Pattern Anal Applic (2015) 18:757–770
DOI 10.1007/s10044-014-0441-3

ORIGINAL ARTICLE

A new random approach for initialization of the multiple restart EM algorithm for Gaussian model-based clustering

Wojciech Kwedlo

Received: 12 March 2014 / Accepted: 21 December 2014 / Published online: 10 January 2015

Abstract The paper proposes a new method for initialization of the multiple restart EM algorithm for Gaussian mixture model-based clustering. The method initializes randomly both the mean vector and covariance matrix of a mixture component. In particular, the mean vector is initialized by a feature vector selected deterministically from a random subset of candidate feature vectors. The selection criterion is the maximum Mahalanobis distance from the already initialized mixture component centers. The covariance matrix of a component is initialized by randomly generating its eigenvalues and eigenvectors. In computational experiments, the used approach was compared with three other random EM initialization methods. The experiments were performed on synthetic datasets generated from the Gaussian mixtures with the different overlap characteristics, as well as on four real-life datasets. The results on synthetic data indicate that, for well separated clusters, for which the maximum pairwise overlap is not excessively high, the described method yields clusterings which correspond better to the original partitions of data, as indicated by the adjusted Rand index. The experiments on real data indicate that the performance of the method is comparable to other three methods for two smaller datasets and significantly better for two larger datasets.

Keywords Gaussian mixture models · EM algorithm initialization · Model-based clustering · Multiple restart EM

W. Kwedlo (✉)
Faculty of Computer Science, Bialystok University of Technology, Wiejska 45a, 15-351 Bialystok, Poland
e-mail: w.kwedlo@pb.edu.pl

1 Introduction

Mixture models [12, 25, 27] are very useful tools, widely applied in pattern recognition for modeling complex probability distributions. A finite mixture model $p(\mathbf{x}|\Theta)$ can be expressed by a weighted sum of K components:

$$p(\mathbf{x}|\Theta) = \sum_{m=1}^{K} \alpha_m p_m(\mathbf{x}|\theta_m), \tag{1}$$

where α_m is m-th mixing proportion and p_m is the probability density function of the m-th component. In (1), θ_m is the set of parameters defining the m-th component and $\Theta = \{\theta_1, \theta_2, \ldots, \theta_K, \alpha_1, \alpha_2, \ldots, \alpha_K\}$ is the complete set of the parameters needed to define the mixture. Θ is unknown and has to be estimated in the mixture learning process. The mixing proportions must satisfy the following conditions:

$$\alpha_m > 0, \quad m = 1, \ldots, K \quad \text{and} \quad \sum_{\mathrm{m}=1}^{\mathrm{K}} \alpha_{\mathrm{m}} = 1. \tag{2}$$

The number of components K is either known a priori or has to be determined during the mixture learning process. In the paper it is assumed that K is known.

The present paper considers the most widely used class of mixture models known as the Gaussian mixture models (GMMs), in which the probability density function of the m-th component is given by:

$$p_m(\mathbf{x}|\theta_m) = \frac{1}{(2\pi)^{d/2}|\Sigma_m|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu_m)^T \Sigma_m^{-1}(\mathbf{x}-\mu_m)\right), \tag{3}$$

where μ_m and Σ_m denote, respectively, the mean vector and covariance matrix, $|\cdot|$ denotes a determinant of a matrix and d is the dimension of the feature space. The set of parameters of the m-th component is $\theta_m = \{\mu_m, \Sigma_m\}$. Thus,

for a GMM Θ is defined by: $\Theta = \{\mu_1, \Sigma_1, \ldots, \mu_K, \Sigma_K, \alpha_1, \ldots, \alpha_K\}$. GMMs are widely used in such applications as data clustering [12, 27], data classification [15], speaker recognition [33], image segmentation and classification [29].

Estimation of the parameters of a GMM can be performed using the maximum likelihood approach. Given a set of independent and identically distributed feature vectors $X = \{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^N\}$, called the learning set, the log likelihood corresponding to a K-component GMM is given by:

$$\log p(X|\Theta) = \log \prod_{i=1}^{N} p(\mathbf{x}^i|\Theta) = \sum_{i=1}^{N} \log \sum_{m=1}^{K} \alpha_m p_m(\mathbf{x}^i|\theta_m). \tag{4}$$

The maximum likelihood estimate of the parameters is given by:

$$\Theta_{ML} = \arg\max_{\Theta} \{\log p(X|\Theta)\}. \tag{5}$$

Since the solution of this maximization problem cannot be obtained in a closed form (cf. [5]), a numerical optimization method has to be employed.

Because of its simplicity, the EM (expectation-maximization) algorithm [9, 24] is the most popular tool for maximum likelihood estimation of the parameters of GMMs. This procedure, however, is highly sensitive to initialization and easily gets trapped in a local optimum of (4). Therefore, the quality of the final solution is strongly dependent on the initial guess of the mixture parameters. The problem can be to some degree alleviated by performing multiple runs of the algorithm, each run starting from different initial conditions and returning the result with the highest $\log p(X|\Theta)$. This approach is called multiple restart EM (MREM).

Model-based clustering [12, 27] is one of the most important applications of GMMs. The aim of clustering can be defined as dividing a set of objects into K disjoint groups, called clusters, in such a way that the objects within the same group are very similar, whereas the objects in different groups are very distinct. In applications of GMMs to clustering, it is assumed that each feature vector was generated from one of K mixture components. The goal of clustering is to identify, for each feature vector, the mixture component from which it was generated. If it is possible to estimate the mixture parameters, clustering can be performed using maximum a posteriori (MAP) rule, by allocating a feature vector to a cluster (mixture component) with the highest posterior probability. From the Bayes theorem, this probability for the mixture component m can be expressed as:

$$h_m(\mathbf{x}^i) = \frac{\alpha_m p_m(\mathbf{x}^i|\theta_m)}{p(\mathbf{x}^i|\Theta)}. \tag{6}$$

Maximization of (6) is equivalent to finding the mixture index m with the highest value $\alpha_m p_m(\mathbf{x}^i|\theta_m)$.

The aim of the paper is the description and thorough evaluation of a new method for the initialization of the EM algorithm for Gaussian model-based clustering. The method, while retaining the probabilistic nature of random initialization, tries to initialize the component means by feature vectors located far away from the already initialized components. The advantage of the method compared to purely random approach is the increased probability of proper initialization of all cluster (components) centers in case of well-separated clusters, which can lead to better clustering results.

The remainder of the paper is organized as follows. In the next section, the research background related to the presented work is discussed. In Sect. 3, the EM algorithm for GMMs is described. Section 4 presents the author's approach to the initialization problem. Section 5 presents the results of computational experiments, in which the approach was compared to three other initialization methods for solving the clustering problem, using both synthetic and real data. The final section offers the concluding remarks.

2 Related research

The EM algorithm is a local optimization method. Because the log likelihood function (4) has numerous local maxima [25], performance of the EM algorithm is strongly dependent on the initial conditions. Many initialization methods have been proposed, but no single strategy outperforms the rivaling procedures in all cases. In this section, only the most popular approaches are mentioned.

The standard procedure for tackling the problem of the EM algorithm initialization is the multiple restart approach (MREM). In this approach the EM algorithm is run many times, each run being started with different random initial conditions. The result of the best run (in the sense of the highest final $\log p(X|\Theta)$) is returned as the final outcome. The most popular random method [25], which in this paper is called *rnd-nearest*, initializes the component means with randomly chosen feature vectors. After initialization of the means, the feature vectors are clustered by assigning a particular feature vector to the cluster represented by the closest mean. Next, the covariance matrix and the mixing proportion of each cluster are used as initial estimates of the parameters. A variant of this method which is known as *rnd-kmeans* uses the K-means algorithm [26], initialized by random feature vectors, to find component means. Next, the covariance matrices and mixing proportions are initialized using the same method as in the *rnd-nearest*

technique. However, the K-means algorithm is in itself susceptible to the local maxima.

Another random initialization method [10, 30] uses mixing proportions equal to $1/K$, selects random feature vectors as component means, and employs a spherical covariance matrix equal to $0.1\sigma^2\mathbf{I}$, where $\mathbf{I}$ is the identity matrix and

$$\sigma^2 = \frac{1}{d}\text{trace}(\Sigma). \tag{7}$$

In the above equation, Σ is the covariance matrix of the learning set X. This approach is referred to as *rnd-spherical*.

In [4], an extension of MREM called *emEM* was proposed. The idea of *emEM* involves performing several short EM runs using different random starting points and a lax convergence criterion. The mixture parameters obtained by the best (in the sense of the highest $\log p(X|\Theta)$) short run are used as a starting point for a long EM run. This strategy can be improved by repeating it many times until the available CPU time is exhausted. A variant of *emEM* called *rndEM* [21] reduces the short EM phase to the evaluation of $\log p(X|\Theta)$ of the random starting position.

In yet another approach to the GMM parameter estimation problem, the EM algorithm is combined with some global optimization method, e.g.. an evolutionary algorithm [1, 30] or a particle swarm optimizer [7]. However, global optimizers have high computational demands and this approach is limited to moderately sized datasets. Other algorithms proposed to deal with the problem of local maxima include versions of EM with split and merge operations [34, 36], a greedy learning method [35], and a component-wise method [10]. Some of these approaches try to estimate simultaneously the number of components K while searching for the optimal set of mixture parameters Θ.

The idea of using distant feature vectors to initialize center-based clustering algorithms is not new. In [17] it was described in the context of Lloyd's method for vector quantization, equivalent to the K-means algorithm. The original approach initialized the centroid of the first cluster with the feature vector with maximum norm and considered all feature vectors (instead of a random subset, as in the method presented in this article) as new cluster centers. For this reason, it had a deterministic nature and was unsuitable for multiple restart scenario. In [18] a probabilistic version of this method for the K-means algorithm was proposed. Since the K-means algorithm can be interpreted in the framework of model-based clustering (see e.g., [21] or [23]), the method proposed in this paper can be considered as a generalization of [18] to the cases of non-spherical and non-homogeneous covariances. The well-known K-means++ algorithm [2] initializes cluster centroids by random feature vectors chosen with the probability proportional to the squared shortest distance to the already initialized centroids.

The present article is an extension of the conference paper [19], in which initial experimental results achieved by the method were reported.

3 EM algorithm for Gaussian mixture models

The EM algorithm [9, 24] is an iterative method which, starting from an initial guess of mixture parameters $\Theta^{(0)}$, generates a sequence of estimates $\Theta^{(1)}, \Theta^{(2)}, \ldots, \Theta^{(j)}, \ldots$ with increasing log likelihood (i.e., $\log p(X|\Theta^{(j)}) > \log p(X|\Theta^{(j-1)})$. Each iteration j of the algorithm consists of two steps called the expectation step (E-step) and the maximization step (M-step). For GMMs, these steps are defined as follows [27, 32]:

- E-Step: Given the set of mixture parameters $\Theta^{(j-1)}$ from the previous iteration, for each $m = 1, \ldots, K$ and $i = 1, \ldots, N$, the posterior probability that a feature vector $\mathbf{x}^i$ was generated from the m-th component is calculated as:

$$h_m^{(j)}(\mathbf{x}^i) = \frac{\alpha_m^{(j-1)} p_m(\mathbf{x}^i|\theta_m^{(j-1)})}{\sum_{k=1}^{K} \alpha_k^{(j-1)} p_k(\mathbf{x}^i|\theta_k^{(j-1)})}, \tag{8}$$

  where $\theta_m^{(j-1)}$ and $\theta_k^{(j-1)}$ denote parameters of components m and k, in the iteration $j-1$, respectively.
- M-Step: Given the posterior probabilities $h_m^{(j)}(\mathbf{x}^i)$, the set of parameters $\Theta^{(j)}$ is calculated as:

$$\alpha_m^{(j)} = \frac{1}{N}\sum_{i=1}^{N} h_m^{(j)}(\mathbf{x}^i) \tag{9}$$

$$\mu_m^{(j)} = \frac{\sum_{i=1}^{N} h_m^{(j)}(\mathbf{x}^i) * \mathbf{x}^i}{\sum_{i=1}^{N} h_m^{(j)}(\mathbf{x}^i)} \tag{10}$$

$$\Sigma_m^{(j)} = \frac{\sum_{i=1}^{N} h_m^{(j)}(\mathbf{x}^i)(\mathbf{x}^i - \mu_m^{(j)})(\mathbf{x}^i - \mu_m^{(j)})^T}{\sum_{i=1}^{N} h_m^{(j)}(\mathbf{x}^i)}. \tag{11}$$

The E-steps and M-steps are applied alternately until a convergence criterion is met. In the experiments presented in this paper, a convergence criterion based on a relative improvement of log likelihood was employed. The EM algorithm was terminated if

$$\frac{\log p(X|\Theta^{(j)}) - \log p(X|\Theta^{(j-1)})}{\log p(X|\Theta^{(j-1)})} < \epsilon, \tag{12}$$

where $\epsilon \ll 1$ was a user-defined termination threshold (in the present case $\epsilon = 1e-5$).

4 Description of the proposed *rnd-maxmin* method

In the discussed method, equal initial mixing proportions ($\alpha_m^{(0)} = 1/K$) are used. In the next two sections, we describe how we generate initial component means and covariances.

4.1 Component means

The rationale behind the presented approach to initialization of the EM algorithm for GMMs was to correct the shortcomings of the random initialization methods shown in Fig. 1. The figure presents the stage of initialization algorithm for a three-component mixture at which two component means (labeled as 1st component and 2nd component) were correctly initialized by one of feature vectors marked by □. To increase the chances of discovering the optimal solution, it would be beneficial to initialize the mean of the third component by one of the feature vectors labeled △. However, in the standard random initialization method, all the feature vectors, including the vectors labeled □, have the same probability of being selected as the means of the third component. In that situation, the selection of one of the vectors marked by □ may lead to a suboptimal solution.

To amend this shortcoming, the method devised by the authors, denoted as *rnd-maxmin*, initializes the component means and covariances incrementally. When selecting new feature vectors as component means, it tries to disregard the vectors located closely to the already initialized components. The method has a probabilistic nature, which allows it to be used in the MREM algorithm. The initialization procedure can be described by the following steps:

1. Choose the mean of the first component $\mu_1^{(0)}$ as a random feature vector. Generate randomly the covariance matrix $\Sigma_1^{(0)}$. Set $m = 2$.

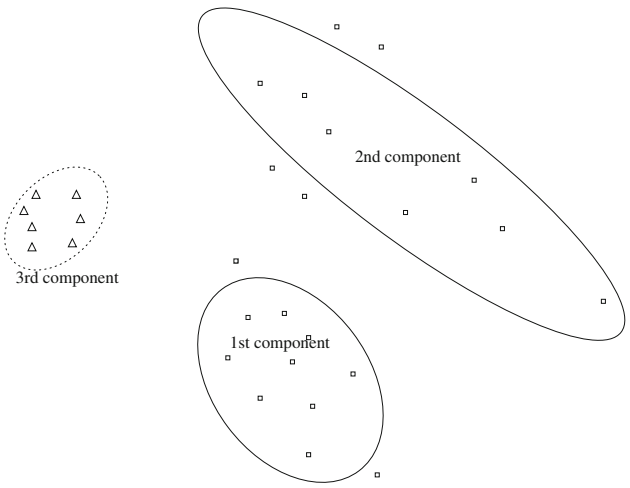

Fig. 1 Example of a three-component mixture illustrating the motivation for the *rnd-maxmin* method

2. Choose t (t is a parameter of the method) random unique feature vectors from the remaining (not yet selected as component means) elements of X. Denote by $X_r = \{\mathbf{x}_{r_1}, \mathbf{x}_{r_2}, \ldots, \mathbf{x}_{r_t}\}$ the set of the chosen vectors. For each $\mathbf{x}_{r_i} \in X_r$, compute the minimal squared Mahalanobis distance to the already chosen component means $\mu_1^{(0)}, \ldots, \mu_{m-1}^{(0)}$:

$$\mathrm{dmin}^2(\mathbf{x}_{\mathrm{r_i}}) = \min_{\mathrm{j}=1,\ldots,\mathrm{m}-1} \mathrm{d}_{\mathrm{M}}^2\left(\mu_{\mathrm{j}}^{(0)}, \Sigma_{\mathrm{j}}^{(0)}, \mathbf{x}_{\mathrm{r_i}}\right), \tag{13}$$

where $\mathrm{d}_{\mathrm{M}}^2(\mu_{\mathrm{j}}^{(0)}, \Sigma_{\mathrm{j}}^{(0)}, \mathbf{x}_{\mathrm{r_i}})$ is given by:

$$\mathrm{d}_{\mathrm{M}}^2(\mu_{\mathrm{j}}^{(0)}, \Sigma_{\mathrm{j}}^{(0)}, \mathbf{x}_{\mathrm{r_i}}) = (\mu_{\mathrm{j}}^{(0)} - \mathbf{x}_{\mathrm{r_i}})\left(\Sigma_{\mathrm{j}}^{(0)}\right)^{-1}(\mu_{\mathrm{j}}^{(0)} - \mathbf{x}_{\mathrm{r_i}})^{\mathrm{T}}. \tag{14}$$

3. Select as the m-th component mean the element of X_r with the largest minimal squared distance:

$$\mu_m^{(0)} = \arg\max_{\mathrm{X}_{r_i} \in \mathrm{X}_r} \mathrm{dmin}^2(\mathbf{x}_{\mathrm{r_i}}). \tag{15}$$

4. Generate randomly the covariance matrix $\Sigma_m^{(0)}$.
5. $m = m + 1$ if $m < K$ then goto Step 2. Otherwise, terminate the algorithm.

The preliminary experiments indicated that the choice of parameter t has an influence on the performance of the above initialization method. In the computational experiments reported in section 5, we have chosen value $t = 5$ for $K > 5$ and $t = K$ for $K \leq 5$, which yielded good results.

4.2 Component covariances

Random generation of the covariance matrix $\Sigma_m^{(0)}$ was based on the eigenvalue decomposition. Since a covariance matrix of a non-degenerate multivariate normal distribution is positive definite, it can be expressed as:

$$\Sigma_m^{(0)} = Q_m \Lambda_m Q_m^T, \tag{16}$$

where Λ_m is a diagonal matrix of eigenvalues with all diagonal elements positive, and Q_m is an orthogonal matrix of eigenvectors. In the presented method, the eigenvalues are generated first. They are generated randomly with the following two restrictions:

- The relation of the largest eigenvalue to the smallest eigenvalue cannot be greater than 10.
- The sum of all eigenvalues should be equal $\frac{1}{10dK}\mathrm{tr}(\Sigma)$, where tr is the trace of a matrix and Σ is the covariance matrix of the learning set X.

Two fulfill these two conditions we first randomly generate d positive numbers $\lambda_1, \lambda_2, \ldots, \lambda_d$ from the uniform distribution. Let $\lambda_{\max}$ be the maximal generated number. For each of the generated numbers λ_i, we check if $\lambda_i < 0.1\lambda_{\max}$.

If this condition holds we assign $\lambda_i = 0.1\lambda_{\max}$. Finally, we scale all the numbers by the same factor, so their sum fulfills the second condition.

In the next step, the orthogonal matrix Q_m is obtained by randomly generating a square $d \times d$ matrix (each element is generated independently from the standard normal distribution) and performing its QR decomposition [13]. The covariance matrix $\Sigma_m^{(0)}$ is then obtained using (16).

4.3 Influence of the proposed method on computational complexity of the MREM algorithm

The computational complexity of a single EM iteration is $O(NKd^2 + Kd^3)$. Because usually $K \ll N$ and $d \ll N$, this complexity can be approximated by $O(NKd^2)$. The computation of $\mathrm{tr}(\Sigma)$ requires $O(Nd)$ time and is performed only once per MREM run. Initialization of a single component by our method requires random generation of covariance matrix and computation of its inverse, which can be accomplished in $O(d^3)$ time. Initialization of a component also involves random generation of the mean which can be accomplished in $O(td^2)$ time. In all experiments, we used $t \leq 5$. For that reason, we can simplify this expression to $O(d^2)$. Since there are K components in a mixture, the total time of initialization is $O(K(d^2 + d^3)) = O(Kd^3)$. Because there are hundreds of EM iterations in a single MREM run, and for typical datasets $K \ll N$ and $d \ll N$, the overhead of initialization is usually very low.

We have performed some profiling experiments with our software, which confirmed the above analysis: the ratio of the runtime of our initialization method to the total runtime of the MREM was less than 1.5 %.

5 Experimental results

In this section, the results of the computational experiments performed on many synthetic datasets and four real-life datasets are presented. The experiments used GMMs for model-based clustering, in which, after the learning of model parameters by the MREM algorithm, the feature vectors were clustered using the MAP rule, as explained in Sect. 1. External cluster validity measure was employed to compare the partitions of data discovered in the course of clustering with the original partitions. In the case of synthetic data, the original partitions were known, because all the datasets were generated by a random number generator, making it possible to track the source (i.e., mixture component) of each feature vector. Also, the real-life datasets come with class labels, which allows to compare their original partitions with the obtained clustering results.

The external cluster validity measure used to compare the results of clustering methods was the adjusted Rand index (ARI) [16]. The expected value of ARI in the case of randomly generated partitions is 0. A higher value of ARI indicates a higher similarity between partitions; the maximum value of 1 means that two partitions are identical.

The main aim of computational experiments with synthetic data was to identify conditions in which MREM using our initialization method produces better clustering results than MREM using other common methods. We also wanted to demonstrate its performance on several real-life datasets.

5.1 The algorithms

The *rnd-maxmin* method described in Sect. 4 was compared with *rnd-nearest*, *rnd-spherical*, and *rnd-kmeans* methods outlined in Sect. 2. It is possible that, after the initialization of component means, *rnd-nearest* or *rnd-kmeans* methods obtain a singular component covariance matrix. This happens, for instance, when the number of objects in the initial cluster is smaller than the number of dimensions d. In this case, the used implementations of *rnd-nearest* and *rnd-kmeans* resort to *rnd-spherical* initialization of the covariance matrix of the component.

The four random methods were always compared on an equal runtime basis. Each of the methods was allocated the same amount of CPU time. Then, the MREM algorithm was run until the allocated time was exhausted. The result of the best (in the sense of the highest log likelihood) solution found by the MREM was considered as the outcome of the method.

It should be noted that the log likelihood for GMMs is unbounded [5]. As a consequence, the EM algorithm may converge to the boundary of the parameter space, with a covariance matrix of a component becoming arbitrarily close to singular and $\log p(X|\Theta)$ approaching infinity. In the present experiments with the multiple restart version of the EM algorithm, this issue was addressed by computing, after each EM iteration, condition numbers of component covariances [10]. If the largest condition number was above a fixed large threshold, the EM run was aborted and a new one was commenced.

5.2 Hardware and software

The EM algorithm and all four random initialization methods were implemented in C++ language and compiled by the Intel C++ compiler version 14.0.1 using optimizing options (-O3 -ipo -march=core2 -fno-alias). The algorithms were run on a Dell Poweredge 1950 server with two quad-core Intel Xeon 5355 (2.66 GHz) processors and 16 GB of RAM, running Ubuntu Linux 12.04. The

implementation of EM was parallelized [20] using OpenMP standard for shared memory computers, taking advantage of all eight cores of the system. Moreover, a cluster of 16 identical Dell servers was put to use, making it possible to perform up to 16 independent experiments in parallel. While the approach with multiple servers did not speed up a single EM run, it was able to speed up batches of EM runs which had to be performed.

To prove that the difference between EM initialization methods was not a random effect, nonparametric statistical tests using R software package [31] were performed.

5.3 Synthetic datasets

In the experiments with synthetic data, a generator recently proposed in [22] was employed which randomly generates Gaussian mixtures according to the user-defined overlap characteristics. The overlap ω_{ij} between two clusters i and j is defined as the sum of two misclassification probabilities $\omega_{j|i}$ and $\omega_{i|j}$ where:

$$\omega_{j|i} = Pr[\alpha_i p(\mathbf{x}|\mu_i, \Sigma_i) < \alpha_j p(\mathbf{x}|\mu_j, \Sigma_j) | \mathbf{x} \sim \mathcal{N}(\mu_i, \Sigma_i)], \tag{17}$$

and similarly:

$$\omega_{i|j} = Pr[\alpha_j p(\mathbf{x}|\mu_j, \Sigma_j) < \alpha_i p(\mathbf{x}|\mu_i, \Sigma_i) | \mathbf{x} \sim \mathcal{N}(\mu_j, \Sigma_j)]. \tag{18}$$

The overlap characteristics of mixtures obtained from the generator [22] were controlled by the two parameters: $\bar{\omega}$ specifying average pairwise overlap between components and $\check{\omega}$ specifying maximum pairwise overlap. In the experiments, the number of components K was fixed at 20 and mixtures with dimension $d \in \{2, 5, 10\}$ were generated. For each dimension, following [22], we used $\bar{\omega} \in \{0.001, 0.01, 0.05\}$. For each value of $\bar{\omega}$ we used three values of $\check{\omega}$: a value specifying very high maximum overlap, a value specifying very low maximum overlap and an intermediate value specifying moderate maximum overlap. Thus, for each dimension d, we used nine $(\bar{\omega}, \check{\omega})$ pairs. The parameter $\pi_\wedge$ determining minimal mixing proportion was set at $\pi_\wedge = 0.25 * 1/K = 0.0125$. Figure 2 shows six example datasets simulated from the mixtures obtained from the generator [22] for three values of $\bar{\omega}$ and extreme values of $\check{\omega}$.

As noticed in [22], both parameters influence overlap characteristics of generated mixtures. For instance, mixtures illustrated in Fig. 2a, b have very low average overlap ($\bar{\omega} = 0.001$) between component pairs. However, in Fig. 2b with high $\check{\omega} = 0.15$, the components are much better separated, except for two component pairs which mostly contribute to average overlap. Similar trends are repeated in Fig. 2c, d. Mixture components in Fig. 2e, f are both poorly separated.

The experimental setup was as follows. For each triplet $(d, \bar{\omega}, \check{\omega})$, 50 different random mixtures were generated using MixSim software [28]. The generated mixture parameters were stored to be used as ground truth for comparison. Then, for each mixture a single dataset consisting of 6000 feature vectors was realized, and for each of the datasets a single run of MREM using a given initialization method was performed. The runtime allocated for MREM was 100 s for $d = 10$, 50 s for $d = 5$, and 20 s for $d = 2$.

The feature vectors were clustered by the MAP rule using the best (in the sense of highest $\log p(X|\Theta)$) set of parameters found by MREM. Next, the partition found by cluster analysis was compared (using ARI) with the original partition of the data obtained by tracking a source (mixture) component which generated each feature vector.

The average ARI values obtained for the clustering based on ground truth mixture parameters were used as the baseline for comparison of the four EM initialization methods. For each method, the result is presented as % error relative to the ground truth mixture parameters. The % error ϵ_A of the method A is computed as $\epsilon_A = (\mathrm{ARI_T} - \mathrm{ARI_A})/\mathrm{ARI_T} * 100$, where $\mathrm{ARI_T}$ is the average (over 50 different mixtures) ARI obtained using the ground truth parameters and $\mathrm{ARI_A}$ is average ARI obtained using mixture parameters estimated by MREM using the initialization method A. A lower value of ϵ_A indicates a better performance; values close to 0 indicate that MREM running the initialization method A achieves a similar performance as clustering using the ground truth mixture parameters.

To assess the statistical significance of the differences in ARI and log likelihood, Wilcoxon signed rank test for paired data [8] was conducted in which, separately for each triplet $(d, \bar{\omega}, \check{\omega})$, the results obtained by the *rnd-maxmin* method were compared with the results obtained by the best of the three remaining methods.

The results of the experiments with synthetic data for dimension $d = 2$ and different values of $\bar{\omega}$ and $\check{\omega}$ are shown in Table 1. ϵ_A represents % error in ARI relative ground truth mixture parameters. The last row of the Table, labeled 'optimal ARI' shows the ARI obtained when clustering was performed using ground truth mixture parameters. L represents average (over 50 different mixtures) log likelihood obtained using a given initialization method. In each column, the best results for ϵ_A and L are shown in italics. In the row presenting the results of the *rnd-maxmin* method, p_ϵ and p_L represent Wilcoxon signed rank test p values in statistical comparison of the *rnd-maxmin* method with the best of the three remaining with respect to ϵ_A and L, respectively. If the comparison is statistically significant at 0.05 level, the corresponding result of *rnd-maxmin* is shown in bold.

The results obtained for $d = 5$ and $d = 10$ are shown in Tables 2 and 3, respectively.

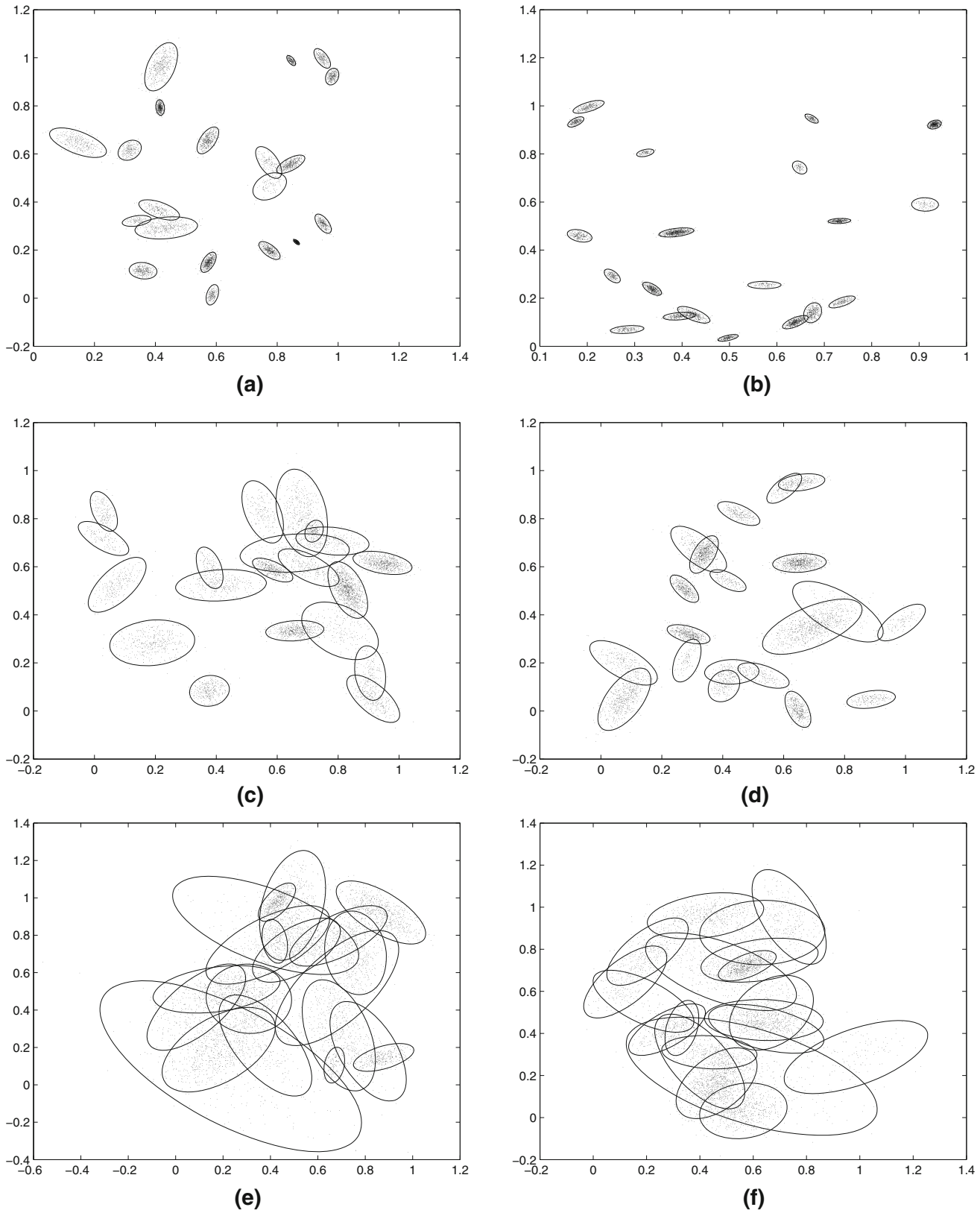

Fig. 2 Two-dimensional datasets simulated from 20-component mixtures with (a) $\bar{\omega} = 0.001$ and $\breve{\omega} = 0.05$, (b) $\bar{\omega} = 0.001$ and $\breve{\omega} = 0.15$, (c) $\bar{\omega} = 0.01$ and $\breve{\omega} = 0.25$, (d) $\bar{\omega} = 0.01$ and $\breve{\omega} = 0.60$, (e) $\bar{\omega} = 0.05$ and $\breve{\omega} = 0.45$, (f) $\bar{\omega} = 0.05$ and $\breve{\omega} = 0.75$. The ellipses are centered around component means and represent 95 % confidence regions

Table 1 Performance of model-based clustering using four EM initialization methods for $d = 2$. The best results for ϵ_A and L are shown in italics. Additionally, the results for the *rnd-maxmin* method are shown in bold if the comparison with the best of the remaining methods is significant at the 0.05 level

Method	$\bar{\omega}$	0.001			0.01			0.05		
	$\check{\omega}$	0.05	0.1	0.15	0.25	0.4	0.6	0.45	0.6	0.75
rnd-maxmin	ϵ_A	***1.16***	***1.72***	***1.13***	***2.71***	***2.96***	*2.74*	*10.72*	10.80	10.91
	p_ϵ	5.027e-05	0.0001269	3.312e-05	0.004019	0.001762	0.1156	0.4064	0.3956	0.8545
	L	***15010.5***	***18642.2***	***21519***	***8404.22***	***8916.42***	***10697.9***	*1891.88*	*2231.61*	*2644.72*
	p_L	1.911e-09	1.425e-09	9.802e-09	0.0002352	1.323e-06	3.454e-05	0.475	0.4064	0.2202
rnd-kmeans	ϵ_A	6.04	8.32	7.96	7.77	6.83	7.08	15.01	13.48	13.80
	L	14881.6	18376.1	21231.3	8382.19	8893.78	10669.4	1890.34	2229.95	2643.59
rnd-nearest	ϵ_A	5.17	6.47	7.31	7.09	6.15	5.94	11.71	12.07	*10.41*
	L	14780	18277.5	21037.4	8352.84	8870.62	10646.3	1888.96	2229.25	2641.7
rnd-spherical	ϵ_A	2.65	3.78	3.34	4.05	4.27	3.59	12.66	*10.02*	10.54
	L	14869.8	18427.7	21298.9	8386.96	8893.28	10671	1891.29	2230.63	2643.49
Optimal ARI		0.9886	0.9863	0.9879	0.8959	0.8860	0.8976	0.6870	0.6489	0.6591

The results from Tables 1–3 indicate that average overlap $\bar{\omega}$ alone is mostly sufficient for determining the attainable ARI, when clustering is performed using the ground truth parameters. However, both $\bar{\omega}$ and $\check{\omega}$ influence the performance of the four initialization methods with respect to ϵ_A.

The comparison of the performance of the *rnd-maxmin* method with the best of the remaining methods indicates that:

- *rnd-maxmin* dominates the others with respect to ARI and log likelihood when the average overlap between components is very low ($\bar{\omega} = 0.001$) regardless of the value of d and $\check{\omega}$.
- For situations with low average overlap ($\bar{\omega} = 0.01$), our method also dominates the others if the maximum overlap between component is not very high. However, the experiments identified the clear weakness of our approach: it usually yields results worse than the other methods (and sometimes by a large margin) if the maximum overlap between components is very high. We conjecture, that in these cases our strategy of locating component means far from already initialized components fails to place two means in close vicinity.
- A similar trend is repeated for $\bar{\omega} = 0.05$.
- The performance trends with respect to ARI and log likelihood are not identical (although they are very similar for $\bar{\omega} = 0.001$).

We summarize the results with synthetic data in Table 4, which shows the number of significant (at 0.05 level) wins scored by each method against the best of the other methods, separately for each value of $\bar{\omega}$. Table 4 demonstrates that if we remove *rnd-maxmin* from the competition, there is no clear winner among the three remaining methods.

5.4 Real datasets

In this section, we demonstrate the performance of the *rnd-maxmin* method on four real-life datasets with known class labels. Two of them: *iris* and *thyroid* are small-scale datasets available from UCI machine learning repository [3]. They are commonly used in comparisons of clustering algorithms. Two other datasets: *Brodatz* and *pendigits* are examples of more challenging problems, because of greater sample size, dimension of feature space and number of classes.

In the experiments the class information was discarded during GMM learning; it was used only to compute ARI. (Figs. 3a–6a and Tables 5–8). MREM algorithm with each of the initialization methods was run for 200 s larger *Brodatz* and *pendigits* datasets and for 0.1 s for smaller *iris* and *thyroid* datasets. This experiment was repeated 50 times, each time starting with a different seed of the random number generator. To assess statistical significances

Table 2 Performance of model-based clustering using four EM initialization methods for $d = 5$

Method	$\bar{\omega}$	0.001			0.01			0.05		
	$\check{\omega}$	0.02	0.03	0.15	0.12	0.45	0.8	0.25	0.5	0.75
rnd-maxmin	ϵ_A	***0.25***	***0.37***	***0.46***	***1.52***	***2.69***	**18.66**	*9.75*	*9.16*	**17.94**
	p_ϵ	4.341e-07	1.394e-07	8.949e-09	0.002767	0.04782	1.513e-09	0.186	0.09116	0.0001604
	L	***14586.5***	***16150.1***	***26070.9***	***3877.54***	*6143.61*	**793.217**	*−6186.06*	−5296.48	**−9373.68**
	p_L	2.683e-07	2.515e-08	2.035e-09	8.581e-06	0.206	9.347e-10	0.6022	0.8849	1.631e-08
rnd-kmeans	ϵ_A	3.92	3.55	5.03	5.13	5.75	14.06	12.14	12.35	16.98
	L	14498.9	16070.7	25914.3	3851.96	6132.82	1089.46	−6187.72	−5298.09	−9348.99
rnd-nearest	ϵ_A	2.32	3.28	3.57	2.56	4.31	10.22	10.96	10.01	*14.24*
	L	14482.2	16026.7	25867.6	3857.99	6131.11	*1237.43*	−6193.37	−5299.95	−9340.61
rnd-spherical	ϵ_A	1.84	2.42	3.12	2.36	3.37	*9.88*	10.40	9.83	14.41
	L	14504.2	16051.9	25902.8	3859.84	6142.69	1208.65	−6190.5	*−5296.36*	*−9340.41*
Optimal ARI		0.9872	0.9870	0.9881	0.8929	0.8980	0.9108	0.6622	0.6734	0.6659

Table 3 Performance of model-based clustering using four EM initialization methods for $d = 10$

Method	$\bar{\omega}$	0.001			0.01			0.05		
	$\check{\omega}$	0.02	0.03	0.15	0.12	0.45	0.8	0.25	0.5	0.75
rnd-maxmin	ϵ_A	***0.24***	***0.79***	***1.20***	***2.23***	***2.75***	**16.80**	***13.08***	***12.99***	**24.91**
	p_ϵ	1.52e-08	3.005e-07	0.0001484	2.813e-07	5.459e-05	2.064e-07	0.02052	9.622e-05	4.328e-06
	L	***8784.88***	***14524.5***	***23539.2***	***−11192.3***	***−10122.5***	**−16748.2**	−30004.1	*−29665.3*	**−37536.3**
	p_L	1.909e-09	4.005e-09	0.0009293	1.631e-08	5.928e-07	9.347e-10	0.7998	0.3667	1.705e-09
rnd-kmeans	ϵ_A	3.98	4.99	4.74	7.58	6.18	11.48	17.04	17.51	21.94
	L	8670.12	14395.2	23484.2	−11244	−10156.9	*−16379.2*	*−30003.1*	−29668.6	*−37458*
rnd-nearest	ϵ_A	2.04	3.20	2.98	4.69	3.98	10.44	16.00	15.97	*19.81*
	L	8645.98	14380.6	23435	−11252.3	−10176.8	−16469.9	−30055.6	−29722.6	−37513.3
rnd-spherical	ϵ_A	2.28	3.17	3.47	4.23	3.91	*9.30*	14.04	14.94	20.60
	L	8664.25	14387.3	23428.8	−11232.5	−10156	−16422.1	−30027.1	−29691.3	−37488
Optimal ARI		0.9879	0.9875	0.9905	0.9069	0.9001	0.9082	0.6750	0.6721	0.6885

of differences in ARI and $\log p(X|\Theta)$, the Wilcoxon rank sum test [8] was performed.

5.4.1 Pendigits dataset

The *pendigits* dataset is available from the UCI machine learning repository [3]. It describes handwritten images of ten digits. Each digit is represented by a 16-dimensional feature vector. In [30] a subset of this dataset describing the first five digits was used for GMM learning. To evaluate the learning algorithm in a more challenging setting, in the presented experiments the complete dataset ($K = 10$) consisting of 10992 feature vectors was used. Similarly to [30], first the dimensionality was reduced using principal component analysis (PCA), retaining the first nine principal components which together captured more than 95 % of the total variance.

Table 4 Summary of the experiments with synthetic datasets. For each value of $\bar{\omega}$ and each method, we report the number of experiments (out of 9 conducted experiments) in which the method performed significantly better (at 0.05 level) than the best of the remaining methods. The numbers before "/" signs concern ARI; the numbers after these signs concern log likelihood

Method	$\bar{\omega} = 0.001$	$\bar{\omega} = 0.01$	$\bar{\omega} = 0.05$
rnd-maxmin	9/9	6/6	2/0
rnd-kmeans	0/0	0/1	0/1
rnd-nearest	0/0	0/1	0/0
rnd-spherical	0/0	0/0	0/0

Figure 3 illustrates the progress toward the final solution of the four random initialization methods. The curves in Fig. 3a were obtained by measuring the ARI of the best (in the sense of highest log likelihood) solution found so far by the MREM algorithm in time steps of 10 s. The curves in Fig. 3b were updated after each EM run (average time of a single EM run was about 0.2 s).

Table 5 shows the comparison of the final solutions of the four random initialization methods based on 50 independent runs. The first column of the table indicates the name of the method. The second column shows the average (over 50 experiments) log likelihood obtained by the

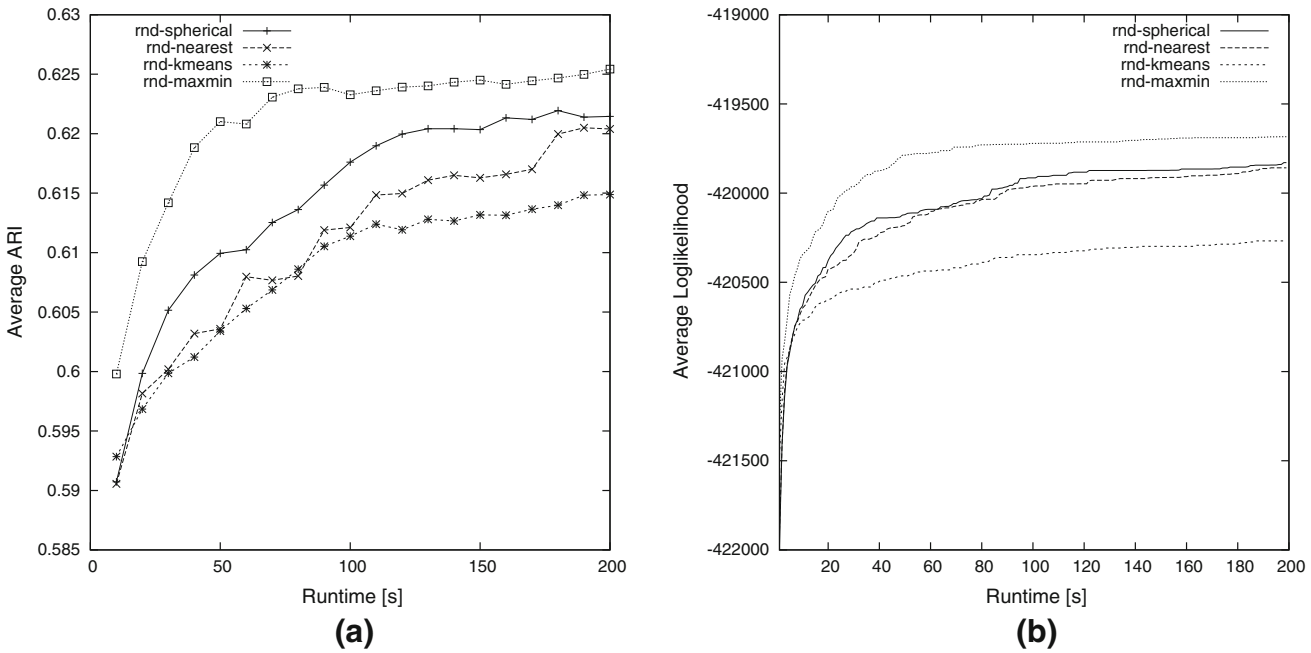

Fig. 3 Progress toward final solution of the four random initialization methods for *pendigits* data: (a) adjusted Rand index, (b) log likelihood of the best solution. The curves represent the averaged results of 50 independent experiments

Table 5 Comparison of four random initialization methods for *pendigits* dataset. The results are based on 50 independent 200 s runs of MREM

Method	$\log p(X\|\Theta)$	*pL*	ARI	*pARI*
rnd-spherical	−419829 ± 142	4.35e–11	0.6214 ± 0.011	1.97e–4
rnd-nearest	−419849 ± 171	6.38e–10	0.6204 ± 0.012	3.35e–3
rnd-kmeans	−420267 ± 117	4.47e–18	0.6149 ± 0.007	2.65e–14
rnd-maxmin	−419687 ± 33.1	–	0.6254 ± 0.005	–

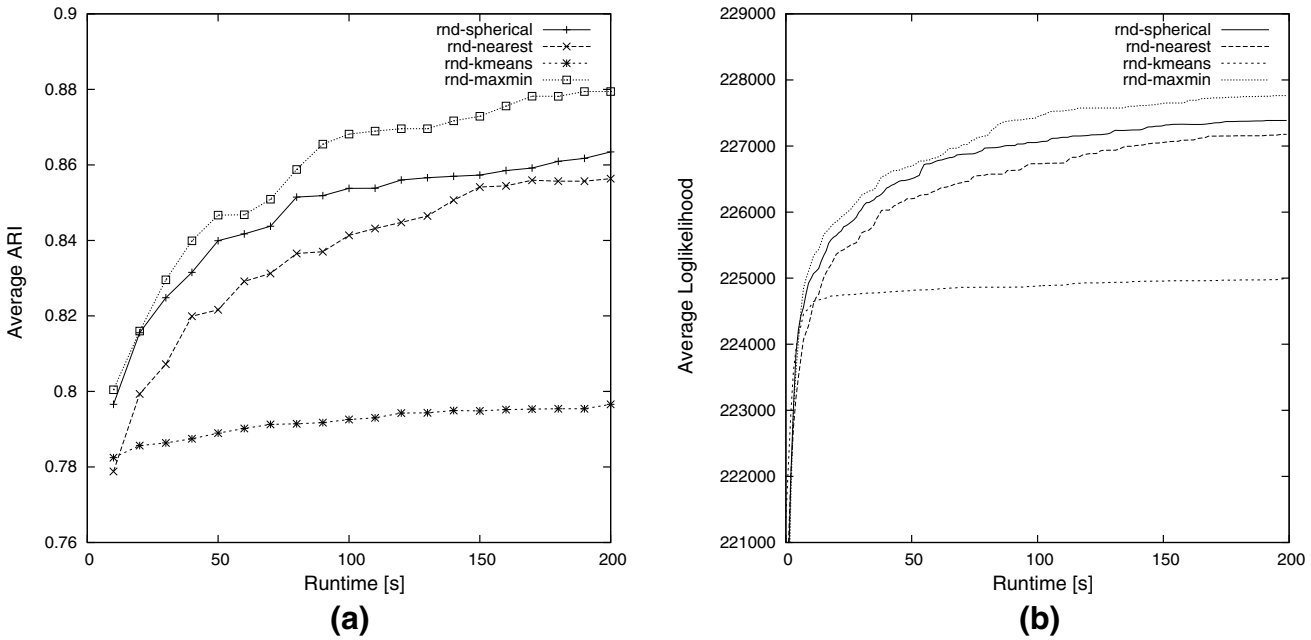

Fig. 4 Progress toward the final solution of the four random initialization methods for *Brodatz* data: (a) adjusted Rand index, (b) log likelihood of the best solution. The curves represent the averaged results of 50 independent experiments

method and the standard deviation after ± sign. The third column shows the p value for a statistical comparison of log likelihood (obtained using the Wilcoxon rank sum test) of the method with *rnd-maxmin*. The fourth column presents average ARI and the standard deviation. The last column shows the p value for a statistical comparison of ARI (obtained using the Wilcoxon rank sum test) of the method with *rnd-maxmin*.

The results in Fig. 3 and Table 5 indicate that the *rnd-maxmin* method outperforms the three other random initialization methods with respect to ARI. Although the relative difference in ARI in comparison with the second best *rnd-spherical* is very small (about 0.5 %), it is statistically significant at the 0.05 level. The difference in log likelihood is significant as well.

5.4.2 Brodatz dataset

The *Brodatz* dataset originated from the Laboratory of Image Processing and Pattern Recognition (INPG-LTIRF). It was used in the framework of the Esprit project ELENA[1]. The original source of data was the Brodatz texture album [6]. Each object in the dataset belongs to 1 of 11 classes representing textures from the album. The objects are described by 40 features extracted using estimation of fourth-order modified moments in four orientations: 0, 45, 90 and 135 degrees [14]. Each of the 11 groups consists of 500 objects.

In this dataset there are linear dependencies between 40 features: the determinant of the covariance matrix is very near zero and the first 37 principal components capture 100 % of the total variance. Thus, this dataset is badly conditioned and a GMM cannot be estimated by the EM algorithm using the original features. To remove linear dependencies from this data, we used PCA to reduce the dimensionality to 37.

Figure 4 illustrates the progress toward the final solution of the four random initialization methods.

It is evident that MREM using *rnd-kmeans* is easily trapped in local minima of the log likelihood function, as its performance is far worse than the performance of the other three methods. Table 6 shows the comparison of the final solutions of the four random initialization methods based on 50 independent runs.

Similarly, as in the case of *pendigits* dataset, the *rnd-maxmin* method outperforms the others with respect to ARI of the clustering solution (1.9 % relative difference) and the log likelihood of estimated GMMs. The differences are significant at the 0.05 level.

5.4.3 Thyroid dataset

The *thyroid* dataset is also available from the UCI repository. It consists of the thyroid disease measurements of 215 patients from the same hospital. Each of the patients belongs to one of three groups with a known diagnosis: a

[1] https://www.elen.ucl.ac.be/neural-nets/Research/Projects/ELENA/elena.htm.

Table 6 Comparison of four random initialization methods for *Brodatz* dataset. The results are based on 50 independent 200 s runs of MREM

Method	$\log p(X\|\Theta)$	*pL*	ARI	*pARI*
rnd-spherical	227429 ± 471	1.21e−4	0.8634 ± 0.026	1.26e−4
rnd-nearest	227181 ± 587	3.97e−7	0.8564 ± 0.028	2.66e−7
rnd-kmeans	225001 ± 209	6.9e−18	0.7966 ± 0.011	9.34e−18
rnd-maxmin	227766 ± 384	–	0.8794 ± 0.018	–

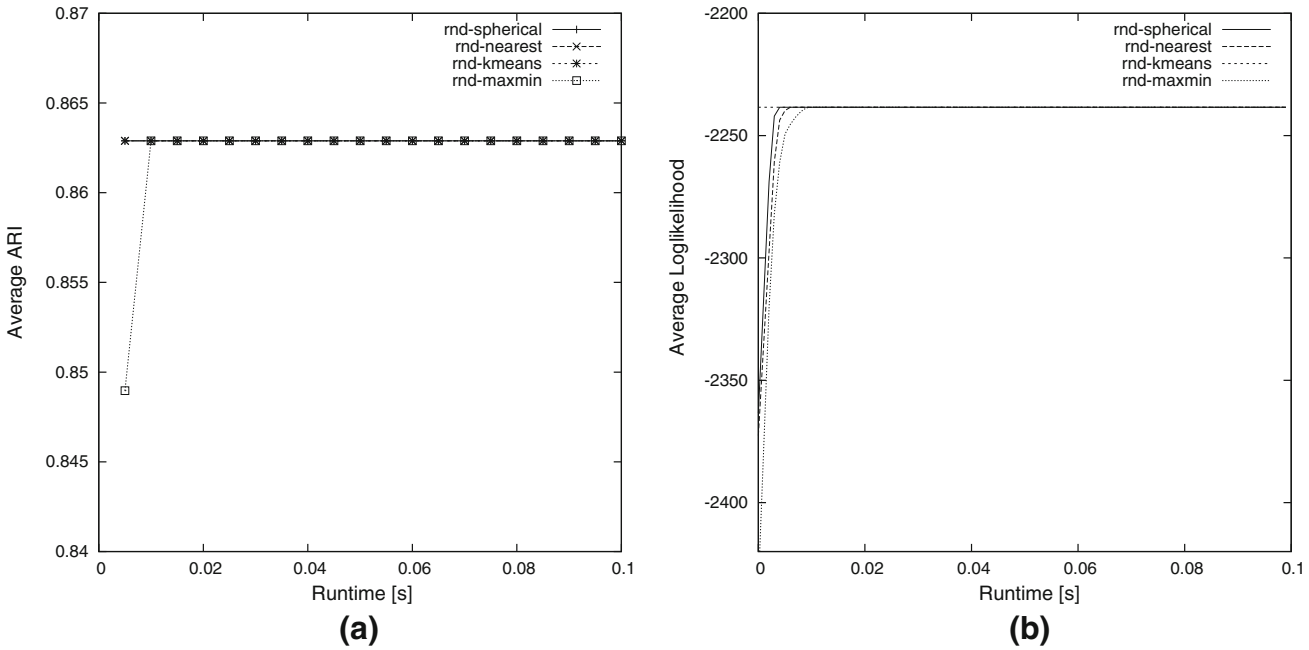

Fig. 5 Progress toward final solution of the four random initialization methods for *thyroid* data: (a) adjusted Rand index, (b) log likelihood of the best solution. The curves represent the averaged results of 50 independent experiments

Table 7 Comparison of four random initialization methods for *thyroid* dataset. The results are based on 50 independent 0.1 s runs of MREM

Method	$\log p(X\|\Theta)$	ARI
rnd-spherical	-2238.39 ± 0	0.8629 ± 0
rnd-nearest	-2238.39 ± 0	0.8629 ± 0
rnd-kmeans	-2238.39 ± 0	0.8629 ± 0
rnd-maxmin	-2238.39 ± 0	0.8629 ± 0

group of healthy patients (150 cases), patients with hyperthyroidism (35 cases), and patients with hypothyroidism (30 cases). Each patient is characterized by the results of five laboratory tests.

Figure 5 illustrates the progress toward the final solution of the four random initialization methods. The curves in Fig. 5a were obtained by measuring the ARI of the best (in the sense of highest log likelihood) solution found so far by the MREM algorithm in time steps of 0.005 s. The curves in Fig. 5b were updated after each EM run (the average time of a single EM run was about 0.002 s).

Table 7 shows the comparison of the final solutions of the four random initialization methods based on 50 independent runs. Since the same partition of data was found by every method in each of the 50 runs (i.e., there was no difference between the methods), we did not perform statistical significance tests.

5.4.4 *Iris dataset*

The well-known *iris* dataset [11], available from the UCI repository, contains four measurements of 150 samples of three iris species. The feature vectors are divided evenly into three classes.

Fig. 6 illustrates the progress toward the final solution of the four random initialization methods.

Table 8 shows the comparison of the final solutions of the four random initialization methods based on 50 independent runs. The results indicate that similarly, as in the

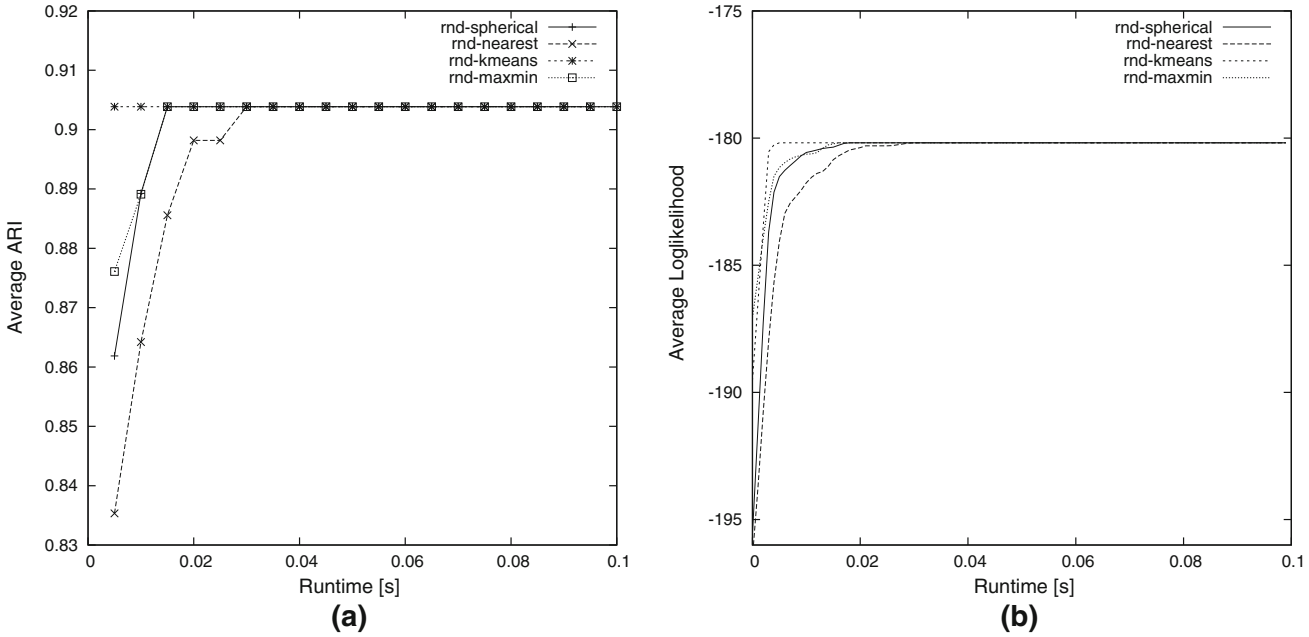

Fig. 6 Progress toward the final solution of the four random initialization methods for *iris* data: (a) adjusted Rand index, (b) log likelihood of the best solution. The curves represent the average results of 50 independent experiments

Table 8 Comparison of four random initialization methods for the *iris* dataset. The results are based on 50 independent 0.1 s runs of MREM

Method	$\log p(X\|\Theta)$	ARI
rnd-spherical	$-180.186 \pm 1e-4$	0.9039 ± 0
rnd-nearest	$-180.186 \pm 1e-4$	0.9039 ± 0
rnd-kmeans	-180.186 ± 0	0.9039 ± 0
rnd-maxmin	$-180.186 \pm 1e-4$	0.9039 ± 0

case of the *thyroid* dataset, there was no discernible difference between the four methods.

6 Conclusions and future work

In this article, a new random method for initialization of the EM algorithm for Gaussian mixture model-based clustering was proposed. The method was compared with three other approaches. The tests were carried out using multiple restart EM algorithm (MREM) on many synthetic datasets with different overlap characteristics and four real-life datasets.

The results of experiments confirm the well-known fact that no single method outperforms the others in all cases. Our approach also has its strengths and weaknesses. Generally, it performs better (or comparable) than other methods if the maximum overlap between clusters is not very high. Otherwise, it produces clustering results worse then the competing algorithms.

Finally, it should be noted that for small-scale clustering problems, as exemplified by experiments with the *iris* and *thyroid* datasets, the performance of model-based clustering solution estimated by the MREM algorithm may be similar (if not virtually the same) irrespective of the initialization method.

There are several possible directions of future work. To correct the deficiencies of our method, we plan to devise a hybrid solution combining *rnd-maxmin* with other random methods. For the MREM approach, the hybrid solution could be easily obtained by alternately running the EM algorithm using two or more different initialization methods (and keeping the mixture with the highest log likelihood). We plan to apply this hybrid method to classification problems and use it to model class-conditional densities in discriminant analysis [15]. This application of GMMs is often characterized by high overlap between mixture components. In this situation, a *rnd-maxmin* strategy alone performs poorly.

An anonymous reviewer has pointed out that there are other possibilities than Mahalanobis distance, for measuring distance between a feature vector and a component mean (for instance, the contribution of the feature vector to the log likelihood). We plan to implement different distance metrics in the *rnd-maxmin* method and test their performance.

Also, there are ideas to combine the described method for initialization of the EM algorithm with a global optimization method, e.g., differential evolution, similarly to that done with the K-means algorithm in [18].

Acknowledgments The author is grateful to Marek Kretowski for his comments on a draft version of the paper. The author would like to thank three anonymous referees whose comments and suggestions greatly improved the quality of the paper. This work was supported by the Bialystok University of Technology grant S/WI/2/2013.

References

1. Andrews JL, McNicholas PD (2013) Using evolutionary algorithms for model-based clustering. Pattern Recognit Lett 34(9):987–992
2. Arthur D, Vassilvitskii S (2007) K-means++: The advantages of careful seeding. In: Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'07), pp 1027–1035.
3. Bache K, Lichman M (2013) UCI machine learning repository. http://archive.ics.uci.edu/ml
4. Biernacki C, Celeux G, Govaert G (2003) Choosing starting values for the EM algorithm for getting the highest likelihood in multivariate Gaussian mixture models. Comput Stat Data Anal 41(3):561–575
5. Bishop CM (2006) Pattern Recognition and Machine Learning. Springer, New York
6. Brodatz P (1966) Textures: a photographic album for artists and designers. Dover, New York
7. Caglar A, Aksoy S, Arikan O (2012) Maximum likelihood estimation of Gaussian mixture models using stochastic search. Pattern Recognit 45(7):2804–2816
8. Conover WJ (1999) Practical Nonparametric Statistics. Wiley, New York
9. Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. J Royal Stat Soc Ser B 39(1):1–38
10. Figueiredo M, Jain A (2002) Unsupervised learning of finite mixture models. IEEE Trans Pattern Analysis Mach Intell 24(3):381–396
11. Fisher RA (1936) The use of multiple measurements in taxonomic problems. Ann Eugenics 7:179–188
12. Fraley C, Raftery AE (2002) Model-based clustering, discriminant analysis, and density estimation. J Am Stat Assoc 97(458):611–631
13. Golub GH, van Loan CF (1996) Matrix Computations. Johns Hopkins, Baltimore, MD
14. Guérin-Dugué A, Avilez-Cruz C (1993) Higher order statistics for natural textured images. In: ATHOS Workshop on System Identification and High-Order Statistics, Sophia-Antipolis, France.
15. Hastie T, Tibshirani R (1996) Discriminant analysis by Gaussian mixtures. J Royal Stat Soc Ser B 58(1):155–176
16. Hubert L, Arabie P (1985) Comparing partitions. J Classif 2(1):193–218
17. Katsavounidis I, Kuo CCJ, Zhang Z (1994) A new initialization technique for generalized Lloyd iteration. IEEE Signal Process Lett 1(10):144–146
18. Kwedlo W (2011) A clustering method combining differential evolution with the *k*-means algorithm. Pattern Recognit Lett 32(12):1613–1621
19. Kwedlo W (2013) A new method for random initialization of the EM algorithm for multivariate Gaussian mixture learning. In: Proceedings of the 8th International Conference on Computer Recognition Systems CORES 2013, Springer, pp 81–90.
20. Kwedlo W (2014) A parallel EM algorithm for Gaussian mixture models implemented on a NUMA system using OpenMP. In: Proceedings of the 22nd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing PDP 2014, IEEE CPS, pp 292–298.
21. Maitra R (2009) Initializing partition-optimization algorithms. IEEE-ACM Trans Comput Biol Bioinform 6(1):144–157
22. Maitra R, Melnykov V (2010) Simulating data to study performance of finite mixture modeling and clustering algorithms. J Comput Graph Stat 19(2):354–376
23. Maitra R, Melnykov V, Lahiri SN (2012) Bootstrapping for significance of compact clusters in multidimensional datasets. J Am Stat Assoc 107(497):378–392
24. McLachlan G, Krishnan T (2008) The EM Algorithm and Extensions. Wiley, New York
25. McLachlan G, Peel D (2000) Finite Mixture Models. Wiley, New York
26. McQueen J (1967) Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, pp 281–297.
27. Melnykov V, Maitra R (2010) Finite mixture models and model-based clustering. Statist Surv 4:80–116
28. Melnykov V, Chen WC, Maitra R (2012) MixSim: an R package for simulating data to study performance of clustering algorithms. J Stat Softw 51(12):1–25
29. Permuter H, Francos J, Jermyn I (2006) A study of Gaussian mixture models of color and texture features for image classification and segmentation. Pattern Recognit 39(4):695–706
30. Pernkopf F, Bouchaffra D (2005) Genetic-based EM algorithm for learning Gaussian mixture models. IEEE Trans Pattern Analysis Mach Intell 27(8):1344–1348
31. R Core Team (2013) R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, http://www.R-project.org/
32. Redner RA, Walker HF (1984) Mixture densities, maximum likelihood and the EM algorithm. SIAM Rev 26(2):195–239
33. Reynolds D, Quatieri T, Dunn R (2000) Speaker verification using adapted Gaussian mixture models. Digit Signal Process 10(1):19–41
34. Ueda N, Nakano R, Ghahramani Z, Hinton GE (2000) SMEM algorithm for mixture models. Neural Comput 12(9):2109–2128
35. Verbeek JJ, Vlassis N, Kröse B (2003) Efficient greedy learning of Gaussian mixture models. Neural Comput 15(2):469–485
36. Zhang Z, Chen C, Sun J, Chan KL (2003) EM algorithms for Gaussian mixtures with split-and-merge operation. Pattern Recognit 36(9):1973–1983