

# Tessellation and Clustering by Mixture Models and Their Parallel Implementations\*

Qiang Du<sup>†</sup>

Xiaoqiang Wang<sup>‡</sup>

## Abstract

Clustering and tessellations are basic tools in data mining. The k-means and EM algorithms are two of the most important algorithms in the Mixture Model-based clustering and tessellations. In this paper, we introduce a new clustering strategy which shares common features with both the EM and k-means algorithms. Our methods also lead to more general tessellations of a spatial region with respect to a continuous and possibly anisotropic density distribution. Moreover, we propose some probabilistic methods for the construction of these clusterings and tessellations corresponding to a continuous density distribution. Some numerical examples are presented to demonstrate the effectiveness of our new approach. In addition, we also discuss the parallel implementation and performance of our algorithms on some distributed memory systems.

## 1 Introduction.

Algorithms for clusterings and tessellations have long been the basic tools in data mining and pattern recognition, as well as popular techniques in mesh generation and optimization. Through clustering, we may efficiently represent data for fast retrieval and reduce the complexity of data [15]. Clusterings and tessellations are in general interchangeable terms since a tessellation induces a clustering and a clustering can also form a tessellation. While clustering may suggest more on grouping behavior, the term tessellation, nevertheless, often implies emphasis on the shape and the arrangement of the patterns, that is, the geometric aspect of the clustering.

Mixture Model-based clustering forms a set of very important algorithms in clustering [11]. Meanwhile, the Centroidal Voronoi tessellations (CVTs) [16] lead to very important and elegant algorithms for optimal tessellations. Together, they put our study here into context.

Among the Mixture Model-based clustering, the so called k-means and EM (Expectation Maximization) algorithms [3, 21] are two of the most popular techniques. In a typical mixture model, each cluster is represented by a parametric distribution, exemplified by either a uniform distribution, a Gaussian, or a Poisson distribution, to represent the distribution of the element within a cluster. A natural idea would be to represent the entire data set by the mixture of these distributions. In one of the simplest setting, the  $k$ -means clustering [12, 18, 20] is aimed at representing all clusters by their cluster means and minimizing the in-cluster variance, i. e., the sum of all mean square distances between the elements in the clusters and their corresponding cluster means. On the other hand, the EM algorithm [3, 21] may be used to estimate the parameters associated with such distributions of all the clusters by the maximum likelihood criterion. With some constraints or conditions imposed, EM algorithm could be changed to a classification EM algorithm (CEM) [24].

CVTs are special Voronoi tessellations for which the generators of the tessellation are also the centers of mass (or means) of the Voronoi cells or clusters. CVTs is a point placement strategy that has been applied to the sensor placement problem, point and patch distribution for mesh-less computation, image segmentation and restoration, image binning and stippling, and mesh optimization [5, 7, 9, 14, 17]. The concept of CVT has also been combined with the Proper Orthogonal Decompositions to give a new way of performing model reduction for complex systems [6]. CVT in many ways resembles the features of k-means methods and it requires the specification of a given density function to tessellate a spatial domain.

In this paper, we provide a new interpretation of the CEM algorithm so that the CEM and the CVT or the k-means can be unified within a common framework. We notice that in practice, however, the EM or the CEM algorithm often gives a good clustering but does not necessarily lead to a nice tessellation; and the CVT, meanwhile, provides very regular tessellations, but it is not good at capturing anisotropic tessellations. A trade off between the CEM algorithms and the CVTs is

---

\*This research was supported in part by NSF-DMS 0196522 and NSF-ITR 0205232.

<sup>†</sup>Department of Mathematics, Penn State University, University Park, PA 16802 ([qdu@math.psu.edu](mailto:qdu@math.psu.edu)).

<sup>‡</sup>Department of Mathematics, Penn State University, University Park, PA 16802 ([wang@math.psu.edu](mailto:wang@math.psu.edu)).

thus proposed here to get a new anisotropic tessellation algorithm that can automatically detect anisotropy.

In many applications, the existing clustering algorithms are often used with a fixed set of sample points. Several issues may arise: one is concerned with the size of sample set we need; another is the efficient selection of the sample points, especially with respect to the large dimensionality of the data set; and the last is how these sample points may be used efficiently. In adopting the EM algorithm to continuous distributions and sample regions, successfully addressing the above issues becomes even more critical than in the discrete cases. In this regard, the *MacQueen's method* [20] can play an important role. We explore this and other probabilistic methods to construct and generalize the tessellation algorithms for both continuous and discrete distributions.

For very large data sets in high dimensional spaces, parallelization of the clustering algorithms is needed in order to efficiently scale up with the dimensionality and the cardinality of the data sets. On today's parallel computing environment, effective parallelization often requires high concurrency, minimal communication and good load balancing. As a huge amount of works have carried out in the last few decades on the subject of clustering, in particular, on the parallel and scalable implementations [1, 13], we do not attempt to survey all the relevant studies on the parallel mixture model based clustering methods. We refer to [4, 22, 23, 28] and the references cited therein. The parallel algorithm we discuss here is new and it couples with probabilistic sampling to achieve good parallel efficiency and clustering quality.

The rest of the paper is organized as follows: we start the paper with some basic introduction to the classical k-means and EM algorithms and their applications to the continuous distributions. We then discuss the generalizations and variants of both algorithms, followed by the presentation of their probabilistic constructions. Their parallel implementations using the message passing interface (MPI) are presented next. Various numerical examples are provided afterwards and some concluding remarks are given at the end.

## 2 Mixture Model-based Clustering

In order to compare the new algorithms under consideration here with the existing algorithms used for clustering, we first review a number of popular mixture model based approaches. Many of these approaches can be derived from different angles and be presented in equivalent forms. We merely outline some very basic formulations here.

Let the  $n$  sample points in  $R^d$  be clustered into  $k$  components with the  $i$ -th component being represented

by a Gaussian distribution parameterized by  $\mu_i$  (the mean), and  $\sigma_i$  (variance) for  $i = 1, 2, \dots, k$ . The density function of the component  $i$  is

$$(2.1) \quad \begin{aligned} f_i(x) &= \Phi(x, \mu_i, \sigma_i) \\ &= \frac{1}{\sqrt{2\pi|\sigma_i|}} \exp\left(\frac{-(x-\mu_i)^T \sigma_i^{-1} (x-\mu_i)}{2}\right) . \end{aligned}$$

Now, suppose that the prior probability (weight) of component  $i$  is  $a_i$ . The mixture density is then given by:

$$(2.2) \quad f(x) = \sum_{i=1}^k a_i f_i(x) = \sum_{i=1}^k a_i \Phi(x, \mu_i, \sigma_i)$$

We let  $\theta$  denote the collection of parameters:  $a_i, \mu_i, \sigma_i$ ,  $i = 1, 2, \dots, k$ . For a discrete data set  $x_1, x_2, \dots, x_n$ , denote the likelihood function by:

$$L(\theta) = \sum_{j=1}^n \log\left(\sum_{i=1}^k a_i \Phi(x_j, \mu_i, \sigma_i)\right)$$

which is the objective function to be maximized by the EM clustering algorithm.  $L(\theta)$  is sometimes called the *mixture likelihood*.

Denote the posterior probability on the  $i$ -th component of a point  $x$  by  $p_i(x)$ . By the Bayes formula,

$$p_i(x) \propto a_i \Phi(x, \mu_i, \sigma_i) ,$$

with  $\sum_{i=1}^k p_i(x) \equiv 1$ . This leads to the EM algorithm which is outlined below for completeness.

### ALGORITHM 2.1. (EM)

1. Initialize the set of parameters  $\theta$ .
2. **E-step:** for all  $j = 1, \dots, n$ ,  $i = 1, \dots, k$ , set

$$p_i(x_j) = \frac{a_i \Phi(x_j, \mu_i, \sigma_i)}{\sum_{l=1}^k a_l \Phi(x_j, \mu_l, \sigma_l)} .$$

3. **M-step:** update  $\theta$  by calculating

$$\begin{aligned} a_i &= \frac{1}{n} \sum_{j=1}^n p_i(x_j) , \\ \mu_i &= \frac{\sum_{j=1}^n p_i(x_j) x_j}{\sum_{j=1}^n p_i(x_j)} , \\ \sigma_i &= \frac{\sum_{j=1}^n p_i(x_j) (x_j - \mu_i)(x_j - \mu_i)^T}{\sum_{j=1}^n p_i(x_j)} . \end{aligned}$$

4. Repeat E-step and M-step until convergence.

Changing the mixture likelihood  $L(\theta)$  to *classification likelihood*:

$$L(\theta) = \sum_{i=1}^n \log(a_{\eta(i)} \Phi(x_{\eta(i)}, \mu_{\eta(i)}, \sigma_{\eta(i)}))$$

with  $\eta(i)$  denoting the index of the cluster the point  $x_i$  belonging into, we are led to the Classification EM Algorithm (CEM).

ALGORITHM 2.2. (CEM)

1. Initial parameters  $\theta$ .
2. **E-step:** for all  $j = 1, \dots, n$ ,  $i = 1, \dots, k$ , set

$$p_i(x_j) = \frac{a_i \Phi(x_j, \mu_i, \sigma_i)}{\sum_{l=1}^k a_l \Phi(x_j, \mu_l, \sigma_l)}.$$

3. **Classification:** let  $\eta(i) = \arg \max_j p_j(x_i)$  and take  $p_j(x_i) = 1$  if  $j = \eta(i)$  and  $p_j(x_i) = 0$  if  $j \neq \eta(i)$ . Let  $G_j = \{x_i \mid p_j(x_i) = 1\}$  with cardinality  $I(G_j)$ .

4. **M-step:** update  $\theta$  by calculating

$$\begin{aligned} a_i &= \frac{\sum_{j=1}^n p_i(x_j)}{n} = \frac{I(G_i)}{n} \\ \mu_i &= \frac{\sum_{j=1}^n p_i(x_j) x_j}{\sum_{j=1}^n p_i(x_j)} = \frac{\sum_{x_j \in G_i} x_j}{I(G_i)} \\ \sigma_i &= \frac{\sum_{j=1}^n p_i(x_j) (x_j - \mu_i)(x_j - \mu_i)^T}{\sum_{j=1}^n p_i(x_j)} \\ &= \frac{\sum_{x_j \in G_i} (x_j - \mu_i)(x_j - \mu_i)^T}{I(G_i)}. \end{aligned}$$

4. Repeat E-step and M-step until the algorithm converges.

Notice that with  $\sigma_i$  being the identity matrix, the CEM gives exactly the k-means algorithm, which, in the conventional case, attempts to divide a sample set points  $x_1, x_2, \dots, x_n$  into  $k$  clusters with cluster means (centroids)  $m_1, m_2, \dots, m_k$  that minimizes the objective function:

$$(2.3) \quad \sum_{j=1}^n \|x_j - m_{\eta(j)}\|^2.$$

Here,  $\eta(j) = i$  gives the association rule that assign the  $j$ -th point to the  $i$ -th cluster. Note also that the CEM with a more general set of  $\{\sigma_i\}$  can provide more precise anisotropic characterization of the local point distribution.

### 3 A New Clustering Algorithm

We now present a new clustering algorithm that incorporates the features of both the  $k$ -means and the EM algorithms. To motivate the new algorithm, let us first give a different interpretation to the CEM.

We begin with the discrete case. Given  $n$  points,  $x_i$ ,  $i = 1, 2, \dots, n$  in  $R^d$ . Dividing these  $n$  points into  $k$  groups, an energy can be defined as follows:

$$\begin{aligned} E &= - \sum_{i=1}^k \sum_{j=1}^n [p_i(x_j) \log a_i \\ &\quad + p_i(x_j) \log \Phi(x_j, \mu_i, M_i)]. \end{aligned}$$

Applying the following constraints on  $p_i = p_i(x)$  and  $a_i$ :

$$(3.4) \quad \sum_{i=1}^k p_i(x) \equiv 1,$$

$$(3.5) \quad \sum_{i=1}^k a_i = 1$$

with  $\Phi$  being a probability density function such as the Gaussian defined in (2.1), we may look for the constrained minimizer of the above energy. The classical CEM algorithm may be viewed as a realization of this minimization process.

Taking the Gaussian distribution into the expression of  $E$ , we get

$$\begin{aligned} E' &= \sum_{i=1}^k \sum_{j=1}^n [-p_i(x_j) \log a_i + \frac{1}{2} p_i(x_j) (\log |M_i| \\ &\quad + (x_j - \mu_i)^T M_i^{-1} (x_j - \mu_i))] + c, \end{aligned}$$

where  $c$  is a constant. Thus, CEM also becomes an algorithm to minimize  $E'$ . Moreover, it is easy to verify that  $E'$  decreases for each step of CEM. In the energy expression, the parameters  $\{a_i\}$  can be regarded as the weights of each group. The more points a group have, the more likely for other points to be clustered into this group. This is in contrast with the standard  $k$ -means algorithm which assigns each group with the same weights. Moreover, the standard  $k$ -means algorithm also takes the identity matrix as the variance matrix  $M$  for each group. To maintain the individual characters of each group while preserving the equal weights for all groups, we construct a generalized version of the  $k$ -means algorithm: first setting all  $a_i$ 's equal, the new energy, after removing a constant, is changed to:

$$\begin{aligned} E^* &= \sum_{i=1}^k \sum_{j=1}^n [p_i(x_j) (\log |M_i| \\ &\quad + (x_j - \mu_i)^T M_i^{-1} (x_j - \mu_i))]. \end{aligned}$$

Then, our new algorithm is to minimize  $E^*$  with a constraint (3.4). Adopting the notation in (2.3), we may re-write  $E^*$  as

$$E^* = \sum_{i=1}^n [\log |M_{\eta(i)}| + (x_i - \mu_{\eta(i)})^T M_{\eta(i)}^{-1} (x_i - \mu_{\eta(i)})]$$

where  $\eta(i) = j$  means the  $i$ th point is assigned to the  $j$ th cluster. For any matrix  $B$ , let  $Tr\{B\}$  denote the trace of  $B$ . We can prove the following theorem.

**THEOREM 3.1.** *Assume that the function  $f(B) = \log |B| + Tr\{X^T B^{-1} X\}$  has a minimum for a matrix  $B$  among all positive definite matrices. Then, we have  $B = X X^T$ .*

Proof: Let  $A = B^{-1}$ , then  $A = (a_{ij})$  gives the minimum of function  $g(A) = -\log |A| + \text{Tr}\{X^T A X\}$ . And

$$\frac{\partial g}{\partial a_{ij}} = -\frac{|A_{ij}|}{|A|} + \sum_{\ell} X_{i\ell} X_{j\ell} = 0$$

where  $A_{ij}$  denotes the minors of  $A$  associated with the entry  $a_{ij}$ . So  $B = A^{-1} = |A|^{-1} A^*$  where  $A^*$  is the matrix whose  $ij$ -th element is  $A_{ij}$ , that is, the adjoint matrix of  $A$ . So  $B = X X^T$ .

This theorem leads directly to a new anisotropic clustering algorithm that adopts the anisotropic properties of EM or CEM as well as the centroid tessellation properties of CVTs.

ALGORITHM 3.1. (New Anisotropic Clustering)

1. Randomly select  $k$  points  $\mu_1, \mu_2, \dots, \mu_k$ , and set  $M_1 = M_2 = \dots = I_{d \times d}$ .
2. Assign a point  $x_i$  to the group  $j$  such that  $x_i$  gives the minimum of  $\log |M_j| + (x_i - \mu_j)^T M_j^{-1} (x_i - \mu_j)$  for  $j = 1, \dots, k, i = 1, \dots, n$ .
3. Update  $\mu_i$  by the centroid and  $M_i$  by the covariance matrix of group  $i$ .
4. Repeat steps 2 and 3 until the algorithm converges.

In the numerical experiments of section 6.2, this new algorithm is shown to give better tessellations than the CEM algorithm and CVTs. By viewing it as a kind of Mixture Model-based clustering method, a continuous version is also naturally available: given a density  $\rho = \rho(x)$  defined in a region  $\Omega$ , the Mixture Model-based clustering can be used to find a suitable parameter set  $\theta$ , such that the function  $f$  defined in (2.2) best 'approximate'  $\rho = \rho(x)$ . From these parameters and the clustering rules, we can also get a tessellation of the region  $\Omega$ . We refer algorithms for such purposes as the *continuous* algorithms, for instance, the continuous EM or continuous  $k$ -means.

With the proper modification of the original (discrete) anisotropic algorithm, the continuous anisotropic clustering algorithm can be formulated as:

ALGORITHM 3.2. (Continuous anisotropic clustering)

1. Randomly select  $k$  points  $\mu_1, \mu_2, \dots, \mu_k$ , and set  $M_1 = M_2 = \dots = I_{d \times d}$ .
2. For  $j = 1, \dots, k$ , assign a point  $x$  to the group  $G_j$  if  $x$  gives the minimum of  $\log |M_j| + (x - \mu_j)^T M_j^{-1} (x - \mu_j)$ .
3. Update  $\mu_i$  by the centroid and  $M_i$  by the covariance matrix of group  $i$ , i.e.

$$\begin{aligned} \mu_i &= \frac{\int_{G_i} \rho(x) x \, dx}{\int_{G_i} \rho(x) \, dx}, \\ M_i &= \frac{\int_{G_i} \rho(x) (x - \mu_i)(x - \mu_i)^T \, dx}{\int_{G_i} \rho(x) \, dx}. \end{aligned}$$

4. Repeat steps 2 and 3 until the algorithm converges.

Similar modifications to the EM, CEM and the  $k$ -means algorithms may also be used for the above purpose and such modified algorithms are called the continuous EM, continuous CEM and the continuous  $k$ -means algorithms.

The new anisotropic algorithms attempt to add more generality to the isotropic counterpart. In some sense, the characterization of the anisotropy may be related to the use of a general Riemannian metric. Thus, the construction outlined here is very much closely related to our earlier work [8] on the anisotropic centroidal Voronoi tessellations which is a generalization to the Riemannian metric case of the standard centroidal Voronoi tessellation defined with the Euclidean metric. The anisotropic CVTs studied in [8] is similar but different from the weight  $k$ -means algorithms based on the weighted Voronoi diagrams [2].

In contrast to the study made in [8] where the Riemannian metric (and thus the anisotropy) is predetermined, the algorithms proposed here gives an adaptive characterization of the parameters for the anisotropic distribution.

For practical applications, the above continuous algorithms may be realized in many different ways. One attempt is to uniformly sample the region and apply the discrete EM algorithm. Such a strategy does not work efficiently for nonuniform density distributions or distributions with sharp peaks. The more efficient ways are to select more sample points where the density function is larger. The details of an efficient algorithm for the continuous Mixture Model are given in the next section.

## 4 Probabilistic Methods for the Continuous Mixture Models

For more efficient implementations of the continuous algorithms, we present some other possible constructions, in particular, we emphasize on the probabilistic implementations.

First, let us recall the *MacQueen's method* [20], a very elegant random sequential sampling method which can be used for the continuous  $k$ -means algorithm.

ALGORITHM 4.1. (MacQueen's Method) Given a region  $\Omega$ , a density function  $\rho(x)$  defined on  $\Omega$ , and a positive integer  $k$ .

1. Choose an initial set of  $k$  points  $\mu_i$  in  $\Omega$  by Monte Carlo method; set  $j_i = 1$  for  $i = 1, \dots, k$ .
2. Sample a point  $x$  in  $\Omega$  by Monte Carlo method, according to the probability density function  $\rho = \rho(x)$ .
3. Find  $\mu_r$  that is the closest to  $x$  among all  $\{\mu_i\}_{i=1}^k$ .

4. Update  $\mu_r$  by  $\mu_r = (j_r \mu_r + x)/(j_r + 1)$  and  $j_r = j_r + 1$ . The other  $\mu_i$ 's remain unchanged.

5. If  $\{\mu_i\}_{i=1}^k$  meets a convergence criterion, terminate; otherwise, go to step 2.

The almost sure convergence of the energy for the random MacQueen's method has been proved [20]. For other studies on the algorithm, we refer to [5] and [10] for more discussions.

MacQueen's method can be directly generalized to the EM algorithm. A generalization of MacQueen's method given in [16] for the construction of CVT provides motivations to develop more efficient and alternative approaches to the EM methods.

The update formula in MacQueen's method can be re-written as

$$\mu_r^{new} = w_1 \mu_r^{old} + w_2 x$$

where  $w_1$  and  $w_2$  are two positive weight functions used in updating the means by new sample point. In general,  $w_1$  and  $w_2$  are functions of the iteration times  $j_r$  with constrain that  $w_1(j_r) + w_2(j_r) = 1$ . Another rule in choosing the weight functions is that  $w_1$  should be an increasing function, at least non-decreasing. These criteria assure that the old data are weighted more and more in calculating the new data. In MacQueen's method,  $w_1(j_r) = j_r/(j_r + 1)$  and  $w_2(j_r) = 1/(j_r + 1)$ . To add more flexibility to MacQueen's method, four parameters  $\alpha_1, \alpha_2, \beta_1, \beta_2$  are introduced in the weight functions:

$$(4.6) \quad w_1(j_r) = \frac{\alpha_1 j_r + \beta_1}{j_r + 1}, \text{ and } w_2(j_r) = \frac{\alpha_2 j_r + \beta_2}{j_r + 1}$$

where  $\alpha_1 + \alpha_2 = 1$  and  $\beta_1 + \beta_2 = 1$ . Thus, if  $\alpha_1 = \beta_2 = 1$  and  $\alpha_2 = \beta_1 = 0$ , it reduces to the MacQueen's method. We point out that if the weight function  $w_1$  is too large, the effect of the newer data maybe too weak which could lead to the need of more iterations. On the other hand, if the weight function  $w_2$  is too large, the effect of the newer data may be too strong and the convergence may not be assured.

Instead of adding only one new sample point, more sample points in each iteration may speed up the algorithms and improve concurrency [16]. And the more new sample points taken, the larger weight function  $w_2$  can be used to enlarge the effect of the new sample points.

Using the weight functions defined in (4.6), we present a new probabilistic method that can be viewed as a probabilistic version of the EM algorithm and a generalization of the random MacQueen's method.

**ALGORITHM 4.2.** (Probabilistic method for continuous EM) Given a region  $\Omega$ , a density function  $\rho = \rho(x)$  defined on  $\bar{\Omega}$ , and a positive integer  $k$ .

0. Choose a positive integer  $q$  and nonnegative constants  $\alpha_1, \alpha_2, \beta_1, \beta_2$ , such that  $\alpha_1 + \alpha_2 = 1$  and  $\beta_1 + \beta_2 = 1$ ; choose an initial set of  $k$  points  $\{\mu_i\}_{i=1}^k$  by using the Monte Carlo method; set  $a_i = 1/k$  for  $i = 1, \dots, k$ ; set the  $d \times d$  matrix  $M_i = I_{d \times d}$  for  $i = 1, \dots, k$ ; set  $j_i = 1$  for  $i = 1, \dots, k$ ;

1. Choose  $q$  points  $\{x_r\}_{r=1}^q$  in  $\Omega$  at random by the Monte Carlo method according to the probability density function  $f$ ;

2. **E-step:** for all  $j = 1, \dots, q$ ,  $i = 1, \dots, k$ , set

$$p_i(x_j) = \frac{a_i \Phi(x_j, \mu_i, \sigma_i)}{\sum_{l=1}^k a_l \Phi(x_j, \mu_l, \sigma_l)}.$$

3. **M-step:** compute:

$$a_i = \frac{(\alpha_1 j_i + \beta_1) a_i + (\alpha_2 j_i + \beta_2) \sum_{j=1}^q (p_i(x_j)/q)}{j_i + 1},$$

$$m_i = \frac{\sum_{j=1}^q p_i(x_j) x_j}{\sum_{j=1}^q p_i(x_j)},$$

$$\mu_i = \frac{(\alpha_1 j_i + \beta_1) \mu_i + (\alpha_2 j_i + \beta_2) m_i}{j_i + 1},$$

$$V_i = \frac{\sum_{j=1}^q p_i(x_j) (x_j - \mu_i)(x_j - \mu_i)^T}{\sum_{j=1}^q p_i(x_j)},$$

$$\sigma_i = \frac{(\alpha_1 j_i + \beta_1) \sigma_i + (\alpha_2 j_i + \beta_2) V_i}{j_i + 1}.$$

4. If the new  $\{a_i, \mu_i, \sigma_i\}_{i=1}^k$  meet some convergence criterion, terminate; otherwise update  $j_i = j_i + 1$  for  $n_i \neq 0$  and return to step 1.

The generalization for the continuous CEM algorithm can also be made very similarly as follows:

**ALGORITHM 4.3.** (Probabilistic method for continuous CEM) Given a region  $\Omega$ , a density function  $\rho(x)$  defined on  $\bar{\Omega}$ , and a positive integer  $k$ .

0. Choose a positive integer  $q$  and nonnegative constants  $\alpha_1, \alpha_2, \beta_1, \beta_2$ , such that  $\alpha_1 + \alpha_2 = 1$  and  $\beta_1 + \beta_2 = 1$ ; choose an initial set of  $k$  points  $\{\mu_i\}_{i=1}^k$  by using Monte Carlo method; set  $a_i = \frac{1}{k}$  for  $i = 1, \dots, k$ ; set  $d \times d$  matrix  $M_i = I_{d \times d}$  for  $i = 1, \dots, k$ ; set  $j_i = 1$  for  $i = 1, \dots, k$ ;

1. Choose  $q$  points  $\{x_r\}_{r=1}^q$  in  $\Omega$  at random by Monte Carlo method according to the probability density function  $f$ ;

2. **E-step:** For  $i = 1, \dots, k$ , gather together in the set  $W_i$  all sampling points  $x_r$  has the largest  $a_i \Phi(x_r, \mu_i, \sigma_i)$  among  $\{a_i, \mu_i, \sigma_i\}_{i=1}^k$ ;

3. **M-step:** let  $n_i$  be the number of points in  $W_i$ ; compute:

$$a_i = \frac{(\alpha_1 j_i + \beta_1) a_i + (\alpha_2 j_i + \beta_2) (n_i/q)}{j_i + 1},$$

let  $m_i$  be the mean of the points in  $W_i$ ; compute:

$$\mu_i = \frac{(\alpha_1 j_i + \beta_1) \mu_i + (\alpha_2 j_i + \beta_2) m_i}{j_i + 1},$$

and let  $V_i$  be the variance metric of the points in  $W_i$ , i.e.

$$V_i = \frac{1}{n_i} \sum_{x \in W_i} (x - \mu_i)(x - \mu_i)^T;$$

compute

$$\sigma_i = \frac{(\alpha_1 j_i + \beta_1) \sigma_i + (\alpha_2 j_i + \beta_2) V_i}{j_i + 1}.$$

4. If the new  $\{a_i, \mu_i, \sigma_i\}_{i=1}^k$  meet some convergence criterion, terminate; otherwise update  $j_i = j_i + 1$  for  $n_i \neq 0$  and return to step 1.

Our new continuous anisotropic clustering algorithm can also be constructed via probabilistic means as in the above algorithm for the continuous CEM by removing the coefficients  $a_i$ :

**ALGORITHM 4.4.** (Probabilistic method for the continuous anisotropic clustering) Given a region  $\Omega$ , a density function  $\rho(x)$  defined on  $\Omega$ , and a positive integer  $k$ .

0. Choose a positive integer  $q$  and nonnegative constants  $\alpha_1, \alpha_2, \beta_1, \beta_2$ , such that  $\alpha_1 + \alpha_2 = 1$  and  $\beta_1 + \beta_2 = 1$ ; choose an initial set of  $k$  points  $\{\mu_i\}_{i=1}^k$  by using Monte Carlo method; set  $d \times d$  matrix  $M_i = I_{d \times d}$  for  $i = 1, \dots, k$ ; set  $j_i = 1$  for  $i = 1, \dots, k$ .

1. Choose  $q$  points  $\{x_r\}_{r=1}^q$  in  $\Omega$  at random by Monte Carlo method according to the probability density function  $f$ .

2. For  $i = 1, \dots, k$ , gather together in the set  $W_i$  all sampling points  $x_r$  has the largest  $\Phi(x_r, \mu_i, \sigma_i)$  among  $\{\mu_i, \sigma_i\}_{i=1}^k$ .

3. Let  $n_i$  be the number of points in  $W_i$  and  $m_i$  be the mean of the points in  $W_i$ ; compute:

$$\mu_i = \frac{(\alpha_1 j_i + \beta_1) \mu_i + (\alpha_2 j_i + \beta_2) m_i}{j_i + 1}$$

let  $V_i$  be the variance metric of the points in  $W_i$ , i.e.

$$V_i = \frac{1}{n_i} \sum_{x \in W_i} (x - \mu_i)(x - \mu_i)^T,$$

and compute

$$\sigma_i = \frac{(\alpha_1 j_i + \beta_1) \sigma_i + (\alpha_2 j_i + \beta_2) V_i}{j_i + 1}.$$

4. If the new  $\mu_i, \sigma_i\}_{i=1}^k$  meet some convergence criterion, terminate; otherwise update  $j_i = j_i + 1$  for  $n_i \neq 0$  and return to step 1.

## 5 Parallel Implementations

In this section, we consider parallel implementations on distributed memory systems of the algorithms discussed in the section 4. For convenience, given two positive integers  $k$  and  $p$  and a non-negative integer  $i$  such as  $p \leq k$  and  $i < p$ , we define

$$R(i, p, k) = \begin{cases} [k/p] & i < k \pmod{p} \\ [k/p] + 1 & \text{otherwise} \end{cases},$$

and

$$\begin{cases} g_0 = G(0, p, k) = 0 \\ g_i = G(i, p, k) = \sum_{j=0}^{i-1} R(j, p, k) \end{cases}.$$

The parallel versions of the MacQueen's method can be found in [16]. Here we give the parallel versions of the EM and CEM discussed in section 4.

**ALGORITHM 5.1.** (Parallel EM algorithm) The region  $\Omega$ , density function  $f$  on  $\bar{\Omega}$ , and a positive integer  $k$  are given. We also have  $p$  processors with rank  $r = 0, 1, \dots, p-1$ .

0. For  $0 \leq r \leq p-1$ , set  $m_r = R(r, p, k)$ ; let each processor independently chooses its own initial set of  $m_r$  points  $\{\mu_i\}_{g_r+1}^{g_r+1+m_r}$  by using, e.g., the Monte Carlo method; set  $a_i = 1/k$ , matrices  $\sigma_i = I_{d \times d}$ , and  $j_i = 1$  for  $i = g_r + 1, \dots, g_{r+1}$ .

1. Communicate among all processors so that all points form a complete initial set of  $\{\mu_i, a_i, \sigma_i, j_i\}_{i=1}^k$ .

2. Each processor selects its own set of  $s$  points  $\{x_r^s\}_1^s$  in  $\Omega$  at random, e.g., by a Monte Carlo method, according to the probability density function  $f$ .

**3. E-step:**

3.a. On each processor, for  $i = 1, \dots, k$  and  $j = 1, \dots, s$ , set

$$p_i(x_j) = \frac{a_i \Phi(x_j, \mu_i, \sigma_i)}{\sum_{l=1}^k a_l \Phi(x_j, \mu_l, \sigma_l)}.$$

3.b. Through communication, each processor sends all of its points  $x_j$  and  $p_i(x_j)$  to the processor with rank  $r$  such that  $g_r < i \leq g_{r+1}$ ,  $j = 1, \dots, s$ . And each processor receives the data sent by the other processors and form points  $x_j$ ,  $j = 1, \dots, sq$  and  $i = g_r + 1, \dots, g_{r+1}$ .

**4. M-step:** On each processor, for  $i = g_r + 1, \dots, g_{r+1}$ , compute

$$a_i = \frac{(\alpha_1 j_i + \beta_1) a_i + (\alpha_2 j_i + \beta_2) \sum_{j=1}^{sq} p_i(x_j) / (sq)}{j_i + 1},$$

$$m_i = \frac{\sum_{j=1}^{sq} p_i(x_j) x_j}{\sum_{j=1}^{sq} p_i(x_j)},$$

$$\mu_i = \frac{(\alpha_1 j_i + \beta_1) \mu_i + (\alpha_2 j_i + \beta_2) m_i}{j_i + 1},$$

$$V_i = \frac{\sum_{j=1}^{sq} p_i(x_j) (x_j - \mu_i)(x_j - \mu_i)^T}{\sum_{j=1}^{sq} p_i(x_j)},$$

$$\sigma_i = \frac{(\alpha_1 j_i + \beta_1)\sigma_i + (\alpha_2 j_i + \beta_2)V_i}{j_i + 1},$$

and update  $j_i = j_i + 1$ .

4. If the new  $\{a_i, \mu_i, \sigma_i\}_{i=1}^k$  meet some convergence criterion, terminate; otherwise return to step 1.

Similarly, we list the parallel version of the CEM algorithm:

**ALGORITHM 5.2.** (Parallel CEM algorithm) Given the region  $\Omega$ , a density function  $f$  on  $\bar{\Omega}$ , a positive integer  $k$ , and  $p$  processors with rank  $r = 0, 1, \dots, p - 1$ .

0. For  $0 \leq r \leq p - 1$ , set  $m_r = R(r, p, k)$ ; then each processor independently chooses its own initial set of  $m_r$  points  $\{\mu_i\}_{i=g_r+1}^{g_{r+1}}$ , e.g., by using Monte Carlo method; set  $a_i = 1/k$ , matrices  $\sigma_i = I_{d \times d}$ ,  $j_i = 1$  for  $i = g_r + 1, \dots, g_{r+1}$ .

1. Communicate among all processors so that all points form a complete initial set of  $\{\mu_i, a_i, \sigma_i, j_i\}_{i=1}^k$ .

2. Each processor selects its own set of  $s$  points  $\{x_\ell^r\}_{\ell=1}^s$  in  $\Omega$  at random, e.g., by a Monte Carlo method, according to the probability density function  $f$ .

### 3. E-step:

3.1. On each processor, for each  $\mu_i$ ,  $i = 1, \dots, k$ , gather together all sampling points  $x_\ell^r$  which has the largest  $a_i \Phi(x_\ell^r, \mu_i, \sigma_i)$  in the set  $W_i^r$ .

3.2. Through communication, each processor sends the point set  $W_i^r$  to other processors with rank  $r$  such that  $g_r < i \leq g_{r+1}$ . And each processor receives the point set sent by other processors and gathers them into the  $W_i$  for  $i = g_r + 1, \dots, g_{r+1}$ .

4. **M-step:** On each processor, for  $i = g_r + 1, \dots, g_{r+1}$ , if  $W_i$  is not empty, let  $n_i$  be the points in  $W_i$ ; compute:

$$a_i = \frac{(\alpha_1 j_i + \beta_1)a_i + (\alpha_2 j_i + \beta_2)n_i/(sp)}{j_i + 1};$$

let  $m_i$  be the mean of the points in  $W_i$ ; compute:

$$\mu_i = \frac{(\alpha_1 j_i + \beta_1)\mu_i + (\alpha_2 j_i + \beta_2)m_i}{j_i + 1};$$

let  $V_i$  be the variance metric of the points in  $W_i$ , i.e.

$$V_i = \frac{1}{n_i} \sum_{x \in W_i} (x - \mu_i)(x - \mu_i)^T,$$

and compute

$$\sigma_i = \frac{(\alpha_1 j_i + \beta_1)\sigma_i + (\alpha_2 j_i + \beta_2)V_i}{j_i + 1};$$

then update  $j_i = j_i + 1$ .

4. If the new  $\{a_i, \mu_i, \sigma_i\}_{i=1}^k$  meet some convergence criterion, terminate; otherwise return to step 1.

Parallelizations of the new anisotropic clustering algorithms can also be made with much similarity as the above algorithms. We omit the details.

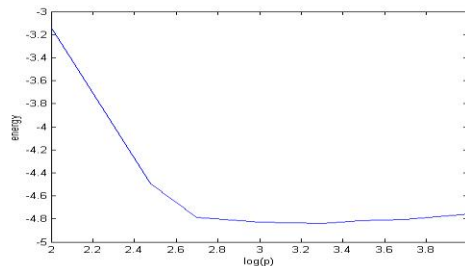


Figure 1: The plot of energy vs. the number of samples per iteration.

## 6 Numerical Experiments.

This section is divided into three subsections. The first subsection discusses the selection of the variables in our algorithms:  $\alpha_1, \beta_1, \alpha_2, \beta_2$ , and  $q$ , the number of samples points in each iteration. The second subsection discusses the application of the algorithm. Especially, we apply those algorithm to some continuous or discrete distribution clustering. We can also see some picture clustering experiments in this section. Finally, we will discuss the efficiency of parallel implementations.

**6.1 Variables selection** We first discuss the selection of  $q$ , the number of samples points in each iteration. As stated in section 4. A larger  $q$  means the more new information we can use in each iteration, thus less iteration steps is needed. On the other hand, a larger  $q$  also means more calculation needed in each iteration. To check the best  $q$  for a problem, we can fixed the number of all the sample points of all the iterations. In this experiment, we try to compare the energies with different  $q$  while the total number of samples is fixed at  $10^6$ . The anisotropic clustering algorithm used in this test. The energy of a clustering for a continuous distribution is approximated by sampling another  $10^5$  points.

In this experiment, the distribution we used here is:

$$f(x) = e^{-40(x-0.5)^2 - 160y^2} + e^{-40(x+0.5)^2 - 160y^2} + e^{-160x^2 - 40(y-0.5)^2} + e^{-160x^2 - 40(y+0.5)^2}.$$

The total number of sample points is  $10^6$  and the number of clusters  $k$  is 30.

Figure 1 show the curve of the energy v.s.  $p$ , the number of samples of each iteration. Because the total sample points are fixed at  $10^6$ , the larger the  $p$  is, the less iteration it used, and the total time will be the same. The quality of the clustering can be seen from the energy. The graph indicates, when  $p$  increase to  $10^3$ , the energy decrease rapidly, and after about  $2 \times 10^3$ , the

$\alpha_1$	$\beta_1$	$\alpha_2$	$\beta_2$	energy
0.5	0.5	0.5	0.5	-4.7996
0.0	0.0	1.0	1.0	-3.6764
1.0	1.0	0.0	0.0	0.9212
1.0	0.0	0.0	1.0	-3.9925
0.0	1.0	1.0	0.0	-4.7045
0.6	0.6	0.4	0.4	-4.7676
0.7	0.7	0.3	0.3	-4.8287
0.8	0.8	0.2	0.2	-4.7871
0.9	0.9	0.1	0.1	-4.7692

Table 1:  $\alpha_1, \beta_1, \alpha_2, \beta_2$  v.s. energy.

energy has small fluctuations Thus,  $p$  should be taken neither too small nor too large. Since the more clusters there are, the more sample points are needed. We may then postulate that there is a linear relationship between  $p$  and  $k$ , the number of cluster. Practically, we find that a good choice of  $p$  is to be set around  $50k$ , which means the average number of new sample points in each cluster is about 50.

The selection of the parameters  $\alpha_1, \beta_1, \alpha_2, \beta_2$  is also discussed in section 4. Several criteria are given there. To compare the effect of different  $\alpha_1, \beta_1, \alpha_2, \beta_2$ , we choose the same distribution and  $k$  as before, and set  $q$  2000. Table 1 gives the results after the same number of iterations. We see that, the choices  $\alpha_1 = \beta_1 = 0.7$  and  $\alpha_2 = \beta_2 = 0.3$  are about the best choices, significantly surpassing the performance of the MacQueen's method ( $\alpha_1 = \beta_2 = 1, \alpha_2 = \beta_1 = 0$ ). This clearly demonstrates the trade off between the old and the new data.

**6.2 Tessellation and clustering** The distribution given earlier as a superposition of four Gaussian distributions can be described by the left graph in Figure 2. We can see the four leaves. Thus we may set  $k = 4$  and try to catch the leaves. Direct calculations by CEM, k-means and the new anisotropic algorithm result almost the same graph given in the Figure 2 (right picture).

All these algorithms can get the centroid of each leaves. But except k-means, the matrix of other methods can provide us the information of the distribution for each cluster. To retrieve this information, we can add an additional so called *background cluster*, which is defined by  $\{x \mid \rho_i(x) < a, i = 1, \dots, k\}$ , where  $\rho_i$  is the distribution for cluster  $i$  and  $a$  can be set to 0.05 or 0.1. For  $a = 0.1$ , CEM and the anisotropic algorithm get the same picture, showed by the left picture in figure 3. Because k-means can not get the distribution of each cluster, at most we can only add a cluster whose points are far away for all the centroids. The picture can be showed by the right picture in figure 3.

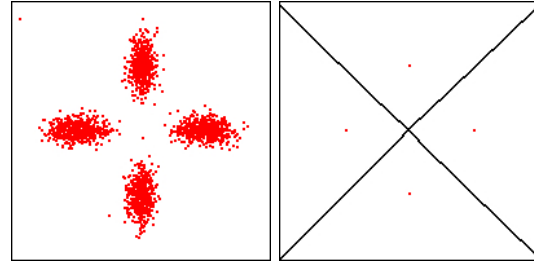


Figure 2: Distribution by sample points and 4 cluster result by CEM, k-means and the anisotropic algorithm. The distribution is given as a superposition of four Gaussian functions.

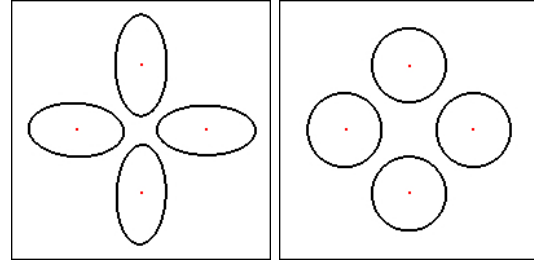


Figure 3: Add a new cluster: by CEM, the anisotropic algorithm (left) and by k-means (right).

From the result, one see that adding a new cluster is reasonable and it leads to a better result. So we apply this technique to all of our CEM and anisotropic clustering. Here, we set  $k = 4$  because we know there are four leaves. If such prior information is not available and  $k = 20$  is taken, the results, shown in Figure 4, are significantly different.

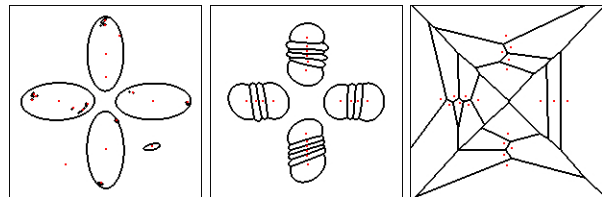


Figure 4: From left to right, the results of CEM, the anisotropic method and k-means when  $k = 20$ .

From this result, we can see, CEM can better keep the main 4 distributions, and the other clusters are either disappear or appear like noise. The anisotropic method can also get the 4 leaves, and it even divided each leaves into several small parts. k-means is not good at this condition.

Figure 5 and Figure 6 shows some other applications for those 3 methods.

In these three examples, CEM is good at getting the right clustering. In the figure 4, one can even detect

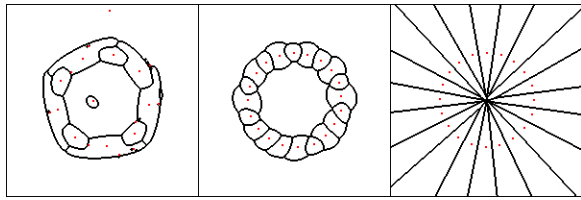


Figure 5: From left to right, the results of CEM, anisotropic method and k-means.  $k = 20$ , distribution is  $\exp(-40|x^2 + y^2 - 0.25|)$ .

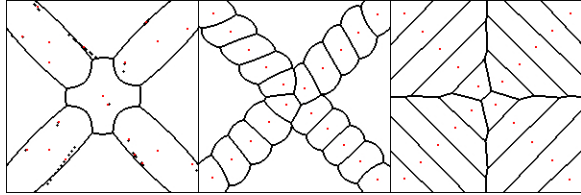


Figure 6: From left to right, the results of CEM, anisotropic method and k-means.  $k = 20$ , distribution is  $\exp(-40(x - y)^2) + \exp(-40(x + y)^2)$ .

how many clusters are exactly needed using CEM, while the anisotropic method and k-means can not carry out such a task. However, in order to get beautiful tessellations or for some reason, to get a further subdivision of the region into much more than four clusters, the anisotropic works better than CEM. Figures 4 and 5 are two good examples for this scenario.

**6.3 Applications to image analysis** Image segmentation and clustering are very important applications of the Mixture Models based clustering. Figure 7 gives another good example that illustrates the differences between those three algorithms.

Figure 7 shows the results get by those 3 methods. The top picture is the original picture. The second row shows the results for  $k = 3$ . The bottom row shows the results for  $k = 8$ . From these pictures, we can see the results from CEM can not give us more detail of the picture than the other two methods. For example, it can not recognize the sea behind the island. On the other hand, k-means shows a lot of details, but sometimes the details become over-crowded and affect the results of the clustering, such as the layers of the cloud and the waves in the sea. The anisotropic method provides a good balance between the CEM and k-means.

To test our algorithm for a much larger data set, we consider the problem of handwriting recognition. A set of normalized handwritten digits, automatically

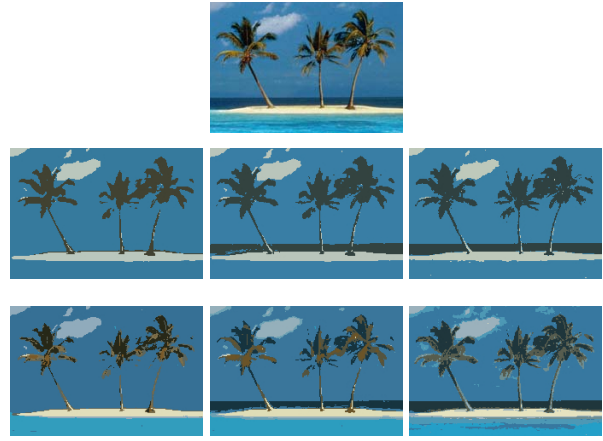


Figure 7: Picture clustering: original (top); results, from left to right, obtained by CEM, our new method, and k-means for  $k = 3$  (center) and  $k = 8$  (bottom).

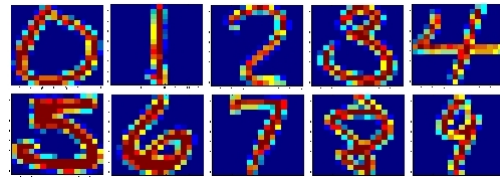


Figure 8: Sample handwriting.

scanned from envelopes by the U.S. Postal Service, is taken. The original scanned digits are binary and of different sizes and orientations; the images here have been deslanted and size normalized, resulting in 16 x 16 grayscale images, see Figure 8. The data set can be divided into 10 groups, each group contains one digit, from 0 to 9 respectively. And each group has 100 samples. Since the dimension of each sample is 256, we first apply the technique of ISOMAP [25] to decrease the dimension.

Figure 9 is the graph of the residuals with different number of nearest neighborhoods used for the dimension reduction. From this graph, the elbow effect is very obvious, indicating that it is good to choose the first 7 dimensions. The data in the reduced dimension is then stored as the new data set.

We then apply the CEM, our new algorithm, and the traditional k-means to this new 7-dimension data set. The rates of correct classification are respectively: 78.7% for CEM, 79.8% for k-means and 85.7% with our new method. More detailed classification results are presented in the tables 2-4. The graph of the first two dimensions of the reduced data set is plotted in the Figure 10, along with the classification diagrams based on the CEM and our new algorithm.

In this 10x10 table, the  $(i, j)$  element represents

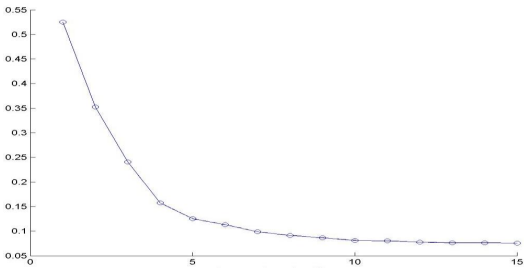


Figure 9: Residue variance as the function of the ISOMAP dimensionality.

15	1	9	71	0	4	0	0	0	0
0	100	0	0	0	0	0	0	0	0
19	1	62	4	12	1	0	1	0	0
8	0	0	85	0	7	0	0	0	0
6	0	5	0	82	0	0	0	0	7
6	1	0	0	0	90	2	0	0	0
2	6	0	0	0	5	87	0	0	0
3	0	3	0	2	4	0	77	0	11
4	0	0	0	0	0	0	0	96	0
1	0	0	0	1	3	0	2	0	93

Table 2: Digits redistribution by the CEM algorithm.

88	0	1	0	1	0	0	0	10	0
0	100	0	0	0	0	0	0	0	0
0	1	77	2	16	2	0	0	2	0
0	0	0	89	0	7	0	0	4	0
0	0	9	0	82	0	0	0	2	7
0	1	4	1	0	90	2	0	0	0
0	6	0	0	0	5	87	0	2	0
0	0	5	0	2	4	0	77	1	10
0	1	8	15	0	0	0	0	72	0
0	0	0	0	3	1	0	4	1	91

Table 3: Digits redistribution by the new algorithm.

100	0	0	0	0	0	0	0	0	0
6	87	0	0	0	1	6	0	0	0
10	3	59	13	2	1	2	8	2	0
0	0	1	91	0	2	0	1	4	1
17	0	0	0	72	0	1	0	0	10
2	4	2	4	5	77	2	1	0	3
0	8	1	0	0	15	75	0	1	0
3	0	0	0	0	0	0	77	0	20
4	0	0	7	1	5	0	0	78	5
1	0	0	1	10	0	0	6	0	82

Table 4: Digits redistribution by k-means.

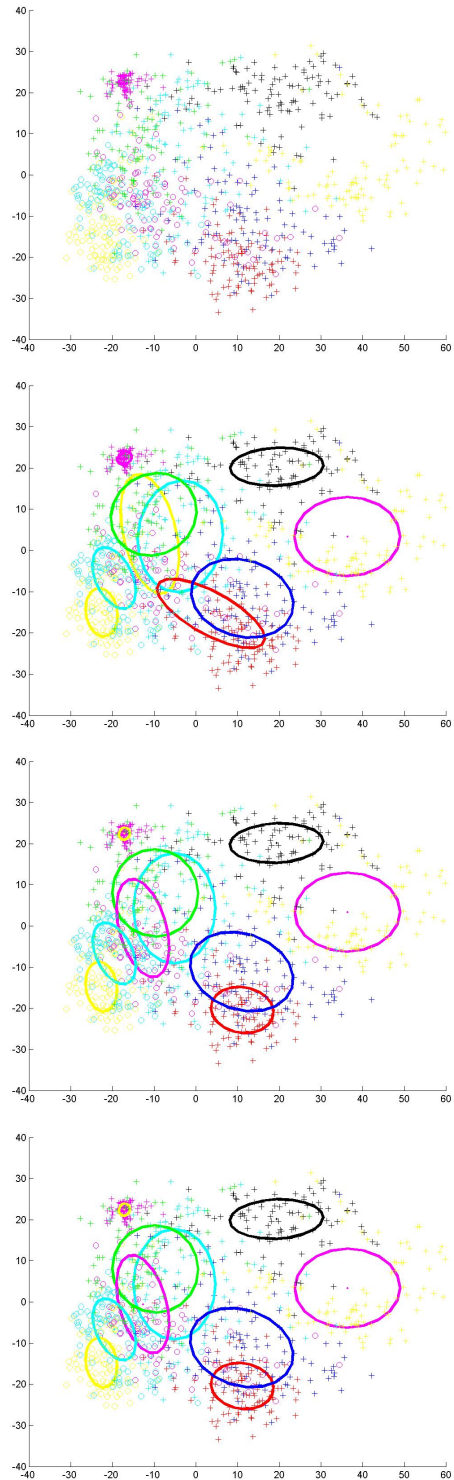


Figure 10: From top to bottom: two dimensional representations of the data and the classification by CEM, our new method and k-means.

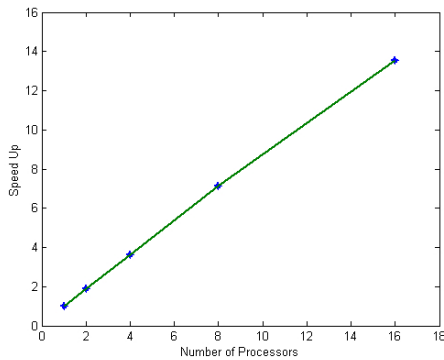


Figure 11: Parallel implementation of the anisotropic algorithm: with 200 clusters, 3000 iterations and 24,000,000 sample points for the distribution  $\exp(-40(x - y)^2) + \exp(-40(x + y)^2)$ .

the number of samples which belong to the  $i$ -th old group but are distributed to the new  $j$ -th cluster (the indices  $i$  and  $j$  correspond to the characters  $i - 1$  and  $j - 1$  respectively.) An examination of the table shows that the diagonal elements dominate the corresponding rows and columns, indicating a good classification. The lower accuracy rate for the CEM is mainly due to the confusion between the digits 0 and 3. The lower rate of the k-means, however, is affected by the lack of recognition of digit 2, and by the use of even weights and isotropic clusters applied to all the characters. Clearly, from Figure 10, we can see even in the reduced dimension the anisotropic clustering with respect to the different digits. Our new algorithm can take advantage of the good features of both the CEM and the k-means algorithms and overcome the anisotropy of the data set to get a better recognition of the characters with a higher accuracy rate.

**6.4 Parallel implementations** In this section, we check the performance of the parallel implementation. We will mainly check the parallel implementation for the anisotropic algorithm. The original distribution is  $\exp(-40(x - y)^2) + \exp(-40(x + y)^2)$ . We try to divide it to  $k = 200$  clusters. There are totally 3000 iterations. And each iteration sample  $q = ps = 8000$  points. So totally sample 24,000,000 sample points for 200 clusters. Figure 11 shows the graph of speed up v.s. number of processors.

From the graph, we can see the speed up is almost linear with respect to the number of processors.

We also consider the speed-up when the number of clusters and the total sample points increase with the number of processors. In Figure 12, the performance

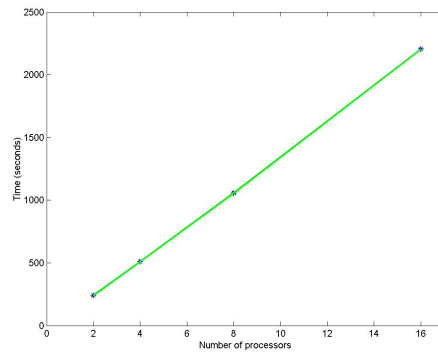


Figure 12: Parallel implementation of the anisotropic algorithm: with  $50p$  clusters, 3000 iterations and  $6,000,000p$  total sample points on  $p$  processors for the distribution  $\exp(-40(x - y)^2) + \exp(-40(x + y)^2)$ .

of the parallel implementation of our new anisotropic algorithm with  $k = 50p$  clusters, fixed 3000 iterations and total  $6,000,000p$  sample points (40 sample points per cluster per iteration) are shown for the distribution  $\exp(-40(x - y)^2) + \exp(-40(x + y)^2)$ , where  $p$  is the number of processors. It is clear that the problem is  $O(k^2)$  complexity when the number of sample points per cluster per iteration is fixed. Thus, when we linearly increase the number of processors, the time needed should be linearly increasing in theory. The verification of the linear increase in the actual timing again shows the good scalability of our algorithm.

## 7 Conclusion.

In the present work, we are interested in studying the mixture model based clusterings and tessellations. The classical algorithms such EM, CEM and k-means are examined. Probabilistic implementations are considered. An anisotropic algorithm which is, in some sense, a combination of the CEM and k-means, is introduced. It is also based a suitable generalization of the concept of centroidal Voronoi tessellations that has been extensively studied recently. Based on the results of some computational experiments, we are able to observe the different behaviors of these algorithms. Often, CEM is more effective in identifying the main clusters but it ignores the details. K-means, on the other hand, may provide too much details. The anisotropic algorithm can reveal the details while not losing sight of finding the main clusters, and it also can automatically identify the anisotropy in the data set. For continuous distributions, an efficient probabilistic implementation is proposed and tested numerically. The experiments also lead to vari-

ous strategies for the selection of the variables used in our algorithms. The inclusion and the fine tuning of these variables can significantly speed up our algorithm and make it far more superior than, for instance, the classical MacQueen's method. The probabilistic nature of the implementation also allows a nearly perfect parallelization. Our numerical results on a parallel cluster confirmed the desired efficiency and speed by.

## References

- [1] AGRAWAL, R., AND SHAFER, J., Parallel mining of association rules. *IEEE Transactions on Knowledge and Data Engineering* 8(6), pp.962–969, 1996.
- [2] F. AURENHAMMER AND H. EDELSBRUNNER, An optimal algorithm for constructing the Weighted Voronoi diagram, *Pattern Recognition*, 17, 1984, pp. 251-257.
- [3] DEMPSTER, A., N. LAIRD AND D. RUBIN, Maximum likelihood from incomplete data via the EM algorithm, *J. of Royal Stat. Society*, 39, pp. 1-38, 1977
- [4] I. DHILLON AND D. MODHA, A data clustering algorithm on distributed memory multiprocessors, *Workshop on Large-Scale Parallel KDD Systems*, 1999.
- [5] Q. DU, V. FABER AND M. GUNZBURGER, *Centroidal Voronoi tessellations: applications and algorithms*, *SIAM Review*, 41, 1999, pp.637-676.
- [6] Q. DU AND M. GUNZBURGER Centroidal Voronoi Tessellation Based Proper Orthogonal Decomposition Analysis, in *Control and estimation of distributed parameter systems*, Desch, W., Kappel, F. and K. Kunis ed., *International series of Numerical Mathematics*, 143, pp.137-150, Birkhauser, 2003
- [7] Q. DU, M. GUNZBURGER, AND L. JU, Meshfree, probabilistic determination of point sets and support regions for meshless computing, *Comput. Methods Appl. Mech. Engrg.*, 191(2002), pp. 1349-1366.
- [8] Q. DU AND D. WANG, Anisotropic centroidal Voronoi tessellations and their applications, submitted to *SIAM J. Sci. Comp.*, 2003
- [9] Q. DU AND D. WANG, Tetrahedral mesh generation and optimization based on centroidal Voronoi tessellations, *Int. J. Numer. Meth. Eng.*, 56, No.9, pp.1355-1373, 2002
- [10] Q. DU AND T. WONG, Numerical studies of the MacQueen's k-means algorithm for computing the centroidal Voronoi Tessellation, *Comp. Math. Appl.*, 44, 511-523, 2002.
- [11] C. FRALEY AND A. RAFTERY, Model-based clustering, discriminant analysis, and density estimation, *J Amer Stat Assoc* 97, 611-631, 2002.
- [12] R. GRAY AND D. NEUHOFF, Quantization, *IEEE Trans. Inform. Theory*, 44, pp.2325-2383, 1998.
- [13] E. HAN, G. KARYPIS AND V. KUMAR, Scalable parallel data mining for association rules, *Proceedings of the 1997 ACM SIGMOD international conference on Management of data*, Tucson, Arizona, p277 - 288, 1997.
- [14] S. HILLER, H. HELLOWIG AND O. DEUSSEN, Beyond stippling - Methods for distributing objects on the plane, *Comput Graph Forum*, 22, pp.515-522, 2003
- [15] A. JAIN, M. MURTY, AND P. FLYNN, Data clustering: a review, *ACM Computing Surveys*, 31(3):264-323, 1999.
- [16] L. JU, Q. DU AND M. GUNZBURGER, Probabilistic methods for centroidal Voronoi tessellations and their parallel implementations, *Journal of Parallel Computing*, 28, p.1477-1500, 2002
- [17] A. KSHEMKALYANI, A fine-grained modality classification for global predicates, *IEEE Tran Parall Distr* 14, pp.807-816, 2003
- [18] S. LLOYD; Least squares quantization in PCM, *IEEE Trans. Infor. Theory*, 28, pp.129–137, 1982.
- [19] J. MA, L. XU, AND M. I. JORDAN, Asymptotic convergence rate of the EM algorithm for Gaussian mixtures. *Neural Computation*, 12, pp.2881-290, 2000.
- [20] J. MACQUEEN; Some methods for classification and analysis of multivariate observations, in *Proc. Fifth Berkeley Symposium on Mathematical Statistics and Probability, I*, Ed. by L. Le Cam and J. Neyman, University of California, 1967, pp. 281–297.
- [21] G.J. MCLACHLAN AND T. KRISHNAN, *The EM algorithm and Extensions*, (1997), Wiley.
- [22] H. NAGESH, S. GOIL AND A. CHOUDHARY, A Scalable Parallel Subspace Clustering Algorithm for Massive Data Sets, *ICPP* 2000.
- [23] C.F. OLSON, *Parallel Algorithms For Hierarchical Clustering*, *Parallel Computing*, 21, 1995.
- [24] REDNER R. AND H. WALKER, Mixture densities, maximum likelihood and the EM algorithm, *SIAM Review*, 26 (2), pp195-239, 1984.
- [25] J. TENENBAUM, V. DE SILVA AND J. LANGFORD, A global geometric framework for nonlinear dimension reduction, *Science*, 290, pp2319-2323, 2000.
- [26] C. WU, On the Convergence Properties of the EM Algorithm, *Ann. Stat.*, 11, no.1, pp.95-103, 1983.
- [27] L. XU AND M. I. JORDAN. *On convergence properties of the EM algorithm for Gaussian mixtures. Neural Computation*, 8 (1), p.129–151, 1996.
- [28] X. XU, JAGER J., KRIEGEL H.-P., A Fast Parallel Clustering Algorithm for Large Spatial Databases, *Data Mining and Knowledge Discovery, An International Journal*, Kluwer Academic Publishers, 1999.