Gaussian Kernel-Based Fuzzy Clustering with Automatic Bandwidth Computation

Francisco de A. T. de Carvalho[1(✉)], Lucas V. C. Santana[1], and Marcelo R. P. Ferreira[2]

[1] Centro de Informatica, Universidade Federal de Pernambuco, Av. Jornalista Anibal Fernandes s/n - Cidade Universitaria, Recife-PE 50740-560, Brazil
fatc@cin.ufpe.br

[2] Departamento de Estatistica, Centro de Ciências Exatas e da Natureza, Universidade Federal da Paraiba, João Pessoa-PB 58051-900, Brazil

Abstract. The conventional Gaussian kernel-based fuzzy c-means clustering algorithm has widely demonstrated its superiority to the conventional fuzzy c-means when the data sets are arbitrarily shaped, and not linearly separable. However, its performance is very dependent on the estimation of the bandwidth parameter of the Gaussian kernel function. Usually this parameter is estimated once and for all. This paper presents a Gaussian fuzzy c-means with kernelization of the metric which depends on a vector of bandwidth parameters, one for each variable, that are computed automatically. Experiments with data sets of the UCI machine learning repository corroborate the usefulness of the proposed algorithm.

1 Introduction

Clustering means the task of organizing a set of items into clusters such that items within a given cluster have a high degree of similarity, while items belonging to different clusters have a high degree of dissimilarity. Clustering has been successfully used in different fields, including bioinformatics, image processing, and information retrieval [14,21].

Hierarchy and Partition are the most popular cluster structures provided by clustering methods. Hierarchical methods yield a complete hierarchy, i.e., a nested sequence of partitions of the input data, whereas partitioning methods aims to obtain a single partition of the data into a fixed number of clusters, usually based on an iterative algorithm that optimizes an objective function.

Partitioning methods can be divided into crisp and fuzzy. Crisp clustering provides a crisp partition in which each object of the dataset belongs to one and only one cluster. Fuzzy clustering [1] generates a fuzzy partition that provides a membership degree for each object in a given cluster. This allows distinguish objects that belong to more than one cluster at the same time [15].

Fuzzy c-means partitioning algorithms often use the Euclidean distance to compute the dissimilarity between the objects and the cluster representatives.

V. Kůrková et al. (Eds.): ICANN 2018, LNCS 11139, pp. 685–694, 2018.
https://doi.org/10.1007/978-3-030-01418-6_67

However, when the data structure is complex (i.e., clusters with non-hyper-spherical shapes and/or linearly non-separable patterns), the conventional fuzzy c-means will not be able to provide effective results. Kernel-based clustering algorithms have been proposed to tackle these limitations [3,6,8,9].

There are two major variations of kernel-based clustering: one is the kernelization of the metric, where the cluster centroids are obtained in the original space and the distances between objects and cluster centroids are computed by means of kernels, while the other is the clustering in feature space, in which the cluster representatives are not in the original space and can only be obtained indirectly in the feature space [3,9].

In kernel-based clustering algorithms it is possible to compute Euclidean distances by using kernel functions and the so-called distance kernel trick [9]. This trick uses a kernel function to calculate the dot products of vectors implicitly in the higher dimensional space using the original space.

The most popular kernel function in applications is the Gaussian kernel. In general, this kernel function provides effective results and requires the tuning of a single parameter, that is, the bandwidth parameter [4]. This parameter is tuned once and for all, and it is the same for all variables. Thus, implicitly the conventional Gaussian kernel fuzzy c-means assumes that the variables are equally rescaled and, therefore, they have the same importance to the clustering task. However, it is well known that some variables have different degrees of relevance while others are irrelevant to the clustering task [7,11,17,20].

Recently, Ref. [5] proposed a Gaussian kernel c-means crisp clustering algorithm with kernelization of the metric, where each variable has its own hyper-parameter that is iteratively computed during the running of the algorithm.

The main contribution of this paper is to provide a Gaussian kernel c-means fuzzy clustering algorithms, with both kernelization of the metric and automated computation of the bandwidth parameters using an adaptive Gaussian kernel. In these kernel-based fuzzy clustering algorithm, the bandwidth parameters change at each algorithm iteration and differ from variable to variable. Thus, these algorithms are able to rescale the variables differently and thus select the relevant ones for the clustering task.

The paper is organized as follows. Section 2 first recalls the conventional kernel c-means fuzzy clustering algorithm with kernelization of the metric. Then presents the Gaussian c-Means fuzzy clustering algorithm with kernelization of the metric and with automatic computation of bandwidth parameters. In Sect. 3, experiments with data sets of the UCI machine learning repository corroborate the usefulness of the proposed algorithm. Section 4 provides the final remarks of the paper.

2 Kernel Fuzzy c-Means with Kernelization of the Metric

This section briefly recalls the basic concepts about kernel functions and the conventional kernel c-means algorithm with kernelization of the metric. Let $E =$

$\{e_1, \ldots, e_n\}$ be a set of n objects described by p real-valued variables. Let $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ be a non-empty set where for $k = 1, \ldots, n$, the k^{th} object e_k is represented by a vector $\mathbf{x}_k = (x_{k1}, \ldots, x_{kp}) \in \mathbb{R}^p$. A function $\mathcal{K} : \mathcal{D} \times \mathcal{D} \to \mathbb{R}$ is called a positive definite Kernel (or Mercer kernel) if, and only if $\mathcal{K}$ is symmetric (i.e., $\mathcal{K}(\mathbf{x}_k, \mathbf{x}_l) = \mathcal{K}(\mathbf{x}_l, \mathbf{x}_k)$ and if the following inequality holds [18]:

$$\sum_{l=1}^{n}\sum_{k=1}^{n} c_l c_k \mathcal{K}(\mathbf{x}_l, \mathbf{x}_k) \geq 0, \forall n \geq 2 \tag{1}$$

where $c_l, c_k \in \mathbb{R} (1 \leq l, k \leq n)$.

Let $\Phi : \mathcal{D} \to \mathcal{F}$ be a nonlinear mapping from the input space $\mathcal{D}$ to a high dimensional feature space $\mathcal{F}$. By applying the mapping Φ, the inner product $\mathbf{x}_l^T \mathbf{x}_k$ in the input space is mapped to $\Phi(\mathbf{x}_l)^T \Phi(\mathbf{x}_k)$ in the feature space. The basic notion in the kernel approaches is that the non-linear mapping Φ does not need to be explicitly specified because each Mercer kernel can be expressed as $\mathcal{K}(\mathbf{x}_l, \mathbf{x}_k) = \Phi(\mathbf{x}_l)^T \Phi(\mathbf{x}_k)$ [18].

One the most relevant implications is that it is possible to compute Euclidean distances in $\mathcal{F}$ without knowing explicitly Φ, by using the so-called distance kernel trick [9]:

$$\begin{aligned} ||\Phi(\mathbf{x}_l) - \Phi(\mathbf{x}_k)|| &= (\Phi(\mathbf{x}_l) - \Phi(\mathbf{x}_k))^T (\Phi(\mathbf{x}_l) - \Phi(\mathbf{x}_k)) \\ &= \Phi(\mathbf{x}_l)^T \Phi(\mathbf{x}_l) - 2\Phi(\mathbf{x}_l)^T \Phi(\mathbf{x}_k) + \Phi(\mathbf{x}_k)^T \Phi(\mathbf{x}_k) \\ &= \mathcal{K}(\mathbf{x}_l, \mathbf{x}_l) - 2\mathcal{K}(\mathbf{x}_l, \mathbf{x}_k) + \mathcal{K}(\mathbf{x}_k, \mathbf{x}_k). \end{aligned}$$

2.1 Kernel Fuzzy c-Means with Kernelization of the Metric

The kernel fuzzy c-means with kernelization of the metric (hereafter named KFCM-K) provides a fuzzy partition of E into c clusters, represented by a matrix of membership degrees $\mathbf{U} = \left(u_{ki}\right) (1 \leq k \leq n; 1 \leq i \leq c)$, and a matrix of cluster representatives (called hereafter matrix of prototypes) $\mathbf{G} = (\mathbf{g}_1, \ldots, \mathbf{g}_c)$ of the fuzzy clusters in the fuzzy partition $\mathbf{U}$. The prototype of cluster $i\,(i = 1, \ldots, c)$ is represented by the vector $\mathbf{g}_i = (g_{i1}, \ldots, g_{ip}) \in \mathbb{R}^p$.

From an initial solution, the matrix of prototypes $\mathbf{G}$ and the fuzzy partition $\mathbf{U}$ are obtained iteratively in two steps (representation and assignment) by the minimization of a suitable objective function, here-below denoted as J_{KFCM-K}, that gives the total heterogeneity of the fuzzy partition computed as the sum of the heterogeneity in each fuzzy cluster:

$$J_{KFCM-K}(\mathbf{G}, \mathbf{U}) = \sum_{i=1}^{c}\sum_{k=1}^{n} (u_{ki})^m \, ||\Phi(\mathbf{x}_k) - \Phi(\mathbf{g}_i)||^2 \tag{2}$$

where $1 < m < \infty$ is the fuzziness parameter. Using the so-called distance kernel trick [9], we have $||\Phi(\mathbf{x}_k) - \Phi(\mathbf{g}_i)||^2 = \mathcal{K}(\mathbf{x}_k, \mathbf{x}_k) - 2\mathcal{K}(\mathbf{x}_k, \mathbf{g}_i) + \mathcal{K}(\mathbf{g}_i, \mathbf{g}_i)$.

Hereafter we consider the Gaussian kernel, the most commonly used in the literature: $\mathcal{K}(\mathbf{x}_l, \mathbf{x}_k) = \exp\left\{-\frac{||\mathbf{x}_l - \mathbf{x}_k||^2}{2\sigma^2}\right\} = \exp\left\{-\frac{1}{2}\sum_{j=1}^{p} \frac{1}{\sigma^2}(x_{lj} - x_{kj})^2\right\}$, where σ^2 is the bandwidth parameter of the Gaussian kernel.

Then, $\mathcal{K}(\mathbf{x}_k, \mathbf{x}_k) = 1, \forall k$, $\mathcal{K}(\mathbf{g}_i, \mathbf{g}_i) = 1, \forall i$, and $||\Phi(\mathbf{x}_k) - \Phi(\mathbf{g}_i)||^2 = 2 - 2\,\mathcal{K}(\mathbf{x}_k, \mathbf{g}_i)$ and thus, the objective function J_{KFMC-K} becomes:

$$J_{KFCM-K}(\mathbf{G}, \mathbf{U}) = 2\sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ki})^m\,(1 - \mathcal{K}(\mathbf{x}_k, \mathbf{g}_i)) \tag{3}$$

During the representation step, the fuzzy partition $\mathbf{U}$ is kept fixed. The objective function J_{KFMC-K} is optimized with respect to the prototypes. Thus, from $\frac{\partial J_{KFMC-K}}{\partial \mathbf{g}_i} = 0$ and after some algebra, the fuzzy cluster prototypes are obtained as follows:

$$\mathbf{g}_i = \frac{\sum_{k=1}^{n}(u_{ki})^m\,\mathcal{K}(\mathbf{x}_k, \mathbf{g}_i)\mathbf{x}_k}{\sum_{k=1}^{n}(u_{ki})^m\mathcal{K}(\mathbf{x}_k, \mathbf{g}_i)}\,(1 \leq i \leq c). \tag{4}$$

In the assignment step, the cluster prototypes are kept fixed. The components $u_{ki}\,(1 \leq k \leq n; 1 \leq i \leq c)$ of the matrix of membership degrees $\mathbf{U}$, that minimizes the clustering criterion given in Eq. (3), are computed as follows:

$$u_{ki} = \left[\sum_{h=1}^{c}\left(\frac{(1 - \mathcal{K}(\mathbf{x}_k, \mathbf{g}_i))}{(1 - \mathcal{K}(\mathbf{x}_k, \mathbf{g}_h))}\right)^{\frac{1}{m-1}}\right]^{-1}. \tag{5}$$

2.2 KFCM-K with Automatic Computation of Bandwidth Parameters

The kernel fuzzy c-means with kernelization of the metric and automatic computation of bandwidth parameters (hereafter named KFCM-K-H) provides a partition of E into c clusters, represented by a matrix of membership degrees $\mathbf{U} = \left(u_{ki}\right)(1 \leq k \leq n; 1 \leq i \leq c)$, a vector of bandwidth parameters (one for each variable) $\mathbf{s} = (s_1^2, \ldots, s_p^2)$ and a matrix of prototypes $\mathbf{G} = (\mathbf{g}_1, \ldots, \mathbf{g}_c)$ of the fuzzy clusters in the fuzzy partition $\mathbf{U}$.

From an initial solution, the matrix of prototypes $\mathbf{G}$, the vector of bandwidth parameters $\mathbf{s}$ and the fuzzy partition $\mathbf{U}$ are obtained interactively in three steps (representation, computation of the bandwidth parameters and assignment) by the minimization of a suitable objective function, here-below denoted as $J_{KFCM-K-H}$, that gives the total heterogeneity of the fuzzy partition computed as the sum of the heterogeneity in each fuzzy cluster:

$$J_{KFCM-K-H}(\mathbf{G}, \mathbf{s}, \mathbf{U}) = \sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ki})^m||\Phi(\mathbf{x}_k) - \Phi(\mathbf{g}_i)||^2 \tag{6}$$

where

$$||\Phi(\mathbf{x}_k) - \Phi(\mathbf{g}_i)||^2 = \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{x}_k) - 2\mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i) + \mathcal{K}^{(\mathbf{s})}(\mathbf{g}_i, \mathbf{g}_i) \tag{7}$$

with

$$\mathcal{K}^{(\mathbf{s})}(\mathbf{x}_l, \mathbf{x}_k) = \exp\left\{-\frac{1}{2}\sum_{j=1}^{p}\frac{1}{s_j^2}(x_{lj} - x_{kj})^2\right\}$$

Because $\mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{x}_k) = 1, \forall k$, $\mathcal{K}^{(\mathbf{s})}(\mathbf{g}_i, \mathbf{g}_i) = 1, \forall i$, and $||\Phi(\mathbf{x}_k) - \Phi(\mathbf{g}_i)||^2 = 2 - 2\,\mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)$, the objective function $J_{KFMC-K-H}$ becomes:

$$J_{KFCM-K-H}(\mathbf{G}, \mathbf{s}, \mathbf{U}) = 2\sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ki})^m(1 - \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)) \tag{8}$$

During the representation step, the vector of bandwidth parameters $\mathbf{s}$ and the fuzzy partition $\mathbf{U}$ are kept fixed. The objective function $J_{KFMC-K-H}$ is optimized with respect to the prototypes. Thus, from $\frac{\partial J_{KFMC-K-H}}{\partial \mathbf{g}_i} = 0$ and after some algebra, the cluster prototypes are obtained as follows:

$$\mathbf{g}_i = \frac{\sum_{k=1}^{n}(u_{ki})^m\mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)\mathbf{x}_k}{\sum_{k=1}^{n}(u_{ki})^m\mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)}\,(1 \leq i \leq c). \tag{9}$$

In the computation of the bandwidth parameters step, the matrix of prototypes $\mathbf{G}$ and the fuzzy partition $\mathbf{U}$ are kept fixed. First, we use the method of Lagrange multipliers with the restriction that $\prod_{j=1}^{p}\left(\frac{1}{s_j^2}\right) = \gamma$, where γ is a suitable parameter, and obtain

$$\mathcal{L}^1_{KFCM-K-H}(\mathbf{G}, \mathbf{s}, \mathbf{U}) = 2\sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ki})^m(1 - \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)) - \omega\left(\prod_{j=1}^{p}\frac{1}{s_j^2} - \gamma\right). \tag{10}$$

Then, we compute the partial derivatives of $\mathcal{L}^1_{KFCM-K-H}$ w.r.t $\frac{1}{s_j^2}$ and ω, and by setting the partial derivatives to zero, and after some algebra we obtain

$$\frac{1}{s_j^2} = \frac{\gamma^{\frac{1}{p}}\left\{\prod_{h=1}^{p}\left[\sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ki})^m\mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)(x_{kh} - g_{ih})^2\right]\right\}}{\sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ki})^m\mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)(x_{kj} - g_{ij})^2}\,(1 \leq j \leq p). \tag{11}$$

In the assignment step, the matrix of fuzzy cluster prototypes $\mathbf{G}$ and the vector of bandwidth parameters $\mathbf{s}$ are kept fixed. First, we use the method of Lagrange multipliers with the restriction that $\sum_{i=1}^{c} u_{ki} = 1$, and obtain

$$\mathcal{L}^2_{KFCM-K-H}(\mathbf{G}, \mathbf{s}, \mathbf{U}) = 2\sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ki})^m(1 - \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)) - \sum_{k=1}^{n}\omega_k\left(\sum_{i=1}^{c}u_{ki} - 1\right). \tag{12}$$

Then, we compute the partial derivatives of $\mathcal{L}^2_{KFCM-K-H}$ w.r.t u_{ki} and ω_k, and by setting the partial derivatives to zero, and after some algebra we obtain

$$u_{ki} = \left[\sum_{h=1}^{c}\left(\frac{(1 - \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i))}{(1 - \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_h))}\right)^{\frac{1}{m-1}}\right]^{-1}\,(1 \leq k \leq n; 1 \leq i \leq c). \tag{13}$$

Algorithm 1. KCM-K and KCM-K-H algorithms

Iput
  $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ (the data set); c (the number of clusters); $\gamma > 0$ (a suitable parameter); T (maximum number of iterations); ϵ (threshold parameter);
Output
  KCM-K-GH and KCM-K-LH: the matrix of prototypes $\mathbf{G} = (\mathbf{g}_1, \ldots, \mathbf{g}_c)$;
  KCM-K-H: the vector of bandwidth parameters $\mathbf{s} = (s_1^2, \ldots, s_p^2)$;
  KCM-K-GH and KCM-K-LH: the matrix of membership degrees $\mathbf{U} = \left(u_{ki}\right)$ $(1 \leq k \leq n; 1 \leq i \leq c)$.
Initialization
  $t = 0$;
  KCM-K and KCM-K-H: randomly select c distinct prototypes $\mathbf{g}_i^{(t)} \in \mathcal{D}$ $(1 \leq i \leq c)$;
  KCM-K-H: set $\frac{1}{(s_j^{(t)})^2} = (\gamma)^{\frac{1}{p}}$ $(1 \leq j \leq p)$;
  KCM-K: compute the components $u_{ki}^{(t)} (1 \leq k \leq n; 1 \leq i \leq c)$ of the the matrix of membership degrees $\mathbf{U}^{(t)}$ according to Eq. (5);
  KCM-K-H: compute the components $u_{ki}^{(t)} (1 \leq k \leq n; 1 \leq i \leq c)$ of the the matrix of membership degrees $\mathbf{U}^{(t)}$ according to Eq. (13).
  KCM-K: compute $J_{KFCM-K}(\mathbf{G}^{(t)}, \mathbf{U}^{(t)})$ according to Eq. (3);
  KCM-K-H: compute $J_{KFCM-K-H}(\mathbf{G}^{(t)}, \mathbf{s}^{(t)}, \mathbf{U}^{(t)})$ according to Eq. (8).
repeat
  $t = t + 1$;
  Step 1: representation.
    KCM-K: compute the cluster representatives $\mathbf{g}_1^{(t)}, \ldots, \mathbf{g}_c^{(t)}$ using Eq. (4);
    KCM-K-H: compute the cluster representatives $\mathbf{g}_1^{(t)}, \ldots, \mathbf{g}_c^{(t)}$ using Eq. (9).
  Step 2: computation of the vector of bandwidth parameters
    KCM-K: skip this step;
    KMC-K-H: compute the vector of bandwidth parameters $\mathbf{s}^{(t)}$ using Eq. (11);
  Step 3: assignment
    KCM-K: compute the components $u_{ki}^{(t)} (1 \leq k \leq n; 1 \leq i \leq c)$ of the the matrix of membership degrees $\mathbf{U}^{(t)}$ according to Eq. (5).
    KCM-K-H: compute the components $u_{ki}^{(t)} (1 \leq k \leq n; 1 \leq i \leq c)$ of the the matrix of membership degrees $\mathbf{U}^{(t)}$ according to Eq. (13).
  KCM-K: compute $J_{KFCM-K}(\mathbf{G}^{(t)}, \mathbf{U}^{(t)})$ according to Eq. (3).
  KCM-K-H: compute $J_{KFCM-K-H}(\mathbf{G}^{(t)}, \mathbf{s}^{(t)}, \mathbf{U}^{(t)})$ according to Eq. (8).
until
  KCM-K: $|J_{KFCM-K}(\mathbf{G}^{(t)}, \mathbf{U}^{(t)}) - J_{KFCM-K}(\mathbf{G}^{(t-1)}, \mathbf{U}^{(t-1)})| < \epsilon$ or $t > T$;
  KCM-K-H: $|J_{KFCM-K-H}(\mathbf{G}^{(t)}, \mathbf{s}^{(t)}, \mathbf{U}^{(t)}) - J_{KFCM-K-H}(\mathbf{G}^{(t-1)}, \mathbf{s}^{(t-1)}, \mathbf{U}^{(t-1)})| < \epsilon$ or $t > T$.

2.3 The Algorithms

The two steps of KFCM-K and the three steps of KFCM-K-H are repeated until the convergence. The Algorithm 1 summarizes these steps.

3 Empirical Results

This section discusses the performance and the usefulness of the proposed algorithm in comparison with the standard KFCM-K and FCM [1] algorithms.

Twelve datasets from the UCI Machine learning Repository [2], namely, Breast tissue, Ecoli, Image segmentation, Iris plants, Leaf, Libras Movement, Multiple features, Seeds, Thyroid gland, Urban land cover, Breast cancer wisconsin (diagnostic), and Wine, with different number of objects, variables and a priori classes, were considered in this study. Table 1 (in which n is the number of objects, p is the number of real-valued variables and K is the number of a priori classes) summarizes these data sets.

Table 1. Summary of the data sets

Data sets	n	p	K	Data sets	n	p	K
Breast tissue	106	9	6	Multiple features	2000	649	10
Ecoli	336	7	8	Seeds	210	7	3
Image segmentation	2100	19	7	Thyroid gland	215	5	3
Iris	150	4	3	Urban land cover	675	148	9
Leaf	310	14	36	Brest cancer winsconsin	569	30	2
Libras Movement	360	90	15	Wine	178	13	3

FCM, KFCM-K and KFCM-K-H were run on these data sets 100 times, with c (the number of clusters) equal to K (the number of a priori classes). The parameter *gamma* of the KFCM-K-H algorithm was set as $\gamma = (\sigma^2)^p$, where σ is the optimal width hyper-parameter used in the conventional KFCM-K algorithm that is estimated as the average of the 0.1 and 0.9 quantiles of $||\mathbf{x}_l - \mathbf{x}_k||^2$, $l \neq k$ [4]. The fuzziness parameter was set as $m = 1.6$ and $m = 2.0$.

To compare the quality of the fuzzy partitions provided by these algorithms, the Rand index for a fuzzy partition (Rand-F) [10], and the Hullemeyer index (HUL) [13] were considered. Rand-F and HUL indexes allow to compare the dataset a priori partition with the fuzzy partitions provided by the algorithms. They range between 0 and 1, where a value equal to one corresponds to total agreement between the partitions.

Table 2 shows the best results (according to the respective objective functions) of the FCM, KCM-K and KCM-K-H algorithms on the data sets of Table 1, according to the Rand-F and HUL indexes and for the fuzziness parameter set as $m = 1.6$ and $m = 2.0$.

It can be observed that whatever the considered indexes (Rand-F and HUL), FCM (for the great majority of the datasets), KFCM-K (for the great majority of the datasets) and KFCM-K-H (for the totality of the datasets) algorithms performed better with the $m = 1.6$. Moreover, whatever the considered indexes and fuzziness parameters, the KFCM-K-H algorithm performed better than the KFCM-K algorithm on the majority of the data sets of the Table 1. Besides, whatever the considered indexes and fuzziness parameters, the standard FCM algorithm outperformed both KFCM-K and KFCM-K-H algorithms on the majority of the datasets of the Table 1. This is not unexpected because

Table 2. Performance of the algorithms: fuzzy partition

	Rand-F						HUL					
	FCM		KFCM-K		KFCM-K-H		FCM		KFCM-K		KFCM-K-H	
Data sets	$m = 1.6$	$m = 2.0$	$m = 1.6$	$m = 2.0$	$m = 1.6$	$m = 2.0$	$m = 1.6$	$m = 2.0$	$m = 1.6$	$m = 2.0$	$m = 1.6$	$m = 2.0$
Breast tissue	0.6236	0.6296	0.7153	0.7280	0.7268	0.7095	0.6205	0.6143	0.6937	0.6183	0.7163	0.6505
Ecoli	0.7624	0.7239	0.7246	0.6969	0.7653	0.7211	0.7445	0.6470	0.6364	0.5263	0.7524	0.6448
Image segmentation	0.7796	0.7787	0.7571	0.7588	0.8671	0.8113	0.7171	0.5974	0.4155	0.3111	0.8404	0.7049
Iris	0.8620	0.8131	0.7723	0.6881	0.7999	0.7037	0.8641	0.8187	0.7661	0.6632	0.8004	0.6880
Leaf	0.9528	0.9450	0.9499	0.9452	0.9551	0.9462	0.9035	0.7416	0.8223	0.5798	0.8772	0.6414
Libras Movement	0.8887	0.8793	0.8813	0.8789	0.8815	0.8788	0.6543	0.2603	0.3998	0.2656	0.4039	0.2599
Multiple features	0.8689	0.8500	0.8284	0.8227	0.8291	0.8226	0.8336	0.7141	0.3915	0.2707	0.3967	0.2590
Seeds	0.8287	0.7608	0.6047	0.5798	0.7515	0.6675	0.8268	0.7543	0.5133	0.4547	0.7412	0.6339
Thyroid gland	0.7195	0.6070	0.4941	0.4913	0.5731	0.4985	0.7213	0.6175	0.4634	0.4647	0.5856	0.5257
Urban land cover	0.7320	0.7477	0.7836	0.7830	0.8039	0.7852	0.6371	0.5184	0.2960	0.2085	0.5442	0.2755
Brest cancer winsconsin	0.7317	0.7151	0.5000	0.5000	0.7531	0.6452	0.7346	0.7234	0.5046	0.5095	0.7731	0.6978
Wine	0.7015	0.6758	0.8198	0.6664	0.7683	0.6471	0.6987	0.6651	0.6934	0.6533	0.7792	0.6279

as pointed out by Ref. [19], kernelization may impose undesirable structures on the data, and hence, the clusters obtained in the kernel space may not exhibit the structure of the original data.

From the fuzzy partition $\mathbf{U}$ it is obtained a crisp partition $\mathcal{Q} = (Q_1, \ldots, Q_c)$, where the cluster $Q_i (i = 1, \ldots, c)$ is defined as: $Q_i = \{e_k \in E : u_{ik} = \max_{m=1}^{c} u_{mk}\}$. To compare the quality of the crisp partitions provided by KFCM-K and KFCM-K-H algorithms, the adjusted Rand index (ARI) [12], and the mutual normalized information (MNI) [16] were considered. ARI and MNI indexes allow to compare the dataset a priori partition with the crisp partitions obtained from the fuzzy partitions provided by the algorithms. ARI index takes its values on the interval $[-1, 1]$, in which the value 1 indicates perfect agreement between partitions. The NMI takes its values on the interval $[0, 1]$, in which the value 1 also indicates perfect agreement between partitions.

Table 3 shows the best results (according to the respective objective functions) of the KCM-K and KCM-K-H algorithms on the data sets of Table 1, according to the ARI and NMI indexes and for the fuzziness parameter set as $m = 1.6$ and $m = 2.0$.

Table 3. Performance of the algorithms: crisp partition

	ARI						NMI					
	FCM		KFCM-K		KFCM-K-H		FCM		KFCM-K		KFCM-K-H	
Data sets	$m = 1.6$	$m = 2.0$	$m = 1.6$	$m = 2.0$	$m = 1.6$	$m = 2.0$	$m = 1.6$	$m = 2.0$	$m = 1.6$	$m = 2.0$	$m = 1.6$	$m = 2.0$
Breast tissue	0.1101	0.1252	0.2065	0.1143	0.2944	0.2934	0.3083	0.3218	0.3428	0.2766	0.5515	0.5356
Ecoli	0.3880	0.3682	0.3230	0.3277	0.4182	0.4015	0.5721	0.5514	0.5232	0.5162	0.6075	0.5927
Image segmentation	0.3116	0.3045	0.2133	0.2832	0.5257	0.4429	0.4920	0.4670	0.2931	0.3885	0.6444	0.6234
Iris	0.7163	0.7294	0.8015	0.7859	0.8856	0.9037	0.7419	0.7496	0.7899	0.7773	0.8641	0.8801
Leaf	0.3145	0.2654	0.3096	0.2877	0.3566	0.3602	0.6652	0.6477	0.6738	0.6538	0.7061	0.6981
Libras Movement	0.3227	0.1667	0.2726	0.1515	0.2419	0.2070	0.5821	0.3767	0.5315	0.4399	0.5239	0.4970
Multiple features	0.4280	0.4206	0.4128	0.3615	0.5324	0.4073	0.5669	0.5612	0.6053	0.5693	0.6397	0.6303
Seeds	0.7166	0.7166	0.7034	0.7034	0.6975	0.6954	0.6949	0.6949	0.6737	0.6737	0.6804	0.6716
Thyroid gland	0.5698	0.4413	0.0588	0.0495	0.1538	0.1819	0.4088	0.3434	0.1500	0.1353	0.2793	0.3240
Urban land cover	0.0356	0.0373	0.0737	0.0491	0.4289	0.2626	0.1345	0.1272	0.1909	0.1383	0.5011	0.3640
Brest cancer winsconsin	0.4810	0.4914	0.0215	0.0176	0.7178	0.7182	0.4567	0.4647	0.0593	0.0561	0.6174	0.6045
Wine	0.3602	0.3539	0.3711	0.3749	0.8332	0.8482	0.4212	0.4167	0.4287	0.4315	0.8199	0.8329

It can be observed that also in this case whatever the considered indexes (ARI and NMI), for the majority of the datasets, FCM, KFCM-K and KFCM-K-H algorithms performed better with the $m = 1.6$. Moreover, whatever the considered indexes and fuzziness parameters, the KFCM-K-H algorithm outperformed both the KFCM-K and FCM algorithms on the majority of the data sets of the Table 1. Besides, for the ARI index and whatever the considered fuzziness parameters, the standard FCM algorithm performed better than the KFCM-K algorithm on the majority of the datasets of the Table 1.

4 Final Remarks and Conclusions

The clustering performance of the conventional KFCM-K, the gaussian kernel-based fuzzy clustering algorithm, is highly related to the estimation of the bandwidth parameter of the Gaussian kernel function, that is estimated once and for all. In this paper we proposed KFCM-K-H, a Gaussian fuzzy c-Means with kernelization of the metric and automatic computation of a vector of bandwidth parameters, one for each variable. In the proposed kernel-based fuzzy clustering algorithm, the bandwidth parameters change at each iteration of the algorithm and are different from variable to variable. Thus, the proposed algorithm is able to select the important variables for the clustering task.

Experiments with twelve data sets from UCI machine learning repository, with different number of objects, variables and a priori classes, showed the performance of the proposed algorithm. It was observed that, for the majority of these data sets, the proposed KFCM-K-H algorithm provided crisp and fuzzy partitions of better quality than those provided by the conventional KFCM-K algorithm. Moreover, the KFCM-K-H algorithm provided crisp partitions of better quality than those provided by the standard FCM algorithm. Besides, it was observed that the FCM algorithm outperformed both KFCM-K and KFCM-K-H algorithms on the majority of these data sets, concerning the quality of the fuzzy partitions. These later finds support the remark provided by Ref. [19], i.e, that the kernelization may impose undesirable structures on the data and the clusters obtained in the kernel space may not exhibit the structure of the original data.

Acknowledgments. The authors are grateful to the anonymous referees for their careful revision, and CNPq and FACEPE (Brazilian agencies) for their financial support.

References

1. Bezdek, J.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum, New York (1981)
2. Blake, C.L., Merz, C.J.: UCI repository of machine learning databases. University of California, Department of Information and Computer Science, Irvine (1998). http://www.ics.uci.edu/mlearn/MLRepository.html
3. Camastra, F., Verri, A.: A novel kernel method for clustering. IEEE Trans. Neural Netw. **27**, 801–804 (2005)

4. Caputo, B., Sim, K., Furesjo, F., Smola, A.: Appearence-based object recognition using SVMs: which kernel should I use? In: Proceedings of NIPS Workshop on Statistical methods for Computational Experiments in Visual Processing and Computer Vision (2002)
5. de Carvalho, F.A.T., Ferreira, M.R.P., Simões, E.C.: A Gaussian kernel-based clustering algorithm with automatic hyper-parameters computation. In: Cheng, L., Liu, Q., Ronzhin, A. (eds.) ISNN 2016. LNCS, vol. 9719, pp. 393–400. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-40663-3_45
6. Cleuziou, G., Moreno, J.: Kernel methods for point symmetry-based clustering. Pattern Recogn. **48**, 2812–2830 (2015)
7. Diday, E., Govaert, G.: Classification automatique avec distances adaptatives. R.A.I.R.O. Inform. Comput. Sci. **11**(4), 329–349 (1977)
8. Fauvel, M., Chanussot, J., Benediktsson, J.: Parsimonious mahalanobis kernel for the classification of high dimensional data. Pattern Recogn. **46**, 845–854 (2013)
9. Filippone, M., Camastra, F., Masulli, F., Rovetta, S.: A survey of kernel and spectral methods for clustering. Pattern Recogn. **41**, 176–190 (2008)
10. Frigui, H., Hwanga, C., Rhee, F.C.H.: Clustering and aggregation of relational data with applications to image database categorization. Pattern Recogn. **40**, 3053–3068 (2007)
11. Huang, J., Ng, M., Rong, H., Li, Z.: Automated variable weighting in k-means type clustering. IEEE Trans. Pattern Anal. Mach. Intell. **27**(5), 657–668 (2005)
12. Hubert, L., Arabie, P.: Comparing partitions. J. Classif. **2**(1), 193–218 (1985)
13. Huellermeier, E., Rifki, M., Henzgen, S., Senge, R.: Comparing fuzzy partitions: a generalization of the rand index and related measures. IEEE Trans. Fuzzy Syst. **20**, 546–556 (2012)
14. Jain, A.K.: Data clustering: 50 years beyond k-means. Pattern Recogn. Lett. **31**, 651–666 (2010)
15. Kaufman, L., Rousseeuw, P.: Finding Groups in Data: An Introduction to Cluster Analysis. Wiley, Hoboken (2005)
16. Manning, C., Raghavan, P., Schuetze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)
17. Modha, D.S., Spangler, W.S.: Feature weighting in k-means clustering. Mach. Learn. **52**(3), 217–237 (2003)
18. Mueller, K.R., Mika, S., Raetsch, G., Tsuda, K., Schoelkopf, B.: An introduction to kernel-based learning algorithms. IEEE Trans. Neural Netw. **12**, 181–202 (2001)
19. Pal, N.R.: What and when can we gain from the kernel versions of c-means algorithm? IEEE Trans. Fuzzy Syst. **22**, 363–369 (2014)
20. Tsai, C., Chiu, C.: Developing a feature weight self-adjustment mechanism for a k-means clustering algorithm. Comput. Stat. Data Anal. **52**, 4658–4672 (2008)
21. Xu, R., Wunusch, D.I.I.: Survey of clustering algorithms. IEEE Trans. Neural Netw. **16**, 645–678 (2005)