# Thesis

Mehdi Sebbar

2017

# Contents

# Todo list

# Chapter 1

# Introduction

## Contents

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 groups have small dissimilarities compared to outside of the group.

The literature is rich on this topic, with different approaches coming from statistics and computer science. We will give a glimpse on 4 well-known techniques, $K$-means, Hierarchical clustering, Spectral clustering and EM on Gaussian mixtures model which will be our topic of main interest. The reader can refer to [Hennig et al., 2015] for an extensive review of the clustering problem.

### 1.1.1 Dimensions in Clustering

Dimensions given in [Hennig et al., 2015]

- By Type of Clustering: Hard vs. Soft: eg K-means vs GMM
- By Type of Clustering: Flat vs. Hierarchical eg Hierarchical Clsutering
- By Data Type or Format We could collect data about each student's grades, study habits, and so on, and use these variables for the purpose of clustering. Alternatively, we could collect data about how students interact with each other, like the network of friendships and collaboration in the class. In the former case, we describe the data points by a vector of features; in the latter, we describe the (pairwise) relations between data points. Hence, the latter is distinguished by terms such as relational clustering or similarity-based clustering.
- By Clustering Criterion: (Probabilistic) Model-Based vs. Cost-Based
- By Regime: Parametric (K Is Input) vs. Nonparametric (Smoothness Parameter Is Input)

### 1.1.2 Clustering Approaches

- Centroid-Based Clustering
- 1.5.2 Agglomerative Hierarchical Methods
- 1.5.3 Spectral Clustering
- 1.5.4 Mixture Probability Models
- 1.5.5 Density-Based Clustering

### 1.1.3 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, $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n$ in $\mathbb{R}^p$, the goal of $K$-means is to find a set of centers $\mathcal{C} = \{\boldsymbol{c}_1, \dots, \boldsymbol{c}_K\}$ that minimize the following objective function:

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

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 $\mathbb{R}^p$ $\mathcal{D} = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_n\}$, $K$ points $\boldsymbol{c}_1, \dots, \boldsymbol{c}_K$ 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 the algorithm are necessary to insure an acceptable clustering. Note that Lloyd's algorithm has several drawbacks:

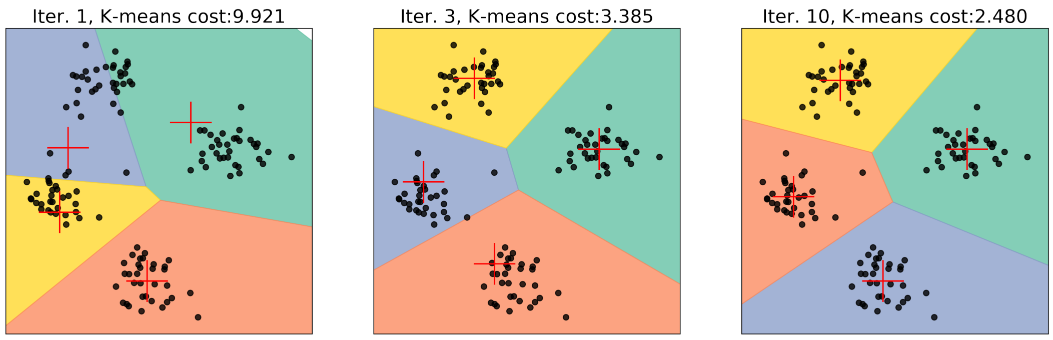


Figure 1.1: Llyod's algorithm with random initialization centers and final Voronoi partitions at different steps: with 1 iteration (left), 3 (middle) and 10 (right) iterations (the algorithm converged). $K$-means costs are given on top.

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 initializa-

```
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 $\widehat{\boldsymbol{c}}_1, \dots, \widehat{\boldsymbol{c}}_K$:
       $$\widehat{\boldsymbol{c}}_i = \frac{1}{|C_i|} \sum_{x_j \in C_i} x_j \quad (1.3)$$
  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

tions 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
    1: 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
2: 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.

### 1.1.4 Agglomerative Hierarchical Methods

The idea of Hierarchical clustering is to form a hierarchy of clusters (i.e. nested clusters) which might be interesting to see how clusters are related to each other (something we cannot see with $K$-means). This method of clustering is very popular due to its simplicity and the resulting structure and organization. There exists two types: 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 $O(2^n)$ , 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 $S = (d_{ij} = d(i,j))_{i,j \in [N]^2}$ (which is symmetric) and consider the couple $(i,j)$ such that $d_{ij}$ is the smallest dissimilarity in $S$. We create a new object $i \cup j$, 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 is detailed in Figure 1.5. This procedure has a complexity of $O(n^2)$[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 Figure 1.5. Lance and Williams developed an updating formula [Lance and Williams, 1967] for the dissimilarities that generalize several methods of agglomerative methods.

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

For instance, the single-linkage method is recovered by setting $\alpha_i = \alpha_j = 1/2$, $\beta = 0$ and $\gamma = -1/2$ and the complete-linkage method with $\alpha_i = \alpha_j = 1/2$, $\beta = 0$ and $\gamma = 1/2$. The reader can find parameters for other methods in Table 6.1 of [Hennig et al., 2015]. Another popular method worth mentioning

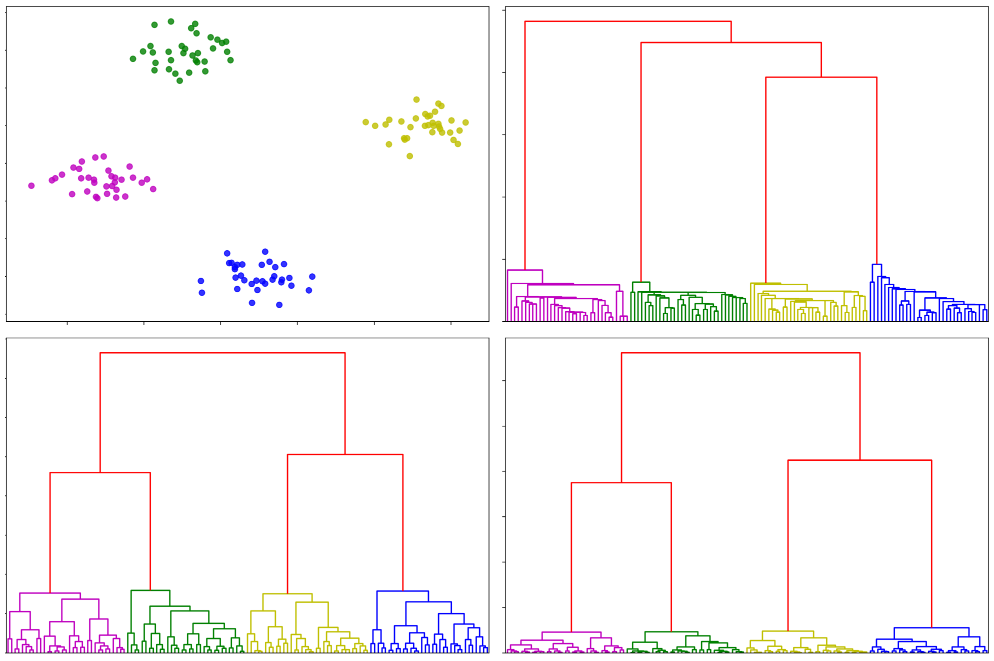


Figure 1.4: The dataset with 4 clusters (top-left) used with the Agglomerative hierarchical clustering and its corresponding dendrogram, single-link (top-right), complete-link (bottom-left) and Ward's method (bottom-right). A simple way for finding clusters would be to cut the dendrogram with a horizontal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

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 such as the sum of squares. Consider the merging cost of combining clusters $A$ and $B$:

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

where $\boldsymbol{c}_i$ is the center of cluster $i$. This quantity is positive, hence the within-group variance increases when merging two clusters. Ward's method seek to minimize this growth. Note that Ward's method can be expressed in the Lance-Williams framework for evaluating $d(A \cup B, K)$ with $\alpha_i = (n_i +$

$n_K)/(n_A + n_B + n_K)$, $\beta = -n_K/(n_A + n_B + n_K)$ and $\gamma = 0$. More efficient algorithms relies on 'Nearest Neighbor Chains'. Density based clustering

```
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

### 1.1.5 Spectral clustering

### 1.1.6 Finding the number of clusters

## 1.2 The Gaussian mixture model

The Gaussian mixture model is an important framework for clustering problems. It assumes that the observations are drawn from a mixture distribution the components of which are Gaussian with parameters $(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$:

$$\varphi_{\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k}(x) = \frac{1}{(2\pi)^{p/2}|\boldsymbol{\Sigma}_k|^{1/2}} \exp\Big( -\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_k)\Big) \tag{1.7}$$

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

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k \varphi_{\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k}(\boldsymbol{x}), \qquad \forall \boldsymbol{x} \in \mathbb{R}^p, \tag{1.8}$$

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 $X$:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{k=1}^{K} \mathbf{P}(Z = k) p_{\theta}(\boldsymbol{x}|Z = k) = \sum_{k=1}^{K} \pi_k \varphi_{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k}(\boldsymbol{x}), \qquad \forall \boldsymbol{x} \in \mathbb{R}^p. \tag{1.9}$$

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

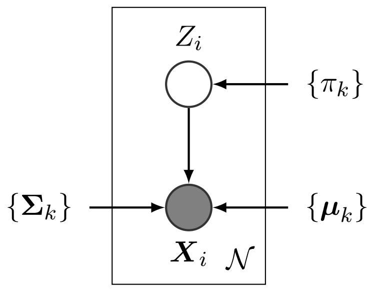
$$\tau_k(\boldsymbol{x}, \boldsymbol{\theta}) = \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}|Z = k)\mathbf{P}(Z = k)}{p_{\boldsymbol{\theta}}(\boldsymbol{x})} = \frac{\pi_k \varphi_{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k}(\boldsymbol{x})}{\sum_{k'=1}^{K} \pi_{k'} \varphi_{\boldsymbol{\mu}_{k'}, \boldsymbol{\Sigma}_{k'}}(\boldsymbol{x})} \tag{1.10}$$

Since the true value of the parameter $\boldsymbol{\theta}$ is not available, formula (1.10) 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

$$\widehat{g}(\boldsymbol{x}) = g^*_{\widehat{\boldsymbol{\theta}}}(\boldsymbol{x}) = \arg\max_{k \in [K]} \tau_k(\boldsymbol{x}, \widehat{\boldsymbol{\theta}}) = \arg\max_{k \in [K]} \widehat{\pi}_k \varphi_{\widehat{\boldsymbol{\mu}}_k, \widehat{\boldsymbol{\Sigma}}_k}(\boldsymbol{x}). \tag{1.11}$$

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 (1.8). The following graphical model depicts the scheme of the observations:



The log-likelihood of the Gaussian mixture model is

$$\ell_n(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_i) = \sum_{i=1}^{n} \log \left\{ \sum_{k=1}^{K} \pi_k \varphi_{(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}(\boldsymbol{x}_i) \right\}. \tag{1.12}$$

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 (1.12) 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:
```

$$\widehat{\boldsymbol{\theta}} \in \arg\max_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \ell(\boldsymbol{\theta} | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \arg\max_{\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}} \sum_{i=1}^{n} \log \left\{ \sum_{k=1}^{K} \pi_k \varphi_{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k}(\boldsymbol{x}_i) \right\}. \tag{1.13}$$

```
2:  Output the clustering rule:
```

$$\widehat{g}(\cdot) = \arg\max_{k \in [K]} \widehat{\pi}_k \varphi_{\widehat{\boldsymbol{\mu}}_k, \widehat{\boldsymbol{\Sigma}}_k}(\cdot). \tag{1.14}$$

Figure 1.6: Clustering under Gaussian mixture modeling

### 1.2.1 EM Algorithm

The goal of the EM algorithm is to approximate a solution of the problem (1.13). 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:

$$\log\left\{\sum_{k=1}^{K}\pi_k\varphi_{\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k}(\boldsymbol{x}_i)\right\} = \max_{\boldsymbol{\tau}\in[0,1]^K\ \boldsymbol{\tau}^\top\mathbf{1}_K=1}\sum_{k=1}^{K}\Big\{\tau_k\log\varphi_{\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k}(\boldsymbol{x}_i)+\tau_k\log(\pi_k/\tau_k)\Big\}. \tag{1.15}$$

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 (1.13) and (1.15), we get

$$\widehat{\boldsymbol{\theta}}\in\arg\max_{\boldsymbol{\theta}=(\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})}\max_{\boldsymbol{\mathcal{T}}}\sum_{i=1}^{N}\sum_{k=1}^{K}\Big\{\tau_{i,k}\log\varphi_{\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k}(\boldsymbol{x}_i)+\tau_{i,k}\log(\pi_k/\tau_{i,k})\Big\}. \tag{1.16}$$

The great advantage of this new representation of the log-likelihood function is that the cost function in (1.16), 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 (1.16) is that the two optimization problems involved in the alternating maximization procedure admit explicit solutions.

**Lemma 1.** *Let us introduce the cost function*

$$F(\boldsymbol{\theta},\boldsymbol{\mathcal{T}})=\sum_{i=1}^{n}\sum_{k=1}^{K}\Big\{\tau_{i,k}\log\varphi_{\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k}(\boldsymbol{x}_i)+\tau_{i,k}\log(\pi_k/\tau_{i,k})\Big\}. \tag{1.17}$$

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

Figure 1.7: EM algorithm for Gaussian mixtures

*Then, the following two optimization problems*

$$\widehat{\boldsymbol{\theta}}(\boldsymbol{\mathcal{T}}) \in \arg\max_{\boldsymbol{\theta}} F(\boldsymbol{\theta}, \boldsymbol{\mathcal{T}}), \qquad \widehat{\boldsymbol{\mathcal{T}}}(\boldsymbol{\theta}) \in \arg\max_{\boldsymbol{\mathcal{T}}} F(\boldsymbol{\theta}, \boldsymbol{\mathcal{T}}) \tag{1.18}$$

*has explicit solutions given by*

$$\widehat{\pi}_k = \frac{1}{n} \sum_{i=1}^n \tau_{i,k}, \qquad \widehat{\boldsymbol{\mu}}_k = \frac{1}{n\widehat{\pi}_k} \sum_{i=1}^n \tau_{i,k} \boldsymbol{x}_i, \qquad \forall k \in [K], \tag{1.19}$$

$$\widehat{\boldsymbol{\Sigma}}_k = \frac{1}{n\widehat{\pi}_k} \sum_{i=1}^n \tau_{i,k} (\boldsymbol{x}_i - \widehat{\boldsymbol{\mu}}_k)(\boldsymbol{x}_i - \widehat{\boldsymbol{\mu}}_k)^\top, \qquad \forall k \in [K], \tag{1.20}$$

$$\widehat{\tau}_{i,k} = \frac{\pi_k \varphi_{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k}(\boldsymbol{x}_i)}{\sum_{k' \in [K]} \pi_{k'} \varphi_{\boldsymbol{\mu}_{k'}, \boldsymbol{\Sigma}_{k'}}(\boldsymbol{x}_i)}, \qquad \forall k \in [K], \ \forall i \in [n]. \tag{1.21}$$

Based on this result, the EM algorithm is defined as in Figure 1.7. 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

$$\min_{\boldsymbol{c}_1,\ldots,\boldsymbol{c}_K} \min_{\boldsymbol{R}\in\{0,1\}^{N\times K}} \sum_{i=1}^{N}\sum_{k=1}^{K} r_{ik}\|\boldsymbol{x}_i - \boldsymbol{c}_k\|^2, \tag{1.22}$$

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

$$2\sum_{i=1}^{N} r_{ik}(\boldsymbol{x}_i - \boldsymbol{c}_k) = 0, \tag{1.23}$$

which leads to

$$\boldsymbol{c}_k = \frac{\sum_{i=1}^{N} r_{ik}\boldsymbol{x}_i}{\sum_{i=1}^{N} r_{ik}}. \tag{1.24}$$

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 referes 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. This is an important issue in the clustering context. In the Gaussian mixture model of $K$ components in dimension $p$, the number of parameters to estimate is:

Note: reecrire cette partie en commencant par le pb de la HD en stats en general, se baser sur les points de Giraud, puis une deuxieme sous section avec le cas du clustering et les pistes de resolution connues, ex: penalization, subspace clustering...

$$\nu = \underbrace{(K-1)}_{\text{Weights}} + \underbrace{Kp}_{\text{Means}} + \underbrace{Kp(p-1)^2}_{\text{Covariances Matrices}} \tag{1.25}$$

Moreover, the evaluation of $\widehat{\tau}_{i,k}$ in eq. (1.21) needs to evaluate the inverse of the covariance matrix $\widehat{\boldsymbol{\Sigma}}_k$ which is called the precision matrix. If $n << \nu$ the matrices $\widehat{\boldsymbol{\Sigma}}_k$ $k = 1, \ldots, K$ are ill conditioned and the precision matrices are prone to large numerical errors or more often singular and the problem can not be solved. In section 1.4.1, we tackle this challenge by studying some nice structural properties of precision matrices. However, an interesting phenomenon occurs in high dimension, Scoot and Thomson showed that high-dimensional spaces are mostly empty, Huber showed that the realizations of a $p$-dimensional random vector with a uniform probability distribution on the unit hypershpere lies with high probability close to the boundary of this hypershere. Therefore, the data belong mostly in a $p-1$ dimensional

Note: Ajouter un tableau des differents types de modeles et le nombre de parametres

Note: citer

Note: cite

Note: cite

Note: attention de pas copier BouveyronBouveyron and Brunet [2013] subspace. Therefore, in the clustering problem, different clusters may live on different subspaces

Note: subspace clustering

### 1.3.1 Bibliographic notes

The reader can find a more thorough study of high dimensional statistics in Giraud [2014]. The reader can refer to [Bouveyron and Brunet, 2013] to have an overview of the different Gaussian mixture models for clustering in high dimension.

## 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

Note: changer en section

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.

### Introduction

We consider $\boldsymbol{X} = (\boldsymbol{X}^{(1)}, \dots, \boldsymbol{X}^{(p)})$ 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, $E = (e_{i,j})_{1 \leq i < j \leq p}$, the edges between the vertices describing the conditional independence relationship among $\boldsymbol{X}^{(1)}, \dots, \boldsymbol{X}^{(p)}$. 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 $\boldsymbol{X} = (\boldsymbol{X}^{(1)}, \dots, \boldsymbol{X}^{(p)})$ 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 $\boldsymbol{X}$

$$\varphi_{\boldsymbol{\mu},\boldsymbol{\Sigma}}(\boldsymbol{x}) = \frac{1}{(2\pi)^{p/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\Big( -\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x} - \boldsymbol{\mu})\Big), \tag{1.26}$$

it can be rewritten as

$$\varphi_{\boldsymbol{\mu},\boldsymbol{\Sigma}}(\boldsymbol{x}) = \exp(\alpha + \beta^T \boldsymbol{x} - \frac{1}{2}\boldsymbol{x}^T \boldsymbol{\Omega} \boldsymbol{x}), \tag{1.27}$$

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

$$\exp\big(\alpha + \sum_{j=1}^{p} \beta_j \boldsymbol{x}^{(j)} - \frac{1}{2}\sum_{j=1}^{p}\sum_{(i=1)}^{p} \omega_{i,j} \boldsymbol{x}^{(j)} \boldsymbol{x}^{(i)}\big). \tag{1.28}$$

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. (1.28), we have $X^{(i)} \perp\!\!\!\perp X^{(j)}|X^{(l)} \iff \omega_{i,j} = 0$. $\square$

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 $X^{(i)} \perp\!\!\!\perp \left\{X^{(j)} : X^{(j)} \in G \setminus \{ne_{X^{(i)}}\}\right\}|X_{ne_{X^{(i)}}}$. 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

$$\theta^{(i)} = \underset{\theta:\theta_i=0}{\arg\min}\, \mathbb{E}\Big[X^{(i)} - \sum_{k=1}^{p} \theta_k X^{(k)}\Big], \tag{1.29}$$

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

$$ne_{X^{(i)}} = \{X^{(j)}, j \in [p] : \omega_{i,j} \neq 0\}. \tag{1.30}$$

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

$$\widehat{\theta}^{i,\lambda} = \underset{\theta:\theta_i=0}{\arg\min} \Big(\frac{1}{n}\|\mathbb{X}^{(i)} - \mathbb{X}\theta\|_2^2 + \lambda\|\theta\|_1\Big). \tag{1.31}$$

The authors proved under several assumptions that

$$P(\widehat{ne}^{\lambda}_{X^{(i)}} = ne_{X^{(i)}}) \to 1 \quad \text{for}\, n \to \infty, \tag{1.32}$$

and for some $\epsilon > 0$,

$$P(\widehat{E}^{\lambda} = E) = 1 - \mathcal{O}(\exp(-cn^{\epsilon})) \quad \text{for}\, n \to \infty. \tag{1.33}$$

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, 2006] as they use the $\ell_1$ penalty but with the likelihood and the addition of a positive definite constraint. The log-likelihood for $\mathbf{\Omega}$ based on a centered random sample $\boldsymbol{X}_1, \dots, \boldsymbol{X}_n$ of $\boldsymbol{X}$ is

Note: ajouter un mot sur la complexité

$$\frac{n}{2}\log(|\mathbf{\Omega}|) - \frac{1}{2}\sum_{i=1}^{n} \boldsymbol{X}_i^T \mathbf{\Omega} \boldsymbol{X}_i \tag{1.34}$$

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

$$\min\Big\{ -\log(|\mathbf{\Omega}|) + \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{X}_i^T \mathbf{\Omega} \boldsymbol{X}_i \Big\} \quad \text{subject to} \quad \sum_{i \neq j} |\omega_{i,j}| \leq t, \tag{1.35}$$

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. (1.35) can be rewritten as

$$\min\Big\{ -\log(|\mathbf{\Omega}|) + \mathrm{tr}(\boldsymbol{S}\mathbf{\Omega}) \Big\} \quad \text{subject to} \quad \sum_{i \neq j} |\omega_{i,j}| \leq t. \tag{1.36}$$

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

$$\mathcal{L}(\lambda, \boldsymbol{\Omega}) = -\log(|\boldsymbol{\Omega}|) + \mathrm{tr}(\boldsymbol{S}\boldsymbol{\Omega}) + \lambda \sum_{i \neq j} |\omega_{i,j}|, \tag{1.37}$$

with $\lambda$ 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*

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

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

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

Note: mettre un commentaire sur ce resultat et aspect algorithmique

Unfortunately, the computational complexity of interior point methods for maximizing eq. (1.37) 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 $\ell_1$ penalty and focusing on the computation complexity by proposing an iterative block coordinate descent algorithm. The problem to maximize is similar to eq. (1.37)

$$\widehat{\boldsymbol{\Omega}} = \arg\max_{\boldsymbol{\Omega} \succ 0} \{\log(|\boldsymbol{\Omega}|) - \mathrm{tr}(\boldsymbol{S}\boldsymbol{\Omega}) - \lambda \|\boldsymbol{\Omega}\|_1\}. \tag{1.38}$$

Note that the $\ell_1$ norm of a matrix $\boldsymbol{\Omega}$ can be expressed as

$$\|\boldsymbol{\Omega}\|_1 = \max_{\|\boldsymbol{U}\|_\infty \leq 1} \mathrm{tr}(\boldsymbol{\Omega U}), \tag{1.39}$$

injecting this in eq. (1.38) gives

$$\max_{\boldsymbol{\Omega}\succ 0}\min_{\|\boldsymbol{U}\|_\infty\leq\lambda}\left\{\log(|\boldsymbol{\Omega}|)-\operatorname{tr}(\boldsymbol{\Omega}(\boldsymbol{S}+\boldsymbol{U}))\right\}. \tag{1.40}$$

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

$$\min_{\|\boldsymbol{U}\|_\infty}\{-\log(|\boldsymbol{S}+\boldsymbol{U}|)-p\}, \tag{1.41}$$

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

$$\widehat{\boldsymbol{\Sigma}}=\widehat{\boldsymbol{\Omega}^{-1}}=\arg\max\log(|\boldsymbol{W}|)\quad \text{s.t}\quad \|\boldsymbol{W}-\boldsymbol{S}\|_\infty\leq\lambda. \tag{1.42}$$

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. 1.8. 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 $\varepsilon$-suboptimal solution to eq. (1.42) 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 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. 1.8 is

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

Note: citer les theoremes et choix du param

$$\min_{\boldsymbol{x}}\boldsymbol{x}^T\boldsymbol{W}^{(j-1)}_{\backslash j\backslash j}\boldsymbol{x}-\boldsymbol{S}_j^T\boldsymbol{x}+\lambda\|\boldsymbol{x}\|_1, \tag{1.43}$$

and strong duality holds, it can best casted as

$$\min_{\boldsymbol{x}}\|\boldsymbol{Q}\boldsymbol{x}-\boldsymbol{b}\|_2^2+\lambda\|\boldsymbol{x}\|_1, \tag{1.44}$$

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

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

Figure 1.8: Block Coordinate Descent Algorithm

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

$$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{W}_{11} & \boldsymbol{w}_{12} \\ \boldsymbol{w}_{21} & w_{22} \end{bmatrix}, \quad \boldsymbol{S} = \begin{bmatrix} \boldsymbol{S}_{11} & \boldsymbol{s}_{12} \\ \boldsymbol{s}_{21} & s_{22} \end{bmatrix}, \tag{1.45}$$

and the Graphical Lasso algorithm is described in fig. 1.9. 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. (1.44) for each column of $\boldsymbol{W}$. It is easy to recover $\boldsymbol{\Omega}$ since

$$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{W}_{11} & \boldsymbol{w}_{12} \\ \boldsymbol{w}_{21} & w_{22} \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{\Omega}_{11} & \boldsymbol{\omega}_{12} \\ \boldsymbol{\omega}_{21} & \omega_{22} \end{bmatrix} = \begin{bmatrix} I_{p-1} & 0 \\ 0 & 1 \end{bmatrix}, \tag{1.46}$$

```
1:  Input: Matrix $\boldsymbol{S}$, parameter $\lambda$ and threshold $\varepsilon$
2:  Output: Estimate of $\boldsymbol{W}$ and $\boldsymbol{B}$ a matrix of parameters.
3:  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.
4:  repeat
5:    for $j = 1, \ldots, p$ do
6:      (a) Let $\boldsymbol{W}^{(j-1)}$ denote the current iterate. Solve the Lasso problem in eq. (1.44)
          $$\widehat{\boldsymbol{x}}^{(j-1)} = \arg\min_{\boldsymbol{x}} \frac{1}{2}\|(\boldsymbol{W}_{11}^{(j-1)})^{1/2}\boldsymbol{x} - \boldsymbol{b}\|_2^2 + \lambda\|\boldsymbol{x}\|_1, \quad (1.47)$$
    with $\boldsymbol{b} := (\boldsymbol{W}_{11}^{(j-1)})^{-1/2}\boldsymbol{s}_{12}$.
7:      (b) Update: $\boldsymbol{W}^{(j)}$ is $\boldsymbol{W}^{(j-1)}$ with $\boldsymbol{w}_{12} = \boldsymbol{W}_{11}^{(j-1)}\widehat{\boldsymbol{x}}^{(j-1)}$.
8:      (c) Save the parameter $\boldsymbol{x}^{(j-1)}$ in the $j^{th}$ column of $\boldsymbol{B}$.
9:      (d) Permute the columns and rows of $\boldsymbol{W}^{(j-1)}$ such that the $j^{th}$ column is $\boldsymbol{w}_{12}$,
    the next target.
10:   end for
11:   Let $\widehat{\boldsymbol{W}}^{(0)} := \boldsymbol{W}^{(p)}$.
12: until convergence occurs.
```

Figure 1.9: Graphical Lasso

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 $j = 1, \ldots, p$, 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.

### 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

$$\ell_n^{pen}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - pen(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log \left\{ \sum_{k=1}^{K} \pi_k \varphi_{(\boldsymbol{\mu}_k, \boldsymbol{\Omega}_k)}(\boldsymbol{x}_i) \right\} - pen(\boldsymbol{\theta}). \tag{1.48}$$

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 $\ell_1$ 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

$$F^{pen}(\boldsymbol{\theta}, \boldsymbol{\mathcal{T}}) = \sum_{k=1}^{K} \left( \sum_{i=1}^{n} \left\{ \tau_{i,k} \log \varphi_{\boldsymbol{\mu}_k, \boldsymbol{\Omega}_k}(\boldsymbol{x}_i) + \tau_{i,k} \log(\pi_k / \tau_{i,k}) \right\} - \lambda_k ||\boldsymbol{\Omega}_k||_{1,1} \right). \tag{1.49}$$

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

Note: ajouter les domaines

$$\widehat{\boldsymbol{\theta}}(\boldsymbol{\mathcal{T}}) \in \arg\max_{\boldsymbol{\theta}} F^{pen}(\boldsymbol{\theta}, \boldsymbol{\mathcal{T}}), \qquad \widehat{\boldsymbol{\mathcal{T}}}(\boldsymbol{\theta}) \in \arg\max_{\boldsymbol{\mathcal{T}}} F^{pen}(\boldsymbol{\theta}, \boldsymbol{\mathcal{T}}). \tag{1.50}$$

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. (1.50) are the same as in the EM algorithm

$$\widehat{\pi}_k = \frac{1}{n} \sum_{i=1}^{n} \widehat{\tau}_{i,k}, \quad \text{and} \quad \widehat{\boldsymbol{\mu}}_k = \frac{1}{n\widehat{\pi}_k} \sum_{i=1}^{n} \widehat{\tau}_{i,k} \boldsymbol{x}_i, \qquad \forall k \in [K] \tag{1.51}$$

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

$$\widehat{\tau}_{i,k} = \frac{\widehat{\pi}_k \varphi_{\widehat{\boldsymbol{\mu}}_k, \widehat{\boldsymbol{\Omega}}_k}(\boldsymbol{x}_i)}{\sum_{k' \in [K]} \widehat{\pi}_{k'} \varphi_{\widehat{\boldsymbol{\mu}}_{k'}, \widehat{\boldsymbol{\Omega}}_{k'}}(\boldsymbol{x}_i)} = p_{\boldsymbol{\theta}}(Z = k | \boldsymbol{X} = \boldsymbol{x}_i), \qquad \forall k \in [K], \ \forall i \in [n]. \tag{1.52}$$

However, due to the penality $\lambda_k||\mathbf{\Omega}_k||_{1,1}$, the estimation of $\mathbf{\Omega}_k$ is not straightforward.

We introduce the weighted empirical covariance matrix

$$\mathbf{\Sigma}_{n,k} = \frac{1}{n}\frac{\sum_{i=1}^{n} \tau_{i,k}(\boldsymbol{x}_i - \widehat{\boldsymbol{\mu}}_k)(\boldsymbol{x}_i - \widehat{\boldsymbol{\mu}}_k)^\top}{\sum_{i=1}^{n} \tau_{i,k}} \tag{1.53}$$

The Gaussian density in equation (1.49) can be expanded as follows

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

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

$$\mathbf{\Omega}_k \in \underset{\mathbf{\Omega}\succeq 0}{\arg\min}\Big\{-\frac{n\pi_k}{2}\log|\mathbf{\Omega}| + \frac{1}{2}\sum_{i=1}^{n}\tau_{i,k}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T\mathbf{\Omega}(\boldsymbol{x}_i - \boldsymbol{\mu}_k) + \lambda_k||\mathbf{\Omega}||_{1,1}\Big\} \tag{1.54}$$

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

$$\mathbf{\Omega}_k \in \underset{\mathbf{\Omega}\succeq 0}{\arg\min}\Big\{-\frac{1}{2}\log|\mathbf{\Omega}| + \frac{1}{2}tr(\mathbf{\Sigma}_{n,k}\mathbf{\Omega}) + \frac{\lambda_k}{n\pi_k}||\mathbf{\Omega}||_{1,1}\Big\} \tag{1.55}$$

Our algorithm solves a graphical lasso problem within each cluster. We use a block coordinate ascent algorithm [Mazumder, 2012] to solve this convex problem as in the graphical lasso implementation in R, see `http://statweb.stanford.edu/~tibs/glasso/` The alternating maximization procedure is summarized in the following algorithm

### 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,

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

Figure 1.10: Graphical lasso algorithm for Gaussian mixtures

Expectation-Maximisation 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.

### Introduction and related work

In previous models, we knew the number of components $K$ in the Gaussian mixture. In reality this parameter is unknown. Several methods exists to select the number of clusters

### Bayesian Information Criterion (BIC)

A common method to select the number of clusters is to use the Bayesian Information Criterion given by:

$$BIC(K) = -\log \ell_n(\widehat{\boldsymbol{\theta}}^K) + K.\log(n) \tag{1.56}$$

And select the model which minimizes the BIC. This can be done by running EM algorithm over a large number of models which is computationally expensive.

### 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 1.49, 1.50 with a regularization term:

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

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

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

and the optimization problem:

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

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:

$$F^{pen}_{\boldsymbol{\theta}}(\boldsymbol{\mathcal{T}}) = f(\boldsymbol{\mathcal{T}}) + g(\boldsymbol{\mathcal{T}}) \tag{1.58}$$

with:

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

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

```
Input:
Output: parameter estimate $\boldsymbol{\mathcal{T}}$
1:  Initialize $t_1 = 1$ and $\boldsymbol{\xi}^0$ with
        $\xi^0_{i,k} = \frac{\pi^0_k \varphi_{\boldsymbol{\mu}^0_k, \boldsymbol{\Omega}^0_k}(\boldsymbol{x}_i)}{\sum_{k' \in [K]} \pi^0_{k'} \varphi_{\boldsymbol{\mu}^0_{k'}, \boldsymbol{\Omega}^0_{k'}}(\boldsymbol{x}_i)}$
2:  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 1.11: $\boldsymbol{\mathcal{T}}$ estimation with FISTA

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

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 $\boldsymbol{\mathcal{T}}$

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

Figure 1.12: Graphical lasso algorithm for Gaussian mixtures with cluster number discovery

### 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:

$$\ell_n(\boldsymbol{\theta}) = -\frac{1}{n}\sum_{i=1}^{n}\log\left\{\sum_{j=1}^{K}\pi_k\varphi_{(\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)}(\boldsymbol{x}_i)\right\} + \lambda\sum_{j=1}^{K-1}\pi_j^{1/\gamma} \quad \gamma \geq 1 \tag{1.59}$$

Such that:

$$\sum_{j}^{K-1}\pi_j \leq 1 \quad \text{and} \quad \pi_K = 1 - \sum_{j}^{K-1}\pi_j \tag{1.60}$$

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

$$\widehat{\boldsymbol{\alpha}} \in \operatorname*{arg\,min}_{\boldsymbol{\alpha}\in\mathbb{R}^{K-1}}\left\{-\frac{1}{n}\sum_{i=1}^{n}\log\left\{\sum_{j=1}^{K}\alpha_j^{\gamma}\varphi_{(\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)}(\boldsymbol{x}_i)\right\} + \lambda\sum_{j=1}^{K-1}\alpha_j\right\} \quad \gamma \geq 1, \tag{1.61}$$

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 $\chi_A$ the indicator function of $A$ (0 in $A$, $\infty$ elsewhere), the minimization problem can be rewritten as

$$\widehat{\boldsymbol{\alpha}} \in \operatorname*{arg\,min}_{\boldsymbol{\alpha}\in\mathbb{R}^{K-1}}\{f_{\boldsymbol{\theta}}(\boldsymbol{\alpha}) + \chi_A(\boldsymbol{\alpha})\}. \tag{1.62}$$

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

$$\widehat{\alpha}^{t+1} = \operatorname{prox}_{\chi_A}(\boldsymbol{\alpha}^t - h\nabla f_{\boldsymbol{\theta}}(\boldsymbol{\alpha}^t)) \tag{1.63}$$

$$= \operatorname*{arg\,min}_{x\in\mathbb{R}^{K-1}}\left\{\chi_A(x) + \frac{1}{2}||x - (\boldsymbol{\alpha}^t - h\nabla f_{\boldsymbol{\theta}}(\boldsymbol{\alpha}^t))||^2\right\} \tag{1.64}$$

$$= P_A(\boldsymbol{\alpha}^t - h\nabla f_{\boldsymbol{\theta}}(\boldsymbol{\alpha}^t)). \tag{1.65}$$

This iteration procedure gives us the following algorithm

```
**Input:** $\boldsymbol{\theta}$
**Output:** parameter estimate $\widehat{\boldsymbol{\pi}} = (\boldsymbol{\alpha}_1^{\gamma}, ..., \boldsymbol{\alpha}_{K-1}^{\gamma}, 1 - \sum_{j=1}^{K-1} \boldsymbol{\alpha}_j^{\gamma})^t$
1:  Initialize $t = 0$, $s_0 = 1$ and $\boldsymbol{\xi}^0 = (\boldsymbol{\pi}_1^{1/\gamma}, ..., \boldsymbol{\pi}_{K-1}^{1/\gamma})$
2:  **Repeat**
3:
        $\boldsymbol{\alpha}^t = P_A(\xi^t - h\nabla f_{\boldsymbol{\theta}}(\xi^t))$   (1.66)
        $s_{t+1} = \frac{1 + \sqrt{1 + 4 * s_t^2}}{2}$   (1.67)
        $\boldsymbol{\xi}^{t+1} = \boldsymbol{\alpha}^t + \left(\frac{s_t - 1}{s_{t+1}}\right)(\boldsymbol{\alpha}^t - \boldsymbol{\alpha}^{t-1})$   (1.68)
5:      increment $t$:  $t = t + 1$.
6:  **Until** stopping rule.
```

Figure 1.13: Estimation of $\alpha$

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 $t = 0$, $\boldsymbol{\theta} = \boldsymbol{\theta}^0$

```
1:  Initialize $t = 0$, $\boldsymbol{\theta} = \boldsymbol{\theta}^0$
2:  **Repeat**
3:  Update the parameter $\boldsymbol{\mathcal{T}}$
```

$$\tau_{i,k}^t = \frac{\pi_k^t \varphi_{\boldsymbol{\mu}_k^t, \boldsymbol{\Omega}_k^t}(\boldsymbol{x}_i)}{\sum_{k'\in[K]} \pi_{k'}^t \varphi_{\boldsymbol{\mu}_{k'}^t, \boldsymbol{\Omega}_{k'}^t}(\boldsymbol{x}_i)}. \tag{1.69}$$

```
4:  Update parameters $\boldsymbol{\mu}, \boldsymbol{\Sigma}$.
```

$$\boldsymbol{\mu}_k^{t+1} = \frac{1}{n\pi_k^{t+1}} \sum_{i=1}^n \tau_{i,k}^t \boldsymbol{x}_i, \tag{1.70}$$

$$\boldsymbol{\Sigma}_k^{t+1} = \frac{1}{n\pi_k^{t+1}} \sum_{i=1}^n \tau_{i,k}^t (\boldsymbol{x}_i - \boldsymbol{\mu}_k^{t+1})(\boldsymbol{x}_i - \boldsymbol{\mu}_k^{t+1})^\top. \tag{1.71}$$

```
5:  Update the parameter $\pi$ with previous algorithm
6:  increment $t$:  $t = t + 1$
7:  **Until** stopping rule.
```

Figure 1.14: 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 éffectuées. Nous représentons les premiers et troisièmes quartiles ainsi que la médiane.

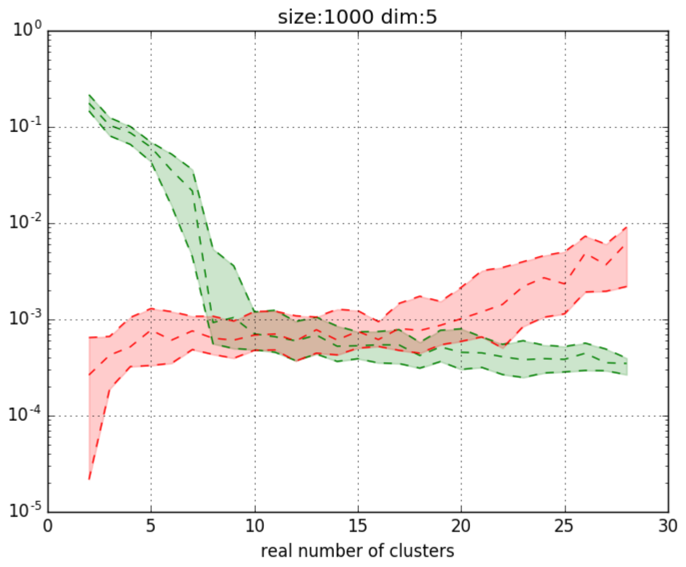


Figure 1.15: Vert: Notre algorithme. Rouge: EM+BIC

## 1.5 structural analysis on $\Sigma$ approach

We consider a multivariate Gaussian distribution with mean $\boldsymbol{\mu}^*$ and covariance $\boldsymbol{\Sigma}^*$ and $Y_1, \ldots, Y_N \in \mathbb{R}^p$ iid drawn from this distribution. We would like to estimate $\boldsymbol{\mu}^*$ and $\boldsymbol{\Sigma}^*$. We know that $\widehat{\boldsymbol{\mu}}_n = \bar{Y}_n$, then wlog we consider $\mu^* = 0$, the problem is to estimate $\boldsymbol{\Sigma}^*$. We will study the precision matrix and consider that $\Sigma^{-1}$ is sparse. We note $\Sigma^{-1} = \Omega$, $Y_n$ the $n$-th random variable and $Y_n^i$ the $i$-th component of this vector. If $\Sigma_{ij}^{-1} = 0 \Rightarrow Y^i \perp\!\!\!\perp Y^j$ conditionally to $Y^{l \neq \{i,j\}}$. Thus, it makes sense to impose a $L_1$ penalty on $\Sigma^{-1}$ to increase its sparsity.

### 1.5.1 Graphical Lasso

Let consider a multivariate normal distribution with parameters $\mu^*$, $\Sigma^*$ with density;

$$\mathcal{N}(x|\mu^*, \Sigma^*) = \frac{1}{(2\pi)^{d/2}|\Sigma^*|^{1/2}} \exp^{-\frac{1}{2}(x-\mu^*)^T \Sigma^{-1*}(x-\mu^*)} \tag{1.72}$$

We consider $\mu = 0$. Given N datapoints $X_1, \ldots, X_N$ and $X_i \in \mathbb{R}^d$, the log-likelihood is given by:

$$\begin{aligned} \mathcal{L}(\Sigma) &= \log\left(\prod_{n=1}^{N} \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp^{-\frac{1}{2}(x_n)^T\Sigma^{-1}(x_n)}\right) \\ &= -\frac{dN}{2}\log 2\pi - \frac{N}{2}\sum_{n=1}^{N}\log|\Sigma^*| - \frac{1}{2}\sum_{n=1}^{N} x_n^T\Sigma^{*,-1}x_n \end{aligned} \tag{1.73}$$

Note that $x_n^T\Sigma^{*,-1}x_n = tr(x_n^T\Sigma^{*,-1}x_n)$, and therefore:

$$\sum_{n=1}^{N} x_n^T\Sigma^{*,-1}x_n = tr\big(\sum_{n=1}^{N} x_n^T\Sigma^{*,-1}x_n\big) = tr\Big(\big[\sum_{n=1}^{N} x_n^T x_n\big]\Sigma^{*,-1}\Big) = tr(S_N\Sigma^*) \tag{1.74}$$

Where $S_N$ is the empirical covariance matrix. We can replace that in the log-likelihood expression:

$$\mathcal{L}(\Sigma) = -\frac{dN}{2}\log 2\pi - \frac{N}{2}\sum_{n=1}^{N}\log|\Sigma^*| - \frac{1}{2}tr(S_N\Sigma^*) \tag{1.75}$$

Finally:

$$\mathcal{L}(\Sigma) = C + \frac{N}{2}\log|\Sigma^{-1}| - \frac{1}{2}tr(S_n\Sigma^{-1}) \tag{1.76}$$

Where C is a constant (dependent on N). Thus, considering the sparsity of the precision matrix $\Omega = \Sigma^{-1}$, we impose a penalization to the maximum likelihood estimator of $\Omega$

$$\widehat{\Omega} \in argmin\big\{\log|\Omega| - tr(S_N\Omega) - \lambda||\Omega||_1\big\} \tag{1.77}$$

A reason to use the $L_1$ penalization instead of the ridge is that for an $L_p$ penalization, the problem is convex for $p \geq 1$ and we have parsimonious property for $p \leq 1$. This is a convex optimization problem, however the complexity is $O(p^3)$ (Source, high dim & var select Buhlmann 2006 ? Wassermann)

### 1.5.2 Column-Wise Lasso

We consider a gaussian vector $Y \in \mathbb{R}^d$, $Y \sim \mathcal{N}(0, \Sigma)$. We can write $Y = (Y^1, Y^{2:d})$. With this decomposition we can write the covariance matrix as following:

$$\Sigma = \begin{pmatrix} \sigma_1^2 & \Sigma_{12}\ \Sigma_{12}^T & \Sigma_{22} \end{pmatrix} \tag{1.78}$$

and according to theorem[?]: If $\Sigma_{22}$ is inversible, then:

$$\begin{aligned} \mathbf{E}[Y^1|Y^{2:d}] &= \Sigma_{12}\Sigma_{22}^{-1}Y^{2:d} \\ Var[Y^1|Y^{2:d}] &= \sigma_1^2 - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{12}^T \end{aligned} \tag{1.79}$$

We have the following identity:

$$\begin{pmatrix}\omega_{11} & \Omega_{12}\,\Omega_{12}^T & \Omega_{22}\end{pmatrix}\begin{pmatrix}\sigma_1^2 & \Sigma_{12}\,\Sigma_{12}^T & \Sigma_{22}\end{pmatrix} = \begin{pmatrix}1 & 0\;0 & I_{p-1}\end{pmatrix} \tag{1.80}$$

Which gives the following equations:

$$\begin{cases} \omega_{11}\sigma_1^2 + \Omega_{12}\Sigma_{12}^T & = 1 \qquad (*) \\ \omega_{11}\Sigma_{12} + \Omega_{12}\Sigma_{22} & = 0 \qquad (**) \\ \Omega_{12}^T\Sigma_{12} + \Omega_{22}\Sigma_{22} & = I_{p-1} \quad (***) \end{cases} \tag{1.81}$$

With (**) we have $-\omega_{11}\Sigma_{12}\Sigma_{22}^{-1} = \Omega_{12}$ and injected to (*) we have:

$$\begin{cases} \mathbf{E}[Y^1|Y^{2:d}] & = -\frac{1}{\omega_{11}}\Omega_{12}Y^{2:d} \\ Var[Y^1|Y^{2:d}] & = \frac{1}{\omega_{11}} \end{cases} \tag{1.82}$$

Finally, $Y^1 - \mathbf{E}[Y^1|Y^{2:d}]$ is a gaussian vector of $\mathbb{R}^{d-1}$, centered, independent of $Y^{2:d}$ and of covariance matrix $\sigma_1^2 - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{12}^T$. If we denote $\xi^1 \sim \mathcal{N}(0,1)$ we have $Y^1 - \mathbf{E}[Y^1|Y^{2:d}] = \frac{1}{\sqrt{\omega_{11}}}\xi^1$.

Therefore, for $Y_1, \ldots, Y_n$ iid of law $\mathcal{N}(0, \Sigma^*)$ we have:

$$\begin{aligned} Y_i^1 &= -\frac{1}{\omega_{11}^*}\Omega_{12}Y_i^{2:d} + \frac{1}{\sqrt{\omega_{11}^*}}\xi_i^1 \\ &= -\sum_{j=2}^d \frac{w_{ij}^*}{\omega_{11}^*}Y_i^j + \frac{1}{\sqrt{\omega_{11}^*}}\xi_i^1 \end{aligned} \tag{1.83}$$

and

$$\beta_1^{*T}Y_i = \frac{1}{\sqrt{\omega_{11}^*}}\xi_i^1 \Rightarrow \beta_1^{*T}\boldsymbol{Y} = \frac{1}{\sqrt{\omega_{11}^*}}\boldsymbol{\xi}^1 \tag{1.84}$$

with

$$\beta_1^* = \frac{1}{\sqrt{\omega_{11}^*}}\begin{bmatrix}w_{11}^*\; w_{12} \vdots w_{1d}\end{bmatrix} \in \mathbb{R}^d \quad \text{and} \quad \boldsymbol{Y} = \begin{bmatrix}verifier\end{bmatrix} \tag{1.85}$$

## 1.6 Overview

# Chapter 2

# Appendix and notes

## 2.1 Notes on "Model-Based Clustering of High-Dimensional Data: A review"

From Bouveyron and Brunet [2013]

> FA-based models choose the latent subspace(s) maximizing the projected variance whereas the Discriminative latent mixture (DLM) model chooses the latent subspace which maximizes the separation between the groups.

The DLM model assumes that $Y$ is linked to a latent variable $X \in \mathbb{E}$ where $\mathbb{E} \subset \mathbb{R}^p$ through a linear relationship

$$Y = UX + \varepsilon \tag{2.1}$$

$\mathbb{E}$ is the most dicriminative subspace of dimension $d \leq K - 1$

# 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.

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.

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.

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.

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.

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.

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.