An Efficient Global K-means Clustering Algorithm

Juanying Xie
School of Electronic Engineering, Xidian University, Xi’an 710071, P. R. China
School of Computer Science, Shaanxi Normal University, Xi’an 710062, P. R. China
xiejuany@snnu.edu.cn

Shuai Jiang
School of Computer Science, Shaanxi Normal University, Xi’an, 710062, P. R. China
jiangshuai@stu.snnu.edu.cn

Weixin Xie
School of Electronic Engineering, Xidian University, Xi’an 710071, P. R. China
National Laboratory of Automatic Target Recognition (ATR), Shenzhen University, Shenzhen 518001, P.R. China
College of Information Engineering, Shenzhen University, Shenzhen 518001, P.R. China
wxxie@szu.edu.cn

Xinbo Gao
VIPS Lab, School of Electronic Engineering, Xidian University, Xi'an 710071, P.R. China
Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education of China, Xidian University, Xi'an 710071, P.R. China
xbgao@xidian.edu.cn

***Abstract*—K-means clustering is a popular clustering algorithm based on the partition of data. However, K-means clustering algorithm suffers from some shortcomings, such as its requiring a user to give out the number of clusters at first, and its sensitiveness to initial conditions, and its being easily trapped into a local solution et cetera. The global K-means algorithm proposed by Likas *et al* is an incremental approach to clustering that dynamically adds one cluster center at a time through a deterministic global search procedure consisting of N (with N being the size of the data set) runs of the K-means algorithm from suitable initial positions. It avoids the depending on any initial conditions or parameters, and considerably outperforms the K-means algorithms, but it has a heavy computational load. In this paper, we propose a new version of the global K-means algorithm. That is an efficient global K-means clustering algorithm. The outstanding feature of our algorithm is its superiority in execution time. It takes less run time than that of the available global K-means algorithms do. In this algorithm we modified the way of finding the optimal initial center of the next new cluster by defining a new function as the criterion to select the optimal candidate center for the next new cluster. Our idea grew under enlightened by Park and Jun’s idea of K-medoids clustering algorithm. We chose the best candidate initial center for the next cluster by calculating the value of our new function which uses the information of the natural distribution of data, so that the optimal initial center we chose is the point which is not only with the highest density, but also apart from the available cluster centers. Experiments on fourteen well-known data sets from UCI machine learning repository show that our new algorithm can significantly reduce the computational time without affecting the performance of the global K-means algorithms. Further experiments demonstrate that our improved global K-means algorithm outperforms the global K-means algorithm greatly and is suitable for clustering large data sets. Experiments on colon cancer tissue data set revealed that our new global K-means algorithm can efficiently deal with gene expression data with high dimensions. And experiment results on synthetic data sets with different proportions noisy data points prove that our global k-means can avoid the influence of noisy data on clustering results efficiently.**

***Index Terms*—clustering, K-means clustering, global K-means clustering, machine learning, pattern recognition, data mining, non-smooth optimization**

I. Introduction

Data clustering is frequently used in many fields, such as data mining, pattern recognition, decision support, machine learning and image segmentation [1-3]. As the most well known technique for performing non-hierarchical clustering, the K-means clustering [4] iteratively finds the k centroids and assigns each sample to the nearest centroid, where the coordinate of each centroid is the mean of the coordinates of the objects in the cluster. Unfortunately, K-means clustering algorithm

This work was supported by the Fundamental Research Funds for the Central Universities under grant GK200901006; and was supported by Natural Science Basis Research Plan in Shaanxi Province of P. R. China (Program No. 2010JM3004).

Corresponding author: xiejuany@snnu.edu.cn; & juanyingxie@gmail.com

doi:10.4304/jcp.6.2.271-279

is known to be sensitive to the initial cluster centers and easy to get stuck to the local optimal solutions [5]. Moreover, when the size of data set is large, it takes enormous time to find the solution. In order to improve the performance of the K-means algorithm, a variety of methods have been proposed.

There are a lot of variations of the K-means clustering algorithm. Here are some versions of them in recent years. Bradley and Fayyad [6] present a technique for initializing the K-means algorithm. They begin by randomly breaking the data into10, or so, subsets. They then perform a K-means clustering on each of the10 subsets, all starting at the same set of initial seeds, which are chosen randomly. The result of the 10 runs is 10K centre points. These 10K points are then themselves input to the K-means algorithm and the algorithm run 10 times, each of the 10 runs initialized using the K final centroid locations from one of the 10 subset runs. The resulting K centre locations from this run are used to initialize the K-means algorithm for the entire dataset. Huang [7] and Sun *et al* [8] extended the K-means paradigm to cluster categorical data. Strehl and Ghosh [9] introduced to combine multiple partitions of a set of objects into a single consolidated clustering without accessing the features or algorithms that determined these partitions. Likas *et al* [10] proposed the global K-means algorithm (The GKM algorithm), which is an incremental approach to clustering that dynamically adds one cluster center at a time through a deterministic global search procedure consisting of N (with N being the size of the data set) executions of the K-means algorithm from suitable initial positions. Experiment results show that the GKM algorithm considerably outperforms the K-means algorithms. Khan and Ahmad [11] proposed an algorithm to compute initial cluster centers for K-means clustering. This algorithm is based on two observations that some of the patterns are very similar to each other and that is why they have same cluster membership irrespective to the choice of initial cluster centers. Also, an individual attribute may provide some information about initial cluster center. Redmond and Heneghan [12] present a method for initializing the K-means clustering algorithm. They hinges on the use of a kd-tree to perform a density estimation of the data at various locations, then sequentially select K seeds, using distance and density information to aid each selection. However, it must be noted that kd-tree are well known to scale poorly with the dimensionality of the dataset. A new version of the GKM algorithm (The MGKM algorithm) was given by Bagirov [13] in 2008. In that article, a starting point for the k-th cluster center was computed by minimizing an auxiliary cluster function. Results of numerical experiment demonstrated the superiority of the new algorithm, but it requires more computational time than the GKM algorithm.

In this paper, a new version of the GKM algorithm is presented. We call it an efficient global K-means clustering algorithm, with EGKM for short. In our new algorithm we proposed a new method of how the next new cluster initial center is created by introducing some idea of K-medoids clustering algorithm suggested by Park and Jun in [14]. At the same time, in our EGKM, we tried to make the next cluster initial center is kept away from the existed centers as far as possible. The Experiments on fourteen well-known data sets from UCI machine learning repository show that our new algorithm outperforms the GKM algorithm greatly, which can reduce the computational load of the GKM without affecting the performance of it. And the experiments on colon cancer tissue data set from reveal that our EGKM can efficiently deal with gene expression data with high dimensions. Additional experiments on some synthetically generated data sets with noisy data demonstrate that our EGKM can not only reduce the computational load of the GKM algorithm without affecting the performance of it, but also avoid the influence of the noisy data on clustering result.

In the following section 2 we describe the GKM algorithm and its variations briefly. Section 3 introduces our proposed efficient global K-means clustering algorithm in detail. Experiment results and comparisons of our EGKM algorithm with the GKM algorithm and its variation with multiple restarts will be given in Section 4. Finally Section 5 concludes the paper.

II. THE GLOBLE K-MEANS AND ITS VARIATION

In this section, we give a brief description of the GKM algorithm [10] and its variation.

The GKM algorithm constitutes a deterministic global optimization method that does not depend on any initial parameter values and employs the K-means algorithm as a local procedure. It proceeds in an incremental way attempting to optimally add one new cluster center at each stage. This algorithm starts with one cluster $(k=1)$ and finds its optimal center position which corresponds to the centroid of the data set X. Then, it calculates the two-clusters problem $(k=2)$, the first cluster center is always placed at the optimal center position for the problem with $(k=1)$, while the second center is placed at the position of the data point x^n $(n=1,...,N)$, where, N is the size of the data set, and for each combination of the initial points the GKM executes the K-means algorithm, finally it chooses the combination of the initial points which gets the best clustering result as the solution for the clustering problem with $(k=2)$. Here, clustering error criterion is used to estimate the performance of the clustering result. And the clustering error used in [10] is the same as that of our MSE defined in (3) ~ (5). In general, let $(m_1^{k-1}, m_2^{k-1}, ..., m_{k-1}^{k-1})$ denotes the final solution for $(k-1)$ -clustering problem. Once the solution for the $(k-1)$ -clustering problem has been found, GKM tries to find the solution of the k -clustering problem as follows: it performs N (with N being the size of the data set) runs of the K-means algorithm with k -clusters where each run starts from the initial state $(m_1^{k-1}, m_2^{k-1}, ..., m_{k-1}^{k-1}, x^n)$, where x^n will travels all samples of data set, that is to say

$x^n, n = 1, \cdots, N$. The best solution obtained from the N runs is considered as the solution $(m_1^k, m_2^k, ..., m_k^k)$ of the k -clustering problem.

It must be noted that this is a rather computational heavy assumption, so this version of the GKM algorithm is not applicable for clustering the middle or large size of the data sets. Two procedures were introduced in [10] to reduce its complexity. We mention here only one of them, because the second procedure is applicable only to low dimensional data sets.

To accelerate the GKM algorithm, a fast GKM algorithm, we can call it FGKM, is proposed in [10], it's a straightforward method. Given the solution $(m_1^{k-1}, m_2^{k-1}, ..., m_{k-1}^{k-1})$ of the $(k-1)$ -clustering problem and the corresponding value $\psi_{k-1}^* = \psi_{k-1}(m^1, ..., m^{k-1})$ of the function ψ_k in (5), this FGKM algorithm does not execute the K-means algorithm for each data point repeatedly to find the optimal solution of the k -clustering problem. Instead it computes an upper bound $\psi_k^* \le \psi_{k-1}^* - b^i$, where,

$$b^i = \sum_{j=1}^{N} \max\{0, d_{k-1}^j - \left\|x^i - x^j\right\|^2\},\ i = 1, ..., N. \quad (1)$$

$$d_{k-1}^j = \min\{\left\|x^j - m_1^{k-1}\right\|^2, ..., \left\|x^j - m_{k-1}^{k-1}\right\|^2\}, \quad (2)$$

Here d_{k-1}^j is the squared distance between x^j and the closest cluster center among $(k-1)$ cluster centers $(m_1^{k-1}, m_2^{k-1}, ..., m_{k-1}^{k-1})$, that is, the squared distance between x^j and the center of a cluster where the sample x^j belongs to. And N is the size of the data set.

Then the data point $x^i \in X$ with the maximum value of b^i is chosen as the optimal initial center for the k -th cluster center.

III. Our Improved Method for Choosing Initial Seeds

In this paper we introduce some thoughts in reference [14] to our new algorithm in the procedure of finding the optimal initial center for the k -th cluster. At the same time, we introduce a new idea into the finding procedure to make the k -th cluster center is apart from the available $k-1$ cluster centers as far as possible. Our aim is to not only reduce the computational complexity shown in GKM algorithm, but also minimize the clustering error and avoid the influence of noisy data as well.

Suppose that the data set X with n objects: $(x^1, ..., x^n)$ having p variables should be grouped into k clusters $(k \prec n)$, here we use the following function as the clustering criterion. We call it MSE in this paper.

$$Minimize \qquad \psi_k(m) \quad (3)$$

$$\text{where,} \quad m = (m^1, ..., m^k) \in IR^{p \times k} \quad (4)$$

$$\text{and,} \quad \psi_k(m^1, ..., m^k) = \sum_{i=1}^{n} \min_{j=1,...,k} \left\|m^j - x^i\right\|^2. \quad (5)$$

Here $\|...\|$ is the Euclidean norm and m^j is the centroid of the j -th cluster. Let us define that the Euclidean distance between object x^i and object x^j is d_{ij} , and d_{ij} is given by:

$$d_{ij} = \sqrt{\sum_{r=1}^{p} (x^{ir} - x^{jr})^2}, \qquad i, j = 1, ..., n \quad (6)$$

Here, x^{ir} is the r -th variable of the object x^i . In order to compute an initial center, we defined v_i for each object x^i as following:

$$v_i = \sum_{j=1}^{n} \frac{d_{ij}}{\sum_{l=1}^{n} d_{jl}}, \quad i = 1, ..., n \quad (7)$$

Obviously, the point x^i that minimizes v_i is the one which has a comparatively high density around it, that is to say the sample with the minimum v_i tends to be the best initial center of one clustering problem. Then we give v_i a parameter to obtain the next initial cluster center. That is to say, we define a new function f_i in (8) to compute the optimal initial center for the next new cluster.

Suppose that the solution of the $(k-1)$ -clustering problem is $(m_1^{k-1}, m_2^{k-1}, ..., m_{k-1}^{k-1})$ and a new cluster center (i.e., the k -th initial center) is added at the location x^i that minimizes f_i as defined in (8). Then we execute the K-means algorithm to obtain the solution with k clusters.

$$f_i = \frac{v_i}{\sum_{j=1}^{k-1} d(x_i, m_j^{k-1})}, \qquad i = 1, ..., n \quad (8)$$

$$i = \arg\min_{i} f_i \, , \qquad i = 1,\ldots,n \qquad (9)$$

The addition of the parameter (i.e. the denominator of f_i) ensures that the new cluster initial center could be far away from the existing cluster centers. It should be noted that the new center we computed it by (8) is an optimal initial center for the new cluster. In order to prove this we will test our proposed algorithm on several well-known data sets in section IV. Now the efficient GKM clustering algorithm we proposed proceeds as follows:

Our efficient GKM clustering algorithm.

Step 1: (Initialization) Calculate the distance between each pair of all objects based on Euclidean distance, then calculate v_i for each object in (7). Select the point that minimize v_i as the first center. Set q=1.

Step 2 (Update centroids) Execute K-means algorithm and preserve the best q -partition obtained and their cluster centers $(m_1, m_2, \ldots, m_q)$.

Sept 3: (Stopping criterion) Set $q = q + 1$. If $q \succ k$, then stop.

Step 4: (Select the optimal initial center for new cluster) Calculate f_i for each object x^i in (8). Select the point which has the minimum value of f_i as the new cluster initial center, now the initial centers is $(m_1, m_2, \ldots, m_q, x^i)$ and go to Step2.

This version of the GKM algorithm proposed by us has an excellent feature that it requires much less calculation amount and shows less computational complexity. The distance between each pair of objects is computed only once, which contributes to the excellent feature. At the same time, the selection of the next cluster initial center can avoid the impact of noisy data on the clustering result. This proposed algorithm will be thoroughly compared with GKM algorithm and its variation in the next section.

IV. Results of Numerical Experiments

In this part, we did experiments of three algorithms of GKM, fast GKM, and our proposed EGKM on two kinds of data sets. They are real data sets from UCI machine learning repository [15] and from Princeton University gene expression project [16] and some artificial data sets, respectively. The experiments are described here in detail.

A. Experiments on real data sets

To verify the efficiency of our proposed algorithm EGKM, we accomplished many numerical experiments on fourteen well-known data sets from UCI machine learning repository [15]. And in order to demonstrate the performance of our algorithm EGKM in dealing with the high dimensional data set, we also conducted experiment on colon cancer tissues data set from Princeton University gene expression project [16]. The colon cancer tissues data set pertained to the article [17]. All these data sets we used in our paper are briefly described in Table 1. All the experiments have been carried out on a PC of Pentium-4 with CPU 1.86 GHz and RAM 512 MB. Here we must note that the Iris data we used here differs from the data presented in [18] in the 35th and 38th samples, which can be found in Iris data web (http://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.names) from UCI machine learning repository. Thanks to the UCI machine learning librarian, we deleted 4 duplicate samples from Liver-disorder data set, which are the 86th, the 150th, the 176th and the 318th, respectively, mentioned by Leon first. The detailed description of wine quality data set can be found in [19].

In order to demonstrate the superiority of our proposed algorithm EGKM on computational time, for each data set we implemented three algorithms: the GKM algorithm, the fast GKM algorithm and our proposed EGKM algorithm. With the data set which has a large number of attributes the PCA is implemented to obtain six-dimensional data points. Table 2 shows the clustering error MSE of these three algorithms on fourteen UCI data sets, and Table 3 displays the corresponding execution time (in seconds) of them.

For colon cancer tissues data set, we conducted experiment on the original data set and on the data set preprocessed by PCA, respectively. The experimental results are compared in table 4.

TABLE I. DESCRIPTIONS OF DATA SETS

Data sets	*Records-number*	*Attributes-number*	*Clusters-number*
Soybean-small	47	35	4
Colon-cancer	62	2000	2
Iris	150	4	3
Wine	178	13	3
SPECTF heart	267	44	2
Liver Disorders	341	6	2
Ionoshpere	351	34	2
Libras Movement	360	90	15
WDBC	569	30	2
Pima Indians Diabetes	768	8	2
Yeast	1484	8	10
Wine quality-red	1599	11	6
Image Segmentation	2310	19	7
Pendigits	3489	16	10
Wine quality-white	4898	11	7

TABLE II. THE MSE ON THE DATA SETS FROM UCI FOR THE THREE ALGORITHMS USING THE CORRECT CLUSTERS

Data sets,# methods	*The GKM*	*The fast GKM*	*Our proposed GKM*
Soybean-small	146.0985	146.0985	146.0985
Iris	78.8514	78.8557	78.8557
Wine	2.3706×10^6	2.3706×10^6	2.3706×10^6
Spect heart	5.1337×10^5	5.1337×10^5	5.1337×10^5
Liver-disorders	4.2240×10^5	4.2240×10^5	4.2240×10^5
Ionoshpere	1.3327×10^3	1.3327×10^3	1.3327×10^3
Movement-libras	210.6687	218.4084	217.7247
Wdbc	7.7942×10^7	7.7942×10^7	7.7942×10^7
Pima Indians Diabetes	5.1363×10^6	5.1363×10^6	5.1363×10^6

Yeast	37.8610	38.7657	45.1102
Wine-red	1.7731×10^5	1.7771×10^5	1.7771×10^5
Segmentation	1.3431×10^7	1.3679×10^7	1.5269×10^7
Pendigits	0.9939×10^7	1.0007×10^7	1.0553×10^7
Wine-white	1.3754×10^6	1.3772×10^6	1.2663×10^6

TABLE III. RUN TIME(S) ON THE DATA SETS FOR THE THREE ALGORITHMS USING THE CORRECT CLUSTERS

Data sets,# methods	***The GKM***	***The fast GKM***	***Our proposed GKM***
Soybean-small	0.109	0	0
Iris	0.437	0	0
Wine	0.704	0	0
SPECTF heart	0.608	0.015	0
Liver Disorders	1.201	0.016	0
Ionoshpere	0.796	0.016	0
Libras Movement	20.109	0.156	0.124
WDBC	1.176	0.016	0
Pima Indians Diabetes	4.227	0.047	0.015
Yeast	203.569	1.076	0.405
Wine quality-red	138.278	0.733	0.171
Image Segmentation	241.114	1.747	0.359
Pendigits	961.475	5.741	0.717
Wine quality-white	1500.473	8.148	1.264

TABLE IV. RESULTS ON COLON CANCER TISSUES DATA SET

methods,# Data sets	***with PCA preprocessed data***		***true data set***	
	T(s)	MSE	T(s)	MSE
GKM	0.067	1.0941×10^{10}	0.608	2.0183×10^{10}
Fast GKM	0	1.1104×10^{10}	0.047	2.0183×10^{10}
Our EGKM	0	1.5995×10^{10}	0.031	2.4974×10^{10}

From the experimental results on fourteen UCI data sets shown in table2 and table3, it can be observed that the GKM algorithm gives the best MSE results for all the data sets, but it has the heaviest computational burden. Our proposed EGKM algorithm provided the same MSE results as that of the fast GKM algorithm, and the comparable performance against the GKM algorithm by without significantly affecting the solution quality of MSE . However our GKM algorithm takes a significantly reduced computation time against GKM and fast GKM algorithms, especially on clustering large data set. Analysis of the results in table 2 and table 3, we can say that our EGKM is the fastest GKM in execution time, while with the comparable clustering error as well. Table 4 implies that our EGKM can deal with the high dimensional data set with the fastest run speed compared to GKM and fast GKM, and without influence the clustering results significantly.

In order to further prove the superiority of our proposed algorithm EGKM revealed on the execution time, some contrast experiments are implemented in the following part of the article.

We selected six data sets from the fourteen UCI data sets above. Then we executed the three algorithms GKM, FGKM and our EGKM for different values of k on the six selected data sets, and recorded the time that these algorithms consumed respectively, and compare the performance of the three clustering algorithms. The results are displayed in Figs.1-6, respectively.

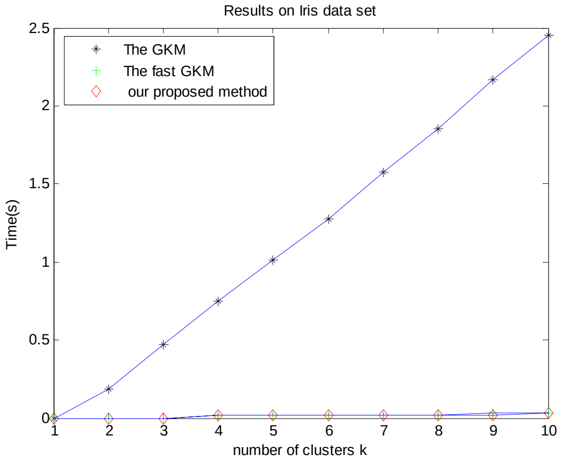

Figure 1. Run time on Iris data for different clusters

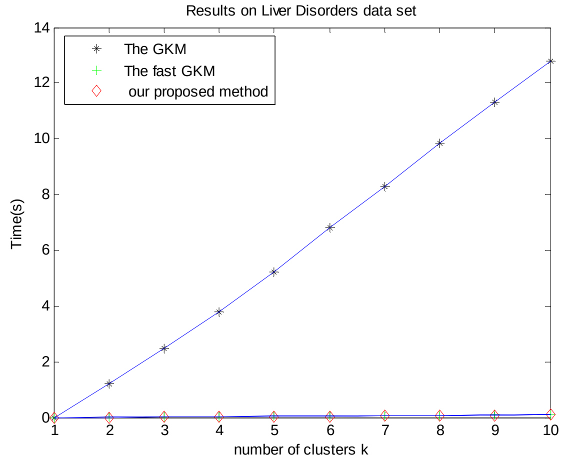

Figure 2. Run time on Liver Disorders data set for different clusters

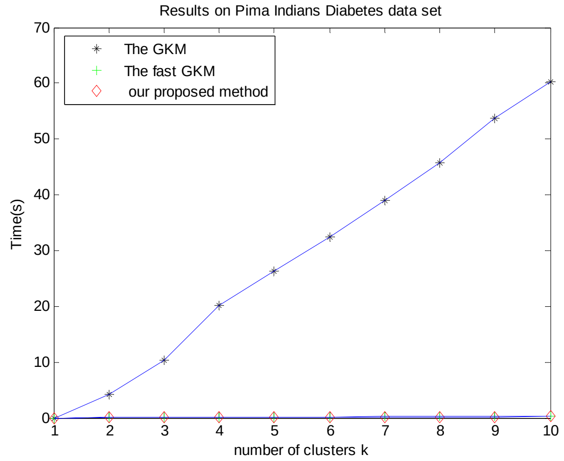

Figure 3. Run time on Pima Indians Diabetes data set for different clusters

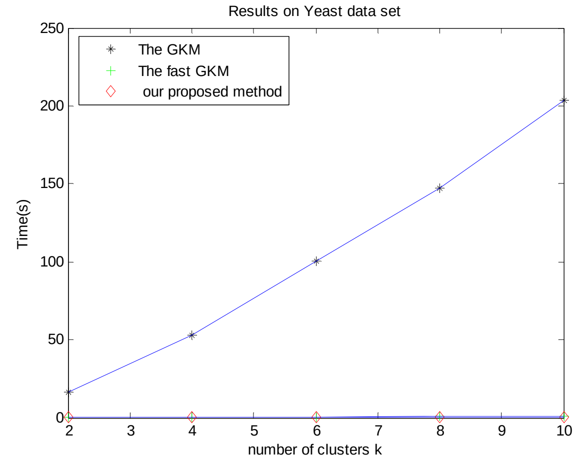

Figure 4. Run time on Yeast data set for different clusters

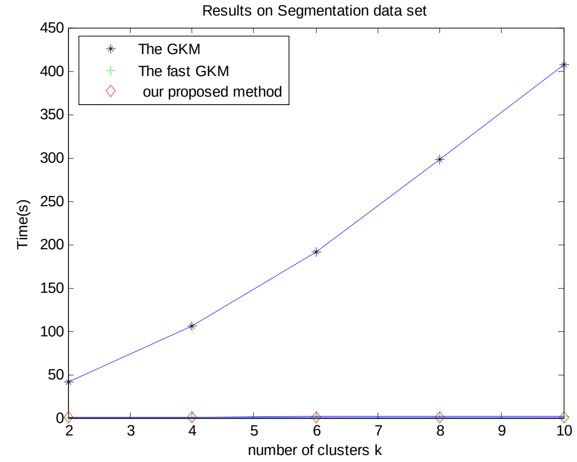

Figure 5. Run time on Segmentation data for different clusters

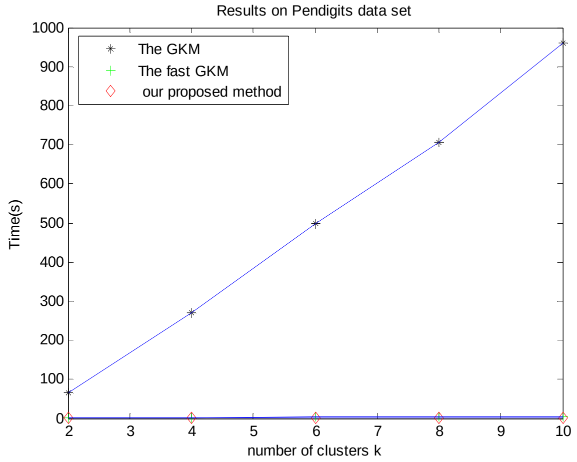

Figure 6. Run time on Pendigits data set for different clusters

From the above figures, we can see that there is a great contrast between the GKM and our proposed EGKM algorithm. It's obvious that our algorithm is much better than the GKM algorithm in computation time, and a slightly better than the fast GKM algorithm. It also can be seen that our algorithm is more suitable to cluster large data sets.

B. Experiments on sythentic data stes

In this subsection we did experiments to demonstrate our proposed algorithm EGKM can avoid the impact of noisy data. We first generated three clusters synthetic data sets with noisy data, and conducted experiments on them. The size of each data set is 120. We call the three clusters as cluster A, B and C, respectively. The parameters we used to generate each cluster data are shown in table 5. We generate x-coordinates in cluster A from normal distribution with mean $\mu_x^A = 0$ and standard deviation $\sigma^A = 1.5$, and y-coordinates from normal distribution with mean $\mu_y^A = 0$ and standard deviation $\sigma^A = 1.5$. That is to say from $N\left(\mu_x^A, \sigma^A\right)$, and $N\left(\mu_y^A, \sigma^A\right)$, respectively. In the same way, we independently generated x-coordinates and y-coordinates in cluster C from $N\left(\mu_x^C, \sigma^C\right)$ and $N\left(\mu_y^C, \sigma^C\right)$, respectively. However, in the process of generating cluster B, ten percentage samples are generated somewhat differently. We assumed that they have a larger standard deviation $\sigma_L^B = 2$. We call the larger standard deviation the abnormal deviation. That is, we generated cluster B with 10% noisy data in it. The clustering error and consumed time in seconds on the artificial data sets of GKM, fast GKM, and our EGKM are included in table 6. Figs 7~10 displayed the clustering results of GKM, fast GKM, and our EGKM on the synthetic data, respectively.

TABLE V. THE PARAMETERS OF SYNTHETIC DATA SETS WITH NOISY DATA

	cluster A	*cluster B*	*cluster C*
means	$\mu_x^A = 0, \mu_y^A = 0$	$\mu_x^B = 6, \mu_y^B = 2$	$\mu_x^C = 6, \mu_y^C = -1$
standard deviation	$\sigma^A = 1.5$	$\sigma^B = 0.5$	$\sigma^C = 0.5$
abnormal deviation		$\sigma_L^B = 2$	

TABLE VI. CLUSTERING RESULTS OF SYNTHETIC DATA STES WITH NOISY DATA FOR THE THREE CLUSTERING ALGORITHMS

	the GKM	*fast GKM*	*our EGKM*
MSE($\times 10^3$)	0.6363	0.6363	0.6363
Time(s)	1.110	0.062	0.031

From the above table 6 and the Figs 7~10, we can say that our EGKM algorithm consumed the least time in clustering procedure without influenced the clustering result. So we can conclude that our EGKM is the best GKM algorithm.

In addition, in order to further estimate the performance of our proposed EGKM algorithm more objectively, we added different proportions of noisy points to the synthetic data sets when we generated them and clustered via three clustering algorithms of GKM, fast GKM and our proposed EGKM, respectively. We employed the adjusted Rand index which is proposed by Hubert and Arabie in reference [20] to test our EGKM. The adjusted Rand index is popularly used for comparison of the clustering result when the external criterion or the true partition is known.

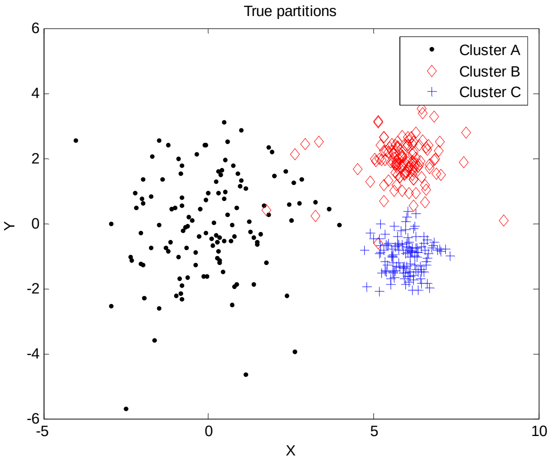

Figure 7. true partions of synthetic data

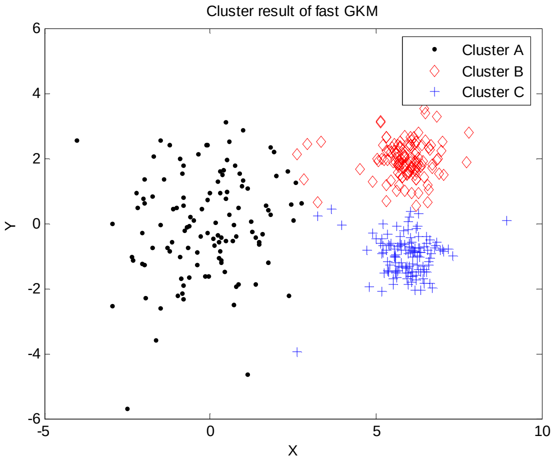

Figure 9. Clustering result of fast GKM

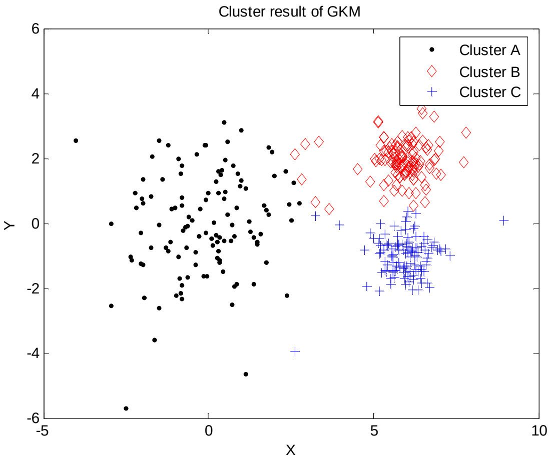

Figure 8. Clustering result of GKM

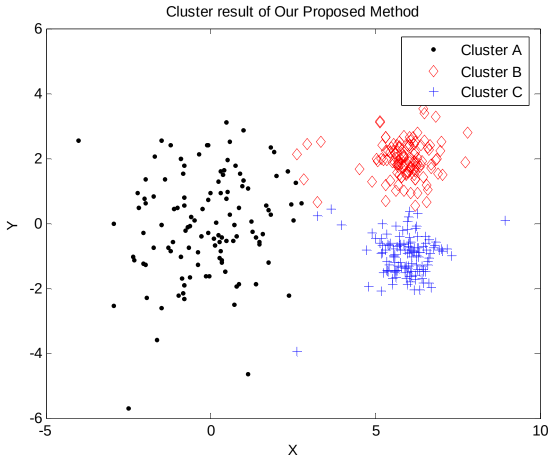

Figure 10. Clustering result of our proposed EGKM

Suppose that U and V represent two different partitions of the dataset which is under consideration, and that U is the true partition and V is a clustering results. Let $\{U(i)\}$ be the set of n cluster labels in U and $\{V(j)\}$ be the set of n cluster labels in V . The numbers of $\{ a, b, c, d \}$ are defined as cardinalities of the sets shown.

$$a = \left|\{(i,j) : i > j, U(i) = U(j), V(i) = V(j)\}\right|$$

$$b = \left|\{(i,j) : i > j, U(i) = U(j), V(i) \neq V(j)\}\right|$$

$$c = \left|\{(i,j) : i > j, U(i) \neq U(j), V(i) = V(j)\}\right|$$

$$d = \left|\{(i,j) : i > j, U(i) \neq U(j), V(i) \neq V(j)\}\right|$$

Thus, a is the number of pairs of data points that are placed in the same class in U and in the same cluster in V, b is the number of pairs of data points that are placed in the same class in U but not in the same cluster in V, c is the number of pairs of data points that are placed in the same cluster in V but not in the same class in U, and d is the number of pairs in different class in U and different cluster in V. Then the adjusted Rand index for the clustering result V is calculated by the equation (10).

$$RI_{adj} = \frac{2(ad - bc)}{(a+b)(b+d) + (a+c)(c+d)} \qquad (10)$$

Table 7 shows the calculated adjusted Rand index of our proposed EGKM compared to the traditional K-means, the GKM and the fast GKM clustering algorithms according to a different proportion of noisy samples contained.

It can be clearly seen from Table 7 that our proposed EGKM performs much better than the traditional K-means clustering algorithm, and has the same performance with GKM and fast GKM clustering algorithms. Meanwhile, it shows that our EGKM algorithm and the GKM and the fast GKM clustering algorithms have the same advantages in avoiding the impact of the noisy points on clustering results.

Finally, we compared the clustering error MSE of three clustering algorithms of GKM, fast GKM and our proposed EGKM on the artificial data with different proportions of noisy points in Table 8. Table 9 displayed the comparison of the run time of the three algorithms on the same synthetic data sets with different proportions of noisy data.

TABLE VII. THE ADJUSTED RAND INDEX OF DIFFERENT ALGORITHMS

% noisy objects	***K-means***	***The GKM***	***The fast GKM***	***our proposed EGKM***
0	0.7903	0.9917	0.9917	0.9917
5	0.8376	0.9418	0.9418	0.9418
10	0.7836	0.9427	0.9427	0.9427
15	0.7957	0.9255	0.9255	0.9255
20	0.7305	0.9502	0.9502	0.9502
25	0.7708	0.9192	0.9192	0.9192
30	0.7750	0.9263	0.9263	0.9263
35	0.7595	0.9179	0.9179	0.9179
40	0.7624	0.8943	0.8943	0.8943

TABLE VIII. THE MSE OF THE SYNTHETIC DATA SETS WITH DIFFERENT PROPOTIONS OF NOISY DATA OF THREE ALGORITHMS

% noisy objects	***GKM***	***Fast GKM***	***Our proposed EGKM***
0%	582.2442	582.2442	582.2442
5%	605.0423	605.0423	605.0423
10%	746.0904	746.1037	746.1037
15%	665.3958	665.3958	665.3958
20%	777.3195	777.3195	777.4367
25%	735.4189	735.4189	735.4189
30%	879.4830	879.4830	879.4830
35%	825.2338	825.2328	825.2328
40%	978.2652	978.7659	978.7659

TABLE IX. THE RUN TIME OF THE THREE ALGORITHMS ON THE SYNTHETIC DATA SETS WITH DIFFERENT PROPORTIONS NOISY DATA

% noisy objects	***GKM***	***Fast GKM***	***Our proposed EGKM***
0%	1.157	0.015	0.016
5%	1.103	0.015	0.015
10%	1.188	0.031	0.016
15%	1.125	0.016	0.016
20%	1.25	0.016	0.016
25%	1.281	0.015	0.015
30%	1.5	0.031	0.016
35%	1.39	0.016	0.015
40%	1.391	0.015	0.015

From table 8 we can see that the GKM, fast GKM and our proposed EGKM nearly have the same performance in clustering data set with noisy data points in it. While Table 9 implied that our proposed EGKM consumed the least time when clustering data set with noisy data among the three algorithms.

V. CONCLUSIONS

In this paper we presented an efficient global K-means clustering algorithm, called EGKM for short. It is known that GKM algorithm constitutes a deterministic clustering method providing excellent results in terms of the mean square clustering error criterion. It does not depend on any initial conditions or parameter values by employing the standard K-means algorithm as a local search procedure. Its outstanding feature is that it proceeds in an incremental way attempting to optimally add one new cluster center at each stage, but which also caused its heavy computational load. The most important step in GKM algorithm is to determine the initial center for the next new cluster center at each stage. Our new version of GKM algorithm reduced its heavy computational load. The main amelioration that we made is the way to select the optimal initial center for the next new cluster at each stage. We defined f_i for each point and chose the one which has the minimum value of f_i as the optimal initial center for the next new cluster at each stage. The most advantage of our propose EGKM algorithm is that it can reduce the computation load greatly. Experiments on fourteen data sets from UCI machine repository show that our variations of the GKM algorithm EGKM outperforms the GKM and fast GKM algorithm in execution time without significantly affecting solution quality, especially on large data sets. Further experiments on colon cancer tissue data set revealed that our EGKM can also efficiently deal with the high dimensional data. Finally we conducted experiments on synthetic data sets with noisy data, and the experiment results imply that our EGKM can avoid the influence of noisy data on clustering result. Further analysis via adjusted Rand index on the artificial data set with different proportions of noisy data points demonstrated the well performance of our EGKM.

Consequently, our proposed algorithm EGKM is more suitable for clustering of large data sets, and outperformed the GKM and fast GKM without significantly influenced the clustering result. At the same time, our EGKM has the strong ability to cluster the data sets with noisy data efficiently, and can cluster the gene expression data set with high dimensions consuming the least time among the three algorithms of the GKM, fast GKM, and our proposed EGKM.

ACKNOWLEDGMENT

The authors wish to thank the helpful comments and suggestions from anonymous reviewers. The authors are grateful to the authors who provide the useful benchmark data sets. We would also like to thank Aristidis Likas, Nikos Vlassib, and Jakob J. Verbeek who made their source Matlab code of the fast global k-means can be downloaded from http://carol.wins.uva.nl/~jverbeek/software/. Our implementation grew from the code. This work was supported in part by the grant of the Fundamental Research Funds for the Central Universities of GK200901006, and was also supported by Natural Science Basis Research Plan in Shaanxi Province of P. R. China (Program No. 2010JM3004).

REFERENCES

[1] M. N. Murty and A. K. Jain, “Data clustering: a review,” ACM Computing Surveys, vol. 31, 1999, pp. 264–323.

[2] B. Everitt, S. Landau, and M. Leese, Cluster Analysis, Arnold, London, 2001.

[3] S. THeodoridis and K. Koutroumbas, Pattern Recognition, 2nd ed., Academic Press, 2003.

[4] T. Kanungo and D. Mount, “An efficient k-means clustering algorithm: analysis and implantation,” IEEE Trans, PAMI, vol. 24, pp. 881–892, 2004.

[5] J. M. Pena, J. A. Lozano, and P. Larranaga, “An empirical comparison of four initialization methods for the k-means

algorithm," Pattern Recognition Letters, vol. 20, pp. 1027–1040, 1999.

[6] P. S. Bradley and U. M. Fayyad, "Refining initial points for k-means clustering," Proceedings of the Fifteenth International Conference on Machine Learning, Morgan Kaufmann Publishers Inc. San Francisco, CA, USA, 1998, pp. 91–99.

[7] Z. Huang, "Clustering large data sets with mixed numerical and categorical value," in: Proceedings of the First Pacific Asia Knowledge Discovery and Data Mining Conference.World Acientific, Singapore, 1997, pp. 21–34.

[8] Y. Sun, Q. M. Zhu and Z. X. Chen, "An iterative initial-points refinement algorithm for categorical data clustering," Pattern Recognition Letters, vol. 23, pp. 875–884, 2002.

[9] A. Strehl and J. Ghosh, "Cluster ensembles – a knowledge ruse framework for combining multiple partitions," Journal of machine Learning Reserch, vol. 3, pp. 583-617, 2002.

[10] A. Likas, M. Vlassis, and J. Verbeek, "The global k-means clustering algorithm," Pattern Recognition, vol. 36, pp. 451–461, 2003.

[11] S. S. Khan and A. Ahmad, "Cluster center initialization algorithm for k-means clustering," Pattern Recognition Letters, vol. 25, pp. 1293–1302, 2004.

[12] S. J. Redmond and C. Heneghan, "A method for initializing the K-means clustering algorithm using kd-trees," Pattern Recognition Letters, vol. 28, pp. 965–973, 2007.

[13] A. M. Bagirov, "Modified global k-means algorithm for minimum sum-of-squares clustering problems," Pattern Recognition, vol. 41, pp. 3192–3199, 2008.

[14] H. S. Park and C. H. Jun, "A simple and fast algorithm for K-medoids clustering," Expert Systems with Applications, vol. 36, pp. 3336–3341, 2009.

[15] UCI Machine Learning Repository. http://archiv.ic.uci.edu/ml/

[16] Princeton University gene expression project. http://genomics-pubs.princeton.edu/oncology/

[17] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine, "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon cancer tissues probed by oligonucleotide arrays," PNAS, vol. 96, pp. 6745-6750, June 1999, Cell Biology.

[18] R. A. Fisher, "The use of multiple measurments in taxonomic problems", Annual Eugenics, vol. 7, part Ⅱ, pp. 179-188, 1936.

[19] P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis, "Modeling wine preferences by data mining from physicochemical properties," Decision Support Systems, Elsevier, vol. 47, pp. 547-553. ISSN:0167-9236.

[20] L. J. Hubert and P. Arabie, "Comparing partitions," Journal of Classification, vol. 2, pp. 193–218, 1985.

Juanying Xie was born in Xi'an City of P. R. China on April 15, 1971. She received the BSc degree in computer science from Shaanxi Normal University, China, in 1993, and MSc degree in computer applied technology from Xidian University, China, in 2004. She is now a Ph. D. candidate in signal and information processing of School of Electronics Engineering of Xidian University, China.

She was an Assistant Lecturer in the Department of Computer Science of Shaanxi Normal University, China, from 1993 to1999. From 1999 to 2004, she was a Lecturer in the School of Computer Science of Shaanxi Normal University, China. She has been an Associate Professor in the School of Computer Science of Shaanxi Normal University, Xi'an, China, since 2004. Her research interests are machine learning, computational intelligence, pattern recognition and data mining.

Shuai Jiang was born in Shenyang of P. R. China on Oct. 3, 1982. She received her BSc and MSc degrees in computer science from Shaanxi Normal University, China, in 1997 and 2010, respectively.

Weixin Xie was born in Guangzhou city, P. R. China, in Dec. 1941. He received the BSc degree in signal and information processing from Xidian University, China, in 1965. He was awarded as a professor of Xidian University in 1986. He has been awarded as a doctoral supervisor by the committee of degree of P. R. China since 1990. He was the vice-president and the director of postgraduate college of Xidian University, China, from 1992 to 1996. He was the president of Shenzhen University, China, from 1996 to 2005. He is the Director of the National Laboratory of Automatic Target Recognition, Shenzhen University, China. His research interests focus on intelligent information processing, fuzzy information processing, etc. He is now the primary editor of the Chinese journal of signal processing, and the vice editor of Chinese of Journal Electronics (English version), and on the editorial boards of journals of Science in China (Series F), etc.

Xinbo Gao received the BSc, MSc and PhD degrees in signal and information processing from Xidian University, China, in 1994, 1997 and 1999 respectively. He is a Professor of Pattern Recognition and Intelligent System, and the Director of the VIPS Lab, Xidian University. His research interests are computational intelligence, machine learning, etc. He is on the editorial boards of journals of EURASIP Signal Processing (Elsevier) etc. He served as general chair/co-chair or program committee chair/co-chair for around 30 major international conferences.