Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Fast global kernel fuzzy c-means clustering algorithm for consonant/vowel segmentation of speech signal*

Xian ZANG[1], Felipe P. VISTA IV[1], Kil To CHONG‡[1,2]

([1]*Department of Electronic Engineering, Jeonbuk National University, Jeonju-si 561-756, Korea*)

([2]*Advanced Electronics and Information Research Center, Jeonbuk National University, Jeonju-si 561-756, Korea*)

E-mail: zangxian@jbnu.ac.kr; boduke@jbnu.ac.kr; kitchong@jbnu.ac.kr

Received Nov. 2, 2013; Revision accepted Apr. 20, 2014; Crosschecked June 19, 2014

Abstract: We propose a novel clustering algorithm using fast global kernel fuzzy c-means-F (FGKFCM-F), where F refers to kernelized feature space. This algorithm proceeds in an incremental way to derive the near-optimal solution by solving all intermediate problems using kernel-based fuzzy c-means-F (KFCM-F) as a local search procedure. Due to the incremental nature and the nonlinear properties inherited from KFCM-F, this algorithm overcomes the two shortcomings of fuzzy c-means (FCM): sensitivity to initialization and inability to use nonlinear separable data. An accelerating scheme is developed to reduce the computational complexity without significantly affecting the solution quality. Experiments are carried out to test the proposed algorithm on a nonlinear artificial dataset and a real-world dataset of speech signals for consonant/vowel segmentation. Simulation results demonstrate the effectiveness of the proposed algorithm in improving clustering performance on both types of datasets.

Key words: Fuzzy c-means clustering, Kernel method, Global optimization, Consonant/vowel segmentation
doi:10.1631/jzus.C1300320 **Document code:** A **CLC number:** TN912; TP391.4

1 Introduction

Clustering (Jain *et al.*, 1999; Xu and Wunsch, 2005), also known as cluster analysis, has been addressed in many contexts and disciplines such as data mining, document retrieval, image segmentation, and pattern classification. The aim of the clustering method is to partition unlabeled patterns into a certain number of clusters (groups) on the basis of the similarity criterion; i.e., patterns in the same cluster should be similar to each other while patterns in different clusters should not. The most popular similarity measure is the Euclidean distance (Duda and Hart, 1973). Two main approaches are taken to clustering: crisp clustering (or hard clustering) and fuzzy clustering. A characteristic of the hard clustering method is that the boundary between clusters is fully defined, which means it either assigns a pattern to one cluster or not. In many practical applications, however, certain input patterns might not just belong to a single class but partially belong to other classes. Fuzzy clustering is more natural than hard clustering in such cases since it allows pattern(s) to belong to multiple clusters simultaneously with different degrees of membership (Balasko *et al.*, 2005). In fuzzy clustering, the fuzzy c-means (FCM) algorithm (Bezdek, 1981) plays an important role in unsupervised data analysis. This clustering is achieved by iteratively minimizing an objective function that is dependent on the distance of the pattern to the cluster prototypes in the input space.

Several applications of fuzzy clustering based approaches have shown popularity even up to the present time (Tsai and Lin, 2011; Bozkir and Sezer, 2013; Hu *et al.*, 2013; Nguyen and Wu, 2013; Zhao, 2013; Gong *et al.*, 2014), but FCM has two well-known limitations: (1) sensitivity to initialization and

‡ Corresponding author

* Project supported by the National Research Foundation (NRF) of Korea (Nos. 2013009458 and 2013068127)

(2) inability to use nonlinear separable data. A simple and very common solution to the sensitivity to initialization is the use of multiple restarts wherein the centers of the clusters are randomly placed at different initial positions to find a better local minimum. This technique, however, depends heavily on the number of restarts with a very high computational cost, but with no certainty whether the initializations attempted are sufficient to obtain a near-optimal minimum. To deal with this problem, Wang *et al*. (2006) proposed the global FCM (GFCM) clustering algorithm to search for a near-optimal solution in an incremental way with no dependency on the initial conditions. Recently kernel methods emerged as an important tool to convert linear methods to nonlinear methods. They were also applied to unsupervised clustering to alleviate the second limitation by mapping data points from the input space to a potentially higher dimensional feature space through an implicit nonlinear transformation, to realize linear separation in the kernelized feature space and then to obtain the nonlinear separated clusters when returning to the original input space. There are two major variations of kernel-based FCM (KFCM) clustering: KFCM-I, which involves keeping prototypes in the input space (Wu *et al.*, 2003; Zhang and Chen, 2003a; 2003b; 2004; Shen *et al.*, 2006; Yu *et al.*, 2011), and KFCM-F, which implicitly leaves the prototypes in the higher dimensional feature space (Li *et al.*, 2001; Girolami, 2002; Zhang and Chen, 2002; Chiang and Hao, 2003; Zhou and Gan, 2004; Kim *et al.*, 2005). Although KFCM-F lacks clear and intuitive description of prototypes in the original input space, it performs better than KFCM-I in terms of analyzing nonlinear separable data.

The global kernel fuzzy c-means-F (GKFCM-F) clustering algorithm is proposed to optimize the clustering performance. The algorithm works in an incremental way by solving all intermediate problems with 1, 2, …, C clusters using KFCM-F as a local search procedure. The near-optimal solution for each intermediate case of k-partition can be obtained by starting with a deterministic initial state composed of the optimal solution to the previous $(k-1)$-partition stage and a single data point, and then selecting the best result with the lowest clustering error after performing KFCM-F on each initial state. The best solution with C clusters can finally be obtained by applying the above procedure iteratively. This algorithm integrates the advantages of GFCM and KFCM-F, which overcomes both limitations of FCM. However, N initializations are attempted during each local search, where N is the dataset size. The dataset is usually very large, and thus the algorithm will have high computational complexity. We propose an accelerating scheme, called fast global KFCM-F (FGKFCM-F), to compensate for such a drawback. The basic idea for this scheme is that we first choose the best set that guarantees the greatest reduction in the objective function at each intermediate step, instead of carrying out the KFCM-F algorithm for all sets of the initial state formed by selecting different data points. Then KFCM-F is executed only once from this initialization. The computation complexity is now independent of the dataset size and the running time of the algorithm is reduced.

2 Fuzzy c-means (FCM)

Let $X=\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$ be a finite unlabeled dataset composed of N patterns for which every $\boldsymbol{x}_i=[x_{i1}, x_{i2}, \ldots, x_{id}]^{\mathrm{T}} \in \mathbb{R}^d$. The well-known FCM clustering algorithm proposed by Dunn (1973) and Bezdek (1981) allows a pattern to belong to more than one cluster. Clustering is thus achieved by iteratively minimizing an objective function of Eq. (1), which is dependent on the distance of the pattern to the cluster prototypes in the input space.

$$J(\boldsymbol{\mu}, \boldsymbol{v}) = \sum_{i=1}^{N} \sum_{k=1}^{C} \mu_{ki}^{m} \left\| \boldsymbol{x}_i - \boldsymbol{v}_k \right\|^2, \tag{1}$$

where $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k, \ldots, \boldsymbol{v}_C\}$ $(\boldsymbol{v}_k \in \mathbb{R}^d)$ denotes the C cluster prototypes of dataset X, m is the weighting exponent determining the degree of fuzziness of the resulting classification ($m=2$ is often adopted), and μ_{ki} represents the membership coefficient of the ith pattern in the kth cluster, which satisfies the following constraint as introduced in fuzzy set theory (Zadeh, 1965):

$$\mu_{ki} \in [0,1] \left| \sum_{k=1}^{C} \mu_{ki} = 1, \forall i \text{ and } 0 < \sum_{i=1}^{N} \mu_{ki} < N, \forall k, \right. \tag{2}$$

and $\{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k, \ldots, \boldsymbol{\mu}_C\}$ ($\boldsymbol{\mu}_k=[\mu_{k1}, \mu_{k2}, \ldots, \mu_{kN}]^{\mathrm{T}}$) realizes fuzzy C-clustering of X.

Subject to the membership constraint, the minimization of objective function (1) is performed by introducing a Lagrangian function (Filippone *et al.*, 2008; Liu *et al.*, 2012) for each pattern:

$$\overline{J}(\boldsymbol{\mu},\boldsymbol{v},\lambda)=\sum_{i=1}^{N}\sum_{k=1}^{C}\mu_{ki}^{m}\left\|\boldsymbol{x}_i-\boldsymbol{v}_k\right\|^2+\sum_{i=1}^{N}\lambda_i\left(\sum_{k=1}^{C}\mu_{ki}-1\right), \tag{3}$$

where λ is the Lagrangian multiplier to connect the objective function to the constraint.

Let the partial derivative of $\overline{J}(\boldsymbol{\mu},\boldsymbol{v},\lambda)$ with respect to μ_{ki} and $\boldsymbol{v}_i$ be equal to zero. We then obtain two necessary but not sufficient conditions for $J(\boldsymbol{\mu}, \boldsymbol{v})$ to be at its local minima:

$$\mu_{ki}=\left\|\boldsymbol{x}_i-\boldsymbol{v}_k\right\|^{\frac{-2}{m-1}}\Big/\sum_{j=1}^{C}\left\|\boldsymbol{x}_i-\boldsymbol{v}_j\right\|^{\frac{-2}{m-1}}, \tag{4}$$

$$\boldsymbol{v}_k=\sum_{i=1}^{N}\mu_{ki}^{m}\boldsymbol{x}_i\Big/\sum_{i=1}^{N}\mu_{ki}^{m}, \tag{5}$$

where i=1, 2, … , N, j=1, 2, …, C, and k=1, 2, …, C. The detailed derivation of the two necessary conditions of Eqs. (4) and (5) using the Lagrangian multiplier method is shown in the Appendix.

Starting with initialized cluster prototypes and iteratively updating the necessary conditions to minimize $J(\boldsymbol{\mu}, \boldsymbol{v})$, the FCM algorithm converges to a local minimum of the objective function. Convergence can be detected when the changes in the membership function or the prototypes at two successive iteration steps satisfy the stopping criterion ε (ε=0.00001 is adopted here). A soft partitioning of the input space is then obtained. Algorithm 1 summarizes the description of the FCM algorithm.

Algorithm 1 FCM clustering

Input:

(1) $X=\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_i \in \mathbb{R}^d$, the dataset;

(2) C, $1<C\leq N$, the number of clusters;

(3) $\varepsilon>0$, the stopping criterion;

(4) $\boldsymbol{v}^{(0)}=(\boldsymbol{v}_1^{(0)}, \boldsymbol{v}_2^{(0)}, \ldots, \boldsymbol{v}_C^{(0)})$, the initials of cluster prototypes;

(5) $m>1$, the weighting exponent.

Output:

(1) $\boldsymbol{v}=(\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_C)$, the final cluster prototypes;

(2) $\boldsymbol{\mu}=(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_C)$, the final memberships.

1 s=1;

2 Update $\boldsymbol{\mu}^{(s)}=(\boldsymbol{\mu}_1^{(s)}, \boldsymbol{\mu}_2^{(s)}, \ldots, \boldsymbol{\mu}_C^{(s)})$ with $\boldsymbol{v}^{(s-1)}$ using Eq. (4);

3 Update $\boldsymbol{v}^{(s)}$ with $\boldsymbol{\mu}^{(s)}$ using Eq. (5);

4 Compare $\boldsymbol{v}^{(s)}$ with $\boldsymbol{v}^{(s-1)}$ in a convenient matrix norm $\|\cdot\|$:

If $\|\boldsymbol{v}^{(s)}-\boldsymbol{v}^{(s-1)}\|<\varepsilon$, STOP and OUTPUT;

Else $s=s+1$ and return to line 2.

3 Global fuzzy c-means (GFCM)

Although FCM clustering is a well-known method, it is sensitive to the initial position of the cluster prototypes and easy to revert to a local minimum or a saddle point when iterating. A simple and very popular solution is the use of multiple restarts, wherein the prototypes are randomly placed at different initial positions so that a better local minimum can be found. This technique, however, depends heavily on the number of restarts with a very high computational cost with no certainty if the initializations attempted are sufficient to obtain a near-optimal minimum. To deal with this problem, Wang *et al.* (2006) proposed the GFCM based on the assumption that the optimal clustering solution to the k-clustering problem can be obtained through N local searches starting from an initial state with the $k-1$ prototypes placed at the optimal positions for the $(k-1)$-clustering problem, and the remaining kth prototype is then placed at a data point $\boldsymbol{x}_i$ ($1\leq i\leq N$) (Likas *et al.*, 2003; Bagirov, 2008).

The proposed technique proceeds in an incremental way to solve the C-clustering problem by solving all intermediate problems with 1, 2, …, C clusters using FCM as a local search procedure. More specifically, the optimal position $\boldsymbol{v}^*(1)$ corresponds to the centroid of dataset X of fuzzy 1-partition. For the fuzzy 2-partition problem, the first initial cluster prototype is placed at the optimal position for fuzzy 1-partition while the second initial cluster prototype is placed at a data point $\boldsymbol{x}_i$ ($1\leq i\leq N$). We then perform the FCM algorithm for each initial state $\{\boldsymbol{v}^*(1), \boldsymbol{x}_i\}$ and select the best solution for this case, expressed as $\{\boldsymbol{v}_1^*(1), \boldsymbol{v}_2^*(2)\}$. In general, let $\{\boldsymbol{v}_1^*(k-1), \boldsymbol{v}_2^*(k-1), ..., \boldsymbol{v}_{k-1}^*(k-1)\}$ denote the optimal solution to fuzzy $(k-1)$-partition. We then perform the FCM algorithm with k clusters from each initial state $\{\boldsymbol{v}_1^*(k-1),$

$\boldsymbol{v}_2^*(k-1), ..., \boldsymbol{v}_{k-1}^*(k-1), \boldsymbol{x}_i\}$. The best result obtained from the N runs is considered as the optimal solution $\{\boldsymbol{v}_1^*(k), \boldsymbol{v}_2^*(k), ..., \boldsymbol{v}_k^*(k)\}$ for the fuzzy k-partition problem. By proceeding in the above manner, the best solution with C clusters could be finally obtained by solving all intermediate problems.

The main advantage of the GFCM algorithm is that the solution with C clusters is built deterministically, so there is no dependency on the initial conditions and near-optimal solutions could be obtained. The algorithm is briefly summarized in Algorithm 2, where FCM refers to the FCM algorithm described in Section 2.

Algorithm 2 GFCM clustering

Input:

(1) $X=\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_i \in \mathbb{R}^d$, the dataset;

(2) C, $1<C\leq N$, the number of clusters;

(3) $\varepsilon>0$, the stopping criterion;

(4) $m>1$, the weighting exponent.

Output:

(1) $\boldsymbol{v}=(\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_C)$, the final cluster prototypes;

(2) $\boldsymbol{\mu}=(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_C)$, the final memberships.

1 $\boldsymbol{v}^*(1)=\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{x}_i$;

2 **for** k=2 to C **do**

3 **for** i=1 to N **do**

4 set the initial state

$\boldsymbol{v}^{(0)}(k)=(\boldsymbol{v}_1^*(k-1), \boldsymbol{v}_2^*(k-1), \ldots, \boldsymbol{v}_{k-1}^*(k-1), \boldsymbol{x}_i)$;

5 $[\boldsymbol{v}^*(k), \boldsymbol{\mu}(k)]\leftarrow$FCM($X$, k, ε, m, $\boldsymbol{v}^{(0)}(k)$);

6 **end**

7 **end**

8 $\boldsymbol{v}\leftarrow\boldsymbol{v}^*(C)$, $\boldsymbol{u}\leftarrow\boldsymbol{u}(C)$.

A comparison of FCM and GFCM is conducted on the 2D artificial dataset (C=5). Fig. 1a shows the clustering result using FCM. The performance of FCM heavily depends on the initialization. Even in the dataset with five well separated clusters, FCM sometimes fails to find the correct structure due to the convergence to a local optimum. Fig. 1b shows the clustering result using GFCM for the same dataset. Without need for initialization, the data points are correctly grouped into five clusters. Fig. 2 illustrates the optimal cluster prototypes obtained at each intermediate step. Using the optimal results from the previous stage for initialization provides a deterministic way to find a new cluster prototype.

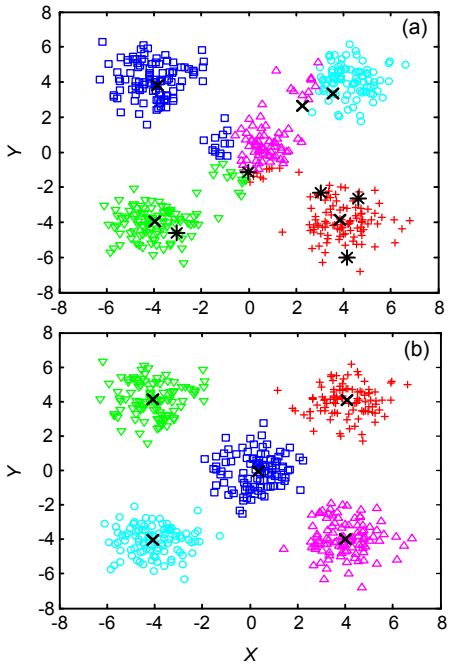

Fig. 1 Clustering results for a 2D artificial dataset
(a) FCM with random initialization; (b) GFCM without initialization. *: random initial position of the cluster prototypes; ×: final cluster prototypes

4 Kernel-based fuzzy c-means (KFCM)

The kernel method was first proposed by Mercer (1909). The essence of kernel-based methods is to transform the original input space $\mathbb{R}^d$ into a higher dimensional feature space F through arbitrary nonlinear mapping Φ such that

$$\Phi: \mathbb{R}^d \to F \quad \boldsymbol{x} \to \Phi(\boldsymbol{x}). \tag{6}$$

The feature space could possibly be of infinite dimensionality. The rationale for higher dimensions is that the nonlinear separable data in the original space may be linearly separated in the feature space (Cover, 1965). The kernel method takes advantage of the fact that the dot products in the feature space can be expressed by a Mercer kernel K given by

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j)=\Phi(\boldsymbol{x}_i)\Phi(\boldsymbol{x}_j)=\Phi(\boldsymbol{x}_i)^{\mathrm{T}}\Phi(\boldsymbol{x}_j), \tag{7}$$

with $K(\boldsymbol{x}_i, \boldsymbol{x}_j)=K(\boldsymbol{x}_j, \boldsymbol{x}_i)$. By employing a specific Mercer kernel, the Euclidean distance in the feature

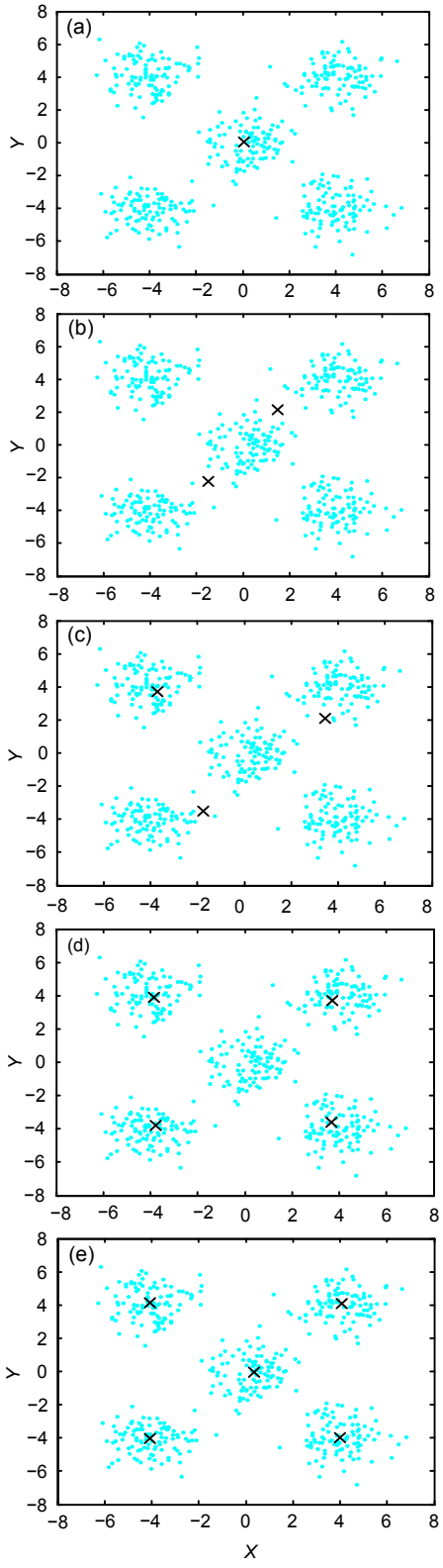

Fig. 2 Optimal cluster prototypes obtained at each intermediate step in GFCM
(a) K=1; (b) K=2; (c) K=3; (d) K=4; (e) K=5

space could be computed without explicit knowledge of Φ:

$$\begin{aligned}&\| \Phi(\boldsymbol{x}_i)-\Phi(\boldsymbol{x}_j)\|^2\\&=[\Phi(\boldsymbol{x}_i)-\Phi(\boldsymbol{x}_j)][\Phi(\boldsymbol{x}_i)-\Phi(\boldsymbol{x}_j)]\\&=\Phi(\boldsymbol{x}_i)\Phi(\boldsymbol{x}_i)+\Phi(\boldsymbol{x}_j)\Phi(\boldsymbol{x}_j)-2\Phi(\boldsymbol{x}_i)\Phi(\boldsymbol{x}_j)\\&=K(\boldsymbol{x}_i,\boldsymbol{x}_i)+K(\boldsymbol{x}_j,\boldsymbol{x}_j)-2K(\boldsymbol{x}_i,\boldsymbol{x}_j).\end{aligned} \tag{8}$$

Two major forms of KFCM clustering are used. The first form, KFCM-I (I represents the input space), comes with prototypes constructed in the input space. In the second form, KFCM-F (F refers to the feature space), the prototypes are retained in the feature space and thus the prototypes must be approximated in the input space by computing an inverse map from the feature space to the input space.

4.1 KFCM-I algorithm

KFCM-I minimizes the following objective function subject to the same membership constraint as FCM given in Eq. (2):

$$J^{\Phi}(\boldsymbol{\mu},\boldsymbol{v})=\sum_{i=1}^{N}\sum_{k=1}^{C}\mu_{ki}^{m}\left\|\Phi(\boldsymbol{x}_i)-\Phi(\boldsymbol{v}_k)\right\|^2. \tag{9}$$

The advantage of the KFCM-I clustering algorithm is that the prototypes reside in the input space and are mapped to the feature space implicitly. Thus, we could apply the distance kernel trick to obtain

$$\| \Phi(\boldsymbol{x}_i)-\Phi(\boldsymbol{v}_k)\|^2=K(\boldsymbol{x}_i,\boldsymbol{v}_k)+K(\boldsymbol{v}_k,\boldsymbol{v}_k)-2K(\boldsymbol{x}_i,\boldsymbol{v}_k). \tag{10}$$

Among the different kinds of kernel functions, the Gaussian radial basis function (GRBF) kernel (Muller *et al.*, 2001)

$$K(\boldsymbol{x},\boldsymbol{y})=\exp(-\|\boldsymbol{x}-\boldsymbol{y}\|^2/\sigma^2) \tag{11}$$

(σ as the adjustable parameter) is used almost exclusively since $K(\boldsymbol{x},\boldsymbol{x})$=1 and the derivative of $J_m^{\Phi}(\boldsymbol{\mu},\boldsymbol{v})$ with respect to $\boldsymbol{v}_k$ allows a kernel trick shown as

$$\frac{\partial K(\boldsymbol{x}_i,\boldsymbol{v}_k)}{\partial \boldsymbol{v}_k}=\frac{2(\boldsymbol{x}_i-\boldsymbol{v}_k)}{\sigma^2}K(\boldsymbol{x}_i,\boldsymbol{v}_k). \tag{12}$$

Thus, the objective (9) is simplified to

$$J^{\Phi}(\boldsymbol{\mu},\boldsymbol{v})=2\sum_{i=1}^{N}\sum_{k=1}^{C}\mu_{ki}^{m}[1-K(\boldsymbol{x}_i,\boldsymbol{v}_k)]. \tag{13}$$

In a similar way to the standard FCM algorithm, using the Lagrangian multiplier method and then zeroing the first derivatives with respect to μ_{ki} and $\boldsymbol{v}_k$ yield the update functions for memberships and cluster prototypes as follows:

$$\mu_{ki}^{-1}=\sum_{j=1}^{C}\left[\frac{1-K(\boldsymbol{x}_i,\boldsymbol{v}_k)}{1-K(\boldsymbol{x}_i,\boldsymbol{v}_j)}\right]^{\frac{1}{m-1}}, \tag{14}$$

$$\boldsymbol{v}_k=\sum_{i=1}^{N}\mu_{ki}^{m}K(\boldsymbol{x}_i,\boldsymbol{v}_k)\boldsymbol{x}_i\Big/\sum_{i=1}^{N}\mu_{ki}^{m}K(\boldsymbol{x}_i,\boldsymbol{v}_k). \tag{15}$$

The algorithm for KFCM-I is detailed as follows:

Algorithm 3 KFCM-I clustering

Input:

(1) $X=\{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_i,\ldots,\boldsymbol{x}_N\}$, $\boldsymbol{x}_i\in\mathbb{R}^d$, the dataset;

(2) C, $1<C\leq N$, the number of clusters;

(3) $\varepsilon>0$, the stopping criterion;

(4) $\boldsymbol{v}^{(0)}=(\boldsymbol{v}_1^{(0)},\boldsymbol{v}_2^{(0)},\ldots,\boldsymbol{v}_C^{(0)})$, the initials of cluster prototypes;

(5) $m>1$, the weighting exponent;

(6) σ, the GRBF kernel parameter.

Output:

(1) $\boldsymbol{v}=(\boldsymbol{v}_1,\boldsymbol{v}_2,\ldots,\boldsymbol{v}_C)$, the final cluster prototypes;

(2) $\boldsymbol{\mu}=(\boldsymbol{\mu}_1,\boldsymbol{\mu}_2,\ldots,\boldsymbol{\mu}_C)$, the final memberships.

1 $s=1$;

2 Update $\boldsymbol{\mu}^{(s)}=(\boldsymbol{\mu}_1^{(s)},\boldsymbol{\mu}_2^{(s)},\ldots,\boldsymbol{\mu}_C^{(s)})$ with $\boldsymbol{v}^{(s-1)}$ using Eq. (14);

3 Update $\boldsymbol{v}^{(s)}$ with $\boldsymbol{\mu}^{(s)}$ using Eq. (15);

4 Compare $\boldsymbol{v}^{(s)}$ with $\boldsymbol{v}^{(s-1)}$ in a convenient matrix norm $\|\cdot\|$:

If $\|\boldsymbol{v}^{(s)}-\boldsymbol{v}^{(s-1)}\|<\varepsilon$, STOP and OUTPUT;

Else $s=s+1$ and return to line 2.

4.2 KFCM-F algorithm

KFCM-F clustering is done by mapping each pattern using the nonlinear transformation Φ and then computing the prototypes in the feature space, denoted as $\boldsymbol{v}_k^{\Phi}$. The objective function is

$$J^{\Phi}(\boldsymbol{\mu},\boldsymbol{v}^{\Phi})=\sum_{i=1}^{N}\sum_{k=1}^{C}\mu_{ki}^{m}\left\|\Phi(\boldsymbol{x}_i)-\boldsymbol{v}_k^{\Phi}\right\|^2, \tag{16}$$

where $\boldsymbol{v}_k^{\Phi}$ represents a linear sum of all $\Phi(\boldsymbol{x}_i)$ as follows:

$$\boldsymbol{v}_k^{\Phi}=\sum_{i=1}^{N}\mu_{ki}^{m}\Phi(\boldsymbol{x}_i)\Big/\sum_{i=1}^{N}\mu_{ki}^{m}=a_k\sum_{i=1}^{N}\mu_{ki}^{m}\Phi(\boldsymbol{x}_i), \tag{17}$$

with

$$a_k=\left(\sum_{i=1}^{N}\mu_{ki}^{m}\right)^{-1}.$$

Using the expression for the prototypes in Eq. (17), the distance in Eq. (16) could be reformulated as

$$\begin{aligned}\left\|\Phi(\boldsymbol{x}_i)-\boldsymbol{v}_k^{\Phi}\right\|^2&=\Phi(\boldsymbol{x}_i)\Phi(\boldsymbol{x}_i)-2\Phi(\boldsymbol{x}_i)\boldsymbol{v}_k^{\Phi}+\boldsymbol{v}_k^{\Phi}\boldsymbol{v}_k^{\Phi}\\&=K(\boldsymbol{x}_i,\boldsymbol{x}_i)-2a_k\sum_{r=1}^{N}\mu_{kr}^{m}K(\boldsymbol{x}_i,\boldsymbol{x}_r)\\&\quad+a_k^2\sum_{r=1}^{N}\sum_{s=1}^{N}\mu_{kr}^{m}\mu_{ks}^{m}K(\boldsymbol{x}_r,\boldsymbol{x}_s).\end{aligned} \tag{18}$$

Substituting Eq. (18) into Eq. (16) generates an objective function without $\boldsymbol{v}^{\Phi}$:

$$\begin{aligned}J^{\Phi}(\boldsymbol{\mu})=\sum_{i=1}^{N}\sum_{k=1}^{C}\mu_{ki}^{m}\Bigg(&K(\boldsymbol{x}_i,\boldsymbol{x}_i)-2a_k\sum_{r=1}^{N}\mu_{kr}^{m}K(\boldsymbol{x}_i,\boldsymbol{x}_r)\\&+a_k^2\sum_{r=1}^{N}\sum_{s=1}^{N}\mu_{kr}^{m}\mu_{ks}^{m}K(\boldsymbol{x}_r,\boldsymbol{x}_s)\Bigg).\end{aligned} \tag{19}$$

Similar to KFCM-I using the Lagrangian multiplier method and partial derivative with respect to μ_{ki}, using the process to the GRBF kernel gives the update function for memberships as

$$\mu_{ki}=\frac{\left(1-2a_k\sum_{r=1}^{N}\mu_{kr}^{m}K(\boldsymbol{x}_i,\boldsymbol{x}_r)+a_k^2\sum_{r=1}^{N}\sum_{s=1}^{N}\mu_{kr}^{m}\mu_{ks}^{m}K(\boldsymbol{x}_r,\boldsymbol{x}_s)\right)^{\frac{-1}{m-1}}}{\sum_{j=1}^{C}\left(1-2a_j\sum_{r=1}^{N}\mu_{jr}^{m}K(\boldsymbol{x}_i,\boldsymbol{x}_r)+a_j^2\sum_{r=1}^{N}\sum_{s=1}^{N}\mu_{jr}^{m}\mu_{js}^{m}K(\boldsymbol{x}_r,\boldsymbol{x}_s)\right)^{\frac{-1}{m-1}}}. \tag{20}$$

The steps for KFCM-F are described in Algorithm 4, while the performances of KFCM-I and KFCM-F for a nonlinear separable ball-ring dataset (C=2) are shown in Figs. 3a and 3b, respectively. Since the prototypes in KFCM-F are retained in the feature space, they lack intuitive descriptions with symbols in the input space. Fig. 3 shows that KFCM-F perfectly classifies the dataset into two clusters of the ball and the ring, but KFCM-I cannot detect the nonlinear data correctly, although prototypes can be visualized in the input space.

Algorithm 4 KFCM-F clustering

Input:

(1) $X=\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_i \in \mathbb{R}^d$, the dataset;

(2) C, $1<C\leq N$, the number of clusters;

(3) $\varepsilon>0$, the stopping criterion;

(4) $\boldsymbol{\mu}^{(0)}=(\boldsymbol{\mu}_1^{(0)}, \boldsymbol{\mu}_2^{(0)}, \ldots, \boldsymbol{\mu}_C^{(0)})$, the initials of memberships;

(5) $m>1$, the weighting exponent;

(6) σ, the GRBF kernel parameter.

Output:

$\boldsymbol{\mu}=(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_C)$, the final memberships.

1 $s=1$;

2 Update $\boldsymbol{\mu}^{(s)}=(\boldsymbol{\mu}_1^{(s)}, \boldsymbol{\mu}_2^{(s)}, \ldots, \boldsymbol{\mu}_C^{(s)})$ with $\boldsymbol{\mu}^{(s-1)}$ using Eq. (20);

3 Compute $E^{(s)}=\max_{i,k}|\mu_{ki}^{(s)}-\mu_{ki}^{(s-1)}|$:

If $E^{(s)}<\varepsilon$, STOP and OUTPUT;

Else $s=s+1$ and return to line 2.

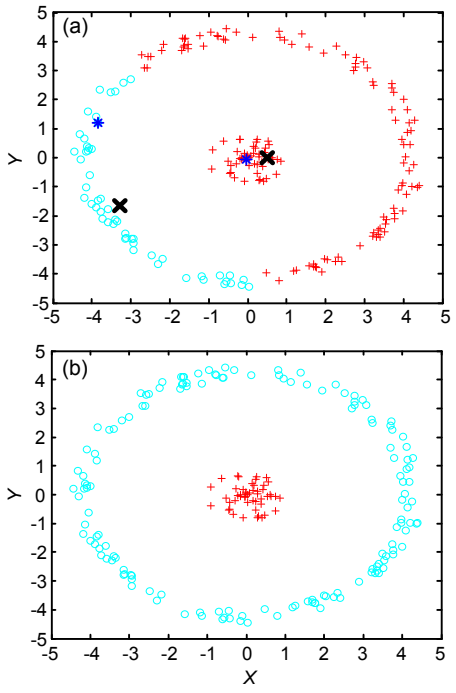

Fig. 3 Clustering results for the ball-ring dataset using KFCM-I (a) and KFCM-F(b)

*: random initial position of the cluster prototypes; ×: final cluster prototypes

5 Novel global kernel-based fuzzy c-means-F (GKFCM-F)

From the previous sections, we find that GFCM and KFCM are improved by starting at different points. To enhance the performance, we propose a novel clustering method by embedding the kernel-based algorithm into GFCM to resolve the two limitations of FCM, namely sensitivity to initialization and inability to use nonlinear separable data. KFCM-F is considered to be integrated with GFCM to form GKFCM-F since KFCM-F performs better than KFCM-I, as observed in Section 4. However, there is some difficulty in realizing GKFCM-F because of the disadvantage that the prototypes lack intuitive descriptions in the input space. To solve this problem, we find the approximate prototypes $\tilde{\boldsymbol{v}}_k$ in the input space by minimizing

$$\begin{aligned} V &= \sum_{k=1}^{C}\left\|\Phi(\tilde{\boldsymbol{v}}_k)-\boldsymbol{v}_k^{\Phi}\right\|^2 \\ &= \sum_{k=1}^{C}\left(\Phi(\tilde{\boldsymbol{v}}_k)\Phi(\tilde{\boldsymbol{v}}_k)-2\Phi(\tilde{\boldsymbol{v}}_k)\boldsymbol{v}_k^{\Phi}+\boldsymbol{v}_k^{\Phi}\boldsymbol{v}_k^{\Phi}\right). \end{aligned} \tag{21}$$

By substituting the expression for $\boldsymbol{v}_k^{\Phi}$ in Eq. (17) into Eq. (21), we obtain

$$\begin{aligned} V = \sum_{k=1}^{C}\Bigg(& K(\tilde{\boldsymbol{v}}_k,\tilde{\boldsymbol{v}}_k)-2a_k\sum_{i=1}^{N}\mu_{ki}^m K(\boldsymbol{x}_i,\tilde{\boldsymbol{v}}_k) \\ & +\sum_{l=1}^{N}\sum_{r=1}^{N}\mu_{kl}^m\mu_{kr}^m K(\boldsymbol{x}_l,\boldsymbol{x}_r)\Bigg). \end{aligned} \tag{22}$$

Since $K(\boldsymbol{x}_l, \boldsymbol{x}_r)$ is independent of $\tilde{\boldsymbol{v}}_k$, $\dfrac{\partial K(\boldsymbol{x}_l,\boldsymbol{x}_r)}{\partial\tilde{\boldsymbol{v}}_k}=0$. Given the GRBF kernel, $K(\tilde{\boldsymbol{v}}_k, \tilde{\boldsymbol{v}}_k)=1$, and then $\dfrac{\partial K(\tilde{\boldsymbol{v}}_k,\tilde{\boldsymbol{v}}_k)}{\partial\tilde{\boldsymbol{v}}_k}=0$. Solving $\dfrac{\partial V}{\partial\tilde{\boldsymbol{v}}_k}=0$ yields

$$\sum_{i=1}^{N}\mu_{ki}^m K(\boldsymbol{x}_i,\tilde{\boldsymbol{v}}_k)\frac{2(\boldsymbol{x}_i-\tilde{\boldsymbol{v}}_k)}{\sigma^2}=0,$$

which leads to

$$\tilde{\boldsymbol{v}}_k=\sum_{i=1}^{N}\mu_{ki}^m K(\boldsymbol{x}_i,\tilde{\boldsymbol{v}}_k)\boldsymbol{x}_i \Big/ \sum_{i=1}^{N}\mu_{ki}^m K(\boldsymbol{x}_i,\tilde{\boldsymbol{v}}_k). \tag{23}$$

Based on the approximate prototypes $\tilde{\boldsymbol{v}}_k$, we could establish an approximate objective function as

$$\begin{aligned} \tilde{J}^{\Phi}(\boldsymbol{\mu},\tilde{\boldsymbol{v}}) &= \sum_{i=1}^{N}\sum_{k=1}^{C}\mu_{ki}^m\left\|\Phi(\boldsymbol{x}_i)-\Phi(\tilde{\boldsymbol{v}}_k)\right\|^2 \\ &= 2\sum_{i=1}^{N}\sum_{k=1}^{C}\mu_{ki}^m\left(1-K(\boldsymbol{x}_i,\tilde{\boldsymbol{v}}_k)\right). \end{aligned} \tag{24}$$

For the problem starting with one cluster (k=1), all the memberships $\mu_{1i}(1)$=1. To locate the optimal approximate prototype $\tilde{\boldsymbol{v}}^*(1)$, it should be noted that the centroid of X in the input space does not correspond to the centroid of $\Phi(X)$ in the feature space due to the nonlinear transformation. Thus, we first place the initial $\tilde{\boldsymbol{v}}(1)$ at the data point, which minimizes Eq. (24) since it guarantees the most reduction in $\tilde{J}^{\Phi}(\boldsymbol{\mu},\boldsymbol{v})$. Then the final $\tilde{\boldsymbol{v}}^*(1)$ around could be rapidly found by iterative updating using Eq. (23). For the subsequent 2-clustering problem, we set the initial state $\left(\tilde{\boldsymbol{v}}_1(2),\tilde{\boldsymbol{v}}_2(2)\right)$ corresponding to $(\tilde{\boldsymbol{v}}^*(1),\boldsymbol{x}_n)$ ($1\le n\le N$). With respect to the nth data point, the memberships at the current stage are initialized as

$$\mu_{hi}^{n}(k)=\frac{\left\|\Phi(\boldsymbol{x}_i)-\Phi\left(\tilde{\boldsymbol{v}}_h(k)\right)\right\|^{-2}}{\sum_{h=1}^{k}\left\|\Phi(\boldsymbol{x}_i)-\Phi\left(\tilde{\boldsymbol{v}}_h(k)\right)\right\|^{-2}}. \tag{25}$$

The optimal memberships $\mu_{hi}^*(2)$ are selected from the N runs of local searches using KFCM-F. Based on $\mu_{hi}^*(2)$, the optimal approximate prototypes $\{\tilde{\boldsymbol{v}}_1^*(2),\tilde{\boldsymbol{v}}_2^*(2)\}$, are obtained using Eq. (22). To generalize, the initial state for each intermediate step would be

$$\{\tilde{\boldsymbol{v}}_1^*(k-1),\tilde{\boldsymbol{v}}_2^*(k-1),\ldots,\tilde{\boldsymbol{v}}_{k-1}^*(k-1),\boldsymbol{x}_i\},$$

where $\{\tilde{\boldsymbol{v}}_1^*(k-1),\tilde{\boldsymbol{v}}_2^*(k-1),\ldots,\tilde{\boldsymbol{v}}_{k-1}^*(k-1)\}$ denotes the optimal solution for the previous (k−1)-partition problem. By selecting each data point $\boldsymbol{x}_n$ to form different initial states, the corresponding initial memberships $\mu_{hi}^n(k)$ are also determined using Eq. (25). The KFCMF algorithm is then performed on each set of the initials and the best updating result is selected as $\mu_{hi}^*(k)$. Then, substituting $\mu_{hi}^*(k)$ into Eq. (23) and updating from the initial state, $\{\tilde{\boldsymbol{v}}_1^*(k-1),\tilde{\boldsymbol{v}}_2^*(k-1),\ldots,\tilde{\boldsymbol{v}}_{k-1}^*(k-1),\boldsymbol{x}_l\}$ (l indicates the corresponding initial state for deriving $\mu_{hi}^*(k)$) yields the optimal approximate prototypes $\{\tilde{\boldsymbol{v}}_1^*(k),\tilde{\boldsymbol{v}}_2^*(k),\ldots,\tilde{\boldsymbol{v}}_k^*(k)\}$ for the current k-partition problem. The best solution for C-clustering could be obtained through the above procedure.

The proposed GKFCM-F algorithm is summarized in Algorithm 5, where KFCM-F refers to the KFCM-F algorithm.

Algorithm 5 GKFCM-F clustering

Input:

(1) $X=\{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_i,\ldots,\boldsymbol{x}_N\}$, $\boldsymbol{x}_i\in\mathbb{R}^d$, the dataset;

(2) C, $1<C\le N$, the number of clusters;

(3) $\varepsilon>0$, the stopping criterion;

(4) $\boldsymbol{\mu}^{(0)}=(\boldsymbol{\mu}_1^{(0)},\boldsymbol{\mu}_2^{(0)},\ldots,\boldsymbol{\mu}_C^{(0)})$, the initials of memberships;

(5) $m>1$, the weighting exponent;

(6) σ, the GRBF kernel parameter.

Output:

(1) $\tilde{\boldsymbol{v}}=(\tilde{\boldsymbol{v}}_1,\tilde{\boldsymbol{v}}_2,\ldots,\tilde{\boldsymbol{v}}_C)$, the final cluster prototypes;

(2) $\boldsymbol{\mu}=(\boldsymbol{\mu}_1,\boldsymbol{\mu}_2,\ldots,\boldsymbol{\mu}_C)$, the final memberships.

Compute $\tilde{\boldsymbol{v}}^*(1)$ using Eq. (23) with initial position $\tilde{\boldsymbol{v}}(1)$ placed at the data point which minimizes Eq. (24);
for k=2 to C **do**
  for n=1 to N **do**
    Set the initial state
    $\tilde{\boldsymbol{v}}^{n(0)}(k)=\left(\tilde{\boldsymbol{v}}_1(k),\tilde{\boldsymbol{v}}_2(k),\ldots,\tilde{\boldsymbol{v}}_{k-1}(k),\tilde{\boldsymbol{v}}_k(k)\right)$
    $=\left(\tilde{\boldsymbol{v}}_1^*(k-1),\tilde{\boldsymbol{v}}_2^*(k-1),\ldots,\tilde{\boldsymbol{v}}_{k-1}^*(k-1),\boldsymbol{x}_n\right)$;
    Set initial memberships $\boldsymbol{\mu}^{n(0)}(k)$ with respect to $\tilde{\boldsymbol{v}}^{n(0)}(k)$ using Eq. (25);
    $\boldsymbol{\mu}^n(k)\leftarrow$ KFCM-F$\left(X,k,\varepsilon,m,\sigma^2,\boldsymbol{\mu}^{n(0)}(k)\right)$;
  end
  $l=\arg\min_{1\le n\le N} J^{\Phi}\left(\boldsymbol{\mu}^n(k)\right)$ using Eq. (19);
  $\boldsymbol{\mu}^*(k)\leftarrow\boldsymbol{\mu}^l(k)$;
  $\tilde{\boldsymbol{v}}^{(0)}(k)\leftarrow\tilde{\boldsymbol{v}}^{l(0)}(k)$;
  s=1;
  Update $\tilde{\boldsymbol{v}}^{(s)}(k)$ with $\tilde{\boldsymbol{v}}^{(s-1)}(k)$ using Eq. (23);
  If $\|\tilde{\boldsymbol{v}}^{(s)}(k)-\tilde{\boldsymbol{v}}^{(s-1)}(k)\|<\varepsilon$
    STOP and $\tilde{\boldsymbol{v}}^*(k)\leftarrow\tilde{\boldsymbol{v}}^{(s)}(k)$;
  Else $s=s+1$ and return to line 12;
end
$\tilde{\boldsymbol{v}}\leftarrow\tilde{\boldsymbol{v}}^*(C)$, $\boldsymbol{\mu}\leftarrow\boldsymbol{\mu}^*(C)$.

6 Accelerating scheme: fast global kernel-based fuzzy c-means-F (FGKFCM-F)

In Section 5, we propose GKFCM-F to overcome the shortcomings of FCM. In each intermediate process for k-partition, $O(N)$ executions of KFCM-F are required; thus, the total computation complexity is $O(CN)$. Consider that if the dataset X has a large size of N, the computation cost will be very high.

To address this problem, we propose an accelerating scheme named FGKFCM-F.

The basic idea is that at each intermediate step, instead of carrying out the KFCM-F algorithm for each set of initial memberships, we first choose the best set that minimizes objective function (19). Then KFCM-F is executed only once from this initialization. In this way, the computation complexity will be reduced to $O(C)$.

The proposed FGKFCM-F is summarized in Algorithm 6. To provide a proper and adaptive parameter σ of the GRBF kernel for different datasets, we use the sample variance to estimate σ^2 (Yang and Tsai, 2008) with

$$\sigma^2=\sum_{i=1}^{N}\|\boldsymbol{x}_i-\bar{\boldsymbol{x}}\|^2/N,\quad \bar{\boldsymbol{x}}=\sum_{i=1}^{N}\boldsymbol{x}_i/N. \tag{26}$$

7 Simulations

To investigate the effectiveness of the proposed algorithm, the first experiment is conducted on the same artificial dataset as that used in Section 4. Fig. 4 shows that FGKFCM-F successfully groups these patterns into the ball cluster and the ring cluster with final prototypes located in the input space. Fig. 5 visualizes the optimal approximate cluster prototypes for each intermediate case. From the results, we could observe that the proposed algorithm uses the advantages of both GFCM and KFCM to realize near-optimal classification. There is no dependency on the random initial values since the optimal solution obtained in each previous stage provides deterministic initialization for the next stage. The poor local minima could be avoided and thus the near-optimal solution is fixed by searching in such an incremental way. Moreover, the inability to accommodate nonlinear separable data is solved with the help of kernel methods.

Algorithm 6 FGKFCM-F clustering

Input:

(1) $X=\{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_i,\ldots,\boldsymbol{x}_N\}$, $\boldsymbol{x}_i\in\mathbb{R}^d$, the dataset;

(2) C, $1<C\leq N$, the number of clusters;

(3) $\varepsilon>0$, the stopping criterion;

(4) $\boldsymbol{\mu}^{(0)}=(\boldsymbol{\mu}_1^{(0)},\boldsymbol{\mu}_2^{(0)},\ldots,\boldsymbol{\mu}_C^{(0)})$, the initials of memberships;

(5) $m>1$, the weighting exponent;

(6) σ, the GRBF kernel parameter.

Output:

(1) $\tilde{\boldsymbol{v}}=(\tilde{\boldsymbol{v}}_1,\tilde{\boldsymbol{v}}_2,\ldots,\tilde{\boldsymbol{v}}_C)$, the final cluster prototypes;

(2) $\boldsymbol{\mu}=(\boldsymbol{\mu}_1,\boldsymbol{\mu}_2,\ldots,\boldsymbol{\mu}_C)$, the final memberships.

1 Compute $\tilde{\boldsymbol{v}}^*(1)$ using Eq. (23) with initial position $\tilde{\boldsymbol{v}}(1)$ placed at the data point that minimizes Eq. (24);
2 **for** k=2 to C **do**
3 **for** n=1 to N **do**
4 Set the initial state
$\tilde{\boldsymbol{v}}^{n(0)}(k)=(\tilde{\boldsymbol{v}}_1(k),\tilde{\boldsymbol{v}}_2(k),\ldots,\tilde{\boldsymbol{v}}_{k-1}(k),\tilde{\boldsymbol{v}}_k(k))$
$=(\tilde{\boldsymbol{v}}_1^*(k-1),\tilde{\boldsymbol{v}}_2^*(k-1),\ldots,\tilde{\boldsymbol{v}}_{k-1}^*(k-1),\boldsymbol{x}_n)$;
5 Set initial memberships $\boldsymbol{\mu}^{n(0)}(k)$ with respect to $\tilde{\boldsymbol{v}}^{n(0)}(k)$ using Eq. (25);
6 $\boldsymbol{\mu}^n(k)\leftarrow\text{KFCM-F}\left(X,k,\varepsilon,m,\sigma^2,\boldsymbol{\mu}^{n(0)}(k)\right)$;
7 **end**
8 $l=\arg\min_{1\leq n\leq N}J^{\Phi}(\boldsymbol{\mu}^n(k))$ using Eq. (19);
9 $\boldsymbol{\mu}^*(k)\leftarrow\boldsymbol{\mu}^l(k)$;
10 $\tilde{\boldsymbol{v}}^{(0)}(k)\leftarrow\tilde{\boldsymbol{v}}^{l(0)}(k)$;
11 s=1;
12 Update $\tilde{\boldsymbol{v}}^{(s)}(k)$ with $\tilde{\boldsymbol{v}}^{(s-1)}(k)$ using Eq. (23);
13 If $\|\tilde{\boldsymbol{v}}^{(s)}(k)-\tilde{\boldsymbol{v}}^{(s-1)}(k)\|<\varepsilon$
14 STOP and $\tilde{\boldsymbol{v}}^*(k)\leftarrow\tilde{\boldsymbol{v}}^{(s)}(k)$;
15 Else $s=s+1$ and return to line 12;
16 **end**
17 $\tilde{\boldsymbol{v}}\leftarrow\tilde{\boldsymbol{v}}^*(C)$, $\boldsymbol{\mu}\leftarrow\boldsymbol{\mu}^*(C)$.

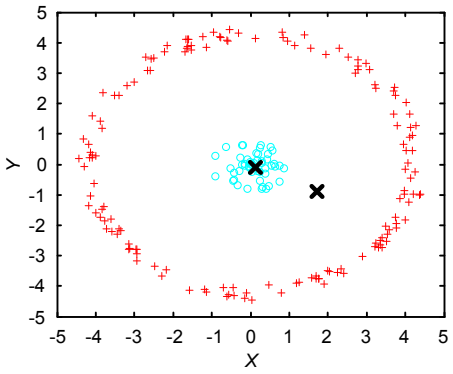

Fig. 4 Clustering results for the ball-ring dataset using FGKFCM-F

While GKFCM-F derives the same results as those shown in Fig. 4, more time is required to complete the clustering since it needs to execute N runs of the local search using KFCM-F at each intermediate step. A comparison of the computation complexity for GKFCM-F and FGKFCM-F is shown in Fig. 6, where each algorithm is executed 10 times. It is fairly

obvious that the running time of FGKFCM-F is much less than that of GKFCM-F, which demonstrates the effectiveness of the acceleration scheme in lowering the computational cost.

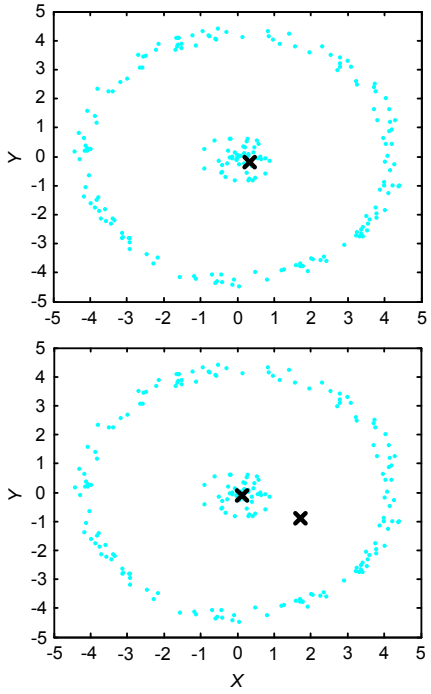

Fig. 5 Optimal approximate cluster prototypes (×) obtained at each intermediate step in FGKFCM-F

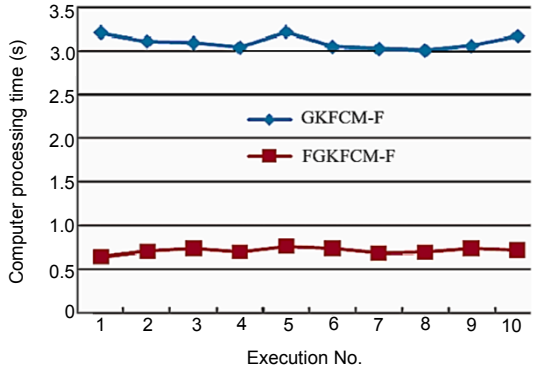

Fig. 6 Comparison of computer processing time between GKFCM-F and FGKFCM-F

Another experiment is then conducted using real-world data speech signal. We expect to realize the segmentation of consonant/vowel (C/V) sounds through the proposed clustering algorithm. The speech dataset is composed of the utterance of 100 speakers from the TIMIT database. We select a total of 1000 words and divide them into two groups: monosyllable words and polysyllable words. For the first group, we subdivide them into three categories: C^+VC^+, VC^+, and C^+V, where C indicates a consonant, V indicates a vowel, and $^+$ denotes one or more consecutive consonants. Each syllable in the English language generally contains only one vowel. The polysyllable words, with various combinations of syllables, are simply categorized into one group. The reference segmenting points are manually marked according to the associated transcription files. The speech signal of each word is divided into frames of 8 ms (at a sampling frequency of 16000 Hz) by a Hamming window with a shift of 4 ms. For each frame, a 256-point discrete Fourier transform (DFT) is computed, and the mel-frequency cepstral coefficients (MFCC) (Picone, 1993; Jamaati and Marvi, 2008) are extracted to compose the speech parametric vector (pattern). When the stopping criterion is reached, each MFCC pattern will be assigned to a specified cluster according to the maximum membership value. Thus, a sequential string of a cluster label will be obtained and the instant with a change in the cluster label value is hypothesized as the segmentation point of a consonant and vowel.

Firstly, we consider a monosyllable word, 'that', for illumination. The syllable structure of this word is CVC; hence, there are two places at which it can be divided into consonant and vowel parts. The segmentation results obtained via FGKFCM-F are shown in Fig. 7. The small matching error indicates the good performance of the proposed algorithm. For the entire speech dataset, the statistics are given in Table 1. It could be observed that for words with a simple syllable structure, the consonant and vowel are partitioned successfully with a rate of over 90%, even at a long time resolution of 20 ms. With increasing complexity of the word structure, the detection rate decreases by a certain degree. This is mainly because there may be different types of consonants around each vowel in complicated words, and some highly confusing voiced consonants, such as semivowels and liquids, have characteristics similar to vowels; accordingly, the boundary between them may be missed or located in a larger deviation.

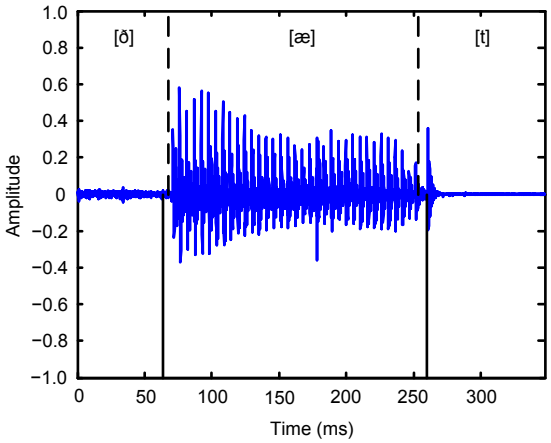

Fig. 7 Consonant/vowel segmentation results for the word 'that' using FGKFCM-F (matching error: 3.9 ms; 6.8 ms)

Dashed lines correspond to the reference segmenting points and solid lines correspond to the hypothesized location for segmentation

Table 1 Consonant/vowel (C/V) segmentation results for 1000 words using FGKFCM-F

Word category	Number of reference points	DETR_T (%)			
		10 ms	20 ms	30 ms	40 ms
Monosyllable*					
C^+VC^+	800	43.75	66.75	76.63	84.00
VC^+	200	75.00	90.50	95.50	98.50
C^+V	200	51.00	93.50	98.00	99.50
Polysyllable	500	38.60	56.6	65.80	72.00

* C indicates a consonant, V indicates a vowel, and + denotes one or more consecutive consonants. DETR_T refers to the detection rate of segmentation points within the specified time resolution of T

Previously, it was demonstrated that GFCM needs no initialization and KFCM-F could handle nonlinear data. Both of them are improved versions of FCM. An experiment is then carried out to investigate the superiority of FGKFCM-F over GFCM and KFCM-F. For the same dataset, a comparison of the average DETR_T between FGKFCM-F, GFCM, and KFCM-F is shown in Fig. 8. As an extension to GFCM and KFCM-F, FGKFCM-F realizes higher accuracy in C/V segmentation at each time resolution. This demonstrates that the proposed algorithm combines the advantages of GFCM and KFCM-F to establish a method capable of overcoming the two limitations previously mentioned. This higher performance in C/V segmentation and lower DETR_T of the FGKFCM-F is slightly tempered with its poor performance in dealing with polysyllable words.

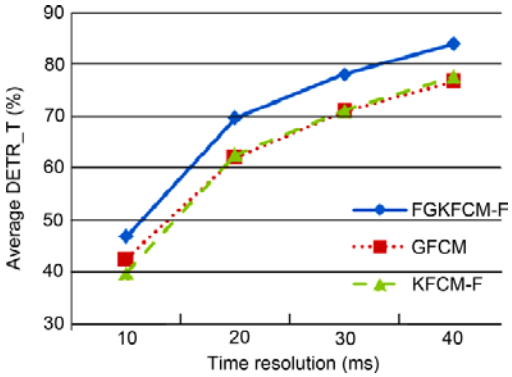

Fig. 8 Average DETR_T of GFCM, KFCM-F, and FGKFCM-F

8 Conclusions

While fuzzy c-means (FCM) is a simple and efficient algorithm for clustering, it has several well-known shortcomings: sensitivity to initialization and inability to use nonlinear separable data. In this paper, we develop fast global kernel fuzzy c-means-F (FGKFCM-F) by reformulating a global fuzzy c-means (GFCM) embedded with a kernel-based fuzzy c-means-F (KFCM-F), which is effective in overcoming these shortcomings. This is an incremental approach to solving the C-clustering problem by solving all intermediate problems with 1, 2, …, C clusters using KFCM-F as a local search procedure. This algorithm integrates the advantages of KFCM-F and GFCM to realize a near-optimal solution for nonlinearly separable data. Moreover, by using an accelerated scheme to choose the best set of initial memberships, KFCM-F is executed only once at each stage, which lowers the computation complexity and improves the convergence speed. Experimental results show that FGKFCM-F is superior to other methods for both the artificial and real-world datasets of speech signal.

Further research could be focused on exploring more generalized kernels to provide flexibility in various data structures and robustness to complex datasets and developing a semi-supervised clustering algorithm to optimize the kernel parameters and performance. We might also consider a better method

of feature extraction to enhance the difference between the two categories of phonemes, especially vowels and confusable consonants, to improve the performance in dealing with polysyllable words so that application could then be extended to continuous speech.

References

Bagirov, A.M., 2008. Modified global k-means algorithm for minimum sum-of-squares clustering problems. *Pattern Recogn.*, **41**(10):3192-3199. [doi:10.1016/j.patcog.2008.04.004]

Balasko, B., Abonyi, J., Feil, B., 2005. Fuzzy Clustering and Data Analysis Toolbox. Department of Process Engineering, University of Veszprem, Veszprem.

Bezdek, J.C., 1981. Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York.

Bozkir, A.S., Sezer, E.A., 2013. FUAT—a fuzzy clustering analysis tool. *Expert Syst. Appl.*, **40**(3):842-849. [doi:10.1016/j.eswa.2012.05.038]

Chiang, J.H., Hao, P.Y., 2003. A new kernel-based fuzzy clustering approach: support vector clustering with cell growing. *IEEE Trans. Fuzzy Syst.*, **11**(4):518-527. [doi:10.1109/TFUZZ.2003.814839]

Cover, T.M., 1965. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE Trans. Electr. Comput.*, **EC-14**(3):326-334. [doi:10.1109/PGEC.1965.264137]

Duda, R.O., Hart, P.E., 1973. Pattern Classification and Scene Analysis. Wiley, New York.

Dunn, J.C., 1973. A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. *J. Cybern.*, **3**(3):32-57. [doi:10.1080/01969727308546046]

Filippone, M., Camastra, F., Masulli, F., *et al*., 2008. A survey of kernel and spectral methods for clustering. *Pattern Recogn.*, **41**(1):176-190. [doi:10.1016/j.patcog.2007.05.018]

Girolami, M., 2002. Mercer kernel-based clustering in feature space. *IEEE Trans. Neur. Netw.*, **13**(3):780-784. [doi:10.1109/TNN.2002.1000150]

Gong, M., Su, L., Jia, M., *et al*., 2014. Fuzzy clustering with a modified MRF energy function for change detection in synthetic aperture radar images. *IEEE Trans. Fuzzy Syst.*, **22**(1):98-109. [doi:10.1109/TFUZZ.2013.2249072]

Hu, Y., Wu, D., Nucci, A., 2013. Fuzzy-clustering-based decision tree approach for large population speaker identification. *IEEE Trans. Audio Speech Lang. Process.*, **21**(4):762-774. [doi:10.1109/TASL.2012.2234113]

Jain, A.K., Murty, M.N., Flynn, P.J., 1999. Data clustering: a review. *ACM Comput. Surv.*, **31**(3):264-323. [doi:10.1145/331499.331504]

Jamaati, M., Marvi, H., 2008. Performance assessment of joint feature derived from Mellin-cepstrum for vowel recognition. *Int. Rev. Electr. Eng. IREE*, **3**(6):1077-1086.

Kim, D.W., Lee, K.Y., Lee, D., *et al*., 2005. Evaluation of the performance of clustering algorithms in kernel-induced feature space. *Pattern Recogn.*, **38**(4):607-611. [doi:10.1016/j.patcog.2004.09.006]

Li, Z., Tang, S., Xue, J., *et al*., 2001. Modified FCM clustering based on kernel mapping. Multispectral Image Processing and Pattern Recognition, International Society for Optics and Photonics, p.241-245. [doi:10.1117/12.441658]

Likas, A., Vlassis, N., Verbeek, J.J., 2003. The global k-means clustering algorithm. *Pattern Recogn.*, **36**(2):451-461. [doi:10.1016/S0031-3203(02)00060-2]

Liu, C., Zhang, X., Li, X., *et al*., 2012. Gaussian kernelized fuzzy c-means with spatial information algorithm for image segmentation. *J. Comput.*, **7**(6):1511-1518. [doi:10.4304/jcp.7.6.1511-1518]

Mercer, J., 1909. Functions of positive and negative type, and their connection with the theory of integral equations. *Phil. Trans. R. Soc. Lond. Ser. A*, **209**(441-458):415-446. [doi:10.1098/rsta.1909.0016]

Muller, K., Mika, S., Ratsch, G., *et al*., 2001. An introduction to kernel-based learning algorithms. *IEEE Trans. Neur. Netw.*, **12**(2):181-201. [doi:10.1109/72.914517]

Nguyen, T.M., Wu, Q.M.J., 2013. Dynamic fuzzy clustering and its application in motion segmentation. *IEEE Trans. Fuzzy Syst.*, **21**(6):1019-1031. [doi:10.1109/TFUZZ.2013.2240689]

Picone, J.W., 1993. Signal modeling techniques in speech recognition. *Proc. IEEE*, **81**(9):1215-1247. [doi:10.1109/5.237532]

Shen, H.B., Yang, J., Wang, S.T., *et al*., 2006. Attribute weighted Mercer kernel based fuzzy clustering algorithm for general non-spherical datasets. *Soft Comput.*, **10**(11):1061-1073. [doi:10.1007/s00500-005-0043-5]

Tsai, D.M., Lin, C.C., 2011. Fuzzy c-means based clustering for linearly and nonlinearly separable data. *Pattern Recogn.*, **44**(8):1750-1760. [doi:10.1016/j.patcog.2011.02.009]

Wang, W., Zhang, Y., Li, Y., *et al*., 2006. The global fuzzy c-means clustering algorithm. 6th World Congress on Intelligent Control and Automation, p.3604-3607. [doi:10.1109/WCICA.2006.1713041]

Wu, Z., Xie, W., Yu, J., 2003. Fuzzy c-means clustering algorithm based on kernel method. Proc. 5th Int. Conf. on Computational Intelligence and Multimedia Applications, p.49-54. [doi:10.1109/iccima.2003.1238099]

Xu, R., Wunsch, D., 2005. Survey of clustering algorithms. *IEEE Trans. Neur. Netw.*, **16**(3):645-678. [doi:10.1109/TNN.2005.845141]

Yang, M.S., Tsai, H.S., 2008. A Gaussian kernel-based fuzzy c-means algorithm with a spatial bias correction. *Pattern Recogn. Lett.*, **29**(12):1713-1725. [doi:10.1016/j.patrec.2008.04.016]

Yu, C.Y., Li, Y., Liu, A.L., *et al*., 2011. A novel modified kernel fuzzy c-means clustering algorithm on image segementation. IEEE 14th Int. Conf. on Computational Science and Engineering, p.621-626. [doi:10.1109/cse.2011.109]

Zadeh, L.A., 1965. Fuzzy sets. *Inf. Contr.*, **8**(3):338-353. [doi:10.1016/S0019-9958(65)90241-X]

Zhang, D.Q., Chen, S.C., 2002. Fuzzy clustering using kernel method. Int. Conf. on Control and Automation, p.162-

163. [doi:10.1109/icca.2002.1229535]
Zhang, D.Q., Chen, S.C., 2003a. Clustering incomplete data using kernel-based fuzzy c-means algorithm. *Neur. Process. Lett.*, **18**(3):155-162. [doi:10.1023/B:NEPL.0000011135.19145.1b]
Zhang, D.Q., Chen, S.C., 2003b. Kernel-based fuzzy and possibilistic c-means clustering. Proc. Int. Conf. on Artificial Neural Network, p.122-125.
Zhang, D.Q., Chen, S.C., 2004. A novel kernelized fuzzy c-means algorithm with application in medical image segmentation. *Artif. Intell. Med.*, **32**(1):37-50. [doi:10.1016/j.artmed.2004.01.012]
Zhao, F., 2013. Fuzzy clustering algorithms with self-tuning non-local spatial information for image segmentation. *Neurocomputing*, **106**:115-125. [doi:10.1016/j.neucom.2012.10.022]
Zhou, S., Gan, J.Q., 2004. Mercer kernel, fuzzy c-means algorithm, and prototypes of clusters. *LNCS*, **3177**:613-618. [doi:10.1007/978-3-540-28651-6_90]

Appendix: Derivation of Eqs. (4) and (5)

Expand Eq. (3) as

$$\begin{aligned}
&\overline{J}(\boldsymbol{\mu},\boldsymbol{v},\lambda)\\
&=\mu_{11}^m\left\|\boldsymbol{x}_1-\boldsymbol{v}_1\right\|^2+\mu_{12}^m\left\|\boldsymbol{x}_2-\boldsymbol{v}_1\right\|^2+\ldots+\mu_{1N}^m\left\|\boldsymbol{x}_N-\boldsymbol{v}_1\right\|^2\\
&+\mu_{21}^m\left\|\boldsymbol{x}_1-\boldsymbol{v}_2\right\|^2+\mu_{22}^m\left\|\boldsymbol{x}_2-\boldsymbol{v}_2\right\|^2+\ldots+\mu_{2N}^m\left\|\boldsymbol{x}_N-\boldsymbol{v}_2\right\|^2\\
&+\mu_{k1}^m\left\|\boldsymbol{x}_1-\boldsymbol{v}_k\right\|^2+\ldots+\underline{\underline{\mu_{ki}^m\left\|\boldsymbol{x}_i-\boldsymbol{v}_k\right\|^2}}+\ldots+\mu_{kN}^m\left\|\boldsymbol{x}_N-\boldsymbol{v}_k\right\|^2\\
&\vdots\\
&+\mu_{C1}^m\left\|\boldsymbol{x}_1-\boldsymbol{v}_C\right\|^2+\mu_{C2}^m\left\|\boldsymbol{x}_2-\boldsymbol{v}_C\right\|^2+\ldots+\mu_{CN}^m\left\|\boldsymbol{x}_N-\boldsymbol{v}_C\right\|^2\\
&+\lambda_1\left(\sum_{k=1}^C\mu_{k1}-1\right)+\ldots+\underline{\underline{\lambda_i\left(\sum_{k=1}^C\mu_{ki}-1\right)}}+\ldots\\
&+\lambda_N\left(\sum_{k=1}^C\mu_{kN}-1\right),
\end{aligned}\tag{A1}$$

where only the two underlined parts involve μ_{ki}; thus, solving $\dfrac{\partial\overline{J}(\boldsymbol{\mu},\boldsymbol{v},\lambda)}{\partial\mu_{ki}}=0$ yields

$$m\mu_{ki}^{m-1}\left\|\boldsymbol{x}_i-\boldsymbol{v}_k\right\|^2+\lambda_i=0,$$

which leads to

$$\mu_{ki}=\left(\frac{-\lambda_i}{m\left\|\boldsymbol{x}_i-\boldsymbol{v}_k\right\|^2}\right)^{\frac{1}{m-1}}=\left(\frac{-\lambda_i}{m}\right)^{\frac{1}{m-1}}\frac{1}{\left\|\boldsymbol{x}_i-\boldsymbol{v}_k\right\|^{\frac{2}{m-1}}}.\tag{A2}$$

Since $\sum_{k=1}^C\mu_{ki}=1$, we have

$$\sum_{k=1}^C\mu_{ki}=\sum_{k=1}^C\left(\frac{-\lambda_i}{m}\right)^{\frac{1}{m-1}}\frac{1}{\left\|\boldsymbol{x}_i-\boldsymbol{v}_k\right\|^{\frac{2}{m-1}}}=1.$$

Thus,

$$\left(\frac{-\lambda_i}{m}\right)^{\frac{1}{m-1}}=\left(\sum_{k=1}^C\frac{1}{\left\|\boldsymbol{x}_i-\boldsymbol{v}_k\right\|^{\frac{2}{m-1}}}\right)^{-1}.\tag{A3}$$

Substituting the expression for $(-\lambda_i/m)^{1/(m-1)}$ into Eq. (A2), we could obtain the update function for memberships as follows:

$$\begin{aligned}
\mu_{ki}&=\left(\sum_{j=1}^C\frac{1}{\left\|\boldsymbol{x}_i-\boldsymbol{v}_j\right\|^{\frac{2}{m-1}}}\right)^{-1}\left\|\boldsymbol{x}_i-\boldsymbol{v}_k\right\|^{-\frac{2}{m-1}}\\
&=\frac{\left\|\boldsymbol{x}_i-\boldsymbol{v}_k\right\|^{\frac{-2}{m-1}}}{\sum_{j=1}^C\left\|\boldsymbol{x}_i-\boldsymbol{v}_j\right\|^{\frac{-2}{m-1}}}.
\end{aligned}\tag{A4}$$

From Eq. (A1), we can find only

$$\mu_{k1}^m\left\|\boldsymbol{x}_1-\boldsymbol{v}_k\right\|^2+\cdots+\mu_{ki}^m\left\|\boldsymbol{x}_i-\boldsymbol{v}_k\right\|^2+\ldots+\mu_{kN}^m\left\|\boldsymbol{x}_N-\boldsymbol{v}_k\right\|^2$$

is dependent on $\boldsymbol{v}_k$; thus, taking the first derivative of $\partial\overline{J}(\boldsymbol{\mu},\boldsymbol{v},\lambda)$ with respect to $\boldsymbol{v}_k$ and setting it to zero, we can obtain the update function for cluster prototypes:

$$\frac{\partial\overline{J}(\boldsymbol{\mu},\boldsymbol{v},\lambda)}{\partial\boldsymbol{v}_k}=0.$$

Thus, we have

$$\sum_{i=1}^N\mu_{ki}^m\cdot2(\boldsymbol{x}_i-\boldsymbol{v}_k)\cdot(-1)=0.$$

That is,

$$\boldsymbol{v}_k=\sum_{i=1}^N\mu_{ki}^m\boldsymbol{x}_i\Big/\sum_{i=1}^N\mu_{ki}^m.\tag{A5}$$