Available online at www.sciencedirect.com

ScienceDirect

Fuzzy Sets and Systems 161 (2010) 522–543

FUZZY sets and systems

www.elsevier.com/locate/fss

Kernel-based fuzzy clustering and fuzzy clustering: A comparative experimental study

Daniel Graves*, Witold Pedrycz

Department of Electrical and Computer Engineering, 9107 – 116th Street, University of Alberta, Edmonton, Alberta, Canada T6G 2V4

Received 5 October 2007; received in revised form 29 September 2009; accepted 26 October 2009
Available online 6 November 2009

Abstract

In this study, we present a comprehensive comparative analysis of kernel-based fuzzy clustering and fuzzy clustering. Kernel based clustering has emerged as an interesting and quite visible alternative in fuzzy clustering, however, the effectiveness of this extension vis-à-vis some generic methods of fuzzy clustering has neither been discussed in a complete manner nor the performance of clustering quantified through a convincing comparative analysis. Our focal objective is to understand the performance gains and the importance of parameter selection for kernelized fuzzy clustering. Generic Fuzzy C-Means (FCM) and Gustafson–Kessel (GK) FCM are compared with two typical generalizations of kernel-based fuzzy clustering: one with prototypes located in the feature space (KFCM-F) and the other where the prototypes are distributed in the kernel space (KFCM-K). Both generalizations are studied when dealing with the Gaussian kernel while KFCM-K is also studied with the polynomial kernel. Two criteria are used in evaluating the performance of the clustering method and the resulting clusters, namely classification rate and reconstruction error. Through carefully selected experiments involving synthetic and Machine Learning repository (http://archive.ics.uci.edu/beta/) data sets, we demonstrate that the kernel-based FCM algorithms produce a marginal improvement over standard FCM and GK for most of the analyzed data sets. It has been observed that the kernel-based FCM algorithms are in a number of cases highly sensitive to the selection of specific values of the kernel parameters.
© 2009 Elsevier B.V. All rights reserved.

Keywords: Fuzzy clustering; Kernels; Fuzzy kernel-based clustering; Fuzzy C-Means; Gustafson–Kessel FCM; Evaluation criteria; Classification rate; Reconstruction error

1. Introduction

Fuzzy clustering has emerged as an important tool for discovering the structure of data. Kernel methods have been applied to fuzzy clustering and the kernelized version is referred to as kernel-based fuzzy clustering. There are two major variations of kernel-based fuzzy clustering: one involves keeping prototypes in the feature space while the other completes an inverse mapping of prototypes from the kernel space to feature space. They are given the acronyms KFCM-F and KFCM-K, respectively. An interesting research quest arises as to the performance of kernel based clustering vis-à-vis some well known standards such as FCM and GK. Given the substantially higher computing overhead of kernelized fuzzy clustering, it becomes of interest to assess to which extent they bring tangible benefits in terms of

* Corresponding author. Tel.: +1 780 492 4661; fax: +1 780 492 1811.
E-mail addresses: dgraves@ualberta.ca (D. Graves), pedrycz@ece.ualberta.ca (W. Pedrycz).

0165-0114/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.fss.2009.10.021

the quality of the produced results. The key objectives for the comparative experimental study of kernel-based fuzzy clustering are to quantify performance and assess parametric sensitivity of these methods. To offer an unbiased and fairly comprehensive evaluation of the results of clustering, we propose two criteria. The first one, which uses a classification rate, emphasizes the quality of clustering in the sense of "purity" of groups formed in the clustering process. Given the fact that this criterion requires labels of patterns, we may stress its *external* character. The second criterion concerns a reconstruction error and could be sought as an *internal* measure of the quality of the constructed clusters.

The paper is organized as follows. We start (Section 2) with a general overview of the use of kernels in clustering (kernelization). A background of GK is given in Section 3. Kernel-based fuzzy clustering is described in Section 4 including a brief background in kernels and kernel functions. A formal description of the evaluation criteria given in Section 5 includes classification rate and reconstruction error. The experimental results and comparative analysis are given in Section 6 followed by the main conclusions presented in Section 7.

Throughout the study, we use the standard notation as encountered in fuzzy clustering. A finite collection of N patterns $\mathbf{x} \in \mathbf{R}^d$ of dimensionality d is clustered into c groups. The structure in data is described by c prototypes $\mathbf{v}_i \in \mathbf{R}^d$, $i = 1, 2, \ldots, c$ and a fuzzy partition matrix $\mathbf{U}$. The notation $\| \;\|$ denotes the Euclidean distance.

2. Kernelization of clustering and fuzzy clustering: an overview

Recent years have seen a substantial level of interest in kernel-oriented clustering. As we indicated in the previous section, in this study we are predominantly interested in the kernel-based augmentation of objective function-based fuzzy clustering such as Fuzzy C-Means (FCM) [4] and its generalization such as Gustafson–Kessel (GK) FCM [2,13,17,18]. The choice of this category of fuzzy clustering is strongly motivated by several factors. First, both FCM and GK come with well-developed algorithmic underpinnings. Second, they can be encountered in various areas of applications. Third, a number of fuzzy clustering algorithms including FCM and GK have been developed and are widely available. Recently kernel-based clustering has been constructed and gained some popularity, cf. [3,5,6,8,10,11,15,16,23,25,27–32].

Kernel-based clustering techniques based on K-Means clusters have been developed in [7,9,16]. The authors of [16] performed a comparative analysis between K-Means and FCM clustering algorithms and their kernelized versions considering 10 publicly available data sets including the iris data set [21]. They concluded that kernel K-Means and kernel FCM perform similarly in terms of classification quality (classification rates) when using a Gaussian kernel function and generally perform better than their "standard" counterparts. They also showed that for the iris data there is an improvement of nearly 6% for kernel K-Means over standard K-Means and an improvement of about 4% for kernel FCM over the generic FCM in terms of classification rate. They reported significant improvement in the classification rate for the synthetic circle data set when using kernel-based K-Means and FCM clustering with the Gaussian kernel function. The authors in [9] reported a classification rate of 89% for the K-Means algorithm on the iris data set when averaged over 20 runs and a classification rate of about 95% using a Gaussian kernel. The study also indicated a very limited improvement in classification rate for kernel K-Means over standard K-Means for the UCI Wisconsin data set and spam data set.

Kernel-based hierarchical clustering was applied to microarray expression data in [23] with hopes of improving clustering performance. J. Qin et al. found little improvement in classification rate using kernel-hierarchical clustering on microarray expression data and concluded that kernels generally do not improve the results of clustering algorithms on the gene expression data under investigation.

Kernel-based fuzzy clustering techniques based on FCM have been developed in [6,8,16,25,26,28–32]. There are two general categories of kernel FCM algorithms present in the literature. The first class of kernel FCM algorithms which will be given the acronym KFCM-F (Kernel-based FCM with prototypes in Feature space) was found to be chiefly used in clustering incomplete data [25,29]. KFCM-F retains the prototypes in the feature space during clustering. Authors in [29] used kernel fuzzy-clustering for data imputation in the iris data set by discarding attributes randomly. When doing that they reported marginal improvements in the misclassification rate over some other imputation methods. A weighted kernel-based fuzzy-clustering technique for both complete and incomplete data sets was developed in [25] and the authors obtained about 96% classification rate on the iris data set. They also proved convergence of their algorithm. Their cluster prototypes are quite similar to standard FCM for the iris data set and they report a misclassification rate that is slightly better than standard FCM and other fuzzy-clustering algorithms.

A second class of kernel FCM algorithms was found [8,28,30,32] and will be given the acronym KFCM-K (Kernel-based FCM with prototypes in Kernel space). KFCM-K implicitly leaves the prototypes in the kernel space during clustering thus an inverse mapping must be performed to obtain prototypes in the feature space. There are several papers [28,30,32] that briefly report the performance of this kernel fuzzy-clustering algorithm. An artificial ring data set from the DELVE repository (http://www.cs.toronto.edu/∼delve/data/ringnorm) is used in [32] and achieves about 98% classification rate with the Gaussian kernel, 96% with the polynomial kernel, and 80% with standard FCM. Reference [32] also claims that the polynomial kernel is able to detect parabolic shaped clusters and the Gaussian kernel is able to detect line clusters, however, specific results are not provided. Authors in [28] reported good performance of the algorithm on a 2-dimensional non-linearly separable synthetic data set and compared the obtained results with those produced by the standard FCM; the classification rate for kernel FCM is much higher than standard FCM. Authors in [30] reported a marginal improvement in the classification rate for the iris data set while reporting significant improvements for an artificial ring data set. FCM achieved a classification rate of about 50% and the kernel FCM achieved a classification rate of about 100% for the ring data set in [30]. We might conclude that while in some cases reported in the literature there has been some improvement, the results do not seem quite conclusive. This is yet another compelling argument behind a careful quantitative investigation of kernelized versions of fuzzy clustering. Very preliminary findings for our experiments were first presented at the IFSA 2007 Congress [12]. They are substantially extended and strengthened in this paper.

3. Fuzzy clustering: a Gustafson–Kessel algorithm

The FCM algorithm reveals structure in data through a minimization of a quadratic objective function (performance index) [4,13]. A structure in data is represented in the form of a fuzzy partition matrix as well as a collection of "c" prototypes. Given the Euclidean distance, FCM favors spherical shapes of the clusters. Weighted Euclidean distance offers more flexibility by being able to search for ellipsoidal shapes whose main diagonals are parallel to the axes of the individual coordinates. Many variations of the FCM algorithm have been developed including the Gustafson–Kessel FCM, Fuzzy C-Means ellipsoids, and more recently kernel-based FCM. All of them form an attempt to support search for more complicated geometry of fuzzy clusters. For the sake of completeness, let us recall the essence of the GK. This clustering algorithm forms a generalization of the FCM algorithm by utilizing the Mahalanobis distance $\|\mathbf{x}_k - \mathbf{v}_i\|^2_{\mathbf{A}_i} = (\mathbf{x}_k - \mathbf{v}_i)^{\mathrm{T}}\mathbf{A}_i(\mathbf{x}_k - \mathbf{v}_i)$ where $\mathbf{A}_i$ is a positive definite matrix. The Gustafson–Kessel FCM minimizes the following objective function:

$$\begin{aligned} Q &= \sum_{i=1}^{c}\sum_{k=1}^{N} u_{ik}^m \|\mathbf{x}_k - \mathbf{v}_i\|^2_{\mathbf{A}_i} \\ &= \sum_{i=1}^{c}\sum_{k=1}^{N} u_{ik}^m (\mathbf{x}_k - \mathbf{v}_i)^{\mathrm{T}}\mathbf{A}_i(\mathbf{x}_k - \mathbf{v}_i) \end{aligned} \tag{1}$$

The optimization of the performance index Q is completed subject to the standard constraints we encounter in fuzzy clustering, that is

$$u_{ik} \in [0,1] \quad \forall i,k \tag{2}$$

$$0 < \sum_{k=1}^{N} u_{ik} < N \quad \forall i$$

$$\sum_{i=1}^{c} u_{ik} = 1 \quad \forall k$$

The matrix $\mathbf{A}_i$ is calculated based on the inverse of fuzzy covariance matrix $\mathbf{C}_i$ for $i = 1, 2, \ldots, c$ using the following expression:

$$\mathbf{A}_i = (\rho_i|\mathbf{C}_i|)^{1/d}\mathbf{C}_i^{-1} \tag{3}$$

where ρ_i is a positive scaling parameter and the determinant of $\mathbf{C}_i$ is expressed as $|\mathbf{C}_i|$. Commonly in the literature we encounter a constraint of the form $\rho_i = 1$. The calculations of the fuzzy covariance matrix are governed by the expression

$$\mathbf{C}_i = \frac{\sum_{k=1}^{N} u_{ik}^m (\mathbf{x}_k - \mathbf{v}_i)(\mathbf{x}_k - \mathbf{v}_i)^{\mathrm{T}}}{\sum_{k=1}^{N} u_{ik}^m} \tag{4}$$

The calculations of the prototypes are completed in the following form:

$$\mathbf{v}_i = \frac{\sum_{k=1}^{N} u_{ik}^m \mathbf{x}_k}{\sum_{k=1}^{N} u_{ik}^m} \tag{5}$$

The partition matrix is computed using the expression

$$u_{ik} = \frac{1}{\sum_{j=1}^{c} \left(\frac{(\mathbf{x}_k - \mathbf{v}_i)^{\mathrm{T}} \mathbf{A}_i (\mathbf{x}_k - \mathbf{v}_i)}{(\mathbf{x}_k - \mathbf{v}_j)^{\mathrm{T}} \mathbf{A}_j (\mathbf{x}_k - \mathbf{v}_j)} \right)^{1/(m-1)}} \tag{6}$$

The optimization algorithm is performed by starting with a random fuzzy partition matrix and iteratively updating the prototypes by (5) and the fuzzy partition matrix using (6). The algorithm is very similar to the standard FCM algorithm. We iterate (5) and (6) until the changes in the fuzzy partition matrix are very small or some other stopping criterion has been met. The complexity of GK for computing the prototypes is $O(cNd)$ and the complexity for computing the partition matrix is $O(cNd^2)$ at each iteration. The complexity of computing $\mathbf{A}_i$ depends on the method of matrix inversion used in the implementation of the clustering algorithm.

4. Kernel-based FCM

The essence of kernel-based methods involves performing an arbitrary non-linear mapping Φ from the original d-dimensional feature space $\mathbf{R}^d$ to a space of higher dimensionality (kernel space), cf. [14,20,24]. The kernel space could possibly be of infinite dimensionality. The rationale for going to higher dimensions is that it may be possible to apply a linear classifier in the kernel space while the original problem in the feature space could be highly non-linear and not separable linearly [14,20]. The kernel method then takes advantage of the fact that dot products in the kernel space can be expressed by a Mercer kernel K given by $\mathrm{K}(\mathbf{x}, \mathbf{y}) \equiv \Phi(\mathbf{x})^{\mathrm{T}} \Phi(\mathbf{y})$ where $\mathbf{x}, \mathbf{y} \in \mathbf{R}^d$. Thus the distance in the kernel space does not have to be explicitly computed because it can be replaced by a Mercer kernel function (typically referred to as a kernel trick). A Mercer kernel function sometimes expressed as simply kernel function is defined as any positive semi-definite symmetric function. Commonly encountered kernel functions given in Table 1 include the Gaussian kernel, hyper-tangent, and polynomial kernel [14,20].

Given two kernel functions, $\mathrm{K}_1(\mathbf{x}, \mathbf{y})$ and $\mathrm{K}_2(\mathbf{x}, \mathbf{y})$, one can show that $\mathrm{K}_1(\mathbf{x}, \mathbf{y}) + \mathrm{K}_2(\mathbf{x}, \mathbf{y})$, $\mathrm{K}_1(\mathbf{x}, \mathbf{y})\mathrm{K}_2(\mathbf{x}, \mathbf{y})$ and $a\mathrm{K}_1(\mathbf{x}, \mathbf{y})$ where $a > 0$ are also kernels [14].

4.1. KFCM-F algorithm

There are two major forms of kernel-based fuzzy clustering. The first one comes with prototypes constructed in the feature space. These clustering methods will be referred to as KFCM-F (with F standing for the feature space). In the second category, abbreviated as KFCM-K, the prototypes are retained in the kernel space and thus the prototypes must be approximated in the feature space by computing an inverse mapping from kernel space to feature space.

Table 1
A list of commonly encountered kernel functions.

Type of Kernel function	Expression
Gaussian	$e^{-\Vert\mathbf{x}-\mathbf{y}\Vert^2/\sigma^2}$, $\sigma^2 > 0$
Polynomial	$(\mathbf{x}^{\mathrm{T}}\mathbf{y} + \theta)^p$, $\theta \geq 0$, $p \in N$
Hyper-tangent	$\tanh(\mathbf{x}^{\mathrm{T}}\mathbf{y} + \theta)$, $\theta \geq 0$

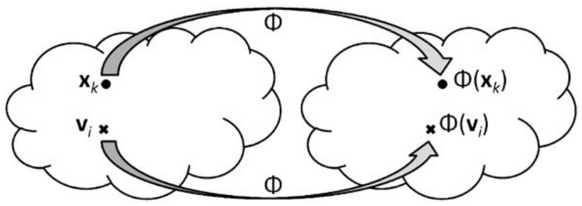

Fig. 1. KFCM-F feature space and kernel space.

KFCM-F minimizes the following objective function subject to the same constraints as both FCM and GK given in (2) [25,29]:

$$Q = \sum_{i=1}^{c}\sum_{k=1}^{N} u_{ik}^{m}\|\Phi(\mathbf{x}_k) - \Phi(\mathbf{v}_i)\|^2 \tag{7}$$

The advantage of the KFCM-F clustering algorithm is that the prototypes reside in the feature space and are implicitly mapped to the kernel space through the use of the kernel function as depicted in Fig. 1.

By constraining ourselves to the Euclidean distance in $\|\Phi(\mathbf{x}_k) - \Phi(\mathbf{v}_i)\|$, the squared distance is computed in the kernel space using a kernel function such that [25,29]

$$\begin{aligned}\|\Phi(\mathbf{x}_k) - \Phi(\mathbf{v}_i)\|^2 &= \Phi(\mathbf{x}_k)^{\mathrm{T}}\Phi(\mathbf{x}_k) + \Phi(\mathbf{v}_i)^{\mathrm{T}}\Phi(\mathbf{v}_i) - 2\Phi(\mathbf{x}_k)^{\mathrm{T}}\Phi(\mathbf{v}_i)\\ &= \mathrm{K}(\mathbf{x}_k, \mathbf{x}_k) + \mathrm{K}(\mathbf{v}_i, \mathbf{v}_i) - 2\mathrm{K}(\mathbf{x}_k, \mathbf{v}_i)\end{aligned} \tag{8}$$

If we confine ourselves to the Gaussian kernel which is used almost exclusively in the literature, then $\mathrm{K}(\mathbf{x}, \mathbf{x}) = 1$ and $\|\Phi(\mathbf{x}_k) - \Phi(\mathbf{v}_i)\|^2 = 2(1 - \mathrm{K}(\mathbf{x}_k, \mathbf{v}_i))$.

The optimization of the partition matrix $\mathbf{U}$ involves the use of the technique of Lagrange multipliers which leads to the expression

$$u_{ik} = \frac{1}{\sum_{j=1}^{c}\left(\frac{1-\mathrm{K}(\mathbf{x}_k,\mathbf{v}_i)}{1-\mathrm{K}(\mathbf{x}_k,\mathbf{v}_j)}\right)^{1/(m-1)}} \tag{9}$$

for $i = 1, 2, \ldots, c$ and $k = 1, 2, \ldots, N$ [25,29]. The derivation of the prototypes depends on the specific selection of the kernel function. The calculation of the prototypes $\mathbf{v}_i$ for $i = 1, 2, \ldots, c$ with the Gaussian kernel proceeds as follows:

$$\nabla_{\mathbf{v}_i} Q = \mathbf{0}$$

$$\sum_{k=1}^{N} u_{ik}^{m}\nabla_{\mathbf{v}_i}(\|\Phi(\mathbf{x}_k) - \Phi(\mathbf{v}_i)\|^2) = \mathbf{0}$$

$$\sum_{k=1}^{N} u_{ik}^{m}\nabla_{\mathbf{v}_i}(2 - 2\mathrm{K}(\mathbf{x}_k, \mathbf{v}_i)) = \mathbf{0}$$

$$\sum_{k=1}^{N} u_{ik}^{m}\nabla_{\mathbf{v}_i}(e^{-\|\mathbf{x}_k-\mathbf{v}_i\|^2/\sigma^2}) = \mathbf{0}$$

$$\sum_{k=1}^{N} u_{ik}^{m}(e^{-\|\mathbf{x}_k-\mathbf{v}_i\|^2/\sigma^2})\left(\frac{2}{\sigma^2}(\mathbf{x}_k - \mathbf{v}_i\right)) = \mathbf{0}$$

$$\mathbf{v}_i\sum_{k=1}^{N} u_{ik}^{m}\mathrm{K}(\mathbf{x}_k, \mathbf{v}_i) = \sum_{k=1}^{N} u_{ik}^{m}\mathrm{K}(\mathbf{x}_k, \mathbf{v}_i)\mathbf{x}_k \tag{10}$$

$$\mathbf{v}_i = \frac{\sum_{k=1}^{N} u_{ik}^{m}\mathrm{K}(\mathbf{x}_k, \mathbf{v}_i)\mathbf{x}_k}{\sum_{k=1}^{N} u_{ik}^{m}\mathrm{K}(\mathbf{x}_k, \mathbf{v}_i)} \tag{11}$$

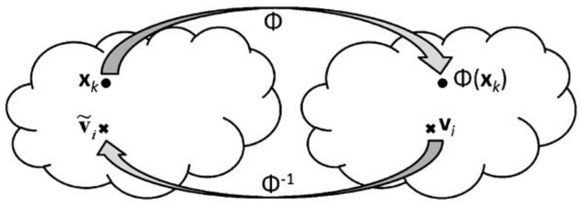

Fig. 2. KFCM-K feature space and kernel space.

The algorithm for KFCM-F is very similar to standard FCM by starting with a random partition matrix and iteratively updating the prototypes using expression (11) and updating the membership partition matrix using expression (9) until some stopping criterion has been met.

If the kernel matrix between patterns (data) and prototypes is computed at the beginning of each iteration, the computational complexity of the partition matrix is $O(cNd)$ and the prototypes is $O(cNd)$. At each iteration, the kernel matrix requires cN kernel function evaluations thus the computational complexity of the KFCM-F algorithm is $O(cNd)$ at each iteration. Convergence of KFCM-F is proved in [25]. Table 2(a) visualizes a detailed processing flow of the KFCM-F algorithm.

4.2. KFCM-K algorithm

The KFCM-K kernel-based fuzzy clustering algorithm [8,28,30,32] is the second commonly used kernel fuzzy clustering algorithm found in literature where the prototypes are located in the kernel space. The KFCM-K algorithm minimizes the following objective function [8,28,30,32]:

$$Q = \sum_{i=1}^{c}\sum_{k=1}^{N} u_{ik}^m \|\Phi(\mathbf{x}_k) - \mathbf{v}_i\|^2 \tag{12}$$

The advantage of KFCM-K is that the prototypes are not constrained to the feature space; however, the disadvantage is that the prototypes are *implicitly* located in the kernel space and thus need to be approximated by an inverse mapping to the feature space, see Fig. 2.

Given the Euclidean distance and optimizing Q with respect to $\mathbf{v}_i$ located in the kernel space such that $\nabla_{\mathbf{v}_i} Q = \mathbf{0}$, we obtain [8,28,30,32]

$$\mathbf{v}_i = \frac{\sum_{k=1}^{N} u_{ik}^m \Phi(\mathbf{x}_k)}{\sum_{k=1}^{N} u_{ik}^m} \tag{13}$$

The computation of the partition matrix for $i = 1, 2, \ldots, c$ and $k = 1, 2, \ldots, N$ involves the well-known constraints (2) which yields the following partition matrix [8,28,30,32]:

$$u_{ik} = \frac{1}{\sum_{j=1}^{c}\left(\frac{\|\Phi(\mathbf{x}_k)-\mathbf{v}_i\|}{\|\Phi(\mathbf{x}_k)-\mathbf{v}_j\|}\right)^{2/(m-1)}} \tag{14}$$

In order to compute the distance standing in (14) we note that [8,28,30,32]

$$\begin{aligned}\|\Phi(\mathbf{x}_k) - \mathbf{v}_i\|^2 &= (\Phi(\mathbf{x}_k) - \mathbf{v}_i)^{\mathrm{T}}(\Phi(\mathbf{x}_k) - \mathbf{v}_i)\\ &= \Phi(\mathbf{x}_k)^{\mathrm{T}}\Phi(\mathbf{x}_k) - 2\Phi(\mathbf{x}_k)^{\mathrm{T}}\mathbf{v}_i + \mathbf{v}_i^{\mathrm{T}}\mathbf{v}_i\end{aligned} \tag{15}$$

Inserting the expression for the prototypes (13) into the expression for distance (15) gives [8,28,30,32]

$$\|\Phi(\mathbf{x}_k) - \mathbf{v}_i\|^2 = \mathrm{K}(\mathbf{x}_k, \mathbf{x}_k) - 2\frac{\sum_{j=1}^{N} u_{ij}^m \mathrm{K}(\mathbf{x}_k, \mathbf{x}_j)}{\sum_{j=1}^{N} u_{ij}^m} + \frac{\sum_{j=1}^{N}\sum_{l=1}^{N} u_{ij}^m u_{il}^m \mathrm{K}(\mathbf{x}_j, \mathbf{x}_l)}{\left(\sum_{j=1}^{N} u_{ij}^m\right)^2} \tag{16}$$

The method outlined in [32] iteratively determines approximate prototypes $\tilde{\mathbf{v}}_i$ in the feature space by inverse mapping as shown in Fig. 2. The objective function to be minimized reads as

$$V = \sum_{i=1}^{c} \|\Phi(\tilde{\mathbf{v}}_i) - \mathbf{v}_i\|^2 = \sum_{i=1}^{c} (\Phi(\tilde{\mathbf{v}}_i)^{\mathrm{T}}\Phi(\tilde{\mathbf{v}}_i) - 2\Phi(\tilde{\mathbf{v}}_i)^{\mathrm{T}}\mathbf{v}_i + \mathbf{v}_i^{\mathrm{T}}\mathbf{v}_i) \tag{17}$$

By making use of the expression for prototypes positioned in the feature space (13), the objective function to minimize is [32]

$$V = \sum_{i=1}^{c} \left(\mathrm{K}(\tilde{\mathbf{v}}_i, \tilde{\mathbf{v}}_i) - 2\frac{\sum_{k=1}^{N} u_{ik}^m \mathrm{K}(\mathbf{x}_k, \tilde{\mathbf{v}}_i)}{\sum_{j=1}^{N} u_{ij}^m} + \frac{\sum_{j=1}^{N}\sum_{l=1}^{N} u_{ij}^m u_{il}^m \mathrm{K}(\mathbf{x}_j, \mathbf{x}_l)}{\left(\sum_{j=1}^{N} u_{ij}^m\right)^2} \right) \tag{18}$$

Solving $\nabla_{\tilde{\mathbf{v}}_i} V = \mathbf{0}$ requires knowledge of the kernel function K. In what follows, the prototype expressions will be derived for both Gaussian and polynomial kernels. Since $\mathrm{K}(\mathbf{x}_j, \mathbf{x}_l)$ is independent of $\tilde{\mathbf{v}}_i$, $\nabla_{\tilde{\mathbf{v}}_i}\mathrm{K}(\mathbf{x}_j, \mathbf{x}_l) = \mathbf{0}$. Given the Gaussian kernel, $\mathrm{K}(\tilde{\mathbf{v}}_i, \tilde{\mathbf{v}}_i) = 1$ is independent of $\tilde{\mathbf{v}}_i$ thus

$$\begin{aligned}
\nabla_{\tilde{\mathbf{v}}_i} V &= \mathbf{0} \\
&= \mathbf{0} - 2\frac{\sum_{k=1}^{N} u_{ik}^m \nabla_{\tilde{\mathbf{v}}_i}\mathrm{K}(\mathbf{x}_k, \tilde{\mathbf{v}}_i)}{\sum_{j=1}^{N} u_{ij}^m} + \mathbf{0} \\
&= \sum_{k=1}^{N} u_{ik}^m e^{-\|\mathbf{x}_k - \tilde{\mathbf{v}}_i\|^2/\sigma^2} \left(\frac{2}{\sigma^2}(\mathbf{x}_k - \tilde{\mathbf{v}}_i)\right) \\
\tilde{\mathbf{v}}_i \sum_{k=1}^{N} u_{ik}^m \mathrm{K}(\mathbf{x}_k, \tilde{\mathbf{v}}_i) &= \sum_{k=1}^{N} u_{ik}^m \mathrm{K}(\mathbf{x}_k, \tilde{\mathbf{v}}_i)\mathbf{x}_k
\end{aligned} \tag{19}$$

The prototype expression for the Gaussian kernel for $i = 1, 2, \ldots, c$ is then given as [32]

$$\tilde{\mathbf{v}}_i = \frac{\sum_{k=1}^{N} u_{ik}^m \mathrm{K}(\mathbf{x}_k, \tilde{\mathbf{v}}_i)\mathbf{x}_i}{\sum_{k=1}^{N} u_{ik}^m \mathrm{K}(\mathbf{x}_k, \tilde{\mathbf{v}}_i)} \tag{20}$$

Considering the polynomial kernel, we obtain

$$\begin{aligned}
\nabla_{\tilde{\mathbf{v}}_i} V &= \mathbf{0} \\
&= \nabla_{\tilde{\mathbf{v}}_i}\mathrm{K}(\tilde{\mathbf{v}}_i, \tilde{\mathbf{v}}_i) - 2\frac{\sum_{k=1}^{N} u_{ik}^m \nabla_{\tilde{\mathbf{v}}_i}\mathrm{K}(\mathbf{x}_k, \tilde{\mathbf{v}}_i)}{\sum_{j=1}^{N} u_{ij}^m} + \mathbf{0} \\
&= 2p\tilde{\mathbf{v}}_i(\tilde{\mathbf{v}}_i^{\mathrm{T}}\tilde{\mathbf{v}}_i + \theta)^{p-1} - \frac{2p}{\sum_{j=1}^{N} u_{ij}^m} \sum_{k=1}^{N} u_{ik}^m \mathbf{x}_k (\mathbf{x}_k^{\mathrm{T}}\tilde{\mathbf{v}}_i + \theta)^{p-1}
\end{aligned} \tag{21}$$

The prototype expression for the polynomial kernel for $i = 1, 2, \ldots, c$ is thus [32]

$$\tilde{\mathbf{v}}_i = \frac{\sum_{k=1}^{N} u_{ik}^m (\mathbf{x}_k^{\mathrm{T}}\tilde{\mathbf{v}}_i + \theta)^{p-1}\mathbf{x}_k}{(\tilde{\mathbf{v}}_i^{\mathrm{T}}\tilde{\mathbf{v}}_i + \theta)^{p-1}\sum_{j=1}^{N} u_{ij}^m} \tag{22}$$

The prototypes are computed iteratively using a Kernel-dependant formula such as the ones given for Gaussian kernels or polynomial kernels after the evaluation of the fuzzy partition matrix.

The complexity of computing the partition matrix at each iteration is $O(cN^2d)$ assuming the kernel matrix is computed only once at the very beginning of the program. The complexity of the one-time computation of the kernel matrix is $O(N^2d)$. The complexity of computing the prototypes at each iteration is $O(cNd)$. The Table 2(b) describes the KFCM-K algorithm in more detail.

Table 2
Kernel-based clustering algorithms.

KFCM-F algorithm (a)	KFCM-K algorithm (b)
Input: data **X**, kernel function K, *c*, *m*	*Input*: data **X**, kernel function K, *c*, *m*
Output: fuzzy partition **U**, prototypes **V**	*Output*: fuzzy partition **U**, prototypes **V**
Algorithm:	Algorithm:
Initialize **U** to random fuzzy partition	Initialize **U** to random fuzzy partition
Do	Do
Update **V** according to (11) for Gaussian kernels	Update **U** according to (14) Until terminationcriteria satisfied
Update **U** according to (9)	Do
Until termination criteria satisfied or maximum iterations reached Return **U** and **V**	Update **V** according to (20) for Gaussian kernelsor (22) for polynomial kernels
	Until termination criteria satisfied or maximum iterations reached Return **U** and **V**

Table 3
Computational complexity.

Algorithm	Overall complexity
FCM	$O(cNd)$
GK	$O(cNd^2)$
KFCM-F	$O(cNd)$
KFCM-K	$O(cN^2d)$

4.3. Computational complexity

A comparison of the computation complexity of a single iteration for the different clustering algorithms is given in Table 3.

The complexity for the kernel algorithms depends on the kernel used. The complexity given in Table 3 is provided for the Gaussian and polynomial kernels. It is also assumed that the kernel matrix is computed once rather than at every iteration for the KFCM-K algorithm.

5. Evaluation criteria

There are two performance criteria that will be used to evaluate the performance of the clustering algorithms in this paper: classification rate and reconstruction error. The first depends on the correct classes being given while the second is a performance measure independent of the correct classes that measures the reconstructability of patterns in a cluster using a collection of prototypes and cluster membership.

5.1. Classification rate

The classification rate is a common measure used to determine how well clustering algorithms perform on a data with a known structure (i.e., classes). The classification rate is determined first by transforming the fuzzy partition matrix into a Boolean partition matrix and by selecting the cluster with the maximum membership value for each pattern. Class labels are assigned to each cluster according to the class that dominates that cluster. The classification rate is the percentage of patterns that belong to a correctly labeled cluster. The classification rate can be determined by building a contingency matrix [19]. Higher classification rates indicate better clustering results. The classification rate is quite often used to measure the performance of fuzzy clustering algorithms.

5.2. Reconstruction error

The reconstruction error is a fuzzy-based measure of fuzzy clustering algorithms by reconstructing patterns using the prototypes and cluster membership values, cf. [22]. The process involves reconstructing patterns based on the codebook (i.e., a collection of already constructed prototypes) and the computed membership values of the original patterns to the elements of the codebook [22]. The reconstructed patterns $\tilde{\mathbf{x}}_k$, $k = 1, 2, \ldots, N$ are calculated as follows:

$$\tilde{\mathbf{x}}_k = \frac{\sum_{i=1}^{c} u_{ik}^m \mathbf{v}_i}{\sum_{i=1}^{c} u_{ik}^m} \tag{23}$$

The error between the reconstructed pattern and the original one quantifies the quality of reconstruction provided by the fuzzy clustering (and a collection of the prototypes, in particular). The reconstruction error is expressed in the form of the following sum of distances

$$r = \frac{\sum_{k=1}^{N} \|\mathbf{x}_k - \tilde{\mathbf{x}}_k\|^2}{N} \tag{24}$$

Smaller values for the reconstruction error "r" indicate better clustering results (in terms of the reconstruction criterion introduced above). Clearly lower number of prototypes implies higher values of the reconstruction error. As the number of prototypes gets closer to the number of patterns the reconstruction error gets closer to zero. The reconstruction error is unlike the classification rate since it does not measure accuracy. Instead the reconstruction error measures the encoding and decoding performance of the patterns with respect to their prototypes and membership values. Visually, the reconstruction error is a measure for the spread of the prototypes across the feature space. In particular, the reconstruction error decreases as prototypes are situated centrally within dense regions of patterns in the feature space. Hence, reconstruction error measures the quality of the prototypes with regards to its representation of the data in terms of clusters.

6. Experimental studies

A series of experiments were run for a variety of data sets using the standard FCM, Gustafson–Kessel FCM and Kernel-based FCM. The objective of this comprehensive suite of experiments is to come up with a thorough comparison of the performance of the algorithms and their kernelized variants. Many two-dimensional synthetic data sets were used; see Fig. 3 in section A. A number of data sets from the UCI Machine Learning databases [1,21] were also used described more fully in section B. All data sets are normalized to have zero mean and unit standard deviation.

The clustering algorithms were run over different values for the clustering parameters including the number of clusters c, the fuzzification coefficient m, and the corresponding kernel parameters. The number of clusters c was varied over the values 2, 3, and 4 for the simpler fuzzy X, parabolic and ring data sets. The number of clusters c was taken from the set $\{2, 3, 4, 5, 6\}$ for the rest of the synthetic data sets. If the data set contained more than 4 classes as was sometimes the case with UCI machine learning data sets, the number of clusters c was set to the number of classes. The fuzzification coefficient m was varied over the pre-selected set of values $\{1.2, 1.4, 1.7, 2.0, 2.5, 3.0\}$ for all but the computationally intensive kernel-based algorithms. The fuzzification coefficient m was varied over the values $\{1.4, 2.0, 2.5\}$ to reduce the number of required runs for the computationally intensive kernel-based clustering algorithms. The Gaussian kernel parameter σ^2 was varied over $\{0.01, 0.05, 0.1, 0.25, 0.5, 1, 2, 4, 8, 12, 16, 20, 25, 30, 40, 50, 75, 100\}$. The polynomial kernel parameters (θ, d) were varied over the Cartesian product of the set of θ values $\{0, 2, 4, 7, 10, 15, 20, 30, 40, 50\}$ and the set of power d values $\{2, 4, 8, 12, 16\}$. For each set of parameters FCM, Gustafson–Kessel FCM and KFCM-F algorithms were repeated 20 times and the means and standard deviations of the classification rate and reconstruction error were recorded. The initial membership matrix is generated randomly; hence, the standard deviation of the results quantifies the sensitivity of the algorithms to their initialization.

The value for ρ_i was set to 1 for $i = 1, 2, \ldots, c$. The values of the determinant of $\mathbf{C}_i$ are monitored so that when $|\mathbf{C}_i| < 10^{-10}$ then the covariance matrix is replaced by the identity matrix. The maximum number of iterations for the algorithms is 100; however, we allow the algorithm to stop when the following condition has been satisfied:

$$\sum_{i=1}^{c} \sum_{k=1}^{N} |u_{ik} - u'_{ik}| < 10^{-8} \tag{25}$$

where u'_{ik} is the partition matrix of the previous iteration. In many cases (25) is satisfied before the maximum number of iterations is reached.

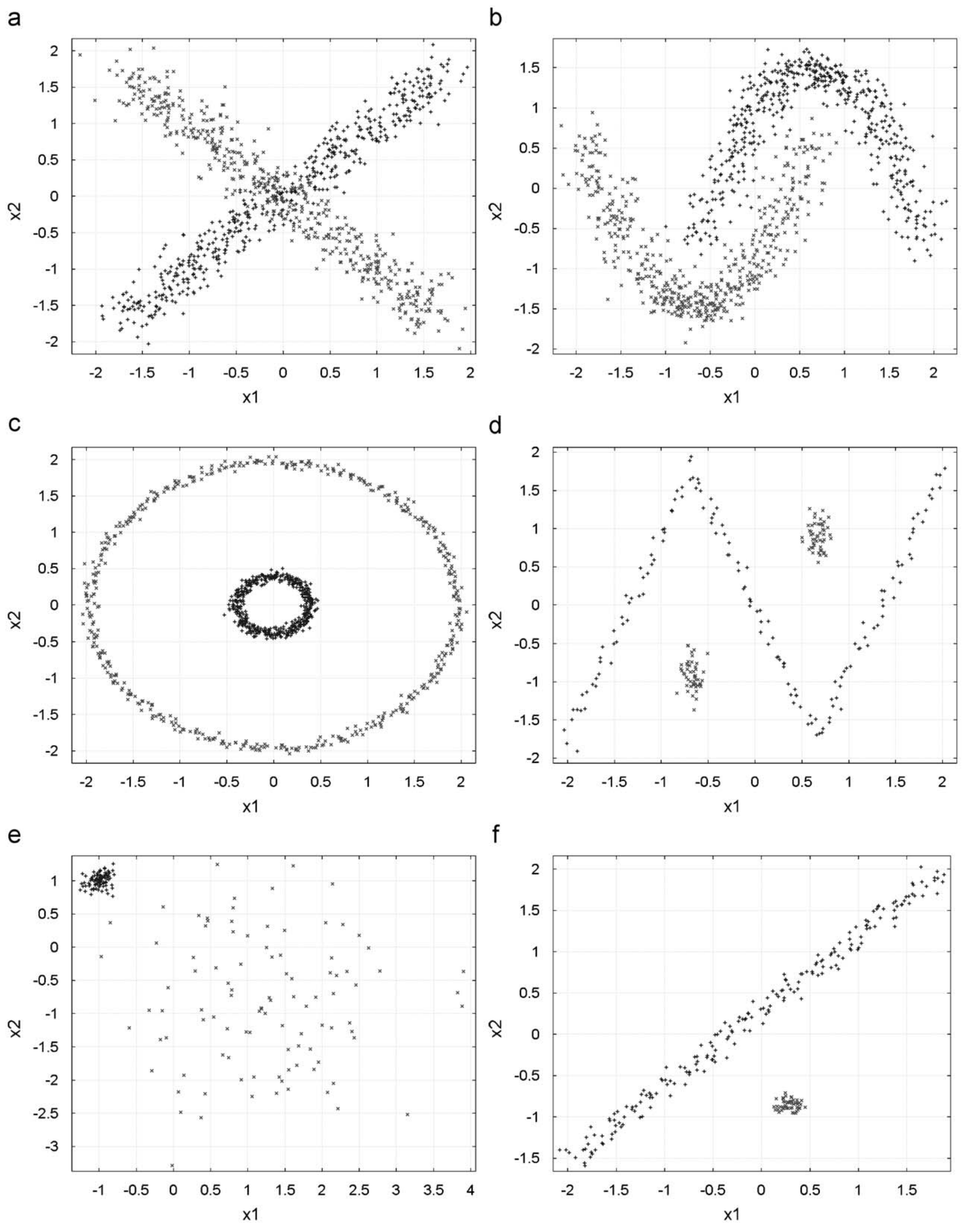

Fig. 3. Synthetic data sets. (a) Fuzzy "X". (b) Parabolic. (c) Ring. (d) Zig-zag (ZZ). (e) Dense (DE). (f) Line (LI). (g) Noisy Zig-zag (NZ). (h) Zig-zag with outliers (ZO). (i) Noisy ring (NR). (j) Ring with outliers (RO).

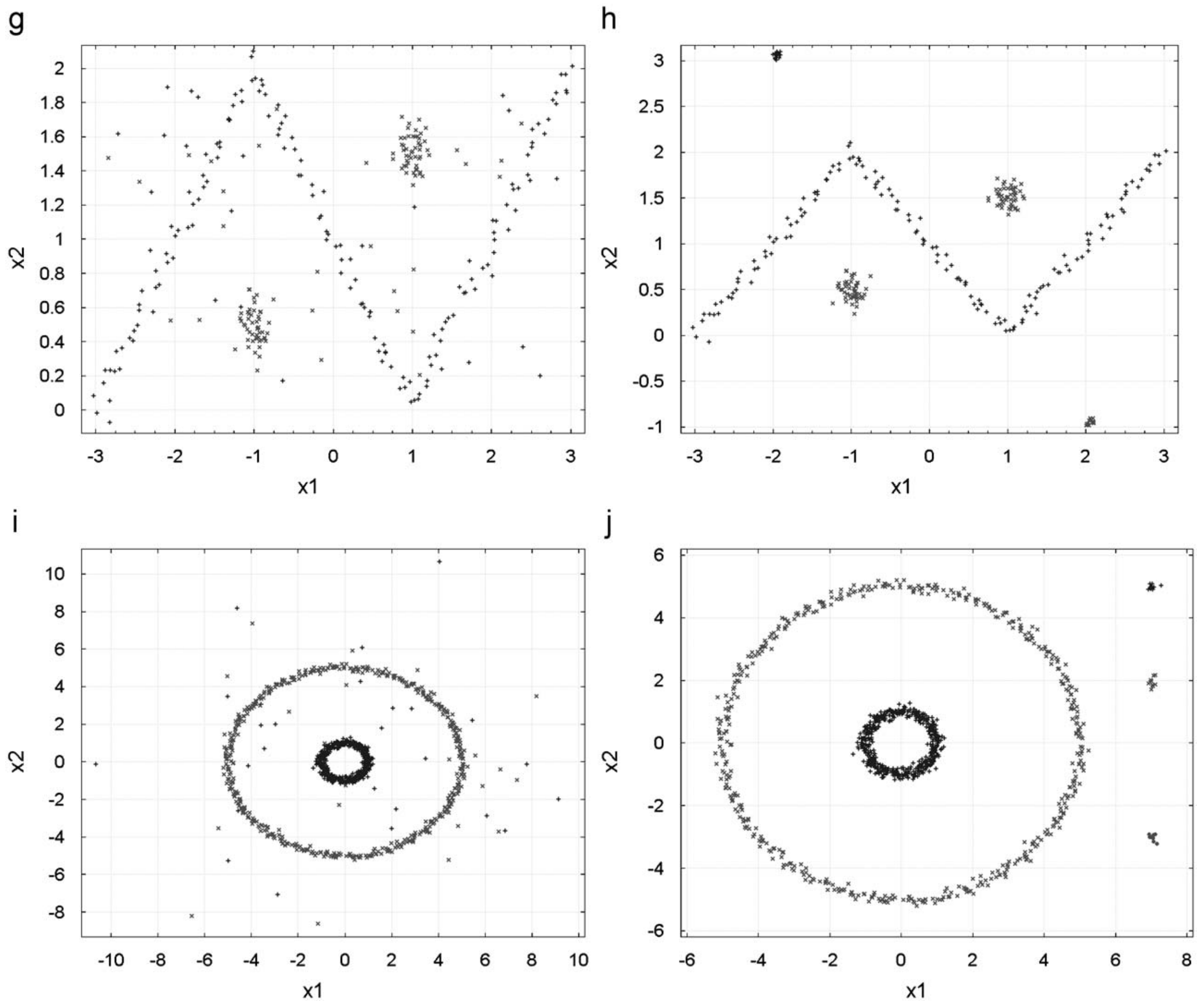

Fig. 3. *continued.*

6.1. *Synthetic data sets*

Ten synthetic data sets are shown their plots provided in Fig. 3.

Most of the synthetic data sets have class distributions that are equal except for the zig-zag and line data sets. The zig-zag data set has two classes where the first class (the zig-zag line) contains 150 points. The other two small circular-shaped clusters each of size 50 are in the second class for a total data size of 250. The line data set is highly skewed with the line class consisting of 200 points and the small circular-shaped cluster containing 50 points for a total of 250.

6.1.1. *Fuzzy X data set*

The results for the clustering of the "fuzzy" X data set show the Gustafson–Kessel (GK) FCM clustering algorithm is the clear winner in terms of classification rate when $c = 2$, however, as the number of clusters increase to 4, all clustering algorithms perform very similarly. Interestingly enough, the KFCM-K algorithm with polynomial kernel and the KFCM-F with Gaussian kernel perform fairly well for $c = 2$ with an average classification rate of about 70%.

Although GK FCM performs quite well on the "fuzzy" X data set, the cluster prototypes are at the center and thus the reconstruction error is fairly large. Gaussian KFCM-K exhibits rotational variations in the cluster boundaries for each run when their clustering parameters are the same. The KFCM-F algorithm was able to cluster about 70% of the

Table 4
Clustering results for “fuzzy” X data set (G—Gaussian and P—polynomial).

c	Clustering	Classification rate (%)	Reconstruction error
2	FCM ($m = 2.5$)	50.8 ± 0.4	1.498 ± 0.004
	GK ($m = 3$)	93.4 ± 0.0	1.998 ± 0.000
	KFCM-F (G) ($m = 2.5, \sigma^2 = 2$)	69.5 ± 0.8	1.779 ± 0.017
	KFCM-K (G) ($m = 1.4, \sigma^2 = 2$)	52.0 ± 0.7	1.923 ± 0.069
	KFCM-K (P) ($m = 2.5, \theta = 0, d = 2$)	75.6 ± 0.7	2.592 ± 0.635
3	FCM ($m = 3$)	82.2 ± 8.2	1.159 ± 0.070
	GK ($m = 1.2$)	93.3 ± 0.4	1.202 ± 0.004
	KFCM-F (G) ($m = 2, \sigma^2 = 25$)	79.8 ± 7.7	1.019 ± 0.129
	KFCM-K (G) ($m = 2.5, \sigma^2 = 8$)	80.1 ± 8.4	1.170 ± 0.048
	KFCM-K (P) ($m = 2.5, \theta = 50, d = 16$)	78.8 ± 8.4	1.281 ± 0.019
4	FCM ($m = 1.4$)	93.8 ± 0.1	0.3572 ± 0.0001
	GK ($m = 2.5$)	94.0 ± 0.0	0.2719 ± 0.0000
	KFCM-F (G) ($m = 2, \sigma^2 = 16$)	93.8 ± 0.0	0.3097 ± 0.0000
	KFCM-K (G) ($m = 2, \sigma^2 = 16$)	93.8 ± 0.0	0.3065 ± 0.0001
	KFCM-K (P) ($m = 1.4, \theta = 50, d = 2$)	93.6 ± 0.1	0.3532 ± 0.0006

data correctly for $c = 2$ by capturing three sides of the X in one cluster; thus, the algorithm was able to achieve an unbalanced optimal set of clusters. The kernel-based algorithms do not provide impressive results on the “fuzzy” X data set. The results for the “fuzzy” X data set are collected in Table 4.

Cluster boundaries shown in black are plotted in Fig. 4. The data set and the prototype points shown by larger points are plotted with the cluster boundaries. The cluster boundaries are reported for the optimal parameters as presented in Table 4.

6.1.2. Parabolic data set

The clustering results obtained for the parabolic data set is not very impressive for any of the clustering algorithms. It should be noted that the performance of the kernel-based algorithms varied quite a bit with respect to the parameters of the kernel (Table 5).

Overall, there does not seem to be significant improvement in the kernel-based clustering algorithms over standard FCM for this data set. The reconstruction error of the kernel clustering algorithms with the Gaussian kernel is fairly poor, especially for larger c, indicating decreasing cluster quality.

6.1.3. Ring data set

The ring data set is the typical data set used with kernel-based fuzzy clustering algorithms and tends to produce very good results with kernel clustering algorithms. The optimal reconstruction error of KFCM-K is almost the same as the optimal error for FCM but the classification rate is much higher for $c = 2$. The kernel-based algorithms provide non-spherical cluster shapes for this particular example. However, kernel parameter selection is very important. Although, KFCM-F is performing better than FCM it does not perform nearly as well as KFCM-K. KFCM-F was able to classify 100% of the ring data set for $c = 2$, however, the clustering results of KFCMK-F were not consistent (Table 6).

We include some plots of the performance measures versus clustering parameters for the ring data set. As Figs. 5 and 6 visualize, the kernel parameter selection is important for the ring data set. For the polynomial kernel a larger value for the power “d” gives higher values for the classification rates but results in higher values of the reconstruction errors. The value of “m” does not significantly affect the classification rate with the Gaussian kernel.

The kernel-based algorithms do not appear to provide an ideal solution to the problem of clustering and in particular with non-spherical shaped clusters. An example is the parabolic data set where all clustering algorithms perform similarly. Although the kernel-based algorithms tend to perform much better for the ring cluster, the performance greatly depends on the selection of the kernel parameters.

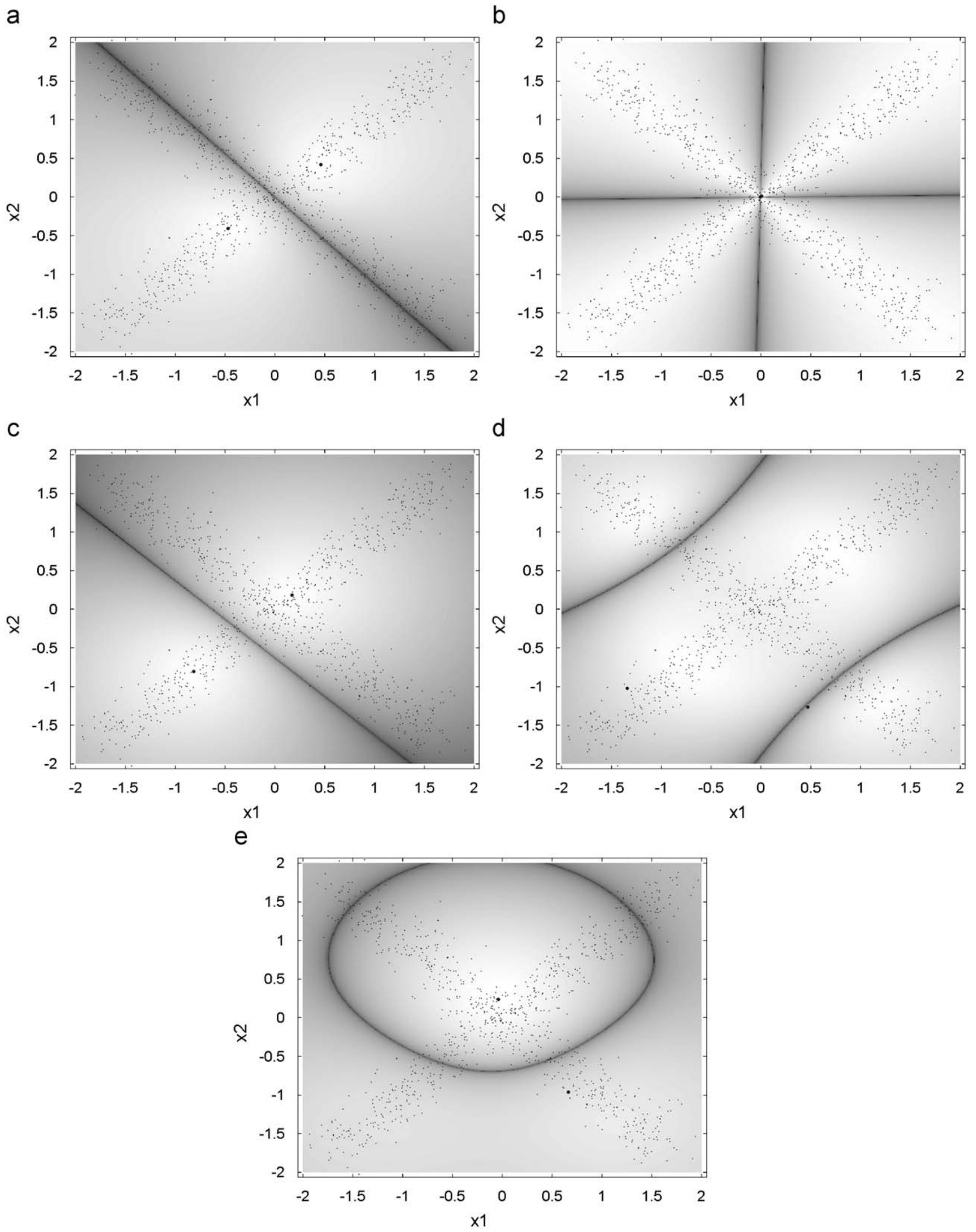

Fig. 4. Contour plots of membership functions produced by FCM: (a) ($m = 2.5$); (b) GK ($m = 3$); (c) KFCM-K for Gaussian ($m = 1.4$, $\sigma^2 = 2$); (d) KFCM-K for polynomial ($m = 2.5$, $\sigma = 0$, $d = 2$); and (e) KFCMK-F ($m = 2.5$, $\sigma^2 = 2$) for the "Fuzzy" X data set.

Table 5
Clustering results for parabolic data set (G—Gaussian and P—polynomial).

c	Clustering	Classification rate (%)	Reconstruction error
2	FCM ($m = 2.5$)	87.4 ± 0.0	0.6748 ± 0.0000
	GK ($m = 3$)	88.5 ± 0.0	0.7089 ± 0.0000
	KFCM-F (G) ($m = 1.4, \sigma^2 = 1$)	88.2 ± 0.0	0.7513 ± 0.0000
	KFCM-K (G) ($m = 2, \sigma^2 = 1$)	89.0 ± 0.0	0.9087 ± 0.0000
	KFCM-K (P) ($m = 2.5, \theta = 10, d = 12$)	87.8 ± 0.0	0.9622 ± 0.0674
3	FCM ($m = 1.2$)	86.9 ± 0.3	0.5261 ± 0.0139
	GK ($m = 1.4$)	84.2 ± 0.7	0.6199 ± 0.0540
	KFCM-F (G) ($m = 1.4, \sigma^2 = 100$)	86.6 ± 0.9	0.5127 ± 0.0482
	KFCM-K (G) ($m = 2.5, \sigma^2 = 1$)	89.0 ± 0.1	2.486 ± 0.536
	KFCM-K (P) ($m = 1.4, \theta = 10, d = 2$)	86.2 ± 0.9	0.5354 ± 0.0539
4	FCM ($m = 1.2$)	88.1 ± 0.0	0.3209 ± 0.0000
	GK ($m = 2.5$)	94.8 ± 3.6	0.4378 ± 0.0714
	KFCM-F (G) ($m = 1.4, \sigma^2 = 30$)	87.9 ± 0.0	0.2777 ± 0.0000
	KFCM-K (G) ($m = 2.5, \sigma^2 = 1$)	89.0 ± 0.1	2.421 ± 0.532
	KFCM-K (P) ($m = 2.5, \theta = 20, d = 16$)	88.5 ± 0.0	0.9879 ± 0.2036

Table 6
Clustering results for ring data set (G—Gaussian and P—polynomial).

c	Clustering	Classification rate (%)	Reconstruction error
2	FCM ($m = 1.2$)	51.5 ± 0.6	1.349 ± 0.004
	GK ($m = 1.2$)	52.5 ± 1.9	1.359 ± 0.004
	KFCM-F (G) ($m = 2, \sigma^2 = 0.05$)	76.3 ± 16.9	2.811 ± 0.751
	KFCM-K (G) ($m = 2.5, \sigma^2 = 8$)	100 ± 0.0	1.998 ± 0.000
	KFCM-K (P) ($m = 2, \theta = 15, d = 8$)	100 ± 0.0	1.998 ± 0.000
3	FCM ($m = 2$)	62.5 ± 2.7	1.035 ± 0.006
	GK ($m = 2$)	87.4 ± 0.9	1.067 ± 0.003
	KFCM-F (G) ($m = 2, \sigma^2 = 0.05$)	88.2 ± 12.6	2.375 ± 0.444
	KFCM-K (G) ($m = 1.4, \sigma^2 = 4$)	100 ± 0.0	1.063 ± 0.002
	KFCM-K (P) ($m = 2, \theta = 2, d = 4$)	100 ± 0.0	2.062 ± 0.127
4	FCM ($m = 1.7$)	100 ± 0.1	0.5866 ± 0.0018
	GK ($m = 1.2$)	87.4 ± 7.3	0.9478 ± 0.1562
	KFCM-F (G) ($m = 1.4, \sigma^2 = 8$)	100 ± 0.0	0.5703 ± 0.0012
	KFCM-K (G) ($m = 1.4, \sigma^2 = 8$)	100 ± 0.0	0.5531 ± 0.0011
	KFCM-K (P) ($m = 1.4, \theta = 50, d = 12$)	100 ± 0.0	0.5700 ± 0.0011

6.1.4. Further synthetic experiments

The rest of the synthetic experiments are described here in Table 7.

The results show that KFCM-K (G) performs very well on many of these problems with fewer clusters especially on the Line, Dense, Zig-Zag, Noisy Zig-Zag, Noisy Ring, and Ring with Outliers data sets. The GK algorithm can perform fairly well as well on the Line and Zig-Zag data sets. It should be noted that the KFCM-K (G) is the only algorithm that can solve the Zig-Zag problem with $c = 3$. Likewise, the KFCM-K (g) does quite well on the Dense data set with $c = 2$ unlike the other algorithms which need more clusters to achieve similar performance.

6.2. UCI machine learning data sets

A number of UCI Machine Learning data sets [1,21] were experimented with including: iris (I), wine (W), glass (G), ionosphere (S), Wisconsin breast cancer (B), Wisconsin diagnostic breast cancer (WDBC) (C), Haberman (H), sonar

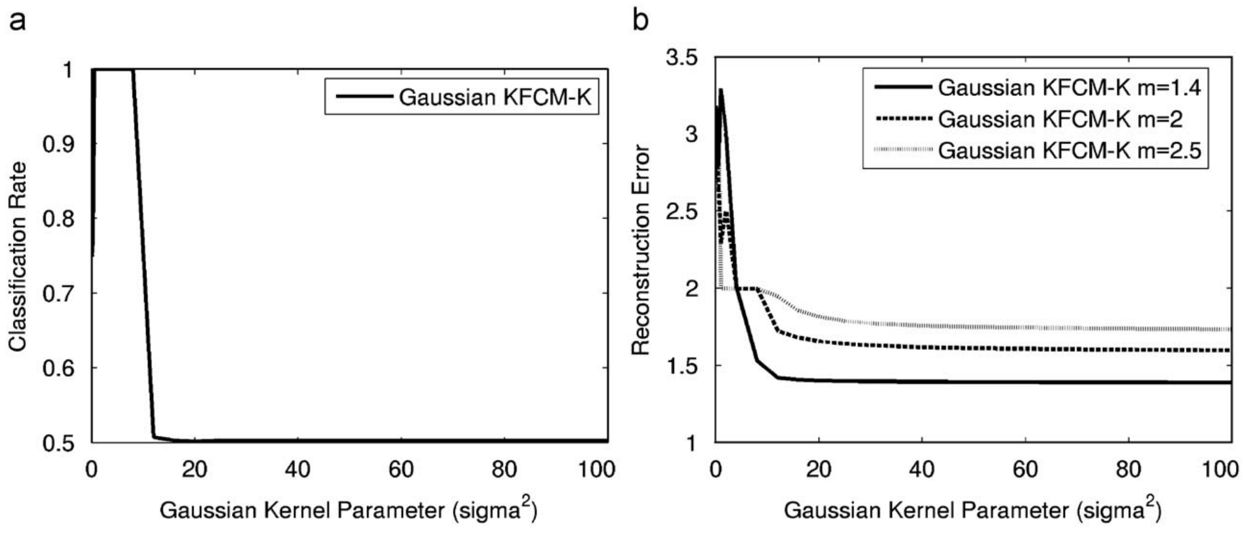

Fig. 5. Classification rate (a) and reconstruction error (b) versus σ^2 for Gaussian KFCM-K on ring ($c = 2$).

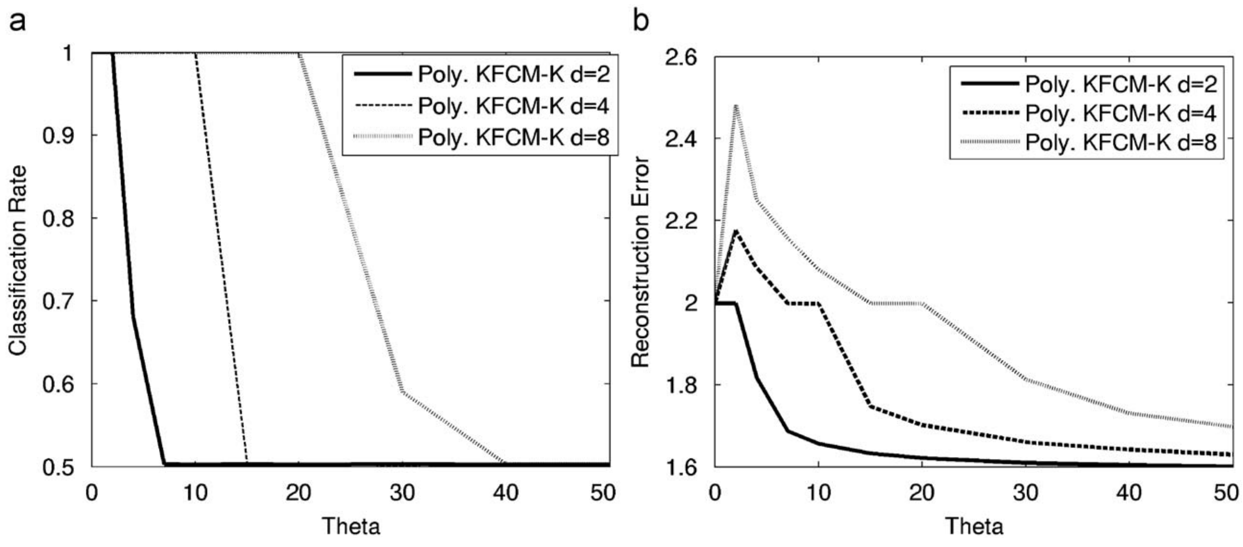

Fig. 6. Classification rate (a) and reconstruction error (b) versus polynomial kernel parameters for polynomial KFCM-K on ring ($c = 2$).

mines versus rocks (O), Pima Indians diabetes (D), ecoli protein localization sites (E), image segmentation (testing data set only) (M), and SPECT heart data (training and testing data sets combined) (P) (Table 8).

The ionosphere is a relatively high-dimensional data set that appears to exhibit statistically significant increase in classification rate for all kernel clustering algorithms. Plots of the classification rate and reconstruction error with respect to kernel parameters and the fuzzification coefficient *m* are given in Figs. 7 and 8.

The average classification rate for iris was just over 5% higher for GK than the kernel algorithms. Interestingly the kernel algorithms do not provide significant improvement in terms of classification rate on the iris, wine, Wisconsin diagnostics breast cancer, sonar, Pima Indian diabetes, and ecoli protein localization sites data sets. The kernel algorithms provide a slight improvement on the glass, Wisconsin breast cancer, and SPECT data sets and a significant improvement on the ionosphere data set. Except with the ionosphere and ecoli protein localization data sets, the FCM algorithm tended to achieve smaller reconstruction errors than the kernel-based algorithms. The results indicate the kernel-based clustering algorithms do not provide significant improvement in classification results and reconstruction errors as

Table 7
Clustering results for ring data set (G—Gaussian and P—polynomial).

	Clustering	Classification rate (%)	Reconstruction error
L	FCM ($m = 1.4, c = 4$)	100.0 ± 0.0	0.1510 ± 0.0000
I	GK ($m = 3, c = 2$)	100.0 ± 0.0	1.802 ± 0.000
	KFCM-F (G) ($m = 1.4, c = 4, \sigma^2 = 1$)	100.0 ± 0.0	0.1814 ± 0.0002
	KFCM-K (G) ($m = 1.4, c = 2, \sigma^2 = 0.5$)	100.0 ± 0.0	2.999 ± 0.944
	KFCM-K (P) ($m = 1.4, c = 4, \theta = 30, p = 2$)	100.0 ± 0.0	0.1457 ± 0.0001
D	FCM ($m = 2.5, c = 6$)	98.8 ± 0.3	0.1643 ± 0.0222
E	GK ($m = 1.4, c = 6$)	98.8 ± 0.3	0.2171 ± 0.0345
	KFCM-F (G) ($m = 2.5, c = 6, \sigma^2 = 50$)	98.8 ± 0.3	0.1688 ± 0.0237
	KFCM-K (G) ($m = 2, c = 2, \sigma^2 = 0.1$)	100.0 ± 0.0	2.926 ± 2.014
	KFCM-K (P) ($m = 2, c = 6, \theta = 2, p = 4$)	99.5 ± 0.0	1.087 ± 0.316
Z	FCM ($m = 2.5, c = 6$)	93.3 ± 2.6	0.1686 ± 0.0680
Z	GK ($m = 1.4, c = 5$)	100.0 ± 0.0	0.7520 ± 0.0000
	KFCM-F (G) ($m = 2, c = 6, \sigma^2 = 8$)	92.2 ± 3.3	0.1993 ± 0.0786
	KFCM-K (G) ($m = 1.4, c = 3, \sigma^2 = 0.25$)	100.0 ± 0.0	2.781 ± 0.768
	KFCM-K (P) ($m = 2.5, c = 5, \theta = 0, p = 2$)	100.0 ± 0.0	1.992 ± 0.000
N	FCM ($m = 2.5, c = 6$)	87.4 ± 3.1	0.1772 ± 0.0515
Z	GK ($m = 2, c = 6$)	92.1 ± 0.2	0.4462 ± 0.0121
	KFCM-F (G) ($m = 2, c = 6, \sigma^2 = 4$)	87.5 ± 2.4	0.2313 ± 0.0755
	KFCM-K (G) ($m = 1.4, c = 3, \sigma^2 = 0.5$)	93.3 ± 0.0	2.157 ± 0.545
	KFCM-K (P) ($m = 2, c = 5, \theta = 0, p = 2$)	92.7 ± 0.0	1.987 ± 0.000
Z	FCM ($m = 2.5, c = 6$)	90.0 ± 0.0	0.4560 ± 0.0000
O	GK ($m = 2, c = 5$)	94.6 ± 0.0	1.1141 ± 0.0002
	KFCM-F (G) ($m = 2.5, c = 6, \sigma^2 = 100$)	90.0 ± 0.0	0.4681 ± 0.0014
	KFCM-K (G) ($m = 2, c = 6, \sigma^2 = 1$)	94.9 ± 0.8	1.747 ± 0.708
	KFCM-K (P) ($m = 2.5, c = 6, \theta = 0, p = 2$)	90.5 ± 3.9	1.848 ± 0.296
N	FCM ($m = 2, c = 4$)	94.5 ± 0.0	0.5169 ± 0.0000
R	GK ($m = 1.7, c = 4$)	91.3 ± 6.7	0.6569 ± 0.1904
	KFCM-F (G) ($m = 2, c = 4, \sigma^2 = 100$)	94.5 ± 0.0	0.5197 ± 0.0000
	KFCM-K (G) ($m = 1.4, c = 2, \sigma^2 = 0.25$)	95.0 ± 0.0	3.565 ± 0.937
	KFCM-K (P) ($m = 2.5, c = 6, \theta = 0, p = 4$)	96.7 ± 0.3	1.936 ± 0.020
R	FCM ($m = 1.4, c = 6$)	95.8 ± 0.0	0.2585 ± 0.0435
O	GK ($m = 1.4, c = 6$)	88.9 ± 5.4	0.5823 ± 0.1590
	KFCM-F (G) ($m = 1.4, c = 6, \sigma^2 = 100$)	95.7 ± 0.2	0.2452 ± 0.0370
	KFCM-K (G) ($m = 1.4, c = 2, \sigma^2 = 0.5$)	95.8 ± 0.0	3.165 ± 0.279
	KFCM-K (P) ($m = 1.4, c = 6, \theta = 0, p = 2$)	98.1 ± 0.0	1.723 ± 0.021

compared to FCM and GK in many of the example data sets. For the data sets where the kernel clustering algorithm performs quite well on, the selection of the kernel parameters was important for obtaining satisfactory classification rates. It should also be noted that for the glass, ionosphere and Wisconsin breast cancer data sets, the fuzzy covariance matrix $\mathbf{C}_i$ in the GK algorithm almost always approached a value of the determinant very close to zero.

6.3. Summary

A two-tailed t-test with unequal variances was performed to compare FCM and GK against the kernel clustering algorithms. The better algorithm and significance reported in terms of obtained p-value is reported in Table 9.

Table 8
Clustering results for selected UCI machine learning data.

	Clustering	Classification rate (%)	Reconstruction error
I	FCM ($m = 2, c = 3$)	84.0 ± 0.0	0.8927 ± 0.0000
	GK ($m = 1.7, c = 3$)	95.3 ± 0.0	0.6890 ± 0.0000
	KFCM-F (G) ($m = 2.5, c = 3, \sigma^2 = 100$)	84.0 ± 0.0	0.9012 ± 0.000
	KFCM-K (G) ($m = 2.5, c = 3, \sigma^2 = 4$)	85.3 ± 0.0	1.524 ± 0.034
	KFCM-K (P) ($m = 2.5, c = 3, \theta = 15, p = 8$)	88.7 ± 0.0	1.942 ± 0.396
W	FCM ($m = 1.4, c = 3$)	96.6 ± 0.0	6.814 ± 0.000
	GK ($m = 1.4, c = 3$)	71.4 ± 5.3	9.121 ± 0.398
	KFCM-F (G) ($m = 1.2, c = 3, \sigma^2 = 2$)	96.1 ± 0.0	8.741 ± 0.000
	KFCM-K (G) ($m = 1.4, c = 3, \sigma^2 = 12$)	97.8 ± 0.0	8.180 ± 0.000
	KFCM-K (P) ($m = 1.4, c = 3, \theta = 20, p = 2$)	97.8 ± 0.0	6.904 ± 0.000
G	FCM ($m = 3, c = 3$)	59.6 ± 0.9	8.756 ± 0.037
	GK ($m = 3, c = 3$)	58.7 ± 2.2	7.590 ± 0.296
	KFCM-F (G) ($m = 2.5, c = 3, \sigma^2 = 50$)	59.7 ± 2.9	8.396 ± 0.077
	KFCM-K (G) ($m = 2, c = 3, \sigma^2 = 16$)	62.6 ± 0.0	8.507 ± 0.009
	KFCM-K (P) ($m = 2.5, c = 3, \theta = 30, p = 2$)	60.6 ± 0.2	8.356 ± 0.000
S	FCM ($m = 1.2, c = 2$)	64.1 ± 0.0	32.10 ± 0.02
	GK ($m = 1.2, c = 2$)	64.1 ± 0.0	32.10 ± 0.02
	KFCM-F (G) ($m = 1.4, c = 2, \sigma^2 = 50$)	70.7 ± 0.0	26.25 ± 0.00
	KFCM-K (G) ($m = 1.4, c = 2, \sigma^2 = 40$)	74.6 ± 0.0	26.85 ± 0.00
	KFCM-K (P) ($m = 1.4, c = 2, \theta = 30, p = 8$)	72.7 ± 0.0	34.62 ± 1.44
B	FCM ($m = 1.2, c = 2$)	95.7 ± 0.0	3.793 ± 0.000
	GK ($m = 2.5, c = 2$)	94.7 ± 0.0	6.603 ± 0.046
	KFCM-F (G) ($m = 1.4, c = 2, \sigma^2 = 2$)	97.1 ± 0.0	5.425 ± 0.003
	KFCM-K (G) ($m = 1.4, c = 2, \sigma^2 = 40$)	97.1 ± 0.0	3.800 ± 0.000
	KFCM-K (P) ($m = 2.5, c = 2, \theta = 50, p = 2$)	95.2 ± 0.0	3.840 ± 0.000
C	FCM ($m = 2.5, c = 2$)	92.1 ± 0.0	22.86 ± 0.00
	GK ($m = 2.5, c = 2$)	92.1 ± 0.0	22.86 ± 0.00
	KFCM-F (G) ($m = 1.4, c = 2, \sigma^2 = 40$)	92.8 ± 0.0	22.34 ± 0.00
	KFCM-K (G) ($m = 1.4, c = 2, \sigma^2 = 100$)	91.6 ± 0.0	24.59 ± 0.00
	KFCM-K (P) ($m = 2.5, c = 2, \theta = 50, p = 2$)	91.2 ± 0.0	25.33 ± 0.00
H	FCM ($m = 1.2, c = 2$)	73.5 ± 0.0	2.170 ± 0.000
	GK ($m = 2.5, c = 2$)	75.2 ± 0.0	2.334 ± 0.000
	KFCM-F (G) ($m = 2.5, c = 2, \sigma^2 = 0.5$)	73.8 ± 0.7	3.287 ± 0.395
	KFCM-K (G) ($m = 1.4, c = 2, \sigma^2 = 100$)	73.5 ± 0.0	2.180 ± 0.000
	KFCM-K (P) ($m = 2.5, c = 2, \theta = 4, p = 4$)	73.9 ± 0.0	4.347 ± 3.496
O	FCM ($m = 1.7, c = 2$)	61.1 ± 0.0	59.71 ± 0.00
	GK ($m = 1.7, c = 2$)	61.1 ± 0.0	59.71 ± 0.00
	KFCM-F (G) ($m = 2, c = 2, \sigma^2 = 16$)	61.4 ± 4.1	73.42 ± 7.99
	KFCM-K (G) ($m = 1.4, c = 2, \sigma^2 = 100$)	56.3 ± 0.0	60.64 ± 0.00
	KFCM-K (P) ($m = 1.4, c = 2, \theta = 50, p = 2$)	59.2 ± 0.1	59.69 ± 0.02
D	FCM ($m = 2, c = 2$)	71.3 ± 0.1	7.932 ± 0.039
	GK ($m = 1.2, c = 2$)	65.2 ± 0.3	6.987 ± 0.092
	KFCM-F (G) ($m = 2, c = 2, \sigma^2 = 50$)	71.0 ± 0.0	7.885 ± 0.039
	KFCM-K (G) ($m = 2, c = 2, \sigma^2 = 100$)	71.4 ± 0.0	7.991 ± 0.000
	KFCM-K (P) ($m = 2.5, c = 2, \theta = 50, p = 8$)	71.5 ± 0.1	7.990 ± 0.000

Table 8
Continued.

	Clustering	Classification rate (%)	Reconstruction error
E	FCM ($m = 2, c = 8$)	81.8 ± 1.5	3.625 ± 0.004
	GK ($m = 1.2, c = 8$)	81.4 ± 2.4	1.565 ± 0.154
	KFCM-F (G) ($m = 2, c = 8, \sigma^2 = 75$)	80.9 ± 1.9	3.668 ± 0.026
	KFCM-K (G) ($m = 2, c = 8, \sigma^2 = 50$)	81.5 ± 0.9	3.809 ± 0.052
	KFCM-K (P) ($m = 1.4, c = 8, \theta = 50, p = 2$)	82.0 ± 1.0	17.55 ± 19.83
M	FCM ($m = 1.7, c = 7$)	69.8 ± 1.3	8.217 ± 0.004
	GK ($m = 1.7, c = 7$)	69.8 ± 1.3	8.217 ± 0.004
	KFCM-F (G) ($m = 2, c = 7, \sigma^2 = 50$)	71.2 ± 0.0	8.772 ± 0.000
	KFCM-K (G) ($m = 1.4, c = 7, \sigma^2 = 40$)	66.8 ± 2.8	7.997 ± 0.118
	KFCM-K (P) ($m = 1.4, c = 7, \theta = 10, p = 2$)	66.5 ± 1.6	7.881 ± 0.083
P	FCM ($m = 1.2, c = 2$)	79.4 ± 0.0	17.87 ± 0.00
	GK ($m = 1.2, c = 2$)	79.4 ± 0.0	20.43 ± 0.96
	KFCM-F (G) ($m = 2.5, c = 2, \sigma^2 = 0.1$)	80.4 ± 2.0	32.14 ± 5.11
	KFCM-K (G) ($m = 2.5, c = 2, \sigma^2 = 0.05$)	84.3 ± 0.0	30.67 ± 2.00
	KFCM-K (P) ($m = 1.4, c = 2, \theta = 50, p = 2$)	79.4 ± 0.0	18.89 ± 0.00

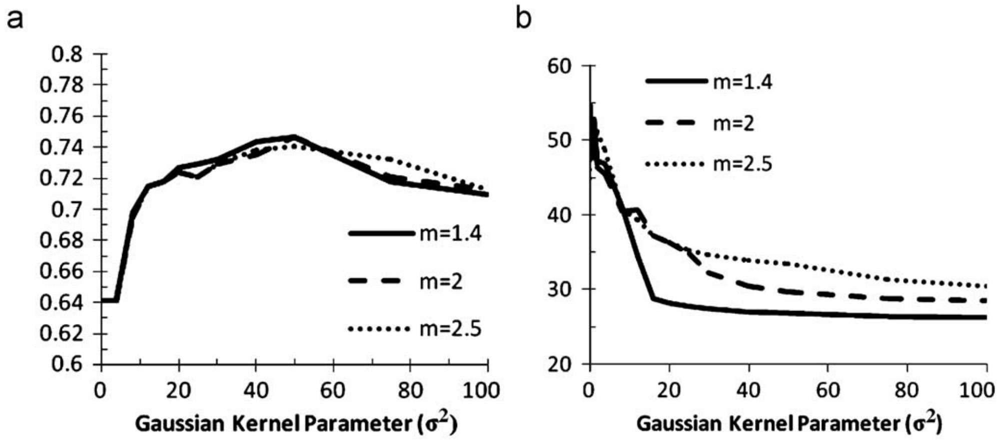

Fig. 7. Classification rate (a) and reconstruction error (b) versus σ^2 for Gaussian KFCM-K ($c = 2$).

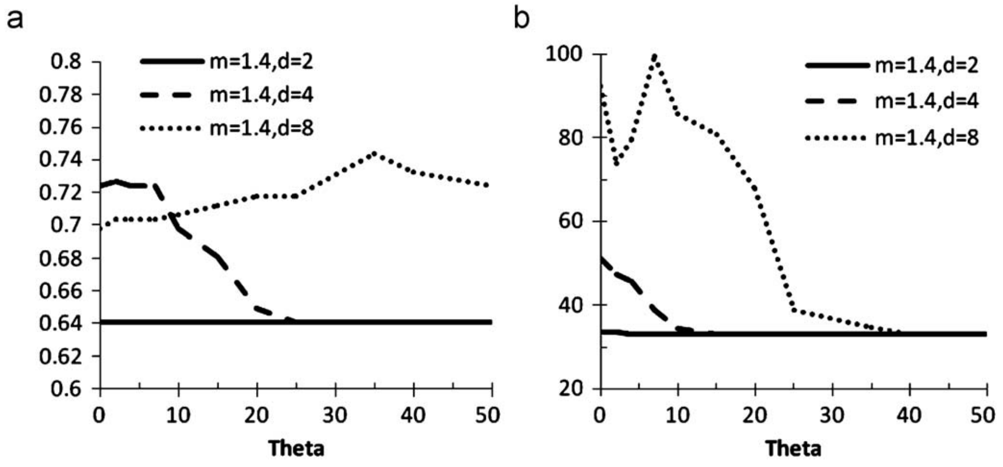

Fig. 8. Classification rate (a) and reconstruction error (b) versus polynomial kernel parameters for polynomial KFCM-K ($c = 2$).

Table 9
Significance of difference in classification rates with 95% confidence by *t*-test.

	FCM versus			GK versus		
	KFCMF	KFCMK (G)	KFCMK (P)	KFCMF	KFCMK (G)	KFCMK (P)
Fuzzy X	KFCMF (2.38e−40)	None (0.519)	KFCMK (8.72e−54)	GK (1.4e−29)	GK (0)	GK (7.2e−31)
Parabolic	KFCMF (5.8e−308)	KFCMK (0)	KFCMK (4.9e−301)	GK (1.1e−256)	KFCMK (3.7e−255)	GK (1.2e−263)
Ring	KFCMF (2.61e−6)	KFCMK (6.21e−38)	KFCMK (6.21e−38)	KFCMF (5.12e−6)	KFCMK (3.75e−28)	KFCMK (3.75e−28)
Dense	None (0.757)	KFCMK (8.77e−15)	KFCMK (9.83e−11)	None (0.774)	KFCMK (2.37e−13)	KFCMK (3.35e−9)
Line	None (1)	None (1)	None (1)	None (1)	None (1)	None (1)
Zig-Zag	None (0.219)	KFCMK (5.2e−10)	KFCMK (5.52e−10)	GK (2.7e−9)	None (1)	None (1)
Noisy Z-Z	None (0.970)	KFCMK (7.27e−8)	KFCMK (4.27e−7)	GK (6.90e−8)	KFCMK (3.48e−17)	KFCMK (4.42e−11)
Z-Z Outlier	None (1)	KFCMK (1.33e−16)	None (0.577)	GK (2.82e−279)	None (0.249)	GK (1.57e−4)
Noisy Ring	None (1)	KFCMK (3.90e−302)	KFCMK (1.09e−18)	KFCMF (4.13e−2)	KFCMK (2.22e−2)	KFCMK (1.76e−3)
Ring Outlier	FCM (4.21e−2)	None (1)	KFCMK (7.50e−268)	KFCMF (2.13e−5)	KFCMK (1.79e−5)	KFCMK (3.62e−7)
Iris	None (1)	KFCMK (0)	KFCMK (2.6e−279)	GK (1.2e−286)	GK (1.3e−285)	GK (0)
Wine	FCM (0)	KFCMK (1.6e−250)	KFCMK (1.6e−250)	KFCMF (1.74e−14)	KFCMK (5.21e−15)	KFCMK (5.21e−15)
Glass	None (0.529)	KFCMK (2.23e−11)	KFCMK (4.11e−5)	KFCMF (0.047)	KFCMK (1.26e−7)	KFCMK (0.00052)
Ionosphere	KFCMF (3.1e−266)	KFCMK (0)	KFCMK (0)	KFCMF (3.1e−266)	KFCMK (0)	KFCMK (0)
Breast Cancer	KFCMF (0)	KFCMK (8.6e−248)	FCM (6.2e−278)	KFCMF (7.51e−38)	GK (7.51e−38)	GK (4.24e−24)
WDBC	KFCMF (1.1e−263)	FCM (0)	FCM (8.1e−260)	KFCMF (1.1e−263)	GK (0)	GK (8.1e−260)
Haberman	None (0.084)	None (1)	KFCMK (0)	GK (3.3e−08)	GK (0)	GK (0)
Sonar	None (0.697)	FCM (1.5e−279)	FCM (1.17e−22)	None (0.697)	GK (1.5e−279)	GK (1.17e−22)
Diabetes	FCM (3.02e−21)	None (0.021)	KFCMK (1.78e−11)	KFCMF (2.61e−26)	KFCMK (6.61e−27)	KFCMK (4.22e−29)
Ecoli	None (0.0811)	None (0.366)	None (0.58)	None (0.452)	None (0.856)	None (0.263)
Segmentation	KFCMF (0.000124)	FCM (0.000231)	FCM (1.68e−8)	KFCMF (0.000124)	GK (0.000231)	GK (1.68e−8)
SPECT	KFCMF (0.0342)	KFCMK (0)	None (1)	KFCMF (0.0342)	KFCMK (0)	None (1)

The kernel clustering algorithms are better statistically than FCM in many of the data sets, however, when GK is compared with the kernel clustering algorithms the results are mixed. Some of *p*-values are very close to zero due to highly consistent clustering results. The ring, ionosphere, glass, wine and Wisconsin breast cancer data sets indicate some statistically significant improvement with the kernel algorithms versus the non-kernel algorithms (FCM and GK). Interestingly the kernel algorithms appear to perform on par with FCM and GK for the ecoli protein localization sites data set. FCM and GK are better than most of the kernel clustering algorithms for the sonar data set. The other results are fairly mixed. The GK algorithm is statistically the best compared with the kernel algorithms for the iris, X, and Haberman data sets (Table 10).

Table 10
Significance of difference in reconstruction errors with 95% confidence by *t*-test.

	FCM versus			GK versus		
	KFCMF	KFCMK (G)	KFCMK (P)	KFCMF	KFCMK (G)	KFCMK (P)
Fuzzy X	FCM (7.67e−27)	FCM (1.66e−41)	FCM (2.92e−7)	KFCMF (5.41e−23)	GK (6.6e−182)	GK (0.000507)
Parabolic	FCM (5e−191)	FCM (6.2e−178)	FCM (7.6e−14)	GK (5.5e−122)	GK (8.7e−135)	GK (7.36e−13)
Ring	FCM (4.69e−08)	FCM (2.29e−44)	FCM (2.28e−44)	GK (5.32e−8)	GK (1.41e−44)	GK (1.4e−44)
Dense	None (0.537)	FCM (6.79e−6)	FCM (6.32e−11)	KFCMF (1.13e−5)	GK (8.72e−6)	GK (1.84e−10)
Line	FCM (4.30e−44)	FCM (3.47e−11)	KFCMK (1.77e−40)	KFCMF (6.88e−77)	GK (1.81e−5)	KFCMK (6.66e−88)
Zig-Zag	None (0.194)	FCM (4.61e−12)	FCM (8.01e−29)	KFCMF (7.48e−18)	GK (3.37e−10)	GK (4.79e−76)
Noisy Z-Z	FCM (1.24e−2)	FCM (1.43e−12)	FCM (4.73e−31)	KFCMF (1.19e−10)	GK (1.74e−11)	GK (1.05e−41)
Z-Z Outlier	FCM (1.37e−19)	FCM (1.28e−7)	FCM (1.25e−14)	KFCMF (2.47e−52)	GK (7.77e−4)	GK (9.52e−10)
Noisy Ring	FCM (2.70e−292)	FCM (9.32e−12)	FCM (6.45e−37)	KFCMF (4.50e−3)	GK (1.43e−11)	GK (1.97e−17)
Ring Outlier	None (0.303)	FCM (5.75e−21)	FCM (7.67e−40)	KFCMF (7.68e−9)	GK (3.03e−26)	GK (6.05e−18)
Iris	FCM (0)	FCM (9.29e−26)	FCM (3.24e−10)	KFCMF (7.8e−98)	GK (9.2e−21)	GK (6.25e−08)
Wine	FCM (1e−147)	FCM (2.1e−188)	FCM (1.4e−295)	KFCMF (0.000412)	KFCMK (2.14e−9)	KFCMK (5.73e−16)
Glass	KFCMF (4.28e−17)	KFCMK (1.82e−18)	KFCMK (3.87e−25)	GK (2.02e−10)	GK (2.25e−11)	GK (3.37e−10)
Ionosphere	KFCMF (1.19e−49)	KFCMK (9.33e−49)	FCM (2.32e−7)	KFCMF (1.26e−49)	KFCMK (9.91e−49)	GK (2.33e−7)
Breast Cancer	FCM (4.58e−54)	FCM (2.3e−157)	FCM (3.9e−238)	KFCMF (1.99e−28)	KFCMK (1.35e−35)	KFCMK (1.78e−35)
WDBC	KFCMF (9.8e−303)	FCM (0)	FCM (2.4e−153)	KFCMF (1.3e−284)	GK (0)	GK (2.4e−153)
Haberman	FCM (1.07e−10)	FCM (1.8e−190)	FCM (0.0118)	GK (1.54e−9)	KFCMK (2e−152)	GK (0.0186)
Sonar	FCM (3.09e−7)	FCM (9.5e−117)	KFCMK (0.000357)	GK (3.09e−7)	GK (3.8e−114)	KFCMK (0.000357)
Diabetes	KFCMF (0.000569)	FCM (1.35e−6)	FCM (2.03e−6)	GK (2.93−24)	GK (1.83e−21)	GK (1.89e−21)
Ecoli	FCM (6.76e−7)	FCM (2.47e−12)	FCM (0.0054)	GK (4.58e−24)	GK (4.91e−27)	GK (0.00109)
Segmentation	FCM (3.73e−42)	KFCMK (9.27e−8)	KFCMK (2.04e−13)	GK (3.81e−42)	KFCMK (9.27e−8)	KFCMK (2.04e−13)
SPECT	FCM (1.31e−10)	FCM (4.15e−17)	FCM (0)	GK (2.76e−9)	GK (4.3e−18)	KFCMK (8.58e−7)

The reconstruction error shows standard FCM as generally the better algorithm with the kernel-based algorithms generally doing a little poorer than standard FCM. The exceptions appear on the glass, ionosphere and segmentation data sets.

Overall there is evidence that for data sets with a certain well-defined and visible structure such as the ring data set the kernel-based fuzzy clustering algorithms and in particular KFCM-K significantly outperforms the other algorithms such as in the ring data set. In general, this is not always the case for any given data set. The performance of the kernel algorithms largely depends on the optimization of the kernel parameters. While the kernel provides extra flexibility in capturing the structure of data in clustering, a learning mechanism is needed for the selection of the kernel parameters which is a significant drawback of the kernel-based fuzzy clustering algorithms. There is increased adjustment and optimization effort required in order to obtain clustering results that in some cases may not be statistically better or are only marginally better than standard FCM or GK. The extra optimization and selection of the kernel are also problem-specific requiring re-optimization with new data.

7. Conclusions

The kernel-based clustering algorithms—especially KFCM-K—can cluster specific non-spherical clusters such as the ring cluster quite well outperforming FCM and GK for the same number of clusters; however, overall the performance of the kernel-based methods is not very impressive due to similar or only slight increases in clustering classification rates compared to FCM. From the perspective of the reconstruction error, KFCM-K often performed similar to that of FCM and KFCM-F. The major disadvantages of kernel-based algorithms are:

- selection of the kernel function and
- optimization of kernel parameters.

In fact in some cases a change in the kernel parameters could reduce the classification rate by as much as one-half and multiply the reconstruction error. Thus kernel-based fuzzy clustering requires tuning in order to achieve optimal performance. In most of the artificial and UCI machine learning test data sets run, the optimal performance of the kernel-based clustering algorithms is not that much of an improvement over FCM and GK. Generally there is no statistically significant difference between the kernel-based algorithms and the FCM and GK algorithms except in a few instances. There is a statistically significant difference albeit small between the classification rates for the KFCM-K kernel-based fuzzy clustering algorithms and both FCM and GK algorithms on the wine and glass data sets. On the ionosphere, ring and Wisconsin breast cancer data sets, the kernel algorithms are statistically better than FCM and GK algorithms. However, on the iris and X data sets, GK is statistically better than the kernel-based algorithms by several percentage points. Overall, the kernel-based algorithms can provide marginal increases in the resulting classification rate.

A future improvement to kernel-based fuzzy clustering could be to develop a partially supervised fuzzy clustering algorithm to optimize the kernel parameters and kernel clustering performance. Another future improvement one might think of would be to construct generalized kernel functions for that have several parameters that provide additional flexibility compared with the current most commonly used kernel functions.

Acknowledgments

Support from the Natural Sciences and Engineering Research Council of Canada (NSERC) and Canada Research Chair (CRC) Program (W. Pedrycz) is gratefully acknowledged.

References

[1] A. Asuncion, D.J. Newman, UCI Machine Learning Repository, University of California, School of Information and Computer Science, Irvine, CA, 2007 [Online] Available: ⟨http://archive.ics.uci.edu/beta/⟩.
[2] R. Babuska, P.J. van der Veen, U. Kaymak, Improved covariance estimation for Gustafson–Kessel clustering, in: Proc. of the IEEE Internat. Conf. on Fuzzy Systems, 2002, pp. 1081–1085.
[3] A. Ben-Hur, D. Horn, H.T. Siegelmann, V. Vapnik, Support vector clustering, Journal of Machine Learning Research 2 (2001) 125–137.
[4] J.C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum, New York, 1981.
[5] S. Borer, W. Gerstner, A new kernel clustering algorithm, in: Proc. of the Ninth Internat. Conf. on Neural Information Processing, Vol. 5, 2002, pp. 2527–2531.
[6] A. Bouchachia, W. Pedrycz, Enhancement of fuzzy clustering by mechanisms of partial supervision, Fuzzy Sets and Systems 157 (13) (2006) 1733–1759.
[7] F. Camastra, A. Verri, A novel kernel method for clustering, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (5) (2005) 801–805.
[8] J.H. Chiang, P.Y. Hao, A new kernel-based fuzzy clustering approach: support vector clustering with cell growing, IEEE Transactions on Fuzzy Systems 11 (4) (2003) 518–527.
[9] I.S. Dhillon, Y. Guan, B. Kulis, Kernel k-means spectral clustering and normalized cuts, in: Proc. of the 10th ACM Internat. Conf. on Knowledge Discovery and Data Mining, 2004, pp. 551–556.

[10] M. Filippone, F. Camastra, F. Masulli, S. Rovetta, A survey of kernel and spectral methods for clustering, Pattern Recognition 41 (1) (2008) 176–190, doi: 10.1016/j.patcog.2007.05.018.
[11] M. Girolami, Mercer kernel-based clustering in feature space, IEEE Transactions on Neural Networks 13 (3) (2002) 780–784.
[12] D. Graves, W. Pedrycz, Fuzzy C-Means, Gustafson–Kessel FCM, and Kernel-based FCM: a comparative study, Advances in Soft Computing 41 (2007) 140–149.
[13] D.E. Gustafson, W.C. Kessel, Fuzzy clustering with a fuzzy covariance matrix, in: IEEE Conf. on Decision Control Internat. 17th Symp. on Adaptive Processes, 1978, pp. 761–766.
[14] R. Herbrich, Learning Kernel Classifiers, MIT Press, Cambridge, MA, 2002.
[15] D. Horn, Clustering via Hilbert space, Physica A 302 (1) (2001) 70–79.
[16] D.W. Kim, K.Y. Lee, D. Lee, K.H. Lee, Evaluation of the performance of clustering algorithms kernel-induced feature space, Pattern Recognition 38 (4) (2005) 607–611.
[17] F. Klawonn, R. Kruse, Constructing a fuzzy controller from data, Fuzzy Sets and Systems 85 (2) (1997) 177–193.
[18] R. Krishnapuram, J. Kim, A note on the Gustafson–Kessel and adaptive fuzzy clustering algorithms, IEEE Transactions on Fuzzy Systems 7 (4) (1999) 453–461.
[19] T. Li, S. Ma, M. Ogihara, Entropy-based criterion in categorical clustering, in: Proc. of the 21st Internat. Conf. on Machine Learning, Vol. 69, 2004, pp. 68–75.
[20] K.R. Muller, S. Mika, G. Ratsch, K. Tsuda, B. Scholkopf, An introduction to kernel-based learning algorithms, IEEE Transactions on Neural Networks 12 (2) (2001) 181–201.
[21] D.J. Newman, S. Hettich, C.L. Blake, C.J. Merz, UCI Repository of machine learning databases, University of California, School of Information and Computer Science, Irvine, CA, 1998 [Online] Available: ⟨http://www.ics.uci.edu/∼ mlearn/MLRepository.html⟩.
[22] W. Pedrycz, Knowledge-based Clustering, Wiley, Hoboken, NJ, 2005.
[23] J. Qin, D.P. Lewis, W.S. Noble, Kernel hierarchical gene clustering from microarray expression data, Bioinformatics 19 (16) (2003) 2097–2104.
[24] B. Scholkopf, A. Smola, K.R. Muller, Nonlinear component analysis as a kernel eigenvalue problem, Neural Computation 10 (1998) 1299–1319.
[25] H. Shen, J. Yang, S. Wang, X. Liu, Attribute weighted Mercer kernel based fuzzy clustering algorithm for general non-spherical datasets, Soft Computing 10 (11) (2006) 1061–1073.
[26] Z.D. Wu, W.X. Xie, J.P. Yu, Fuzzy c-means clustering algorithm based on kernel method, in: Proc. of the Internat. Conf. on Computational Intelligence and Multimedia Applications, 2003, pp. 49–54.
[27] Z.R. Yang, Probabilistic mercer kernel clusters, in: Proc. of the IEEE Internat. Conf. on Neural Networks and Brain, Vol. 3, 2005, pp. 1885–1890.
[28] L. Zeyu, T. Shiwei, X. Jing, J. Jun, Modified FCM clustering based on kernel mapping, in: Proc. of the Internat. Society for Optical Engineering, Vol. 4554, 2001, pp. 241–245.
[29] D.Q. Zhang, S.C. Chen, Clustering incomplete data using Kernel-based Fuzzy C-Means algorithm, Neural Processing Letters 18 (3) (2003) 155–162.
[30] D. Zhang, S. Chen, Fuzzy clustering using kernel method, in: Proc. of the Internat. Conf. on Control and Automation, 2002, pp. 123–127.
[31] L. Zhang, C. Zhou, M. Ma, X. Liu, C. Li, C. Sun, M. Liu, Fuzzy kernel clustering based on particle swarm optimization, in: Proc. of the IEEE Internat. Conf. on Granular Computing, 2006, pp. 428–430.
[32] S. Zhou, J. Gan, Mercer kernel fuzzy c-means algorithm and prototypes of clusters, in: Proc. of Conf. on Internat. Data Engineering and Automated Learning, Vol. 3177, 2004, pp. 613–618.