NNT : 2016SACLY000

THÈSE DE DOCTORAT

de

l'Université Paris-Saclay

École doctorale de mathématiques Hadamard (EDMH, ED 574)

Établissement d'inscription : École nationale de la statistique et de l'administration économique

Établissement d'accueil : *Laboratoire d'accueil :* ENSAE-X Centre d'économie, statistique et sociologie, UMR 9194 CNRS

Spécialité de doctorat : Mathématiques appliquées

Mehdi SEBBAR

Titre de la thèse

Date de soutenance : Jour Mois Année

Après avis des rapporteurs : Prénom NOM (Institution)
Prénom NOM (Institution)

Jury de soutenance :

	Prénom NOM	(Institution) Rapporteur
	Arnak DALALYAN	(ENSAE) Codirecteur de thèse
	Prénom NOM	(Institution) Examinateur
	Prénom NOM	(Institution) Examinateur
	Prénom NOM	(Institution) Codirecteur de thèse
	Prénom NOM	(Institution) Invité

Todo list

🟧 Note 17

🟧 Note 19

🟧 Note 27

🟧 Note 29

🟧 Note 29

🟧 Note 30

🟧 Note 30

🟧 Note 31

🟧 Note 31

🟧 Note 33

Remerciements

Contents

1 Introduction 11
1.1 Clustering and density estimation problem . . . 11
1.1.1 Centroid-Based Clustering: K-means . . . 12
1.1.2 Agglomerative Hierarchical Methods . . . 15
1.1.3 Spectral clustering . . . 16
1.1.4 Finding the number of clusters . . . 20
1.2 The Gaussian Mixture Model . . . 21
1.2.1 EM Algorithm . . . 23
1.2.2 K-means from the EM angle . . . 25
1.3 The curse of dimensionality . . . 25
1.4 Some contributions . . . 27
1.4.1 Graphical Lasso for Gaussian mixtures . . . 27
1.4.2 Estimating the number of clusters . . . 36
1.4.3 Sparse Weights Vector Estimation . . . 37
1.5 A word on density estimation . . . 41

Résume

Chapter 1

Introduction

Contents

1.1 Clustering and density estimation problem 11

1.2 The Gaussian Mixture Model 21

1.3 The curse of dimensionality 25

1.4 Some contributions 27

1.5 A word on density estimation 41

In this thesis, we study the unsupervised learning problem through the study of the clustering of high dimensional Gaussian mixtures and density estimation. In this chapter, we introduce the clustering problem in the first section and the Gaussian mixtures framework in the second. In the third section, we highlight the complexities inherent to the high dimension. Then we will discuss some of the work carried out during this thesis but has not been the subject of a completed work.

1.1 Clustering and density estimation problem

The goal of cluster analysis is to find groups in data so that each element within a group are similar compared to outside of the group. The literature is rich on this topic, with different approaches coming from statistics and computer science. A clustering problem has several dimensions, the input data can be a distance or similarity matrix, the similarity can be for instance a kernel chosen by expert knowledge. The input can also be a raw $N \times p$ data matrix with features. Another dimension for clustering methods is 'hard versus soft' assignment of points to clusters. In hard assignment, a point is assigned to cluster. In soft assignment a point is assigned to cluster with a confidence, generally a probability. This particularity is specific to probabilistic methods. A third dimension is flat versus hierarchical clustering. In flat clustering, the output is a partition of the dataset into disjoint clusters whereas in hierarchical clustering the output is a tree of nested clusters. We will give a glimpse on 4 well-known clustering techniques, K-means, Hierarchical clustering, Spectral clustering and the Gaussian mixtures model with Expectation-Maximization algorithm which will be our topic of main interest. The reader can refer to [Hennig et al., 2015] for an extensive review of cluster analysis.

1.1.1 Centroid-Based Clustering: K-means

K-means is a popular method of clustering which aims to partition the data into K clusters such that the within-cluster sum of squares is minimal. It has been introduced in signal theory for vector quantization by [MacQueen, 1967]. Given N points, $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ in $\mathbb{R}^p$, the goal of K-means is to find a set of centers $\mathcal{C} = \{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_K\}$ that minimize the following objective function:

$$\mathcal{L}_{k\text{-means}}(\boldsymbol{c}_1, \ldots, \boldsymbol{c}_K) = \sum_{i=1}^{N} \min_{\boldsymbol{c} \in \mathcal{C}} \|\boldsymbol{x}_i - \mathbf{c}\|^2. \tag{1.1}$$

Clearly this objective function is not convex and finding an exact solution of this problem is known to be NP-hard, even for 2-means [Dasgupta, 2008, Aloise et al., 2009]. As a matter of fact, for K and p fixed, the problem can be solved exactly in $O(n^{Kp})$ iterations [Inaba et al., 1994]. A simple and yet widely used approximation method to resolve the K-means minimization problem is the Lloyd's algorithm [Lloyd, 1982]. Today, because of its popularity, Lloyd's method is assimilated with the minimization problem of K-means (eq. (1.1)). A key element of this method is the Voronoi partitioning:

Definition 1. *(Voronoi Partition) Given n points in $\mathbb{R}^p$ $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$, K points $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_K$ and a distance d, a Voronoi partition of $\mathcal{D}$ consists on K disjoint clusters such that for $i \in [K]$, cluster i is the set of points satisfying $d(\boldsymbol{x}, \boldsymbol{c}_i) \leq d(\boldsymbol{x}, \boldsymbol{c}_j)$ for all $j \neq i$.*

The procedure consists to build a Voronoi partition of the data from K initial randomly chosen centers and iterate partitioning with the cell-means of the previous partition. The Llyod's procedure is described in Figure 1.2. The following lemma will help us to understand the convergence of the algorithm:

Lemma 1.1.1. *Consider a set $\mathcal{X} \subset \mathbb{R}^p$ and $\boldsymbol{\mu}$ its mean. For any $\mathbf{y} \in \mathbb{R}^p$, we have that*

$$\sum_{\boldsymbol{x} \in \mathcal{X}} d(\boldsymbol{x}, \mathbf{y})^2 = \sum_{\boldsymbol{x} \in \mathcal{X}} d(\boldsymbol{x}, \boldsymbol{\mu})^2 + |\mathcal{X}| d(\boldsymbol{\mu}, \mathbf{y})^2. \tag{1.2}$$

The reader can refer to Fact 5.1 of Hennig et al. [2015] for a simple proof. This lemma claims that, after a Voronoi partitioning, replacing a center by the mean of the cell can not increase the K-means cost. Unfortunately, Lloyd's algorithm tends to reach local optimums of the K-means objective. Hence, several runs of

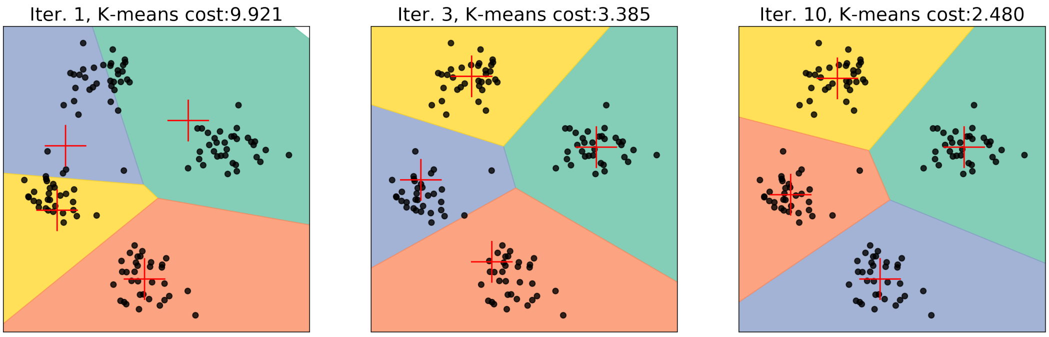


Figure 1.1: Llyod's algorithm with random initialization centers and final Voronoi partitions at different steps: with 1 iteration (left), 3 (middle) and 10 (right) iterations (the algorithm converged). K-means costs are given on top.

```
Input: N points $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n \in \mathbb{R}^p$ and the number of clusters $K$.
Output: Cluster centers $\widehat{\boldsymbol{c}}_1, \dots, \widehat{\boldsymbol{c}}_K$ and clusters assignments.
Init: Set $\mathcal{L}_{\text{old}} = \infty$. and chose $K$ seed points $\boldsymbol{c}_1, \dots, \boldsymbol{c}_K$. Compute the $K$-means cost $\mathcal{L}_{\text{curr}}$ given in eq. (1.1) with these points as centers.
while $\mathcal{L}_{\text{curr}} < \mathcal{L}_{\text{old}}$ do
    1: Compute the Voronoi partitioning of the data with $\boldsymbol{c}_1, \dots, \boldsymbol{c}_K$ as centers. Get $K$ clusters, $C_1, \dots, C_K$.
    2: On each clusters, compute the sample means $\widehat{\boldsymbol{c}}_1, \dots, \widehat{\boldsymbol{c}}_K$:
       $$\widehat{\boldsymbol{c}}_i = \frac{1}{|C_i|} \sum_{x_j \in C_i} x_j \qquad (1.3)$$
    3: Set $\mathcal{L}_{\text{old}} = \mathcal{L}_{\text{curr}}$ and compute the new $K$-means cost $\mathcal{L}_{\text{curr}}$ with $\widehat{\boldsymbol{c}}_1, \dots, \widehat{\boldsymbol{c}}_K$ as centers.
end while
```

Figure 1.2: K-means Lloyd's algorithm

the algorithm are necessary to insure an acceptable clustering. Note that Lloyd's algorithm has several drawbacks:

1. It is a hard-assignment method since it assigns points to clusters and does not reflect a level of uncertainty on the assignments such as a probability of belonging to a cluster.

2. The number of clusters has to be given, we will see some techniques to select the number of clusters in section 1.4.2.

3. The worst-case time complexity $T(n)$ is superpolynomial, $T(n) = 2^{\Omega(\sqrt{n})}$ iterations [Arthur and Vassilvitskii, 2006] (not bounded above by any polynomial). Fortunately, in practice it is observed that Lloyd's algorithm converges quickly to a local minimum.

4. If the initial centers are chosen randomly, the resulting K-means cost can be

made arbitrarily bad compared to the optimal clustering (see section 5.2 of [Hennig et al., 2015]). K-means++ [Arthur and Vassilvitskii, 2007] address this problem by choosing carefully the initializations centers in Lloyd's algorithm, see Figure 1.3 for the procedure, and showed that K-means++ is a $\log K$ approximation algorithm for the K-means objective, see Theorem 1.1.1.

Theorem 1.1.1. *[Arthur and Vassilvitskii, 2007] Let S be the set of centers output by the algorithm K-means++ and $\mathcal{L}(S)$ be the K-means cost of the clustering obtained using S as the centers. Then $\mathbb{E}[\mathcal{L}(\mathcal{S})] \leq O(\log(K))\mathcal{L}^*$, where $\mathcal{L}^*$ is the cost of the optimal K-means solution.*

5. K-means can not distinguish noise or select relevant features. This last point is particularly important in the case of high dimensional data, since it is generally accepted that the most relevant clusters lies in subspaces of much smaller dimension, we will discuss this phenomenon in section 1.3. An idea would be to adapt Lloyd's method to the weighted p^{th}-root of the Minkowski metric

$$d_{\boldsymbol{w}}(\boldsymbol{x}, \mathbf{y}) = \sum_{l=1}^{p} w_l |\boldsymbol{x}_l - \mathbf{y}_l|^p, \tag{1.4}$$

with $\boldsymbol{w}$ a weight vector updated at each iterations. A first method of weighted K-means has been introduced in [Makarenkov and Legendre, 2001] and further developed in [Zhexue Huang et al., 2007] (WK-Means) for the Euclidean norm. An extension to the Minkowski metric is proposed in [Cordeiro de Amorim and Mirkin, 2012] (MWK-Means) that outperforms K-means and WK-Means. Note that the use of a different metric has a profound impact on the implementation and running costs since the computation of Minkowski centers is not straightforward.

```
Input: N points $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n \in \mathbb{R}^p$ and the number of clusters $K$.
Init: Chose one center $\boldsymbol{c}_1$ uniformly at random among the data points and add it to the set $\mathcal{S}$.
for $j = 2$ to $K$ do
    Chose a point $\boldsymbol{x}$ with probability proportional to $\min_{\boldsymbol{c} \in \mathcal{S}} d(\boldsymbol{x}, \boldsymbol{c})^2$ and add it to $\mathcal{S}$.
end for
Proceed with $K$-means algorithm and the set $\mathcal{S}$ as initialization clusters.
```

Figure 1.3: K-means++ algorithm

The research on K-means is dense and several variants of this method has been developed. For instance K-medoids [Kaufman and Rousseeuw, 1990] uses points of the data as centers, Mini-batch K-means[Sculley, 2010] takes mini-batches of data to reduce significantly computational times without penalizing too much the K-means cost or clustering algorithms that enjoy strong theoretical guarantees on non-worst case scenarios using notion of stability[Ostrovsky et al., 2006]. The reader can refer to [Hennig et al., 2015] for further details on this topic.

1.1.2 Agglomerative Hierarchical Methods

In this section, we will present the Agglomerative Hierarchical clustering, a very popular method due to its simplicity and the resulting nested structure of clusters that it produces. The idea of Hierarchical clustering is to form a hierarchy of clusters (i.e. nested clusters) according to a merging rule which helps us to see how clusters are related to each other (a structure unavailable with K-means). There exists two types of hierarchical clustering: agglomerative and divisive. The first type consists to start from N sets, each containing one element of the dataset and merge sets iteratively into larger groups according to an agglomeration rule and a similarity, i.e. building a hierarchy, until finding only one cluster that contains the whole data. The similarity can be a Minkowski distance, the cosine similarity or other distance such as Hamming, Hellinger or Mahalanobis. The divisive procedure is the opposite of the agglomerative, starting from the whole dataset and splitting iteratively until obtaining N sets. Divisive methods are generally very expensive, with a complexity of $O(2^n)$ [Guénoche et al., 1991] , and are therefore not used in practice. Let us consider the agglomerative procedure and a metric d, a simple implementation is to build the dissimilarity matrix of the N original clusters $\{\boldsymbol{x}_1\}, \ldots, \{\boldsymbol{x}_N\}$ noted $S = (d_{ij} = d(i,j))_{i,j \in [N]^2}$ (which is symmetric) and consider the couple (i,j) such that d_{ij} is the smallest dissimilarity in S. We create a new cluster $i \cup j$, add it to the matrix S with the rule $d_{i \cup j,k} = \min\{d_{ik}, d_{jk}\}$ and remove the rows and columns of sets i and j in S. The iteration of this procedure leads to one final cluster containing all points in the dataset. This method is called 'Single-linkage' clustering [Graham and Hell, 1985] and a naive implementation is given in Figure 1.5 with a complexity of $O(n^3)$. Note that it can be optimized to $O(n^2)$[Murtagh and Contreras, 2012]. The hierarchy can be visualized via a binary tree called 'dendrogram' in Figure 1.4. This method has a severe drawback called 'chaining phenomenon' where clusters can be merged due to close points even if it contains other points very distant. A alternative method called 'Complete-linkage' clustering solves this problem by taking the maximum instead of the minimum in step 1 of the single-linkage algorithm in Figure 1.5. Similarly to Single-linkage method, the complexity of the naive method is $O(n^3)$ but can be optimized to $O(n^2)$. Another popular method worth mentioning for its use of cluster centers is the Ward's method[Jr., 1963] also called Ward's minimum variance method which consists to optimize an objective function, generally the sum of squares. Let us consider the merging cost of combining clusters A and B:

$$\begin{aligned}\Delta(A,B) &= \sum_{i\in A\cup B}\|\boldsymbol{x}_i-\boldsymbol{c}_{A\cup B}\|^2-\sum_{i\in A}\|\boldsymbol{x}_i-\boldsymbol{c}_A\|^2-\sum_{i\in B}\|\boldsymbol{x}_i-\boldsymbol{c}_B\|^2\\ &= \frac{n_An_B}{n_A+n_B}\|\boldsymbol{c}_A-\boldsymbol{c}_B\|^2,\end{aligned}$$

where $\boldsymbol{c}_i$ and n_i are the center of cluster i and its size respectively. This quantity is positive, hence the within-group variance increases when merging two clusters. Ward's method seek to minimize this growth. Alternatively, this leads to achieve the maximum between-cluster variance.

We can notice that agglomerative methods differ on the computation of dissimilarities following the agglomeration process (step 2 in fig. 1.5). Lance and Williams developed an updating formula [Lance and Williams, 1967] for these dissimilarities that generalize several agglomerative methods. The dissimilarity between a new merged cluster $i\cup j$ and cluster k is

$$d(i\cup j,k)=\alpha_i d(i,k)+\alpha_j d(j,k)+\beta d(i,j)+\gamma|d(i,k)-d(j,k)|, \qquad (1.5)$$

where the parameters $\alpha_i,\alpha_j,\beta,\gamma$ depend on the clustering criterion. For instance, the single-linkage method is recovered by setting $\alpha_i=\alpha_j=1/2$, $\beta=0$ and $\gamma=-1/2$, the complete-linkage method with $\alpha_i=\alpha_j=1/2$, $\beta=0$ and $\gamma=1/2$ and Ward's method can be expressed in this framework [Batagelj, 1988, Murtagh, 1985, Jambu, 1989] with $\alpha_i=(n_i+n_k)/(n_i+n_j+n_k)$, $\alpha_j=(n_j+n_k)/(n_i+n_j+n_k)$, $\beta=-n_k/(n_i+n_j+n_k)$ and $\gamma=0$. The reader can find parameters for other methods in Table 6.1 of [Hennig et al., 2015]. More efficient algorithms relies on 'Nearest Neighbor Chains'. The reader can refer to [Murtagh and Contreras, 2012] for a detailed review on Agglomerative hierarchical methods.

1.1.3 Spectral clustering

Recently, spectral clustering has become widely used thanks to its performance compared to traditional clustering techniques and it's computational attractiveness. One interesting feature of spectral clustering is that it does not make any assumptions on the form of the clusters contrary to K-means. This method of clustering relies deeply on the graph theory [Donath and Hoffman, 1973, Fiedler, 1973]. The reader can refer to [Luxburg, 2007] and [Spielman and Teng, 2007] for a study of the literature on this topic. Although several methods exists which all refer to the term 'Spectral clustering' we will address the simplest formulation of this method. Let us consider $\boldsymbol{x}_1,\dots,\boldsymbol{x}_N$, N points in $\mathbb{R}^p$ and a similarity measure $s_{ij}\geq 0$ between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. We can construct the similarity matrix $\boldsymbol{S}=(s_{ij})_{i,j\in[N^2]}$ which can be represented by a similarity graph $G=(V,E)$ where the vertices set $v_1,\dots,v_N$ correspond to the points $\boldsymbol{x}_1,\dots,\boldsymbol{x}_N$ and the edge between v_i and v_j exists if $s_{ij}\neq 0$ and hence has weight s_{ij}. Note that G is an undirected graph, i.e. $s_{ij}=s_{ji}$. The main

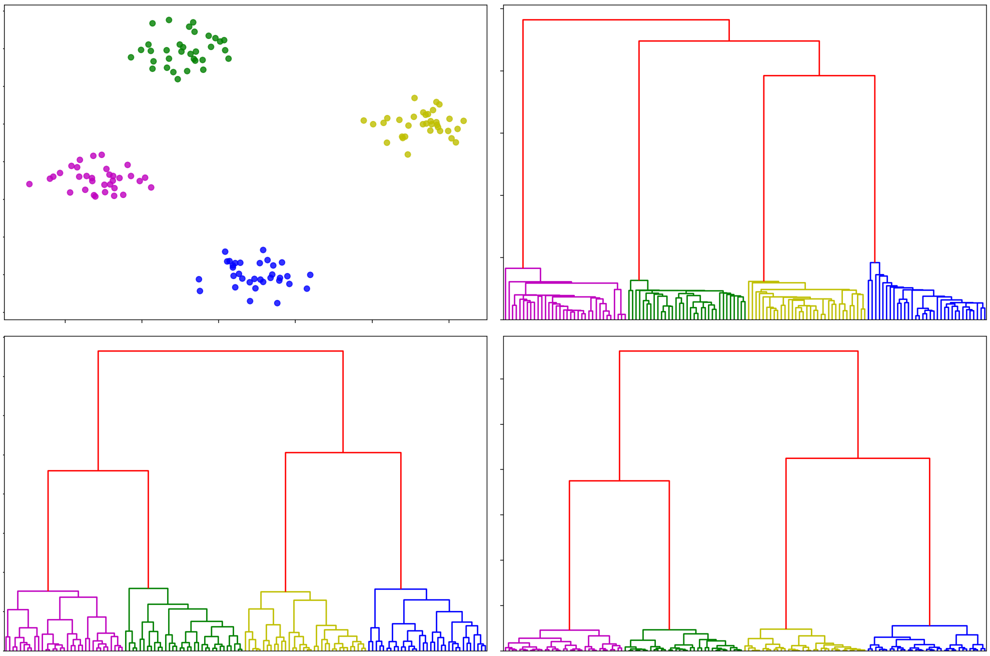


Figure 1.4: A dataset with 4 clusters (top-left) used with the Agglomerative hierarchical clustering and its corresponding dendrogram, single-link (top-right), complete-link (bottom-left) and Ward's method (bottom-right). A simple way for finding clusters would be to cut the dendrogram with a horizontal line from bottom to top until finding the number of clusters desired. Note the difficulty to select the 4 original clusters.

idea of spectral clustering is to find a partition of G with minimal cuts, that is find a partition such that the edges between different group has low weight and those within a group have high weight. This can be done by analyzing the spectrum of the Laplacian matrix $\boldsymbol{L}$ of $\boldsymbol{S}$ and a clustering such as K-means in a low-dimensional subspace spanned by relevant eigenvectors of $\boldsymbol{L}$. It is clear that a sparse graph G is interesting for such cutting problem. There exists several methods to sparsify $\boldsymbol{S}$:

Note: changer en W

K-nearest neighbor graphs: Modify the similarity matrix $\boldsymbol{S}$ by keeping for each nodes the k-nearest neighbors and set $s_{ij} = 0$ for the other vertices. We can make this graph undirected in different ways, see section 2.2 of [Luxburg, 2007].

ε-neighborhood graph: We connect nodes v_i and v_j if $d(\boldsymbol{x}_i, \boldsymbol{x}_j) \leq \varepsilon$, this graph is usually unweighted.

Fully connect graph: Used with a similarity function such as Gaussian $s_{ij} = \exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/2\sigma^2)$. σ has a similar role to ε in the ε-neighborhood graph.

We will note $\boldsymbol{W}$ the resulting weighted adjacency matrix. The reader can find more details and behavior of these different graph in section 8 of [Luxburg, 2007]. The most simple approach for spectral clustering is to consider the 'unnormalized graph

```
Input: A dissimilarity matrix $S$.
while at least 2 objects remain in S do
    1: Determine the smallest dissimilarity $d_{ij}$ in $S$.
    2: Let $m$ be the size of $S$, compute the dissimilarities for the new cluster $i \cup j$:
           $d_{i\cup j,k} = \min\{d_{ik}, d_{jk}\}, \quad k \in [m], m \neq i, j.$    (1.6)
    3: Add the dissimilarities of $i \cup j$ in $S$ and remove those of clusters $i$ and $j$.
end while
```

Figure 1.5: Simple single-linkage hierarchical clustering

Laplacian', $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$, where $\boldsymbol{D}$ is a diagonal matrix called the 'degree matrix' and the element i of its diagonal is the degree of the vertex v_i, $d_i = \sum_{j=1}^{N} s_{ij}$. An important property of this matrix is that its smallest eigenvalue is 0 and its corresponding eigenvector is the constant one vector (see proposition 1 of [Luxburg, 2007]). For the following, we say that $A \subset G$ is connected if any two vertices in A can be joined with a path where the intermediate vertices lies in A. A is a connected component if it is connected and there are no connections between A and $\bar{A}$. An important result for spectral clustering is the following proposition:

Proposition 1 (Number of connected components, proposition 2 in [Luxburg, 2007])**.** *The multiplicity k of the eigenvalue 0 of $\mathbf{L}$ is the number of connected components $A_1, \dots, A_k$ in the graph G. The eigenspace of eigenvalue 0 is spanned by the indicator vectors $1_{A_1}, \dots, 1_{A_k}$ of these components.*

The most simple implementation of the spectral clustering is given in Figure 1.6 Two other types of Laplacian matrices are used in the literature called 'normalized graph Laplacians' and offer theoretical advantages compared to the unnormalized Laplacian (see section 8.4 of [Luxburg, 2007]). They are defined as following:

$$\boldsymbol{L}_{\text{sym}} = \boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{W}\boldsymbol{D}^{1/2} \quad \text{and} \quad \boldsymbol{L}_{\text{rw}} = \boldsymbol{I} - \boldsymbol{D}^{-1}\boldsymbol{W}. \tag{1.7}$$

We will refrain from addressing these two matrices, we shall content ourselves with saying that there exists more efficient spectral clustering algorithms called 'Normalized spectral clustering' that are in the same spirit as Figure 1.6, the reader can refer to [Shi and Malik, 2000, Ng et al., 2001, Luxburg, 2007] for a deeper analysis of the use of this Laplacians. We will simply give an insight on the mechanics behind the spectral clustering algorithm and we shall study the problem from a graph point of view. The spectral algorithm is an approximation to the problem of partitioning the graph G. For A and B, two disjoint subsets of G we define

$$cut(A, B) = \sum_{i \in A, j \in B} w_{ij}. \tag{1.8}$$

```
Input: A similarity matrix $\boldsymbol{S} \in \mathbb{R}^{N \times N}$ and the number of clusters to find $k$.
Output: Clusters $A_1, \dots, A_k$.
1: Construct the weighted adjacency matrix $\boldsymbol{W}$.
2: Compute the unnormalized Laplacian $\boldsymbol{L}$.
3: Compute the $k$ eigenvectors $\boldsymbol{v}_1, \dots, \boldsymbol{v}_k$ of $\boldsymbol{L}$ corresponding to the $k$ smallest eigenvalues of $\boldsymbol{L}$.
4: Let $\boldsymbol{V} \in \mathbb{R}^{N \times k}$ be the matrix containing the vectors $\boldsymbol{v}_1, \dots, \boldsymbol{v}_k$ as columns.
5: For $i \in [N]$, let $\mathbf{y}_i \in \mathbb{R}^k$ be the vector corresponding to the $i^{th}$ row of $\boldsymbol{V}$.
6: Cluster the points $(\mathbf{y}_i)_{i \in [N]} \in \mathbb{R}^k$ with the $k$-means algorithm into clusters $C_1, \dots, C_k$.
7: Construct clusters $A_1, \dots, A_k$ with $A_i = \{i, \mathbf{y}_i \in C_i\}$
```

Figure 1.6: Unnormalized spectral clustering according to [Luxburg, 2007]

e: voir si ne noter sij

Two common objective function to minimize for such partitioning are RatioCut [Hagen and Kahng, 1992] and Ncut [Shi and Malik, 2000] defined as

$$\text{RatioCut}(A_1, \dots, A_k) = \sum_{i=1}^{k} \frac{cut(A_i, \bar{A}_i)}{|A_i|},$$

$$\text{Ncut}(A_1, \dots, A_k) = \sum_{i=1}^{k} \frac{cut(A_i, \bar{A}_i)}{\text{vol}(A_i)},$$

where $|A|$ is the number of vertices in A and $\text{vol}(A) = \sum_{i \in A} d_i$. Note that these two objective functions try to achieve a ”balanced” partitioning, a small component leads to a high value of these objective functions. Unfortunately, such partition problem is NP-hard [Wagner and Wagner, 1993, Luxburg, 2007]. Hopefully a relaxation of this problem with RatioCut is:

$$\min_{\boldsymbol{H} \in \mathbb{R}^{N \times K}} Tr(\boldsymbol{H}^T \boldsymbol{L} \boldsymbol{H}) \quad \text{subject to} \quad \boldsymbol{H}^T \boldsymbol{H} = \boldsymbol{I}. \tag{1.9}$$

It turns out that choosing $\boldsymbol{H}$ as the matrix with the first k eigenvectors of $\boldsymbol{L}$ as columns is a solution of Equation (1.9) (see 5.2 of [Luxburg, 2007]) which is exactly the step 4 in Figure 1.6. The same relaxation can be done for the Ncut objective function

$$\min_{\boldsymbol{U} \in \mathbb{R}^{N \times k}} Tr(\boldsymbol{U}^T \boldsymbol{D}^{-1/2} \boldsymbol{L} \boldsymbol{D}^{-1/2} \boldsymbol{U}) \quad \text{subject to} \quad \boldsymbol{U}^T \boldsymbol{U} = \boldsymbol{I}. \tag{1.10}$$

These relaxation does not give guarantees on the quality of the solutions and the resulting partition can be arbitrary worse than the optimal one in regards to RatioCut and Ncut [Guattery and Miller, 1998, Nadler and Galun, 2007]. In particular, spectral clustering methods are global methods and fail to identify clusters at different scales [Nadler and Galun, 2007]. But these approximations are computationally attractive and very simple to solve, especially with a sparse weighted adjacency matrix. Usually, it is preferred to use $\boldsymbol{L}_{\text{rw}}$ and the normalized spectral clustering method.

1.1.4 Finding the number of clusters

The determination of the number of clusters in a dataset is fundamental and still unsolved problem. Numerous approaches to this problem has been developed over the years, see [Hardy, 1996, Milligan and Cooper, 1985] and Chapter 26 of [Hennig et al., 2015]. Several popular heuristics relies on a graphical interpretation of the quality of clustering. The most popular are the Elbow criterion which consists to perform clusterings with different number of clusters K and compute the ratio of the between group variance versus the total variance (the F-test statistic) for each of them. The detection of an 'elbow' indicates the appropriate number of clusters i.e. with larger K, the clusters does not give more information on the variance. Another technique that relies on a graphical interpretation is the Silhouette method [Rousseeuw, 1987] which asses how well a point is assigned to its cluster compared to nearest neighbor cluster. A more formal approach is the 'Gap Statistic' developed in [Tibshirani et al., 2001], an efficient statistical procedure that compare the change in within-cluster dispersion with that expected under a reference null distribution. In the case of the spectral clustering, one can use the 'eigengap' heuristic on the eigenvalues $\lambda_1, \dots, \lambda_N$ of the Laplacian matrix by choosing K such that $\lambda_1, \dots, \lambda_K$ are small and λ_{K+1} is relatively large (see section 8.3 of [Luxburg, 2007]). Finally, a model selection criterion widely used on probabilistic models and important in our work is the Bayesian Information Criterion (BIC). This method has been developed in [Schwarz, 1978] following the work of Akaike on the AIC [Akaike, 1973]. The idea of this method under the assumption that the observations $\{\boldsymbol{x}_i, \dots, \boldsymbol{x}_N\}$ are drawn from an exponential family is to derive from the approximation of the asymptotic expansion of the Bayes estimator the following quantity

$$\text{BIC} = \widehat{\ell}_j(\widehat{\theta}, \boldsymbol{x}) - \frac{1}{2} k_j \log(N), \tag{1.11}$$

where $\widehat{\ell}_j(\widehat{\theta}, \boldsymbol{x})$ is the maximized likelihood of model j, $\widehat{\theta}$ is the MLE and k_j is number of free parameters of model j. Therefore, the model selection rule is to chose the model for which the BIC is the largest. BIC has several nice properties, it penalizes the complexity of the model which is interesting since choosing the model only on the criterion of the likelihood in the case of Gaussian mixtures leads to select as many components as there are points. One can remark that all procedures mentioned previously require to perform a large number of clusterings and select the best model according to those criteria. This approach is computationally expensive. In Section 1.4.3, we try to address this challenge by iteratively penalizing the weight vector of large Gaussian mixture in an EM-like procedure.

1.2 The Gaussian Mixture Model

We will now focus on the main tool of this thesis. The Gaussian Mixture Model (GMM) is an important framework for clustering problems. Unlike the other methods previously seen, it is a probabilistic approach for clustering. One of the main advantages of model-based clustering is that the resulting partition can be interpreted statistically. It assumes that the observations are drawn from a mixture distribution the components of which are Gaussian with parameters $(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$:

$$\varphi_{\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k}(x) = \frac{1}{(2\pi)^{p/2}|\boldsymbol{\Sigma}_k|^{1/2}} \exp\Big(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_k)\Big) \tag{1.12}$$

Let $\boldsymbol{\theta}$ be the list containing all the unknown parameters of a Gaussian mixture model: the family of means $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K) \in (\mathbb{R}^p)^K$, the family of covariance matrices $\boldsymbol{\Sigma} = (\boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_k) \in (\mathcal{S}^p_{++})^K$ and the vector of cluster probabilities $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_k) \in [0,1]^K$ such that $\mathbf{1}_p^\top \boldsymbol{\pi} = 1$. The density of one observation $\boldsymbol{X}_1$ is then given by:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{k=1}^K \pi_k \varphi_{\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k}(\boldsymbol{x}), \qquad \forall \boldsymbol{x} \in \mathbb{R}^p, \tag{1.13}$$

where $\boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi})$. This model can be interpreted from a latent variable perspective. Let Z be a discrete random variable taking its values in the set $[K]$ and such that $\mathbf{P}(Z = k) = \pi_k$ for every $k \in [K]$. The random variable Z indicates the cluster from which the observation $\boldsymbol{X}$ is drawn. Considering that all the conditional distributions $\boldsymbol{X}|Z = k$ are Gaussian, we get the following formula for the marginal density of X:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{k=1}^K \mathbf{P}(Z = k) p_\theta(\boldsymbol{x}|Z = k) = \sum_{k=1}^K \pi_k \varphi_{\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k}(\boldsymbol{x}), \qquad \forall \boldsymbol{x} \in \mathbb{R}^p. \tag{1.14}$$

In the clustering problem, the goal is to assign X to a cluster or, equivalently, to predict the cluster Z of the vector $\boldsymbol{X}$. A prediction function in such a context is $g : \mathbb{R}^p \to [K]$ such that $g(\boldsymbol{X})$ is as close as possible to Z. If we measure the risk of a prediction function g in terms of misclassification error rate $R_{\boldsymbol{\theta}}(g) = \mathbf{P}_{\boldsymbol{\theta}}(g(\boldsymbol{X}) \neq Z)$, then it is well known that the optimal (Bayes) predictor $g^*_{\boldsymbol{\theta}} \in \arg\min_g R_{\boldsymbol{\theta}}(g)$ is provided by the rule

$$g^*_{\boldsymbol{\theta}}(\boldsymbol{x}) = \arg\max_{k\in[K]} \tau_k(\boldsymbol{x}, \boldsymbol{\theta}),$$

where $\tau_k(\boldsymbol{x}, \boldsymbol{\theta}) = p_{\boldsymbol{\theta}}(Z = k|\boldsymbol{X} = \boldsymbol{x})$ stands for the conditional probability of the latent variable Z given $\boldsymbol{X}$. In the Gaussian mixture model, Bayes's rule implies that

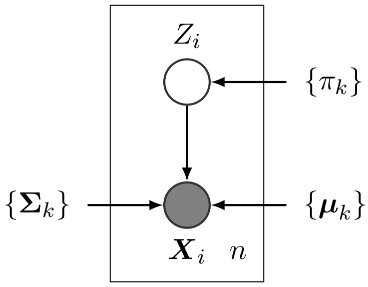
$$\tau_k(\boldsymbol{x}, \boldsymbol{\theta}) = \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}|Z = k)\mathbf{P}(Z = k)}{p_{\boldsymbol{\theta}}(\boldsymbol{x})} = \frac{\pi_k \varphi_{\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k}(\boldsymbol{x})}{\sum_{k'=1}^K \pi_{k'} \varphi_{\boldsymbol{\mu}_{k'},\boldsymbol{\Sigma}_{k'}}(\boldsymbol{x})} \tag{1.15}$$

Since the true value of the parameter $\boldsymbol{\theta}$ is not available, formula (1.15) can not be directly used for solving the problem of clustering. Instead, a natural strategy is to estimate $\boldsymbol{\theta}$ by some vector $\widehat{\boldsymbol{\theta}}$, based on a sample $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n$ drawn from the density $p_{\boldsymbol{\theta}}$, and then to define the clustering rule by

$$\widehat{g}(\boldsymbol{x}) = g^*_{\widehat{\boldsymbol{\theta}}}(\boldsymbol{x}) = \arg\max_{k\in[K]} \tau_k(\boldsymbol{x}, \widehat{\boldsymbol{\theta}}) = \arg\max_{k\in[K]} \widehat{\pi}_k \varphi_{\widehat{\boldsymbol{\mu}}_k, \widehat{\boldsymbol{\Sigma}}_k}(\boldsymbol{x}). \tag{1.16}$$

A common approach to estimating the parameter $\boldsymbol{\theta}$ is to rely on the likelihood maximization. Let $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n$ with $\boldsymbol{X}_i \in \mathbb{R}^p$ be a set of iid observations drawn from the density $p_{\boldsymbol{\theta}}$ given by (1.13). The following graphical model depicts the scheme of the observations: The log-likelihood of the Gaussian mixture model is

$$\ell_n(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_i) = \sum_{i=1}^{n} \log \left\{ \sum_{k=1}^{K} \pi_k \varphi_{(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}(\boldsymbol{x}_i) \right\}. \tag{1.17}$$



Because of the presence in this equation of the logarithm of a sum, the maximization of the log-likelihood is a difficult nonlinear and nonconvex problem. In particular, this is not a exponential family distribution yielding simple expressions. A commonly used approach for approximately maximizing (1.17) with respect to $\boldsymbol{\theta}$ is the Expectation-Maximization (EM) Algorithm [Dempster et al., 1977] that we recall below.

Summarizing the content of this section, we can describe the following natural approach to solving the clustering problem under Gaussian mixture modeling assumption:

```
Input: data vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^p$ and the number of clusters $K$
Output: function $\widehat{g} : \mathbb{R}^p \to [K]$
1: Estimate $\boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ by maximizing the log-likelihood:

    $\widehat{\boldsymbol{\theta}} \in \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \ell(\boldsymbol{\theta}|\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \arg\max_{\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}} \sum_{i=1}^{n} \log \left\{ \sum_{k=1}^{K} \pi_k \varphi_{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k}(\boldsymbol{x}_i) \right\}.$   (1.18)

2: Output the clustering rule:

    $\widehat{g}(\cdot) = \arg\max_{k\in[K]} \widehat{\pi}_k \varphi_{\widehat{\boldsymbol{\mu}}_k, \widehat{\boldsymbol{\Sigma}}_k}(\cdot).$   (1.19)
```

Figure 1.7: Clustering under Gaussian mixture modeling

1.2.1 EM Algorithm

The goal of the EM algorithm is to approximate a solution of the problem (1.18). Since this optimization problem contains a nonconvex cost function, it is impossible to design a polynomial time algorithm that provably converges to the global maximum point. Instead, the EM algorithm provides a sequence $\{\widehat{\boldsymbol{\theta}}(t)\}_{t\in\mathbb{N}}$ of parameter values such that the cost function (*i.e.*, the log-likelihood) evaluated at these values forms an increasing sequence that converges to a local maximum.

The main idea underlying the EM algorithm is the following representation of the log-likelihood of one observation derived from the log-sum inequality:

$$\log\left\{\sum_{k=1}^{K}\pi_k\varphi_{\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k}(\boldsymbol{x}_i)\right\} = \max_{\boldsymbol{\tau}\in[0,1]^K\ \boldsymbol{\tau}^\top\mathbf{1}_K=1}\sum_{k=1}^{K}\Big\{\tau_k\log\varphi_{\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k}(\boldsymbol{x}_i)+\tau_k\log(\pi_k/\tau_k)\Big\}. \tag{1.20}$$

Let us denote by $\boldsymbol{\mathcal{T}}=(\tau_{i,k})$ a $n\times K$ matrix with nonnegative entries such that $\boldsymbol{\mathcal{T}}\mathbf{1}_K=\mathbf{1}_n$, that is each row of $\boldsymbol{\mathcal{T}}$ is a probability distribution on $[K]$. Combining (1.18) and (1.20), we get

$$\widehat{\boldsymbol{\theta}}\in\arg\max_{\boldsymbol{\theta}=(\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})}\max_{\boldsymbol{\mathcal{T}}}\sum_{i=1}^{N}\sum_{k=1}^{K}\Big\{\tau_{i,k}\log\varphi_{\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k}(\boldsymbol{x}_i)+\tau_{i,k}\log(\pi_k/\tau_{i,k})\Big\}. \tag{1.21}$$

The great advantage of this new representation of the log-likelihood function is that the cost function in (1.21), considered as a function of $\boldsymbol{\theta}$ and $\boldsymbol{\mathcal{T}}$, is biconcave, *i.e.*, it is concave with respect to $\boldsymbol{\theta}$ for every fixed $\boldsymbol{\mathcal{T}}$ and concave with respect to $\boldsymbol{\mathcal{T}}$ for every fixed $\boldsymbol{\theta}$. In such a situation, one can apply the alternating maximization approach to sequentially improve on an initial point. In the present context, an additional attractive feature of the cost function in (1.21) is that the two optimization problems involved in the alternating maximization procedure admit explicit solutions.

Lemma 1. *Let us introduce the cost function*

$$F(\boldsymbol{\theta},\boldsymbol{\mathcal{T}})=\sum_{i=1}^{n}\sum_{k=1}^{K}\Big\{\tau_{i,k}\log\varphi_{\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k}(\boldsymbol{x}_i)+\tau_{i,k}\log(\pi_k/\tau_{i,k})\Big\}. \tag{1.22}$$

Then, the following two optimization problems

$$\widehat{\boldsymbol{\theta}}(\boldsymbol{\mathcal{T}})\in\arg\max_{\boldsymbol{\theta}}F(\boldsymbol{\theta},\boldsymbol{\mathcal{T}}),\qquad \widehat{\boldsymbol{\mathcal{T}}}(\boldsymbol{\theta})\in\arg\max_{\boldsymbol{\mathcal{T}}}F(\boldsymbol{\theta},\boldsymbol{\mathcal{T}}) \tag{1.23}$$

has explicit solutions given by

$$\widehat{\pi}_k=\frac{1}{n}\sum_{i=1}^{n}\tau_{i,k},\qquad \widehat{\boldsymbol{\mu}}_k=\frac{1}{n\widehat{\pi}_k}\sum_{i=1}^{n}\tau_{i,k}\boldsymbol{x}_i,\qquad \forall k\in[K], \tag{1.24}$$

$$\widehat{\boldsymbol{\Sigma}}_k=\frac{1}{n\widehat{\pi}_k}\sum_{i=1}^{n}\tau_{i,k}(\boldsymbol{x}_i-\widehat{\boldsymbol{\mu}}_k)(\boldsymbol{x}_i-\widehat{\boldsymbol{\mu}}_k)^\top,\qquad \forall k\in[K], \tag{1.25}$$

$$\widehat{\tau}_{i,k}=\frac{\pi_k\varphi_{\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k}(\boldsymbol{x}_i)}{\sum_{k'\in[K]}\pi_{k'}\varphi_{\boldsymbol{\mu}_{k'},\boldsymbol{\Sigma}_{k'}}(\boldsymbol{x}_i)},\qquad \forall k\in[K],\ \forall i\in[n]. \tag{1.26}$$

```
Input: data vectors $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n \in \mathbb{R}^p$ and the number of clusters $K$
Output: parameter estimate $\widehat{\boldsymbol{\theta}} = \{\widehat{\boldsymbol{\mu}}_k, \widehat{\boldsymbol{\Sigma}}_k, \pi_k\}_{k\in[K]}$
Initialize $t = 0$, $\boldsymbol{\theta} = \boldsymbol{\theta}^0$.
Repeat
    Update the parameter $\boldsymbol{\mathcal{T}}$:
        $\tau_{i,k}^t = \frac{\pi_k^t \varphi_{\boldsymbol{\mu}_k^t, \boldsymbol{\Sigma}_k^t}(\boldsymbol{x}_i)}{\sum_{k'\in[K]} \pi_{k'}^t \varphi_{\boldsymbol{\mu}_{k'}^t, \boldsymbol{\Sigma}_{k'}^t}(\boldsymbol{x}_i)}.$
    Update the parameter $\boldsymbol{\theta}$:
        $\pi_k^{t+1} = \frac{1}{n}\sum_{i=1}^n \tau_{i,k}^t, \qquad \boldsymbol{\mu}_k^{t+1} = \frac{1}{n\pi_k^{t+1}} \sum_{i=1}^n \tau_{i,k}^t \boldsymbol{x}_i,$
        $\boldsymbol{\Sigma}_k^{t+1} = \frac{1}{n\pi_k^{t+1}} \sum_{i=1}^n \tau_{i,k}^t (\boldsymbol{x}_i - \boldsymbol{\mu}_k^{t+1})(\boldsymbol{x}_i - \boldsymbol{\mu}_k^{t+1})^\top.$
    increment $t$: $t = t + 1$.
Until stopping rule.
Return $\boldsymbol{\theta}^t$.
```

Figure 1.8: EM algorithm for Gaussian mixtures

Based on this result, the EM algorithm is defined as in Figure 1.8. The algorithm operates iteratively and needs a criterion to determine when the iterations should be stopped. There is no clear consensus on this point in the statistical literature, but it is a commonly used practice to stop when one of the following conditions is fulfilled:

i) The number of iterations t exceeds a pre-specified level $t_{\max}$.

ii) The increase of the log-likelihood over past t_0 iterations is not significantly different from zero: $\ell_n(\boldsymbol{\theta}^t) - \ell_n(\boldsymbol{\theta}^{t-t_0}) \leq \varepsilon$ for some pre-specified values $t_0 \in \mathbb{N}$ and $\varepsilon > 0$.

EM is conceptually easy and each iteration increases the log-likelihood:

$$\ell_n(\boldsymbol{\theta}^{t+1}) \geq \ell_n(\boldsymbol{\theta}^t), \qquad \forall t \in \mathbb{N}.$$

The complexity at each step of the EM algorithm is $O(Knp^2)$ and it usually requires many iterations to converge. In a high-dimensional setting when p is large, the quadratic dependence on p may result in prohibitively large running times. However, the computation of the elements of the covariance matrices $\boldsymbol{\Sigma}_k^t$ and the mean vectors $\boldsymbol{\mu}_k^t$ can be parallelized which may lead to considerable savings in the running time.

1.2.2 K-means from the EM angle

In this section, we will see that the K-means problem is closely related to the EM algorithm. We rewrite the minimization problem of K-means defined in eq. (1.1) as following

$$\min_{\boldsymbol{c}_1,\ldots,\boldsymbol{c}_K} \min_{\boldsymbol{R}\in\{0,1\}^{N\times K}} \sum_{i=1}^{N}\sum_{k=1}^{K} r_{ik}\|\boldsymbol{x}_i - \boldsymbol{c}_k\|^2, \tag{1.27}$$

where the matrix $\boldsymbol{R}$ follows the 1-of-K coding scheme, i.e. if a point $\boldsymbol{x}_i$ is assigned to the cluster m, then $r_{im} = 1$ and $r_{il} = 0, \forall l \in [K] \neq m$. One can see the underlying iterative procedure, the first one consists to minimize the objective function with respect to $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_K$ with $\boldsymbol{R}$ fixed (Maximization step) and the second one consists to minimize the objective function with respect to $\boldsymbol{R}$ with $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_K$ fixed (Expectation step). Consider the **E**-step, the objective function is linear with respect to $\boldsymbol{R}$. It consists for a data point $\boldsymbol{x}_i$, to find the cluster k such that $k = \arg\min_{j\in[K]} \|\boldsymbol{x}_i - \boldsymbol{c}_j\|^2$. For the **M**-step, setting the gradient with respect to $\boldsymbol{c}_k$ to 0 gives us

$$2\sum_{i=1}^{N} r_{ik}(\boldsymbol{x}_i - \boldsymbol{c}_k) = 0, \tag{1.28}$$

which leads to

$$\boldsymbol{c}_k = \frac{\sum_{i=1}^{N} r_{ik}\boldsymbol{x}_i}{\sum_{i=1}^{N} r_{ik}}. \tag{1.29}$$

Since $\sum_{i=1}^{N} r_{ik}$ is the size of the cluster k, we recovered the Lloyd's algorithm.

1.3 The curse of dimensionality

The expression 'Curse of dimensionality' introduced by R.Bellman in his book on dynamic programming [Bellman, 1957] refers to the problems linked with high dimension. One can see that evaluating a function on the segment $(0, 1)$ with a step size of 0.1 is straightforward. However, evaluating the function in a grid of dimension 10 requires 10^{10} computations which can be intractable even today within a reasonable time. Many computational and statistical problems arise in this setting. Sometimes the literature refers to a 'high dimensional' setting when $p \gg n$ and more precisely when the model considered has more parameters or degree of freedom than there are observations. In the following, we recall some classical phenomenons that appear in this context and focus on the clustering with high dimensional data.

We saw previously different clustering methods that rely on a distance such as the Euclidean distance. It turns out that in a high dimensional setting, the notion of nearest point vanishes: the minimal distance increases but on the other hand the variance of the distance between points has a slower increase. Consider 2 p-dimensional random variables $\boldsymbol{X}, \boldsymbol{X}'$ and the Euclidean norm, the scaled deviation is then

$$\frac{\mathrm{sdev}[\|\boldsymbol{X}-\boldsymbol{X}'\|^2]}{\mathbb{E}[\|\boldsymbol{X}-\boldsymbol{X}'\|^2]} \approx \frac{1}{\sqrt{p}}, \tag{1.30}$$

and goes to 0 when $p \to \infty$. A direct consequence of such distance concentration phenomenon is the loss of relevance of the methods based on discriminating near and far neighbors such as those studied in the previous section (nearest center for K-means, agglomeration in hierarchical clustering or constructing adjacency graph for spectral clustering). In the clustering context, a strong assumption for ensuring the separation of clusters would be to consider the inter-cluster distance dominant compared to the variance within each clusters. Another phenomenon is the "error accumulation", consider the classical linear regression setting $\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta}^* + \boldsymbol{\epsilon}$ with $\boldsymbol{X} \in \mathbb{R}^{N\times p}$ and $\boldsymbol{\epsilon}_1, \dots, \boldsymbol{\epsilon}_N$ i.i.d. centered with variance σ^2. The least-square estimator $\widehat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}\in\mathbb{R}^p} \|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}\|^2$ verify the following estimation error

$$\mathbb{E}[\|\widehat{\beta} - \boldsymbol{\beta}^*\|^2] = p\sigma^2. \tag{1.31}$$

Therefore we can see that the estimation error increases linearly with the dimension. Furthermore, an interesting phenomenon that occurs in high dimension is that spaces are mostly empty and the realizations of a p-dimensional random vector with a uniform probability distribution on the unit hypersphere lies with high probability close to the boundary of this hypersphere. Therefore, the data belong mostly in a $p-1$ dimensional subspace. Interestingly, the ratio of the volume of a hypersphere versus the volume of the hypercube goes to 0 as $p \to \infty$ (see section 2.3 of [Zimek et al., 2012]). This means that most of the volume lies in the corner of the hypercube. Therefore, any methods based on a spherical distance such as the Euclidean norm is meaningless in this context. One could consider a probabilistic approach to overcome the issues with high dimension. Unfortunately, model-based clustering suffers overparametrization. In the Gaussian mixture model of K components in dimension p, the number of free parameters to estimate is

$$\nu = \underbrace{(K-1)}_{\text{Weights}} + \underbrace{Kp}_{\text{Means}} + \underbrace{Kp(p-1)^2}_{\text{Covariances Matrices}}, \tag{1.32}$$

which for $p = 10$ and $K = 5$ is 4104. Moreover, the evaluation of $\widehat{\tau}_{i,k}$ in eq. (1.26) needs to evaluate the inverse of the covariance matrix $\widehat{\boldsymbol{\Sigma}}_k$ which is called the precision matrix. If $n << \nu$ the matrices $\widehat{\boldsymbol{\Sigma}}_k$ with $k \in [K]$ are ill conditioned and the precision matrices are prone to large numerical errors or more often are singular and the problem can not be solved.

Several popular methods are used to overcome these issues. One can reasonably consider that multiple variables are correlated or that most of the dimensions are irrelevant and therefore clusters may live on a lower-dimensional subspace. A first approach would be to perform a dimension reduction like Principal Component Analysis (PCA) but this leads to a decoupling of the dimension reduction task from the clustering task and may lead to a poor selection of the subspace [Bouveyron and Brunet, 2013], keeping information from irrelevant dimensions. Moreover, the interpretation of the resulting linearly transformed dimensions are difficult to interpret. Another approach called 'feature selection' consists to select the most relevant features but fails when clusters live in different subspaces. This scenario leads to the development of 'subspace clustering' techniques that goes one step further by selecting the most relevant features for each clusters separately (see [Parsons et al., 2004] for a review on this topic).

In our work, we will focus in the regularization technique and make sparse assumption on the structure of the Gaussian mixture model. The goal is to reduce the number of free parameters and tackle the problem of estimating the inverse of the covariance matrix. In section 1.4.1, we tackle this challenge by studying some nice structural properties of precision matrices.

The reader can find a more thorough study of high dimensional statistics in Giraud [2014], Zimek et al. [2012]. For an overview of clustering in high dimension see [Bouveyron and Brunet, 2013, Parsons et al., 2004].

1.4 Some contributions

In this section, we present some works carried out during this thesis which have unfortunately not been able to be the subject of an in-depth study that can be published. The first part deals with the sparse hypothesis of the precision matrices within a high dimensional Gaussian mixture and adapts the single-component Graphical Lasso from [Friedman et al., 2007] to the mixture setting. In the second part, we assume that the weight vector of the mixture is sparse in order to obtain an estimator of the number of components in the mixture that is generally unknown.

1.4.1 Graphical Lasso for Gaussian mixtures

As we saw in the introduction chapter, the number of free parameters in a full GMM with K components in dimension p are $(K-1)+Kp+Kp(p+1)/2$ which means that for $K=5$ and $p=100$ we have 125704 parameters to estimate. In this high dimensional setting, the EM algorithm experiences severe performance degradation. In particular, the inversion of the covariance matrices are challenged. One way to tackle these problems is to use regularization. We will make the assumption on some structure on the inverse of the covariance matrix of a component called the precision or concentration matrix. The work presented in this chapter is inspired by [Friedman et al., 2007], [Banerjee et al., 2008], [Yuan and Lin, 2007] and [Meinshausen and Bühlmann, 2006] in which they penalize the components of the precision matrix of a Gaussian graphical model. We generalize this work to the Gaussian mixture model.

Note: changer en section

Introduction

We consider $\boldsymbol{X} = (\boldsymbol{X}^{(1)}, \ldots, \boldsymbol{X}^{(p)})$ a random vector admitting a p-dimensional normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with $\boldsymbol{\Sigma}$ non-singular. One can construct an undirected graph $G = (V, E)$ with p vertices corresponding to each coordinates and, $E = (e_{i,j})_{1 \le i < j \le p}$, the edges between the vertices describing the conditional independence relationship among $\boldsymbol{X}^{(1)}, \ldots, \boldsymbol{X}^{(p)}$. If in this graph, $e_{i,j}$ is absent in E if and only if $X^{(i)}$ and $X^{(j)}$ are independent conditionally to the other variables $\{X^{(l)}\}$ with $l \neq i, j$ (noted $X^{(i)} \perp\!\!\!\perp X^{(j)} | X^{(l)}\, l \neq i, j$), then G is called the Gaussian concentration graph model for the Gaussian random vector $\boldsymbol{X}$. This property is particularly interesting in the study of the inverse of the covariance matrix. Let us denote $\boldsymbol{\Sigma}^{-1} = \boldsymbol{\Omega} = (\omega_{i,j})$ the precision matrix. The components of this matrix verify $\omega_{i,j} = 0$ if and only if $X^{(i)} \perp\!\!\!\perp X^{(j)}$ conditionally to the other variables. We recall in the following lemma this well known result

Lemma 1.4.1 (Conditional independence in Gaussian concentration graph model). *Consider $\boldsymbol{X} = (\boldsymbol{X}^{(1)}, \ldots, \boldsymbol{X}^{(p)})$ a p-dimensional random vector with a multivariate normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, note $\boldsymbol{\Sigma}^{-1} = \boldsymbol{\Omega} = (\omega_{i,j})$, then $X^{(i)} \perp\!\!\!\perp X^{(j)} | X^{(l)} \iff \omega_{i,j} = 0$ with $l \neq i, j$*

Proof. This result can be found in [Edwards, 2000], consider the density of $\boldsymbol{X}$

$$\varphi_{\boldsymbol{\mu},\boldsymbol{\Sigma}}(\boldsymbol{x}) = \frac{1}{(2\pi)^{p/2} |\boldsymbol{\Sigma}|^{1/2}} \exp\Big(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu}) \Big), \tag{1.33}$$

it can be rewritten as

$$\varphi_{\boldsymbol{\mu},\boldsymbol{\Sigma}}(\boldsymbol{x}) = \exp(\alpha + \beta^T \boldsymbol{x} - \frac{1}{2} \boldsymbol{x}^T \boldsymbol{\Omega} \boldsymbol{x}), \tag{1.34}$$

with $\beta = \boldsymbol{\Omega}\boldsymbol{\mu}$ and $\alpha = \frac{1}{2}\log(|\boldsymbol{\Omega}|) - \frac{1}{2}\boldsymbol{\mu}^T \boldsymbol{\Omega} \boldsymbol{\mu} - \frac{p}{2}\log(2\pi)$. Then, the previous equation can be rewritten as

$$\exp\big(\alpha + \sum_{j=1}^{p} \beta_j \boldsymbol{x}^{(j)} - \frac{1}{2} \sum_{j=1}^{p} \sum_{(i=1)}^{p} \omega_{i,j} \boldsymbol{x}^{(j)} \boldsymbol{x}^{(i)}\big). \tag{1.35}$$

Now, for X, Y, Z three random variables, we have $X \perp\!\!\!\perp Y | Z$ iff the joint density can be factorized into two factors $f_{X,Y,Z}(x, y, z) = h(x, z) g(y, z)$ with h anf g two functions. Then, at the light of eq. (1.35), we have $X^{(i)} \perp\!\!\!\perp X^{(j)} | X^{(l)} \iff \omega_{i,j} = 0$. □

The literature on this subject focused on a first hand on the estimation of the graph structure, [Dempster, 1972] developed a greedy forward or backward search method to estimate the set of non-zero components in the concentration matrix. The forward method relies on initializing an empty set and select iteratively an edge with an MLE fit for $\mathcal{O}(p^2)$ different parameters. The procedure stops according to a suitable selection criterion. The backward method performs in the same manner by starting with all edges and performing deletions. It is obvious that such methods are computationally intractable in high dimension. In [Meinshausen and Bühlmann, 2006], the authors studied a neighborhood selection procedure with lasso. The goal is to estimate the neighborhood $ne_{X^{(i)}}$ of a node $X^{(i)}$ which is the smallest subset of $G \setminus \{X^{(i)}\}$ such that $X^{(i)} \perp\!\!\!\perp \big\{X^{(j)} : X^{(j)} \in G \setminus \{ne_{X^{(i)}}\}\big\} | X_{ne_{X^{(i)}}}$. The estimation of the neighborhood is cast as a regression problem with a lasso penalization. The authors showed that this procedure is consistent for sparse high dimensional graphs and computationally efficient. More precisely, let $\theta^{(i)} \in \mathbb{R}^p$ be the vector of coefficient of the optimal prediction,

Note: banerjee p488, consistency lies on choice of penalty

$$\theta^{(i)} = \arg\min_{\theta:\theta_i=0} \mathbb{E}\Big[X^{(i)} - \sum_{k=1}^{p} \theta_k X^{(k)}\Big], \tag{1.36}$$

then the components of $\theta^{(i)}$ are determined by the precision matrix, $\theta_j^{(i)} = -\omega_{i,j}/\omega_{i,i}$. Therefore, the set of neighbors of $X^{(i)} \in G$ is given by

$$ne_{X^{(i)}} = \{X^{(j)}, j \in [p] : \omega_{i,j} \neq 0\}. \tag{1.37}$$

Now, let $\mathbb{X}$ be the $n \times p$-dimensional matrix such that the column $\mathbb{X}^{(i)}$ is the n observations vector of $X^{(i)}$, given a regularization parameter $\lambda \geq 0$ carefully chosen, the Lasso estimate $\widehat{\theta}^{i,\lambda}$ of $\theta^{(i)}$ is given by

$$\widehat{\theta}^{i,\lambda} = \arg\min_{\theta:\theta_i=0} \Big(\frac{1}{n}\|\mathbb{X}^{(i)} - \mathbb{X}\theta\|_2^2 + \lambda\|\theta\|_1\Big). \tag{1.38}$$

The authors proved under several assumptions that

$$P(\widehat{ne}^{\lambda}_{X^{(i)}} = ne_{X^{(i)}}) \to 1 \quad \text{for}\, n \to \infty, \tag{1.39}$$

and for some $\epsilon > 0$,

$$P(\widehat{E}^{\lambda} = E) = 1 - \mathcal{O}(\exp(-cn^{\epsilon})) \quad \text{for}\, n \to \infty. \tag{1.40}$$

Therefore, this method recovers the conditional independence structure of sparse high-dimensional Gaussian concentration graph at exponential rates. However, this method performs model selection but does not estimate the parameters of the model. One could estimate the parameters of a model which has been selected by this method. Such procedure often leads to instability of the estimator since small changes on the data would change the model selected [Yuan and Lin, 2007], [Breiman, 1996]. One major difficulty of a method that would perform both tasks is to ensure that the estimator of the precision matrix is positive definite. [Yuan and Lin, 2007] proposed a penalized-likelihood method that performs model selection and parameter estimation simultaneously as well as ensuring the positive definiteness of the precision matrix. Their approach is similar to [Meinshausen and Bühlmann,

Note: ajouter un mot sur la complexité

2006] as they use the ℓ_1 penalty but with the likelihood and the addition of a positive definite constraint. The log-likelihood for $\boldsymbol{\Omega}$ based on a centered random sample $\boldsymbol{X}_1, \dots, \boldsymbol{X}_n$ of $\boldsymbol{X}$ is

$$\frac{n}{2}\log(|\boldsymbol{\Omega}|) - \frac{1}{2}\sum_{i=1}^{n}\boldsymbol{X}_i^T\boldsymbol{\Omega}\boldsymbol{X}_i \tag{1.41}$$

and the constrained minimization problem over the set of positive definite matrices is

$$\min\Big\{-\log(|\boldsymbol{\Omega}|) + \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{X}_i^T\boldsymbol{\Omega}\boldsymbol{X}_i\Big\} \quad \text{subject to} \quad \sum_{i\neq j}|\omega_{i,j}| \leq t, \tag{1.42}$$

with $t \geq 0$ a tuning parameter. Note that $\widehat{\boldsymbol{\mu}} = \bar{\boldsymbol{X}}$. Consider the empirical covariance matrix $\boldsymbol{S} = 1/n\sum_{i=1}^{n}\boldsymbol{X}_i^T\boldsymbol{X}_i$, the eq. (1.42) can be rewritten as

$$\min\Big\{-\log(|\boldsymbol{\Omega}|) + \operatorname{tr}(\boldsymbol{S}\boldsymbol{\Omega})\Big\} \quad \text{subject to} \quad \sum_{i\neq j}|\omega_{i,j}| \leq t. \tag{1.43}$$

Since the whole problem is convex, the Lagrangian form is given by

$$\mathcal{L}(\lambda, \boldsymbol{\Omega}) = -\log(|\boldsymbol{\Omega}|) + \operatorname{tr}(\boldsymbol{S}\boldsymbol{\Omega}) + \lambda\sum_{i\neq j}|\omega_{i,j}|, \tag{1.44}$$

with λ the tuning parameter. A non-negative garrote-type estimator is provided in [Yuan and Lin, 2007] but can be only applied when a good estimator of $\boldsymbol{\Omega}$ is available. Therefore, we will continue our study of the Lasso-type estimator, the authors provided an asymptotic result

Note: regarder de plus pres

Theorem 1.4.1 (Theorem 1 from [Yuan and Lin, 2007]). *If $\sqrt{n}\lambda \to \lambda_0 \geq 0$ as $n \to \infty$, the lasso-type estimator is such that*

$$\sqrt{n}(\widehat{\boldsymbol{\Omega}} - \boldsymbol{\Omega}) \to \arg\min_{\boldsymbol{U}=\boldsymbol{U}^T}(V),$$

in distribution where

$$V(\boldsymbol{U}) = \operatorname{tr}(\boldsymbol{U\Sigma U\Sigma}) + \operatorname{tr}(\boldsymbol{UW}) + \lambda_0\sum_{i\neq j}\big\{u_{i,j}\operatorname{sign}(\omega_{i,j})I(\omega_{i,j}\neq 0) + |u_{i,j}|I(\omega_{i,j}=0)\big\}$$

in which $\boldsymbol{W}$ is a random symmetric $p\times p$ matrix such that $\operatorname{vec}(\boldsymbol{W}) \sim \mathcal{N}(0,\Lambda)$, and Λ is such that

$$\operatorname{cov}(w_{i,j}, w_{i',j'}) = \operatorname{cov}(X^{(i)}X^{(j)}, X^{(i')}X^{(j')}).$$

Note: mettre un commentaire sur ce resultat et aspect algorithmique

Unfortunately, the computational complexity of interior point methods for maximizing eq. (1.44) is $\mathcal{O}(p^6)$ and at each steps, we have to compute and store a Hessian matrix of size $\mathcal{O}(p^2)$. These prohibitive complexities led the research on more specialized methods. [Banerjee et al., 2008] worked on the same approach, solving a maximum likelihood problem with an ℓ_1 penalty and focusing on the computation complexity by proposing an iterative block coordinate descent algorithm. The problem to maximize is similar to eq. (1.44)

$$\widehat{\boldsymbol{\Omega}} = \arg\max_{\boldsymbol{\Omega}\succ 0}\{\log(|\boldsymbol{\Omega}|) - \operatorname{tr}(\boldsymbol{S\Omega}) - \lambda\|\boldsymbol{\Omega}\|_1\}. \tag{1.45}$$

```
Input: Matrix $\boldsymbol{S}$, parameter $\lambda$ and threshold $\varepsilon$
Output: Estimate of $\boldsymbol{W}$
Initialize $\boldsymbol{W}^{(0)} := \boldsymbol{S} + \lambda I$
repeat
    for $j = 1, \ldots, p$ do
        (a) Let $\boldsymbol{W}^{(j-1)}$ denote the current iterate. Solve the quadratic program
                $\widehat{\mathbf{y}} := \arg\min_{\mathbf{y}} \{\mathbf{y}^T (\boldsymbol{W}^{(j-1)}_{\backslash j \backslash j})^{-1} \mathbf{y} : \|\mathbf{y} - \boldsymbol{S}_j\|_\infty \leq \lambda\}.$
        (b) Update the rule: $\boldsymbol{W}^{(j)}$ is $\boldsymbol{W}^{(j-1)}$ with column/row $\boldsymbol{W}_j$ replaced by $\widehat{\mathbf{y}}$.
    end for
    Let $\widehat{\boldsymbol{W}}^{(0)} := \boldsymbol{W}^{(p)}$.
until convergence occurs when
        $\mathrm{tr}\big((\widehat{\boldsymbol{W}}^{(0)})^{-1}\boldsymbol{S}\big) - p + \lambda\big\|(\widehat{\boldsymbol{W}}^{(0)})^{-1}\big\|_1 \leq \varepsilon.$
```

Figure 1.9: Block Coordinate Descent Algorithm

Note that the ℓ_1 norm of a matrix $\boldsymbol{\Omega}$ can be expressed as

$$\|\boldsymbol{\Omega}\|_1 = \max_{\|\boldsymbol{U}\|_\infty \leq 1} \mathrm{tr}(\boldsymbol{\Omega}\boldsymbol{U}), \tag{1.46}$$

injecting this in eq. (1.45) gives

$$\max_{\boldsymbol{\Omega} \succ 0} \min_{\|\boldsymbol{U}\|_\infty \leq \lambda} \Big\{ \log(|\boldsymbol{\Omega}|) - \mathrm{tr}(\boldsymbol{\Omega}(\boldsymbol{S} + \boldsymbol{U})) \Big\}. \tag{1.47}$$

After exchanging the min and the max, we solve the problem for $\boldsymbol{\Omega}$ by setting the gradient to 0 which gives $(\boldsymbol{\Omega}^{-1})^T - (\boldsymbol{S} + \boldsymbol{U})^T = 0$ then $\boldsymbol{\Omega} = (\boldsymbol{S} + \boldsymbol{U})^{-1}$. The dual problem is then

$$\min_{\|\boldsymbol{U}\|_\infty} \{- \log(|\boldsymbol{S} + \boldsymbol{U}|) - p\}, \tag{1.48}$$

or by setting $\boldsymbol{W} = \boldsymbol{S} + \boldsymbol{U}$,

$$\widehat{\boldsymbol{\Sigma}} = \widehat{\boldsymbol{\Omega}^{-1}} = \arg\max \log(|\boldsymbol{W}|) \quad \text{s.t} \quad \|\boldsymbol{W} - \boldsymbol{S}\|_\infty \leq \lambda. \tag{1.49}$$

We observe the presence of a log-barrier adding the implicit constraint $(\boldsymbol{S} + \boldsymbol{U}) \succ 0$. Furthermore, the dual problem estimates the covariance matrix... To solve this maximization problem, the authors proposed a Block Coordinate Descent Algorithm described in fig. 1.9. For any symmetric matrix $\boldsymbol{A}$, let $\boldsymbol{A}_{\backslash k \backslash j}$ be the matrix produced by removing column k and row j to $\boldsymbol{A}$. Let $\boldsymbol{A}_j$ the j^{th} column of $\boldsymbol{A}$ with the element $\boldsymbol{A}_{jj}$ removed. They proved that the Block Coordinate Descent algorithm converges, achieving an ε-suboptimal solution to eq. (1.49) and each iterates produce a strictly positive definite matrix. For a fixed number of sweeps K, the complexity of this algorithm is $\mathcal{O}(Kp^4)$. They provide also another algorithm using Nesterov's first

Note: pourquoi $\Sigma_{kk} = S_{kk} + \lambda$?, p488

Note: citer les theoremes et choix du param order method which has a $\mathcal{O}(p^{4.5}/\epsilon)$ complexity for $\varepsilon > 0$ the desired accuracy. It is interesting to note that the dual problem of line 6 in fig. 1.9 is

$$\min_{\boldsymbol{x}} \boldsymbol{x}^T \boldsymbol{W}^{(j-1)}_{\backslash j \backslash j} \boldsymbol{x} - \boldsymbol{S}_j^T \boldsymbol{x} + \lambda \|\boldsymbol{x}\|_1, \tag{1.50}$$

and strong duality holds, it can best casted as

$$\min_{\boldsymbol{x}} \|\boldsymbol{Q}\boldsymbol{x} - \boldsymbol{b}\|_2^2 + \lambda \|\boldsymbol{x}\|_1, \tag{1.51}$$

with $\boldsymbol{Q} = (\boldsymbol{W}^{(j-1)}_{\backslash j \backslash j})^{1/2}$ and $\boldsymbol{b} := \frac{1}{2}\boldsymbol{Q}^{-1}\boldsymbol{S}_j$. Therefore, we recover the Lasso problem, more precisely, the algorithm can be interpreted as a sequence of iterative Lasso problems. This approach is similar to another paper that we would like to mention [Friedman et al., 2007]. The authors proposed a faster algorithm based on the Block Coordinate Descent algorithm from [Banerjee et al., 2008] called Graphical Lasso. They estimate the matrix $\boldsymbol{W} = \boldsymbol{\Omega}^{-1}$ by performing iterative permutations of the columns of this matrix to make the target column the last for a coupled Lasso problem. The matrices $\boldsymbol{W}$ and $\boldsymbol{S}$ will be presented as following

$$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{W}_{11} & \boldsymbol{w}_{12} \\ \boldsymbol{w}_{21} & w_{22} \end{bmatrix}, \quad \boldsymbol{S} = \begin{bmatrix} \boldsymbol{S}_{11} & \boldsymbol{s}_{12} \\ \boldsymbol{s}_{21} & s_{22} \end{bmatrix}, \tag{1.52}$$

and the Graphical Lasso algorithm is described in fig. 1.10. The Lasso problem can be solved via a coordinate descent, the reader can refer to [Friedman et al., 2007] for the procedure. In this problem, the algorithm estimates $\widehat{\boldsymbol{\Sigma}}$ and returns also $\boldsymbol{B} = (\boldsymbol{b}^{(1)}, \dots, \boldsymbol{b}^{(p)})$, the matrix where each column is the solution of the Lasso problem in eq. (1.51) for each column of $\boldsymbol{W}$. It is easy to recover $\boldsymbol{\Omega}$ since

$$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{W}_{11} & \boldsymbol{w}_{12} \\ \boldsymbol{w}_{21} & w_{22} \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{\Omega}_{11} & \boldsymbol{\omega}_{12} \\ \boldsymbol{\omega}_{21} & \omega_{22} \end{bmatrix} = \begin{bmatrix} I_{p-1} & 0 \\ 0 & 1 \end{bmatrix}, \tag{1.53}$$

and

$$\begin{aligned} \boldsymbol{\omega}_{12} &= -\boldsymbol{W}_{11}^{-1}\boldsymbol{w}_{12}\omega_{22} \\ \omega_{22} &= 1/(w_{22} - \boldsymbol{w}_{12}^T \boldsymbol{W}_{11}^{-1} \boldsymbol{w}_{12}). \end{aligned}$$

Therefore, for $j = 1, \dots, p$, the permuted target components of $\boldsymbol{\Omega}$ are

$$\begin{aligned} \boldsymbol{\omega}_{12} &= -\boldsymbol{b}^{(j)}\widehat{\omega}_{22} \\ \omega_{22} &= 1/(w_{22} - \boldsymbol{w}_{12}^T \boldsymbol{b}^{(j)}). \end{aligned}$$

In what follows, we will adapt these methods on a Gaussian mixture models, more precisely we will assume that each clusters present a sparse Gaussian concentration graph. We will rely on the Graphical Lasso for estimating the precision matrix and derive a EM algorithm.

```
Input: Matrix $\boldsymbol{S}$, parameter $\lambda$ and threshold $\varepsilon$
Output: Estimate of $\boldsymbol{W}$ and $\boldsymbol{B}$ a matrix of parameters.
Initialize $\boldsymbol{W}^{(0)} := \boldsymbol{S} + \lambda I$ and $\boldsymbol{B} = 0_{p\times p}$. The diagonal of $\boldsymbol{W}$ remained unchanged in what follows.
repeat
  for $j = 1, \dots, p$ do
    (a) Let $\boldsymbol{W}^{(j-1)}$ denote the current iterate. Solve the Lasso problem in eq. (1.51)
        $$\widehat{\boldsymbol{x}}^{(j-1)} = \arg\min_{\boldsymbol{x}} \frac{1}{2}\|(\boldsymbol{W}_{11}^{(j-1)})^{1/2}\boldsymbol{x} - \boldsymbol{b}\|_2^2 + \lambda\|\boldsymbol{x}\|_1, \quad (1.54)$$
        with $\boldsymbol{b} := (\boldsymbol{W}_{11}^{(j-1)})^{-1/2}\boldsymbol{s}_{12}$.
    (b) Update: $\boldsymbol{W}^{(j)}$ is $\boldsymbol{W}^{(j-1)}$ with $\boldsymbol{w}_{12} = \boldsymbol{W}_{11}^{(j-1)}\widehat{\boldsymbol{x}}^{(j-1)}$.
    (c) Save the parameter $\boldsymbol{x}^{(j-1)}$ in the $j^{th}$ column of $\boldsymbol{B}$.
    (d) Permute the columns and rows of $\boldsymbol{W}^{(j-1)}$ such that the $j^{th}$ column is $\boldsymbol{w}_{12}$, the next target.
  end for
  Let $\widehat{\boldsymbol{W}}^{(0)} := \boldsymbol{W}^{(p)}$.
until convergence occurs.
```

Figure 1.10: Graphical Lasso

Graphical Lasso on Gaussian mixtures

In this section, we present our contribution. We consider a Gaussian mixture model of K components and our task is to estimate the parameters $\boldsymbol{\theta} = (\theta_1, \dots, \theta_K)$ with $\theta_k = (\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Omega}_k)$ where $\boldsymbol{\Omega}_k$ is the precision matrix regarding the k^{th} component of the mixture. We denote $\varphi_{(\boldsymbol{\mu}_k,\boldsymbol{\Omega}_k)}$ the Gaussian density of mean $\boldsymbol{\mu}_k$ and precision matrix $\boldsymbol{\Omega}_k$. The penalized log-likelihood is

$$\ell_n^{pen}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - pen(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log\left\{\sum_{k=1}^{K} \pi_k \varphi_{(\boldsymbol{\mu}_k,\boldsymbol{\Omega}_k)}(\boldsymbol{x}_i)\right\} - pen(\boldsymbol{\theta}). \quad (1.55)$$

We suppose that each component of the mixture has a sparse Gaussian concentration graph. Therefore, in the scope of [Banerjee et al., 2008] and [Friedman et al., 2007], we consider an ℓ_1 regularization $pen(\theta_k) = \sum_{k=1}^{K} \lambda_k ||\boldsymbol{\Omega}_k||_{1,1}$ with $\lambda_k > 0$. The penalization of the log-likelihood concerns only the precision matrices $\boldsymbol{\Omega}_k$. Regarding the other parameters $(\pi_k, \boldsymbol{\mu}_k)$, our algorithm is the same as EM and we can use the same iteration technique as in lemma 1 to maximize the following cost function

$$F^{pen}(\boldsymbol{\theta}, \boldsymbol{\mathcal{T}}) = \sum_{k=1}^{K}\left(\sum_{i=1}^{n}\left\{\tau_{i,k}\log\varphi_{\boldsymbol{\mu}_k,\boldsymbol{\Omega}_k}(\boldsymbol{x}_i) + \tau_{i,k}\log(\pi_k/\tau_{i,k})\right\} - \lambda_k||\boldsymbol{\Omega}_k||_{1,1}\right). \quad (1.56)$$

The maximization of this function over $\boldsymbol{\theta}$ and $\boldsymbol{\mathcal{T}}$ leads to the two following optimization problems

Note: ajouter les domaines

$$\widehat{\boldsymbol{\theta}}(\boldsymbol{\mathcal{T}}) \in \arg\max_{\boldsymbol{\theta}} F^{pen}(\boldsymbol{\theta}, \boldsymbol{\mathcal{T}}), \qquad \widehat{\boldsymbol{\mathcal{T}}}(\boldsymbol{\theta}) \in \arg\max_{\boldsymbol{\mathcal{T}}} F^{pen}(\boldsymbol{\theta}, \boldsymbol{\mathcal{T}}). \tag{1.57}$$

For a given $\widehat{\boldsymbol{\mathcal{T}}}$, estimates of ($\pi_1, \dots, \pi_K$ and $\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_K$) obtained by the first optimization problem in eq. (1.57) are the same as in the EM algorithm

$$\widehat{\pi}_k = \frac{1}{n}\sum_{i=1}^{n} \widehat{\tau}_{i,k}, \quad \text{and} \quad \widehat{\boldsymbol{\mu}}_k = \frac{1}{n\widehat{\pi}_k}\sum_{i=1}^{n} \widehat{\tau}_{i,k}\boldsymbol{x}_i, \qquad \forall k \in [K] \tag{1.58}$$

And for a given $\widehat{\boldsymbol{\theta}}$, the estimate of $\boldsymbol{\mathcal{T}}$ obtained by the second optimization problem is

$$\widehat{\tau}_{i,k} = \frac{\widehat{\pi}_k \varphi_{\widehat{\boldsymbol{\mu}}_k, \widehat{\boldsymbol{\Omega}}_k}(\boldsymbol{x}_i)}{\sum_{k' \in [K]} \widehat{\pi}_{k'} \varphi_{\widehat{\boldsymbol{\mu}}_{k'}, \widehat{\boldsymbol{\Omega}}_{k'}}(\boldsymbol{x}_i)} = p_{\boldsymbol{\theta}}(Z = k | \boldsymbol{X} = \boldsymbol{x}_i), \qquad \forall k \in [K],\ \forall i \in [n]. \tag{1.59}$$

However, due to the penalty $\lambda_k ||\boldsymbol{\Omega}_k||_{1,1}$, the estimation of $\boldsymbol{\Omega}_k$ is not straightforward. We introduce the weighted empirical covariance matrix

$$\boldsymbol{\Sigma}_{n,k} = \frac{1}{n}\frac{\sum_{i=1}^{n} \tau_{i,k}(\boldsymbol{x}_i - \widehat{\boldsymbol{\mu}}_k)(\boldsymbol{x}_i - \widehat{\boldsymbol{\mu}}_k)^\top}{\sum_{i=1}^{n} \tau_{i,k}} \tag{1.60}$$

The Gaussian density in equation (1.56) can be expanded as follows

$$\begin{aligned} F^{pen}(\boldsymbol{\theta}, \boldsymbol{\mathcal{T}}) =& \sum_{k=1}^{K}\Bigg(\sum_{i=1}^{n}\Big\{\tau_{i,k}\Big(- \frac{p}{2}\log(2\pi) + \frac{1}{2}\log|\boldsymbol{\Omega}_k| \\ & - \frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T \boldsymbol{\Omega}_k (\boldsymbol{x}_i - \boldsymbol{\mu}_k)\Big) + \tau_{i,k}\log(\pi_k/\tau_{i,k})\Big\} - \lambda_k ||\boldsymbol{\Omega}_k||_{1,1}\Bigg) \\ =& - \frac{np}{2}\log(2\pi) + \sum_{k=1}^{K}\Bigg(\frac{n\pi_k}{2}\log|\boldsymbol{\Omega}_k| \\ & + \sum_{i=1}^{n}\Big\{ - \frac{\tau_{i,k}}{2}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T \boldsymbol{\Omega}_k (\boldsymbol{x}_i - \boldsymbol{\mu}_k) + \tau_{i,k}\log(\pi_k/\tau_{i,k})\Big\} - \lambda_k ||\boldsymbol{\Omega}_k||_{1,1}\Bigg). \end{aligned}$$

The opposite minimization problem regarding each $\boldsymbol{\Omega}_k$ is

$$\boldsymbol{\Omega}_k \in \arg\min_{\boldsymbol{\Omega} \succeq 0}\Big\{ - \frac{n\pi_k}{2}\log|\boldsymbol{\Omega}| + \frac{1}{2}\sum_{i=1}^{n}\tau_{i,k}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T \boldsymbol{\Omega}(\boldsymbol{x}_i - \boldsymbol{\mu}_k) + \lambda_k ||\boldsymbol{\Omega}||_{1,1}\Big\} \tag{1.61}$$

Using the well-known commutativity property of the trace operator and dividing by $n\pi_k$

$$\boldsymbol{\Omega}_k \in \arg\min_{\boldsymbol{\Omega} \succeq 0}\Big\{ - \frac{1}{2}\log|\boldsymbol{\Omega}| + \frac{1}{2}tr(\boldsymbol{\Sigma}_{n,k}\boldsymbol{\Omega}) + \frac{\lambda_k}{n\pi_k}||\boldsymbol{\Omega}||_{1,1}\Big\} \tag{1.62}$$

At the light of this equation, one can notice that we solves a graphical lasso problem within each cluster. We used a block coordinate ascent algorithm described in [Mazumder, 2012] to solve this convex problem. The alternating maximization procedure is summarized in Figure 1.11.

```
Input: data vectors $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n \in \mathbb{R}^p$ and the number of clusters $K$
Output: parameter estimate $\widehat{\boldsymbol{\theta}} = \{\widehat{\boldsymbol{\mu}}_k, \widehat{\boldsymbol{\Omega}}_k, \widehat{\pi}_k\}_{k\in[K]}$
Initialize $t = 0$, $\boldsymbol{\theta} = \boldsymbol{\theta}^0$.
Repeat
    Update the parameter $\boldsymbol{\mathcal{T}}$:
        $\tau_{i,k}^t = \frac{\pi_k^t \varphi_{\boldsymbol{\mu}_k^t,\boldsymbol{\Omega}_k^t}(\boldsymbol{x}_i)}{\sum_{k'\in[K]} \pi_{k'}^t \varphi_{\boldsymbol{\mu}_{k'}^t,\boldsymbol{\Omega}_{k'}^t}(\boldsymbol{x}_i)}.$
    Update the parameter $\boldsymbol{\theta}$:
        $\pi_k^{t+1} = \frac{1}{n}\sum_{i=1}^n \tau_{i,k}^t,$
        $\boldsymbol{\mu}_k^{t+1} = \frac{1}{n\pi_k^{t+1}}\sum_{i=1}^n \tau_{i,k}^t \boldsymbol{x}_i$
        $\boldsymbol{\Sigma}_{n,k} = \frac{1}{n^2\pi_k^{t+1}}\sum_{i=1}^n \tau_{i,k}^{t+1}(\boldsymbol{x}_i - \widehat{\boldsymbol{\mu}}_k^{t+1})(\boldsymbol{x}_i - \widehat{\boldsymbol{\mu}}_k^{t+1})^\top$
        $\boldsymbol{\Omega}_k^{t+1} \in \arg\min_{\boldsymbol{\Omega}\succeq 0}\Big\{ -\frac{1}{2}\log|\boldsymbol{\Omega}| + \frac{1}{2}tr(\boldsymbol{\Sigma}_{N,k}\boldsymbol{\Omega}) + \frac{\lambda_k}{n\pi_k^{t+1}}||\boldsymbol{\Omega}||_{1,1}\Big\}$
    increment $t$:   $t = t + 1$.
Until stopping rule.
Return $\boldsymbol{\theta}^t$.
```

Figure 1.11: Graphical lasso algorithm for Gaussian mixtures

1.4.2 Estimating the number of clusters

In this chapter, we will focus on the open problem of estimating the number of clusters. Most of current clustering methods such that K-Means, Expectation-Maximization with Gaussian mixture model or hierarchical clustering need a this parameter in input. Different methods are being used to perform a selection of the best model according to a criterion, unfortunately with a computational cost. In this work, we will try to tackle this challenge.

Our First method

The idea is to add a regularization term on the estimation of the $n \times K$ matrix $\boldsymbol{\mathcal{T}}$, the estimate of the number of clusters K will be the number of non-empty columns of $\boldsymbol{\mathcal{T}}$.

We consider a maximum number of clusters M, we note the convex set $A = \{\tau \in \mathbb{R}^M : \sum_{k=1}^{M} \tau_k = 1, \tau_k \geq 0 \quad \forall k \in [M]\}$ and the "indicator" function $\chi_A(.)$ defined by:

$$\chi_A(x) = \begin{cases} 0 & \text{if } x \in A, \\ \infty & \text{if } x = 0 \end{cases}$$

We note $\boldsymbol{\mathcal{T}}_{.,k}$ the k^{th} column and $\boldsymbol{\mathcal{T}}_{i,.}$ the i^{th} line of $\boldsymbol{\mathcal{T}}$. We will estimate $\boldsymbol{\mathcal{T}}$ using the same equation 1.56, 1.57 with a regularization term:

$$F^{pen}(\boldsymbol{\theta}, \boldsymbol{\mathcal{T}}) = \sum_{k=1}^{K} \left(\sum_{i=1}^{n} \left\{ \tau_{i,k} \log \varphi_{\boldsymbol{\mu}_k, \boldsymbol{\Omega}_k}(\boldsymbol{x}_i) + \tau_{i,k} \log(\pi_k / \tau_{i,k}) \right\} - \lambda_k ||\boldsymbol{\Omega}_k||_{1,1} \right) + \sum_{k=1}^{K} ||\boldsymbol{\mathcal{T}}_{.,k}||_2 + \sum_{i=1}^{n} \chi_A(\boldsymbol{\mathcal{T}}_{i,.})$$

Removing the penalization on $\boldsymbol{\Omega}$:

$$F^{pen}(\boldsymbol{\theta}, \boldsymbol{\mathcal{T}}) = \sum_{k=1}^{K} \left(\sum_{i=1}^{n} \left\{ \tau_{i,k} \log \varphi_{\boldsymbol{\mu}_k, \boldsymbol{\Omega}_k}(\boldsymbol{x}_i) + \tau_{i,k} \log(\pi_k / \tau_{i,k}) \right\} + \sum_{k=1}^{K} ||\boldsymbol{\mathcal{T}}_{.,k}||_2 + \sum_{i=1}^{n} \chi_A(\boldsymbol{\mathcal{T}}_{i,.})$$

and the optimization problem:

$$\widehat{\boldsymbol{\mathcal{T}}}(\boldsymbol{\theta}) \in \arg\max_{\boldsymbol{\mathcal{T}}} F^{pen}(\boldsymbol{\theta}, \boldsymbol{\mathcal{T}}) \tag{1.63}$$

Unfortunately, the regularization term prevents to derive explicit solution as in previous chapters. Furthermore, we cant separate the objective function since we optimize along columns and lines of $\boldsymbol{\mathcal{T}}$. The objective function $F^{pen}(\boldsymbol{\theta}, \boldsymbol{\mathcal{T}})$ rewritten $F^{pen}_{\boldsymbol{\theta}}(\boldsymbol{\mathcal{T}})$ can be split into two terms:

$$F^{pen}_{\boldsymbol{\theta}}(\boldsymbol{\mathcal{T}}) = f(\boldsymbol{\mathcal{T}}) + g(\boldsymbol{\mathcal{T}}) \tag{1.64}$$

```
Input:
Output: parameter estimate $\boldsymbol{\mathcal{T}}$
Initialize $t_1 = 1$ and $\boldsymbol{\xi}^0$ with
    $\xi^0_{i,k} = \frac{\pi^0_k \varphi_{\boldsymbol{\mu}^0_k, \boldsymbol{\Omega}^0_k}(\boldsymbol{x}_i)}{\sum_{k' \in [K]} \pi^0_{k'} \varphi_{\boldsymbol{\mu}^0_{k'}, \boldsymbol{\Omega}^0_{k'}}(\boldsymbol{x}_i)}$
Repeat
    $\boldsymbol{\mathcal{T}}^k = \underset{\boldsymbol{\mathcal{T}}: \forall K, \boldsymbol{\mathcal{T}}^k \in A}{\arg\min} \left( ||\boldsymbol{\mathcal{T}} - (\boldsymbol{\xi}^k - \lambda \nabla f(\boldsymbol{\xi}^k))||^2_2 \right)$
    $t^{k+1} = \frac{1 + \sqrt{1 + 4 * (t^k)^2}}{2}$
    $\boldsymbol{\xi}^{k+1} = \boldsymbol{\mathcal{T}}^k + \left( \frac{t^k - 1}{t^{k+1}} \right) (\boldsymbol{\mathcal{T}}^k - \boldsymbol{\mathcal{T}}^{k-1})$
```

Figure 1.12: $\boldsymbol{\mathcal{T}}$ estimation with FISTA

with:

$$\begin{aligned} f(\boldsymbol{\mathcal{T}}) =& \sum_{k=1}^{K} \left(\sum_{i=1}^{n} \left\{ \tau_{i,k} \log \varphi_{\boldsymbol{\mu}_k, \boldsymbol{\Omega}_k}(\boldsymbol{x}_i) + \tau_{i,k} \log(\pi_k / \tau_{i,k}) \right\} + \sum_{k=1}^{K} ||\boldsymbol{\mathcal{T}}_{.,k}||_2 \right. \\ g(\boldsymbol{\mathcal{T}}) =& \sum_{i=1}^{n} \chi_A(\boldsymbol{\mathcal{T}}_{i,.}) \end{aligned}$$

f is convex and differentiable on its domain, g is also convex but not smooth. We will tackle this problem by using a proximal method:

$$\begin{aligned} \boldsymbol{\mathcal{T}}^{k+1} =& \mathbf{prox}_{\lambda g}(\boldsymbol{\mathcal{T}}^k - \lambda \nabla f(\boldsymbol{\mathcal{T}}^k)) \ = & P_A(\boldsymbol{\mathcal{T}}^k - \lambda \nabla f(\boldsymbol{\mathcal{T}}^k)) \\ =& \underset{\boldsymbol{\mathcal{T}}: \forall K, \boldsymbol{\mathcal{T}}^k \in A}{\arg\min} \left(||\boldsymbol{\mathcal{T}} - (\boldsymbol{\mathcal{T}}^k - \lambda \nabla f(\boldsymbol{\mathcal{T}}^k))||^2_2 \right) \end{aligned}$$

The gradient of f on $\boldsymbol{\mathcal{T}}$ is given by:

$$\begin{aligned} \left[\nabla_{\boldsymbol{\mathcal{T}}} f(\boldsymbol{\mathcal{T}}) \right]_{i,j} =& \left[\frac{\partial f}{\partial \boldsymbol{\mathcal{T}}_{ij}}(\boldsymbol{\mathcal{T}}) \right]_{i,j} \\ =& \log(\varphi_{\boldsymbol{\mu}_j, \boldsymbol{\Omega}_j}(x_i)) + \log(\frac{\pi_j}{\tau_{i,j}}) + \frac{\tau_{i,j}}{||\boldsymbol{\mathcal{T}}_{.,j}||_2} - 1 \end{aligned}$$

We will use FISTA to accelerate the convergence We use the algorithm of last chapter with the new estimation procedure of $\boldsymbol{\mathcal{T}}$

1.4.3 Sparse Weights Vector Estimation

We fit a model with an arbitrarily large number of components K and penalyze the weights vector $\boldsymbol{\pi}$. The penalized negative log-likelihood is:

$$\ell_n(\boldsymbol{\theta}) = -\frac{1}{n} \sum_{i=1}^{n} \log \left\{ \sum_{j=1}^{K} \pi_k \varphi_{(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}(\boldsymbol{x}_i) \right\} + \lambda \sum_{j=1}^{K-1} \pi_j^{1/\gamma} \quad \gamma \geq 1 \qquad (1.65)$$

```
Input: data vectors $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n \in \mathbb{R}^p$ and the number of clusters $K$
Output: parameter estimate $\widehat{\boldsymbol{\theta}} = \{\widehat{\boldsymbol{\mu}}_k, \widehat{\boldsymbol{\Omega}}_k, \widehat{\pi}_k\}_{k \in [K]}$
Initialize $t = 0$, $\boldsymbol{\theta} = \boldsymbol{\theta}^0$.
Repeat
    Update the parameter $\boldsymbol{\mathcal{T}}$ with previous algorithm
    Update the parameter $\boldsymbol{\theta}$:
        $\pi_k^{t+1} = \frac{1}{n} \sum_{i=1}^{n} \tau_{i,k}^{t},$
        $\boldsymbol{\mu}_k^{t+1} = \frac{1}{n\pi_k^{t+1}} \sum_{i=1}^{n} \tau_{i,k}^{t} \boldsymbol{x}_i$
        $\boldsymbol{\Sigma}_{n,k} = \frac{1}{n^2\pi_k^{t+1}} \sum_{i=1}^{n} \tau_{i,k}^{t+1} (\boldsymbol{x}_i - \widehat{\boldsymbol{\mu}}_k^{t+1})(\boldsymbol{x}_i - \widehat{\boldsymbol{\mu}}_k^{t+1})^\top$
        $\boldsymbol{\Omega}_k^{t+1} \in \arg\min_{\boldsymbol{\Omega} \succeq 0} \Big\{ -\frac{1}{2} \log |\boldsymbol{\Omega}| + \frac{1}{2} tr(\boldsymbol{\Sigma}_{N,k} \boldsymbol{\Omega}) + \frac{\lambda_k}{n\pi_k^{t+1}} ||\boldsymbol{\Omega}||_{1,1} \Big\}$
    increment $t$: $t = t + 1$.
Until stopping rule.
Return $\boldsymbol{\theta}^t$.
```

Figure 1.13: Graphical lasso algorithm for Gaussian mixtures with cluster number discovery

Such that:

$$\sum_{j}^{K-1}\pi_j \leq 1 \quad \text{and} \quad \pi_K = 1-\sum_{j}^{K-1}\pi_j \tag{1.66}$$

and $\sum_j^{K-1}\pi_j^{1/\gamma}$ is not convex, to rectify it let note $\alpha_j = \pi^{1/\gamma}$, then:

$$\widehat{\boldsymbol{\alpha}} \in \underset{\boldsymbol{\alpha}\in\mathbb{R}^{K-1}}{\arg\min}\left\{-\frac{1}{n}\sum_{i=1}^{n}\log\left\{\sum_{j=1}^{K}\alpha_j^{\gamma}\varphi_{(\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)}(\boldsymbol{x}_i)\right\}+\lambda\sum_{j=1}^{K-1}\alpha_j\right\} \quad \gamma\geq 1, \tag{1.67}$$

such that: $\sum_j^{K-1}\alpha_j^{\gamma} \leq 1$ and $\alpha_K^{\gamma} = 1-\sum_j^{K-1}\alpha_j^{\gamma}$. We denote $f_{\boldsymbol{\theta}}(\boldsymbol{\alpha})$ this cost function.

If we note A the $K-1$ dimensional unit sphere and χ_A the indicator function of A (0 in A, ∞ elsewhere), the minimization problem can be rewritten as

$$\widehat{\boldsymbol{\alpha}} \in \underset{\boldsymbol{\alpha}\in\mathbb{R}^{K-1}}{\arg\min}\{f_{\boldsymbol{\theta}}(\boldsymbol{\alpha})+\chi_A(\boldsymbol{\alpha})\}. \tag{1.68}$$

To solve this minimization problem, we can use a proximal gradient method and Nesterov acceleration for the following iterative procedure:

$$\widehat{\alpha}^{t+1} = \text{prox}_{\chi_A}(\boldsymbol{\alpha}^t - h\nabla f_{\boldsymbol{\theta}}(\boldsymbol{\alpha}^t)) \tag{1.69}$$

$$= \underset{x\in\mathbb{R}^{K-1}}{\arg\min}\left\{\chi_A(x)+\frac{1}{2}||x-(\boldsymbol{\alpha}^t - h\nabla f_{\boldsymbol{\theta}}(\boldsymbol{\alpha}^t))||^2\right\} \tag{1.70}$$

$$= P_A(\boldsymbol{\alpha}^t - h\nabla f_{\boldsymbol{\theta}}(\boldsymbol{\alpha}^t)). \tag{1.71}$$

This iteration procedure gives us the following algorithm

```
Input: $\boldsymbol{\theta}$
Output: parameter estimate $\widehat{\boldsymbol{\pi}} = (\boldsymbol{\alpha}_1^{\gamma},...,\boldsymbol{\alpha}_{K-1}^{\gamma}, 1-\sum_{j=1}^{K-1}\boldsymbol{\alpha}_j^{\gamma})^t$
Initialize $t=0$, $s_0=1$ and $\boldsymbol{\xi}^0 = (\boldsymbol{\pi}_1^{1/\gamma},...,\boldsymbol{\pi}_{K-1}^{1/\gamma})$

Repeat

    $\boldsymbol{\alpha}^t = P_A(\xi^t - h\nabla f_{\boldsymbol{\theta}}(\xi^t))$    (1.72)
    $s_{t+1} = \frac{1+\sqrt{1+4*s_t^2}}{2}$    (1.73)
    $\boldsymbol{\xi}^{t+1} = \boldsymbol{\alpha}^t + \left(\frac{s_t-1}{s_{t+1}}\right)(\boldsymbol{\alpha}^t - \boldsymbol{\alpha}^{t-1})$    (1.74)

        increment $t$:   $t=t+1$.
Until stopping rule.
```

Figure 1.14: Estimation of α

and the final algorithm for estimating the gaussian mixture with a penalized weight vector is

```
**Input:** data vectors $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n \in \mathbb{R}^p$ and a large number of clusters $K$
**Output:** parameter estimate $\widehat{\boldsymbol{\theta}} = \{\widehat{\boldsymbol{\mu}}_k, \widehat{\boldsymbol{\Sigma}}_k, \widehat{\pi}_k\}_{k\in[K]}$ Initialize $t = 0$, $\boldsymbol{\theta} = \boldsymbol{\theta}^0$
Initialize $t = 0$, $\boldsymbol{\theta} = \boldsymbol{\theta}^0$
**Repeat**
    Update the parameter $\mathcal{T}$
        $$\tau_{i,k}^t = \frac{\pi_k^t \varphi_{\boldsymbol{\mu}_k^t, \boldsymbol{\Omega}_k^t}(\boldsymbol{x}_i)}{\sum_{k'\in[K]} \pi_{k'}^t \varphi_{\boldsymbol{\mu}_{k'}^t, \boldsymbol{\Omega}_{k'}^t}(\boldsymbol{x}_i)}. \quad (1.75)$$
    Update parameters $\boldsymbol{\mu}, \boldsymbol{\Sigma}$.
        $$\boldsymbol{\mu}_k^{t+1} = \frac{1}{n\pi_k^{t+1}} \sum_{i=1}^n \tau_{i,k}^t \boldsymbol{x}_i, \quad (1.76)$$
        $$\boldsymbol{\Sigma}_k^{t+1} = \frac{1}{n\pi_k^{t+1}} \sum_{i=1}^n \tau_{i,k}^t (\boldsymbol{x}_i - \boldsymbol{\mu}_k^{t+1})(\boldsymbol{x}_i - \boldsymbol{\mu}_k^{t+1})^\top. \quad (1.77)$$
    Update the parameter $\pi$ with previous algorithm
    increment $t$:  $t = t + 1$
**Until** stopping rule.
```

Figure 1.15: Algorithm for estimating sparse weights vector on GMM

Ci-dessous, les résultats de l'algorithme d'estimation parcimonieuse des poids du mélange sur des données simulées. En vert notre algorithme et en rouge la méthode EM+BIC. En abscisse le nombre de vrais clusters, K. En ordonnée, le logarithme de l'erreur $\|\widehat{\boldsymbol{\pi}} - \boldsymbol{\pi}^*\|_1$. Pour chaque K, 50 simulations ont été effectuées. Nous représentons les premiers et troisièmes quartiles ainsi que la médiane.

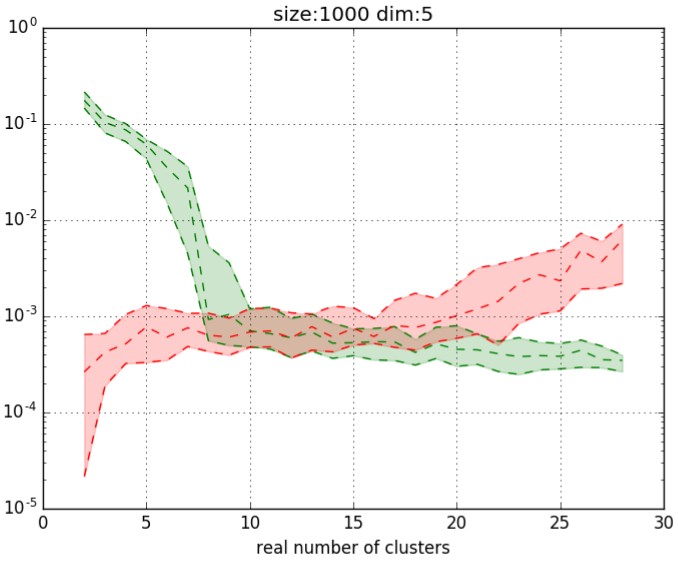


Figure 1.16: Vert: Notre algorithme. Rouge: EM+BIC

1.5 A word on density estimation

Bibliography

C. Hennig, M. Meila, F. Murtagh, and R. Rocci. Handbook of Cluster Analysis. Chapman & Hall/CRC Handbooks of Modern Statistical Methods. Taylor & Francis, 2015. ISBN 9781466551886.

J. MacQueen. Some methods for classification and analysis of multivariate observations. In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics, pages 281–297, Berkeley, Calif., 1967. University of California Press.

S. Dasgupta. The Hardness of K-means Clustering. Technical report (University of California, San Diego. Department of Computer Science and Engineering). Department of Computer Science and Engineering, University of California, San Diego, 2008.

Daniel Aloise, Amit Deshpande, Pierre Hansen, and Preyas Popat. Np-hardness of euclidean sum-of-squares clustering. Mach. Learn., 75(2):245–248, May 2009. ISSN 0885-6125. doi: 10.1007/s10994-009-5103-0. URL http://dx.doi.org/10.1007/s10994-009-5103-0.

Mary Inaba, Naoki Katoh, and Hiroshi Imai. Applications of weighted voronoi diagrams and randomization to variance-based k-clustering: (extended abstract). In Proceedings of the Tenth Annual Symposium on Computational Geometry, SCG '94, pages 332–339, New York, NY, USA, 1994. ACM. ISBN 0-89791-648-4. doi: 10.1145/177424.178042. URL http://doi.acm.org/10.1145/177424.178042.

S. Lloyd. Least squares quantization in pcm. IEEE Transactions on Information Theory, 28(2):129–137, March 1982. ISSN 0018-9448. doi: 10.1109/TIT.1982.1056489.

David Arthur and Sergei Vassilvitskii. How slow is the k-means method? In Proceedings of the Twenty-second Annual Symposium on Computational Geometry, SCG '06, pages 144–153, New York, NY, USA, 2006. ACM. ISBN 1-59593-340-9. doi: 10.1145/1137856.1137880. URL http://doi.acm.org/10.1145/1137856.1137880.

David Arthur and Sergei Vassilvitskii. K-means++: The advantages of careful seeding. In Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '07, pages 1027–1035, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics. ISBN 978-0-898716-24-5. URL http://dl.acm.org/citation.cfm?id=1283383.1283494.

Vladimir Makarenkov and Pierre Legendre. Optimal variable weighting for ultrametric and additive trees and k-means partitioning: Methods and software. Journal of Classification, 18(2):245–271, 2001.

Joshua Zhexue Huang, Jun Xu, Michael Ng, and Yunming Ye. Weighting Method for Feature Selection in K-Means. Chapman and Hall/CRC, 2007. ISBN 978-1-58488-878-9. doi: doi:10.1201/9781584888796.ch10.

Renato Cordeiro de Amorim and Boris Mirkin. Minkowski metric, feature weighting and anomalous cluster initializing in k-means clustering. Pattern Recogn., 45(3): 1061–1075, March 2012. ISSN 0031-3203. doi: 10.1016/j.patcog.2011.08.012. URL http://dx.doi.org/10.1016/j.patcog.2011.08.012.

L. Kaufman and Peter J. Rousseeuw. Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley, 1990.

D. Sculley. Web-scale k-means clustering. In Proceedings of the 19th International Conference on World Wide Web, WWW '10, pages 1177–1178, 2010. ISBN 978-1-60558-799-8.

R. Ostrovsky, Y. Rabani, L. J. Schulman, and C. Swamy. The effectiveness of lloyd-type methods for the k-means problem. In 2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06), pages 165–176, Oct 2006. doi: 10.1109/FOCS.2006.75.

A. Guénoche, P. Hansen, and B. Jaumard. Efficient algorithms for divisive hierarchical clustering with the diameter criterion. Journal of Classification, 8 (1):5–30, Jan 1991. ISSN 1432-1343. doi: 10.1007/BF02616245. URL https://doi.org/10.1007/BF02616245.

R. L. Graham and Pavol Hell. On the history of the minimum spanning tree problem. IEEE Ann. Hist. Comput., 7(1):43–57, January 1985. ISSN 1058-6180. doi: 10.1109/MAHC.1985.10011. URL https://doi.org/10.1109/MAHC.1985.10011.

Fionn Murtagh and Pedro Contreras. Algorithms for hierarchical clustering: an overview. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2(1):86–97, 2012. ISSN 1942-4795. doi: 10.1002/widm.53. URL http://dx.doi.org/10.1002/widm.53.

Joe H. Ward Jr. Hierarchical grouping to optimize an objective function. Journal of the American Statistical Association, 58(301):236–244, 1963. doi: 10.1080/01621459.1963.10500845.

G. N. Lance and W. T. Williams. A general theory of classificatory sorting strategies1. hierarchical systems. The Computer Journal, 9(4):373–380, 1967. doi: 10.1093/comjnl/9.4.373. URL +http://dx.doi.org/10.1093/comjnl/9.4.373.

Vladimir Batagelj. Generalized ward and related clustering problems. In In H.H. Bock (Ed.), Classification and Related Methods of Data Analysis, pages 67–74. North-Holland, 1988.

F. Murtagh. Multidimensional clustering algorithms. 1985.

M. Jambu. Exploration informatique et statistique des données. Collection technique et scientifique des télécommunications. Dunod, 1989. ISBN 9782040188405.

W. E. Donath and A. J. Hoffman. Lower bounds for the partitioning of graphs. IBM J. Res. Dev., 17(5):420–425, September 1973. ISSN 0018-8646. doi: 10.1147/rd.175.0420. URL http://dx.doi.org/10.1147/rd.175.0420.

Miroslav Fiedler. Algebraic connectivity of graphs. Czechoslovak Mathematical Journal, 23(2):298–305, 1973.

Ulrike Luxburg. A tutorial on spectral clustering. Statistics and Computing, 17(4):395–416, December 2007. ISSN 0960-3174. doi: 10.1007/s11222-007-9033-z. URL http://dx.doi.org/10.1007/s11222-007-9033-z.

Daniel A. Spielman and Shang-Hua Teng. Spectral partitioning works: Planar graphs and finite element meshes. Linear Algebra and its Applications, 421(2):284 – 305, 2007. ISSN 0024-3795. doi: https://doi.org/10.1016/j.laa.2006.07.020. Special Issue in honor of Miroslav Fiedler.

Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Mach. Intell., 22(8):888–905, August 2000. ISSN 0162-8828. doi: 10.1109/34.868688. URL http://dx.doi.org/10.1109/34.868688.

Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS, pages 849–856. MIT Press, 2001.

L. Hagen and A. B. Kahng. New spectral methods for ratio cut partitioning and clustering. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 11(9):1074–1085, Sep 1992. ISSN 0278-0070. doi: 10.1109/43.159993.

Dorothea Wagner and Frank Wagner. Between Min Cut and Graph Bisection, pages 744–750. Springer Berlin Heidelberg, Berlin, Heidelberg, 1993. ISBN 978-3-540-47927-7. doi: 10.1007/3-540-57182-5_65. URL https://doi.org/10.1007/3-540-57182-5_65.

Stephen Guattery and Gary L. Miller. On the quality of spectral separators. SIAM Journal on Matrix Analysis and Applications, 19(3):701–719, 1998. doi: 10.1137/S0895479896312262. URL https://doi.org/10.1137/S0895479896312262.

Boaz Nadler and Meirav Galun. Fundamental limitations of spectral clustering. In in Advanced in Neural Information Processing Systems 19, B. Schölkopf and, pages 1017–1024, 2007.

André Hardy. On the number of clusters. Comput. Stat. Data Anal., 23(1):83–96, November 1996. ISSN 0167-9473. doi: 10.1016/S0167-9473(96)00022-9. URL http://dx.doi.org/10.1016/S0167-9473(96)00022-9.

Glenn W. Milligan and Martha C. Cooper. An examination of procedures for determining the number of clusters in a data set. Psychometrika, 50(2):159–179, Jun 1985. ISSN 1860-0980. doi: 10.1007/BF02294245. URL https://doi.org/10.1007/BF02294245.

Peter Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math., 20(1):53–65, November 1987. ISSN 0377-0427. doi: 10.1016/0377-0427(87)90125-7. URL http://dx.doi.org/10.1016/0377-0427(87)90125-7.

Robert Tibshirani, Guenther Walther, and Trevor Hastie. 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, 2001. ISSN 1467-9868. doi: 10.1111/1467-9868.00293. URL http://dx.doi.org/10.1111/1467-9868.00293.

Gideon Schwarz. Estimating the dimension of a model. Ann. Statist., 6(2):461–464, 03 1978. doi: 10.1214/aos/1176344136. URL http://dx.doi.org/10.1214/aos/1176344136.

H. Akaike. Information theory and an extension of the maximum likelihood principle. In B. N. Petrov and F. Csaki, editors, Second International Symposium on Information Theory, pages 267–281, Budapest, 1973. Akadémiai Kiado.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society. Series B (Methodological), 39, No. 1:1–38, 1977.

Richard Bellman. Dynamic Programming. Princeton University Press, Princeton, NJ, USA, 1 edition, 1957. URL http://books.google.com/books?id=fyVtp3EMxasC&pg=PR5&dq=dynamic+programming+richard+e+bellman&client=firefox-a#v=onepage&q=dynamic%20programming%20richard%20e%20bellman&f=false.

Arthur Zimek, Erich Schubert, and Hans-Peter Kriegel. A survey on unsupervised outlier detection in high-dimensional numerical data. Statistical Analysis and Data Mining, 5(5):363–387, 2012. ISSN 1932-1872. doi: 10.1002/sam.11161. URL http://dx.doi.org/10.1002/sam.11161.

Charles Bouveyron and Camille Brunet. Model-Based Clustering of High-Dimensional Data: A review. Computational Statistics and Data Analysis, 71:52–78, 2013. doi: 10.1016/j.csda.2012.12.008. URL https://hal.archives-ouvertes.fr/hal-00750909.

Lance Parsons, Ehtesham Haque, and Huan Liu. Subspace clustering for high dimensional data: A review. SIGKDD Explor. Newsl., 6(1):90–105, June 2004. ISSN 1931-0145. doi: 10.1145/1007730.1007731. URL http://doi.acm.org/10.1145/1007730.1007731.

C. Giraud. Introduction to High-Dimensional Statistics. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. Taylor & Francis, 2014. ISBN 9781482237948.

J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. Biostatistics, 2007.

O. Banerjee, L. El Ghaoui, and A. d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. Journal of Machine Learning Research, 2008.

Ming Yuan and Yi Lin. Model selection and estimation in the gaussian graphical model. Biometrika, 94(1):19, 2007. doi: 10.1093/biomet/asm018. URL +http://dx.doi.org/10.1093/biomet/asm018.

Nicolai Meinshausen and Peter Bühlmann. High-dimensional graphs and variable selection with the lasso. Ann. Statist., 34(3):1436–1462, 06 2006. doi: 10.1214/009053606000000281. URL http://dx.doi.org/10.1214/009053606000000281.

D. Edwards. Introduction to Graphical Modelling. Springer Texts in Statistics. Springer New York, 2000. ISBN 9780387950549.

A. P. Dempster. Covariance selection. Biometrics, 28(1):157–175, 1972. ISSN 0006341X, 15410420. URL http://www.jstor.org/stable/2528966.

Leo Breiman. Heuristics of instability and stabilization in model selection. Ann. Statist., 24(6):2350–2383, 12 1996. doi: 10.1214/aos/1032181158. URL http://dx.doi.org/10.1214/aos/1032181158.

R. Mazumder. Topics in sparse multivariate statistics (thesis). 2012.

universite PARIS-SACLAY

École doctorale
de mathématiques
Hadamard (EDMH)

Titre : Titre en français

Mots Clefs : Mettre de 3 à 6 mots clefs

Résumé :

Title : Title in english

Keys words : 3 to 6 key words (in english)

Abstract :