l1-spectral clustering algorithm: a spectral clustering method using l1-regularization

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

ℓ_1-spectral clustering algorithm: a spectral clustering method using ℓ_1-regularization

Camille Champion · Magali Champion · Mélanie Blazère · Rémy Burcelin · Jean-Michel Loubes

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. In this paper, we present a variant of the spectral clustering algorithm, called ℓ_1-spectral clustering, based on Lasso regularization and adapted to perturbed graph models. Contrary to the spectral clustering, this procedure does not require the use of the k-means: it detects the hidden natural cluster structure of the graph by promoting sparse eigenbases solutions of specific ℓ_1-minimization problems. 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

C. Champion
Université Paris-Saclay, INRAE, MGP, Jouy-en-Josas, France
E-mail: camille.champion@inrae.fr

M. Champion
Université de Paris, CNRS, MAP5 UMR8145, Paris, France
E-mail: magali.champion@u-paris.fr

M. Blazère
Institut de Mathématiques de Toulouse, UMR 5219, Université de Toulouse, CNRS, France

R. Burcelin
Université Paul Sabatier (UPS), UMR1297, Institut des Maladies Métaboliques et Cardiovasculaires, INSERM, France

J.M. Loubes
Institut de Mathématiques de Toulouse, UMR 5219, Université de Toulouse, CNRS, France

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, 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 the present 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 using 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 their 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. As presented in Luxburg (2007), a large number of extensions of the original spectral clustering algorithm 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 paper, we develop an alternative method of the spectral clustering, called ℓ_1-spectral clustering algorithm and based on Lasso regularization (Tibshirani et al., 2001). Note that research papers have explored regularized spectral clustering to robustly identify clusters in large networks. Although Zhang and Rohe (2018) and Joseph and Yu (2016) show the effect of regularization on spectral clustering through graph conductance and respectively through SBM. Equally, Lara and Bonald (2020), shows on a simple block model that the spectral regularization separates the underlying blocks of the graph. 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 $V=\{1,\dots,n\}$ 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 $(d_1,\dots,d_n)$ on its diagonal and zero elsewhere:*

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

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 $C_1,\dots,C_k$ are a k-partition of the set V of vertices if the three following conditions hold:*

1. *they are non-empty: $\forall i\in [\![1,k]\!], C_i\neq\emptyset$,*
2. *they are pairwise disjoints: $\forall (i,j)\in [\![1,k]\!]^2, C_i\cap C_j=\emptyset$,*
3. *their union forms the set of all vertices: $\overset{k}{\underset{i=1}{\cup}} C_i = V$.*

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

$$(\mathbf{1}_{C_i})_j=\begin{cases}1 & \textit{if vertex } j \textit{ belongs to } C_i,\\ 0 & \textit{otherwise.}\end{cases}$$

In the present paper, we assume that the graph $\mathcal{G}$ is the union of k complete graphs, whose set of vertices $C_1,\dots,C_k$ forms 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:

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

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 (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 noisy settings, 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

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 noisy 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:

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

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, by definition of L, these indicators correspond this time to the k eigenvectors $v_{n-k+1}, \dots, v_n$, 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 of A.

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 $(\mathbf{1}_{C_j})_{1 \le j \le n} \in \mathcal{V}_k$, 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 $c_1 \le ... \le c_k$ 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:

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

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 $(\mathcal{P}_0^i)_{2 \le i \le k}$ 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 $\|v\|_1 = \sum_{1 \le j \le n} |v_j|$.

In the next section, we show that the solutions of the ℓ_0-optimization problems remain 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 $(\mathbf{1}_{C_j})_{1 \le j \le n} \in \mathcal{V}_k$, 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 $j \in \{1, \dots, k\}$ 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 $c_1 \le ... \le c_k$ the sizes of the k connected components. The solution of $(\mathcal{P}_1)$, which minimizes Equation (3), is thus given by setting $\alpha = (\alpha_1, 0, \dots, 0)$ 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 constraints:

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:*

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

The minimization problem

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

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 a perturbed graph (see Algorithm 2 below). It is available as an `R`-package on CRAN at `https://cran.r-project.org/web/packages/l1spectral`. In the next paragraphs, details about the algorithm and parameters setting are given.

Algorithm 2 ℓ_1-spectral clustering algorithm

```
1: 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.
2: Perform the spectral decomposition of $A$, sort the eigenvalues by increasing order and store the associated eigenvectors: $V := (v_1, ..., v_n)$.
3: for $j = 1$ to $\hat{k}$ do
4:     Define $U_{\hat{k},j}$ as the matrix that contains the $n-\hat{k}-j+1$ first columns of $V$:
           $U_{\hat{k},j} := (v_1, ..., v_{n-\hat{k}-j+1})$.
5:     Split $U_{\hat{k},j}$ into two parts:
           $\begin{cases} w^T := U_{\hat{k},j}^{i_j} & \text{the } i_j\text{-th row of } U_{\hat{k},j}, \\ W^T := U_{\hat{k},j}^{-i_j} & \text{the other rows of } U_{\hat{k},j}. \end{cases}$
6:     Solve the $\ell_1$-minimization problem $(\tilde{\mathcal{P}}_1)$:
           $v^* := \underset{\substack{v\in\mathbb{R}^{n-1}\\ Wv=-w}}{\arg\min}\ \|v\|_1.$
7:     Recover the indicator of the $j$-th component:
           $\hat{\mathbf{1}}_{C_j} = (v_1^*, \dots, v_{i_j-1}^*, 1, v_{i_j}^*, \dots, v_n^*)$.
8:     Update $v_j$ in $V$: $v_j \leftarrow \hat{\mathbf{1}}_{C_j}$.
9:     Perform Gram-Schmidt orthogonalization on $V$ to ensure orthogonality between $v_j$ and the rest of the columns of $V$:
           $V \leftarrow \text{Gram-Schmidt}(V)$.
10: end for
11: Output: $(\hat{\mathbf{1}}_{C_j})_{1\le j\le\hat{k}}$ the indicators of the $\hat{k}$ connected components.
```

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.

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. 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, wrongly adding few nodes. 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, especially when the perturbation grows. To avoid this, at each step of the algorithm, we check whether the nodes with the k highest scores are clustered together. If so, they are removed from the list of potential representative elements and the algorithm is re-run 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 kinds of data sets. 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 breast cancer tissues, we demonstrate the ability of our algorithm to discover relevant groups of patients that characterize breast cancer subtypes.

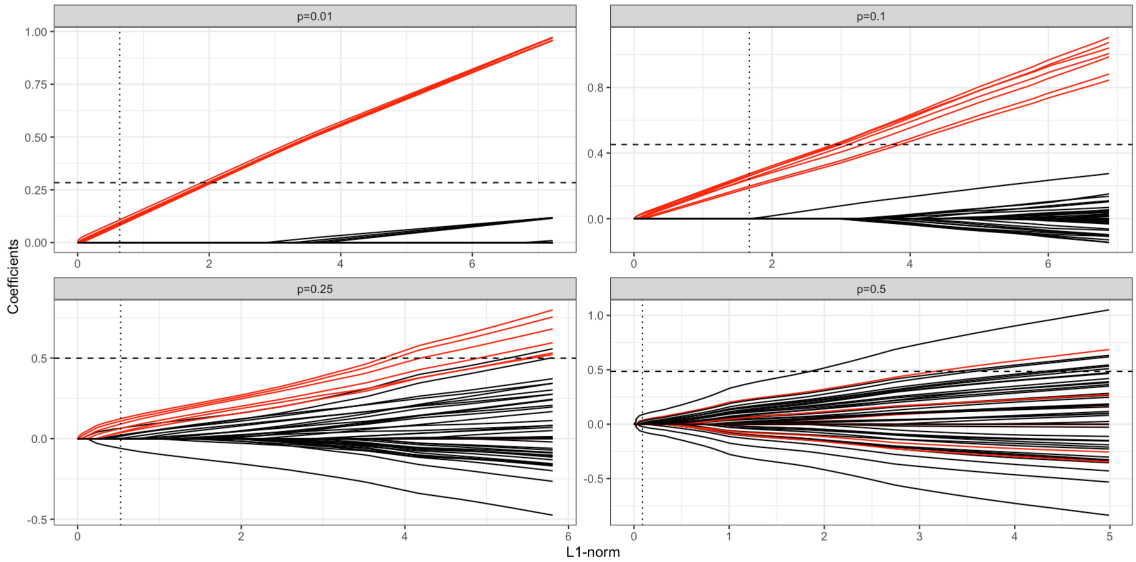


Fig. 1: Evolution of the coefficients of v, solution of $(\mathcal{P}_{\text{Lasso}})$, with respect to $\|v\|_1$ for different perturbations of the ideal graph (from top left to bottom right $p = 1\%$, 10%, 25% and 50%). Red lines correspond to the coefficients belonging to the component we aim at recovering, 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.

5.1 Application to toy data sets

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 paragraph.

Simulated data set
We generated random ideal graphs for a given number of nodes n ($n = 50$, 100, 500 and $1,000$) and a given number of connected components k depending on n ($k/n = 1\%$, 2%, 10% and 20%). 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 100 perturbed versions of the same graphs.

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 is assumed to be known, and the self-tuned one, for which it is directly extracted from the graph (see Section 4.3). Note that for both versions of the algorithm, the representative elements are selected as explained in Section 4.4. 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 when the number of samples is large enough (greater than 50) or leave-one-out cross-validation otherwise, on a grid of 100 values ranging from 0 to 1.

Comparison with state-of-the-art
We compare both versions of the ℓ_1-spectral clustering with two types of graph-based clustering algorithms: (i) non self-tuned algorithms, which require the number of clusters as input, and (ii) self-tuned algorithms, which, by contrast, automatically detect the optimal number of clusters. The first category of clustering methods we compare with include the original spectral clustering, presented in Algorithm 1 and implemented in the function `specc` of the `R`-package `kernlab` and a regularized version of the spectral clustering from Qin and Rohe (2013), which allows more flexibility of the nodes degree and is available in the `R`-package `greed` (function `spectral`).

For the self-tuned methods, we first choose to compare our ℓ_1-spectral clustering algorithm with the self-tuned version of the spectral clustering from Zelnik-Manor and Perona (2005), which improves the original one by removing the final post-processing step (k-means) and carefully analyzing the eigenvectors' structure to infer the number of clusters. The associated Python code is available on Github at `https://github.com/w00L/STSC`. We also compare our results with the hybrid algorithm of Côme et al. (2021), implemented in the function `greed` of the `R`-package `greed`, which uses a genetic algorithm to maximize the integrated classification likelihood and to find the best partition of the graph. We finally run the Markov clustering Algorithm, developped by van Dongen (2000) in the context of clustering bioinformatics data and available in the function `mcl` of the `R`-package `MCL`. The latter finds natural cluster structures by performing random walks calculated using Markov chains upon the graph.

Performance metrics
Performances are measured by comparing the learned components with the true ones, which are known in the context of simulated data. Among the large number of existing scores, we focus on the Adjusted Mutual Information (AMI) score, a corrected for chance version of the mutual information score, for its ability to compare clusters that could be of different sizes. The closer to 1 the AMI score, the better the classification. We also compare the percentage of missclassified nodes for the non self-tuned algorithms, where the true number of clusters is available, and the estimated number of clusters for the self-tuned versions of the algorithms. We finally report the computational times, obtained after running these algorithms on one core of an Intel Xeon E5645 2.40GHz processor with 66Go of RAM.

5.1.2 Effect of the dimension and cluster sizes on perturbed graphs

First, we aim at exploring the effect of the dimension and cluster sizes on the performances of the self-tuned version of the ℓ_1-spectral clustering algorithm. For n ranging from 50 to $1,000$ and k/n from 1% to 20%, results, in terms of AMI scores and number of estimated clusters are summarized in Table 1 (a) and (b). Note that all results are averaged over the 100 replicates of the perturbed graphs.

First of all, one can obviously note that, for small perturbations ($p_{in}, p_{out} < 0.25$), the ℓ_1-spectral clustering works quite well. However, increasing the perturbations makes the clustering problem more complex to solve and the results' quality lower. This phenomenon is even more significant while the dimension increases (from top left to bottom right for each value of n and at k/n fixed). For perturbations of 0.5 (last line), the AMI scores do not barely exceed 0.3, which means that the ℓ_1-spectral clustering algorithm almost fails to recover the components. However, we must keep in mind that imposing such a perturbation 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%. One can also note the huge impact of the true number of clusters k on the performances, which become really poor for large k (see for example $k = 50$, 100, 200). In this case, Table 1 (b) indicates that the number of clusters is hardly estimated. Imposing a graph structure of 2 clusters ($n = 100$ and $k/n = 2\%$) also leads to weak results, with AMI scores that do not exceed 0.6, a large quantity of nodes being forgotten by the ℓ_1-spectral clustering. In contrast, a good trade-off between k and n (e.g. $k/n = 1\%$ and, at a little extent 2%) provides excellent results, even for highly perturbed graphs.

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 evaluate its performances in comparison with the algorithms described in Section 5.1.1. when clustering different perturbed versions of a graph made of $n = 100$ and 500 nodes and a fixed number of clusters of 10. For each perturbation, we generated 100 graphs and computed the clustering performances using the AMI score. Results can be visualized in Figure 2.

For both values of n ($n = 100$ and 500), the performances of the three non self-tuned methods are very similar: quite good for small perturbations while adversely affected by larger perturbations imposed on the graphs. However, looking a little bit deeper, the percentages of missclassified nodes, indicated by red diamonds in Figure 2, are almost always in favor of the ℓ_1-spectral clustering. This is a consequence of the ℓ_1-constrained optimization form of the algorithm: it does not cluster all the nodes, ensuring a smaller number of missclassifications, while reducing sometimes the associated AMI coefficients.

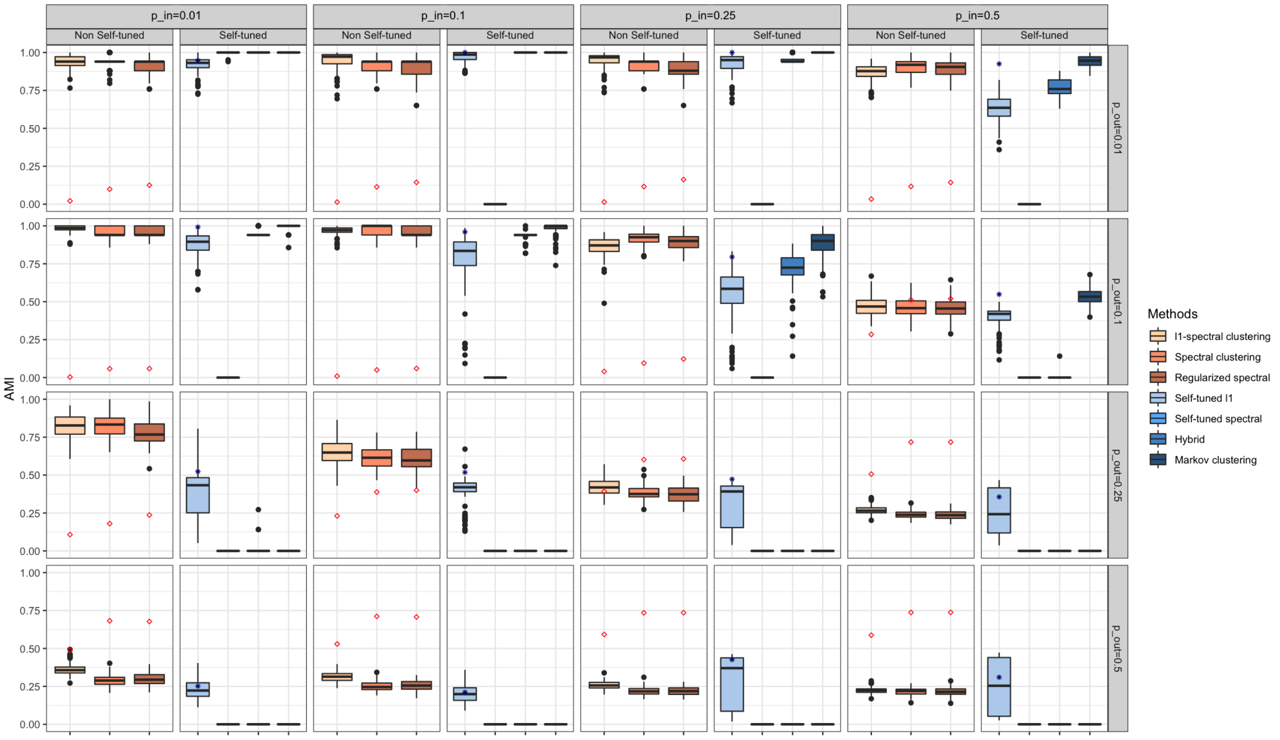
As regards the self-tuned methods, a huge discrepancy can be observed in terms of AMI: the self-tuned version of the spectral clustering seems to not work at all for perturbed graphs with more than 100 nodes. The three other algorithms work well until a certain level of perturbation is reached (0.25 or 0.5 depending on the number of nodes). For $n = 100$ and slightly perturbed graphs ($p_{out} < 0.25$), the performances of the self-tuned ℓ_1-spectral algorithm are lower than those of the hybrid and Markov ones (see Figure 2 (a)). However, removing the non-classified nodes from the classification results leads to the same performances, which are indicated by

Table 1: Performance results obtained after clustering perturbed graphs of different sizes using the self-tuned version of the ℓ_1-spectral clustering algorithm in terms of AMI scores (a) and number of estimated clusters (b). All results are averaged over 100 replicates.

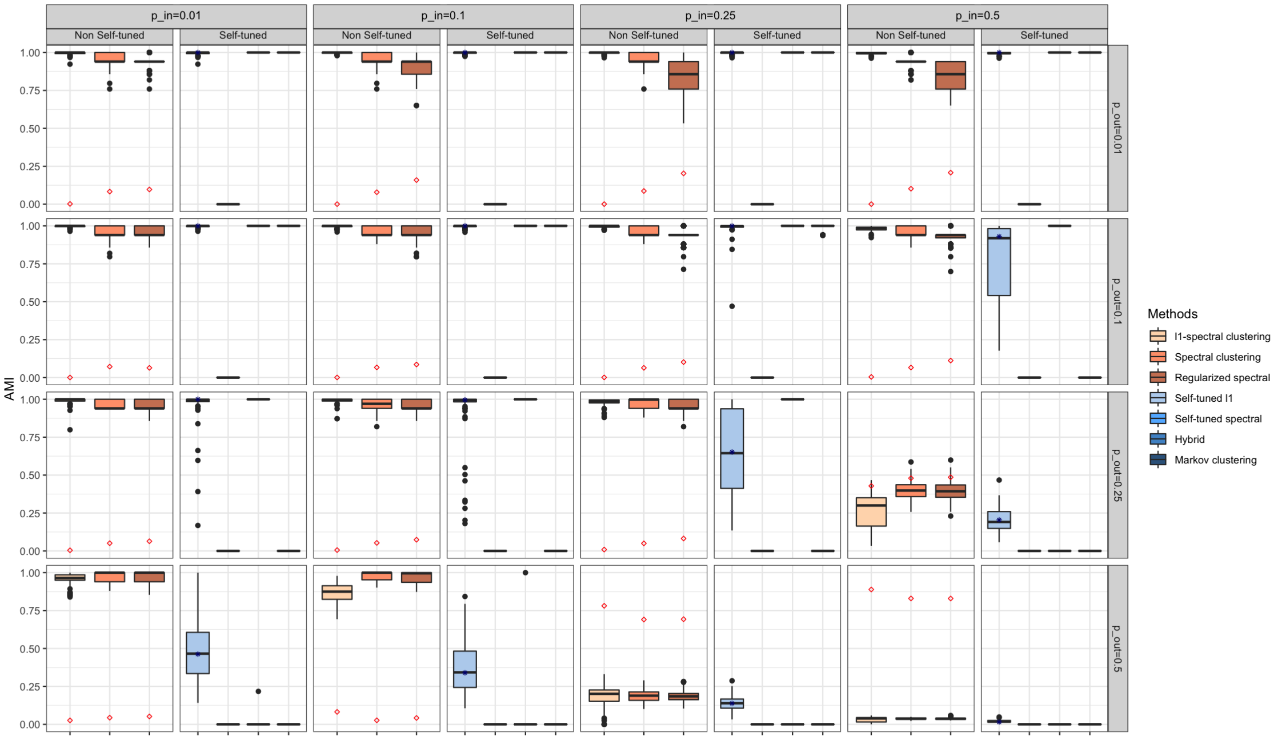
		n=50		n=100			n=500				n=1,000			
k/n		10%	20%	2%	10%	20%	1%	2%	10%	20%	1%	2%	10%	20%
k		5	10	2	10	20	5	10	50	100	10	20	100	200
p_{in}	p_{out}													
(a) 0.01	0.01	0.82	0.95	0.54	0.92	0.82	0.93	0.99	0.65	0.09	1.00	1.00	0.68	0.083
	0.1	0.81	0.53	0.51	0.88	0.40	0.95	1.00	0.19	0.19	1.00	0.99	0.09	0.10
	0.25	0.61	0.38	0.46	0.40	0.39	0.94	0.97	0.10	0.13	0.99	0.89	0.07	0.10
	0.5	0.38	0.28	0.15	0.23	0.33	0.91	0.51	0.09	0.11	0.72	0.17	0.06	0.09
0.1	0.01	0.80	0.84	0.65	0.97	0.63	0.94	1.00	0.65	0.09	1.00	1.00	0.05	0.08
	0.1	0.69	0.54	0.52	0.78	0.33	0.94	1.00	0.18	0.13	1.00	0.98	0.09	0.10
	0.25	0.54	0.37	0.23	0.40	0.34	0.92	0.94	0.10	0.13	1.00	0.24	0.07	0.10
	0.5	0.30	0.28	0.15	0.20	0.33	0.85	0.38	0.07	0.12	0.41	0.12	0.06	0.09
0.25	0.01	0.78	0.75	0.62	0.93	0.62	0.94	1.00	0.08	0.08	1.00	1.00	0.04	0.08
	0.1	0.64	0.50	0.56	0.55	0.29	0.95	0.99	0.12	0.13	1.00	0.65	0.08	0.09
	0.25	0.44	0.41	0.42	0.32	0.25	0.93	0.66	0.09	0.13	0.89	0.24	0.06	0.10
	0.5	0.30	0.28	0.29	0.28	0.21	0.53	0.14	0.07	0.12	0.24	0.04	0.05	0.09
0.5	0.01	0.69	0.50	0.58	0.63	0.58	0.95	0.99	0.07	0.08	1.00	0.94	0.07	0.09
	0.1	0.42	0.45	0.40	0.39	0.21	0.93	0.78	0.09	0.11	0.96	0.22	0.06	0.08
	0.25	0.31	0.44	0.28	0.26	0.20	0.86	0.21	0.08	0.12	0.34	0.06	0.05	0.09
	0.5	0.28	0.31	0.06	0.25	0.18	0.61	0.02	0.07	0.11	0.08	0.01	0.05	0.09
(b) 0.01	0.01	4.37	9.94	2.00	9.28	17.2	5.00	10.0	40.5	2.41	10.0	20.0	105.6	2.74
	0.1	4.96	5.58	2.00	10.0	9.89	5.00	10.0	15.7	38.1	10.0	19.7	4.09	4.11
	0.25	4.63	4.88	2.00	15.0	27.3	5.00	9.81	4.20	4.75	10.0	19.7	4.38	4.63
	0.5	4.35	4.97	6.21	4.73	4.84	5.01	6.06	4.67	4.65	7.82	4.30	4.44	4.36
0.1	0.01	4.45	9.47	2.00	10.0	19.0	5.00	10.0	42.0	2.53	10.0	20.0	2.50	2.73
	0.1	4.07	6.02	2.00	9.52	9.60	5.00	10.0	4.28	4.32	10.0	19.4	4.24	4.23
	0.25	4.25	5.41	5.35	16.8	21.8	5.00	9.56	4.53	4.73	10.0	4.42	4.45	4.40
	0.5	18.9	5.22	6.07	5.64	23.0	5.01	5.32	4.30	4.92	4.92	4.30	4.72	4.55
0.25	0.01	4.75	8.16	2.00	9.73	30.9	5.00	10.0	2.66	2.60	10.0	20.0	2.13	2.97
	0.1	4.18	6.65	2.00	11.0	6.91	5.00	9.96	4.24	4.32	10.0	12.3	4.06	4.21
	0.25	4.84	21.5	2.00	24.7	8.51	5.00	7.20	4.55	4.98	9.08	5.02	4.27	4.76
	0.5	23.6	4.85	6.19	33.1	6.93	2.96	4.92	4.59	5.00	4.38	4.16	4.34	4.32
0.5	0.01	5.29	14.3	2.00	11.3	62.4	5.00	10.0	2.51	3.27	10.0	18.5	3.19	3.34
	0.1	5.50	18.6	2.00	15.4	5.04	5.00	8.19	4.13	4.19	9.62	4.42	3.79	4.15
	0.25	23.9	23.7	8.20	23.6	5.33	4.36	5.09	4.73	4.80	4.92	4.26	4.40	4.75
	0.5	23.5	17.5	28.2	31.7	5.04	5.04	4.75	4.70	4.57	4.00	4.28	4.20	4.49

the blue stars. This is also confirmed by the number of estimated clusters (see Table 2), which is closed to the true number of clusters ($k = 10$). For $n = 500$, the clustering task is easier to solve: the results of the hybrid and Markov clustering algorithms become almost binary, with an AMI of 0 for $p_{out} = 0.5$ (0.25 for the Markov clustering) and 1 otherwise (see for instance Figure 2 (b)). As can be seen in Table 2, when the problem becomes too complex, both algorithms only create one group containing all nodes (averaged number of estimated clusters of 1.00), leading to an AMI of 0. In contrast, the self-tuned ℓ_1-spectral clustering algorithm seems to cope with such perturbed situations. Even if the results' variances are quite large, which indicates that the algorithm fails to recover the clustering structure of some graphs, the results appear encouraging.

In terms of computational time (Table 3), the ℓ_1-spectral clustering algorithm and its self-tuned version take an average of 8.75s/10.8s and 16.3/15.4s ($n = 100/500$) to run, which can be explained by the stabilization step described in Section 4.4 that implies to re-run the algorithm with more appropriate representative elements. The spectral clustering algorithm, its regularized version and the Markov clustering algorithm are fast, performing in less than 0.05s for $n = 100$.



(a) $n = 100$ and $k = 10$.



(b) $n = 500$ and $k = 10$.

Fig. 2: AMI scores obtained after clustering 100 versions of perturbed graphs with $n = 100$ (a) and 500 (b) nodes and $k = 10$ clusters for the seven compared methods. Red diamonds indicate the percentage of missclassified nodes for the non-self tuned methods only, whereas blue stars indicate the AMI scores of the self-tuned version of the ℓ_1-spectral clustering algorithm after removing the non-classified nodes.

Table 2: Estimated number of clusters after running the self-tuned algorithms on 100 versions of perturbed graphs with $n = 100$ and 500 nodes and $k = 10$ clusters. All results are averaged over the 100 replicates.

		n=100 & k=10				n=500 & k=10			
Methods		Self-tuned	Self-tuned	Hybrid	Markov	Self-tuned	Self-tuned	Hybrid	Markov
p_{in}	p_{out}	l_1	spectral		clustering	l_1	spectral		clustering
0.01	0.01	9.28	10.0	10.0	10.0	10.0	1.00	10.0	10.0
	0.1	10.0	1.00	9.14	9.92	10.0	1.00	10.0	10.0
	0.25	15.0	1.00	1.04	1.00	9.81	1.00	10.0	1.00
	0.5	4.73	1.00	1.00	1.00	6.06	1.00	1.02	1.00
0.1	0.01	10.0	1.00	10.0	10.0	10.0	1.00	10.0	10.0
	0.1	9.52	1.00	8.89	9.71	10.0	1.00	10.0	10.0
	0.25	16.8	1.00	1.00	1.00	9.56	1.00	10.0	1.00
	0.5	5.64	1.00	1.00	1.00	5.32	1.00	1.18	1.00
0.25	0.01	9.73	1.00	9.25	10.0	10.0	1.00	10.0	10.0
	0.1	11.0	1.00	6.14	9.94	9.96	1.00	10.0	9.86
	0.25	24.7	1.00	1.00	1.00	7.20	1.00	10.0	1.00
	0.5	33.1	1.00	1.00	1.00	4.92	1.00	1.00	1.00
0.5	0.01	11.3	1.00	6.30	12.3	10.0	1.00	10.0	10.0
	0.1	15.4	1.00	1.01	16.7	8.19	1.00	10.0	1.00
	0.25	23.6	1.00	1.00	1.00	5.09	1.00	1.00	1.00
	0.5	31.7	1.00	1.00	1.00	4.75	1.00	1.00	1.00

Table 3: Averaged computational time in seconds for running each of the 7 methods on 100 versions of perturbed graphs with $n = 100$ and 500 nodes and $k = 10$ clusters (perturbations varying from 0.1 to 0.5).

Methods	n=100 & k=10	n=500 & k=10
ℓ_1-spectral	8.75	10.8
Spectral clustering	0.041	1.96
Regularized spectral	0.0025	0.043
Self-tuned ℓ_1	16.3	15.4
Self-tuned spectral	13.7	16.2
Hybrid	3.46	16.2
Markov clustering	0.031	3.72

However, increasing the dimension multiplies by 20, 50 and 120 the computational times for the regularized spectral, the spectral and the Markov clustering algorithms respectively. The same holds at a little extent for the hybrid algorithm, whereas the self-tuned spectral, whose results are almost zero, and both versions of the ℓ_1-spectral clustering don't seem to be impacted.

5.2 Application to cancer data

This section is dedicated to the application of the ℓ_1-spectral clustering algorithm on a real breast cancer data set from The Cancer Genome Atlas 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 breast cancer data set

The Cancer Genome Atlas (TCGA) is a huge American project from the National Cancer Institute (NCI) and the National Human Genome Research Institute (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 BReast invasive CArcinoma, abbreviated to BRCA thereafter. BRCA is the most common diagnosed cancer among women, affecting around 2 millions of women worldwide each year. Risks of developing the disease include increasing age but also lifestyle (absence of physical activities, use of alcohol and smoking) and genetic predispositions. Over the years, the development of new treatments and prevention strategies have increased the survival rates to around 90% but scientific investigations are still needed to improve detection and surgical management of patients. In this work, we extracted gene expression data for BRCA from the TCGA data portal `http://gdac.broadinstitute.org/`. These data were produced using RNA-sequencing for a total number of $16,021$ genes and $1,081$ cancer patients. After preprocessing the arrays by log-transformation and quantile normalization and filtering genes based on variance, we only kept 75% of them, i.e. $12,015$ genes.

5.2.2 ℓ_1-spectral clustering algorithm on breast cancer data

Applying the ℓ_1-spectral clustering algorithm to cluster patients into subgroups requires the knowledge of an initial network that models the relationships between them. To create such a network, we computed the correlation matrix, based on Pearson's correlation, between all pairs of patients and then thresholded the matrix by removing edges with correlation smaller in absolute value than 0.7. We then applied the ℓ_1-spectral clustering algorithm on the adjacency matrix associated with the network described above. Among the 1,081 patients, 1,028 were clustered into 4 components, from size 160 to 407 (see Section 5.2.3 for a more detailed description of these groups). These components are represented in Figure 3 with different colors.

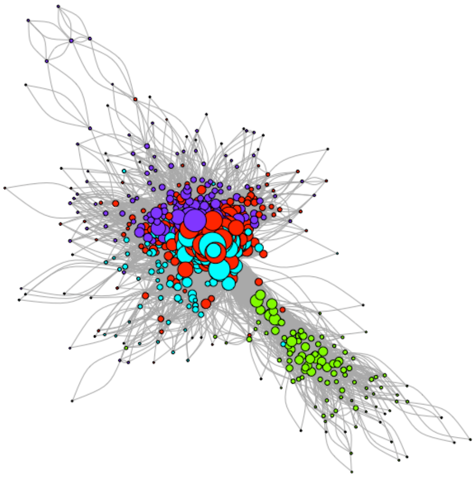


Fig. 3: The 4 components discovered by applying the ℓ_1-spectral clustering algorithm on the correlation network.

5.2.3 Clusters as subtypes of breast cancer

In this section, we evaluate the performances of the clustered network and investigate the biological hypotheses that can be deduced from it. Obviously, performances are rather hard to evaluate in the context of real data as the true cluster structure is unknown. One solution consists in measuring the mean silhouette coefficient rather than standard metrics such that the NMI, AMI or ARI, which requires the truth to be known. Here, we chose to compare to a well-defined breast cancer classification of patients (The Cancer Genome Atlas, 2012), which was performed using PAM50, a 50 gene expression assay based on microarray and quantitative real time that was developed by analyzing a set of 189 breast tumor samples (Parker et al., 2009). Patients are classified into 4 subtypes:

- Luminal A, the most common breast cancer subtype, enriched in hormone-receptor positive tumors with negative HER2 and low Ki67 (proliferating cell nuclear antigen) and is associated with good prognosis,
- Luminal B, similar to luminal A but with high levels of Ki67, a more aggressive phenotype and a slightly worse prognosis,
- Basal, also referred to as triple-negative, corresponding to negative hormone-receptors (both estrogen and progesterone) and negative HER2, the most aggressive breast cancer type,
- HER2-positive, characterized by high expression of HER2 and other genes associated with the HER2 pathway, high proliferation and more aggressive biological and clinical behavior.

Table 4 compares the 4 clusters identified using the ℓ_1-spectral clustering with the PAM50 classification. Note that a fifth subgroup, called normal, corresponding to data from sample tissues, was added to fit with the global data set. With an AMI of 0.3008, the results seem very poor but the p-value obtained by running a chi-squared independence test, which is under 10^{-16}, indicates its strength.

Table 4: Comparison between the PAM50 classification (in rows) and the ℓ_1-spectral clustering classification (in columns) of the BRCA patients.

	Cluster 1	Cluster 2	Cluster 3	Cluster 4
Lum A	289	148	99	0
Lum B	63	38	95	0
Basal	17	5	1	154
HER2	22	41	14	3
Normal	16	19	1	3

In more details, the Basal subgroup, which is associated with the worse prognosis, is almost perfectly recovered by cluster 4, with only 23 missclassified patients over 177. The three other clusters are combinations of Lum A, Lum B and HER2. To go a little bit further, we ran a Principal Component Analysis (PCA) on the gene expression data set to identify differential gene expression patterns within the four breast cancer patient profiles. Results in the form of a biplot are presented in Figure 4. As expected, cluster 4 is separated from clusters 1, 2 and particularly 3 by the first dimension, which is highly correlated with genes MKi67 (also referred to as Ki67 in the literature) and MCM2. These two genes are highly expressed in breast tumors of high histological grades (Yousef et al., 2017), confirming the existing link between cluster 4 and the Basal subgroup. Regarding the three other groups, cluster 3 slightly differs from the last two, separated by dimension 2, correlated with gene ETS1. ETS1 is a transcription factor that contributes to tumor angiogenesis and invasion of cancer cells (Fujimoto et al., 2002; Khatun et al., 2003). Even if its role on the development of breast cancer is still ambiguous, high expression of ETS1 is associated with the presence of metastases and a poor prognosis (Furlan et al., 2019; Kim et al., 2020). As can be seen in Figure 4, cluster 3 thus corresponds to patients with low expression of ETS1 and a better prognosis, mainly belonging to luminal A and B sutbypes.

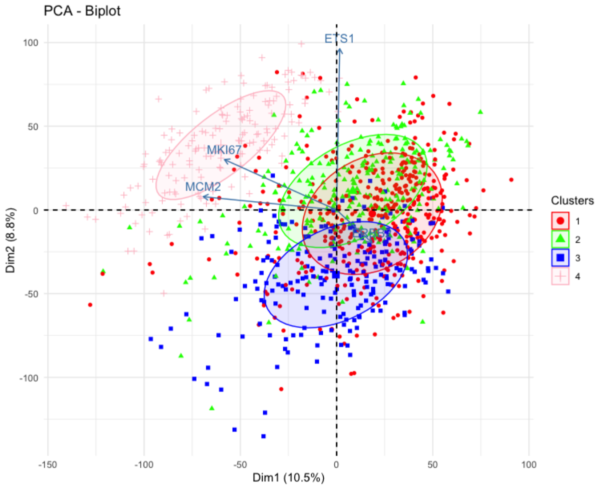


Fig. 4: Visualization of genomic variables by Principal Component Analysis (PCA) according to the four ℓ_1-spectral clustering clusters found.

Conclusion

In this paper, we propose 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 the algorithm are highlighted through numerical experiments, with competitive results when compared to the 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 global Stochastic Block Model (SBM), where the edge probabilities depend on the community membership.

References

Abbe E (2017) Community detection and stochastic block models: recent developments. J Mach Learn Res 18(1):6446–6531

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

Bojchevski A, Matkovic Y, Günnemann S (2017) Robust spectral clustering for noisy data: Modeling sparse corruptions improves latent embeddings. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '17, pp 737–746, DOI https://doi.org/10.1145/3097983.3098156

Côme E, Jouvin N, Latouche P, Bouveyron C (2021) Hierarchical clustering with discrete latent variable models and the integrated classification likelihood. Advances in Data Analysis and Classification 15(4):957–986, DOI 10.1007/s11634-021-00440-z

Davidson E, Levin M (2005) Gene regulatory networks. PNAS 102(14):4935–4935, DOI 10.1073/pnas.0502024102

van Dongen S (2000) Graph clustering by flow simulation. PhD thesis, University of Utrecht

Fujimoto J, Aoki I, Toyoki H, Khatun S, Tamaya T (2002) Clinical implications of expression of ETS-1 related to angiogenesis in uterine cervical cancers. Ann Oncol 13:1598–1604, DOI https://doi.org/10.1093/annonc/mdf248

Furlan A, Vercamer C, Heliot L, Wernert N, Desbiens X, Pourtier A (2019) ETS-1 drives breast cancer cell angiogenic potential and interactions between breast cancer and endothelial cells. Int J Oncol 54(1):29–40, DOI 10.3892/ijo.2018.4605

Girvan M, Newman E (2002) Community structure in social and biology networks. PNAS 99(12):7821–7826, DOI 10.1073/pnas.122653799

Hagen L, Kahng A (1992) New spectral methods for ratio cut partitioning and clustering. IEEE Transactions on CAD 11(9):1074–1085, DOI 10.1109/43.159993

Handcock M, Gile K (2010) Modeling social networks form sampled data. Ann Appl Stat 4(1):5–25, DOI 10.1214/08-AOAS221

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 J Sci Comput 16:452–469, DOI 10.1137/0916028

Jeong H, B Tombor RA, Oltvai Z, Barabasi A (2000) The large-scale organization of metabolic networks. Nature 407(6804):651–654, DOI 10.1038/35036627

Joseph A, Yu B (2016) Impact of regularization on spectral clustering. Ann stat 44(4):1765–1791

Khatun S, Fujimoto J, Toyoki H, Tamaya T (2003) Clinical implications of expression of ETS-1 in relation to angiogenesis in ovarian cancers. Cancer Sci 94:769–773

Kim G, Lee C, Verma R, Rudra D, Kim T, Kang K, Nam J, Kim Y, Im S, Kwon H (2020) ETS1 suppresses tumorigenesis of human breast cancer via trans-activation of canonical tumor suppressor genes. Front Oncol 10:642

Lara ND, Bonald T (2020) Spectral embedding of regularized block models (arXiv:1912.10903 [cs.LG]), 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, vol 33, pp 4221–4228, DOI 10.1609/aaai.v33i01.33014221

Luxburg U (2007) A tutorial on spectral clustering. Stat Comput 17(4):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, vol 1, pp 281–297

Newman E, Girvan M (2004) Finding and evaluating community structure in networks. Physical review E 69(2):026–113

Ng A, Jordan M, Weiss Y (2002) On spectral clustering: Analysis and an algorithm. In: Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic, NIPS'01, pp 849–856

Parker J, Mullins M, Cheang M, Leung S, Voduc D, Vickery T, Davies S, Fauron C, He X, Hu Z, Quackenbush J, Stijleman I, Palazzo J, Marron J, Nobel A, Mardis E, Nielsen T, Ellis M, Perou C, Bernard P (2009) Supervised risk predictor of breast cancer based on intrinsic subtypes. J Clin Oncol 27(8):1160–1167, DOI 10.1200/JCO.2008.18.1370

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, vol 4, Springer Netherlands, pp 323–368. DOI 10.1007/978-94-011-5412-3_12

Qin T, Rohe K (2013) Regularized spectral clustering under the degree-corrected stochastic blockmodel. In: Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2, NIPS'13, p 3120–3128

Shi J, Malik J (2000) Normalized cuts and image segmentation. IEEE PAMI 22(8):888–905, DOI 10.1109/34.868688

Smith S (1997) The integration of communications networks in the intelligent building. Automation in Construction 6(5):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, Proceedings of Machine Learning Research, vol 99, pp 2831–2860

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

The Cancer Genome Atlas (2012) Comprehensive molecular portraits of human breast tumours. Nature 490:61–70

Tibshirani R (1996) Regression shrinkage and selection via the lasso. J R Stat Series B Methodological 58(1):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(2):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

Yousef E, Furrer D, Laperriere D, Tahir M, Mader S, Diorio C, Gaboury L (2017) MCM2: An alternative to Ki-67 for measuring breast cancer cell proliferation. Mod Pathol 30(5):682–697, DOI 10.1038/modpathol.2016.231

Zelnik-Manor L, Perona P (2005) Self-tuning spectral clustering. In: Advances in Neural Information Processing Systems, vol 17, pp 1601–1608

Zhang Y, Rohe K (2018) Understanding regularized spectral clustering via graph conductance. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems, NIPS'18, pp 10631–10640