focus
04/03/2024	15/03/2024	Presentation
10/01/2024	15/03/2024	Conduct all necessary experiments to test our algorithm
15/03/2024	30/04/2024	Start writing the final report

Figure 7.3.1: Project Timeline

as clustering with noisy queries, graph clustering via min-cut, and community detection via group action, in order to identify a specific topic that I could pursue.

Once the specific topic was determined, tasks in Term 2 proceeded smoothly. I completed all tasks on time and had extra time to explore some other interesting directions that were still within the scope of this project, such as the combined algorithm.

Bibliography

[Abb18] Emmanuel Abbe. 'Community Detection and Stochastic Block Models: Recent Developments'. In: *Journal of Machine Learning Research* 18.177 (2018), pp. 1–86. URL: http://jmlr.org/papers/v18/16-480.html.

[Abb23] Emmanuel Abbe. *Community Detection and Stochastic Block Models*. 2023. arXiv: 1703.10146 [math.PR].

[AS15] Emmanuel Abbe and Colin Sandon. *Community detection in general stochastic block models: fundamental limits and efficient recovery algorithms*. 2015. arXiv: 1503.00609 [math.PR].

[AS16a] Emmanuel Abbe and Colin Sandon. 'Achieving the KS threshold in the general stochastic block model with linearized acyclic belief propagation'. In: *Advances in Neural Information Processing Systems*. Ed. by D. Lee et al. Vol. 29. Curran Associates, Inc., 2016. URL: https://proceedings.neurips.cc/paper_files/paper/2016/file/6c29793a140a811d0c45ce03c1c93a28-Paper.pdf.

[AS16b] Emmanuel Abbe and Colin Sandon. 'Crossing the KS threshold in the stochastic block model with information theory'. In: (2016), pp. 840–844. DOI: 10.1109/ISIT.2016.7541417.

[AS16c] Emmanuel Abbe and Colin Sandon. *Detection in the stochastic block model with multiple clusters: proof of the achievability conjectures, acyclic BP, and the information-computation gap*. 2016. arXiv: 1512.09080 [math.PR].

[BLM15] Charles Bordenave, Marc Lelarge and Laurent Massoulié. *Non-backtracking spectrum of random graphs: community detection and non-regular Ramanujan graphs*. 2015. arXiv: 1501.06087 [math.PR].

[CY06] Jingchun Chen and Bo Yuan. 'Detecting functional modules in the yeast protein–protein interaction network'. In: *Bioinformatics* 22.18 (July 2006), pp. 2283–2290. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btl370. eprint: https://academic.oup.com/bioinformatics/article-pdf/22/18/2283/48841481/bioinformatics_22_18_2283.pdf. URL: https://doi.org/10.1093/bioinformatics/btl370.

[Dal21] Lorenzo Dall'amico. 'Spectral methods for graph clustering'. Theses. Université Grenoble Alpes [2020-....], Oct. 2021. URL: https://tel.archives-ouvertes.fr/tel-03454227.

[Dec+11] Aurelien Decelle et al. 'Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications'. In: *Physical Review E* 84.6 (2011). ISSN: 1550-2376. DOI: 10.1103/physreve.84.066106. URL: http://dx.doi.org/10.1103/PhysRevE.84.066106.

[DF89] M.E Dyer and A.M Frieze. 'The solution of some random NP-hard problems in polynomial expected time'. In: *Journal of Algorithms* 10.4 (1989), pp. 451–489. ISSN: 0196-6774. DOI: https://doi.org/10.

1016/0196-6774(89)90001-1. URL: https://www.sciencedirect.com/science/article/pii/0196677489900011.

[FH16] Santo Fortunato and Darko Hric. 'Community detection in networks: A user guide'. In: *Physics Reports* 659 (Nov. 2016), pp. 1–44. DOI: 10.1016/j.physrep.2016.09.002. URL: https://doi.org/10.1016%2Fj.physrep.2016.09.002.

[For10] Santo Fortunato. 'Community detection in graphs'. In: *Physics Reports* 486.3-5 (2010), pp. 75–174. ISSN: 0370-1573. DOI: 10.1016/j.physrep.2009.11.002. URL: http://dx.doi.org/10.1016/j.physrep.2009.11.002.

[HLL83] Paul Holland, Kathryn B. Laskey and Samuel Leinhardt. 'Stochastic blockmodels: First steps'. In: *Social Networks* 5 (1983), pp. 109–137. URL: https://api.semanticscholar.org/CorpusID:34098453.

[Jin+21] Di Jin et al. *A Survey of Community Detection Approaches: From Statistical Modeling to Deep Learning*. 2021. arXiv: 2101.01669 [cs.SI].

[Kar96] George Karypis. 'Graph partitioning and its applications to scientific computing'. PhD thesis. USA, 1996.

[Krz+13] Florent Krzakala et al. 'Spectral redemption in clustering sparse networks'. In: *Proceedings of the National Academy of Sciences* 110.52 (Nov. 2013), pp. 20935–20940. ISSN: 1091-6490. DOI: 10.1073/pnas.1312486110. URL: http://dx.doi.org/10.1073/pnas.1312486110.

[Lux07] Ulrike von Luxburg. *A Tutorial on Spectral Clustering*. 2007. arXiv: 0711.0189 [cs.DS].

[Mas14] Laurent Massoulié. 'Community detection thresholds and the weak Ramanujan property'. In: STOC '14 (2014), pp. 694–703. DOI: 10.

1145/2591796.2591857. URL: https://doi.org/10.1145/2591796.2591857.

[MNS14] Elchanan Mossel, Joe Neeman and Allan Sly. *A Proof Of The Block Model Threshold Conjecture*. 2014. arXiv: 1311.4115 [math.PR].

[MNS15] Elchanan Mossel, Joe Neeman and Allan Sly. 'Reconstruction and estimation in the planted partition model'. In: *Probability Theory and Related Fields* 162.3-4 (2015), pp. 431–461. ISSN: 0178-8051. DOI: 10.1007/s00440-014-0576-6.

[Moo17] Cristopher Moore. 'The Computer Science and Physics of Community Detection: Landscapes, Phase Transitions, and Hardness'. In: *Bull. EATCS* 121 (2017). URL: https://api.semanticscholar.org/CorpusID:1213533.

[MW03] James Moody and Douglas R. White. 'Structural Cohesion and Embeddedness: A Hierarchical Concept of Social Groups'. In: *American Sociological Review* 68.1 (2003), pp. 103–127. ISSN: 00031224. URL: http://www.jstor.org/stable/3088904.

[Nd11] Mariá C.V. Nascimento and André C.P.L.F. de Carvalho. 'Spectral methods for graph clustering – A survey'. In: *European Journal of Operational Research* 211.2 (2011), pp. 221–231. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2010.08.012. URL: https://www.sciencedirect.com/science/article/pii/S0377221710005497.

[New04] M. E. J. Newman. 'Fast algorithm for detecting community structure in networks'. In: *Physical Review E* 69.6 (2004). ISSN: 1550-2376. DOI: 10.1103/physreve.69.066133. URL: http://dx.doi.org/10.1103/PhysRevE.69.066133.

[New06] M. E. J. Newman. 'Modularity and community structure in networks'. In: *Proceedings of the National Academy of Sciences* 103.23 (2006), pp. 8577–8582. ISSN: 1091-6490. DOI: 10.1073/pnas.0601602103. URL: http://dx.doi.org/10.1073/pnas.0601602103.

[NG04] M. E. J. Newman and M. Girvan. 'Finding and evaluating community structure in networks'. In: *Physical Review E* 69.2 (2004). ISSN: 1550-2376. DOI: 10.1103/physreve.69.026113. URL: http://dx.doi.org/10.1103/PhysRevE.69.026113.

[NN12] Raj Rao Nadakuditi and M. E. J. Newman. 'Graph Spectra and the Detectability of Community Structure in Networks'. In: *Physical Review Letters* 108.18 (May 2012). ISSN: 1079-7114. DOI: 10.1103/physrevlett.108.188701. URL: http://dx.doi.org/10.1103/PhysRevLett.108.188701.

[Sch07] Satu Elisa Schaeffer. 'Graph clustering'. In: *Computer Science Review* 1.1 (2007), pp. 27–64. ISSN: 1574-0137.

[Sha48] C. E. Shannon. 'A mathematical theory of communication'. In: *The Bell System Technical Journal* 27.3 (1948), pp. 379–423. DOI: 10.1002/j.1538-7305.1948.tb01338.x.

[SPK22] Hyungsik Shin, Jeryang Park and Dongwoo Kang. 'A Graph-Cut-Based Approach to Community Detection in Networks'. In: *Applied Sciences* 12.12 (2022). ISSN: 2076-3417. DOI: 10.3390/app12126218. URL: https://www.mdpi.com/2076-3417/12/12/6218.

[WLJ20] Zhe Wang, Yingbin Liang and Pengsheng Ji. 'Spectral algorithms for community detection in directed networks'. In: *J. Mach. Learn. Res.* 21.1 (2020).

[Zha+12] Pan Zhang et al. 'Comparative study for inference of hidden classes in stochastic block models'. In: *Journal of Statistical Mechanics: Theory and Experiment* 2012.12 (2012), P12021. DOI: 10.1088/1742-5468/2012/12/P12021. URL: https://dx.doi.org/10.1088/1742-5468/2012/12/P12021.

APPENDIX A

key Code

```
def generate_SBM(n, c_in, c_out, communities_size):
    A = np.zeros((n, n))
    true_communities = {}
    start_idx = 0
    community_label = 0
    for size in communities_size:
        for i in range(start_idx, start_idx + size):
            true_communities[i] = community_label
            for j in range(i, start_idx + size):
                if np.random.rand() < c_in/n:
                    A[i, j] = 1
                    A[j, i] = 1

        start_idx += size
        community_label += 1

    for i in range(n):
        for j in range(i + 1, n):
            if true_communities[i] != true_communities[j] and np.
   random.rand() < c_out/n:
                A[i, j] = 1
                A[j, i] = 1
    G = nx.from_numpy_matrix(A)
    return G, true_communities # G is undirected
```

Listing A.1: Implementation of Stochastic Block Model for 2 communities case

```
def non_backtracking_matrix(G):
    G = G.copy()
    G_directed = G.to_directed()
    directed_edges = list(G_directed.edges())

    m = len(directed_edges)
    B = lil_matrix((m, m))

    edge_to_idx = {edge: idx for idx, edge in enumerate(
   directed_edges)}

    for edge in directed_edges:
        u, v = edge
        for w in G_directed.neighbors(v):
            if w != u:
                B[edge_to_idx[(u, v)], edge_to_idx[(v, w)]] = 1

    return B, edge_to_idx   # B is based on directed graph
```

Listing A.2: Implementation of Non-Backtracking Matrix

```
def Spectral_Algo(G):
    G = G.to_directed()
    B, edge_to_idx = non_backtracking_matrix(G)
    try:
        eigenvalues, eigenvectors = eigs(B, k=2, which='LR', tol=1e-3)
    except ArpackNoConvergence as e:
        print("ARPACK did not converge! Using what was found so far.")
        eigenvalues, eigenvectors = e.eigenvalues, e.eigenvectors


    second_eigenvector = np.real(eigenvectors[:, 1])
    node_sums = {node: 0 for node in G.nodes()}

    # Sum the components of the second eigenvector for incoming edges
    for u, v in G.edges():
        edge_uv_idx = edge_to_idx[(u, v)]
        node_sums[v] += second_eigenvector[edge_uv_idx]

    # Assign labels based on the sign of the sum
    labels = {node: 1 if node_sums[node] > 0 else 0 for node in G.nodes()}

    return labels
```

Listing A.3: Implementation of the Spectral Algorithm based on Non-Backtracking Matrix

```
def Improved_Greedy_recovery(G, R):
    updated_R = R.copy()
    q = deque()
    for v in updated_R:
        q.append(v)

    while q:
        v = q.popleft()
        for u in set(G.neighbors(v)) - set(updated_R):
            known_neighbors = set(G.neighbors(u)).intersection(
   updated_R)
            if known_neighbors:
                community_counts = Counter([updated_R[n] for n in
   known_neighbors])
                most_common_community = community_counts.
   most_common(1)[0][0]
                updated_R[u] = most_common_community
                q.append(u)  # Add the updated vertex to the queue
    for further propagation

    # Assign remaining unassigned vertices randomly to one of the
   communities
    unassigned_vertices = set(G.nodes()) - set(updated_R.keys())
    for v in unassigned_vertices:
        updated_R[v] = np.random.choice([0, 1])  # the two
   communities labeled 0 and 1

    return updated_R
```

Listing A.4: Implementation of Greedy Recovery Algorithm

For SBM of multiple groups, we utilise the function stochastic_block_model from NetworkX package.

```
def generate_SBM_improved(n, c_in, c_out, communities_size):
    p_in = c_in / n
    p_out = c_out / n

    block_sizes = communities_size
    k = len(communities_size)
    block_probs = [[p_in if i == j else p_out for j in range(k)]
   for i in range(k)]

    G = nx.stochastic_block_model(
        sizes=block_sizes,
        p=block_probs,
        directed=False
    )

    true_communities = {}
    start_idx = 0
    for community_label, size in enumerate(communities_size):
        for i in range(start_idx, start_idx + size):
            true_communities[i] = community_label
        start_idx += size

    return G, true_communities
```

Listing A.5: Implementation of SBM for multiple groups