University of Sheffield

Spectral Clustering with Fairness Constraints

Maxime Fontana

Supervisor: Pan Peng

A report submitted in fulfilment of the requirements
for the degree of MComp in Computer Science

in the

Department of Computer Science

December 26, 2021

Declaration

All sentences or passages quoted in this report from other people's work have been specifically acknowledged by clear cross-referencing to author, work and page(s). Any illustrations that are not the work of the author of this report have been used with the explicit permission of the originator and are specifically acknowledged. I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this project and the degree examination as a whole.

Name: Maxime Fontana

Signature:

Date: December 26, 2021

Abstract

Considering the growing importance of reducing bias in the world of Artificial Intelligence (AI) where algorithmic decision-making is often involved, we thereby introduce how fairness could be included as part of the evaluation of an AI system.
To support this idea, we work in the continuity of the paper that aims at incorporating fairness in Spectral Clustering [6], a graph partitioning algorithm.
First of all, we replicate the findings of the paper [6] and secondly, we extend the original testing on both synthetic data, by assessing the algorithms introduced by [6] on LFR benchmark networks [2] that aim at imitating real-world datasets' behaviours and real-world data, we also attempt to assess the resilience of these algorithms by rising and varying the perturbation parameters on both portions of our testing.
We thereby show that these algorithms do improve fairness on data and are also resilient to strong networks' perturbation with respect to this metric, however they hold some strong limitation in relation with the number of clusters and are therefore to be used given specific circumstances.

COVID 19 Impact Statement

This year has been particularly challenging to deal with, as being an international student with no family in the United Kingdom, it has been difficult to deal with being far from my family and my girlfriend. However, being in Sheffield was a necessity to me as I found it even more difficult to work in the country I am from.
Moreover, being extremely used and in need of social interaction, for instance, the fact that I used to work in cafés or in study rooms, I found it particularly hard not to be able to meet in person with both my friends and my supervisor throughout this project.
This pandemic has brought challenges to the way I used to live and work and has surely had impact on this dissertation procedure.

Acknowledgements

First of all, I would like to thank my supervisor for providing constructive feedback throughout this year, I particularly appreciated that the scope of our discussions were extended beyond a pure academic scope.
Then, I would also like to thank my girlfriend, Camille, who had to endure all my complaints throughout this year but who I love dearly and look forward to spend the summer with here, in the UK, in my new friendly environment.
Furthermore, I would like to thank all my friends both here in Sheffield and in Paris, France with whom I have been able to escape from university work throughout this year spent mostly in my room.
Last but not least, I would like thank my family, especially my parents and my brother that had managed to cheer me up even in most stressful and difficult times and who always know how to find the words to get me back on track. I love you all!

Contents

Chapter 1

Introduction

Machine Learning, since its first foundation the 1950s, has known a great period of prosperity and is nowadays, thanks to the emergence of Deep Learning this last decade, at the heart of the technical progress of the 21st century.
There is no doubt about the helpfulness of data, many organisations are driving their business strategies over data analysis in order to boost efficiency, profit and sales.
With a constant growth in population comes a constant growth in data, unfortunately, there is too much data that humans can handle and computers are here to help in order to draw valuable information out of these, hence the tremendous rise in Data Science over the last years.
These algorithms are meant to be implemented and developed to be situation-specific by computer scientists, or in other terms, humans. However, humans are known to be biased by many factors, by their ideas, by attributes, and this is a natural behaviour.

Therefore, this raises an issue of fairness in Machine Learning, as long as we deal with humans, we have to face snags in relation with Ethnicity, Equity and many others. This is a tackled topic today and as result, a lot of world-leading companies have shown interest in trying to bring free-from-bias systems around such as IBM, Google and Microsoft. This issue represents a segment of the social impact of Artificial Intelligence. Indeed, applications that would result in decision needs to be carefully considered in terms of discrimination, it would be unfair that individuals' personal attributes translate into discrimination in tasks where these individuals would be allocated financial or social help, for instance dealing with : Ethnicity or Socioeconomic Groups, however, it is important to understand that these features are necessary to a successful task's realisation and therefore relying only on these or removing all sensitive attributes would not be optimal. Also, it is worth pointing out that the so-considered fair outcome of an AI system does not simply rely on on the attributes that it uses but primarily on it's outcome.

This example is one out of hundreds but highlights well the complexity of fairness from a social point of view. The study we are going to be presenting is analogous to these problems,

especially to the problem of representation of minorities that need to be elevated to remedy this issue.

Moreover, this study extensively discusses the topic of **Clustering**, an un-supervised (that does not require any label to learn from data) technique to cluster data points that are similar into groups. Clustering is also extensively used nowadays due to it's efficiency and it's reliability, in fact, Clustering, as we demonstrate in this study, is embodied by a big body of research throughout the world and provides a great framework to expand on, many techniques used in Machine Learning rely on these and therefore researchers work on the improvement of it's efficiency or aim at incorporating all sorts of mathematical constraints within it.
In the context of this study, clustering could be thought as a pre-processing tool to choose a more 'fair' set of data points on which further data analysis could be performed or as an evaluation tool to see how our model performed in terms of discrimination for instance.

In this study we show how fairness can be incorporated in **Spectral Clustering**, which is a graph-partitioning algorithm widely used nowadays that make use of the spectrum on graph-interpretable matrices. This type of clustering is particularly appreciated as it is one of the few clustering algorithms that partitions data without making assumptions on the internal structure of this data.
This algorithm owns its foundations in 1970's in many papers in which the authors gradually introduce efficient ways to compute and use values at the heart of the Spectral Clustering's process. We are therefore to introduce the landscape of Clustering and Spectral Clustering.

Following on with the presentation of the remaining of this study, we will attempt to carry out a **1) Literature Survey on Fair Clustering** on the implications of fairness and the history of Spectral Clustering. Then, we will introduce the **2) Requirements And Analysis**, that being the aims of this project and we will conduct an analysis of the paper from Kleindessner et. al [6] presenting the main concepts in details. Furthermore, we will present the **3) Implementation** as well as the technical implications on both the replication of [6] and our extensions on Testing. Additionally, we will thereby introduce the **4) Results and Discussion** of the testing that has been performed and we will remind of the context of these results. Finally we will present the **5) Conclusions** derived from our experiments summarising our findings.

Chapter 2

Literature Review

In this Literature Review, we start by quickly going over what is implied by **Fairness** and how it came to be such a pre-dominant topic nowadays in the world of Artificial Intelligence, then, we explore the clustering landscape focusing especially on **Spectral Clustering**, finally, we will see how recent work has managed to combine both of these notions into algorithms that gave birth to the concept of **Fairness in Clustering**.

2.0.1 Fairness in Artificial Intelligence

First and foremost, it is important to highlight what initiated that body of research to investigate techniques that could solve issues related to fairness. Therefore, we are referring to some non-technical work that allows us to get the bigger picture and the implication of this problem.

Firstly, we can notice that this whole area of research was driven by the the popularity of some infamous incidents, for instance the Google Photos incident where black people were tagged as gorillas in 2015, this had a huge impact and became quite infamous. This brings the topic of race-representation in training data, we can imagine that if the algorithms behind this incident had been trained on representative data, this mistake would have most likely not occur. However, this is not the only regrettable event that happened that was related to fairness, some studies like the Human-Centered Artificial intelligence Department at the University of Stanford [4], has shown how healthcare datasets that had been used as training data were biased and could lead to a biased-decision making into patient's health condition [1]. Many other cases have been investigated, sometimes wrongly such as the COMPAS software which is a decision support tool to identify likelihood of a defendant to become a recidivist and which was used in major US states, but it is worth noticing the importance of free-from-bias systems in these really sensitive fields. That is why many world-leading industries have attempted to provide guidance on how to put fairness into practice by introducing guidance rule or evaluation systems, also, many lawful organisations have been introducing rules, like the European Data Protection Board.

Now and rightfully, algorithms are accountable for what they are meant to provide.

2.0.2 Spectral Clustering

It appears Spectral Clustering find it's roots in the early 1970's up to end of the 1990's, indeed, numerous early papers tackled graph partitioning using what is known as the graph spectrum, in other words, the eigenvalues and eigenvectors of a laplacian graph (a matrix representation of a graph), later on in this study we will walk through an example of Spectral Clustering. However, the work of Andrew Y. Ng et. al (2002) is the first paper to have popularised this clustering methods and to have brought formal proof to the efficiency of the partitioning of the graph [7], their algorithm is what we know today as Spectral Clustering. Ever since the 2000's, Spectral Clustering has been pre-dominant in the world of Data, it has found many applications such as image processing and speech segmentation.
Also, as we will see later in this study, Spectral Clustering's history is closely related to another popular clustering technique : *K-means*, indeed, this algorithm is used as the last step of the Spectral Clustering algorithm.
K-means clustering has been introduced by Hugo Steinhaus (1956) and later on developed by James MacQueen (1967), this is an un-supervised way to gather data points, initialising centroids in the data domain and recursively aiming at finding a correct partition of the data points based on the distance to these centroids.

2.0.3 Fairness in Clustering

At first, one of the kick-off paper in this area was initiated by Chierichetti et. al (2017), this paper shows how fairness objectives can be implemented in any clustering framework through the notion of fair-lets that represent fair subsets that preserve the clustering quality, also, this paper introduces a first notion of balance, that is that in each cluster, each group (population) should be represented within the same proportions as in the initial population[3]. This is the notion we will be using throughout this study. Then, Kleindessner et. al (2019) will be of a particular interest to us as their work aim at incorporating fairness into the Ratio-Cut minimisation problem (a graph cut objective function method) using a linear constraint on the data to which we are going to apply our K-means algorithm[6]. This area of research is still really new, and, as a result, this paper presents the only way so far to incorporate fairness into the Spectral Clustering framework, throughout this study we will extensively rely on it's findings and will further discuss what testing has been performed in this study.

Chapter 3

Requirements and Analysis

3.1 Standard Spectral Clustering

3.1.1 From Graph to Matrices

In this section we review how Spectral Clustering works and why it is a good option to graph-partitioning. We see it's link with other state-of-the-art algorithms. Moreover, Von Luxburg (2007) provides a detailed tutorial on Spectral Clustering in this paper[10], however here, we attempt to provide a descriptive walk-through from a linear algebra perspective in a more visual fashion.
Below is a pseudo-code for Un-normalised Spectral Clustering, we are therefore to analyse what is behind each step of the calculation.

Unnormalized Spectral Clustering

1. **Input** : A weighted adjacency matrix $W \in \mathbb{R}^{n*n}$
2. **Output** : A clustering of $[n]$ into k clusters
    - Calculate the Laplacian Matrix L
    - Compute the k smallest eigenvalues and store the corresponding eigenvectors as columns of a matrix H^{n*k}
    - Apply the K-means clustering to H

First of all, the Spectral Clustering (SC) algorithm is given the *adjacency matrix* as input, this matrix is a trivial way to represent a graph and it's internal structure as you can see below, notice this is a square matrix, in other words it has the same number of rows and columns as it represents internal connection in between the nodes.

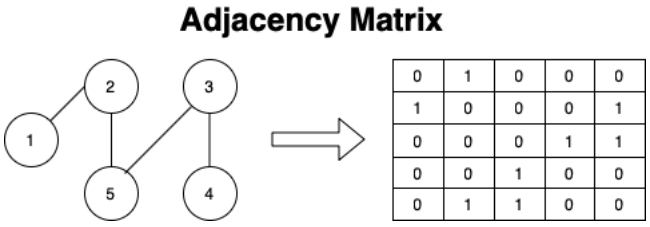


Figure 1 : Adjacency Matrix Graph Representation

As noticeable, row $\boldsymbol{i^{th}}$ represents all the connections of node $\boldsymbol{i}$.
Then, a second graph-representation matrix that is going to be useful to Spectral Clustering is the degree-matrix, this matrix is represented below in Figure 2, this matrix is square and symmetric as it is a diagonal matrix (all entries outside the main diagonal are all 0s).

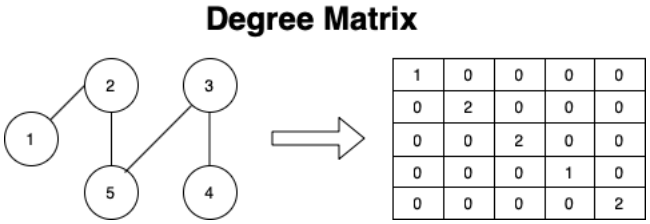


Figure 2 : Degree Matrix Graph Representation

Thanks to these 2 matrices, we can work out the Laplacian Matrix, this matrix has many useful properties that we are going to take advantage of later on in the algorithm. The formula for this is :

$$L = D - W \tag{3.1}$$

Where $\boldsymbol{D}$ is the *degree matrix* and $\boldsymbol{W}$ is the *adjacency matrix*.

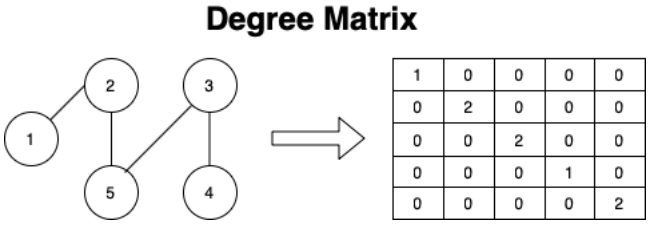


Figure 3 : Laplacian Matrix Graph Representation

This matrix is the most important of the Spectral Clustering framework, its properties are useful to various machine learning algorithms, but especially, they are of interest as it allows us to derive the *eigenvalues* and *eigenvectors* that are going to be used for the graph-partitioning problem that we introduce soon. But first, let's see what are these values, also called *the spectrum*.

3.1.2 The Spectrum

There are many ways to decompose matrices to find specific attributes of a matrix, here we focus on *Eigen-decomposition* in order to find the *Spectrum* of the matrix, in the context of Spectral Clustering, we are guaranteed to find an Eigen-Decomposition on our Laplacian matrix as it is a real-symmetric matrix, as opposed to some complex-matrices in which an eigen-decomposition is not guaranteed. We can summarise the problem of the eigen-decomposition as follows :

$$A \cdot v = \lambda \cdot v \tag{3.2}$$

Which can further simplified and summarised as follows :

$$A - (\lambda \cdot I) = 0 \tag{3.3}$$

Where $\boldsymbol{A}$ is the matrix we want to decompose, $\boldsymbol{\lambda}$ is the eigenvalue and $\boldsymbol{I}$ is the identity matrix which is a diagonal/square matrix. We summarise below how to solve this problem.

- Matrix A : $\boldsymbol{A - \lambda I}$
- Compute the eigenvalues by : $\boldsymbol{det(A) = 0}$ (polynomial-equation roots)
- Find the eigenvectors by computing : $\boldsymbol{nullspace(A - \lambda_n I)}$

We thereby obtain what is known as the Spectrum of the matrix, once decomposed we can use these eigen-values and eigen-vectors in such a way that it is going to allow us to, later on, assign data points computed from these to clusters and obtain our final clustering, this is because finding eigenvalues is equivalent to performing *Principal Component Analysis* (PCA), a technique to project our data onto lower-dimensions, we will not introduce PCA here but it is still the original idea behind eigen-decomposition. Now, let's see, how this Spectrum behaves with respect to a particular graphs.

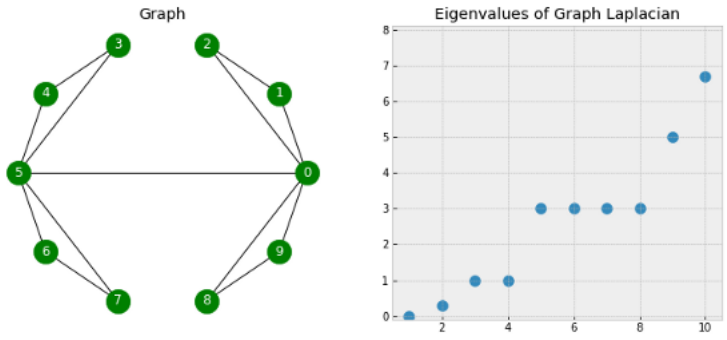


Figure 4 : Relation between Graph Connectivity and Eigenvalues
(https://towardsdatascience.com/spectral-clustering-aba2640c0d5b)

To explain, the number of 0 eigenvalues represent the disconnected components. Many studies have shown the strong geometrical correlation between the eigen-decomposition of the graph laplacian and the connectivity of the graph[9], also, the notion of *Cut* is at stake here, there are various ways to incorporate an objective function to get a cut (a partition of the vertices within a graph) that serves a specific function, by either aiming at maximising or minimising the objective function. In the context of Spectral Clustering, we aim at minimising the *Ratio-Cut* and it has been proven that finding the values of the eigenvectors of the un-normalised Laplacian Matrix is the approximated solution of the Ratio-Cut minimisation problem, we can therefore exploit this domain in order to apply our K-means algorithm.

3.1.3 K-means

This algorithm allows us to assign our data points (eigenvectors' values) within our domain to our pre-defined number of clusters $\boldsymbol{k}$. In it's most simplistic way, *K-means* runs as follows:

K-Means Algorithm

1. **Input** : number of clusters $\boldsymbol{k}$
2. **Output** : A clustering of $[n]$ into k clusters
    - Randomly position k centroids within the domain
    - **Until the centroids do not change :**
    - Assign each data point to it's closest centroid
    - Re-adjust the k centroids' position (as a mean of it's assigned data points)

Multiple variations of the K-means algorithm has been studied, especially in relation with the initialisation of the initial centroids, however here, we consider in this study the k-means++ [7] which makes sure that selected data points are away from each other, the K-means algorithm also work through many iterations where the best output is being kept in terms of convergence.

Now that our landscape has been fully defined, we will now see how the notion of 'fairness' from Chierichetti (2017) [3] can be implemented in it and we will introduce the algorithms presented in [6].

3.2 Fair Spectral Clustering

3.2.1 Notion of Fairness

First things first, in order to implement our constraints within the SC framework, we need to define our notion of fairness so we can, later on, implement it into our *cut objective function*. The notion at stake here is the one introduced by Chierichetti (2017) [3], it states as mentioned previously, that each group of population (given a sensitive attribute in our case) be represented within the same proportion as the initial population in our final clustering. This statement could be mathematically represented as a function of a cluster within our final clustering :

$$balance(C_l) = min \frac{|V_s \cap C_l|}{|V_{s'} \cap C_l|} \in [0, 1] \tag{3.4}$$

Where $\boldsymbol{V_s}$ represents a cluster before the run of fair algorithm, and $\boldsymbol{C_l}$ is a cluster obtained after the run, therefore, given a specific cluster, we iterate through every $\boldsymbol{V}$ and return the minimum value computed.

3.2.2 Un-normalised Fair Clustering

[6] presents a way to incorporate these constraints within the Ratio-Cut minimisation, they introduce this solution by putting a linear constraint (where results are forced into equalities or inequalities w.r.t other values) on the eigenvectors matrix fed to the K-means algorithms. This gives rise to the un-normalised version of fair Spectral Clustering.

Unnormalized Spectral Clustering with Fairness

1. **Input** : A weighted adjacency matrix $W \in \mathbb{R}^{n*n}$; group-membership vectors of length $n \in \{0, 1\}$
2. **Output** : A clustering of $[n]$ into k clusters
    - Calculate the Laplacian Matrix L
    - Compute the matrix F with columns v - V_s/n (v being the group-membership vector)
    - Calculate the matrix Z whose columns form an ortho-normal basis of the nullspace of F^T
    - Compute the k smallest eigenvalues and store the corresponding eigenvectors of $Z^T L Z$ as columns of a matrix Y
    - Compute a matrix $H = ZY$ and apply the K-means algorithms.

[6] introduces a detailed description of how theses matrices and calculations relate back to solving the fairness constraints implementation in the Ratio-Cut minimisation problem. In this study, we provide an code implementation of this algorithm and we will demonstrate how the implications of some computations done throughout this algorithm can be worked out.

3.2.3 Normalised Fair Clustering

One might wonder what Graph Laplacian to use, whether normalised or not, we remind that normalisation, similar to regularisation, is the fact to narrow down the domain's range to the range [0,1]. First of all, when it comes to the *Cut* problem, the normalised Graph Laplacian is the relaxation of a different objective function, the *NCut* minimisation problem, as seen before, this differentiation lies within the Spectrum of this matrix which holds different properties that the un-normalised one. Also, normalisation only matters in the case of *weighted* graphs, in other words, where connections (edges) between 2 vertices (nodes) differ within the structure of the graph. We will explore this concept further throughout this study.
[10] discusses the different aspects to take into consideration when choosing which Graph Laplacian to choose when partitioning with Spectral Clustering, his arguments are that the choice of which to use lies when the degrees of a graph are broadly distributed and therefore the trend is to opt in for the normalised for many advantages related to the spectrum, as initially suggested by [7].

Normalised Spectral Clustering with Fairness

1. **Input** : A weighted adjacency matrix $W \in \mathbb{R}^{n*n}$; group-membership vectors of length $n \in \{0, 1\}$
2. **Output** : A clustering of $[n]$ into k clusters
    - Calculate the Laplacian Matrix L
    - Compute the matrix F with columns v - V_s/n (v being the group-membership vector)
    - Compute the square root Q of $Z^T DZ$
    - Compute the k smallest eigenvalues and store the corresponding orthonormal eigenvectors of $Q^{-1}Z^T LZQ^{-1}$ as columns of a matrix X
    - Compute a matrix $H = ZQ^{-}1X$ and apply the K-means algorithms.

Above is a pseudo-code description of the algorithm using the normalised Graph Laplacian that we will be testing in this study. We also provide a code implementation of it and we explain the algorithmic implications of it.

3.3 Stochastic Block Model

The *Stochastic Block Model* (SBM), introduced in 1983 [8], introduces a way to generate graphs and communities in a probabilistic fashion, indeed, connections (or edges) are assigned based on probabilities with respect to a partition of the entire size of the population. Therefore, it introduces an easy, flexible and reliable way to test many community-finding algorithms to fully assess their capacity to perform *Exact Recovery* with respect to a user-defined graph. That allows us to set a ground-truth representing our pre-defined communities in order to quantify the efficiency of any clustering algorithm.

[6] comes up with a variant as there is a need in that specific context for 2 distinct communities in order to assess the fairness with respect to a standard clustering. Indeed, four probabilities are set in order to generate the graphs on which experiments are going to take place. Here, a distinction is in order, 'groups' are referred to as clusters that would be produced by regular clustering algorithms, those are brought up for obvious reasons, they are needed in order to assess fairness, whereas partitions produced by clustering algorithms holding fairness constraints are referred to as 'clusters'. This gives rise to the authors' variant of the Stochastic Block Model whose main parameters can be summed up below :

$$P(i,j) = \begin{cases} a, \ \textit{node } i \textit{ and node } j \in V_l = C_l \\ b, \ \textit{node } i \textit{ and node } j \in V_l \neq C_l \\ c, \ \textit{node } i \textit{ and node } j \in C_l \neq V_l \\ d, \ \textit{node } i \textit{ and node } j \notin C_l \notin V_l \end{cases}$$

Where $\boldsymbol{a, b, c}$ and $\boldsymbol{d}$ represents different probabilities of node $\boldsymbol{i}$ and $\boldsymbol{j}$ to connect together. Here we must respect the following : $\boldsymbol{a > b > c > d}$ so that our 2-communities set-up works properly.
A ground-truth will then be set and we therefore aim at retrieving it using our algorithms and we refer to a way to measure the efficiency of the fair algorithms as a way of returning the number of miss-classified vertices upon the said ground-truth. The insinuated way to retrieve this set is through the use of the symmetric difference between 2 sets :

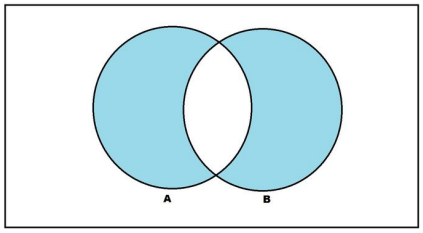


Figure 3 : Symmetric Difference of two sets (highlighted in blue)

We see in this article how to model such a variant of the *Stochastic Block Model* and how we can effectively put these measures and implications in practice with respect to our ground truth.

3.4 Aims

In the context of this study, our line of work would be to firstly, **replicate** all the results obtained by Kleindessner et. al (2019) by re-implementing the algorithms, the entire Stochastic Block Model framework and the results. Secondly, we attempt to bring the **testing** to a bigger scope by testing the algorithms on different data, both synthetic and real-world and also by bringing this data to different levels of perturbation. Lastly, we will try to summarise and derive the main **conclusions**.

Chapter 4

Design, Results and Discussion

4.1 Environment

We provide an **Python**-based implementation that makes extensive use of *NetworkX*, a library for graph manipulation and that allows us to use different visualisation tools to display results.
Also, we will be using *Numpy*, this library will also us to easily manipulate matrices as this project relies heavily on linear algebra concepts. Furthermore, we will use *Scikit-Learn*, which implements many heavily used Machine Learning algorithms and allow us to tweak their hyper-parameters in the way we want. In order to display our plots, we will be using *Matplotlib* throughout this study.
Finally, we thereby provide a link to a GitHub account gathering the different *Jupyter Notebooks* which are annotated with markdowns where appropriate in order to. fully explain the concepts behind the implementations : `https://github.com/Klodivio355/Fairness_Clustering`. We also make sure that all results presented later on are reproducible via this GitHub repository.

4.2 Synthetic Data

Initially here, we aim at testing our 'fair' algorithms on synthetic data, on random graphs. This is mainly to prove that our algorithms are able to recover a fair ground-truth as shown by [6]. We will therefore introduce how we design such a variant of the Stochastic Block Model, we will discuss it's relations with our 'fair algorithms', the metrics we use in order to assess their efficiency. Also we will present the experiments carried out in this context as well as the discussions that can be deduced from those.

4.2.1 Stochastic Block Model

As presented before, the Stochastic Block Model allows us to build a random graph with respect to user-defined variables and therefore allows us to test ground-truth recovery on

many community-finding algorithms. Now, we look at how we design this in our study and also we further discuss the limitations introduced by Kleindessner et. al . This is starting off by analysing to what extent our algorithms are capable of recovering the ground-truth we set before running these experiments.

Theorem

Throughout [6]'s mathematical proofs, the authors introduce upon which it is being proven that with high-probability that the two fair algorithms are able to recover the fair ground-truth if the population size is large enough. This theorem is as follows :

$$|V_s \cap C_l = \frac{n}{kh} \tag{4.1}$$

With $\boldsymbol{V_s}$ being the classical 'unfair' cluster, $\boldsymbol{C_l}$ being the fair cluster we aim at recovering, the size of the population, $\boldsymbol{k}$ being the number of fair clusters and finally $\boldsymbol{h}$ the number of original unfair clusters also referred to as groups.
In other words, this theorem represents the distribution of our clusters in terms of population size, this theorem is entirely satisfied when a clustered dataset is partitioned such that every cluster is of the same size. We will therefore try to replicate these results and see to what extent this theorem can be unsatisfied.
The necessity to assess the *recovery* capacity of our algorithms resides in highlighting how these algorithms truly work in their partitioning process, and also this is a first glance taken at the robustness, in other words, what is the environment in which these algorithms can thrive and perform at their best and to what extent this environment can be distorted, and in what fashion.

Ground Truth

Therefore, we must define how, precisely, we set our ground truth. We have seen previously the two parameters $\boldsymbol{k}$ and $\boldsymbol{h}$, the aim is to tune these parameters and therefore create different models on which we can try out our algorithms. One way to achieve this, is to take the $\boldsymbol{h}$ groups and to divide each into $\boldsymbol{k}$ subsets, if $\frac{\boldsymbol{h}}{\boldsymbol{k}}$ is a whole number and the theorem is satisfied we should obtain a recovery by these algorithms. We can summarise the way we create these models as follows :

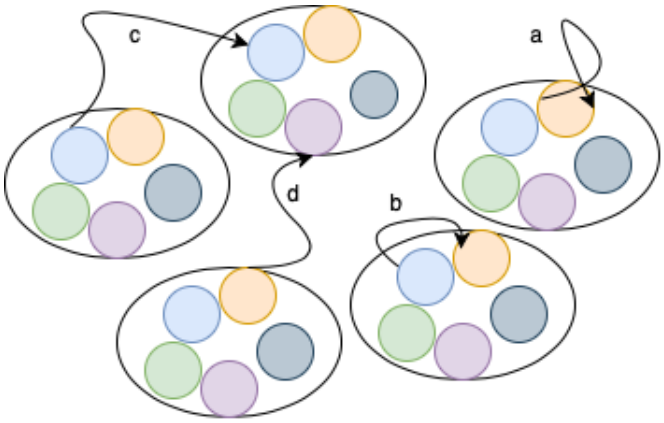


Figure 4 : Ground Truth Generation Illustration

We therefore construct our model by assigning probabilities previously defined in-between those sub-clusters in order to link them together. We refer to the algorithm below to show how a ground-truth respecting the theorem **(4.1)** fully can be computed.

Algorithm 1 GroundTruthGenerator($\boldsymbol{h}$, $\boldsymbol{k}$, $\boldsymbol{arr = 1...n}$)

count = 0
idx = k
for $i = 1, 2, \ldots h$ **do**
    for $i = 1, 2, \ldots k$ **do**
        groups ± arr[count : length(labels) // (h×k) + count]
        count ± length(labels) // (h×k)
    end for
end for
// Now we process all the groups of size $\frac{\boldsymbol{n}}{\boldsymbol{hk}}$
while $idx < \boldsymbol{hk}$ **do**
    if $idx + k < \boldsymbol{h} * \boldsymbol{k}$ **then** results ± results[index+k]
    end if
end while
return results

An 'unbalanced' clustering, that being a ground-truth that does not respect the theorem and that would be unbalanced, would have to be tuned in order to take into account the percentages that each cluster occupies.
Similarly, with respect to Figure 4, we could see how an unbalanced clustering could be constructed, indeed, subsets could be added to each **group** and therefore boosting the percentage of a specific cluster.
We have thereby presented how we create our ground-truth, please refer to the Github repository previously linked in the notebook into the *GroundTruthGen*-called file.
Now, we need a way to make sense of that ground-truth, we need a function to assess the actual recovery capacity of the algorithms we are trying out, this is what we will present in the next section.

Error

As previously mentioned, we refer to the error as the portion of mis-classified vertices, we have also seen the close relation to the *symmetric difference* between those sets. We therefore refer to the algorithm below as a way of accomplishing that.

Algorithm 2 Error(***groundTruth***, ***fairLabels***)

for $i = 1, 2, \ldots max(fairLabels)$ **do**
    Return the indices of the elements of the same group and store them
end for

for $i = 1, 2, \ldots len(groundTruth)$ **do**
    for $i = 1, 2, \ldots len(fairLabels)$ **do**
        compute the symmetric difference between every subsets across the labels and the ground truth
        return the minimum as the percentage
    end for
end for

Now, we have a way to make sense of our ground-truth, we can assess the recovery of our fair algorithms.
Earlier, we discussed another metric, the balance, so far, we have not designed how to replicate the balance notion from Chierichetti (2017). We will therefore look into that in the next section.

Balance

We aim at measuring the fairness of a data-set by relying on a *sensitive attribute*, this a notion that us, as human, we make sense of as being a logical attribute for which it makes sense to have fairness on, for instance, many ethnicity or gender attributes are sensitive attributes whereas height and color of eye is rarely the case.
We thereby introduce a way to evaluate this notion.

Algorithm 3 Balance(***population***, ***fairLabels***)

for $i = 1, 2, \ldots max(fairLabels)$ **do**
    Compute the distribution (as a percentage) of each population.
    score = 0
    score -= The difference between the initial distributions and the actual distribution from fairLabels.
end for
Return the score averaged over the number of clusters in fairLabels

We refer to the algorithm 3's pseudo-code described above as an intuitive way to implement this notion. We will therefore assess the balance of our algorithms' output using this algorithm.

Results and Discussion

We therefore run experiments in order to replicate the results obtained by [6] on their variant of the Stochastic Block Model. The very first step we aim at showing here how the algorithms recover the ground-truth as a function of n :

- 1 : Theorem satisfied
- 2 : Theorem gradually unsatisfied

First of all, let's gather our results for when the theorem is satisfied and provide discussion on them.

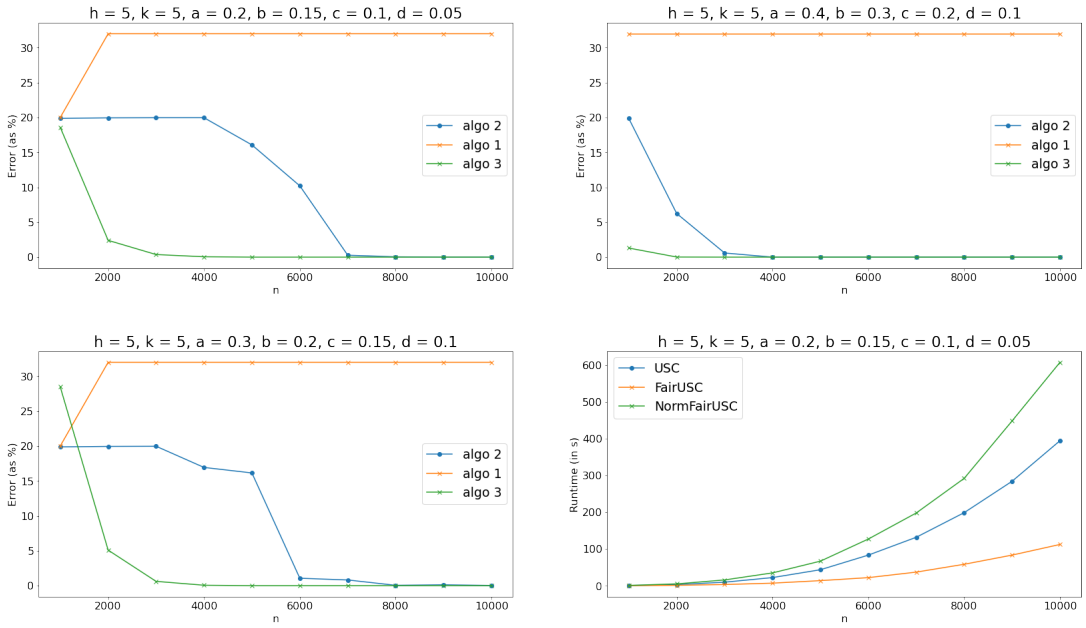


Figure 5 : SBM Results - Theorem Satisfied

The results in Figure 5 replicates the results of the upper row of Figure 2 in [6]. The two plots at the top and the plot at the bottom left-hand corner all represents experiments carried-out to assess the recovery, they are all assessing the same ground-truth with $\boldsymbol{h = 5}$ and $\boldsymbol{k = 5}$, this configuration is analogous to Figure 4, we therefore play around with the parameters $\boldsymbol{a, b, c}$ and $\boldsymbol{d}$ which we will refer to as being the *variant's parameters* introduced by the variant introduced by [6]. On the bottom right-hand corner, we analyse the run-time of those algorithms as a function of n.

What is worth noting here is the similarity with the results computed by the paper, indeed, we can see that with clearly distinct variant's parameters, for instance the plot at the top right-hand corner, both algorithm 2 (un-normalised fair) and algorithm 3 (normalised fair) recover the ground-truth fully. However, when these parameters variant's variance start to decrease, being more close together, we can see that algorithm 2 recovers the ground truth only when $\boldsymbol{n}$ is large whereas algorithm 3 is able to recover much more quickly, this behaviour

is also noticeable on the bottom left-hand corner for which the parameters are also slightly unevenly spread.
Also, in terms of run-time we can see that despite the fact that the algorithm 3 recovers the ground-truth sets much more quickly it is also at the expense of a larger run-time. This also comes from the proofs and studies of Kleindessner et. al on their algorithms, however, all these run-times are in $\boldsymbol{\Theta(n^3)}$.
The take-home message here is to notice that algorithm 3 can recover the ground-truth faster than algorithm 2, but at the expense of a higher running time and also that when the theorem is satisfied, we can recover the ground-truth with both algorithms.

We now look into the results for when the theorem is slightly unsatisfied, for instance by setting $\boldsymbol{k_1 = 60\%}$ and $\boldsymbol{k_1 = 40\%}$. Figure 6 below summarises this design.

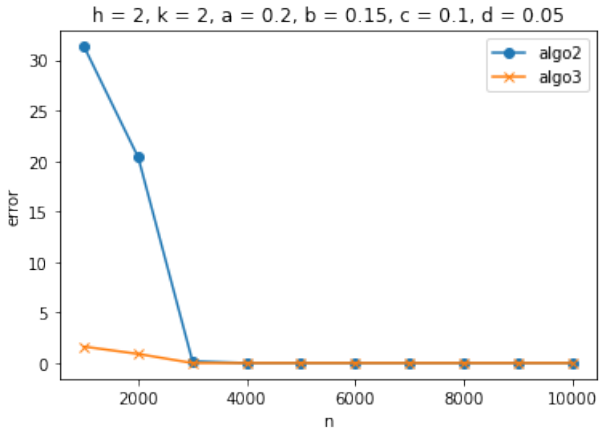


Figure 6 : Theorem slighly unsatisfied

Therefore, we can notice that when the theorem is slightly unsatisfied, both the algorithms can easily recover the ground-truth.

We now look into more severe violations of the theorem, for instance, by setting $\boldsymbol{k_1 = 30\%}$, $\boldsymbol{k_1 = 30\%}$, $\boldsymbol{k_3 = 20\%}$, $\boldsymbol{k_4 = 10\%}$ and $\boldsymbol{k_5 = 10\%}$, this ground-truth has been computed with $\boldsymbol{h = 3}$. Figure 7 below presents such a unevenly distributed clustering.

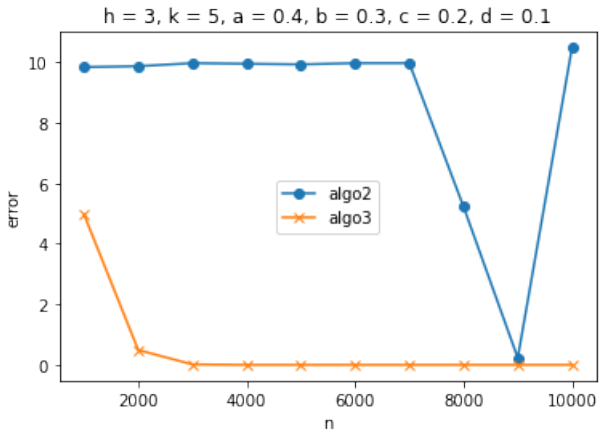


Figure 7 : Theorem strongly violated

It is worth noting here that, still in accordance with [6], algorithm 2 (un-normalised fair algorithm) is unable to recover the ground-truth, even though an anomaly can be notified on $\boldsymbol{n = 8000}$ and $\boldsymbol{n = 9000}$, in 100% of the cases, the algorithm was still unable to recover once $\boldsymbol{n = 10000}$, we do not have a clear-cut explanation on this behaviour, however, it is probably due to luck. What is key in this figure is that we can clearly see that where algorithm 2 fails at recovering the ground truth, algorithm 3 perfectly succeeds and manages to recover quickly this ground-truth at a early stage of the experiment, at $\boldsymbol{n = 3000}$.

Perturbation

In this section, we attempt to derive conclusions in relation with the ground-truth recovery when different perturbation parameters are added to the models before the algorithms are being run.
In real-life situations, networks are consistently changing, this can be simulated by adding perturbation to these, in this study, we present two different types of perturbation being carried out on the graphs we perform ground-truth recovery on.

The first perturbation simulation we implement is **Random vertices removal**, it simply consists in removing a portion of vertices at random.

The second perturbation simulation we show is **Degree-based vertices removal strategy**, here we sort the vertices with respect to their degrees, which is the number of connections they hold, and we then remove a portion of those randomly. This strategy helps us show how robust our algorithms are with respect to the *Cut* objective function they aim at optimising.

The experiments carried out can be summed up in Figure 8 below.

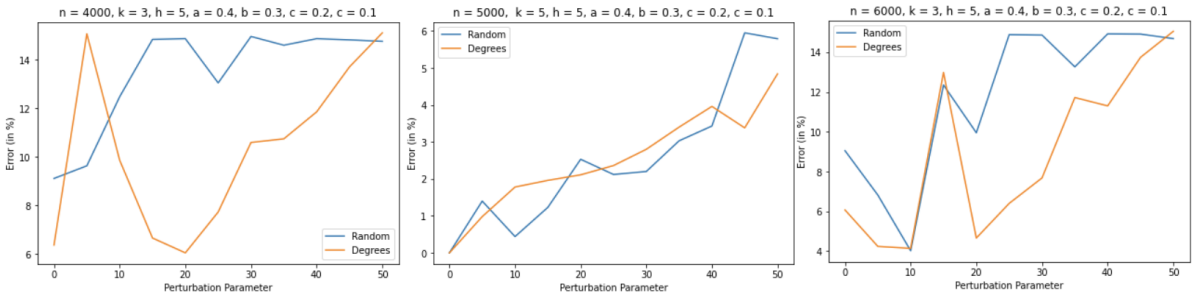


Figure 8 : Perturbation to the SBM

As an interpretation of those results, we can clearly notice that the algorithms are impacted to those perturbations and seem to be close but still unable to recover the ground truth. It is normal not to obtain full recovery, however, an ideal behaviour with respect to this perturbation would be what has been obtained with the *middle* plot where the parameters fully satisfies the *theorem*, that being a **linear** and **moderate** impact. On the other hand, we can notice that when the model is close to convergence and violates the theorem, the perturbation has an strong impact on the recovery and make the algorithms sensible to those parameters, that is the case on both the **left** and **right** figure.
As for the type of perturbation the model is more sensitive to, we can see that the *random vertices removal* strategy seems to have a bigger impact on the Stochastic Block models where the theorem is not satisfied, and where ground-truth relative error is considered, however there is no clear-cut answer.
The take-home message here is that the algorithms are still quite robust to these perturbations, even brought to high levels. We are therefore to try to implement those perturbation strategies onto real-world data.

4.2.2 LFR benchmark networks

A widely-used method to community-detection is to use the LFR benchmark networks introduced in [2], we will therefore describe how these networks work and how they are of a particular interest here. In this section, we will evaluate the fairness, or the balance, of these networks with respect to the number of clusters we want to partition the data into.

Description

The LFR networks are real-world like synthetic graphs, in order to accomplish that, they make use of real-life plausible mathematical concepts in order to model the distribution of node degrees and communities size and to build random graphs, such as the **power law** distribution which models the domination of minorities in many contexts involving sociology as shown in Figure 9.

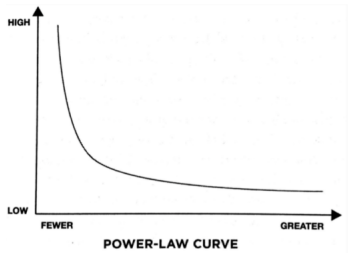


Figure 9 : Power Law Distribution
(https://visible.vc/blog/understanding-the-power-law-curve-of-vc/)

This is why these networks are particularly appreciated in *Graph studies*. Thanks to this real-life plausible distribution, it makes sense to assess the metric at the heart of our study : the *fairness* or in other terms, our balance.

Results and Discussion

First of all, let's assess the balance of some models with respect to algorithm 2 (un-normalised fair) as a function of k, Figure 10 below presents the experiments that have been carried out with respect to this configuration.

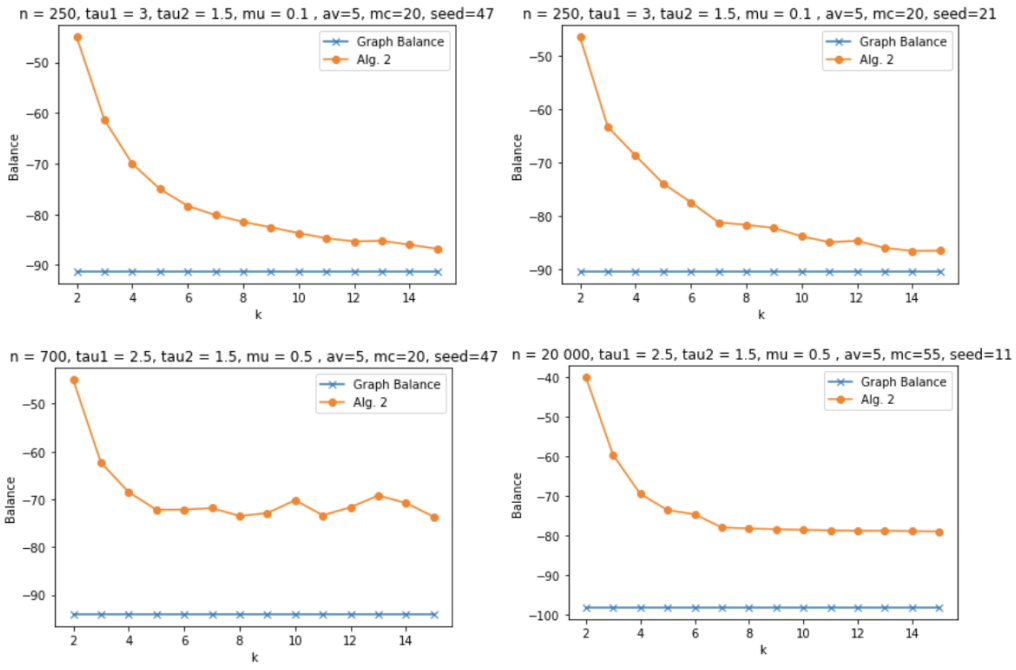


Figure 10 : Assessing balance of algorithm 2 on LFR networks

Throughout these experiments, we can clearly notice that the balance decreases as the number of clusters we aim at partitioning our graph into increases. These plots represents different communities as it uses different parameters with different population sizes. However,

the main point is that the algorithms do improve the fairness of these graphs, as we can see that the initial fairness of the graphs is always lower than the output of algorithm 2.
Furthermore, we will attempt to bring the testing further by implementing real-world data testing and we show the actual difference in terms of real-world performance with respect to a given sensitive attribute, because so far, the ground-truth was just set on the vertices with no given specific attribute.

4.3 Real-World Data

Testing these algorithms on real-world data is a necessity as the improvement of fairness over this type of data is the primary use of these algorithms. As mentioned previously in this study, one good use-case of *Fairness Clustering* would be to use it as a pre-processing technique in order to choose a more fair and representative dataset.
Therefore we will start by introducing the different datasets we are going to work on and why they are of interest here, then we will present the different results of the experiments carried out and their interpretation. Finally, we will introduce perturbation into those and assess the capacity of our algorithms to improve the balance of these data sets.
Make sure to check out the Github Repository where data analysis is being performed on each of the dataset in respective files.

Datasets

We thereby introduce the different data sets and their respective attributes we are going to work on :

- 1 : **DrugNet**[11] with respect to the gender of the data points (samples). This is a network representing the acquaintanceship between drug users, it consists of 286 vertices.
- 2 : **DrugNet**[11] with respect to the ethnicity of the data points (samples). This is a network representing the acquaintanceship between drug users, it consists of 286 vertices. The different ethnicities at stake here are : African Americans, Latinos and others.
- 3 : **FriendShipNet**[5] with respect to the gender. This consists of 133 data points representing friendship between students.
- 4 : **FacebookNet**[5] with respect to the gender. This consists of 155 vertices indicating friendship on the social network : Facebook.

These data-sets therefore represent strong social interest and are straightforward to interpret, we will therefore refer to the ground-truth as being the distribution of the networks with respect to a given attribute.
We therefore aim at showing whether algorithm 2 and 3 (both un-normalised and normalised version of fair spectral clustering) are improving fairness compared to standard clustering.

Results and Discussion

Starting off with **DrugNet**[11], we will run experiments with respect to the ground-truth of the sensitive attribute of **gender**.
Figure 11 below summarises the experiments that have been performed on this dataset with respect to this attribute.

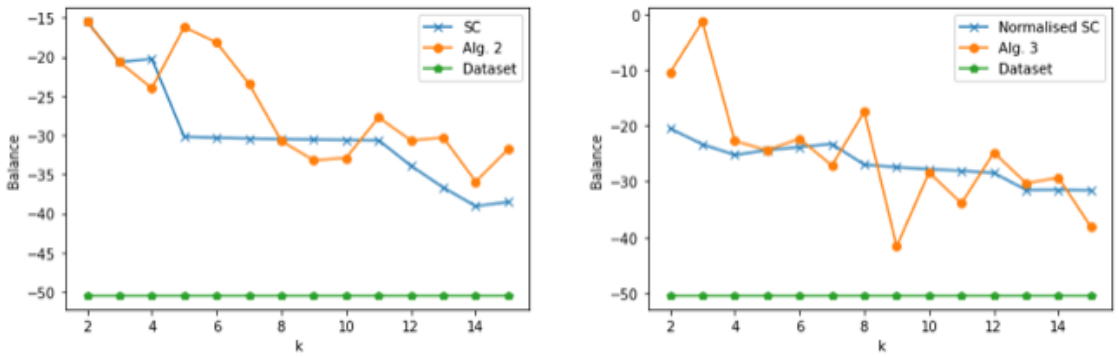


Figure 11 : **DrugNet**[11] (gender) balance assessment

In the figure above, we notice that in both cases, the fair clustering algorithms do improve the fairness of the partitioned datasets over standard clustering (4% on the left, 2% on right) averaged over $\boldsymbol{k}$ and even more with respect to the dataset's balance.
However, we can also notice that the balance decreases as the number $\boldsymbol{k}$ increases, this is, once again, noticeable on real-life data.

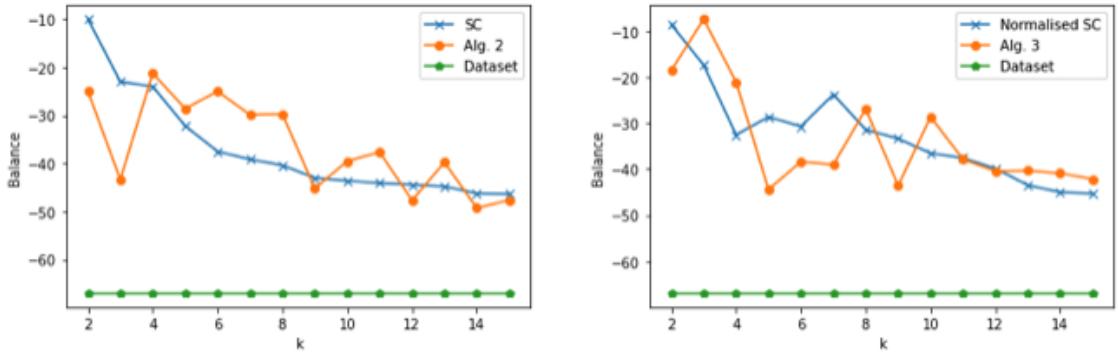


Figure 12 : **DrugNet**[11] (ethnicity) balance assessment

As for the ethnicity-based ground truth, the same observations can be pointed out. However, we notice that the performance of algorithm 3 over standard normalised Spectral Clustering is much more weak.

Moving on to **FriendshipNet**[5] (Figure 13) and **FacebookNet**[5] (Figure 14), we provide

further testing assessing the balance over real-world data and overall, the same analysis can be carried out.

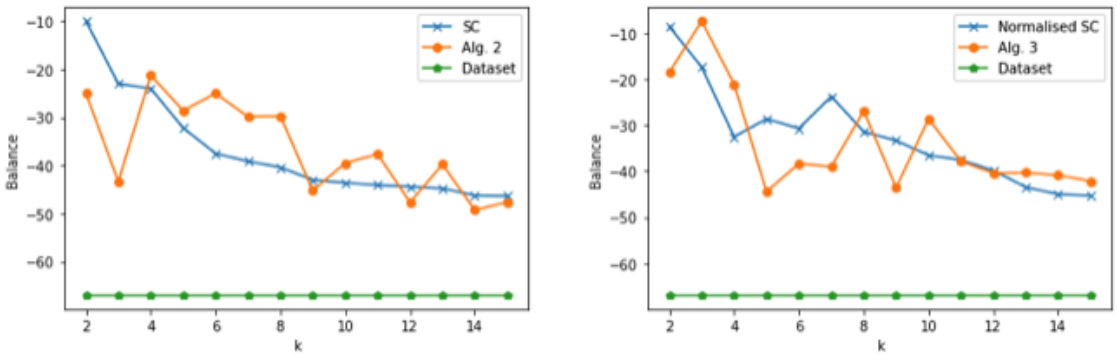


Figure 13 : **FriendshipNet**[5] (gender) balance assessment

On this dataset, the fair algorithms perform better in terms of fairness in comparison with standard clustering with respectively (from left to right) 4% and 2.5% boost in balance.

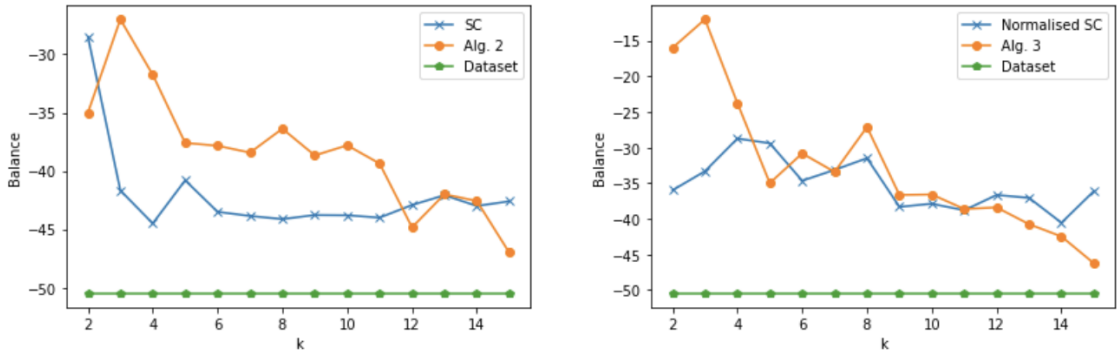


Figure 14 : **FacebookNet[5]** (gender) balance assessment

On the Facebook Network [5], the fair algorithms upgrade the balance to respectively 4% and 3% compared to standard clustering.

In all cases, we can neatly observe the negative correlation with the number of clusters $\boldsymbol{k}$. We can also observe that, on average over $\boldsymbol{k}$, the fair clustering algorithms do improve the balance of the partitioned dataset compared to standard clustering, even more when it comes to comparing with the initial dataset's balance.
We can also observe that the fair clustering performs outstandingly well for low $\boldsymbol{k}$s.

Fairness being at the essence of this study, we show in the next part to what extent real-world data can be subject to perturbation.

Perturbation

In this section, we assess the efficiency of the fair spectral clustering to guide the standard Spectral Clustering framework towards a more fair partitioned dataset despite it being perturbed using the 2 methods previously mentioned :

- 1 : Random vertices removal
- 2 : Degree-based vertices removal strategy

Taking the example of the ***Friendship*** dataset, our experiments can be summed up in the table below :

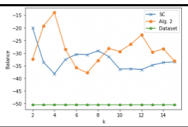
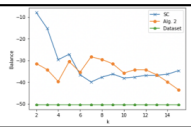
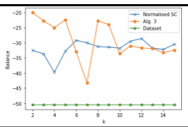
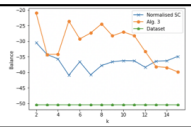
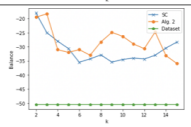
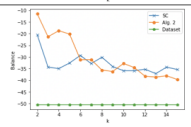
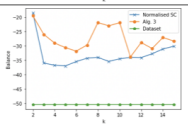
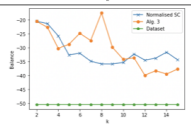
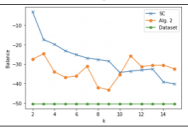
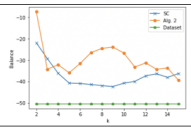
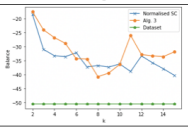
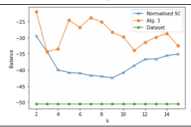
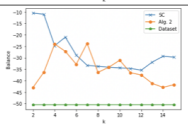
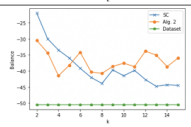
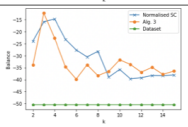
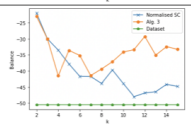
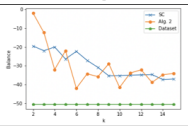
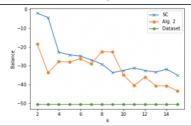
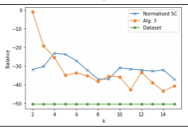
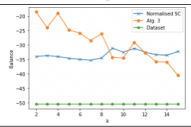
FriendShip	Degrees (algo2)	Random (algo2)	Degrees (algo3)	Random (algo3)
10%				
20%				
30%				
40%				
50%				

Figure 15 : **FacebookNet**[5] Perturbation Table

Look at Figure 15, we can see that despite the data being brought to high level of perturbations, this does not prevent it from improving the balance on average, this is due to the way that fairness is incorporated within the *Cut-minimisation* problem, as it guides the current standard clustering towards a more fair output if it exists, it also means that these fair algorithms are highly dependent on the initial distribution of the data and the internal structure of the network.

Chapter 5

Conclusions

Throughout this study, we have discussed and provided explanations on why fairness is important and how Spectral Fair clustering could be a handy tool towards a more fair way to both assess and choose more representative datasets on which life-changing decision-making algorithms are being run on.
We have performed analysis on the Spectral Clustering framework and how objective cut functions could be tweaked to incorporate notions and in this case, fairness. [6].
We have shown and tested the efficiency of the fair algorithms on ground-truth recovery and tested the variant of the Stochastic Block Model introduced by [6], we have concluded that despite a more dominant running time, the Normalised Spectral Fair Clustering is accurate and fast at retrieving a ground where the Un-normalised Spectral Fair Clustering algorithms fails when data clustering is being unevenly distributed.
Moreover, we have tested these algorithms on the LFR benchmark networks[2] in relation with the balance, that allowed us to notice a negative convergence with the number of clusters we aimed at getting and also the fact that the fair algorithms were actually making the data distribution more fair.
Furthermore we have brought our data to different levels of perturbation by simulating 2 types of networks misbehaviour, either targeted randomly or at high-degree nodes in order to see what impact would this feature have on the cut function our algorithms aim at minimising, we have concluded that in terms of recovery, our algorithms were sensitive and presented non-linear behaviours with respect to their portions of mis-classified vertices.
Lastly, we tested our data on Real-World data both non-perturbed and perturbed, that allowed us to prove [6]'s explanations on how the algorithms were incorporating fairness and that there was no guarantee as for the outcome's fairness of the partitioned data, indeed on both perturbed and non-perturbed, fairness was improved overall as an average over $\boldsymbol{k}$, still struggling as this parameter increases but most importantly, that the algorithm was highly dependent on the initial network's distribution in order to derive a fair output.

Bibliography

[1] Amit Kaushal, R. A., and Langlotz, C. Toward fairness in health care training data.

[2] Andrea Lancichinetti, Santo Fortunato, F. R. Benchmark graphs for testing community detection algorithms.

[3] Flavio Chierichetti, Ravi Kumar, S. L. S. V. Fair clustering through fairlets.

[4] Lee, M. K., Grgić-Hlača, N., Tschantz, M. C., Binns, R., Weller, A., Carney, M., and Inkpen, K. Human-centered approaches to fair and responsible ai.

[5] Mastrandrea, R., F. J., and Barrat, A. http://www.sociopatterns.org/datasets/high-school-contact-and-friendship- networks/.

[6] Matthäus Kleindessner, Samira Samadi, P. A. J. M. Guarantees for spectral clustering with fairness constraints.

[7] Ng, A. Y., Jordan, M. I., and Weiss, Y. On spectral clustering: Analysis and an algorithm.

[8] Nidheesh N., K. A. A. N. An enhanced deterministic k-means clustering algorithm for cancer subtype prediction from gene expression data.

[9] Tao Xiang, S. G. Spectral clustering with eigenvector selection.

[10] von Luxburg, U. A tutorial on spectral clustering.

[11] WEEKS, M. R., C. S. B. S. P. R. K. . S. J. J. https://sites.google.com/site/sfeverton18/research/cohesion-and-clustering.