l1-spectral clustering algorithm: a robust spectral clustering using Lasso regularization

Camille Champion, Magali Champion, Mélanie Blazère, Jean-Michel Loubes

ℓ_1-spectral clustering algorithm: a robust spectral clustering using Lasso regularization

Camille Champion[a,b], Magali Champion[c,*], Mélanie Blazère[a], Jean-Michel Loubes[a]

[a] *Institut de Mathématiques de Toulouse, UMR 5219, Université de Toulouse, CNRS, France*
[b] *Université Paul Sabatier (UPS), UMR1297, Institut des Maladies Métaboliques et Cardiovasculaires, INSERM, France*
[c] *Université de Paris, CNRS, MAP5 UMR8145, Paris, France*

Abstract

Detecting cluster structure is a fundamental task to understand and visualize functional characteristics of a graph. Among the different clustering methods available, spectral clustering is one of the most widely used due to its speed and simplicity, while still being sensitive to perturbations imposed on the graph. This paper presents a robust variant of spectral clustering, called ℓ_1-spectral clustering, based on Lasso regularization and adapted to perturbed graph models. By promoting sparse eigenbases solutions of specific ℓ_1-minimization problems, it detects the hidden natural cluster structure of the graph. The effectiveness and robustness to noise perturbations of the ℓ_1-spectral clustering algorithm is confirmed through a collection of simulated and real biological data.

Keywords: Unsupervised learning, Spectral clustering, ℓ_1-penalty, Biological networks

1. Introduction

Graphs play a central role in complex systems as they can model interactions between variables of the system. They are commonly used in a wide range of applications, from social sciences (*e.g.* social networks (Handcock and Gile, 2010)) to technologies (*e.g.* telecommunications (Smith, 1997), wireless sensor networks (Akyildiz et al., 2002)) or biology (gene regulatory networks (Davidson and Levin, 2005), metabolic networks (Jeong et al., 2000)). One of the most relevant features when analyzing graphs is the identification of their underlying structures, such as cluster structures, generally defined as connected subsets of nodes that are more densely connected to each other than to the rest of the graph. These clusters can provide an invaluable help in understanding and visualizing the functional components of the whole graph (Girvan and Newman,

*Corresponding author

2002; Newman and Girvan, 2004; Abbe, 2017). For instance, in genetics, groups of genes with high interactions are likely to be involved in a same function that drives a specific biological process.

Since the pioneering exploratory works in the early 50s, a large number of clustering methods have launched. Among them, partitioning algorithms, which include the well-known k-means (MacQueen, 1967), classify nodes into a predefined number of groups based on a similarity measure and hierarchical clustering algorithms (Hastie et al., 2001) build a hierarchy of clusters through dendrogram representations. More recently, spectral clustering algorithms, popularized over years by Shi and Malik (2000); Ng et al. (2002), particularly draw the attention of the community research due to their speed, simplicity and numerical performances. As its name suggest, spectral clustering algorithms mainly use the spectral properties of the graph by (i) computing the eigenvectors of the associated Laplacian matrix (or one of its derivatives), which gives information about the structure of the graph, and (ii) performing k-means on it to recover the induced cluster structure. A large number of extensions of the original spectral clustering algorithm, as presented in Luxburg (2007), have been proposed, with applications to different fields (Zelnik-Manor and Perona, 2005; Wang and Davidson, 2010; Li et al., 2019).

While spectral clustering is widely used in practice, handling noise sensitivity remains a tricky point (Bojchevski et al., 2017), mainly due to the k-means algorithm, which is highly sensitive to noise. This issue has been considerably studied with extensions of the k-means to noisy settings so that it recovers the cluster structure in spite of the unstructured part of the input data (Tang and Khoshgoftaar (2004); Pelleg and Baras (2007)). More generally, the robustness of spectral clustering algorithms has recently been investigated for perturbed graphs derived from stochastic block models (SBM) (Stephan and Massoulié (2019); Peche and Perchet (2020)). In this context, Joseph and Yu (2016) explored the effect of regularization on spectral clustering, as proposed in (Amini et al., 2013), and Zhang and Rohe (2018) particularly highlighted its benefit for clustering sparse perturbed graphs. Equally, Lara and Bonald (2020) showed on a simple block model that spectral regularization separates the underlying blocks of the graph. In this paper, we develop an alternative regularized method of the spectral clustering, called ℓ_1-spectral clustering algorithm and based on Lasso regularization (Tibshirani et al., 2001). In our model, as in the spectral clustering algorithm, we carefully explore the underlying structure of the graph through the Laplacian matrix spectrum to cluster nodes. However, by directly promoting a sparse eigenvectors basis solution to an ℓ_1-norm optimization problem, it does not require the k-means step to extract clustering structures, making it more robust in highly perturbed graph situations.

The paper is organized as follows: in Section 2, we introduce some preliminary concepts about graph clustering and more specifically spectral clustering. In Section 3 and 4, we present the ℓ_1-spectral clustering we developed, from a theoretical and an algorithmic point of view. In Section 5, we finally show its efficiency and accuracy through experiments on simulated and biological real data set and compare it with state-of-the-art clustering methods.

2. Reminders about graph and spectral clustering

2.1. Graphs modeling and notations

This work considers the framework of an unknown undirected graph $\mathcal{G}(V, E)$, with no retroactive loop, consisting of n vertices $V = \{1, \ldots, n\}$ and a set of edges $E \subseteq V \times V$ connecting each pair of vertices. As usual, the graph $\mathcal{G}$ is represented by its associated adjacency matrix $A = (A_{ij})_{(i,j)\in E}$ of size $n \times n$, whose non-zero elements correspond to the edges of $\mathcal{G}$:

$$\forall (i,j) \in [\![1,n]\!]^2, \quad A_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

As $\mathcal{G}$ is undirected with no retroactive loop, the adjacency matrix A is symmetric with zero on its diagonal. Before turning to the next section, we recall some useful graph definitions.

Definition 1. *The degree d_i of a node $i \in V$ of $\mathcal{G}$ is defined as the number of edges that are incident to i: $d_i = \sum_{j=1}^{n} A_{ij}$. The induced degree matrix D is then the $n \times n$ matrix containing $(d_1, \ldots, d_n)$ on its diagonal and zero elsewhere:*

$$D = diag\ (d_1, \ldots, d_n).$$

Definition 2. *A connected component C of $\mathcal{G}$ is a subset of nodes from V such that each pair of nodes of C is connected by a path and there is no connection between vertices in C and outside C. Connected components $C_1, \ldots, C_k$ are a k-partition of the set V of vertices if the three following conditions hold:*

(i) they are non-empty: $\forall i \in [\![1,k]\!], C_i \neq \emptyset$,

(ii) they are pairwise disjoints: $\forall (i,j) \in [\![1,k]\!]^2, C_i \cap C_j = \emptyset$,

(iii) their union form the set of all vertices: $\bigcup_{i=1}^{k} C_i = V$.

Definition 3. *Let $C_1, ..., C_k$ be a k-partition of the set of vertices V of $\mathcal{G}$. Then, the indicators $(\mathbf{1}_{C_i})_{i\in\{1,\ldots,k\}}$ of this partition are defined as the vectors of size n, whose coefficients satisfy:*

$$\forall i \in [\![1,k]\!], \forall j \in [\![1,n]\!], \quad (\mathbf{1}_{C_i})_j = \begin{cases} 1 & \textit{if vertex } j \textit{ belongs to } C_i, \\ 0 & \textit{otherwise.} \end{cases}$$

In the present paper, we assume that the graph $\mathcal{G}$ is the union of k complete graphs, whose set of vertices $C_1, \ldots, C_k$ form a k-partition of $\mathcal{G}$. We denote by $c_1, \cdots, c_k$ their respective size ($\sum_{i=1}^{k} c_i = n$). To simplify, we assume that the nodes, labeled from 1 to n, are ordered with respect to their block membership and the size of the blocks. From a matrix point of view, the associated adjacency matrix A is a k-block diagonal matrix of size $n \times n$ of the form:

$$A = \begin{bmatrix} \underbrace{\begin{matrix} 0 & 1 & \cdots & 1 \\ 1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 1 \\ 1 & \cdots & 1 & 0 \end{matrix}}_{c_1} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \underbrace{\begin{matrix} 0 & 1 & \cdots & 1 \\ 1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 1 \\ 1 & \cdots & 1 & 0 \end{matrix}}_{c_k} \end{bmatrix}. \tag{1}$$

2.2. Graph clustering through spectral clustering

Graph clustering consists in grouping the vertices of the graph $\mathcal{G}$ into clusters according to its edge structure. Whereas some of the most traditional clustering algorithms are based on partitions (*e.g.* k-means) and hierarchies (*e.g.* hierarchical clustering algorithm), spectral clustering takes advantage of the spectral properties of the graph. A large number of spectral clustering algorithms exists in the literature. The most common version, presented in (Luxburg, 2007) and recapped in Algorithm 1 below, uses the properties of the Laplacian matrix (Definition 4) to detect clusters in the graph.

Definition 4. *Given a graph $\mathcal{G}$, the Laplacian matrix L is defined as:*

$$L = D - A,$$

where A is the adjacency matrix and D the degree matrix associated to $\mathcal{G}$.

Algorithm 1 Spectral clustering algorithm

Require: $\mathcal{G}$ a graph, A its associated adjacency matrix, $\hat{k}$ number of clusters to build.

1: Compute the Laplacian matrix $L = D - A$.
2: Perform the spectral decomposition of L and store the $\hat{k}$ first eigenvectors $U := (u_1, \cdots, u_{\hat{k}})$.
3: Cluster U with the k-means algorithm into clusters $C_1, \cdots, C_{\hat{k}}$.
4: **return** Clusters $C_1, \cdots, C_{\hat{k}}$.

By definition, the diagonal of L is equal to the degrees of the nodes. Moreover, in the ideal case where $\mathcal{G}$ has an underlying partition form with k connected components and a block diagonal adjacency matrix A, as given in Equation (1), the eigenvalue 0 of L is of multiplicity k and the associated eigenvectors correspond to the indicator vectors of the k components. These k components can then be recovered only by performing spectral decomposition of L. However, in the perturbed case, any perturbation caused by introducing and/or removing edges between and/or inside the components makes $k - 1$ of the k eigenvalues 0 slightly larger than 0 and changes the corresponding eigenvectors. The final

cluster structure is thus no longer explicitly represented. The spectral clustering algorithm then uses the k-means algorithm on these eigenvectors to discover the hidden underlying structure, which is hampered by perturbations.

Since the first development of the spectral clustering algorithm, it has been studied a lot and extended many times in different communities (Hagen and Kahng, 1992; Hendrickson and Leland, 1995; Pothen, 1997; Shi and Malik, 2000; Ng et al., 2002; Zelnik-Manor and Perona, 2005) with powerful results. Refinements include the use of normalized versions of the Laplacian matrix, such as the symmetric and the random walk normalized ones (Luxburg, 2007). Nevertheless, the performances of the spectral clustering have shown to be very sensitive to perturbations, which often occurs when dealing with real data (Bojchevski et al., 2017). To provide more robustness with respect to perturbations, we thus developed the ℓ_1-spectral clustering algorithm, described in Section 3.

3. An ℓ_1-version of the spectral clustering algorithm

In this section, we describe the ℓ_1-spectral clustering algorithm we developed as an alternative to the standard spectral clustering for clustering perturbed graphs. In this context, to ensure a good recovery of the connected components, the eigenvectors basis should be carefully defined. The key point is to replace the k-means procedure, which fails while the perturbation grows, by selecting relevant eigenvectors that provide useful information about the graph structure. As the spectral clustering algorithm, the ℓ_1-spectral clustering focuses on the spectral properties of the graph.

Let $\mathcal{G} = (V, E)$ be a graph formed of k connected components, as defined in Section 2, and A its associated adjacency matrix. We denote by $(\lambda_i)_{1\leq i\leq n}$ the n-eigenvalues of A, sorted in increasing order:

$$\lambda_1 \leq ... \leq \lambda_n,$$

and $v_1, ..., v_n$ their associated eigenvectors. We define by $\mathcal{V}_k$ the eigenspace generated by the k largest eigenvectors:

$$\mathcal{V}_k := \mathrm{Span}(v_{n-k+1}, ..., v_n).$$

In the ideal case, where the graph is not perturbed, the indicators $(\mathbf{1}_{C_i})_{1\leq i\leq k}$ of the connected components $C_1, \ldots, C_k$ correspond exactly to the eigenvectors of the Laplacian matrix L associated to the eigenvalue 0 of multiplicity k (see Section 2.2). As regards the adjacency matrix A, these indicators correspond this time to the k eigenvectors $v_{n-k+1}, \ldots, v_n$, associated to the k largest eigenvalues $\lambda_{n-k+1}, ..., \lambda_n$. In the perturbed case, unlike the traditional spectral clustering, the ℓ_1-spectral clustering algorithm does not directly use the subspace $\mathcal{V}_k$ to recover the k connected components but computes another eigenbasis that promotes sparse solutions, as detailed in the next sections.

3.1. General ℓ_0-minimization problem

Propositions 1 and 2 below show that the connected components indicators $(\mathbf{1}_{C_i})_{i\in\{1,\dots,k\}}$ are solutions of ℓ_0-minimization problems.

Proposition 1. *The minimization problem*

$$\underset{v\in\mathcal{V}_k\setminus\{0\}}{\arg\min} \quad \|v\|_0 \tag{$\mathcal{P}_0$}$$

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

In other words, $\mathbf{1}_{C_1}$ is the sparsest non-zero eigenvector in the space spanned by the eigenvectors associated to the k largest eigenvalues.

Proof. We recall that, for all $v \in \mathbb{R}^n$, $\|v\|_0 = |\{j \in [\![1, n]\!],\ \ v_j \neq 0\}|$. Let $v \in \mathcal{V}_k\setminus\{0\}$. As $(\mathbf{1}_{C_j})_{1\leq j\leq n} \in \mathcal{V}_k$, v can be decomposed as $v = \sum_{j=1}^{k} \alpha_j \mathbf{1}_{C_j}$ where $\alpha = (\alpha_1, \dots, \alpha_k) \in \mathbb{R}^k$ and there exists $j \in \{1, ..., k\}$ such that $\alpha_j \neq 0$. By definition of the ℓ_0-norm, we then have:

$$\|v\|_0 = \mathbf{1}_{\alpha_1\neq 0}c_1 + \cdots + \mathbf{1}_{\alpha_k\neq 0}c_k, \tag{2}$$

with $c_1 \leq ... \leq c_k$ the sizes of the k connected components. The solution of $(\mathcal{P}_0)$, which minimizes Equation (2), is thus given by setting $\alpha = (\alpha_1, 0, \dots, 0)$ with $\alpha_1 \neq 0$. □

Proposition 1 can then be generalized to iteratively find the indicators associated to the largest connected components introducing sparsity and orthogonality constraints. For $i \in [\![2, k]\!]$, let $\mathcal{V}_k^i$ refers to:

$$\mathcal{V}_k^i := \{v \in \mathcal{V}_k,\ \ \forall l = 1, \dots, i-1,\ \ v \perp \mathbf{1}_{C_l}\}.$$

Proposition 2. *Let $i \in [\![2, k]\!]$. The minimization problem*

$$\underset{v\in\mathcal{V}_k^i\setminus\{0\}}{\arg\min} \quad \|v\|_0 \tag{$\mathcal{P}_0^i$}$$

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

Solving $(\mathcal{P}_0)$ and $(\mathcal{P}_0^i)_{2\leq i\leq k}$ is a NP-hard problem, which is not computationally feasible. To tackle this issue, the classical idea consists in replacing the ℓ_0-norm by its convex relaxation, the ℓ_1-norm, defined for all $v \in \mathbb{R}^n$ as $\|v\|_1 = \sum_{1\leq j\leq n} |v_j|$.

In the next section, we show that the solution of the ℓ_0 optimization problems remains the same by replacing the ℓ_0-norm by the ℓ_1-norm, at the price of slight constraints on the connected components.

3.2. Relaxed ℓ_1-minimization problem

From now on, we assume that we know one representative element for each component, that is a node belonging to each component, denoted by $(i_1, ..., i_k)$ thereafter. Let $\tilde{\mathcal{V}}_k = \{v \in \mathcal{V}_k, v_{i_1} = 1\}$. Then, it is straightforward to see that the indicator vector of the smallest component is solution to the following optimization problem:

Proposition 3. *The minimization problem*

$$\underset{v \in \tilde{\mathcal{V}}_k}{\arg\min} \quad \|v\|_1 \tag{$\mathcal{P}_1$}$$

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

Proof. We recall that, for all $v \in \mathbb{R}^n$, $\|v\|_1 = \sum_{j=1}^n |v_j|$. Let $v \in \tilde{\mathcal{V}}_k$. As $(\mathbf{1}_{C_j})_{1 \leq j \leq n} \in \mathcal{V}_k$, v can be decomposed as $v = \sum_{j=1}^k \alpha_j \mathbf{1}_{C_j}$ where $\alpha = (\alpha_1, \ldots, \alpha_k) \in \mathbb{R}^k$ and there exists $j \in \{1, \ldots, k\}$ such that $\alpha_j \neq 0$. By definition of the ℓ_1-norm, we then have:

$$\|v\|_1 = |\alpha_1| c_1 + \cdots + |\alpha_k| c_k, \tag{3}$$

with $c_1 \leq ... \leq c_k$ the sizes of the k connected components. The solution of $(\mathcal{P}_1)$, which minimizes Equation (3), is thus given by setting $\alpha = (\alpha_1, 0, \ldots, 0)$ with $\alpha_1 = 1$. □

To simplify and without loss of generality, we assume that i_1 corresponds to the first node. We can then rewrite $(\mathcal{P}_1)$ as:

$$\underset{\substack{v \in \mathbb{R}^{n-1} \\ (1,v)^T \in \mathcal{V}_k}}{\arg\min} \quad \|v\|_1.$$

Constraints in $(\mathcal{P}_1)$ can be converted into the following equality contraints:

Proposition 4. *Let $U_k := (v_1, ..., v_{n-k})$ the matrix formed by the eigenvectors associated with the $n-k$-smallest eigenvalues. We denote by w^T its first row and W^T the matrix obtained after removing w^T from U_k:*

$$U_k := (v_1, ..., v_{n-k}) = \begin{bmatrix} \boxed{w^T} \\ \boxed{W^T} \end{bmatrix} \tag{4}$$

The minimization problem

$$\underset{\substack{v \in \mathbb{R}^{n-1} \\ Wv = -w}}{\arg\min} \quad \|v\|_1 \tag{$\tilde{\mathcal{P}}_1$}$$

has a unique solution v^* such that $(1, v^*)^T = \mathbf{1}_{C_1}$.

Proof. Since A is symmetric, its eigenvectors form an orthogonal basis and, for all $v \in \mathcal{V}_k$, we have $U_k^T v = 0$. Let $(1, v)^T \in \mathcal{V}_k$. Using Equation (4), we deduce that:

$$U_k^T \begin{pmatrix} 1 \\ v \end{pmatrix} = w + Wv = 0.$$

The constraint in $(\tilde{\mathcal{P}}_1)$ is thus equivalent to the constraint in $(\mathcal{P}_1)$, which ends the proof. □

3.3. Generalization of the relaxed ℓ_1-minimization problem

Obviously, the indicator vector $\mathbf{1}_{C_1}$ alone is not sufficient to know the complete graph structure. However, Proposition 4 can be extended to find the remaining indicator vectors. To do so, as in Proposition 2, we add the constraint that the target vector is orthogonal to the previously computed vectors, which is done in practice by applying a Gram-Schmidt orthonormalization procedure (see Section 4 below for more details about the procedure).

4. The ℓ_1-spectral algorithm

4.1. Global overview of the algorithm

In this section, we present a global overview of the ℓ_1-spectral clustering algorithm we implemented to recover the components of any perturbed graph (see Algorithm 2 below). It is available as a `R`-package on GitHub at https://github.com/championcamille/l1-SpectralClustering. Some precisions about the algorithm and parameters setting are given in the next paragraphs.

4.2. Solving the ℓ_1-minimization problem

This section is devoted to the resolution of the constrained ℓ_1-optimization problem $(\tilde{\mathcal{P}}_1)$ (line 6 of Algorithm 2). To be simplified, it can be equivalently written as the following penalized problem:

$$\underset{v \in \mathbb{R}^{n-1}}{\arg\min} \ \|Wv + w\|_2^2 + \lambda \|v\|_1, \qquad (\mathcal{P}_{\text{Lasso}})$$

where, for all $v \in \mathbb{R}^{n-1}$, $\|v\|_2^2 = \sum_{j=1}^{n-1} v_j^2$ and $\lambda > 0$ is the regularization parameter that controls the balance between the constraint and the sparsity. Two methods are proposed thereafter to solve $(\mathcal{P}_{\text{Lasso}})$.

Lasso solution.
The most traditional method to deal with such an ℓ_1-minimization problem is the Lasso procedure, developed by Tibshirani (1996). As for all regularizing methods, the choice of λ is of great importance. Here, especially, taking λ too large will lead to an over-constrained problem and a large number of nodes of $\mathcal{G}$ may not be clustered into components. In practice, K-fold cross validation, as implemented in the `glmnet` `R`-package, can be used to optimally set λ.

Algorithm 2 ℓ_1-spectral clustering algorithm

Input: $\mathcal{G}$ a graph, A its associated adjacency matrix, $\hat{k}$ number of clusters to recover and $(i_j)_{j\in\{1,\dots,\hat{k}\}}$ family of representative elements of each cluster.
Perform the spectral decomposition of A, sort the eigenvalues by increasing order and store the associated eigenvectors: $V := (v_1, ..., v_n)$.
for $j = 1$ **to** $\hat{k}$ **do**
    Define $U_{\hat{k},j}$ as the matrix that contains the $n - \hat{k} - j + 1$ first columns of V:

$$U_{\hat{k},j} := (v_1, ..., v_{n-\hat{k}-j+1}).$$

    Split $U_{\hat{k},j}$ into two parts:

$$w^T := U^{i_j}_{\hat{k},j}, \quad \text{the } i_j\text{-th row of } U_{\hat{k},j},$$

$$W^T := U^{-i_j}_{\hat{k},j}, \quad \text{the other rows of } U_{\hat{k},j}.$$

    Solve the ℓ_1-minimization problem $(\tilde{\mathcal{P}}_1)$:

$$v^* := \underset{\substack{v\in\mathbb{R}^{n-1}\\ Wv=-w}}{\arg\min} \ \|v\|_1.$$

    Recover the indicator of the j-th component:

$$\hat{\mathbf{1}}_{C_j} = (v_1^*, \dots, v_{i_j-1}^*, 1, v_{i_j}^*, \dots, v_n^*).$$

    Update v_j in V: $v_j \leftarrow \hat{\mathbf{1}}_{C_j}$.
    Perform Gram-Schmidt orthogonalization on V to ensure orthogonality between v_j and the rest of the columns of V:

$$V \leftarrow \text{Gram-Schmidt}(V).$$

end for
Output: $(\hat{\mathbf{1}}_{C_j})_{1\le j\le\hat{k}}$ the indicators of the $\hat{k}$ connected components.

Thresholded least-squares solution.
Another method consists in solving the least-squares problem:

$$v^* := \underset{v\in\mathbb{R}^{n-1}}{\arg\min} \ \|Wv + w\|_2^2,$$

and then thresholding v^* given some predefined threshold t:

$$\forall j \in [\![1, n-1]\!], \quad v_j^* = \begin{cases} 1 & \text{if } v_j^* > t, \\ 0 & \text{otherwise.} \end{cases}$$

Of course, this thresholding step imposes sparsity on the solution. However, we can wonder if nodes with large coefficients should really be clustered together.

In our model, the ideal parameters to recover (indicators of the components) do not take continuous values. Enforcing the coefficients of all representative elements to be equal to 1, under small perturbations, the coefficients of all other nodes belonging to the same components should then be close to 1. This specific behavior is underlined in Figure 1. In this example, we generated a graph $\mathcal{G}$ with 50 nodes, split into 5 connected components. We perturbed the structure of the graph by adding and removing edges with a probability p of 1%, 10%, 25% and 50%. We then solved ($\mathcal{P}_{\text{Lasso}}$) to recover the first component only.

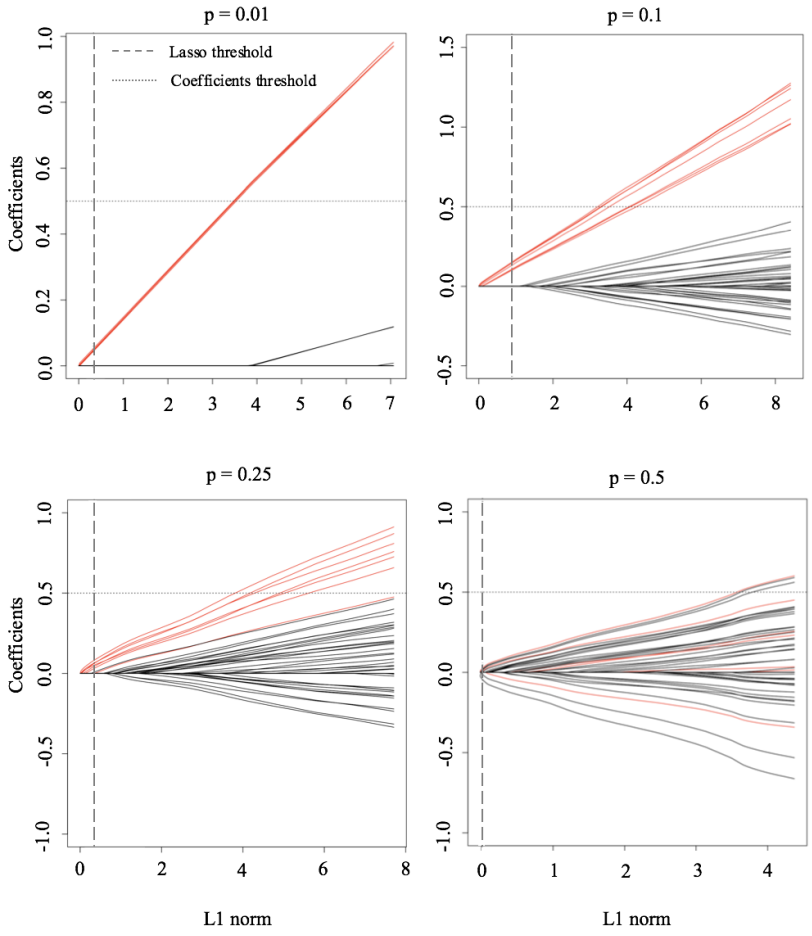

Figure 1: Evolution of the coefficients of v, solution of ($\mathcal{P}_{\text{Lasso}}$), with respect to $\|v\|_1$ for different perturbations of the ideal graph (from top left to bottom right $p = 1\%$, 10%, 25% and 50%). Red lines correspond to the coefficients belonging to the component we aim at recovering, in contrast with black ones. Dotted lines are related to the ℓ_1-norm-threshold (vertical), associated with the Lasso solution, and the threshold on the value of the coefficients (horizontal), associated with the thresholded least-squares solution.

As can be seen in Figure 1, the Lasso and thresholded least-squares solutions give almost the same results: for small perturbations ($p \leq 10\%$, at the top), the whole component is perfectly retrieved. For $p = 25\%$ (at the bottom left), all coefficients are tighter but both methods still work, forgetting only one node. As the perturbation becomes too large ($p = 50\%$, at the bottom right), the selection of nodes belonging to the first component fails.

4.3. Optimally tuning the number of clusters

Traditional clustering algorithms, such as k-means, require the user to specify the number of connected components of the graph $\mathcal{G}$ to recover, which is, in practice, unavailable. Determining the optimal number of components $\hat{k}$ thus becomes a fundamental issue. A large number of methods have been developed in this sense: the hierarchical clustering for example looks for a hierarchy of components using dendrograms. The Elbow, average silhouette and gap statistic methods (Tibshirani et al., 2001) are also frequently used in addition to clustering techniques.

In our work, as proposed by Luxburg (2007), we focus on the heuristic eigengap method, which consists in choosing $\hat{k}$ such that it maximizes the eigengap, defined as the difference between consecutive eigenvalues of the Laplacian matrix L. This procedure is particularly well-suited in a spectral context. Indeed, in the ideal case, perturbation theory ensures that there exists a gap between the eigenvalue 0 of multiplicity k and the next $k+1$-th one. In the perturbed case, while being less strong, an eigengap still exists.

4.4. Finding the representative elements

In addition to the number of connected components, to run the ℓ_1-spectral clustering algorithm, we need to know at least one representative element of each component. This assumption may be restrictive when working with real data. However, it makes sense in a large number of situations where clusters are chosen to classify nodes around specific elements of the graph.

To avoid an arbitrary choice of such elements, one solution consists in estimating them using a rough partitioning algorithm. Another solution is to explore the structure of the graph to find hubs of densely connected parts. In this work, this is done by computing the betweeness centrality score of all nodes. In graph theory, the betweeness score S_b measures the centrality of a node based on the number of shortest paths passing through it:

$$\forall \ell \in [\![1, n]\!], \quad S_b(\ell) = \sum_{1 \leq i,j \leq n} \frac{\#\text{ shortest paths from } i \text{ to } j}{\#\text{ shortest paths from } i \text{ to } j \text{ passing through } \ell}.$$

In practice, the representative elements of the k components are chosen to maximize this score.

Note that the nodes with the highest betweeness scores should be those that connect the densest parts of the graph. The risk of clustering two nodes from different connected components may thus be high. To avoid this, we add

a stabilization step to our algorithm. As soon as one of the nodes with the k highest scores is added to a component during the minimization step, it is removed from the list of potential representative elements. We then re-run the algorithm using the k nodes taken among the $k+1$ ones with the highest scores, and so on until stabilization of the list of representative elements.

5. Numerical experiments

This section is dedicated to experimental studies to assess numerical performances of the ℓ_1-spectral clustering algorithm through two datasets. First, we show that it behaves well on simulated data with a variety of different settings and in comparison with state-of-the-art spectral clustering methods. Then, using a gene expression data set from kidney cancer patients, we demonstrate the ability of our algorithm to discover relevant groups of genes that act together to characterize the disease.

5.1. Application to toy datasets

5.1.1. Numerical settings

To explore the capabilities and the limits of the ℓ_1-spectral clustering algorithm with respect to state-of-the-art methods, we first considered simulated data, whose settings are detailed in the next paragraphs.

Simulated data set.
We generated random ideal graphs for a given number of nodes n ($n = 20$, 50 and 100) and a given number of connected components k ($k = 2$, 5, 10). The component sizes $(c_j)_{1\leq j\leq k}$ were chosen in a balanced way: $\forall j \in [\![1, k-1]\!], c_j = \lfloor n/k \rfloor$, with $\sum_{j=1}^{k} c_j = n$. With a probability p_{in} and p_{out} of removing an edge from a component and of introducing an edge between two components varying from 0.01 to 0.5, we created multiple perturbed versions of the graph. All experiments were replicated 100 times each for better robustness.

Algorithm parameters.
As some of the methods we compare with require the number of components to form, we focus on two versions of the ℓ_1-spectral clustering: the one presented in Algorithm 2, for which the number of clusters and a list of representative elements are assumed to be known, and the self-tuned one, for which both of them are extracted from the graph, as explained in Section 4. The results being very similar, we choose to focus on the thresholded least-squares solution to solve the ℓ_1-minimization problem $(\tilde{\mathcal{P}}_1)$ in Algorithm 2. The corresponding threshold parameter t is fixed using 5-fold cross-validation.

Comparison with state-of-the-art.
We compare the ℓ_1-spectral clustering with three other graph-based clustering algorithms: first, the spectral clustering (Algorithm 1), which is available in the `R`-package `anocva`, then, SpectACl, which was developped by Hess et al. (2019) with the aim of exhibiting both minimum cut and maximum density of

the clusters. This algorithm can be viewed as a combination of DBSCAN, a density-based clustering algorithm (Ester et al., 1996) which is mainly used to identify clusters of any shape in a data set containing noise and outliers, and spectral clustering. SpectACl is publicly available on the Bitbucket platform as a Python code at https://bitbucket.org/Sibylse/spectacl/src/master/. Both methods requiring the number of components to cluster as an input, we finally run the Self-Tuning Spectral Clustering from Zelnik-Manor and Perona (2005), available on GitHub as a Python code at https://github.com/wOOL/STSC. The latter is an improved version of the spectral clustering, in which the final postprocessing step (k-means) is removed and the structure of the eigenvectors is carefully analyzed to automatically infer the number of clusters. It is thus used to evaluate the performances of the self-tuned version of the ℓ_1-spectral algorithm.

Performance metrics.
Performances are measured by comparing the learnt components with the true ones, which are obviously known in the context of simulated data. Among the large number of existing scores, we used the Normalized Mutual Information (NMI) score, for its ability to compare clusters that could be of different sizes. The closer to 1 the NMI score, the better the classification.

5.1.2. Effect of the dimension and cluster sizes on perturbed graphs

First, we aimed at exploring the effect of the dimension and cluster sizes on the performances of the ℓ_1-spectral clustering algorithm. For n ranging from 20 to 100 and k from 2 to 10, results, in terms of NMI scores, are summarized in Table 1. On the one hand, for fixed n and k, when the perturbations are small (p_{in}, $p_{out} < 0.25$), one may note that the ℓ_1-clustering algorithm works well (it is clearly a favorable situation). On the other hand, a crude decrease in performance results can be observed as the perturbation grows (p_{in} or $p_{out} \geq 0.25$). In that case, the perturbed graph is far from the original one, which makes hard the recovery of the components. As expected, this becomes even more significant while the dimension increases (from top left to bottom right for each value of n). For perturbations of 0.5 (last line), the NMI scores do not exceed 0.5, which means that the ℓ_1-spectral clustering algorithm almost fails to recover the components. However, we must keep in mind that imposing a perturbation of 0.5 on a graph strongly affects its structure, with a probability of removing an edge inside a component and introducing an edge between components of 50%.

5.1.3. Performance results with respect to state-of-the-art

To give more credit to the ℓ_1-spectral clustering algorithm, we also evaluated its robustness in comparison with the spectral and SpectACl algorithms (see Section 5.1.1) for clustering different perturbed versions of a graph with $n = 100$ and $k = 10$. For each perturbation, we generated 100 graphs and computed the clustering performances using NMI scores.

Table 1: NMI scores obtained after clustering perturbed graphs of different sizes using the ℓ_1-spectral clustering algorithm. All results are averaged over 100 replicates.

		n=20		n=50			n=100	
p_{in}	p_{out}	$k=2$	$k=5$	$k=2$	$k=5$	$k=10$	$k=5$	$k=10$
0.01	0.01	1	1	1	1	1	1	1
	0.1	1	1	1	1	1	0.99	0.99
	0.25	1	0.92	1	1	0.71	0.99	0.88
	0.5	0.99	0.65	1	0.76	0.51	0.96	0.40
0.1	0.01	1	0.99	1	1	1	0.99	1
	0.1	1	0.98	1	1	0.96	0.99	0.99
	0.25	0.99	0.84	1	0.98	0.63	0.99	0.69
	0.5	0.91	0.60	1	0.51	0.49	0.79	0.35
0.25	0.01	0.99	0.96	1	1	0.98	0.99	0.98
	0.1	0.99	0.91	1	0.99	0.79	0.99	0.87
	0.25	0.94	0.69	1	0.78	0.54	0.95	0.47
	0.5	0.54	0.54	0.75	0.32	0.46	0.27	0.30
0.5	0.01	0.95	0.84	0.99	0.93	0.82	0.99	0.86
	0.1	0.83	0.68	0.97	0.73	0.55	0.90	0.51
	0.25	0.52	0.54	0.73	0.34	0.46	0.32	0.30
	0.5	0.22	0.46	0.11	0.21	0.43	0.10	0.25

Results can be visualized in Figure 2, which also indicates the 50% confidence interval. As can be seen, the ℓ_1-spectral and spectral clustering algorithms are very similar, especially for small perturbations ($p_{out} < 0.25$). However, as the perturbations grow, the ℓ_1-spectral clustering algorithm shows a smaller impact to noise sensitivity than the spectral one, being, almost ever, the best method. The results of SpectACl are oddly bad but this may be due to the fact that it was developed for clustering nonconvex shapes, which is beyond the scope of the present work.

An interesting question is how the self-tuning version of the ℓ_1-spectral clustering, for which the number of clusters and the representative elements are self-evaluated, compare with the Self-Tuning Spectral Clustering (see Section 5.1.1). For $n = 20$, $k = 2$ and perturbations ranging from 0.01 to 0.5, we generated 100 versions of the same graph. Results are given in Figure 3. At the top, the NMI scores indicate that the performances of the self-tuning ℓ_1-spectral clustering (in red) decrease while the perturbations grow. On the contrary, the Self-Tuning Spectral Clustering (in blue) seems to be less sensitive to the increase of p_{in} but provide bad results for $p_{in} < 0.25$ and $p_{out} \geq 0.25$, even though it is a more favorable situation.

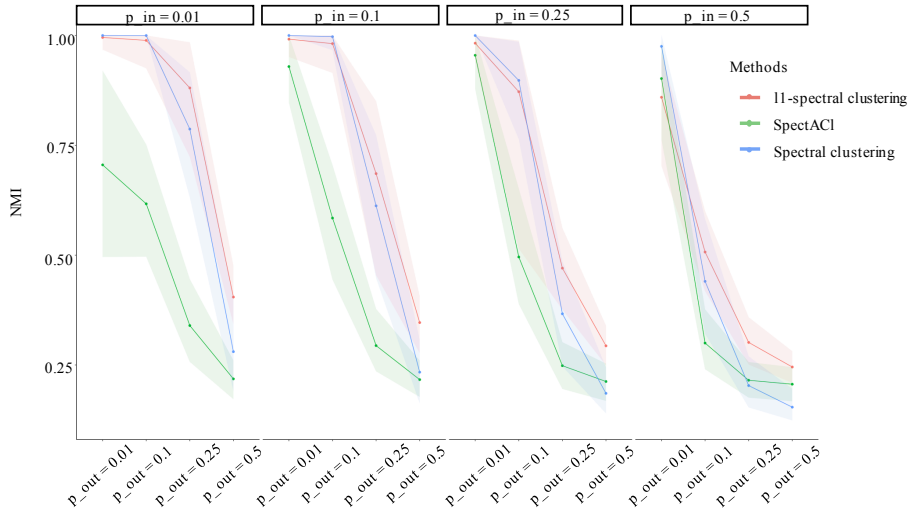

Figure 2: Clustering results, in terms of NMI, of the ℓ_1-spectral clustering (in red), SpectACl (in green) and spectral clustering (in blue) algorithms applied on perturbed graphs for perturbations ranging from 0.01 to 0.5 and the associated 50% confidence intervals.

Some of the lowest performances of the self-tuning ℓ_1-spectral clustering can be explained by observing that this algorithm was developed in a sparse form, that is all nodes from a perturbed graph are not automatically clustered into components. In practice, this has huge consequences on the NMI score, which processes the non-classified nodes as wrongly-classified ones. When considering only the classified nodes, the NMI scores of the self-tuning ℓ_1-spectral clustering can be found in green, with, of course, better results.

More generally, the NMI scores are particularly sensitive to the number of estimated clusters. At the bottom of Figure 3, for each perturbation, the estimated number of clusters of both methods, which should be close to $k = 2$, the true number of clusters, can be visualized. It is clear that the further to 2 the estimation is, the smaller the associated NMI scores. The self-tuning version of the ℓ_1-spectral clustering was not optimized in this sense but it should be the key for an improvement and a stabilization of the performances.

5.2. Application to cancer data

This section is dedicated to the application of the ℓ_1-spectral clustering algorithm on a real kidney cancer data set from The Cancer Genome Atlas (TCGA) project. After describing the data (Section 5.2.1), results are presented in Section 5.2.2 and followed by a discussion (Section 5.2.3).

5.2.1. The kidney cancer data set

The Cancer Genome Atlas (TCGA) is an american project from the National Cancer Institute (NCI) and the National Human Genome Research Institute

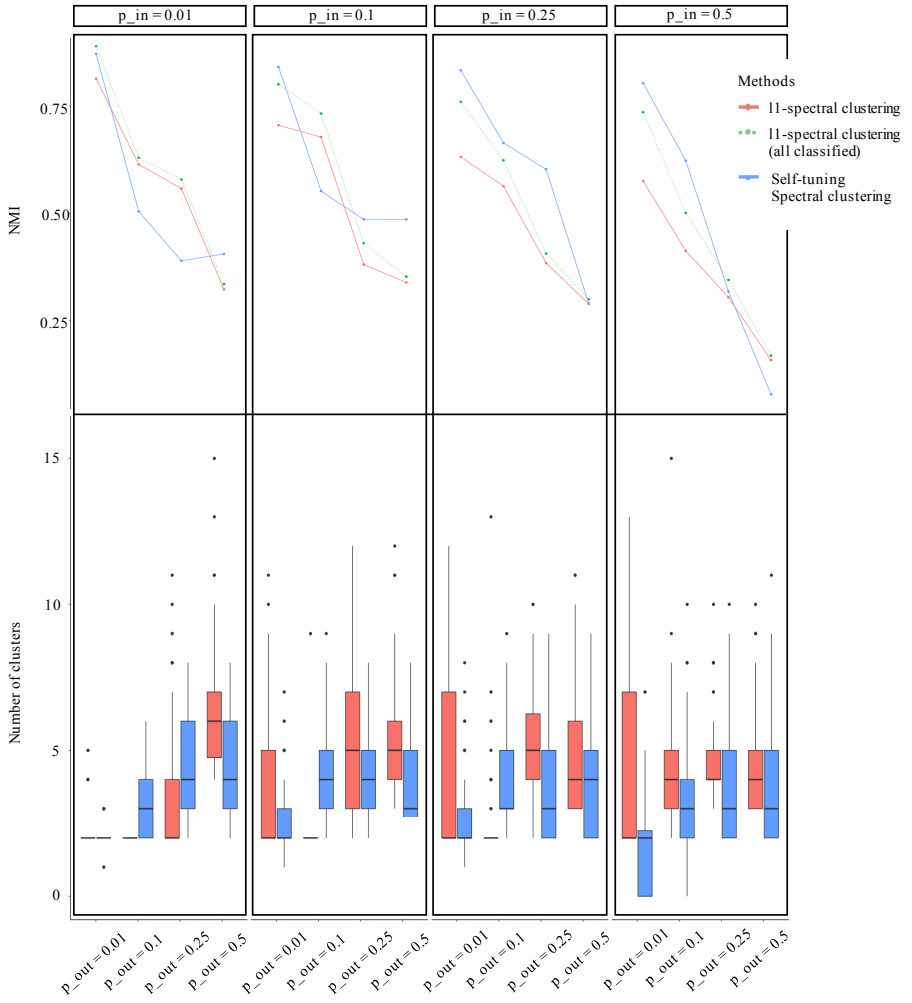

Figure 3: At the top, clustering results, in terms of NMI, of the self-tuning ℓ_1-spectral clustering (in red and in green, where only classified nodes are taken into accounts) and the Self-Tuning Spectral Clustering (in blue) algorithms applied on perturbed graphs for perturbations ranging from 0.01 to 0.5. At the bottom, the associated estimated number of clusters of both methods across the 100 perturbed versions of the graphs.

(NHGRI), which was launched fifteen years ago with the aim of characterizing genetic mutations responsible for cancer using genome sequencing and bioinformatics methods. Since then, millions of data have been produced and made publically available. In this work, we focused on KIdney Renal clear cell Carcinoma, abbreviated to KIRC thereafter. KIRC is one of the most common types of cancer, usually affecting people (mainly men) around 60 years old. Even if the chances of surgical cures are good, KIRC is hard to detect with no early

symptoms, which makes it even dangerous.

In this work, we extracted gene expression data for KIRC from the TCGA data portal http://gdac.broadinstitute.org/. These data were produced using RNA-sequencing for a total number of $16,123$ genes and 532 cancer patients. After preprocessing by log-transformating, quantile normalizing the arrays and filtering genes based on variance, we only kept 75% of them, i.e. $12,092$ genes.

5.2.2. ℓ_1-spectral clustering algorithm on kidney cancer data

Applying the ℓ_1-spectral clustering algorithm to cluster genes into components require the knowledge of an initial network that models the relationships between genes. The latter are usually represented through Gene Regulatory Networks (GRNs), which are directed graphs that connect genes based on regulations (activations/inhibitions). Here, we focused on co-regulated networks, a simplified version of GRNs, where no causality exists. Edges are thus undirected, representing co-regulations, or correlations in terms of expression between genes. To create such a network, we computed the correlation matrix, based on Pearson's correlation, between all pairs of genes and then thresholded the matrix by removing edges with correlation smaller in absolute value than 0.7. This network is made of 4,982 genes (see Figure 4 (a) for an overview of the network).

We then applied ℓ_1-spectral clustering algorithm on the adjacency matrix associated with the co-regulatory network described above. The 4,982 genes were clustered into 186 components, from size 2 to 986, with an averaged number of genes of 27. These components are represented in Figure 4 (b), with different colors.

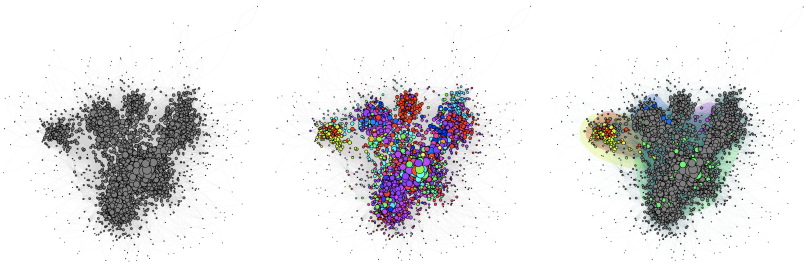

Figure 4: From left to right: (a) the co-regulatory network representing co-regulations between genes in KIRC, (b) the components discovered by applying the ℓ_1-spectral clustering algorithm and highlighted with different colors, (c) the five components we particularly focus on.

5.2.3. Clusters as hallmarks of kidney cancer

In this section, we investigate the biological hypotheses that can be deduced from the network. First, to assign biological meaning to each component of the network, we performed gene set enrichment analysis. To this aim, we used the databases GeneSetDB (Culhane et al., 2010) and MSigDB (Liberzon et al.,

2015), restricted to hallmark (H), curated (C2), GO (C5), oncogenic (C6) and immunologic signatures (C7) gene sets, which include the gene sets most relevant to cancer gene expression profiles. Enrichments were evaluated by performing hypergeometric tests, corrected for multiple testing using the FDR (Benjamini and Hochberg, 1995). Among the 186 identified components, four particularly drew our attention (see Figure 4 (c)). These components are described in details in the next paragraphs.

Transmembrane activity cluster. The first cluster we identified (Figure 4 (c), red cluster) is made of 43 genes. This cluster gathers genes involved in the same "transmembrane activity" pathway. Indeed, among the 43 genes of the component, 11 are members of the same group SLC of solute carriers transporters, which aims at facilitating the transport of substrates across membranes. This is confirmed by the gene set enrichment analysis we performed, with p-values ranging from 1.97×10^{-6} from 1.89×10^{-9}.

Epithelial-mesenchymal cluster. The second cluster (Figure 4 (c), purple cluster) of seven genes is highly enriched in Epithelial-Mesenchymal Transition (EMT) pathways, a natural process that converts epithelial cells into mesenchymal phenotypes and is often altered in cancers. This cluster includes the gene SERPINH1, a known EMT-related gene, which has been identified as a potential biomarker of kidney cancers (Qi et al., 2018).

T-cells associated cluster. The third cluster (Figure 4 (c), blue cluster) includes 38 genes, mostly enriched in T-cells and inflammatory response associated pathways. To confirm this, we tested the correlation of the cluster expression (defined as the averaged expression across all genes from the cluster) with CD4+ and CD8+ T-cells, encoded by the genes CD4, CD8A/B, which play a major role in cancer immunotherapy (Tay et al., 2020). With correlations ranging from 0.46 (CD8B) to 0.86 (CD4) and associated p-values smaller than $10e^{-16}$, we validate this relationship. In addition, Kawashima et al. (2020) recently reported that both CD4+ and CD8+ T-cells were up-regulated in the patients with kidney cancer of high grade. We obtained the same results when comparing the cluster expression with low (grades 1 and 2) and high (grades 3 and 4) kidney cancer grades: the higher the expression, the worst the grade (p-values for t-test of 0.009, see Figure 5).

Liver-signature cluster. In the last cluster (Figure 4 (c), yellow cluster), we found 96 genes, most of which belong to liver gene signatures, characterizing liver cancers. To a little extent, these genes are also associated with glutathione, an antioxidant that protects cells from important damages. Xiao and Meierhofer (2019) have recently shown that an increased level of glutathione is a hallmark of kidney cancer. To go further, we investigated whether the cluster expression could be used to predict survival in kidney cancer. To this aim, we used Cox proportional hazards modelling. Hazard ratios were used to report the direction of the survival effect and the Wald test was used to determine its significance.

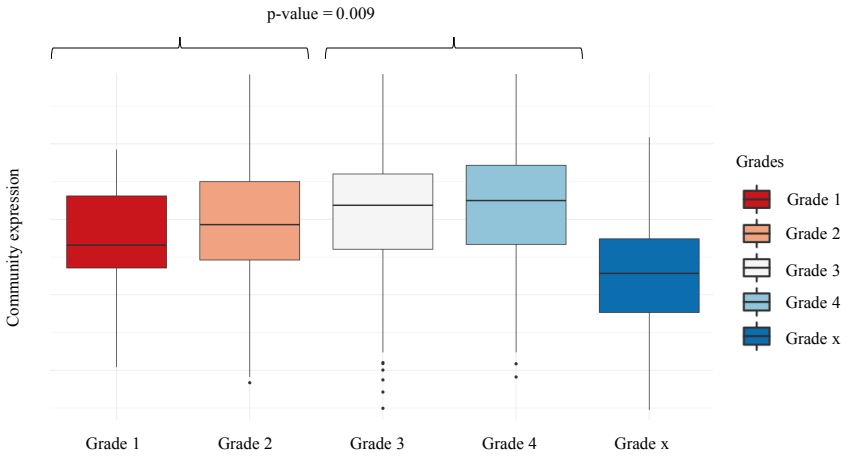

Figure 5: Boxplots representing the association between the cluster expression (averaged expression across all genes from the cluster) and kidney cancer grades, which range from 1 (low grade) to 4 (high grade), grade x indicating that the grade could not be evaluated.

As shown in Figure 6, high gene expression is significantly associated with good survivals (Hazard ratio of 0.66, p-value of 0.009). This indicates that this cluster may be used as a prognosis of kidney cancers.

6. Conclusion

In this paper, we proposed a new spectral clustering algorithm, called ℓ_1-spectral clustering, for detecting cluster structures in perturbed graphs. To tackle the noise robustness issue of the traditional spectral clustering, the k-means is removed and replaced by writting the indicators of the components as solutions of explicit ℓ_1-constrained minimization problems. The performances of this algorithm are highlighted through numerical experiments, with competitive results when compared to state-of-the-art. Nevertheless, many opportunities for further improvements can be considered. Firstly, from an algorithmic point of view, it would be interesting to better explore solutions for calibrating the optimal number of clusters and its representative elements. Secondly, future works include theoretical study of the eigenvectors stability, in order to validate the performances of the algorithm. A particular attention may be paid to the more general stochastic block model (SBM), where the edge probabilities depend on the community membership.

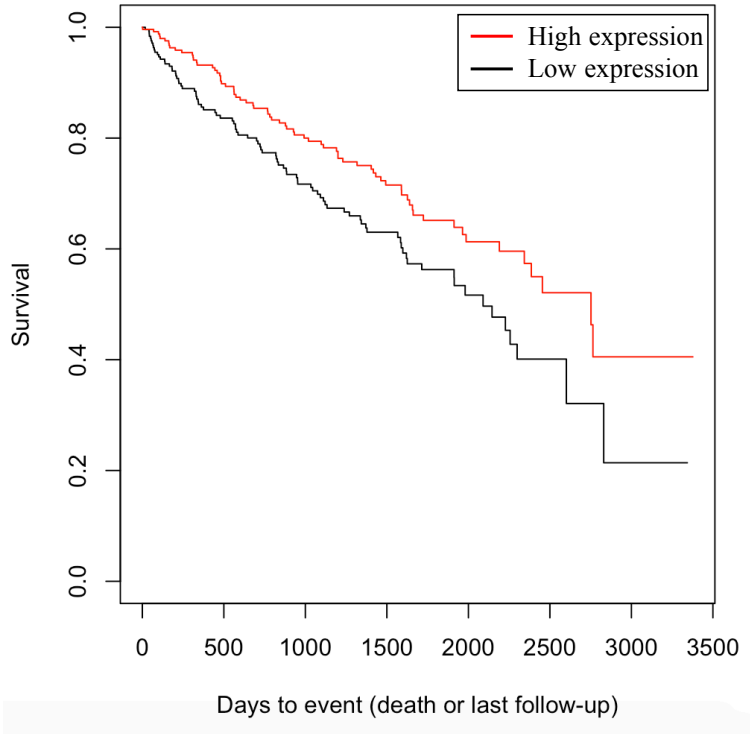

Figure 6: Kaplan-Meier curves representing the association between high/low cluster expression and survival.

References

Abbe, E., 2017. Community detection and stochastic block models: recent developments. The Journal of Machine Learning Research 18, 6446–6531.

Akyildiz, I., Su, W., Sankarasubramaniam, Y., Cayirci, E., 2002. Wireless sensor networks: a survey. Computer Networks 38, 393 – 422. doi:https://doi.org/10.1016/S1389-1286(01)00302-4.

Amini, A., Chen, A., Bickel, P., Levina, E., 2013. Pseudo-likelihood methods for community detection in large sparse networks. Annals of Statistics 41, 2097–2122.

Benjamini, Y., Hochberg, Y., 1995. Controlling the false discovery rate: a practical and powerful approach to multiple testing. Journal of the Royal Statistical Society Series B (Statistical Methodology) 57, 289–300.

Bojchevski, A., Matkovic, Y., Günnemann, S., 2017. Robust spectral clustering for noisy data: Modeling sparse corruptions improves latent embeddings, in:

KDD17: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 737–746.

Culhane, A.C., Schwarz, l., Sultana, R., Picard, K.C., Picard, S.C., Lu, T.H., Franklin, K.R., French, S.J., Papenhausen, G., Corell, M., Quackenbush, J., 2010. GeneSigDB, a curated database of gene expression signatures. Nucleic Acids Research 38, D716–D725.

Davidson, E., Levin, M., 2005. Gene regulatory networks. Proceedings of the National Academy of Sciences 102, 4935–4935. doi:10.1073/pnas.0502024102, arXiv:https://www.pnas.org/content/102/14/4935.full.pdf.

Ester, M., Kriegel, H., Sander, J., Xu, X., 1996. A density- based algorithm for discovering clusters in large spatial databases with noise, in: KDD96: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, pp. 226–231.

Girvan, M., Newman, E., 2002. Community structure in social and biology networks, in: Proceedings of the national academy of sciences, pp. 7821–7826.

Hagen, L., Kahng, A., 1992. New spectral methods for ratio cut partitioning and clustering. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 11, 1074–1085.

Handcock, M., Gile, K., 2010. Modeling social networks form sampled data. The Annals of Applied Statistics 4, 5–25.

Hastie, T., Tibshirani, R., Friedman, J., 2001. The Elements of Statistical Learning. Springer Series in Statistics, Springer New Yok Inc.

Hendrickson, B., Leland, R., 1995. An improved spectral graph partitioning algorithm for mapping parallel computations. SIAM Journal on Scientific Computing 16, 452–469.

Hess, S., Duivesteijn, W., Honysz, P., Morik, K., 2019. The SpectACl of non convex clustering: a spectral approach to density-based clustering, in: Proceedings of the AAAI conference on artificial intelligence, pp. 3788–3795.

Jeong, H., B. Tombor, R.A., Oltvai, Z., Barabasi, A., 2000. The large-scale organization of metabolic networks. Nature 407, 651–654.

Joseph, A., Yu, B., 2016. Impact of regularization on spectral clustering. The Annals of Statistics 44, 1765–1791.

Kawashima, A., Kanazawa, T., Kidani, Y., Yoshida, T., Hirata, M., Nishida, K., Nojima, S., Yamamoto, Y., Kato, T., Hatano, K., Ujike, T., Nagahara, A., Fujita, K., Moritomo-Okazawa, A., Iwahori, K., Uemura, M., Imamura, R., Ohkura, N., Morii, E., Sakaguchi, S., Wada, H., Nonomura, N., 2020. Tumour grade significantly correlates with total dysfunction of tumour tissue-infiltrating lymphocytes in renal cell carcinoma. Scientific Reports 10, 6220.

Lara, N.D., Bonald, T., 2020. Spectral embedding of regularized block models URL: https://arxiv.org/abs/1912.10903.

Li, X., Kao, B., Zaochung, R., Dawei, Y., 2019. Spectral clustering in heterogeneous information networks, in: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 4221–4228.

Liberzon, A., Birger, C., Thorvaldsdottir, H., Ghandi, M., Mesirov, J.P., Tamayo, P., 2015. The molecular signatures database (MSigDB) hallmark gene set collection. Cell systems 1, 417–425.

Luxburg, U., 2007. A tutorial on spectral clustering. Statistics and Computing 17, 395–416.

MacQueen, B., 1967. Some methods for classification and analysis of multivariate observations, in: Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297.

Newman, E., Girvan, M., 2004. Finding and evaluating community structure in networks. Physical review E 69, 026–113.

Ng, A., Jordan, M., Weiss, Y., 2002. On spectral clustering: Analysis and an algorithm. In Advances in neural information processing systems , 849–856.

Peche, S., Perchet, V., 2020. Robustness of community detection to random geometric perturbations, in: Proceedings of the 34th Conference on Neural Information Processing Systems.

Pelleg, D., Baras, D., 2007. K-means with large and noisy constraint sets, in: Machine Learning: ECML 2007, Springer Berlin Heidelberg. pp. 674–682.

Pothen, A., 1997. Graph partitioning algorithms with applications to scientific computing. Parallel Numerical Algorithms 4, 323–368.

Qi, Y., Zhang, Y., Peng, Z., Wang, L., Wang, K., Feng, D., He, J., Zheng, J., 2018. SERPINH1 overexpression in clear cell renal cell carcinoma: association with poor clinical outcome and its potential as a novel prognostic marker. Journal of Cellular Molecular Medicine 22, 1224–1235.

Shi, J., Malik, J., 2000. Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 22, 888–905.

Smith, S., 1997. The integration of communications networks in the intelligent building. Automation in Construction 6, 511 – 527. doi:https://doi.org/10.1016/S0926-5805(97)00028-9.

Stephan, L., Massoulié, L., 2019. Robustness of spectral methods for community detection, in: Beygelzimer, A., Hsu, D. (Eds.), Proceedings of the Thirty-Second Conference on Learning Theory, PMLR, Phoenix, USA. pp. 2831–2860.

Tang, W., Khoshgoftaar, T.M., 2004. Noise identification with the k-means algorithm, in: 16th IEEE International Conference on Tools with Artificial Intelligence, pp. 373–378.

Tay, R., Richardson, E., Toh, H., 2020. Revisiting the role of CD4+ T cells in cancer immunotherapy - new insights into old paradigms. Cancer Gene Therapy , https://doi.org/10.1038/s41417–020–0183–x.

Tibshirani, R., 1996. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society. Series B. Methodological 58, 267–288.

Tibshirani, R., Walther, G., Hastie, T., 2001. Estimating the number of clusters in a data set via the gap statistic. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 63, 411–423.

Wang, X., Davidson, I., 2010. Flexible constrained spectral clustering, in: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Association for Computing Machinery. pp. 563–572.

Xiao, Y., Meierhofer, D., 2019. Glutathione metabolism in renal cell carcinoma progression and implications for therapies. International Journal of Molecular Sciences 20, 3672.

Zelnik-Manor, L., Perona, P., 2005. Self-tuning spectral clustering, in: Advances in Neural Information Processing Systems, pp. 1601–1608.

Zhang, Y., Rohe, K., 2018. Understanding regularized spectral clustering via graph conductance. In Advances in Neural Information Processing Systems , 10631–10640.