On K-means and PCA

Probabilistic Graphical Models, Master MVA

Chloé Habasque, Colin Coërchon, Maëlle Fontaine

December 2024

1 Introduction

1.1 Presentation of the articles

In the scope of this project, we reviewed three articles: *K-means Clustering via Principal Component Analysis* [1], *PCA-guided search for K-means* [3] and *Dimensionality Reduction using PCA and K-Means Clustering for Breast Cancer Prediction* [2].

These articles examine the interplay between Principal Component Analysis (PCA) and K-means clustering. PCA is an unsupervised technique for dimensionality reduction, whereas K-means is an unsupervised clustering algorithm. The first article [1] establishes a strong link between the two methods, demonstrating that the principal components correspond to the continuous solutions of the membership indicators in K-means clustering. Building on these findings, *PCA-guided search for K-means* [3] proposes a method to enhance the efficiency of K-means clustering. Specifically, it introduces the 'PCA-guided search' approach, which uses PCA to improve the initialization of the K-means algorithm, thereby making the clustering process faster and more accurate. This emphasis on initialization is crucial, as the convergence of K-means to local solutions depends heavily on its initialization. The third article, *Dimensionality Reduction using PCA and K-Means Clustering for Breast Cancer Prediction* [2], further reinforces the connection between PCA and K-means. It focuses on reducing the dimensionality of features for machine learning models by comparing the two methods and showing that K-means clustering can act as a dimensionality reduction technique by generating clusters as new features. Moreover, [3] illustrates the utility of PCA in unsupervised learning contexts, while [2] highlights how K-means clustering can be leveraged for dimensionality reduction. Together, these findings underscore the similarity between the two techniques and validate the insights presented in [1].

1.2 Approach

This project builds on the findings of [1] and the applications explored in [3]. We reproduce the results from [3] and extend them by proposing a Gaussian Mixture Model (GMM) approach. In this method, clusters in the PCA-reduced space are assigned using the Expectation-Maximization (EM) algorithm, providing a probabilistic perspective on clustering. By comparing the performance of GMM with K-means, we aim to highlight the benefits of combining PCA with probabilistic clustering.

1.3 Contributions

Although all team members contributed to various parts of the project, here are the main contributions of each individual: Chloé Habasque worked on the PCA guided search method and the different initialisations of K-Means, Colin Coërchon worked on the theoretical connection between PCA and K-means, as well as the theory and implementation of the EM algorithm for GMM. Maëlle Fontaine focused on the third section, studying the contribution of GMM to PCA-guided search and the question of the number of clusters.

2 Method

The algorithms PCA and K-means, which form the basis of the methods developed in this project, are detailed in the appendix (Appendix A, Appendix B).

2.1 PCA as a continuous solution to K-means clustering

The standard iterative solution of K-means faces a key limitation: **the solutions tend to converge to local minima** due to the greedy nature of its updates. To mitigate this issue, initialization plays a critical role. In [1], a theoretical link between K-means and PCA is established, showing that **principal components provide a continuous solution of cluster membership indicators in K-means**.

This insight can enhance K-means results, as PCA provides a relaxed solution to the K-means clustering problem, enabling spectral methods inspired by PCA to approximate clusters before applying K-means.

Proof

The proof is detailed in the Appendix E. Here are the main lines.

The foundation of the proof relies on rewriting the within-cluster inertia J_K, which K-means aims to minimize:

$$J_K = \sum_{k=1}^{K} \sum_{i \in C_k} \|x_i - m_k\|^2$$

$$\text{For } K = 2\text{: } J_K = n\bar{y}^2 - \frac{J_D}{2} \qquad \text{For } K > 2\text{: } J_K = \text{Tr}(X^T X) - \text{Tr}(H_K^T X^T X H_K)$$

This reformulation reveals the link between PCA and K-means. **Why ?**
By relaxing the binary constraint on cluster indicators h_k, the K-means objective J_K is reformulated as an eigenvalue problem. Specifically, minimizing J_K reduces to this optization problem :

$$\max_{Q_{K-1}} \text{Tr}(Q_{K-1}^T (Y^T Y) Q_{K-1})$$

The solution lies in the space spanned by the first $K-1$ eigenvectors of the covariance matrix $Y^T Y$. To illustrate, using the eigenvectors from PCA to determine clusters is straightforward for $K = 2$. The clusters C_1, C_2 are given by:

$$C_1 = \{i \mid v_1(i) \leq 0\} \quad \text{and} \quad C_2 = \{i \mid v_1(i) > 0\}$$

2.2 PCA-guided search for K-means

The approach proposed in [3] focuses on the initialization of the algorithm, aiming for faster convergence towards a better clustering solution. Random initialization carries the risk of

converging to a local minimum far from the optimal solution. To mitigate this, methods like K-means++ have been developed.
This paper leverages the relationship between K-means and PCA to propose a new initialization strategy. The method first reduces the dimensionality of the data to the number of clusters using PCA. K-means is then performed in this reduced space, where the risk of falling into a poor local minimum is lower. The resulting centroids are projected back into the original space to serve as initial centroids for K-means. This approach not only improves convergence but also accelerates computation, as K-means in the reduced space is less costly. When projected back, the centroids are already close to the final solution.

We tested several initialization methods for K-means centroids in the reduced space, including:

- *Random R1*: Points are randomly assigned to clusters, and centroids are computed for each cluster.
- *Random R2*: K points are randomly selected as centroids.
- *K-means++*: The first centroid is chosen randomly, with subsequent centroids selected iteratively based on a probability proportional to the distance from the nearest existing centroid.
- *KKZ*: The first centroid is the point with the largest norm, and subsequent centroids are chosen iteratively as the farthest points from existing centroids, with no randomness here.

3 Discussion and extensions

3.1 Limitations of K-means

Let Z_{ik} be an indicator variable equal to 1 if individual i belongs to cluster k, and 0 otherwise. This variable is unknown, and the objective of K-means is to estimate it using the observed data X_i. K-means assumes that each cluster has an equal probability of assignment: $\mathbb{P}(Z_{ik} = 1) = \pi = 1/K \ \forall i, k$. Furthermore, it assumes each cluster has its own mean μ_k, while all clusters share the same uniform spread in all directions, represented by an identical covariance matrix $\Sigma_k = I \ \forall k$, where I is the identity matrix.

These strong assumptions limit K-means' flexibility and can lead to poor clustering performance, especially in reduced-dimensional spaces where data points are closer together, amplifying these constraints. While the PCA-guided search helps by clustering in reduced dimensions, it still relies on the inherent assumptions of K-means, which may result in suboptimal initialization and clustering performance.

3.2 Extensions for PCA-guided search

The PCA-guided search improves clustering results. However, it depends on an initial step using random K-means initialization. We propose extending this approach by comparing the various K-means initialization methods discussed in the paper, applied in the reduced-dimensional space. Nonetheless, the limitations of K-means are even more pronounced in lower-dimensional spaces, often leading to unrealistic assumptions during initialization.

To address these limitations, we propose replacing the K-means initialization step in the reduced space with a Gaussian Mixture Model (GMM) with the Expectation-Maximization (EM) algorithm. Unlike K-means, GMM allows for cluster-specific covariance structures and provides a probabilistic assignment of data points to clusters, making it more flexible and realistic. This initialization method ensures that centroids in the reduced space are better aligned with the underlying data distribution. Once initialized with GMM, the centroids can be projected back into the original space to serve as the starting points for K-means. This hybrid

approach balances the flexibility and accuracy of GMM with the computational efficiency of K-means in higher dimensions, resulting in improved clustering performance and a more robust initialization process.

3.3 GMM and EM

3.3.1 Gaussian Mixture Models (GMM)

In a GMM, the latent variables introduced earlier are assumed to follow a multinomial distribution. Specifically, $\mathbb{P}(Z_{ik} = 1) = \pi_k \ \forall i$, which can be interpreted as each cluster having a distinct "attractiveness" level. Moreover, the observed data X_i is modeled as following a Gaussian distribution conditional on the cluster it belongs to: $X_i|Z_{ik} = 1 \sim \mathcal{N}(\mu_k, \Sigma_k)$. Unlike K-means, GMM allows Σ_k to have non-zero covariances and feature-specific variances that can vary across clusters. Figure 1 illustrates these differing assumptions in a clustering task performed in a 2D reduced space, using the MNIST dataset. We observe that, K-means assumes clusters to be spherical, while EM (GMM) allows to detect clusters with elliptical shapes or different orientations in space.

3.3.2 EM algorithm

The EM (Expectation-Maximization) algorithm for GMM maximizes the exact data log-likelihood of the model. It relies on two critical properties regarding the law of the cluster indicator variables Z_i given the observations and parameters: their independence and their analytical multinomial distribution. Detailed steps of the algorithm are provided in Appendix F.

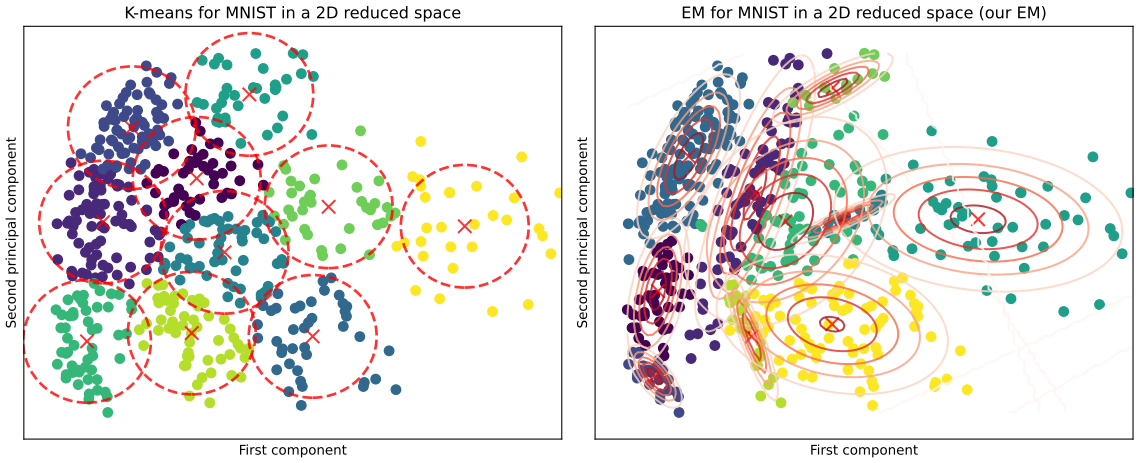

Figure 1: Illustration of EM and K-means clustering in a PCA reduced space on MNIST data

3.4 Experiments

As discussed in subsection 3.2, we initialized the PCA-guided search with a more robust model: a GMM, using the EM algorithm. This algorithm is applied to the PCA-reduced data, where we retained 20 dimensions in the reduced space. This choice was made empirically, considering the percentage of explained variance.

Following the methodology in [3], we performed 1000 runs of various clustering algorithms. Figure 2 presents the obtained inertias sorted in decreasing order, and Appendix C displays the optimal results. We included K-means, K-means++, and PCA-guided search with random initialization in the reduced space. Additionally, we introduced algorithms using alternative initializations for K-means in the reduced space, most notably the GMM-based initialization,

as well as K-means++ and KKZ initialization.

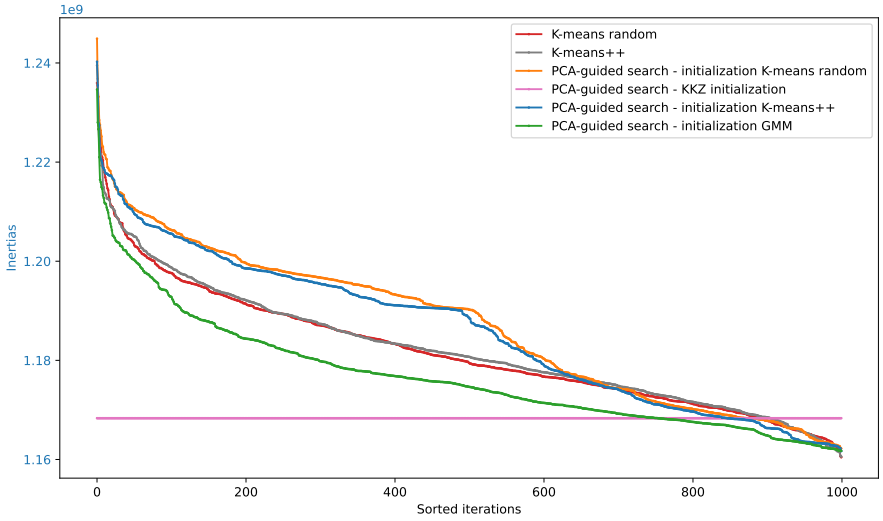

Figure 2: Decreasing inertias in 1000 rollouts of different clustering algorithms

The results show that the GMM-based approach clearly outperforms the alternatives in terms of inertia. However, when focusing on the best result across all iterations, all random-based algorithms achieve a similar minimal inertia. This can be attributed to the relatively small size of the dataset: over 1000 runs, randomness allows convergence to the same optimal solution at least once. However, given the shape of the curves, we can expect that with a larger dataset, the EM-based approach would yield a more significant advantage. Besides, In cases where repeated iterations of the algorithm are computationally impractical, the GMM-based initialization is clearly the preferred choice for achieving superior results with fewer runs.

3.5 Adding clusters

In our experiments, we initially considered $K = 10$, corresponding to the 10 classes in the MNIST dataset. However, here we explore the potential benefits of increasing the number of clusters. As shown in Figure 4, some digits appear to occupy multiple clusters. For instance, the digit 0 seem to be assigned to two clusters, while digits 3 and 4 seem to lack dedicated clusters. Digits 9 and 7 show poor separation, and two clusters appear to act as catch-all classes. This suggests that allowing certain digits to occupy multiple clusters could "free up" space for new distinct clusters, improving separation and addressing differences in writing styles. The goal is to strike a balance: achieving better separation while avoiding an excessive number of clusters that could fragment meaningful groups.

Determining the optimal number of clusters K remains a challenge, as no perfect method exists. To address this, we used empirical criteria based on the Bayesian Information Criterion (BIC) and Akaike's Information Criterion (AIC). The EM algorithm was run for values of K ranging from 10 to 20 in a reduced-dimensional space (15 dimensions retained). For each K, we ran the algorithm 50 times, recording the highest log-likelihood result and computing the associated criterion. If $L_k(\hat{\pi}, \hat{\theta})$ denotes the log-likelihood for k clusters, the BIC is given by: $BIC(k) = L_k(\hat{\pi}, \hat{\theta}) - (M_k/2)log(n)$, where $M_k = k - 1 + kd + kd(d+1)/2$ represents the number of free parameters in the model, with $d = 15$ dimensions and $n = 500$ images. The AIC criterion is $AIC(k) = L_k(\hat{\pi}, \hat{\theta}) - M_k$.

The results showed that AIC displayed a strictly increasing trend, while BIC produced a strictly decreasing curve. To address this issue, we adjusted the penalization with a balanced

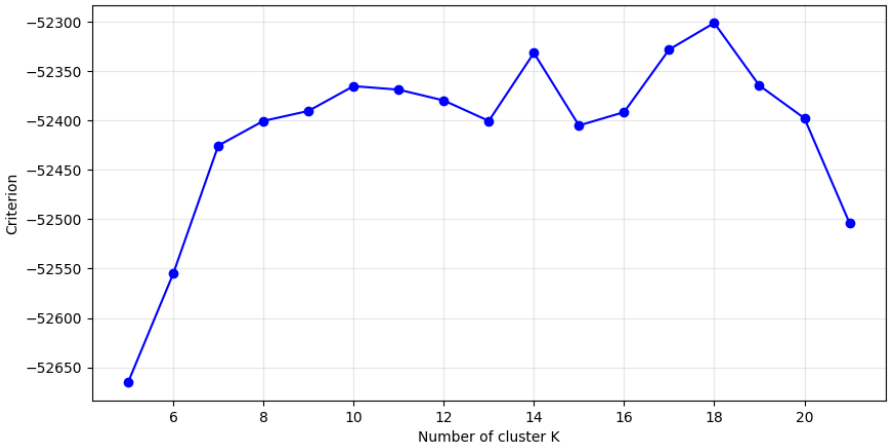

Figure 3: Revisited BIC criterion for different values of K

intermediate weighting. Specifically, we adopted the criterion: $C(k) = L_k(\hat{\pi}, \hat{\theta}) - (M_k/4)log(n)$. In other words, we used half of the BIC penalization.

The graph in Figure 3 illustrates the evolution of our criterion across different values of K. The maximum value is reached at $K = 18$. Consequently, we performed a new PCA-guided clustering using EM initialization, with 18 clusters. The Figure 5 (in Appendix D) displays sample images from each cluster. The improvement is clear: every digit now has at least one dedicated cluster, and there is little mixing between classes.

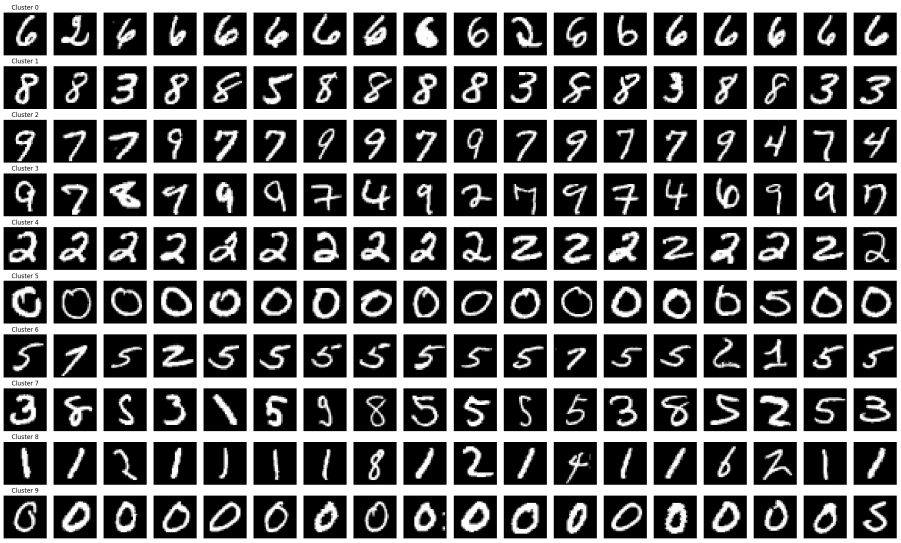

Figure 4: Samples from each cluster (by raw) with the GMM approach ($K = 10$)

4 Conclusion

In this work, we explored the close relationship between dimensionality reduction and clustering. We highlighted the critical role of initialization in clustering algorithms and demonstrated how dimensionality reduction can aid in this process. Furthermore, we implemented and proposed improvements, particularly through the use of GMM and the EM algorithm. However, it is worth noting that more suitable deep learning approaches might better address this problem. Unlike clustering, which operates on individual pixels and is then highly sensitive to translations, shape deformations, and augmentations, deep learning methods can leverage spatial and structural information to provide more robust solutions.

References

[1] Chris Ding and Xiaofeng He. K-means clustering via principal component analysis. In *Twenty-first international conference on Machine learning - ICML '04*, New York, New York, USA, 2004. ACM Press.

[2] Ade Jamal, Annisa Handayani, Ali Akbar Septiandri, Endang Ripmiatin, and Yunus Effendi. Dimensionality reduction using PCA and K-Means clustering for breast cancer prediction. *Lontar Komputer Jurnal Ilmiah Teknologi Informasi*, page 192, December 2018.

[3] Qin Xu, Chris Ding, Jinpei Liu, and Bin Luo. PCA-guided search for k-means. *Pattern Recognit. Lett.*, 54:50–55, March 2015.

Appendix

A PCA

The **Principal Component Analysis** (PCA) algorithm is a statistical method used to reduce the dimensionality of data while preserving as much variance as possible. The main steps are as follows:

1. **Data Centering**: The data is centered by subtracting the mean of each feature. Given a data matrix $X \in \mathbb{R}^{n \times d}$, where n is the number of samples and d is the number of features, each data point x_i is centered as:

$$\tilde{x}_i = x_i - \frac{1}{n} \sum_{i=1}^{n} x_i.$$

2. **Covariance Matrix**: The covariance matrix $C \in \mathbb{R}^{d \times d}$ of the centered data matrix $\widetilde{X}$ is computed as:

$$C = \frac{1}{n} \widetilde{X}^T \widetilde{X},$$

3. **Eigenvalue and Eigenvector Decomposition**: The covariance matrix C is decomposed to find its *eigenvalues* $\lambda_1, \lambda_2, \ldots, \lambda_d$ and their associated *eigenvectors* $v_1, v_2, \ldots, v_d$, such that $\forall i \in \{1, ..., d\}, Cv_i = \lambda_i v_i$.

4. **Projection onto Principal Components**: The data is projected onto the first k principal components (those corresponding to the k largest eigenvalues) to obtain a reduced-dimensional representation:

$$Z = \widetilde{X} V_k,$$

   where the columns of $V_k \in \mathbb{R}^{d \times k}$ are the eigenvectors corresponding to the k largest eigenvalues.

The projected data Z retains most of the variance from the original data while reducing its dimensionality.

B K-means algorithm

The **k-means clustering** algorithm is a popular unsupervised learning method used to partition a dataset into k distinct clusters based on feature similarity. The objective is to assign clusters in such a way as to minimize the total within-cluster variance. The steps of the algorithm are as follows:

1. **Initialization**: Selects k initial cluster centroids $\mu_1, \mu_2, \ldots, \mu_k$ given a chosen heuristic. This initialization has a huge importance on the convergence of the algorithm, which we focus on the project.

2. **Cluster Assignment Step**: Each data point x_i is assigned to the cluster whose centroid is the closest, according to the Euclidean distance. This step minimizes the within-cluster sum of squares and the cluster assignment for the i-th data point is:

$$c_i = \arg\min_{j \in [\![1,k]\!]} \|x_i - \mu_j\|_2^2$$

3. **Centroid Update Step**: The centroids of each cluster is recomputed by taking the mean of all points assigned to that cluster:

$$\mu_j = \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i$$

   where C_j is the set of data points assigned to cluster j.

4. **Convergence Check**: The cluster assignment and centroid update steps are repeated until the centroids no longer change (or the change is below a predefined threshold), or a maximum number of iterations is reached.

If there is convergence, the algorithm return a local minima of the total within-cluster variance J:

$$J = \sum_{j=1}^{k} \sum_{x_i \in C_j} \|x_i - \mu_j\|^2$$

C Inertia depending on models

Models	Min Inertia (10e9)	Execution time
PCA-guided Search with Random R2	1.1622	1.59
PCA-guided Search with KKZ	1.1683	2.27
PCA-guided Search with KR	1.1616	2.82
PCA-guided Search with K-means++	1.1910	1.21
PCA-guided Search with GMM	1.1617	2.35

Table 1: Results for the MNIST Dataset

D Results for K=18 clusters

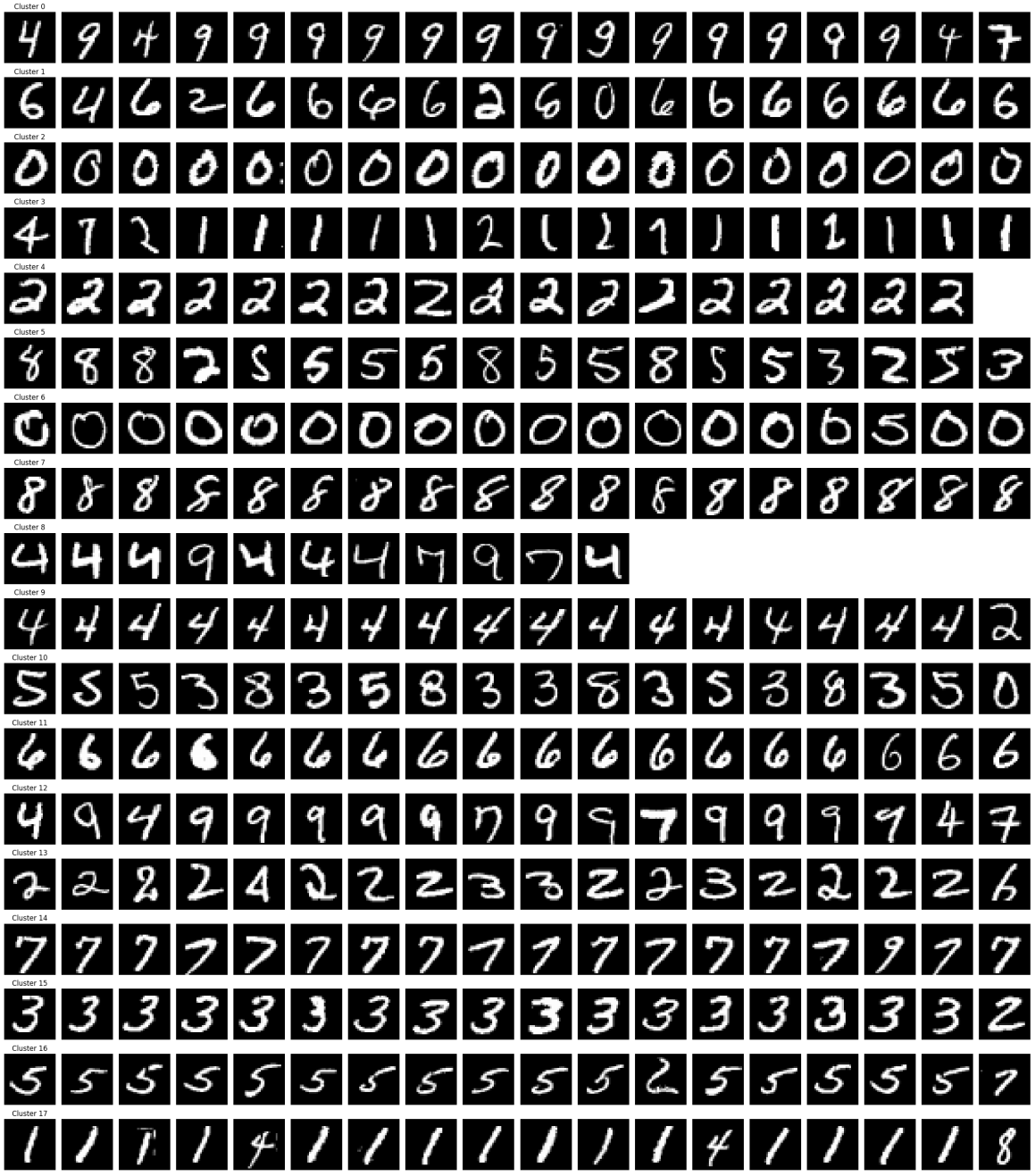

Figure 5: Samples from each cluster (by raw) with the GMM approach ($K = 18$)

E PCA as a continuous solution to K-means clustering: proof

In K-means, we minimize the **intra-cluster inertia** J_K, which corresponds to the sum of Euclidean distances between each point and its associated centroid:

$$J_K = \sum_{k=1}^{K} \sum_{i \in C_k} \|x_i - m_k\|^2$$

To help proving the theoretical connection between the K-means algorithm and PCA, the article introduces some notations, particularly for the singular value decomposition:

- X represents the original data matrix.
- $Y = (y_1, \cdots, y_n)$, où $y_i = x_i - \bar{x}$, represents the centered data matrix.
- The covariance matrix (ignoring the factor $1/n$) is given by: YY^T.
- The principal directions u_k eand the principal components v_k are the eigenvectors satisfying:

$$YY^T u_k = \lambda_k u_k, \quad Y^T Y v_k = \lambda_k v_k, \quad v_k = \frac{Y^T u_k}{\lambda_k^{1/2}} \tag{1}$$

- These are the equations defining the Singular Value Decomposition (SVD) of Y :

$$Y = \sum_k \lambda_k u_k v_k^T$$

To better understand this proof, it is important to focus on this theoretical connection when the number of clusters K is only 2.

E.1 Proof for K = 2

Firstly, J_K is not convex, so it may be helpful to rewrite this intra-cluster inertia:

Lemma E.1 : Rewriting J_K

Let $\forall K > 1$, $n_k = |C_k|$ and $m_k = \frac{1}{n_k} \sum_{i \in C_k} x_i$, we have :

$$J_K = \sum_{k=1}^{K} \sum_{i \in C_k} \|x_i - m_k\|^2 = \sum_{k=1}^{K} \sum_{i,j \in C_k} \frac{\|x_i - x_j\|^2}{2n_k}$$

Proof.

Let $k \in [\![1, K]\!]$, let's expand the second expression for cluster k.

$$\begin{aligned}
\sum_{i,j\in C_k} \frac{\|x_i - x_j\|^2}{2n_k} &= \sum_{i,j\in C_k} \frac{\|x_i\|^2 - 2\langle x_i, x_j\rangle + \|x_j\|^2}{2n_k} \\
&= \frac{1}{2n_k} \sum_{i,j\in C_k} \|x_i\|^2 + \frac{1}{2n_k} \sum_{i,j\in C_k} \|x_j\|^2 - \frac{1}{n_k} \sum_{i,j\in C_k} \langle x_i, x_j\rangle \\
&= \frac{1}{n_k} \sum_{i,j\in C_k} \|x_i\|^2 - \frac{1}{n_k} \sum_{i,j\in C_k} \langle x_i, x_j\rangle \\
&= \sum_{i\in C_k} \|x_i\|^2 - \underbrace{\frac{1}{n_k} \sum_{i,j\in C_k} \langle x_i, x_j\rangle}_{\triangleq S_1} \qquad \text{(because } |C_k| = n_k\text{)}
\end{aligned}$$

Let's simplify S_1 :

$$S_1 = \frac{1}{n_k} \sum_{i\in C_k} \sum_{j\in C_k} \langle x_i, x_j\rangle = \sum_{i\in C_k} \langle x_i, m_k\rangle = \left\langle \sum_{i\in C_k} x_i, m_k \right\rangle = n_k \langle m_k, m_k\rangle = n_k \|m_k\|^2$$

And so :

$$\sum_{i,j\in C_k} \frac{\|x_i - x_j\|^2}{2n_k} = \sum_{i\in C_k} \|x_i\|^2 - n_k \|m_k\|^2$$

Expanding the first expression for the same $k \in [\![1, K]\!]$ gives:

$$\begin{aligned}
\sum_{i\in C_k} \|x_i - m_k\|^2 &= \sum_{i\in C_k} (\|x_i\|^2 - 2\langle x_i, m_k\rangle + \|m_k\|^2) \\
&= \sum_{i\in C_k} \|x_i\|^2 - 2 \sum_{i\in C_k} \langle x_i, m_k\rangle + n_k \|m_k\|^2 \\
&= \sum_{i\in C_k} \|x_i\|^2 - 2n_k \langle m_k, m_k\rangle + n_k \|m_k\|^2 \\
&= \sum_{i\in C_k} \|x_i\|^2 - n_k \|m_k\|^2
\end{aligned}$$

Thus, we have proven that for all $\forall k \in [\![1, K]\!]$, we have:

$$\boxed{\sum_{i\in C_k} \|x_i - m_k\|^2 = \sum_{i,j\in C_k} \frac{\|x_i - x_j\|^2}{2n_k}}$$

Summing over all finite k, the proof of the lemma is complete.

□

To obtain a more interesting and usable expression for the intra-cluster inertia J_K, the article [1] introduces an inertia J_D.

For K=2, this inertia is defined as:

$$J_D = \frac{n_1 n_2}{n} \left[2\frac{d(C_1, C_2)}{n_1 n_2} - \frac{d(C_1, C_1)}{n_1^2} - \frac{d(C_2, C_2)}{n_2^2} \right]$$

We also define the distance between two classes as:

$$d(C_k, C_l) = \sum_{i\in C_k} \sum_{j\in C_k} (x_i - x_j)^2$$

So, Lemma E.2 provides the desired rewriting of J_K.

Lemma E.2 : Second rewritting of J_K

Noting $n = \sum_{k=1}^{K} n_k$ and $\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$, we define $\overline{y^2} = \frac{1}{n}\sum_{i=1}^{n} \|y_i\|^2$.
Therefore, we have :

$$J_K = n\overline{y^2} - \frac{J_D}{2}$$

Proof.

First, let's recall the decomposition of the total inertia in terms of the inter-class and intra-class inertia:

$$\sum_{i=1}^{n} \|x_i - \overline{x}\|^2 = \underbrace{\sum_{k=1}^{K}\sum_{i\in C_k} \|x_i - m_k\|^2}_{\text{intra-class inertia}} + \underbrace{\sum_{k=1}^{K} n_k \|m_k - \overline{x}\|^2}_{\text{inter-class inertia}}$$

Now, $J_K = \sum_{k=1}^{K}\sum_{i\in C_k} \|x_i - m_k\|^2$. This relation can be written as:

$$\sum_{i=1}^{n} \|x_i - \overline{x}\|^2 = J_K + \sum_{k=1}^{K} n_k \|m_k - \overline{x}\|^2 \tag{2}$$

By definition:

$$\overline{y^2} = \frac{1}{n}\sum_{i=1}^{n} \|y_i\|^2 = \frac{1}{n}\sum_{i=1}^{n} \|x_i - \overline{x}\|^2$$

Multiplying by n and substituting into equation (2), we obtain:

$$n\overline{y^2} = J_K + \sum_{k=1}^{K} n_k \|m_k - \overline{x}\|^2 \tag{3}$$

We now need to link the quantity $\sum_{k=1}^{K} n_k \|m_k - \overline{x}\|^2$ to the expression of J_D. We aim to prove that:

$$\boxed{\sum_{k=1}^{K} n_k \|m_k - \overline{x}\|^2 = \frac{J_D}{2}}$$

Proof for $K = 2$:

Consider two clusters C_1 and $C_2$of sizes n_1 and n_2 with $n = n_1 + n_2$. The given expression of J_D (for K=2) is:

$$J_D = \frac{n_1 n_2}{n}\left[2\frac{d(C_1, C_2)}{n_1 n_2} - \frac{d(C_1, C_1)}{n_1^2} - \frac{d(C_2, C_2)}{n_2^2}\right]$$

1. **Decompostion of $d(C_1, C_2)$:**

   Let's write $\|x_i - x_j\|^2$ by introducing the centroids m_1 et m_2:

   $$\|x_i - x_j\|^2 = \|(x_i - m_1) + (m_1 - m_2) + (m_2 - x_j)\|^2$$

   When summing $i \in C_1$ and $j \in C_2$, the linear terms in $(x_i - m_1)$ and $(m_2 - x_j)$ disappear because $\sum_{i\in C_1}(x_i - m_1) = 0$ and $\sum_{j\in C_2}(m_2 - x_j) = 0$. Therefore, the cross

terms vanish. Finally:

$$d(C_1, C_2) = n_1 n_2 \|m_1 - m_2\|^2 + n_2 \sum_{i \in C_1} \|x_i - m_1\|^2 + n_1 \sum_{j \in C_2} \|x_j - m_2\|^2$$

Dividing by $n_1 n_2$ gives:

$$\frac{d(C_1, C_2)}{n_1 n_2} = \|m_1 - m_2\|^2 + \frac{\sum_{i \in C_1} \|x_i - m_1\|^2}{n_1} + \frac{\sum_{j \in C_2} \|x_j - m_2\|^2}{n_2}$$

2. **Substitution in J_D :**

   Replacing $d(C_1, C_2)/(n_1 n_2)$ and each $d(C_k, C_k)/n_k^2$:

   - We have $\frac{d(C_k, C_k)}{n_k^2} = 2\frac{\sum_{i \in C_k} \|x_i - m_k\|^2}{n_k}$.
   - The expression inside the brackets:

$$\begin{aligned} & 2\left(\|m_1 - m_2\|^2 + \frac{\sum_{i \in C_1} \|x_i - m_1\|^2}{n_1} + \frac{\sum_{j \in C_2} \|x_j - m_2\|^2}{n_2} \right) \\ & - 2\frac{\sum_{i \in C_1} \|x_i - m_1\|^2}{n_1} - 2\frac{\sum_{j \in C_2} \|x_j - m_2\|^2}{n_2} \\ = \; & 2\|m_1 - m_2\|^2 \end{aligned}$$

   Hence $J_D = \frac{n_1 n_2}{n} \cdot 2\|m_1 - m_2\|^2$, which implies:

$$\frac{J_D}{2} = \frac{n_1 n_2}{n} \|m_1 - m_2\|^2$$

3. **Relation between $\sum_k n_k \|m_k - \bar{x}\|^2$ et $\|m_1 - m_2\|^2$:**

   We have:

$$\begin{aligned} & \bar{x} = \frac{n_1 m_1 + n_2 m_2}{n} \\ \Longrightarrow \quad & n_1(m_1 - \bar{x}) + n_2(m_2 - \bar{x}) = 0 \end{aligned}$$

   This implies a linear relationship between $m_1 - \bar{x}$ and $m_2 - \bar{x}$. By manipulating this relationship (and performing a small algebraic calculation), we obtain:

$$\sum_{k=1}^{2} n_k \|m_k - \bar{x}\|^2 = \frac{n_1 n_2}{n} \|m_1 - m_2\|^2$$

   Therefore:

$$\frac{J_D}{2} = \sum_{k=1}^{2} n_k \|m_k - \bar{x}\|^2$$

4. **Conclusion :**

   We had in (3) :

$$J_K = n\bar{y}^2 - \sum_{k=1}^{2} n_k \|m_k - \bar{x}\|^2$$

   and then:

$$\boxed{J_K = n\bar{y}^2 - \frac{J_D}{2}}$$

   This is exactly the desired equality.

By extension, for K clusters, a similar reasoning (or by induction) allows us to generalize this result. Thus, we obtain:

$$\sum_{k=1}^{K} n_k \|m_k - \overline{x}\|^2 = \frac{J_D}{2}. \tag{4}$$

It is then enough to substitute (4) in (3) :

$$J_K = n\overline{y^2} - \sum_{k=1}^{K} n_k \|m_k - \overline{x}\|^2 = n\overline{y^2} - \frac{J_D}{2}$$

Which prove the second rewriting of J_K.

□

We recall that the goal of the K-means algorithm is to minimize the intra-cluster inertia J_K. But, Lemma E.2 allows us to write J_K as $n\overline{y^2} - \frac{J_D}{2}$.

- $\overline{y^2}$ is by definition a constant.
- We showed in the proof of the lemma E.2 that :

$$J_D = \frac{2n_1 n_2}{n} \|m_1 - m_2\|^2$$

  Hence J_D is **always positive**.

Thus, we can summarize all this information in a single very useful theorem :

Theorem E.1 : K-means seen as a maximization

For $K = 2$, minimization of K-means cluster objective function J_K is equivalent to maximization of the distance objective J_D, which is always positive.

The theorem E.1 leads to a solution of the K-means algorithm through Principal Component Analysis (PCA). This is what we will directly observe in the following theorem E.2.

Theorem E.2 : K-means clustering via PCA

For K-means clustering where $K = 2$, the continuous solution of the cluster indicator vector is the principal component v_1, i.e., clusters C_1, C_2 are given by

$$C_1 = \{i \mid v_1(i) \leq 0\} \quad \text{et} \quad C_2 = \{i \mid v_1(i) > 0\}$$

Proof.

We will not detail all the calculations here, but our comprehension of the intuition behind the proof.

Consider the discrete indicator vector $q \in \mathbb{R}^n$ such that:

$$q(i) = \begin{cases} \sqrt{\dfrac{n_2}{n_1 n}} & \text{si } i \in C_1 \\ -\sqrt{\dfrac{n_1}{n_2 n}} & \text{si } i \in C_2 \end{cases}$$

Its definition is directly in link with the definition of J_D. Indeed, we easily have:

$$q^T D q = -J_D$$

Where $D = (d_{ij})$ is the matrix of squared distances, $d_{ij} = \|x_i - x_j\|^2$.

This vector satisfies $\sum_i q(i) = 0$ and $\|q\|_2 = 1$, and it is chosen in such a way that the objective J_K (or J_D) is expressed in terms of q. The problem then becomes finding q that minimizes J_K (or maximizes J_D) which resembles a difficult combinatorial optimization problem due to the constraint on the values of q.

Transition to the continuous problem :

To path through this difficulty, **we relax the discrete constraint on** q and allow $q(i)$ to take continuous real values in the range $[-1, 1]$. We then consider the optimization problem:

$$\min_{\substack{q \in [-1,1]^n \\ s.c. \sum_i q(i)=0,\ \|q\|_2=1}} q^T D q \tag{P}$$

The optimal vector q is then associated with **the smallest eigenvalue** (the most negative) of the matrix D. However, to obtain a more usable solution, we introduce a centered version of D, denoted $\widehat{D}$, obtained by subtracting the row means, column means, and the overall mean. After an algebraic calculation, it can be shown that this centered matrix is written in the form:

$$\boxed{\widehat{D} = -2Y^T Y}$$

But, the eigenvectors of YY^T correspond precisely to the principal components, which are the directions of greatest variance in the data.

This matrix $\widehat{D}$ has the convenient property of being defined such that:

$$q^T \widehat{D} q = q^T D q = -J_D$$

And, since $\widehat{D} = -2Y^T Y$, it is therefore sufficient to solve the following new optimization problem:

$$\boxed{\max_{\substack{q \in [-1,1]^n \\ \text{s.t. } \sum_i q(i)=0,\ \|q\|_2=1}} q^T (Y^T Y) q} \tag{P'}$$

Therefore, minimizing J_K (or maximizing J_D) in this continuous space amounts to finding **the eigenvector associated with the largest eigenvalue of** YY^T, in other words, the first principal axis of PCA (denoted u_1). This vector provides a continuous partition of the data.

Returning to a discrete solution:

To return to a binary partition, we simply classify the points based on the sign of their coordinate on this first axis:

$$C_1 = \{i \mid v_1(i) \leq 0\} \quad \text{et} \quad C_2 = \{i \mid v_1(i) > 0\}$$

This classification naturally separates the dataset into two groups, C_1 and C_2.

□

In summary, the continuous solution to the K-means problem with $K = 2$ is provided by the first eigenvector of the covariance matrix, which is, the first principal component. By assigning points based on the sign of this component, we obtain the desired partition, showing **the deep connection between K-means and PCA**.

E.2 Proof for arbitrary K

In the general case where $K > 2$, the objective of the K-means remains the same: we are looking for a partition of $\{1, \ldots, n\}$ into K clusters of sizes n_k, with centroids $m_k = \frac{1}{n_k} \sum_{i \in C_k} x_i$, in order to minimize the function

$$J_K = \sum_{k=1}^{K} \sum_{i \in C_k} \|x_i - m_k\|^2$$

To represent the membership of points to the clusters, we first introduce a family of K discrete indicator vectors $h_k \in \mathbb{R}^n$ defined by:

$$h_k = (0, \ldots, 0, \underbrace{1, \ldots, 1}_{n_k}, 0, \ldots, 0)^T / \sqrt{n_k}$$

Each h_k has exactly n_k components equal to $1/\sqrt{n_k}$ and the others are zero, representing the membership of a contiguous block of points to the cluster C_k. These vectors are then grouped into a matrix $H_K = (h_1 \cdots h_K)$.

By rewriting J_K using H_K, it is shown in the paper [1] that:

$$\boxed{J_K = \mathrm{Tr}(X^T X) - \mathrm{Tr}(H_K^T X^T X H_K)}$$

However, the vectors h_k have redundancy. Indeed, $\sum_{k=1}^{K} \sqrt{n_k} h_k = \sqrt{n}\, e$, where $e = (1, \ldots, 1)^T$. This introduces a linear dependence. To eliminate this and obtain a solution closer to the one seen in the case $K = 2$, we perform **an orthonormal linear transformation** $T \in \mathbb{R}^{K \times K}$ such that the last column of T is $t_K = (\sqrt{n_1/n}, \ldots, \sqrt{n_K/n})^T$. Thus, we define:

$$Q_K = H_K T = (q_1, \ldots, q_K) \tag{5}$$

By this transformation, we obtain a set of indicator vectors q_k that satisfy a stricter criterion, analogous to the zero-sum condition (as in the case $K = 2$). For $K = 2$, this procedure reduces the problem to the vectors already studied. For $K > 2$, we encounter a similar situation, but this time with $K - 1$ **continuous indicator vectors**.

Transition to the continuous problem:

Following the logic used for $K = 2$, we can show that minimizing J_K by relaxing the binary constraint on the h_k (or equivalently on the q_k) amounts to searching for the eigenvectors corresponding to the $K - 1$ largest eigenvalues of the covariance matrix Y^TY (where Y is the centered data matrix).

As in the case $K = 2$, it is possible to rewrite J_K using the covariance matrix Y^TY:

$$J_K = \mathrm{Tr}(Y^TY) - \mathrm{Tr}(Q_{K-1}^T Y^T Y Q_{K-1})$$

Thus, it is sufficient to solve the following optimization problem:

$$\boxed{\max_{Q_{K-1}} \mathrm{Tr}(Q_{K-1}^T (Y^TY) Q_{K-1})} \qquad \text{(P")}$$

Problem (P") is therefore a generalization of the optimization problem (P') that we encountered in the case $K = 2$.

The corresponding theorem is stated as follows:

Theorem E.3 : K-means clustering via PCA (arbitrary K)

Let Y^TY be the centered covariance matrix, then the continuous solution to the K-means problem lies in the space spanned by the first $K - 1$ eigenvectors $v_1, \ldots, v_{K-1}$ of Y^TY.

Returning to a discrete solution:

Although the continuous solution to the K-means problem is simply obtained via the first $K - 1$ principal components v_k, **it remains difficult to revert to a perfect discrete solution**, that is, to recover the indicator vectors h_k and thus the exact partition of the K clusters. The problem mainly lies in the calculation of the transformation T and the return to the discrete form.

The paper proposes a trick that bypasses the need to know T and directly reconstructs H_K. Instead of trying to recover H_K from the q_k's, we aim to recover $H_K H_K^T$. Using equation (5), we have:

$$H_K H_K^T = (H_K T_K)(H_K T_K)^T = Q_K Q_K^T$$

Here, the q_k's are the discrete indicator vectors defined after the transformation T. The idea is to directly use their "continuous versions" v_k (the principal components) instead of the q_k's. As shown in the paper, we approximate:

$$Q_{K-1} Q_{K-1}^T \simeq V_{K-1} V_{K-1}^T = \sum_{k=1}^{K-1} v_k v_k^T,$$

This construction provides a practical tool: it is no longer necessary to explicitly know T or reconstruct each h_k. Instead, we use P to help us find a consistent discrete solution. Even though this solution remains approximate and often requires additional steps (such as heuristics or discretization algorithms), this approach allows us to bypass the initial difficulty and remains very practical.

F EM algorithm for a GMM model

The EM algorithm allows to estimate the parameters θ of a mixture of m Gaussians from observed data $(X_i)_{i\in[\![1,n]\!]}$.

F.1 Initialization

Before beginning the iterations, it is it is necessary to initialize the parameters $\theta = \{\alpha_j, \mu_j, \Sigma_j \mid j \in [\![1, m]\!]\}$:

- α_j represents the proportion of the j-th gaussian component,
- μ_j is the mean of the jth gaussian,
- Σ_j is the covariance matrix of the jth gaussian.

The initialization can be random or based on a chosen heuristic.

F.2 E Step (Expectation)

The E step consists of computing the expectation of the conditional log-likelihood of the latent variables, given the parameters estimated in the previous iteration, denoted $\theta^{(q)}$.

$$Q(\theta, \theta^{(q)}) = \mathbb{E}_{Z\,|\,X,\theta^{(q)}}\left[\log \mathbb{P}(X, Z; \theta)\right]$$

In the case of a Gaussian mixture, this involves determining the probability of each observation X_i belonging to the jth Gaussian component, which we denote by $\omega_{ij}^{(q)}$:

$$\omega_{ij}^{(q)} = \mathbb{P}\left(Z_i = j \mid X_i = x_i, \theta^{(q)}\right).$$

Using Bayes' inversion rule, we obtain:

$$\begin{aligned}
\omega_{ij}^{(q)} &= \mathbb{P}\left(Z_i = j \mid X_i = x_i, \theta^{(q)}\right) \\
&= \frac{\mathbb{P}\left(Z_i = j, X_i = x_i \mid \theta^{(q)}\right)}{\mathbb{P}\left(X_i = x_i \mid \theta^{(q)}\right)} \\
&= \frac{\mathbb{P}(Z_i = j \mid \theta^{(q)})\,\mathbb{P}(X_i = x_i \mid Z_i = j, \theta^{(q)})}{\sum_{\ell=1}^{m} \mathbb{P}(Z_i = \ell \mid \theta^{(q)})\,\mathbb{P}(X_i = x_i \mid Z_i = \ell, \theta^{(q)})} \qquad \text{(law of total probability)} \\
&= \frac{\alpha_j^{(q)}\,\mathcal{N}(x_i \mid \mu_j^{(q)}, \Sigma_j^{(q)})}{\sum_{\ell=1}^{m} \alpha_\ell^{(q)}\,\mathcal{N}(x_i \mid \mu_\ell^{(q)}, \Sigma_\ell^{(q)})}
\end{aligned}$$

And therefore:

$$\boxed{\omega_{ij}^{(q)} = \frac{\alpha_j^{(q)}\,\mathcal{N}(x_i \mid \mu_j^{(q)}, \Sigma_j^{(q)})}{\sum_{\ell=1}^{m} \alpha_\ell^{(q)}\,\mathcal{N}(x_i \mid \mu_\ell^{(q)}, \Sigma_\ell^{(q)})}}$$

The calculation is not yet complete, we need to determine $Q(\theta, \theta^{(q)})$.

Computation of $\mathbb{P}(X, Z; \theta)$:

$$\mathbb{P}(X, Z; \theta) = \prod_{i=1}^{n} \mathbb{P}(x_i, z_i; \theta) = \prod_{i=1}^{n}\prod_{j=1}^{m} \mathbb{P}(x_i, z_i; \theta)^{\mathbb{1}_{\{z_i = j\}}}$$

The notation in terms of powers using the latent variables z_i helps to simplify the calculation of the log-likelihood.

Now, let us take the logarithm of $\mathbb{P}(X, Z; \theta)$ to obtain the log-likelihood of the complete data:

$$\begin{aligned}
\log \mathbb{P}(X, Z; \theta) &= \sum_{i=1}^{n} \sum_{j=1}^{m} \mathbb{1}_{\{z_i=j\}} \log \mathbb{P}(x_i, z_i; \theta) \\
&= \sum_{i=1}^{n} \sum_{j=1}^{m} \mathbb{1}_{\{z_i=j\}} \log \left(\mathbb{P}(z_i = j; \theta) \, \mathbb{P}(x_i \mid z_i = j; \theta) \right) \\
&= \sum_{i=1}^{n} \sum_{j=1}^{m} \mathbb{1}_{\{z_i=j\}} \log \left(\alpha_j \, \mathcal{N}(x_i \mid \mu_j, \Sigma_j) \right) \qquad \text{(by definition)}
\end{aligned}$$

Transition to the conditional expectation with respect to Z_{ij}

The next step is to take the conditional expectation of this log-likelihood of the complete data with respect to the conditional distribution of Z given X and $\theta^{(q)}$:

$$\begin{aligned}
Q(\theta, \theta^{(q)}) &= \mathbb{E}_{Z|X,\theta^{(q)}} \left[\log \mathbb{P}(X, Z; \theta) \right] \\
&= \mathbb{E}_{Z|X,\theta^{(q)}} \left[\sum_{i=1}^{n} \sum_{j=1}^{m} \mathbb{1}_{\{z_i=j\}} \log \left(\alpha_j \, \mathcal{N}(x_i \mid \mu_j, \Sigma_j) \right) \right] \\
&= \sum_{i=1}^{n} \sum_{j=1}^{m} \mathbb{E}_{Z|X,\theta^{(q)}} \left[\mathbb{1}_{\{z_i=j\}} \right] \log \left(\alpha_j \, \mathcal{N}(x_i \mid \mu_j, \Sigma_j) \right)
\end{aligned}$$

Because $\mathbb{E}_{Z|X,\theta^{(q)}} \left[\mathbb{1}_{\{z_i=j\}} \right] = \mathbb{P}\left(Z_i = j \mid X_i = x_i, \theta^{(q)} \right) = \omega_{ij}^{(q)}$, we finally obtain :

$$\boxed{Q(\theta, \theta^{(q)}) = \sum_{i=1}^{n} \sum_{j=1}^{m} \omega_{ij}^{(q)} \log \left(\alpha_j \, \mathcal{N}(x_i \mid \mu_j, \Sigma_j) \right)}$$

This expression of $Q(\theta, \theta^{(q)})$ will be used in the M step to update the parameters θ in maximizing this function.

F.3 M step (Maximization)

The M step consist in maximizing the function Q given the parameters θ.

$$\theta^{(q+1)} = \arg\max_{\theta} Q(\theta, \theta^{(q)})$$

To do this, we use the values of $\omega_{ij}^{(q)}$ computed in the E-step to obtain the new estimates of the parameters $\theta^{(q+1)}$:

$$Q(\theta, \theta^{(q)}) = \sum_{i=1}^{n} \sum_{j=1}^{m} \omega_{ij}^{(q)} \log \left(\alpha_j \, \mathcal{N}(x_i \mid \mu_j, \Sigma_j) \right).$$

By differentiating this expression with respect to the parameters, we obtain the following update rules:

- **Proportions**:

$$\boxed{\alpha_j^{(q+1)} = \frac{1}{n} \sum_{i=1}^{n} \omega_{ij}^{(q)}}$$

- **Means**:

$$\boxed{\mu_j^{(q+1)} = \frac{\sum_{i=1}^{n} \omega_{ij}^{(q)} x_i}{\sum_{i=1}^{n} \omega_{ij}^{(q)}}}$$

- **Covariances**:

$$\boxed{\Sigma_j^{(q+1)} = \frac{\sum_{i=1}^{n} \omega_{ij}^{(q)} (x_i - \mu_j^{(q+1)})(x_i - \mu_j^{(q+1)})^T}{\sum_{i=1}^{n} \omega_{ij}^{(q)}}}$$