# Combining Mixture Models and Spectral Clustering for Data Partitioning

Julien Muzeau, Maria Oliver-Parera, Patricia Ladret, Pascal Bertolino

# Combining Mixture Models and Spectral Clustering for Data Partitioning*

Julien Muzeau[0000−0002−2221−382X], Maria Oliver-Parera[0000−0001−7921−4826], Patricia Ladret[0000−0001−9150−5319], and Pascal Bertolino[0000−0001−7690−2447]

Univ. Grenoble Alpes, CNRS, Grenoble INP, GIPSA-lab, 38000 Grenoble, France
{julien.muzeau, maria.oliver-parera, patricia.ladret, pascal.bertolino}@gipsa-lab.grenoble-inp.fr

**Abstract.** Gaussian Mixture Models are widely used nowadays, thanks to the simplicity and efficiency of the Expectation-Maximization algorithm. However, determining the optimal number of components is tricky and, in the context of data partitioning, may differ from the actual number of clusters. We propose to apply a post-processing step by means of Spectral Clustering: it allows a clever merging of similar Gaussians thanks to the Bhattacharyya distance so that clusters of any shape are automatically discovered. The proposed method shows a significant improvement compared to the classical Gaussian Mixture clustering approach and promising results against well-known partitioning algorithms with respect to the number of parameters.

**Keywords:** Gaussian mixture model, spectral clustering, Bhattacharyya coefficient, Bayesian information criterion

## 1 Introduction

Cluster analysis is a fundamental task in data science as it allows to gather individuals that show similar features. Clustering belongs to the set of unsupervised learning methods: they are among the most challenging ones in machine learning as they aim at blindly determining the label of each point. In other words, the objective is to find out which group each point belongs to, without having any ground-truth available. Clustering has been applied to a variety of fields such as community detection, segmentation and natural language processing [9].

Due to the amount of topics clustering can be applied to, numerous techniques have been proposed to tackle this problem. They can be classified as: hierarchical (agglomerative or divisive), centroid-based, density-based, graph-based or distribution-based [17]. This work focuses on distribution-based cluster analysis algorithms, which address the problem from a statistical point of view by considering the probability density of the data. In particular, we study the case of Gaussian Mixture (GM) Clustering which models the data by a combination of normal distributions: each Gaussian represents one cluster of points.

* Supported by Auvergne-Rhône-Alpes region.

Determining the right number of GM components only from the data is still a current topic of research. Three strategies exist regarding this challenge. The first one consists in initializing the Gaussian Mixture Model (GMM) with a low number of components and increasing it until convergence [21]. The second approach aims at iteratively merging components until a stopping criterion is met [6]. Finally, the third group of methods applies an optimization process to minimize a criterion over possible numbers of components [12]. Many criteria have been proposed such as the Bayesian Information Criterion (BIC) [19], the Minimum Message Length (MML) [20] or the Akaike Information Criterion (AIC) [1].

Nevertheless, the aforementioned approaches may suffer from overfitting. In fact, model selection is based on the minimization of $\mathcal{L}$, the likelihood function. Yet, adding more components leads to a decrease in $\mathcal{L}$. As a consequence, the resulting GMM often ends up with too many components: the model accurately represents the data density but overestimates the number of actual clusters. A fusion step can subsequently be added to reduce the number of components by merging similar Gaussians. In most cases, the GMM generates strong overlapping components and gathering them allows a simplification of the model without any loss of information. Hence, several methods have been developed.

In order to automatically detect the correct number of clusters, no matter their distribution, we propose a three-fold process. First, the data is approximated by a GMM, optimally selected through the minimization of the BIC, leading to an overfitted model with too many components compared to the number of clusters. Then, to decide if two components belong to the same group or not, the Bhattacharyya coefficients are computed to estimate the similarity between each pair of components of the GMM. Lastly, the final clusters are determined by merging similar Gaussians thanks to Spectral Clustering (SC).

The remainder of this paper is organized as follows: the idea of Gaussian Mixture is explained and detailed in Section 2. Section 3 describes our proposal and results are given in Section 4. Finally, Section 5 draws conclusions and gives some prospects about future work.

## 2 Gaussian Mixture Model

### 2.1 Principle

A mixture model is a probabilistic model that approximates the density of a dataset by a weighted sum of probability distributions of the same kind but differently parameterized. This article is focused on Gaussian Mixture Models, *i.e.* the aforementioned distributions are assumed to be normal in any dimension.

A Gaussian Mixture Model $\mathcal{M}$ with $K$ components can be defined as

$$\mathcal{M} = \sum_{i=1}^{K} \pi_i \, \mathcal{N}(\boldsymbol{\mu_i}; \boldsymbol{\Sigma_i}), \tag{1}$$

where $\pi_i$ is the weight associated to the $i^{\text{th}}$ component with $\sum_{i=1}^{K} \pi_i = 1$, $\mathcal{N}(\boldsymbol{\mu_i}; \boldsymbol{\Sigma_i})$ is the multivariate normal distribution with mean $\boldsymbol{\mu_i} \in \mathbb{R}^n$ and covariance matrix $\boldsymbol{\Sigma_i} \in \mathbb{R}^{n \times n}$, and $n$ represents the dimensionality of the data to be modeled. In other words, for any vector $\boldsymbol{x}$ in $\mathbb{R}^n$,

$$p(\boldsymbol{x}|\boldsymbol{\mu_i}, \boldsymbol{\Sigma_i}) = \frac{1}{(2\pi)^{n/2}|\boldsymbol{\Sigma_i}|^{1/2}} \exp\left(-\frac{C_i}{2}\right), \tag{2}$$

with $C_i = (\boldsymbol{x} - \boldsymbol{\mu_i})^T \boldsymbol{\Sigma_i}^{-1}(\boldsymbol{x} - \boldsymbol{\mu_i})$.

### 2.2 Expectation-Maximization Algorithm

When the number of components $K$ is known, it is possible to determine such a mixture model only from the data $\{\boldsymbol{x_i} \in \mathbb{R}^n, i = 1, \ldots, N\}$, as defined in Eq. (1). To do this, one makes use of the Expectation-Maximization (EM) algorithm developped by Dempster *et al.* in 1977 [4]. It consists in determining the parameters of the Gaussian Mixture, namely $\pi_1, \ldots, \pi_K$, $\boldsymbol{\mu_1}, \ldots, \boldsymbol{\mu_K}$, $\boldsymbol{\Sigma_1}, \ldots, \boldsymbol{\Sigma_K}$, that maximize the likelihood function associated to $\mathcal{M}$, in an iterative manner.

In real terms, the algorithm can be broken down into four steps.

1. **Parameters initialization**
   The classical approaches set $\pi_1 = \cdots = \pi_K = \frac{1}{K}$ and $\boldsymbol{\Sigma_1} = \cdots = \boldsymbol{\Sigma_K} = \boldsymbol{\Sigma}$ (the whole data covariance matrix). Regarding the means, it is common to initialize them with randomly chosen data points.
2. **E(xpectation) step**
   One computes here the probability $\gamma_{ik}$ of data at index $i$ being generated by component $k$ for all $i$ in $\{1, \ldots, N\}$ and for all $k$ in $\{1, \ldots, K\}$. That is

$$\gamma_{ik} = \frac{\pi_k \, \mathcal{N}(\boldsymbol{x_i}|\boldsymbol{\mu_k}; \boldsymbol{\Sigma_k})}{\sum_{j=1}^{K} \pi_j \, \mathcal{N}(\boldsymbol{x_i}|\boldsymbol{\mu_j}; \boldsymbol{\Sigma_j})}. \tag{3}$$

3. **M(aximization) step**
   The parameters are updated thanks to the previously computed probabilities. For each component $k \in [\![1; K]\!]$:

$$\pi_k = \frac{1}{N} \sum_{i=1}^{N} \gamma_{ik}, \tag{4}$$

$$\boldsymbol{\mu_k} = \frac{\sum_{i=1}^{N} \gamma_{ik} \boldsymbol{x_i}}{\sum_{i=1}^{N} \gamma_{ik}}, \tag{5}$$

$$\boldsymbol{\Sigma_k} = \frac{\sum_{i=1}^{N} \gamma_{ik} (\boldsymbol{x_i} - \boldsymbol{\mu_k})^T (\boldsymbol{x_i} - \boldsymbol{\mu_k})}{\sum_{i=1}^{N} \gamma_{ik}}. \tag{6}$$

4. Steps 2 and 3 are repeated until convergence.

### 2.3 High-Dimensional Gaussian Mixture Model

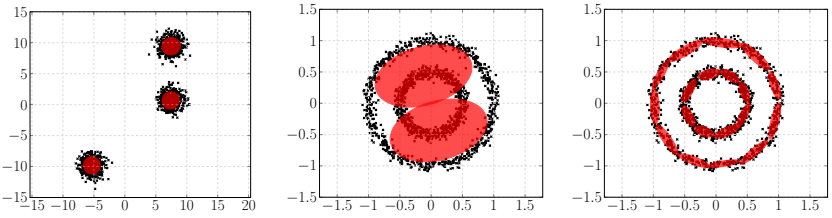
Even though the EM algorithm returns accurate outputs in most cases, issues occur when working in high-dimensional spaces (when the dimensionality is greater than five). As a matter of fact, what is called the "curse of dimensionality" causes a breakdown of any kind of distances as the number of dimensions increases. A small value, repeated by the dimensionality, leads to a huge error.

This issue appears in the Gaussian Mixture models and the EM algorithm. In particular, the Mahalanobis distance $C_i$ of Eq. (2) suffers from this phenomenon, especially due to the covariance matrix. In fact, the number of elements in such a matrix increases quadratically with respect to the dimensionality, enforcing small errors to accumulate, thus leading to an unrealistic Mahalanobis distance.

To overcome this challenge, the algorithm developed in [16] is used. The idea is to add a regularization parameter in the maximization step of the EM algorithm (see Subsection 2.2). More precisely, after the covariance matrix of each component is computed, the Graphical Lasso method, proposed by Friedman *et al.* [8], is applied to each of those matrices. The idea is as follows: an $l_1$-norm penalization term imposes the covariance matrices to become sparse, that means with a maximum of zeros outside of the diagonal. Eventually, the elements close to zero, that would yield small errors, are set to zero: an immediate consequence is the vanishing of the errors that disturbs the final computed distance.

### 2.4 Clustering via GMM

In addition to its ability to approximate data set distributions, clustering by Gaussian Mixture model is also possible. For instance, the three blobs pictured by black crosses in Fig. 1a can readily be partitioned thanks to a GMM with 3 components whose covariances are represented by red ellipses. Each point is associated to one Gaussian only and the three clusters are retrieved.



(a) 3-component model. (b) 2-component model. (c) 19-component model.

Fig. 1: 2-D data sets and associated Gaussian Mixture models.

However, this kind of clustering fails in cases where the different data groups are not spherical. As a case of point, when applied to two concentric circles of

different radii (see Fig. 1b), the method completely misses the data distribution and influences the subsequent partitioning in the wrong direction.

Several remarks have to be done at this point. First, this issue occurs in this case because the true clusters (the two rings) share the same mean. Moreover, the number of components is chosen equal to the number of clusters (2 here). It implies that the number of groups is available prior to the execution, which is unrealistic in most practical situations. It also shows that setting the number of components equal to the number of clusters may be inappropriate.

## 3 Gaussian Spectral Clustering

In this section, we propose a parameter-free Gaussian Mixture based clustering method: Gaussian Spectral Clustering (GSC). The idea is twofold: first, the input data is modeled by a GMM, whose number of components exceeds the actual number of clusters. Then, those Gaussians are merged in a smart manner, thanks to Spectral Clustering [13], to discover the real clusters.

### 3.1 Assess Mixture Model Quality

As pointed out in Subsection 2.4, the choice of the number of components of a mixture model is critical and is often not related to the number of distinct data groups. We propose an exhaustive search for the optimal data modeling: more precisely, we try several models on the input data (or equivalently, several number of components) and select the one that fits the data the best. We remind that the number of components may mismatch the true number of clusters.

A question which rightfully arises next is the choice of the criterion that allows model selection. Many techniques have been developed through the years to address this issue [12]. We propose to use the Bayesian Information Criterion (BIC) [19] because it does not underestimate (asymptotically) the number of true components [11] and the resulting density estimate is consistent with the groundtruth [10,15]. It is defined as:

$$BIC = t \ln N - 2 \ln \mathcal{L}, \tag{7}$$

where $t$ is the number of parameters to be estimated in the model, $N$ the number of data points and $\mathcal{L}$ the likelihood function associed to the model $\mathcal{M}_K$.

The model leading to the lowest BIC value is assumed to be optimal, as it perfectly fits the original data. In particular, in the situation shown in Fig. 1b-1c, the minimum is reached for 19 components and such a model adapts better to the data. According to definition (7), one can understand that this criterion is a trade-off between how good the model fits the data, represented by the likelihood function $\mathcal{L}$, and its complexity, embodied in $t$ (the number of parameters).

### 3.2 Determine Gaussians Similarity

The main consequence of determining the optimal GM model through the minimization of the BIC value lies in the fact that the final number of components

may not be indicative of the actual number of clusters present in the data. Due to the fact that a normal distribution is only able to accurately model an elliptical data group, one can see a Gaussian Mixture model as an approximation of a data set distribution by Gaussians. It follows that the number of components of the optimal mixture model is necessarily greater than or equal to the true number of clusters, as non-elliptical data clusters are decomposed into several Gaussians. It is also clear that normal distributions which belong to the same cluster show similarities among themselves, contrary to those from distant clusters.

Consider for example the 19-component GMM (see Fig. 1c) of the two concentric circles depicted in Fig. 1. This model fits the data better than the 2-component one, although the data is composed of 2 clusters. Moreover, the Gaussians which belong to the outer ring are similar to each other, pair by pair, but differ from the ones of the inner ring, and vice-versa. Merging them into one only set leads to a better data partitioning.

In order to measure similarity between Gaussians, we introduce in this subsection the Bhattacharyya distance and coefficient [3]. The Bhattacharyya distance $d_B$ and coefficient $c_B$ between two multivariate normal distributions $p \sim \mathcal{N}(\boldsymbol{\mu_p}, \boldsymbol{\Sigma_p})$ and $q \sim \mathcal{N}(\boldsymbol{\mu_q}, \boldsymbol{\Sigma_q})$ are defined as:

$$d_B(p,q) = \frac{1}{8}(\boldsymbol{\mu_p} - \boldsymbol{\mu_q})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu_p} - \boldsymbol{\mu_q}) + \frac{1}{2}\ln\frac{|\boldsymbol{\Sigma}|}{\sqrt{|\boldsymbol{\Sigma_p}||\boldsymbol{\Sigma_q}|}}, \tag{8}$$

$$c_B(p,q) = \exp\Big(-d_B(p,q)\Big), \tag{9}$$

with $\boldsymbol{\Sigma} = (\boldsymbol{\Sigma_p} + \boldsymbol{\Sigma_q})/2$ and $|.|$ the determinant of a (square) matrix.

A geometrical interpretation is to be drawn from this coefficient: it actually approximates the overlap ratio between two statistical distributions (normal in our case). It approaches 1 when the two compared distributions are quasi-identical and tends towards 0 in the case of two dissimilar ones.

Let us now suppose that, from a specific data set, an optimal Gaussian Mixture model with $C$ components $\mathcal{N}_1, \ldots, \mathcal{N}_C$ is determined, as explained in Subsection 3.1. Consequently, we can build the similarity matrix $\boldsymbol{S} = (S_{ij})_{i=1\ldots C, j=1\ldots C}$ whose elements are equal to the pairwise Bhattacharyya coefficients. Precisely:

$$S_{ij} = \begin{cases} c_B(\mathcal{N}_i, \mathcal{N}_j) & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases} \quad \forall (i,j) \in [\![1; C]\!]^2. \tag{10}$$

This gives a matrix of correspondences, where each coefficient reflects the similarity between the two Gaussians involved. One can also highlight that this matrix indeed is symmetric. The next subsection details the method used to decide when Gaussians are overlapping enough to be considered as belonging to the same cluster and to be merged.

### 3.3 Apply Spectral Clustering

The similarity matrix defined in the previous subsection (Eq. (10)) embeds the similarity between each pair of Gaussians.

In the perspective of clustering, the idea that comes next consists in partitioning this matrix so that sets of significant overlapping normal distributions are discovered. In other words, we want to determine clusters among which a "path" from one Gaussian to another is readily available, either directly or through other Gaussians from the same cluster.

This objective can be achieved by various means, we propose in this paper to make use of the spectral clustering approach proposed in [13]. Assuming the number $N_c$ of clusters to be discovered, five steps have to be executed.

1. Normalize $\boldsymbol{S}$ by $\boldsymbol{L} = \boldsymbol{D}^{-1/2}\boldsymbol{S}\boldsymbol{D}^{-1/2}$ where $\boldsymbol{D}$ is the row-wise sum of $\boldsymbol{S}$ (its non-diagonal elements equal 0).
2. Obtain $\boldsymbol{V} \in \mathbb{R}^{C\times C}$ the eigenvectors of $\boldsymbol{L}$. We assume they are sorted in descending order of the eigenvalues.
3. Crop $\boldsymbol{V}$ as to keep the $N_c$ largest eigenvectors: $\boldsymbol{X}$ is then of size $C \times N_c$.
4. Obtain $\boldsymbol{Y}$ by normalizing the rows of $\boldsymbol{X}$. In the end, the norm of each row of $\boldsymbol{Y}$ is equal to 1.
5. Apply K-means algorithm [14] to $\boldsymbol{Y}$, considering each row as a data point.

From this point, the rows of $\boldsymbol{Y}$, which correspond to the normal distributions of the Gaussian Mixture model, are partitionned into $N_c$ clusters. Consequently, the label of each GMM component is modified according to the K-means clustering output. Each original data point label is affected the same way.

The central disadvantage of spectral clustering is the need to specify the number of clusters. It implies an interaction with the user and the prior knowledge of how many groups are hidden in the data: this last piece of information is surrealist in most practical applications, especially in higher dimensions.

To overcome this issue, many ideas have been proposed. We propose to iterate over steps 3–5 from the spectral clustering algorithm: the idea is to provide an exhaustive search for the number of clusters, from 1 to $C$. At each iteration, the distortion of the K-means output (*i.e.* the sum of squared distances from each data point to the centroid of its cluster) is computed: the number of clusters yielding the lowest distortion value is assumed to be the actual number of groups.

At this point, the authors would like to highlight two elements from the exhaustive search step. First of all, the idea seems to show similarities with internal clustering validation measures (as a reminder, such a metric aims at comparing two partitionning of the same data set, possibly computed by two different algorithms, without any ground-truth). However, in our case, the data is evolving at each iteration, the dimensionality is increasing as well. Secondly, as was said earlier, it is almost impossible in real life scenarios to have a guess about the number of clusters. Nonetheless, a range of possible values seems more reasonable. This algorithm allows the inclusion of such prior knowledge which leads to a process acceleration.

### 3.4 Summary

We summarize in this subsection the proposed method. Given as input the data set $\{\boldsymbol{x_i} \in \mathbb{R}^n, i = 1 \dots N\}$, our algorithm is made of the three following steps:

1. Determine several Gaussian Mixture models of the data with an increasing number of components. Keep the model yielding the lowest BIC.
2. Compute the similarity matrix $\boldsymbol{S}$ (Bhattacharyya coefficient between each pair of Gaussians).
3. Apply spectral clustering on $\boldsymbol{S}$ for different number of clusters. Keep the value which leads to the lowest K-means distortion.

The proposal is fully non-parametric: it is however possible to add, if available, constraints about the number of components in the mixture model or about the number of data clusters.

## 4 Experiments

This section is devoted to the comparison of our approach with other methods. Subsection 4.2 is divided in three groups. First, we compare GSC with GMM, as we want to show that our method outperforms GMM. Secondly, as most techniques need as input the number of clusters, we devise an extension of our method in this direction and compare the performance of GSC in both situations. Finally, we compare our method against well-known clustering algorithms.

### 4.1 Databases and Assessment Metric

To evaluate our method, we use more than 100 datasets from the `Clustering benchmark`[1]. We then compare our predicted partitioning with the available ground-truth. More precisely, we use the Fowlkes-Mallows (FM) score [7]:

$$FM = \sqrt{\frac{TP}{TP+FP} \times \frac{TP}{TP+FN}}, \tag{11}$$

where TP is the number of true positives, FP the number of false positives and FN the number of false negatives. It equals 1 in the case of a predicted clustering similar to the ground-truth, 0 otherwise.

For the following experiments, the range of the number of components in which to seek the optimal GMM is fixed to $[1; 75]$. Moreover, the SC stage of our algorithm tries all numbers of clusters from 1 to $C$, where $C$ is the number of components of the optimal mixture model determined in the first stage.

### 4.2 Results

**Comparison against GM clustering** As a preliminary, Fig. 2 displays a visual comparison between the proposed algorithm and the classical Gaussian Mixture (GM) approach on a few bidimensional synthetic datasets. One can observe identical results for the second and fourth columns. However, our method (top row) outperforms GM clustering (bottom row) in the other two cases: the latter is indeed unable to retrieve non-elliptical data clusters. We also highlight that, unlike GM clustering for which the number of clusters has to be specified by the user, GSC accurately discovers the right number of groups autonomously.

[1] https://github.com/deric/clustering-benchmark

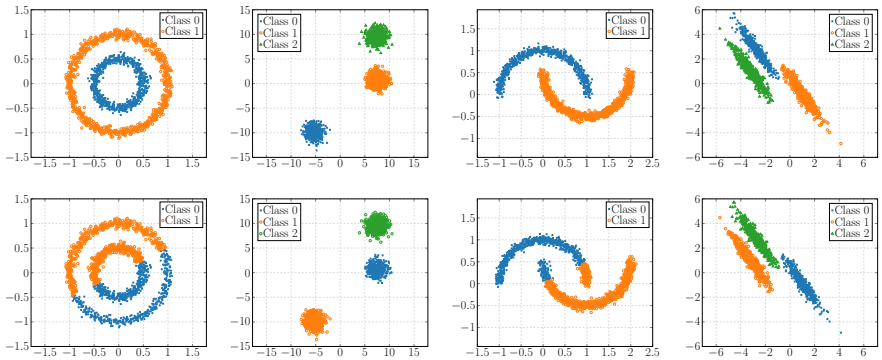



Fig. 2: Comparison of our approach (top row) with the classical GM clustering (bottom row) on 4 datasets. Each marker/color represents a different class.

**Comparison of two variants of GSC** We compare in this section the results given by the proposed approach in two situations: (1) when the number of clusters is given as input, denoted as GSC_gt, and (2) when it is totally parameter-free, denoted as GSC_free. For this purpose, we make use of the histogram of the Fowlkes-Mallows scores computed over all the datasets. Fig. 3 depicts the superposition of both histograms and both means (dashed lines).

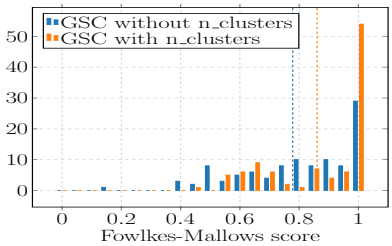



Fig. 3: Histograms of GSC_gt (in orange) and GSC_free (in blue).

As expected, GSC_gt outperforms GSC_free. Actually, the mean over all the datasets obtained by GSC_gt is 0.8604 while the one for GSC_free equals 0.7783. This difference stems from the K-Means iterations within Spectral Clustering (step 3 of Subsection 3.4) and can be explained according to three factors:

1. K-Means is applied on a small number of data points, namely the number of components in the optimal Gaussian Mixture Model.
2. The distortion only takes into account the distances between the points and their associated centroid. The number of clusters and the dimensionality are not included in this computation.
3. The limit case where K-Means algorithm is launched with the same number of clusters than the number of data points leads to a zero distortion, it is then considered the best clustering for any data set.

**Comparison against other clustering algorithms** In order to provide a fair comparison, we put ourselves in conditions where the number of clusters is

Table 1: Fowlkes-Mallows score of our method and four others on 23 datasets. The number of GMM components determined by our method is displayed in the first column. The average score for each datasets is reported in the last row.

| Dataset | | GSC (ours) | DBSCAN | GM | Kmeans | SC |
|---|---|---|---|---|---|---|
| 2d-20c-no0 | (22) | 0.987 | 0.9705 | 0.9479 | 0.9658 | **0.9916** |
| 2d-3c-no123 | (4) | **0.9814** | 0.8897 | 0.937 | 0.8206 | 0.9405 |
| 2d-4c-no4 | (6) | **0.9999** | 0.9625 | 0.8985 | 0.9819 | 0.9922 |
| 2d-4c-no9 | (5) | **1** | 0.8961 | 0.9915 | 0.9241 | 0.9752 |
| 2d-4c | (4) | **1** | **1** | **1** | **1** | **1** |
| curves1 | (14) | **1** | **1** | 0.499 | 0.499 | **1** |
| curves2 | (13) | **0.9065** | 0.3763 | 0.9047 | 0.9465 | 0.952 |
| dartboard2 | (63) | 0.5698 | **0.9246** | 0.547 | 0.5465 | 0.5484 |
| donut1 | (21) | **0.9958** | 0.991 | 0.5263 | 0.5046 | 0.992 |
| elly-2d10c13s | (8) | 0.9367 | 0.707 | **0.9368** | 0.9076 | 0.929 |
| engytime | (2) | 0.6141 | 0.732 | 0.6895 | 0.736 | **0.9696** |
| pathbased | (7) | 0.6923 | **0.9797** | 0.8556 | 0.7984 | 0.9794 |
| pmf | (6) | 0.9916 | 0.9251 | 0.9759 | **0.9932** | **0.9932** |
| spherical_5_2 | (4) | **1** | **1** | 0.9891 | **1** | **1** |
| spherical_6_2 | (6) | **1** | **1** | 0.5281 | 0.5162 | **1** |
| square1 | (4) | 0.9411 | 0.4992 | 0.9411 | 0.9372 | **0.9469** |
| square2 | (4) | 0.8925 | 0.4992 | 0.8922 | 0.8943 | **0.8978** |
| tetra | (4) | 0.6499 | **0.7068** | 0.5956 | 0.574 | 0.7053 |
| twenty | (20) | 0.9999 | 0.7067 | **1** | **1** | **1** |
| twodiamonds | (6) | 0.8651 | 0.947 | 0.9315 | 0.9297 | **1** |
| wingnut | (5) | 0.9953 | 0.9421 | 0.9953 | 0.9953 | **0.9993** |
| zelnik3 | (6) | **1** | **1** | **1** | 0.6467 | **1** |
| zelnik5 | (4) | 0.8098 | **1** | 0.7889 | 0.7907 | 0.8078 |
| Average | | 0.8604 | 0.8516 | 0.7887 | 0.76 | **0.884** |

known, as it is a mandatory argument in Gaussian Mixture clustering, K-means [2] and Spectral Clustering [13]. We proceed as follows: algorithms are executed for each dataset, repeated 50 times for results stability and the mean Fowlkes-Mallows score is kept. Table 1 provides an excerpt of the results obtained over 23 datasets. The best score for each point set is represented in bold and the average of each algorithm is given in the last row of the table.

The proposed method shows better performance than classical Gaussian Mixture clustering and K-means. Thus, in the case where the right number of clusters is known and provided as input to both algorithms, Gaussian Spectral Clustering is able to cluster 9.1% better compared to GM approach. Both methods give similar results when data groups are elliptical, in other cases ours is able to discover clusters with more complex data distribution. It is also to be highlighted that GM clustering performs better than K-means as the latter is a specific case of the former for spherical clusters. In order to show the outperformance of GSC

over GM, we perform a z-test, as we know the variance and the mean of the distributions and both of them follow a gaussian. Given the means, $\mu_{GSC} = 0.860$ and $\mu_{GM} = 0.798$, and the standard deviations, $\sigma_{GSC} = 0.171$ and $\sigma_{GM} = 0.197$, of both models computed on $n = 101$ datasets, we consider the null hypothesis $\mathcal{H}_0$: $\mu_{GM} = \mu_{GSC}$ and the alternative hypothesis $\mathcal{H}_1$: $\mu_{GM} < \mu_{GSC}$. We want to show with a 99% confidence ($u_{0.01} = 2.33$) that $\mathcal{H}_1$ holds. Then,

$$z = \frac{\mu_{GM} - \mu_{GSC}}{\sqrt{\sigma_{GM}^2/n + \sigma_{GSC}^2/n}} = -2.735 < -2.33 = -u_{0.01}. \tag{12}$$

Thus, we can reject the null hypothesis $\mathcal{H}_0$.

GSC shows better performance than DBSCAN [5,18] but is outpaced by Spectral Clustering. It is however important to keep in mind that several parameters have to be set for those methods:

- In DBSCAN, the radius $\varepsilon$ of a spherical neighborhood and the number $MinPts$ of data points in order for such a neighborhood to be valid.
- In SC, the standard-deviation of the RBF kernel and the number of clusters.

A quasi-exhaustive search over all the parameters is conducted, only the greatest FM score is kept in memory. Such a process is usually unpractical in most real-world clustering applications. We remind that our method requires, at this point, only the number of clusters and is automatic except from this parameter.

In summary, our model gives competitive results in most datasets but three, namely DARTBOARD2, ENGYTIME and TETRA. Those bad scores are mainly caused by inaccurate GM models due to strongly overlapping clusters or data groups with too few points.

## 5 Conclusion and Prospects

We propose in this article an improvement of GM clustering. Our method combines a modelling of a specific data set by a mixture of weighted normal distributions. Then, similar Gaussians are merged using the SC algorithm. Our model, named Gaussian Spectral Clustering (GSC), is able to retrieve clusters with complex shape, contrary to GM partitioning which is limited to elliptical data groups. Constraints on the number of components, the number of clusters or both, can be applied in order to speed GSC up and obtain realistic results regarding to specific applications. A non-parametric algorithm is also derived.

Several leads for improvement will be considered for future work. First, the selection of the optimal GMM, *i.e.* trying all models within a range of possible ones, is naive and expensive. Moreover, the random initialization of the EM algorithm makes it difficult to obtain stable results. Finally, since the estimation of the number of clusters in the SC step is prone to errors, an auto-determination technique or another evaluation criterion for K-Means may be used.

## References


1. Akaike, H.: A new look at the statistical model identification. IEEE Transactions on Automatic Control **19**(6), 716–723 (Dec 1974)
2. Arthur, D., Vassilvitskii, S.: K-means++: the advantages of careful seeding. In: ACM-SIAM symposium on discrete algorithms (Jan 2007)
3. Bhattacharyya, A.: On a measure of divergence between two statistical populations defined by their probability distributions. Bulletin of the Calcutta Mathematical Society pp. 99–109 (1943)
4. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. J. Royal Stat. Soc. **39**(1), 1–38 (1977)
5. Ester, M., Hans-Peter, K., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: International conference on knowledge discovery and data mining. pp. 226–231 (Dec 1997)
6. Figueiredo, M., Jain, A.: Unsupervised learning of finite mixture models. IEEE Trans. on Pattern Anal. and Mach. Intell. **24**(3), 381 – 396 (Mar 2002)
7. Fowlkes, E.B., Mallows, C.L.: A method for comparing two hierarchical clusterings. Journal of the American Statistical Association **78**(383), 553–569 (Sep 1983)
8. Friedman, J., Hastie, T., Tibshirani, R.: Sparse inverse covariance estimation with the graphical lasso. Biostatistics **9**(3), 432–441 (Jul 2008)
9. Ghosal, A., Nandy, A., Das, A.K., Goswami, S., Panday, M.: A short review on different clustering techniques and their applications. In: Emerging Technology in Modelling and Graphics, pp. 69–83. Springer (2020)
10. Keribin, C.: Consistent estimation of the order of mixture models. Sankhyā: The Indian Journal of Statistics, Series A pp. 49–66 (2000)
11. Leroux, B.G.: Consistent estimation of a mixing distribution. The Annals of Statistics pp. 1350–1360 (1992)
12. McLachlan, G.J., Rathnayake, S.: On the number of components in a gaussian mixture model. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **4**(5), 341–355 (2014)
13. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: analysis and an algorithm. In: Advances in Neural Information Processing Systems. pp. 849–856 (2001)
14. Pelleg, D., Moore, A.: X-means: Extending k-means with efficient estimation of the number of clusters. In: Int. Conf. on Machine Learning. pp. 727–734 (2000)
15. Roeder, K., Wasserman, L.: Practical bayesian density estimation using mixtures of normals. Journal of the American Statistical Association **92**(439), 894–902 (1997)
16. Ruan, L., Yuan, M., Zou, H.: Regularized parameter estimation in high-dimensional gaussian mixture models. Neural Computation **23**(6), 1605–1622 (Mar 2011)
17. Saxena, A., Prasad, M., Gupta, A., Bharill, N., Patel, O.P., Tiwari, A., Er, M.J., Ding, W., Lin, C.T.: A review of clustering techniques and developments. Neurocomputing **267**, 664–681 (2017)
18. Schubert, E., Sander, J., Ester, M., Kriegel, H.P., Xu, X.: Dbscan revisited, revisited: Why and how you should (still) use dbscan. ACM Transactions on Database Systems **42**(3), 1–21 (Jul 2017)
19. Schwarz, G.: Estimating the dimension of a model. Ann. Stat. **6**(2), 461–464 (1978)
20. Wallace, C.S.: Statistical and Inductive Inference by Minimum Message Length. Springer-Verlag New York (2005)
21. Zhang, Z., Chen, C., Sun, J., Chan, K.L.: EM algorithms for gaussian mixtures with split-and-merge operation. Pattern recognition **36**(9), 1973–1983 (2003)