Bin-EM-CEM algorithms of general parsimonious Gaussian mixture models for binned data clustering

Abstract—Data binning is a well-known data pre-processing technique in statistics. It was applied to model-based clustering approaches to reduce the number of data and facilitate the processing. EM and CEM algorithms are commonly used in model-based approaches. Thus EM and CEM algorithms applied to binned data were developed: binned-EM algorithm for mixture approach, and bin-EM-CEM algorithm for classification approach. At another side, fourteen parsimonious Gaussian mixture models for EM and CEM algorithms were proposed by considering a parametrization of the variance matrices of the clusters. Due to different characteristics of each model, fourteen models can adapt to data of different structures so as to simplify the clustering process. The experimental results of EM algorithms of fourteen parsimonious models also show that the model which fits the data gives a better result than the other models. Previously, binned-EM algorithms of fourteen parsimonious Gaussian mixture models were developed. The result shows to be of interest to combine the advantages of binned data and parsimonious models on model-based clustering approaches. So in this paper, we develop bin-EM-CEM algorithms of the eight most general parsimonious Gaussian mixture models. The performances of the developed algorithms applied to different models of data are studied and analyzed.

I. INTRODUCTION

In data analysis, clustering is the process of exploring the structure of data without any information. Among many clustering approaches, basing on certain possibility models becomes a common approach. Mixture approach and classification approach are two most well-known model-based clustering approaches. The comparison between two approaches was analyzed in the paper of Celeux and Govaert [1]. The mixture approach maximizes the likelihood by Expectation Maximization (EM) algorithm over the mixture parameters while the classification approach maximizes the likelihood by Classification Expectation Maximization (CEM) algorithm over the mixture parameters and over the identifying labels of the mixture component origin for each data. CEM algorithm can be considered as the classification version of the EM algorithm.

With the development of technology, the capacity of data that we can obtain gets bigger and bigger. As a consequence, the EM and CEM algorithms become very slow. In order to solve this problem, binned data were introduced into clustering subject. It has played a major role in reducing the number of data and facilitate the process. Moreover, binned data can exist systematically when the measuring has limited precision.

Thus, EM and CEM algorithms applied to binned data were proposed: binned-EM algorithm [2], [3], [4] and bin-EM-CEM algorithm [5], [6]. The idea of obtaining binned data is to divide the space into sub-spaces which are also called bins. The frequency of the bin is the only information of the data. It indicates the amount of data within each bin. Examples of obtaining binned data were demonstrated in some previous papers [7], [8], [9]. The experiment result shows that bin-EM-CEM algorithm executes faster than CEM algorithm when the number of data is enough big [5].

Since in classification approach, the identifying labels of the mixture components are considered as parameters, thus the number of parameters increases indefinitely with the size of data. Simplifying the model complexity in order to reduce the number of parameters can be efficient in solving this problem. A parsimonious model is a model with less parameters, which is less complex. According to Banfield and Raftery [10], we can consider a parametrization of the variance matrix of the clusters: \( \Sigma_k = \lambda_k D_k A_k D_k^T \), where \( \lambda_k = |\Sigma_k|^{1/d} \) determining the volume of the \( k \)th cluster, \( D_k \) which is the matrix of eigenvectors of \( \Sigma_k \) determining its orientation, and its shape determined by a diagonal matrix \( A_k \) with the normalized eigenvalues of \( \Sigma_k \) in a decreasing order on the diagonal, and \( |A_k|=1 \). By supposing one, some of these parameters to vary among the clusters, eight general models were proposed: \( [\lambda D A D]^T \), \( [\lambda_k D A D]^T \), \( [\lambda D A D]^T \), \( [\lambda_k D A D]^T \), \( [\lambda D A D]^T \), \( [\lambda_k D A D]^T \), \( [\lambda D A D]^T \), \( [\lambda_k D A D]^T \) [11].

To combine the advantages of binning data and parsimonious models on EM algorithm, binned-EM algorithms of fourteen Gaussian parsimonious models were developed [7], [8], [9]. The experimental results show that the execution becomes faster by introducing binned data into EM algorithm and the best result is obtained by the model which has the same structure as the data. In this paper, we will develop bin-EM-CEM algorithms of general parsimonious Gaussian mixture models.

This paper is organized as follows. Section II presents the essential notations, the background and the derivation of eight bin-EM-CEM algorithms of general parsimonious models. In Section III, numerical experiments of bin-EM-CEM algorithms of general parsimonious Gaussian mixture models on simulated data are demonstrated. At last, the paper is ended by a concluding summary in Section IV.
II. NOTATIONS AND BIN-EM-CEM ALGORITHMS OF GENERAL MODELS

We assume that \(x = (x_1, \ldots, x_n)\) is an independent sample issued from a \(K\)-component mixture distribution defined on \(\mathbb{R}^p\):
\[
    f(x; \Phi) = \sum_{k=1}^{K} \pi_k f_k(x; \theta_k)
\]
with \(\Phi = (\pi_1, \ldots, \pi_K, \theta_1, \ldots, \theta_K)\), where \(\pi_k\) \((k = 1, \ldots, K)\) denote the mixing proportions of the mixtures \((0 < \pi_i < 1\) and \(\sum_{k=1}^{K} \pi_k = 1)\), and \(\theta_k = (\mu_k, \Sigma_k)\) \((k = 1, \ldots, K)\) are the parameters of Gaussian distribution functions \(f_k\) of components: mean vectors \(\mu_k\) and variance matrices \(\Sigma_k\). Vector \(z = (z_1, \ldots, z_n)\) is the class label of \(x\), where \(z_i = 1, \ldots, K\) for \(i = 1, \ldots, n\). \(z_i = k\) when \(x_i\) comes from the \(k\)-th component.

The whole sample space \(\mathbb{R}^p\) is divided into \(v\) bins with a partition \((\mathcal{H}_1, \ldots, \mathcal{H}_v)\) and we assume that the only observed information is a set of frequencies \(n_r\) \((r = 1, \ldots, v)\) where each frequency \(n_r\) indicates the number of \(x_i\) belonging to the bin \(\mathcal{H}_r\). The set of frequencies is denoted by vector \(a = (n_1, \ldots, n_v)\), with \(\sum_{r=1}^{v} n_r = n\).

The probability that \(x\) belongs to bin \(\mathcal{H}_r\) is denoted by:
\[
    p_r(\Phi) = P(x \in \mathcal{H}_r | \Phi) = \sum_{k=1}^{K} \pi_k \int_{\mathcal{H}_r} f_k(x; \theta_k)dx
\]
and the probability that \(x\) belonging to the bin \(\mathcal{H}_r\) comes from component \(k\) of the mixture is denoted by:
\[
    p_{k|r}(\Phi) = \frac{\pi_k \int_{\mathcal{H}_r} f_k(x; \theta_k)dx}{p_r(\Phi)}
\]
The only observed information vector \(a\) follows a multinomial distribution
\[
p(a, \Phi) = c \prod_{r=1}^{v} (p_r(\Phi))^{n_r}
\]
where \(c = n! / \prod_{r=1}^{v} n_r!\).

According to the space division with a partition \((\mathcal{H}_1, \ldots, \mathcal{H}_v)\), the complete information of the data can be denoted as \((x, z) = ((x_{r_1, z_1}), \ldots, (x_{r_n, z_n}))\) for \(r = 1, \ldots, v\), where \(x_{r,s}\) points to the \(s\)-th data in bin \(\mathcal{H}_r\). The density of each point \(x_{r,s}\) is \(\pi_{z_{r,s}} f_{z_{r,s}}(x_{r,s}; \theta_{z_{r,s}})/p_r(\Phi)\) and the complete data probability function is
\[
p(a, z; \Phi) = c \prod_{r=1}^{v} \prod_{s=1}^{n_r} \pi_{z_{r,s}} f_{z_{r,s}}(x_{r,s}; \theta_{z_{r,s}})
\]
The complete log-likelihood is:
\[
    L(\Phi; a, z) = \sum_{k=1}^{K} \sum_{r=1}^{v} \sum_{s=1}^{n_r} z_{k,s} \log \pi_k f_k(x_{r,s}; \theta_k) + \log c
\]
where \(z_{k,s} = 1\) if \(z_{r,s} = k\) and 0 otherwise.

Since there is no information about the exact location of the data within each bin, we assume that all the data comes from the same component within each bin. So \(p_r(\Phi)\) can be expressed as:
\[
p_r(\Phi) = \pi_k \int_{\mathcal{H}_r} f_z(x; \theta_z)dx
\]
Then we have the joint density function as follows:
\[
p(a, z; \Phi) = c \prod_{r=1}^{v} (p_r(\Phi))^{n_r}
\]
and the log-likelihood can be expressed as:
\[
    L(\Phi; a, z) = \sum_{r=1}^{v} n_r \log(p_r(\Phi)) + \log c
\]

The bin-EM-CEM algorithm aiming to maximize \(L(\Phi; a, z)\) starts from a random initialization \(\Phi^{(0)}\), and follows two steps iteratively until it converges.

Step 1 (Expectation and Classification): calculate
\[
    z^{(q+1)} = \arg \max_{z} L(\Phi^{(q)}; a, z)
\]
we have
\[
    z^{(q+1)} = \arg \max_{1 \leq s \leq K} \sum_{r=1}^{v} n_r \log(p_r(\Phi^{(q)})) = \arg \max_{1 \leq s \leq K} \sum_{r=1}^{v} n_r \log(c_{k|r}(\Phi^{(q)}))
\]
This step can be divided into step E (Expectation) and step C (Classification) [12]:

At the E step, we calculate \(p_{k|r}^{(q)}\) for all \(k, r\);

At the C step, we obtain the partition \(z^{(q+1)}\) by maximizing \(p_{k|r}^{(q)}\) \(z^{(q+1)} = \arg \max_{k} p_{k|r}^{(q)}\), which means for each \(r\), we replace the biggest \(p_{k|r}^{(q)}\) by 1, and 0 for the others.

Step 2 (Maximization): calculate
\[
    \Phi^{(q+1)} = \arg \max_{\Phi} L(\Phi; a, z^{(q+1)})
\]
Because it is not easy to maximize \(L(\Phi; a, z)\) directly, thus we apply an internal EM algorithm to obtain this maximization. As the EM algorithm, we maximize the expectation of the complete log-likelihood instead of log-likelihood:
\[
    Q(\Phi, \Phi^{(q)}) = E(L(\Phi; a, z)|a, z^{(q+1)}; \Phi^{(q)})
\]
\[
    = E(\sum_{k=1}^{K} \sum_{r=1}^{v} \sum_{s=1}^{n_r} z_{k,s} \log \pi_k f_k(x_{r,s}; \theta_k) + \log c | z^{(q+1)}; \Phi^{(q)})
\]
In the cycle of the inner EM algorithm, let’s denote:
\[
    \Phi^{*} = E(L(\Phi; a, z)|a, z^{(q+1)}; \Phi^{(q)})
\]
where
\[
    \Phi^{*} = \Phi^{(q)}
\]
Then we have:
\[
    Q(\Phi, \Phi^{*}) = \sum_{k=1}^{K} \sum_{r=1}^{v} \sum_{s=1}^{n_r} z_{k,s} \ln(\pi_k) + \ln \left( \int_{\mathcal{H}_r} \frac{1}{f_{k}^{(q)}(x; \theta_k)} dx \right) + \ln c
\]
(1)
Maximizing Equation (1) equals maximizing:

\[
A = \sum_{k=1}^{K} \sum_{r=1}^{v} z_{kr} n_r \int_{\mathcal{H}_r} f_k(x; \theta_k^r) dx
\]

\[
\cdot \left( \int_{\mathcal{H}_r} \ln(f_k(x; \theta_k^r)) f_k(x; \theta_k^r) dx \right)
\]

\[
= \sum_{k=1}^{K} \sum_{r=1}^{v} z_{kr} n_r \int_{\mathcal{H}_r} f_k(x; \theta_k^r) dx
\]

\[
\cdot \left( \int_{\mathcal{H}_r} (\ln(2\pi)^{n/2} - \ln|\Sigma_k|^{1/2}) \right)
\]

\[
- \frac{1}{2} \left( \right) \Sigma_k^{-1} \right) dx
\]

And finally it leads to the minimization of

\[
B = \sum_{k=1}^{K} \sum_{r=1}^{v} z_{kr} n_r \int_{\mathcal{H}_r} (x - \mu_k^*)' \Sigma_k^{-1} (x - \mu_k^*) f_k(x; \theta_k^r) dx
\]

\[
= \sum_{k=1}^{K} \sum_{r=1}^{v} z_{kr} n_r \ln |\Sigma_k|
\]

where

\[
G_k^{**} = \sum_{r=1}^{v} z_{kr} n_r \int_{\mathcal{H}_r} (x - \mu_k^*)' (x - \mu_k^*) f_k(x; \theta_k^r) dx
\]

and

\[
p_{r/k}^* = \int_{\mathcal{H}_r} f_k(x; \theta_k^r) dx
\]

We get

\[
\pi_k^* = \sum_{r=1}^{v} n_r z_{kr}^{(q)} / n
\]

and

\[
\mu_k^* = \frac{\sum_{r=1}^{v} n_r z_{kr}^{(q)} f_k(x; \theta_k^r) dx}{\sum_{r=1}^{v} n_r z_{kr}^{(q)}}
\]

where

\[
\Phi^{**} = \Phi^{(q+1)}
\]

The results of \(\Sigma_k^{**}\) are different depending on the parsimonious models.

To obtain parsimonious Gaussian mixture models, a way of decomposition of the variance matrices \(\Sigma_k\) was mentioned in the introduction of this paper: \(\Sigma_k = \lambda_k A_k D_k D_k^T\).

By allowing variation of these three parameters among clusters, eight different general models can be developed: \([\lambda A D A^T]\), \([\lambda A D A^T]\), \([\lambda A D A_k D^T]\), \([\lambda A D A_k D^T]\), \([\lambda A D_k A D^T]\), \([\lambda A D_k A D^T]\), \([\lambda A_k D_k A D^T]\), \([\lambda A_k D_k A D^T]\).

Model \([\lambda A D A^T]\). For this simplest model, maximizing equation (1) equals the minimization of

\[
M_1(\Sigma) = \sum_{k=1}^{K} \text{tr}(\Sigma^{-1} G_k^{**}) + \sum_{k=1}^{K} \sum_{r=1}^{v} z_{kr}^{(q)} n_r \ln \Sigma
\]

variance matrix \(\Sigma\) is estimated by

\[
\Sigma^{**} = \frac{\sum_{k=1}^{K} G_k^{**}}{\sum_{k=1}^{K} \sum_{r=1}^{v} z_{kr}^{(q)} n_r}
\]

Model \([\lambda_k A D A_k D^T]\). In this situation, we put \(\Sigma_k = \lambda_k C\) with \(C = D A D^T\). Maximizing equation (1) equals the minimization of

\[
M_2(\lambda_k, C) = \sum_{k=1}^{K} \frac{1}{\lambda_k} \text{tr}(C^{-1} G_k^{**}) + d \sum_{k=1}^{K} \sum_{r=1}^{v} z_{kr}^{(q)} n_r \ln |\lambda_k|
\]

To find \(\lambda_k^{**}\) and \(C\), an iteration has to be performed:

- Keep \(C\) fixed, the \(\lambda_k^{**}\) are

\[
\lambda_k^{**} = \frac{\text{tr}(G_k^{**} C^{-1})}{d \sum_{r=1}^{v} z_{kr}^{(q)} n_r}
\]

- keep \(\lambda_k^{**}\) fixed, the matrix \(C\) is

\[
C^{**} = \frac{\sum_{k=1}^{K} \frac{1}{\lambda_k} G_k^{**}}{\left| \sum_{k=1}^{K} \frac{1}{\lambda_k} G_k^{**} \right|^{1/d}}
\]

Model \([\lambda_k D A_k D_k^T]\). Maximizing equation (1) leads to the minimization of

\[
M_3(\lambda, D, A_k) = \frac{1}{\lambda} \sum_{k=1}^{K} \text{tr}(G_k^{**} D A_k^{-1} D^T)
\]

\[
+ d \sum_{k=1}^{K} \sum_{r=1}^{v} z_{kr}^{(q)} n_r \ln \lambda
\]

To minimize \(M_3\) is to calculate \(\lambda\) and minimize \(\sum_{k=1}^{K} \text{tr}(G_k^{**} D A_k^{-1} D^T)\) using an iterative method as following. First step for \(\lambda\)

\[
\lambda^{**} = \frac{\sum_{k=1}^{K} \text{tr}(G_k^{**} D A_k^{-1} D^T)}{d \sum_{k=1}^{K} \sum_{r=1}^{v} z_{kr}^{(q)} n_r}
\]

The second step is to minimize \(\sum_{k=1}^{K} \text{tr}(G_k^{**} D A_k^{-1} D^T)\):

- Keeping \(D\) fixed, from Corollary A.1 of the appendix in the paper of Celeux and Govaert [11], we get

\[
A_k^{**} = \frac{\text{diag}(D^T G_k^{**} D)}{|\text{diag}(D^T G_k^{**} D)|^{1/d}}
\]

- Keeping \(A_k^{(q+1)}\), ..., \(A_k^{(q+1)}\) fixed, we adapt an algorithm of Flury [13] aiming to minimize \(f(D) = \sum_{k=1}^{K} \pi_k^{(q)} \text{tr}(G_k^{(q+1)} D A_k^{-1} D^T)\): First initial a solution \(D = (d_1, ..., d_d)\). For any couple \((l, m) | (l \neq m) \in \{1, ..., d\}\), we find a corresponding couple \((\delta_l, \delta_m)\) which are orthogonal vectors, linear combination of \(d_l\) and \(d_m\), minimizing the criterion \(f(D)\). We have

\[
\sum_{k=1}^{K} \text{tr}(D \pi_k^{(q)} A_k^{-1} D^T G_k^{(q+1)}) = \sum_{k=1}^{K} \sum_{j=1}^{d} d_j^T G_k^{(q+1)} \pi_k^{(q)} d_j
\]
Thus, it equals to find \((\delta_l, \delta_m)\) minimizing \(S(d_l, d_m)\). We can write

\[ \delta_l = (d_l, d_m) q_1 \]

\[ \delta_m = (d_l, d_m) q_2 \]

where \(q_1\) and \(q_2\) are two orthogonal vectors of \(R^2\). We have

\[ S(\delta_l, \delta_m) = \sum_{k=1}^{K} \frac{q_1^T (d_l, d_m)^T G_k^{(q+1)} \pi_k (d_l, d_m) q_1}{a_k} + \sum_{k=1}^{K} \frac{q_2^T (d_l, d_m)^T G_k^{(q+1)} \pi_k (d_l, d_m) q_2}{a_k^m} \]

\[ = \sum_{k=1}^{K} q_1^T Z_k q_1 + \sum_{k=1}^{K} q_2^T Z_k q_2 \]

where

\[ Z_k = (d_l, d_m)^T G_k^{(q+1)} \pi_k (d_l, d_m) \]

Denoting \(Q = (q_1, q_2)\), we get

\[ q_1^T Z_k q_1 + q_2^T Z_k q_2 = tr(Q^T Z_k Q) = tr(Z_k) \]

And the problem reduces to the optimization of

\[ S(d_l, d_m) = \sum_{k=1}^{K} \frac{q_1^T Z_k q_1}{a_k} + \sum_{k=1}^{K} \frac{tr(Z_k - q_1^T Z_k q_1)}{a_k^m} \]

which is equivalent to the minimization of

\[ q_1^T \left( \sum_{k=1}^{K} \left( \frac{1}{a_k} - \frac{1}{a_k^m} \right) Z_k \right) q_1 \]

Hence, \(q_1\) is the second eigenvector of the matrix \(\sum_{k=1}^{K} (\frac{1}{a_k} - \frac{1}{a_k^m}) Z_k\). Repeat the procedure above until \(f(D)\) converges.

Model \([\lambda_k D_k A_k D_k^T]\). For this case, it is more convenient to write \(\Sigma_k = DA_k D_k^T\) where \(|A_k| = |\Sigma_k|\). Maximizing equation (1) equals the minimization of

\[ M_4(D, A_k) = \sum_{k=1}^{K} tr(DA_k^{-1} D_k^T G_k^{(q+1)}) + \sum_{k=1}^{K} \sum_{r=1}^{v} z_{kr} n_r \ln |A_k| \]

As previously presented, the minimization of \(M_4\) can be achieved in the similar way:

- Keeping \(D\) fixed, from Corollary A.2 of the appendix in the paper of Celeux and Govaert [11], we get

\[ A_k^{**} = diag((DG_k^{(q+1)} D_k^T)^T) \sum_{r=1}^{v} z_{kr} n_r \]

- For fixed \(A_k^{**}, \ldots, A_k^{**}\), it can be making use of the same algorithm described above since minimizing \(M_4\) is equivalent to minimize \(\sum_{k=1}^{K} tr(DA_k^{-1} D_k^T G_k^{(q+1)})\).

Model \([\lambda D_k A_k D_k^T]\). Maximizing equation (1) equals the minimization of

\[ M_5(\lambda, D_k, A) = \frac{1}{\lambda} \sum_{k=1}^{K} tr(D_k A_k^{-1} D_k^T G_k^{(q+1)}) + d \sum_{k=1}^{K} \sum_{r=1}^{v} z_{kr} n_r \ln |\lambda| \]

Considering for \(k = 1, \ldots, K\) the eigenvalue decomposition \(G_k^{(q+1)} = L_k^{q+1} \Omega_k^{q+1} L_k^{q+1 T}\) of the symmetric definite positive matrix \(G_k\) with the eigenvalues in the diagonal matrix \(\Omega_k\) in decreasing order, we have

\[ M_5(\lambda, D_k, A) = \frac{1}{\lambda} \sum_{k=1}^{K} tr(D_k L_k^{q+1} \Omega_k^{q+1} L_k^{q+1 T} D_k A_k^{-1}) \]

From Theorem 1 in Celeux and Govaert’s paper [11], we get \(D_k = L_k\), and we have

\[ M_5(\lambda, D_k, A) = \frac{1}{\lambda} \sum_{k=1}^{K} \sum_{r=1}^{v} z_{kr} n_r \ln |\lambda| \]

From which, we deduce the optimal \(A\) and \(\lambda\)

\[ A^{**} = \frac{\sum_{k=1}^{K} \Omega_k^{**}}{\sum_{k=1}^{K} \Omega_k^{**}} \]

\[ \lambda^{**} = \frac{1}{\sum_{k=1}^{K} \Omega_k^{**}} \]

Model \([\lambda_k D_k A_k D_k^T]\). Use again the eigenvalue decomposition \(G_k^{(q+1)} = L_k^{q+1} \Omega_k^{q+1} L_k^{q+1 T}\). Maximizing equation (1) leads to the minimization of

\[ M_6(\lambda_k, D_k, A) = \frac{1}{\lambda_k} \sum_{k=1}^{K} \sum_{r=1}^{v} z_{kr} n_r \ln |\lambda_k| \]

The minimization of \(M_6\) has to be achieved iteratively:

\[ A^{**} = \frac{\sum_{k=1}^{K} \lambda_k^{**}}{\sum_{k=1}^{K} \lambda_k^{**}} \]
and 
\[ \lambda^* = \frac{tr(\Omega_k^* A^{-1})}{d \sum_{r=1}^v z_{kr}^q n_r} \]

Model \([\lambda_k D_k A_k D_k^T]\). We write \(\Sigma_k = \lambda C_k\) where \(C_k = D_k A_k D_k^{-1}\). Then, maximizing equation (1) equals to minimize

\[ M_T(\lambda, C_k) = \frac{1}{\lambda} \sum_{k=1}^K tr(C_k^{-1} G_k^{**}) + \sum_{k=1}^K \sum_{r=1}^v z_{kr}^q n_r \ln |\lambda| \]

Simple calculation give us:
\[ C_k^{**} = \frac{G_k^{**}}{|G_k^{**}|^{1/d}} \]

and
\[ \lambda^{**} = \frac{\sum_{k=1}^K |G_k^{**}|^{1/d}}{\sum_{k=1}^K \sum_{r=1}^v z_{kr}^q n_r} \]

Model \([\lambda_k D_k A_k D_k^T]\). This is the most general situation. Maximizing equation (1) leads to the minimization of

\[ M_B(\Sigma_k) = \sum_{k=1}^K tr(\Sigma_k^{-1} G_k^{**}) + \sum_{k=1}^K \sum_{r=1}^v z_{kr}^q n_r \ln \Sigma_k \]

and the variance matrices \(\Sigma_k\) are estimated by
\[ \Sigma_k^{**} = \frac{G_k^{**}}{\sum_{r=1}^v z_{kr}^q n_r} \]

III. NUMERICAL EXPERIMENT

A. Experiment of bin-EM-CEM algorithms of general models on simulated data

Comparing the performances of bin-EM-CEM algorithms of different parsimonious models is a manner of learning the characteristic of each model. In this part, we will test bin-EM-CEM algorithms of eight general models to simulated data. In order to simplify the experiment and be able to demonstrate the result in a visualized space, all the simulated data are of two dimensions. According to each general model, we generate 30 samples of size= 5000. Bin-EM-CEM algorithms of eight models are applied respectively on each sample. The space is divided into square bins sized of 0.5 - 0.5. Because of different characteristic of each sample, the space is cut into different number of bins depending on different sample, which will be detailed in the data description. The average of the results of 30 samples of each model are considered as the final result to be studied. We record the accuracy and the standard error of the accuracy as the evaluation of the algorithm performance. Accuracy indicates the percentage of the data which are correctly clustered. A distance value \(\delta\) is defined to calculate the distance between two components:

\[ \delta = \sqrt{(\mu_1 - \mu_2)^T (\Sigma_1 + \Sigma_2)^{-1} (\mu_1 - \mu_2)} \]

The parameters of the simulated data is detailed as follows:

- Data type 1: according to model \([A D_1 D_2 D_3 D_4 D_5 D_6 D_7]\) with \(\lambda = 1, D = (\sqrt{2} - \sqrt{3}, \sqrt{2} - \sqrt{3}), A = Diag(3,1/3), \mu_1 = (-1.5,0), \mu_2 = (1.5,0), \delta = 3.8, \) Number of bins: 26 × 19.
- Data type 2: according to model \([\lambda_k D_1 D_2 D_3 D_4 D_5 D_6 D_7]\) with \(\lambda_1 = 1, \lambda_2 = 5, D = (\sqrt{2} - \sqrt{3}, \sqrt{2} - \sqrt{3}), A = Diag(3,1/3), \mu_1 = (-2,0), \mu_2 = (2,0), \delta = 2.9, \) Number of bins= 39 × 41.
- Data type 3: according to model \([\lambda_1 D_1 D_2 D_3 D_4 D_5 D_6 D_7]\) with \(\lambda_1 = 1, D = (\sqrt{2} - \sqrt{3}, \sqrt{2} - \sqrt{3}), A_1 = Diag(1,1), A_2 = Diag(2,1/2), \mu_1 = (-1.5,0), \mu_2 = (1.5,0), \delta = 3.0, \) Number of bins= 22 × 16.
- Data type 4: according to model \([\lambda_1 D_1 D_2 D_3 D_4 D_5 D_6 D_7]\) with \(\lambda_1 = 1, \lambda_2 = 2, D = (\sqrt{2} - \sqrt{3}, \sqrt{2} - \sqrt{3}), A_1 = Diag(1,1), A_2 = Diag(2,1/2), \mu_1 = (-1.5,0), \mu_2 = (2,0), \delta = 2.9, \) Number of bins= 24 × 24.
- Data type 5: according to model \([\lambda_1 D_1 D_2 D_3 D_4 D_5 D_6 D_7]\) with \(\lambda_1 = 1, D_1 = (\sqrt{2} - \sqrt{3}, \sqrt{2} - \sqrt{3}), D_2 = Diag(1,1), A = Diag(3,1/3), \mu_1 = (-2,1), \mu_2 = (1,0), \delta = 3.1, \) Number of bins= 27 × 16.
- Data type 6: according to model \([\lambda_1 D_1 D_2 D_3 D_4 D_5 D_6 D_7]\) with \(\lambda_1 = 1, D_1 = (\sqrt{2} - \sqrt{3}, \sqrt{2} - \sqrt{3}), D_2 = Diag(1,1), A_1 = Diag(3,1/3), A_2 = Diag(1,1), \mu_1 = (-2,0), \mu_2 = (1,0), \delta = 3.0, \) Number of bins= 22 × 18.
- Data type 7: according to model \([\lambda_1 D_1 D_2 D_3 D_4 D_5 D_6 D_7]\) with \(\lambda_1 = 3, \lambda_2 = 1, D_1 = (\sqrt{2} - \sqrt{3}, \sqrt{2} - \sqrt{3}), D_2 = Diag(1,1), A_1 = Diag(3,1/3), A_2 = Diag(1,1), \mu_1 = (-2,0), \mu_2 = (2,0), \delta = 3.0, \) Number of bins= 30 × 31.

The result turns out that the best result (highest clustering accuracy) is obtained respectively by the feature model which has exactly the same structure as the data. It is of interest to notice that even the most complex model cannot provide a higher accuracy than the most suitable model.

But the more complex model which can represent the data structure (exclude the feature model) generally have a better result than the other models. For instance, \([\lambda_1 D_1 A_k D_1^T]\), \([\lambda_1 D_1 A_k D_1^T]\), \([\lambda_1 D_1 A_k D_1^T]\) and \([\lambda_1 D_1 A_k D_1^T]\) obtain higher accuracy than the other models when applying to the data of structure \([\lambda_1 D_1 D_2 D_3 D_4 D_5 D_6 D_7]\).

So we can conclude: The best result is obtained by the right model which best fits the data. A model which is more complex than the data structure and include the situation of the data structure can receive a relatively good result. So choosing the right model is an important step in clustering.

B. Experiment of bin-EM-CEM algorithm with different size of bins on simulated data

In bin-EM-CEM algorithm, size of bin which decides the number of bins as well as the accuracy of the clustering result,
is an very important element. This experiment shows how the change of the size of bins affects the result of bin-EM-CEM algorithm.

In this experiment, the side-length of the bins varies from 0.1 to 2.8 with an interval of 0.1. For each side-length, 30 samples of size $= 5000$ are generated according to the model $\{\operatorname{AD}_k, \operatorname{AD}^T_k\}$ with $\lambda = 1$, $D_1 = (\frac{x^2}{2} - \frac{x^2}{2}; \frac{x^2}{2} - \frac{x^2}{2})$, $D_2 = \operatorname{Diag}(1, 1)$, $A = \operatorname{Diag}(3, 1/3)$, $\mu_1 = (-2, 1)$, $\mu_2 = (1, 0)$ and $\delta = 3.1$. Bin-EM-CEM algorithm of model $\{\operatorname{AD}_k, \operatorname{AD}^T_k\}$ of different bin size is applied on each sample. The CPUtime and accuracy are recorded as evaluation of the experiment. The average of 30 samples’ results is considered as the final result for each bin size.

The results are displayed in Figure 1.

![Figure 1: Results of bin-EM-CEM algorithm on simulated data with different size of bins.](image)

From the Figure 1, we can find out some laws in the result. Firstly, we can see that, the accuracy increases significantly (from 0.8486 to 0.9410) with the number of bins from 15 to 1000. Then the accuracy’s growing tendency becomes slow when the number of bins gets over 1000. Secondly, for the aspect of CPUtime, as in our programming, we only take into account the non-empty-bins, as a result the CPUtime increases following the growing of the number of non-empty-bin. The speed of the increasing of CPUtime stays the same until the number of bins reaches 589. When the size of bin is smaller than 0.5 · 0.5, the bin-EM-CEM algorithm takes a lot of CPUtime. We notice that when the size of bins gets too small, binning data loses its meaning in the approach, since in this case the number of bins gets too big.

This experiment shows that the size of bins can directly affect the result of bin-EM-CEM algorithm. Bigger bin leads to less computational time but lower accuracy. When the size of bins gets too small, it deviates the purpose the binning data. A criterion is needed to decide the size of bins according to the practical situation.

IV. Conclusion

In this paper, we developed the bin-EM-CEM algorithms of eight general parsimonious Gaussian mixture models. Deriva-