HAL
open science

A variant of the spectral clustering method using l1 -penalty to promote sparse eigenvectors basis

Mélanie Blazère

To cite this version:

Mélanie Blazère. A variant of the spectral clustering method using l1 -penalty to promote sparse eigenvectors basis. 2016. hal-01294831

HAL Id: hal-01294831

https://hal.science/hal-01294831v1

Preprint submitted on 12 Apr 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A variant of the spectral clustering method using ℓ_1-penalty to promote sparse eigenvectors basis

Mélanie Blazère

Abstract

The spectral clustering method is one of the most well-known graph clustering algorithm, that is used in a wide range of fields and applications. In this paper, we propose a variant of the spectral clustering algorithm, called ℓ_1-spectral clustering. This procedure does not require the use of k-means to cluster the nodes of the graph, but directly estimates the indicators of the communities by computing a specific eigenbasis using ℓ_1 penalty.

Keywords

Spectral clustering, community detection, eigenvectors basis, ℓ_1-penalty.

1 Introduction

Since many systems of various kinds can be represented as networks, graphs play a central role in complex system [HG10],[Str01],[New03],[AB02]. Fields of application are many and various, ranging from mathematics (graph theory, combinatoric problems) to physics (systems of particles [Hop82]), sociology (social networks [HG10]), marketing (consumers preferences graphs), informatics (decision trees, combinatorial optimization, World Wild Web [PSV07]) or biology (neural, proteins networks, genes [JTA+00]). In biology, there exists a lot of applications [MLF+09], [YH07],[GA05],[ZH+05],[DHJ+04],[MDJL04], just to name a few.

Community detection is an important task for complex networks, that can help to understand the functional modules on the whole network. Community structures are believed to play an important role in the functioning of the complex systems modeled by graphs, so that detecting theses structures is of the highest importance. There exists different ways of defining community structures. The heuristic one corresponds to groups of nodes that are densely connected with sparse connections in between [GN02],[NG04]. Among different community detection methods based on graph structures, the spectral clustering method is currently one of the most popular and has been widely used in various fields. This method uses the eigenvectors of adjacency type matrices to cluster the nodes of a graph into a given number of communities. The nodes are not directly clustered but k-means is applied to the eigenvectors to detect the communities. If this method is so popular, it is mainly because spectral clustering is very easy to implement and easy to use. The computations are very fast and efficient, even for very large graphs. However, there is no guarantee to reach the best or most natural partitioning in general cases.

The purpose of this work is to present an alternative method to the classical spectral clustering for a specific random graph model, called ℓ_1-spectral clustering. This is an alternative method in the sense that the partitioning of the graph is essentially based on the computation of the singular value decomposition of the adjacency matrix (or normalized

adjacency matrix). The associated algorithm actually aims at finding a sparse basis of the eigenvectors of the adjacency matrix, starting from an initial basis that is not necessarily sparse but can be computed very quickly by any eigensolvers. This procedure turns out to solve an optimization problem of the form

$$\underset{v \in \mathcal{S}}{\operatorname{argmin}} \parallel v \parallel_1,$$

where $\mathcal{S}$ is a subset in $\in \mathbb{R}^{n-1}$ based on some transformations of the initial eigenvectors of the adjacency matrix. The remainder of this paper is organized as follows. In Section 2, we introduce the notations. Section 3 is devoted to the presentation of the model we suggest to work on, i.e. a random graph model that is closely related to a stochastic block model. We actually assume that the observed graph results from a deterministic graph with an exact community structure, whose edges have been disturbed by Bernoulli variables. In Section 4, we recall the basic principles of the spectral clustering method. Then, we present a new method to estimate block membership of nodes, from the observation of the noisy graph. The performance of this procedure is strongly dependent on the perturbation of the adjacency matrix and more specifically on the Frobenius norm of the noise matrix. Some results on the expectation of the Frobenius norm of the noisy matrix (associated to the random model we consider) are given in Section 5. Finally, we study the performances of the suggested method on simulated data. Results are presented in Section 6. Experimental results indicate that this algorithm works well on simulated datasets and is effective at finding the good communities.

2 Graph notation

A graph G refers to a set of vertices and a set of edges. The nodes represent the individuals or objects and the edges the interactions and relationships between them. From a mathematical point of view, a graph G is a pair (V, E) where V is the set of vertices and E refers to the set of edges that pairwise connect the vertices [Die05]. An edge $e \in E$ that connects a node i and a node j is denoted by $e = (i, j)$.

An important object associated to the graph is the adjacency matrix $A = (A_{ij})_{(i,j)\in V^2}$. The element A_{ij} represents the weight on the edge between node i and node j. For unweighted graphs, the adjacency matrix is defined by

$$A_{ij} = \begin{cases} 1 & \text{if there is an edge between } i \text{ and } j \\ 0 & \text{otherwise} \end{cases}.$$

If the graph contains n nodes i.e. $\mid V \mid = n$, then $A \in \mathbb{M}_n(\mathbb{R})$. When the graph is undirected, A is a symmetric matrix. Its diagonal elements are equal to zero when there is no loop.

The *degree* d_i of a node i is equal to the number of edges incident to i

$$d_i = \sum_{j=1}^{n} A_{ij}.$$

The matrix D, called the *degree matrix*, contains the degree of the nodes $d_1, ..., d_n$ on the diagonal and zero anywhere else

$$D = \begin{pmatrix} d_1 & & 0 \\ & \ddots & \\ 0 & & d_n \end{pmatrix}.$$

Given a subset of vertices $C \subset V$, we denote by $\overline{C}$ its complement, that is $V \setminus C$. We define the indicator vector $\mathbf{1}_C \in \mathbb{R}^n$ as the vector whose entries are defined by

$$(\mathbf{1}_C)_i = \begin{cases} 1 & \text{if vertex } i \text{ belongs to } C \\ 0 & \text{otherwise} \end{cases} .$$

A subset $C \subset V$ of a graph is said to be *connected* if any two vertices in C are connected by a path in C (sequence of vertices in C connected by edges that joined the two initial vertices). In addition, C is called a *connected component* if there are no connections between vertices in C and $\overline{C}$. Non empty sets $C_1, .., C_k$ form a *partition* of the graph if $C_i \cap C_j = \emptyset$ and $C_1 \cup ... \cup C_k = V$.

3 Presentation of the model

There is no formal definition of what should be a community. But, there exists a consensus on the fact that it should refer to groups of vertices highly connected with sparse connections in between.

In this section, we assume that the observed graph, denoted by $\hat{G}$, is actually a perturbed version of a graph G that has k fully connected components (called the communities). We actually assume that the observed graph $\hat{G}$ results from a deterministic graph with an exact community structure, whose edges have been perturbed by Bernoulli variables. We consider only undirected graph with no loops and we assume that the observed graph is connected (otherwise, it would be equivalent to work on each of the connected components of the graph).

Hereafter, observed quantities will be denoted by a hat.

3.1 The ideal graph

The ideal graph G is assumed to be the union of k complete graphs that are disconnected from each other. The number of vertices is n. The vertices are labelled from 1 to n. We allow the number of vertices in each subgraph to be different. We denote by $s_1, ..., s_k$ (≥ 2) their respective size (of course $\sum_{i=1}^{k} s_i = n$). To simplify and without loss of generality, we assume that the nodes $\{1, ..., n\}$ are ordered with respect to their block membership, in increasing order with respect to the size of the blocks. In other words, the nodes in the smallest cluster are denoted by $1, ..., s1$ and so on.

From a matricial point of view, the associated adjacency matrix A is block diagonal and has the following form

$$A = \underbrace{\begin{bmatrix} A^{11} & A^{12} & \dots & A^{1k} \\ A^{12} & A^{22} & \dots & A^{2k} \\ \vdots & \vdots & \vdots & \vdots \\ A^{1k} & A^{2k} & \dots & A^{kk} \end{bmatrix}}_{n} \in \mathbb{S}_n(\mathbb{R})$$

where

$$A^{ij} := 0_{s_i, s_j} \quad 1 \leq i \neq j \leq k$$

and

$$A^{ii} := N_{s_i} - I_{s_i} = \underbrace{\begin{bmatrix} 0 & 1 & \dots & 1 \\ 1 & 0 & \dots & 1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & \dots & 0 \end{bmatrix}}_{s_i} \in \mathbb{S}_{s_i}(\mathbb{R}), \quad 1 \leq i \leq k.$$

I_{s_i} denote the identity matrix of size s_i and N_{s_i} the square matrix of same size s_i, whose entries are all equal to one.

In other words,

$$A = \underbrace{\begin{bmatrix} \underbrace{\begin{bmatrix} 0 & 1 & \dots & 1 \\ 1 & 0 & \dots & 1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & \dots & 0 \end{bmatrix}}_{s_1} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \underbrace{\begin{bmatrix} 0 & 1 & \dots & 1 \\ 1 & 0 & \dots & 1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & \dots & 0 \end{bmatrix}}_{s_k} \end{bmatrix}}_{n}. \tag{1}$$

3.2 Perturbed version $\hat{G}$ of the graph G

We assume that we do not have access to this graph G. This is the case, for instance, if the graph has been inferred by estimating the dependencies between the vertices. We just observe a perturbed version of G denoted by $\hat{G}$ (some edges have been added between the communities and others have been removed within the communities independently with respect to a given probability). For instance, this perturbation may arise because of a partial knowledge of the graphs. This can be due to errors of measurement or to estimation noise (if the graph is estimated).

Let us detail what we exactly mean here by perturbations. Let $\hat{A}$ be the adjacency matrix associated to $\hat{G}$. We assume that

$$\hat{A} = A \overset{2}{\oplus} B \tag{2}$$

where

- $\widehat{A}_{ij} = \left\{ A \overset{2}{\oplus} B \right\}_{ij} = A_{ij} + B_{ij} \pmod 2.$
- B is a symmetric matrix of size n, whose upper entries are realizations of independent Bernoulli variables
$$\begin{cases} B_{ij} \sim \mathcal{B}(p) \text{ i.i.d., } i < j \\ B_{ii} = 0 \\ B_{ij} = B_{ji} \end{cases}.$$

B is therefore the adjacency matrix of an Erdos-Renyi graph $G(n,p)$. We recall that an Erdos-Renyi graph $G(n,p)$ has n vertices. The entries of its adjacency matrix are independent (up to the constraint that the adjacency matrix is symmetric) and equal 1 with probability p and 0 with probability $1-p$ (except on the diagonal where the entries are deterministic and always equal to 0). A value of B_{ij} equal to one means that the edge between nodes i and j is removed if the edge exists, or conversely that an edge is added if there was no edge before. To sum up

- If $B_{ij} = 1$ and $A_{ij} = 1$, we remove an edge between vertex i and j in G.
- If $B_{ij} = 1$ and $A_{ij} = 0$, we add an edge between vertex i and j in G.
- If $B_{ij} = 0$ we do not change any thing on the initial graph.

The noise is independent from one edge to the other, so that a change appears with the same probability for all the edges. An edge is removed in a community with probability p and added between two communities with the same probability. So that, two nodes are connected with probability $1-p$ if they belong to the same community and with probability p when they do not belong to the same community. We can easily generalize this model by considering different perturbation probabilities within and between each of the blocks. To simplify, in what follows, we keep the same value of p for all of the entries.

The difference between purely random graphs and our model is that we do not assume that a group structure emerges by chance but because there exists an underlying structure. This structure can be based on a real physical or biological mechanism that have been significantly changed. Hence, the model is random but the underlying group structure is considered as fixed and is well defined. This is a kind of probabilistic-statistical model that is well suited to model graphs that have been inferred from observations.

The model we consider is similar to the stochastic block-model, in the sense that the probability of an edge between two nodes depends only on the communities membership. But here, the membership of a node to a community is assumed to be fixed and not randomly assigned as in the stochastic block model. In the stochastic block-model, we have

$$\mathbb{P}\left[A \mid \tau\right] = \prod_{\substack{(i,j)\in V^2 \\ i\neq j}} \mathbb{P}\left[A_{ij} \mid \tau(i), \tau(j)\right] = \prod_{\substack{(i,j)\in V^2 \\ i\neq j}} \left(1 - P_{\tau(i),\tau(j)}\right)^{1-A_{ij}},$$

where τ is the random block membership function. This means that, conditionally to τ, the entries A_{ij} of the adjacency matrix are independent Bernoulli random variables with parameter given by $P_{\tau(i),\tau(j)}$ (that depends on the community membership of the nodes i and j). The equivalent quantity of p_i in a stochastic block model is somehow the nodes density of the communities. In our setting, there is no conditioning. The sizes of the communities are assumed to be fixed and equal to $(s_i)_{1\leq i\leq k}$. In addition, the number of nodes is kept fixed. But, it is the probability of an edge between two nodes that is supposed to vary. We implicitly assume that this probability p depends on a parameter and tends to zero when this parameter tends to infinity. For instance, we can imagine that the nodes of the graph represent features whose interactions have been estimated based on m observations, so that $p \underset{m\to+\infty}{\longrightarrow} 0$.

Because the graph and the adjacency matrix are equivalent, in what follows, we forget the graph and we only focus on the adjacency matrix.

3.3 Notations

Let us now detail some of the notations we will need later.

For $l \in \{1, ..., k\}$, let

$$\mathcal{M}_l = \left\{ i \in [1, n] : \sum_{r=0}^{l-1} s_r + 1 \leq i \leq \sum_{r=0}^{l} s_r \right\}.$$

$\mathcal{M}_l$ represents the indices of the nodes in community l. $\{\mathcal{M}_1, ..., \mathcal{M}_k\}$ is a partition of $[1, n]$, so that $[1, n] = \bigcup_{1 \leq l \leq k} \mathcal{M}_l$. We also define

$$\mathcal{M}_l^c = [1, n] \setminus \mathcal{M}_l = \bigcup_{\substack{1 \leq m \leq k \\ m \neq l}} \mathcal{M}_m.$$

For $l \in \{1, ..., k\}$ and $m \in \{1, ..., k\}$, we define

$$\mathcal{J}_{lm} = \mathcal{M}_l \otimes \mathcal{M}_m = \left\{ (i, j) \in [1, n]^2 : i \in \mathcal{M}_l, j \in \mathcal{M}_m \right\}.$$

$\{\mathcal{J}_{lm}\}_{1 \leq l \leq m \leq k}$ is a partition of $[1, n]^2$, so that $[1, n]^2 = \bigcup_{1 \leq l, m \leq k} \mathcal{J}_{lm}$. We also define

$$\mathcal{J}_{ll}{}^{-} = \{(i, j) \in \mathcal{J}_{ll},\ i \neq j\}.$$

For any adjacency matrix F, we denote by D_F the degree matrix associated to F. Its diagonal entries are denoted by d_i^F, where we recall that $d_i^F = \sum_{j=1}^n F_{ij}$. The ideal adjacency matrix A given by Equation (1) has its degrees equal to

$$d_i^A := \sum_{j=1}^n A_{ij} = s_l - 1, \quad \text{if} \quad i \in \mathcal{M}_l.$$

For $l = 1, ..., k$, let $d_l := s_l - 1$. We have

$$D_A = \text{diag}(\underbrace{d_1, \ldots, d_1}_{s_1}, ... \underbrace{d_k, ..., d_k}_{s_k}).$$

4 A new graph community detection procedure: l_1-spectral clustering,

In this section, we suggest an alternative to spectral clustering to find the k underlying communities of a graph G from the observation of the noisy graph $\hat{G}$. This new algorithm is based on the research of a "sparse" eigenvectors basis through l_1 minimization. We denote by $C_1, ..., C_k$ the k connected components of G, that match the k communities. We recall that these connected components are sorted in increasing order of size, with size equal to $s_1, ..., s_k$.

4.1 Assumption

In addition to the number of communities, we assume that we know one representative for each of the community. By a representative, we mean a node belonging to this community. This assumption is not so restrictive compared to traditional spectral clustering where the number of communities is assumed to be known. In practice, the number of communities can be inferred from the structure of the graph or from the knowledge of the biological or physical process that drives the graphs. For instance, in genes networks, the groups of genes we search for often represent metabolism functions. These functions are

a priori known and some genes implied in these metabolic functions have already been evidences by experiments. The aim is to cluster genes around these initial well-known ones to detect new genes and gain thereby new understanding of the complex system. If we do not exactly know a representative for each group, we can estimate them by first applying a rough partitioning algorithm or just an algorithm that aims at finding the hub of very densely connected parts of the graph.

We denote by $i_1, ..., i_k$ the indices of these initial elements.

4.2 Background on spectral clustering

One of the most commonly used method to cluster a graph into communities is the spectral clustering method [VL07]. The algorithm behind spectral clustering has been rediscovered and reapplied in various and different fields, since the initial work of Fiedler [FIE]. For a detailed history of spectral clustering we refer to [ST07] and to [VLBB08].

The idea behind spectral clustering is to use eigenvectors of matrices, based on the adjacency matrix, to cluster the graph. In the spectral clustering procedure, ones use the first k eigenvectors of a normalized or unormalized version of the Laplacian matrix (derived from the adjacency one) to cluster the nodes of the graph.

Let $G = (V, E)$ a graph whose adjacency matrix is denoted by A. There are mainly three matrices whose spectral properties can be studied to discover structures in a graph [SR95], [VL07]. These matrices are listed below and essentially depend on the adjacency matrix A.

1. The **Laplacian** matrix
$$L = D - A.$$

2. The **symmetric Laplacian** matrix
$$L_{sym} = D^{-1/2} A D^{-1/2},$$
denoted by L_{sym} because it is a symmetric version of the Laplacian matrix.

3. The **random walk Laplacian** matrix
$$L_{rw} = D^{-1} L = I - D^{-1} A,$$
denoted by L_{rw} because $D^{-1}A$ may represent the transition matrix of a random walk.

The L matrix is often referred as the unormalized Laplacian and L_{sym}, L_{rw} as the normalized Laplacian matrices. To implement spectral clustering one has to compute the first k eigenvectors (those corresponding to the k smallest eigenvalues) of one of the Laplacian matrix above. This step allows to get a representation of the of the initial data into a low-dimensional space in order to be clustered, by k-means for instance. We refer to [VL07] for details.

If these matrices are so important in graph clustering, it is because, as explained in Proposition 4.1 below, the distribution of its eigenvalues indicates the number of connected components in the graphs. In addition, its eigenvectors fully describe the vertices that lie in each of these connected components.

Proposition 4.1. *[VL07]*

The multiplicity k of the 0 eigenvalue of L and L_{rw} and the multiplicity k of the generalized 0 eigenvalue of L_{sym} are equal to the number of connected components $C_1, ..., C_k$ in the graph.

For L and L_{rw}, the eigenspace associated to 0 is spanned by the indicator vectors $\{\mathbf{1}_{C_i}\}_{1\leq i\leq n}$.

For L_{sym}, the eigenspace associated 0 is spanned by $\left\{D^{1/2}\mathbf{1}_{C_i}\right\}_{1\leq i\leq n}$.

If the Laplacian (or one of its equivalent) is so appealing, it is because the multiplicity of the null eigenvalue (that corresponds to the smallest eigenvalue) is equal to the number of connected components. In addition, a particular basis of the associated eigenspace is spanned by the community indicators. Therefore, if the graph is made of exactly k connected components, the computation of the eigenvectors of L,L_{sym} and L_{rw} enables to recover these components. When these components are sufficiently connected, they represent communities. In practice, the graph is not made of connected components, but of densely connected subgraphs that are sparsely connected to each other. These densely connected subgraphs represent somehow a perturbed version of the initial connected components that form the communities. If the perturbation is not too high, we can still hope that the eigenvectors of the perturbed Laplacian matrix (associated to this perturbed graph) still contain enough information on the graph structure to detect these communities. In particular, for one specific eigenvectors basis, the vectors coefficients are likely to be very close for indices corresponding to nodes in the same community. Therefore, it is natural to then apply k-means to the rows of the matrix containing these eigenvectors in columns.

The way the eigenvectors basis of the adjacency (or Laplacian matrix) is built is of the highest importance to ensure a good recovery of the communities. To better understand where the idea of this new method comes from, we go back to the ideal case where the different blocks are not connected. In this case, the indicators of the communities $\{\mathbf{1}_{C_i}\}_{1\leq i\leq k}$ are the eigenvectors of the normalized and unormalized adjacency or Laplacian matrix. Let U be the concatenation of these vectors. In this situation, we easily see that rows corresponding to indices of nodes in the same class are equal. Hence, clustering the rows of U provides, by the same way, the knowledge of the blocks. Of course, if the adjacency matrix or the Laplacian matrix are perturbed by an additive noise, then the eigenvectors will also be modified. But, if the perturbation is small, we can still hope that the rows (of the modified version of U) will remain close enough, so that k-means on these rows find the good clusters. However, since 0 is a repeated eigenvalue, the eigenspace associated to 0 is spanned by $\{\mathbf{1}_{C_i}\}_{1\leq i\leq k}$. Hence, there is no guarantee that the implementation of the eigensolvers provides the community indicators as eigenvectors. They can be replaced by a linear combination of the community indicators. For instance, the eigensolvers can find the sum of the indicators as an eigenvector. In this case all the coefficients of the vector are equal to one. Therefore, they are of no use for clustering the nodes of the graph. Once we add a perturbation on the adjacency matrix, the eigenvalues are not multiple anymore and the eigenspace becomes of dimension one. But, the eigenvalues can remain very close and if the eigengap is small the first top eigenvectors given by an eigensolvers may no be very useful to well cluster the groups. The first eigenvectors are not equally informative. This may explain why the original spectral clustering method can fail to recover the good communities in some case. The choice of the k eigenvectors is in fact of the highest importance. The key is to select relevant eigenvectors that provide useful information about the natural grouping of the data. In what follows, we do not use directly the subspace spanned by the first eigenvectors to find the communities but we first compute another eigenbasis that promotes sparse solutions for the eigenvectors.

4.3 A new approach

The alternative method we suggest is based on the same principle as spectral clustering. We still focus on the space spanned by the k first eigenvectors. But, instead of computing and directly using one basis among others, we wisely compute from this initial basis (that has been fastly computed by any eigensolver) a basis better suited for clustering.

We do not work on the Laplacian or normalized Laplacian but directly on the adjacency matrix. However, the idea remains the same if we replace the adjacency matrix by the Laplacian or its normalized version by the normalized Laplacian matrix.

The community indicator are the eigenvectors associated to the largest eigenvalues: the eigenvalues of A associated to $\{\mathbf{1}_{C_i}\}_{1\leq i\leq k}$ are equal to $d_1, ..., d_k$ and the other eigenvalues are equal to -1. We assume that the eigenvalues of A denoted by $\lambda_1, ..., \lambda_n$ are sorted in decreasing order. Let $u_1, .., u_n$ be the associated normalized eigenvectors given by any eigensolvers, so that the k first eigenvectors of A (associated to the k largest eigenvalues) are denoted by $u_1, ..., u_k$. We denote by U_k the matrix that contains $u_1, ...u_k$ in columns and by V_k the one that contains $u_{k+1}, ..., u_n$. We define

$$\mathcal{U}_k = \text{Span}\left\{u_1, ..., u_k\right\}.$$

4.3.1 Community detection in the ideal case

Proposition 4.2 and Proposition 4.3 below show that the community indicators are solution of some specific minimization problem.

Proposition 4.2. *The minimization problem*

$$\underset{v\in\mathcal{U}_k\setminus\{0\}}{\textit{argmin}} \parallel v \parallel_0$$

has a unique solution (up to a constant) given by $\mathbf{1}_{C_1}$.

We recall that $\parallel v \parallel_0 = \mid \{j : v_j \neq 0\} \mid$. In other words, $\mathbf{1}_{C_1}$ is the sparsest non-zero eigenvector in the space spanned by the first k eigenvectors.

Proof. Since $\{\mathbf{1}_{C_i}\}_{1\leq i\leq k}$ is a basis of $\mathcal{U}_k$, a vector v in $\mathcal{U}_k$ can be written as $v = \sum_{j=1}^{k} \alpha_j \mathbf{1}_{C_j}$ where $(\alpha_1, ..., \alpha_k) \in \mathbb{R}^k \setminus \{0\}$. Therefore, we deduce that $\parallel v \parallel_0$ is equal to $\mathbf{1}_{\alpha_1\neq 0}s_1 + ... + \mathbf{1}_{\alpha_1\neq 0}s_k$. Because at least one of the α_i is non zero and $s_1 \leq s_2 \leq ... \leq s_k$, the vector in $\mathcal{U}_k$ with the smallest l_0 norm is the one for which $\alpha_1 \neq 0$ and $\alpha_i = 0$ for all $i \neq 1$. □

We can generalized Proposition 4.2 to find the other indicators of the communities. For $i = 2, ..., k$, let $\mathcal{U}_k^i = \{v \in \mathcal{U}_k : v \perp \mathbf{1}_{C_l},\ l = 1, ..., i-1\}$.

Proposition 4.3. *For* $i = 2, ..., k$, *the minimization problem*

$$\underset{v\in\mathcal{U}_k^i\setminus\{0\}}{\textit{argmin}} \parallel v \parallel_0$$

has a unique solution (up to a constant) given by $\mathbf{1}_{C_i}$.

Notice that the constraints are linear. However, because of the l_0-norm this minimization problem is NP-hard. Fortunately, based on the knowledge of one representative for each group, we can replace the l_0-norm by its convex relaxation given by the l_1-norm.

Recall that $i_1, ..., i_k$ are the indices of these representative elements and let

$$\tilde{\mathcal{U}_k} = \{v \in \mathcal{U}_k : v_{i_1} = 1\}.$$

This is straightforward to see that the community indicator of the smallest community is solution of the following optimization problem.

Proposition 4.4. *The minimization problem* $(\mathcal{P}_1)$

$$\underset{v \in \tilde{\mathcal{U}}_k}{argmin} \parallel v \parallel_1$$

has a unique solution given by $\mathbf{1}_{C_1}$.

Proof. Since $\{\mathbf{1}_{C_i}\}_{1 \leq i \leq k}$ is also a basis of $\mathcal{U}_k$, a vector v in $\mathcal{U}_k$ can be written as $v = \sum_{j=1}^{k} \alpha_j \mathbf{1}_{C_i}$ where $(1, ..., \alpha_k) \in \mathbb{R}^k$. We deduce that $\parallel v \parallel_1$ is equal to $\alpha_1 s_1 + ... + \alpha_k s_k$. Because at least one of the α_i is not zero and $s_1 \leq s_2 \leq ... \leq s_k$, the vector in $\mathcal{U}_k$ that satisfies $\parallel v \parallel_\infty = 1$ and has the smallest l_1 norm is the one for which $\alpha_1 \neq 0$ and $\alpha_i = 0$ for all $i \neq 1$ and $\alpha_1 = 1$. □

To simplify and without loss of generality, we assume that i_1 corresponds to the first index (up to a permutation).

Corollary 4.5. *Problem* $(\mathcal{P}_1)$ *is equivalent to*

$$\underset{\substack{\tilde{v} \in \mathbb{R}^{n-1} \\ (1, \tilde{v}) \in \mathcal{U}_k}}{min} \parallel \tilde{v} \parallel_1 .$$

Because the columns of the matrix U form an orthonormal basis, $v \in \mathcal{U}_k$ is equivalent to $V_k^T v = 0$, where we recall thatV_k is the restriction of U to the last $n - k$ columns.

Let w be the first column of V_k^T. We define W as the matrix V_k^T whose first column w has been deleted.

Proposition 4.6. *The solution* v^* *of problem* $(\tilde{\mathcal{P}}_1)$ *is given by* $v^* = (1, \tilde{v}^*)$ *where*

$$\tilde{v}^* \in \underset{\substack{\tilde{v} \in \mathbb{R}^{n-1} \\ W\tilde{v} = -w}}{argmin} \parallel \tilde{v} \parallel_1 .$$

This solution v^* is equal to $\mathbf{1}_{C_1}$. There exists very efficient algorithms that solve this type of l_1-minimization problem with linear constraints. They have been proved to converge to the right solution and can be easily solved using R or Matlab. For instance, we can use the R Optimization Infrastructure (ROI) package or the l_1`-eq` function, from the Matlab optimization package l_1-Magic.

The other community indicators are computed in the same way, adding the constraints that the target vector is orthogonal to the previous computed vectors. In practice, we can deflate the matrix A. We get a matrix $\tilde{A}$, so that the indicator of the second smallest community corresponds now to the sparsest eigenvector associated to the space spanned by the k largest eigenvalues of $\tilde{A}$. Hence, the problem is traced back to the first one and so on.

4.3.2 Minimization problem under perturbations

In practice, we do not have access to this unperturbed graph. So, the question is what if the adjacency is perturbed as in Equation (2) ? We denote by $\hat{U}_k$ the perturbed version of U_k. This matrix contains the first k eigenvectors of the adjacency matrix $\hat{A}$ associated to the observed graph $\hat{G}$. $\hat{V}_k$ denotes the matrix containing the others $n - k$ eigenvectors. To simplify, we just present the solution to find the first community indicators (for the others, the idea remains the same except that we add the constraint of being orthogonal to the previous solution vectors). The solution consists in releasing the equality constraints

of Problem $(\tilde{\mathcal{P}}_1)$ given in Proposition 4.6. This is equivalent to solve the minimization problem $(\hat{\mathcal{P}}_1)$

$$\underset{\tilde{v}\in\mathbb{R}^{n-1}}{\operatorname{argmin}} \parallel \hat{W}\tilde{v} + \hat{w} \parallel_2^2 + \lambda \parallel \tilde{v} \parallel_1, \qquad (\hat{\mathcal{P}}_1)$$

where $\lambda > 0$ is the penalty parameter, $\hat{W} \in \mathbb{M}_{n-k,n-1}$ is the matrix $\hat{V}_k^T$ whose first column $\hat{w}$ has been deleted. The term $\parallel \hat{W}\tilde{v} + \hat{w} \parallel_2^2$ means that we search for a vector close to the space spanned by the first k eigenvectors of $\hat{A}$ and whose first coefficient is equal to one. $\parallel \tilde{v} \parallel_1$ is what controls the sparsity of the solution. As for all regularizing methods depending on a parameter, the main issue is the choice of λ. Figure 1 below shows that the behaviour of the estimated coefficients is of two kinds. The simulations have been performed on a model with nine communities with size between 10 and 30 and a perturbation equal to 0.1. There is a clear splitting in the behaviour of the coefficients, depending if they belong to the first community or not. By looking in more details to the data, we see that the upper batch of curves are associated to nodes in the first community and the lower batch of curves to ones in the other communities. This is true whatever is the value of the penalty parameter. Figure 1 below does not represent a particular situation but a typical behaviour of the Lasso when applied to our model (we have implemented models with various parameters).

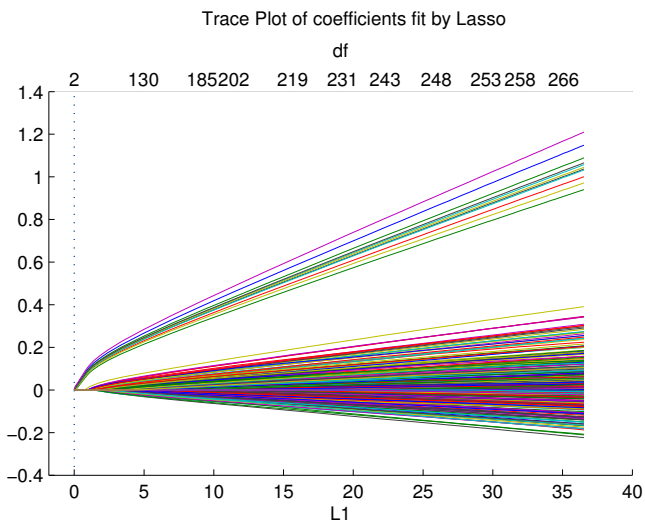
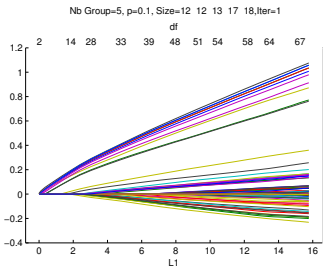
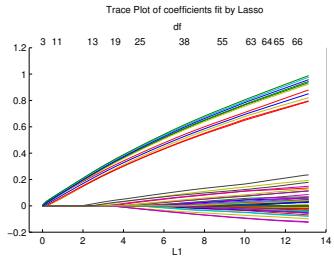


Figure 1: Lasso path for the first community indicators estimates

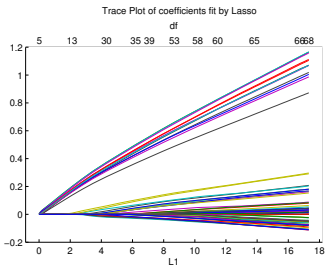
Figure 2 represents the coefficients Lasso path in the estimation of the community indicators. The graph has five communities, of size between 20 and 50, that have been perturbed with a value of p equal to 0.2. Here again, we retrieve this particular behaviour of the estimated coefficients. Of course this splitting vanished as p increases.



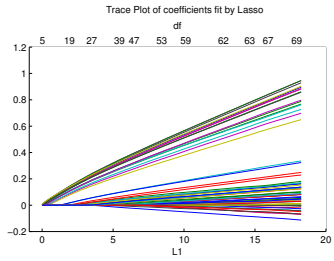
First vector



Second vector



Third vector



Fourth vector

Figure 2: Illustration of the Lasso path for the four community indicators estimates

This peculiar behaviour can be explained by the fact that the ideal target parameter has coefficients equal to zero or one and does not take continuous values. In addition, the fact that one of the coefficients in the true community is already equal to one forces the other coefficients in the same community to be close to one too, under small perturbations. So that we can hope to discriminate the membership of the nodes by keeping an equality constraint and then hardly thresholding the coefficients with respect to one-half.

Finally, the first community indicators $\mathbf{1}_{C_1}$ is estimated as follows.

1. Compute
$$\tilde{v} \in \underset{\substack{u \in \mathbb{R}^{n-1} \\ \hat{W} v = -\hat{w}}}{\operatorname{argmin}} \| v \|_1 .$$

2. Compute $\hat{v}_1 = (1, \tilde{v})$.

For the other vectors $((\hat{v}_i)_{2 \leq i \leq k})$, we add the constraint of being orthogonal to the previous computed vectors and we do the same to estimate the other community indicators. Then, for all $i = 1, ..., k$, $\mathbf{1}_{C_i}$ is estimated by $\hat{\mathbf{1}}_{C_i}$ where

$$\left(\hat{\mathbf{1}}_{C_i}\right)_j := \begin{cases} 1 & \text{if} \quad (\hat{v}_i)_j > \frac{1}{2} \text{ and } (\hat{v}_l)_j \neq 1, \ l = 1, ..., i-1 \\ 0 & \text{if} \quad\quad\quad\quad\quad\quad (\hat{v}_i)_j \leq \frac{1}{2} \end{cases} .$$

4.4 Algorithm

We detail below the algorithm we suggest to use to cluster nodes of a graph into k communities. This algorithm can be easily implemented using R or Matlab and is called the l_1-spectral clustering algorithm. The l_1-spectral clustering method sequentially build

Algorithm 1 l_1-spectral clustering algorithm

INPUT Adjacency matrix A of the observed graph $G = (E, V)$, number of communities k and the community representatives $c_1, ..., c_k$.
$F = []$.
For $i = 1, ..., k$
Step 1: Consider c_i the edge representative associated to the i-th community.
Step 2: Compute the eigendecomposition $[U, D]$ of A i.e. $A = UD^tU$.
Step 3: Sort the eigenvalues of A in increasing order and do the same for the associated eigenvectors.
Step 4: Compute the matrix R that contains the eigenvectors associated to the $n-k+i+1$ smallest eigenvalues
Step 5: Compute $V =^t R$.
Step 6: Compute $W = V^{-c_i}$ and $w = V^{c_i}$ (where $w = V^{c_i}$ is the c_i^{th} column of V and V^{-c_i} is the V matrix where we have removed the c_i^{th} column).
Step 7: Compute the solution s^i of the following problem

$$\min_{\substack{u \in \mathbb{R}^{n-1} \\ Wu = -w}} \| u \|_1 .$$

(using for instance the `l1eq` Matlab function).
Step 8: Form the solution $f_i = \left[s_1^i \ s_2^i ... s_{c_i-1}^i \ 1 \ s_{c_i}^i ... s_n^i\right]$.
Step 9: Compute $F = [F f_i]$ and deflate A with f_i.
End For.
Step 10: Do

$$\begin{cases} F_{ij} = 1 & if \quad \tilde{F}_{ij} \geq 1/2 \text{ and } F_{il} \neq 1, \ l = 1, ..., i-1 \\ \tilde{F}_{ij} = 0 & if \quad F_{ij} < 1/2 \end{cases}$$

OUTPUT k vectors $v^1, ..., v^k$ that are the columns of F ($v_j^i = 1$ means that the edge j belongs to community i).

k vectors representing the k community indicators. Algorithm 1 details the main steps of the procedure.

In algorithm 1, we can replace A by $D^{-1}A$ and the singular value decomposition of A by the computation of the generalized eigenelements of $D^{-1}A$.

Remarks:

1. One of the advantage of this algorithm is that it is computationally feasible, even for large graphs, thanks to efficient algorithms that solve l_1 minimization problems.

2. In practice, we may just hope to have at hand a representative for each of the k communities. However, we do not know which one belongs to the smallest community. If the number of community is not too large, we can still proceed as described above by reviewing the k possible choices and choosing the one for which the objective function is minimal. This requires $k!$ more step but if k is small this is not very time-consuming.

5 Frobenius norm of the perturbation

To sum up, finding communities in the ideal case turns out to be equivalent to solve

$$\min_{\substack{\tilde{v}\in\mathbb{R}^{n-1}\\ W\tilde{v}=-w}} \| \tilde{v} \|_1 \qquad (\mathcal{P})$$

After perturbation, the community indicators are estimated by solving

$$\min_{\substack{\tilde{v}\in\mathbb{R}^{n-1}\\ \hat{W}\tilde{v}=-\hat{w}}} \| \tilde{v} \|_1 \,. \qquad (\tilde{\mathcal{P}})$$

The objective function still remains the same. Only the constraints space has been perturbed. Let $E = A - \hat{A}$ be the perturbation applied to the initial adjacency matrix. A natural question is under which conditions on E (and thus on the parameter p of the Bernoulli matrix that represents the noise) can we expect that the solution of Problem $(\tilde{\mathcal{P}})$ remains closed to the one of Problem $(\mathcal{P})$ and what is the rate of convergence? A noise perturbation on A implies a perturbation on U and thereby on W and w (since all these objects depend on the eigenvalues of U). If a small level of noise on A leads to a small perturbation of the eigenvectors of the associated normalized adjacency matrix then there is a hope to recover the community structure. So the main issue is what is the stability of the eigenvectors to matrix perturbations? Matrix perturbation theory [SSJ90] indicates that it essentially depends on the Frobenius norm of E and on the eigengap. This is the Davis-Kahan theorem stated below.

Theorem 5.1. *[SSJ90] Let $A, E \in \mathbb{M}_n(\mathbb{R})$ be symmetric matrices and let $\| \,.\, \|_F$ be the Frobenius norm. Consider $\tilde{A} := A + E$ as a perturbed version of A. Let $S_1 \subset \mathbb{R}$. be an interval. Denote by $\sigma_{S_1}(A)$ the set of eigenvalues of A which are contained in S_1, and by V_1 the eigenspace corresponding to all those eigenvalues. Denote by $\sigma_{S_1}(\tilde{A})$ and by $\tilde{V_1}$ the analogous quantities for $\tilde{A}$. Define the distance between S_1 and the spectrum of A outside of S_1 as*

$$\delta = min\{| \lambda - s |;\ \lambda \text{ eigenvalue of } A, \lambda \notin S_1,\ s \in S_1\}\,.$$

Then the distance $d(V_1, \tilde{V_1}) := \| \sin\Theta(V_1, \tilde{V_1}) \|_F$ between the two subspaces V_1 and $\tilde{V_1}$ is bounded by

$$d(V_1, \tilde{V_1}) \leq \frac{\| E \|}{\delta}.$$

Here after, we assume without loss of generality that each node in the observed graph has a degree larger than one.

5.1 Expression of the error term

We recall that $A_{ii} = 0$ and $A_{ij}, i \neq j$ is defined by

$$\widehat{A}_{ij} = A_{ij} + B_{ij} \text{ (mod 2)} = \begin{cases} 1 - B_{ij} & \text{if} \quad (i,j) \in \bigcup_{1 \leq l \leq k} \mathcal{J}_{ll}^{-} \\ B_{ij} & \text{otherwise} \end{cases}.$$

Therefore, the degree matrix of $\widehat{A}$ is given by

$$D_{\widehat{A}} = \text{diag}\left(d_1^{\widehat{A}}, ..., d_n^{\widehat{A}}\right)$$

where, for all $l = 1, ..., k$ and $i \in \mathcal{M}_l$,

$$d_i^{\widehat{A}} = \sum_{j=1}^{n} \widehat{A}_{ij} = \sum_{\substack{j \in \mathcal{M}_l \\ j \neq i}} (1 - B_{ij}) + \sum_{j \in \mathcal{M}_l^c} B_{ij} = d_l - \sum_{\substack{j \in \mathcal{M}_l \\ j \neq i}} B_{ij} + \sum_{j \in \mathcal{M}_l^c} B_{ij}.$$

Proposition 5.2. *$\hat{A} = A + E$ where $E \in \mathbb{M}_n(\mathbb{R})$ is given by*

$$E_{ij} = A_{ij} + B_{ij} \text{ (mod 2)} = \begin{cases} -B_{ij} & \text{if} \quad (i,j) \in \bigcup_{1 \leq l \leq k} \mathcal{J}_{ll}^{-} \\ B_{ij} & \text{otherwise} \end{cases}$$

when $i \neq j$ and $E_{ii} = 0$.

Proposition below provides an expression of the difference between $\left(D_{\widehat{A}}\right)^{-1} \widehat{A}$ and $(D_A)^{-1} A$ in terms of the elements of A and B.

Proposition 5.3. *$\left(D_{\hat{A}}\right)^{-1} \hat{A} = (D_A)^{-1} A + E$ where the entries of $E \in \mathbb{M}_n(\mathbb{R})$ are given by*

$$\begin{cases} E_{ij} & = & \begin{cases} -\dfrac{1}{d_i^{A_\varepsilon} + d_l} \left[\frac{d_i^{A_\varepsilon}}{d_l} + B_{ij}\right] & \text{if} \quad (i,j) \in \mathcal{J}_{ll}^{-}, 1 \leq l \leq k, i \neq j \\ \dfrac{B_{ij}}{d_i^{A_\varepsilon} + d_l} & \text{if} \quad (i,j) \in \mathcal{J}_{lm},\ 1 \leq l \neq m \leq k \end{cases} \\ E_{ii} & = & 0 \end{cases}$$

where

- $d_l := d_i^A = \sum_{j=1}^{n} A_{ij}$, $i \in \mathcal{M}_l$.
- $d_i^{A_\varepsilon} := d_i^{\hat{A}-A} = -\sum_{\substack{j \in \mathcal{M}_l \\ j \neq i}} B_{ij} + \sum_{j \in \mathcal{M}_l^c} B_{ij} = d_i^B - 2 \sum_{\substack{j \in \mathcal{M}_l \\ j \neq i}} B_{ij}$.
- $d_i^B = \sum_{\substack{j=1 \\ j \neq i}}^{n} B_{ij}$.

Proof. See Subsection 5.3.

5.2 Frobenius norm of the error

Proposition 5.6 and Proposition 5.4 below gives the expression of the Frobenius norm of the unormalized and normalized adjacency matrix.

Proposition 5.4. *The Frobenius norm of E, that satisfies $\hat{A} = A + E$, is equal to*

$$\| E \|_F^2 = \sum_{\substack{j=1 \\ j \neq i}}^{n} B_{ij} = 2 \sum_{i=1}^{n} \sum_{j>i} B_{ij}.$$

This proposition is a straightforward consequence of the definition of B and of the fact that $B_{ij}^2 = B_{ij}$ and $B_{ij} = B_{ji}$.

Corollary 5.5.

$$\mathbb{E}\left[\| E \|_F^2\right] = n(n-1)p$$

and

$$\mathbb{E}\left[d\left(\mathcal{U}_k, \hat{\mathcal{U}}_k\right)\right] \leq \frac{n(n-1)p}{s_1},$$

where we recall that s_1 is the size of the smallest community.

In particular, if all the communities are of equal size, we get

$$\mathbb{E}\left[d\left(\mathcal{U}_k, \hat{\mathcal{U}}_k\right)\right] \leq (n-1)kp.$$

If the number of nodes is assumed to be fixed, we easily see that the distance between $\mathcal{U}_k$ and $\hat{\mathcal{U}}_k$ will tend to zero as p goes to zero.

Proposition 5.6. *The Frobenius norm of E, that satisfies $\left(D_{\widehat{A}}\right)^{-1} \widehat{A} = (D_A)^{-1} A + E$, is equal to*

$$\| E \|_F^2 = \sum_{l=1}^{k} \sum_{i \in \mathcal{M}_l} \left[\frac{d_i^B}{\left(d_i^{A_\varepsilon} + d_l\right) d_l} \right] = \sum_{l=1}^{k} \frac{1}{d_l} \left[\sum_{i \in \mathcal{M}_l} \frac{d_i^B}{d_l} \left(1 + \frac{d_i^{A_\varepsilon}}{d_l}\right)^{-1} \right],$$

where we recall that

- $d_l := d_i^A = \sum_{j=1}^{n} A_{ij}$, $i \in \mathcal{M}_l$.
- $d_i^{A_\varepsilon} := d_i^{\hat{A}-A} = -\sum_{\substack{j \in \mathcal{M}_l \\ j \neq i}} B_{ij} + \sum_{j \in \mathcal{M}_l^c} B_{ij} = d_i^B - 2 \sum_{\substack{j \in \mathcal{M}_l \\ j \neq i}} B_{ij}$.
- $d_i^B = \sum_{\substack{j=1 \\ j \neq i}}^{n} B_{ij}$.

Another expression of the Frobenius norm only in terms of $(B_{ij})_{1 \leq i \neq j \leq n}$ is given by

$$\| E \|_F^2 = \sum_{l=1}^{k} \frac{1}{d_l} \sum_{i \in \mathcal{M}_l} \sum_{\substack{j=1 \\ j \neq i}}^{n} \left[\frac{B_{ij}}{\sum_{\substack{j \in \mathcal{M}_l \\ j \neq i}} (1 - B_{ij}) + \sum_{j \in \mathcal{M}_l^c} B_{ij}} \right].$$

Because, it is not possible to exactly compute the expectation of the ratio $\dfrac{d_i^B}{d_i^{A_\varepsilon}+d_l}$, we just provide an approximation of the expectation. Let R and S be two discrete random variables where S has no mass at 0. Using a Taylor expansion of $f:(x,y)\mapsto \frac{x}{y}$, we have

$$\mathbb{E}\left(\frac{R}{S}\right)\approx\frac{\mathbb{E}\left(R\right)}{\mathbb{E}\left(S\right)}\quad (first\ order)$$

and

$$\mathbb{E}\left(\frac{R}{S}\right)\approx\frac{\mathbb{E}R}{\mathbb{E}S}-\frac{\mathrm{Cov}(R,S)}{\mathbb{E}^2S}+\frac{\mathrm{Var}(S)\mathbb{E}R}{\mathbb{E}^3S}\quad (second\ order).$$

We apply this result with $R=d_i^B$ and $S=d_i^{A_\varepsilon}+d_l$.

Corollary 5.7. *In a first order approximation, we have*

$$\mathbb{E}\left[\parallel E\parallel_F^2\right]\approx\sum_{l=1}^{k}\frac{s_l}{d_l}\frac{(n-1)p}{d_l(1-2p)+(n-1)p}\quad (first\ order)$$

and

$$\mathbb{E}\left[d\left(\mathcal{U}_k,\hat{\mathcal{U}}_k\right)\right]\lessapprox\frac{s_k-1}{s_k}\sum_{l=1}^{k}\frac{s_l}{d_l}\frac{(n-1)p}{d_l(1-2p)+(n-1)p}\quad (first\ order)$$

where we recall that s_k is the size of the largest community.

In particular, if all the communities are of equal size, we get

$$\mathbb{E}\left[d\left(\mathcal{U}_k,\hat{\mathcal{U}}_k\right)\right]\lessapprox\frac{pk^2(n-1)}{(n-k)(1-2p)+(n-1)kp}\quad (first\ order).$$

These results provide a first understanding of what theoretical results could be expected for the l_1-spectral clustering method and how the level of noise p impacts on the eigenspace associated to the community indicators.

5.3 Proof

5.3.1 Proof of Proposition 5.3

Proof. We have

$$\left(D_{\hat{A}}\right)^{-1}\hat{A}=\left(D_A+D_{A_\varepsilon}\right)^{-1}\left(A+A_\varepsilon\right),$$

where $A_\varepsilon=\hat{A}-A$ and

- $A_\varepsilon:=\hat{A}-A$. The entries of this matrix are equal to 1 when an edge is added and to -1 when an edge is removed. The other entries are 0. In other words,

$$(A_\varepsilon)_{ij}=\widehat{A}_{ij}-A_{ij}=\begin{cases}-B_{ij} & \text{if }(i,j)\in\bigcup\limits_{1\le l\le k}\mathcal{J}_{ll}^{-}\\ B_{ij} & \text{otherwise}\end{cases}.$$

- $D_{A_\varepsilon}:=D_{\hat{A}}-D_A$. This matrix is diagonal and its non zero entries denoted by $\left(d_i^{A_\varepsilon}\right)_{1\le i\le n}$ represent the difference of degree between the observed and the ideal graph, for each of the n vertices. We have, for all $l\in\{1,...,k\}$ and $i\in\mathcal{M}_l$,

$$d_i^{A_\varepsilon}=-\sum_{\substack{j\in\mathcal{M}_l\\ j\neq i}}B_{ij}+\sum_{j\in\mathcal{M}_l^c}B_{ij}.$$

To simplify the notations, let $F := (D_A + D_{A_\varepsilon})^{-1}$ and $C = D_A^{-1} - (D_A + D_{A_\varepsilon})^{-1}$. We have

$$D_{\hat{A}}^{-1}\hat{A} = D_A^{-1}A + E,$$

where

$$E := FA_\varepsilon - CA.$$

F is a diagonal matrix and its entries are equal to

$$f_i := F_{ii} = \frac{1}{d_i^{A_\varepsilon} + d_l},$$

for all $l = 1, ..., k$ and $i \in \mathcal{M}_l$.

Then, computing the inverse of the diagonal matrices $D_A + D_{A_\varepsilon}$ and D_A, we deduce that C

is also a diagonal matrix whose entries $(c_i)_{1\leq i\leq n}$ are equal to

$$c_i := \frac{d_i^{A_\varepsilon}}{d_l}\frac{1}{d_i^{A_\varepsilon} + d_l},$$

for all $l = 1, ..., k$ and $i \in \mathcal{M}_l$.

Thus, on the one hand,

$$(FA_\varepsilon)_{ij} = \begin{cases} -\frac{B_{ij}}{d_i^{A_\varepsilon}+d_l} & \text{if } (i,j) \in \mathcal{J}_{ll}^-,\ 1 \leq l \leq k \\ \frac{B_{ij}}{d_i^{A_\varepsilon}+d_l} & \text{if } (i,j) \in \mathcal{J}_{lm},\ 1 \leq l \neq m \leq k \end{cases}$$

and

$$(FA_\varepsilon)_{ii} = 0.$$

On the other hand, CA is a block diagonal matrix and its entries are defined by

$$(CA)_{ij} = \begin{cases} \frac{d_i^{A_\varepsilon}}{d_l}\frac{1}{d_i^{A_\varepsilon}+d_l} & \text{if } (i,j) \in \mathcal{J}_{ll}^-, 1 \leq l \leq k \\ 0 & \text{if } (i,j) \in \mathcal{J}_{lm},\ 1 \leq l \neq m \leq k \end{cases}$$

and

$$(CA)_{ii} = 0.$$

To conclude, for $i \neq j$

$$E_{ij} = \begin{cases} -\dfrac{1}{d_i^{A_\varepsilon} + d_l}\left[\frac{d_i^{A_\varepsilon}}{d_l} + B_{ij}\right] & \text{if } (i,j) \in \mathcal{I}_l, 1 \leq l \leq k \\ \dfrac{B_{ij}}{d_i^{A_\varepsilon} + d_l} & \text{if } (i,j) \in \mathcal{J}_{lm},\ 1 \leq l \neq m \leq k \end{cases}$$

and it is straightforward to see that

$$E_{ii} = 0.$$

□

5.3.2 Proof of Proposition 5.6

Proof.

$$\| E \|_F = \sum_{i=1}^{n}\sum_{j=1}^{n} E_{ij}^2$$

$$= \sum_{1\le l\neq m\le k}\sum_{(i,j)\in\mathcal{J}_{lm}} \frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}(B_{ij})^2 + \sum_{l=1}^{k}\sum_{(i,j)\in\mathcal{J}_{ll}^-}\left[\frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\left(\frac{d_i^{A_\varepsilon}}{d_l}+B_{ij}\right)^2\right].$$

$$= \sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l}\frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\sum_{\substack{m=1\\ m\neq l}}^{k}\sum_{j\in\mathcal{M}_m}(B_{ij})^2 + \sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l}\frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\sum_{\substack{j\in\mathcal{M}_l\\ j\neq i}}(B_{ij})^2$$

$$+\sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l}\sum_{\substack{j\in\mathcal{M}_l\\ j\neq i}}\frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\left(\frac{d_i^{A_\varepsilon}}{d_l}\right)^2 + \sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l}\sum_{\substack{j\in\mathcal{M}_l\\ j\neq i}}\frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\left(2\frac{d_i^{A_\varepsilon}}{d_l}B_{ij}\right).$$

On the one hand,

$$\sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l}\frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\sum_{\substack{m=1\\ m\neq l}}^{k}\sum_{j\in\mathcal{M}_m}(B_{ij})^2 + \sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l}\frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\sum_{\substack{j\in\mathcal{M}_l\\ j\neq i}}(B_{ij})^2$$

$$= \sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l}\frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\left[\sum_{\substack{m=1\\ m\neq l}}^{k}\sum_{j\in\mathcal{M}_m}(B_{ij})^2 + \sum_{\substack{j\in\mathcal{M}_l\\ j\neq i}}(B_{ij})^2\right]$$

$$= \sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l}\frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\left[\sum_{j\in\bigcup_{m=1}^{k}\mathcal{M}_m\setminus\{i\}}(B_{ij})^2\right]$$

$$= \sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l}\frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\left[\sum_{\substack{j=1\\ j\neq i}}^{n}(B_{ij})^2\right].$$

On the other hand,

$$\sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l}\sum_{\substack{j\in\mathcal{M}_l\\ j\neq i}}\frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\left(\frac{d_i^{A_\varepsilon}}{d_l}\right)^2 = \sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l}\frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\frac{\left(d_i^{A_\varepsilon}\right)^2}{d_l}$$

using that $\sum_{\substack{j\in\mathcal{M}_l\\ j\neq i}} 1 = d_l$

and

$$\sum_{l=1}^{k}\sum_{\substack{j\in\mathcal{M}_l\\ j\neq i}}\sum_{i\in\mathcal{M}_l}\frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\left(2\frac{d_i^{A_\varepsilon}}{d_l}B_{ij}\right)$$

$$= -\sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l}\frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\frac{\left(d_i^{A_\varepsilon}\right)^2}{d_l} + \sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l}\frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\frac{d_i^{A_\varepsilon}}{d_l}\sum_{\substack{j=1\\ j\neq i}}^{n}B_{ij}$$

based on the fact that

$$2\sum_{\substack{j\in\mathcal{M}_l\\ j\neq i}} B_{ij} = \sum_{\substack{j=1\\ j\neq i}}^{n} B_{ij} - d_i^{A_\varepsilon}.$$

To conclude

$$\begin{aligned}
\| E \|_F = & \sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l} \frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\left[\sum_{\substack{j=1\\ j\neq i}}^{n}(B_{ij})^2\right] \\
& + \sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l} \frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\frac{\left(d_i^{A_\varepsilon}\right)^2}{d_l} - \sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l} \frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\frac{\left(d_i^{A_\varepsilon}\right)^2}{d_l} \\
& + \sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l} \frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\frac{d_i^{A_\varepsilon}}{d_l}\sum_{\substack{j=1\\ j\neq i}}^{n} B_{ij} \\
= & \sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l} \frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\left[\sum_{\substack{j=1\\ j\neq i}}^{n}(B_{ij})^2\right] + \sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l} \frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\frac{d_i^{A_\varepsilon}}{d_l}\left[\sum_{\substack{j=1\\ j\neq i}}^{n} B_{ij}\right] \\
= & \sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l}\left[\frac{1}{\left(d_i^{A_\varepsilon}+d_l\right)^2}\left(\sum_{\substack{j=1\\ j\neq i}}^{n} B_{ij}\right)\left(1+\frac{d_i^{A_\varepsilon}}{d_l}\right)\right]
\end{aligned}$$

because $B_{ij}\in\{0,1\}$ so $B_{ij}^2 = B_{ij}$ and thus $\sum_{\substack{j=1\\ j\neq i}}^{n} B_{ij}^2 = \sum_{\substack{j=1\\ j\neq i}}^{n} B_{ij}$,

$$= \sum_{l=1}^{k}\sum_{i\in\mathcal{M}_l}\left[\frac{d_i^B}{\left(d_i^{A_\varepsilon}+d_l\right)d_l}\right].$$

□

6 Test of the new algorithm on simulated data

In Section 4, we have introduced a new algorithm (called l_1-spectral clustering) that aims at detecting community structures in complex graphs. This algorithm use spectral vector partitioning techniques to classify nodes. To illustrate the performances of the l_1-spectral clustering, we simulate random undirected graphs generated from the model presented in Section 3 to then apply the algorithm of Subsection 4.4.

Using Matlab, we first generate random graphs with an exact group structure. We choose different number of blocks and different sizes for the blocks. Then, we add a noise on the associated adjacency matrices. Once the matrix is disturbed, we have no block structure anymore. To recover this underlying block structure, we apply the l_1-spectral clustering algorithm. We refer to Figure 3 for a summary of these different steps.

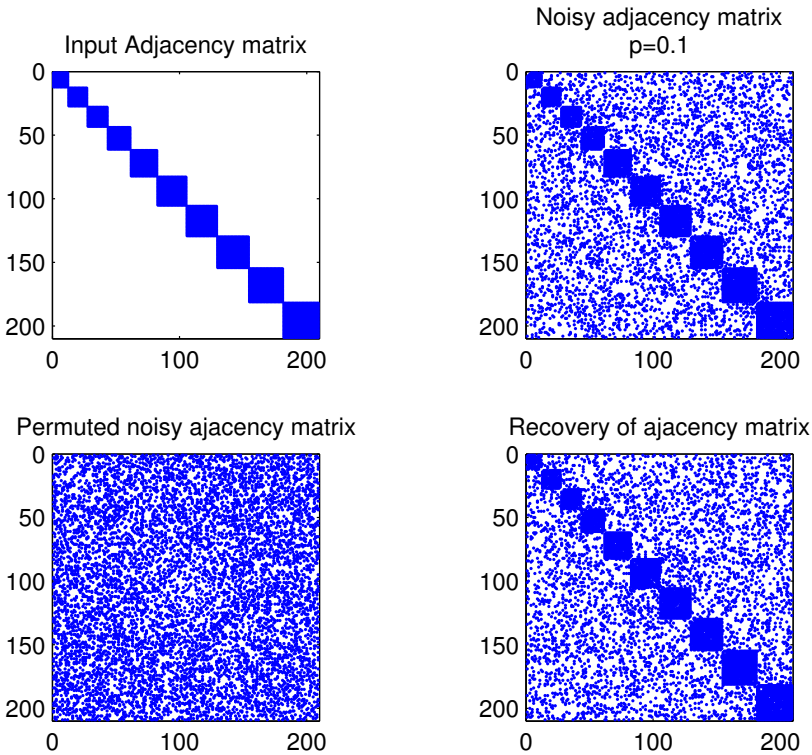


Figure 3: Recovery of the block structure

Figure 4 below gives an example of the distribution of the different eigenvectors that are involved in spectral clustering methods for a graph with nine communities, whose sizes have been randomly chosen between 20 and 50 and for a value of the perturbation equal to $p = 0.2$. Subfigure a) represents the histogram of the community indicators, b) the histogram of the eigenvectors as given by the Matlab eigensolver (these eigenvectors are the ones used to cluster the data in the traditional spectral clustering method), c) the histogram of the estimated community indicators using l_1-spectral clustering, d) same thing but applying the traditional spectral clustering method with k-means. In this specific case, the l_1 estimated eigenvectors exactly fit the true ones, whereas k-means makes a few mistakes.

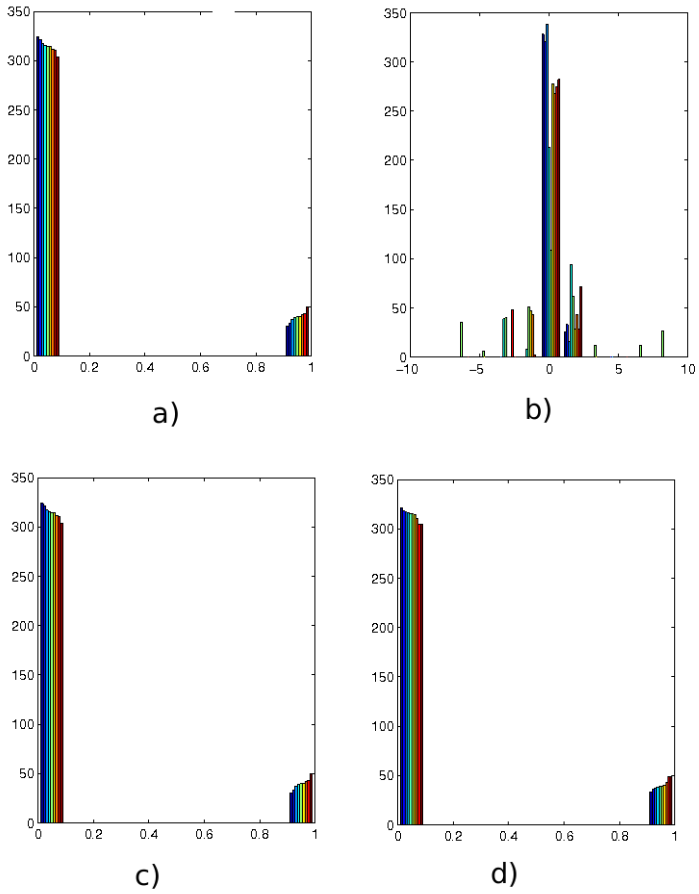


Figure 4: Histogram of a) the true community indicators, b) the eigenvectors given by SVD b) the estimated ones by l_1 spectral clustering, c) the estimated ones by k-means

Then, we test the robustness to perturbations and the performance of the algorithm. To do so, we consider different values of p. p represents the level of noise that has been discretized between 0 and 0.5. For each fixed value of p, we simulate 100 Monte-Carlo replicates of the random model. We apply the l_1-spectral clustering algorithm to cluster the nodes. Then, we evaluate the performances of the algorithm by computing the percentage of misassigned nodes in average defined as $\frac{1}{100}\sum_{j=1}^{100} \mid \{i \in V : \tau(i) \neq \hat{\tau}_j(i)\} \mid$, where τ_j is the block membership function and $\hat{\tau}_j$ is the estimated membership function for the j-th model. The obtained results have been plotted in Figure 5 and in Figure 6.

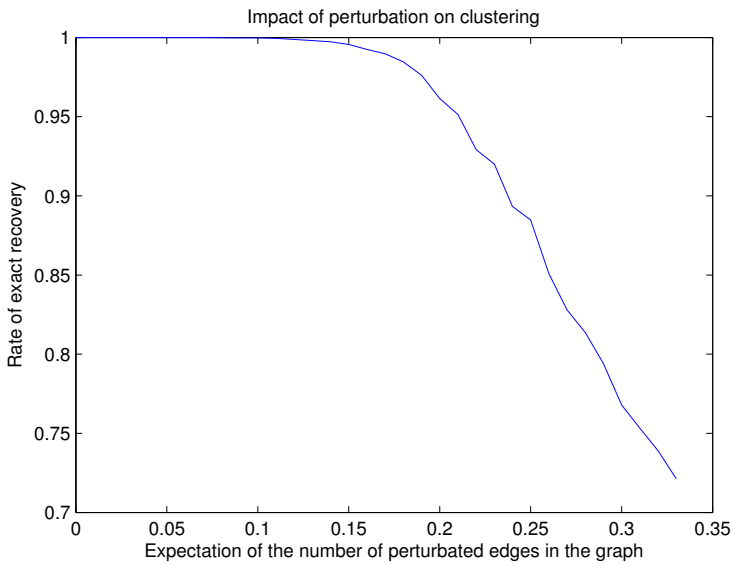


Figure 5: Fraction of nodes correctly classified using l_1 spectral clustering, as the level of noise varies in random generated graphs of the type described above. Size of the groups between 20 and 30. Number of groups between 10 and 20.

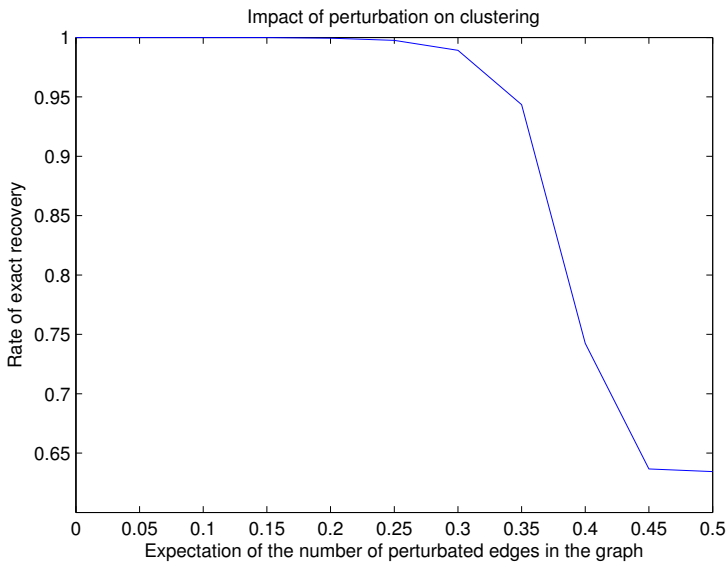


Figure 6: Fraction of nodes correctly classified using l_1 spectral clustering, as the level of noise varies in random generated graphs of the type described above. Size of the groups between 50 and 100. Number of groups between 20 and 30.

The results are good and the algorithm seems to work well on simulated data for small perturbations. The rate of exact assignment is equal or very close to one for small perturbations. In addition, thank to the l_1 norm this algorithm is very fast, even for a large number of nodes, when the number of communities is small. These results should be investigated further to better understand the advantages but also the drawbacks induced by this method. It would be of interest to see if it could be possible to set phase transition phenomena for this model, in the same vein as the ones stated by [DKMZ11], [MNS12,

MNS13, MNS14],[ABH14], [AS15] for the stochastic block model when the number of nodes tends to infinity.

7 Conclusion

We have introduced a random graph model that is closely related to a stochastic block model. The observed graph is assumed to result from a deterministic graph with an exact community structure, whose edges have been perturbed by Bernoulli variables. We have characterized the indicators of the communities in the ideal graph as the ones that have the minimal l_1-norm with respect to a specific restricted space. We have presented a variant of the spectral clustering algorithm where the vectors used to partitionned the graph are a kind of denoised version. The main advantages of this method is that the objective function is clear, simple and easy to implement even for very large graphs and this procedure seems to work well on simulated data. We have investigated some of the properties of the noise matrix and we have studied the performances of this method on simulated data. However, the main limitation is that it requires the knowledge of one representative for each of the communities.

References

[AB02] Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Reviews of modern physics*, 74(1):47, 2002.

[ABH14] Emmanuel Abbe, Afonso S Bandeira, and Georgina Hall. Exact recovery in the stochastic block model. *arXiv preprint arXiv:1405.3267*, 2014.

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

[DHJ+04] Adrian Dobra, Chris Hans, Beatrix Jones, Joseph R Nevins, Guang Yao, and Mike West. Sparse graphical models for exploring gene expression data. *Journal of Multivariate Analysis*, 90(1):196–212, 2004.

[Die05] Reinhard Diestel. Graph theory (graduate texts in mathematics). 2005.

[DKMZ11] Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physical Review E*, 84(6):066106, 2011.

[FIE]

[GA05] Roger Guimera and Luis A Nunes Amaral. Functional cartography of complex metabolic networks. *Nature*, 433(7028):895–900, 2005.

[GN02] Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.

[HG10] Mark S Handcock and Krista J Gile. Modeling social networks from sampled data. *The Annals of Applied Statistics*, 4(1):5–25, 2010.

[Hop82] John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

[JTA+00] Hawoong Jeong, Bálint Tombor, Réka Albert, Zoltan N Oltvai, and A-L Barabási. The large-scale organization of metabolic networks. *Nature*, 407(6804):651–654, 2000.

[MDJL04] Michael Z Man, Greg Dyson, Kjell Johnson, and Birong Liao. Evaluating methods for classifying expression data. *Journal of Biopharmaceutical statistics*, 14(4):1065–1084, 2004.

[MLF+09] David Meunier, Renaud Lambiotte, Alex Fornito, Karen D Ersche, and Edward T Bullmore. Hierarchical modularity in human brain functional networks. *Frontiers in neuroinformatics*, 3, 2009.

[MNS12] Elchanan Mossel, Joe Neeman, and Allan Sly. Stochastic block models and reconstruction. *arXiv preprint arXiv:1202.1499*, 2012.

[MNS13] Elchanan Mossel, Joe Neeman, and Allan Sly. A proof of the block model threshold conjecture. *arXiv preprint arXiv:1311.4115*, 2013.

[MNS14] Elchanan Mossel, Joe Neeman, and Allan Sly. Consistency thresholds for binary symmetric block models. *arXiv preprint arXiv:1407.1591*, 2014.

[New03] Mark EJ Newman. The structure and function of complex networks. *SIAM review*, 45(2):167–256, 2003.

[NG04] Mark EJ Newman and Michelle Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[PSV07] Romualdo Pastor-Satorras and Alessandro Vespignani. *Evolution and structure of the Internet: A statistical physics approach*. Cambridge University Press, 2007.

[SR95] Andrew J Seary and William D Richards. Partitioning networks by eigenvectors. In *Proceedings of the International Conference on Social Networks*, volume 1, pages 47–58, 1995.

[SSJ90] Gilbert W Stewart, Ji-guang Sun, and Harcourt Brace Jovanovich. *Matrix perturbation theory*, volume 175. Academic press New York, 1990.

[ST07] Daniel A Spielman and Shang-Hua Teng. Spectral partitioning works: Planar graphs and finite element meshes. *Linear Algebra and its Applications*, 421(2):284–305, 2007.

[Str01] Steven H Strogatz. Exploring complex networks. *Nature*, 410(6825):268–276, 2001.

[VL07] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

[VLBB08] Ulrike Von Luxburg, Mikhail Belkin, and Olivier Bousquet. Consistency of spectral clustering. *The Annals of Statistics*, pages 555–586, 2008.

[YH07] Andy M Yip and Steve Horvath. Gene network interconnectedness and the generalized topological overlap measure. *BMC bioinformatics*, 8(1):22, 2007.

[ZH+05] Bin Zhang, Steve Horvath, et al. A general framework for weighted gene co-expression network analysis. *Statistical applications in genetics and molecular biology*, 4(1):1128, 2005.