Pattern Recognition 79 (2018) 370–386

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Gaussian kernel c-means hard clustering algorithms with automated computation of the width hyper-parameters

Francisco de A.T. de Carvalho[a,*], Eduardo C. Simões[a], Lucas V.C. Santana[a], Marcelo R.P. Ferreira[b]

[a] Centro de Informatica, Universidade Federal de Pernambuco, Av. Jornalista Anibal Fernandes s/n - Cidade Universitaria, CEP, Recife, PE 50740-560, Brazil
[b] Departamento de Estatistica, Centro de Ciências Exatas e da Natureza, Universidade Federal da Paraiba, CEP, João Pessoa, PB 58051-900, Brazil

ARTICLE INFO

Article history:
Received 26 December 2016
Revised 18 January 2018
Accepted 18 February 2018
Available online 19 February 2018

Keywords:
Gaussian kernel clustering
Kernelization of the metric
Feature space
Width hyper-parameter

ABSTRACT

Conventional Gaussian kernel c-means clustering algorithms are widely used in applications. However, Gaussian kernel functions have an important parameter, the width hyper-parameter, which needs to be tuned. Usually this parameter is tuned once and for all and it is the same for all variables. Thus, implicitly, all the variables are equally rescaled and therefore, they have equal importance on the clustering task. This paper presents Gaussian kernel c-means hard clustering algorithms with automated computation of the width hyper-parameters. In these kernel-based clustering algorithms, the hyper-parameters change at each iteration of the algorithm, they differ from variable to variable and can differ from cluster to cluster. Because each variable is rescaled differently according to its own hyper-parameter, these algorithms can select the important variables in the clustering process. Experiments using synthetic data sets and using UCI machine learning repository data sets corroborate the usefulness of the proposed algorithms.

1. Introduction

Clustering is a well-known task in pattern recognition that aims to divide a given unlabeled data set into clusters, whereby data objects within the same cluster are similar to each other and dissimilar to objects belonging to other clusters [25,31,32,57]. Clustering has been successfully used in different fields, including bioinformatics, image processing, and information retrieval.

C-means-type algorithms often use the Euclidean distance to compute the dissimilarity between the objects and the cluster representatives. This is suitable when the data is not corrupted by noise or outliers, and when the clusters are linearly separable. If this is not the case, the conventional c-means will not be able to provide effective results.

To overcome these limitations, kernel clustering methods have been proposed [5,7,13,19,22,26,35,53]. The fuzzy *c*-means [2] algorithm, self organizing maps (SOM) [36–39], the mountain method [58], and the neural gas algorithm [44] have been modified to incorporate kernels and a variety of clustering algorithms based on kernels have also been proposed [22]. Kernel-based fuzzy clustering methods have been proposed in [11,62]. Macdonald and Fyfe [42] and Inokuchi and Miyamoto [30] developed a kernelized version of SOM. Ref. [61] proposed a semi-supervised adaptive kernel clustering algorithm with metric learning. In [34], a kernel mountain method was presented, while in [49], a kernel version of the neural gas algorithm was proposed. The basic idea behind kernel-based techniques is to project the data into a higher dimensional space through nonlinear mapping. It is expected that in high-dimensional spaces it will be possible to obtain well-defined and linearly separable groups [24,50]. Refs. [26,33] have shown that kernelized clustering algorithms often outperform conventional clustering algorithms.

More recently, kernel clustering methods based on approximated kernel matrices and multiple kernels have been proposed. In Ref. [12], the authors offer an approximation schema for the kernel matrix based on randomization. A fast kernel matrix computation method is proposed in Ref. [52] in the context of big data clustering. Ref. [40] introduces two schemes for kernel fuzzy clustering based on random feature mapping and dimension reduction, whereas Ref. [55] evaluates the Nystrom approximation for kernel *k*-means clustering. Various algorithms based on multiple kernels were proposed in Refs. [1,17,27,41,63,64]. Applications on SAR images segmentation can be found in Refs. [56,60]. In Ref. [18], a kernel-based fuzzy *c*-means clustering algorithm based on a genetic algorithm is proposed, whereas Ref. [59] deals with robustness in kernel clustering methods.

* Corresponding author.
E-mail addresses: fatc@cin.ufpe.br, francisco.carvalho@pq.cnpq.br (F.d.A.T. de Carvalho), ecs4@cin.ufpe.br (E.C. Simões), lvcs@cin.ufpe.br (L.V.C. Santana), marcelo@de.ufpb.br (M.R.P. Ferreira).

https://doi.org/10.1016/j.patcog.2018.02.018
0031-3203/

There are two major variations of kernel-based clustering: one is the kernelization of the metric, where the cluster representatives are obtained in the original space, 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 [7,22].

In kernel-based clustering algorithms, Euclidean distances are computed using the so-called distance kernel trick [22]. This trick uses a kernel function to calculate the dot products of vectors implicitly in the higher dimensional space using the original space. Thus, Euclidean distances in the higher dimensional space are able to be measured using dot products.

Among the different kernel functions used in clustering algorithms, the Gaussian kernel is the most popular. In general, this kernel function provides effective results and requires the tuning of a single parameter, that is, the width hyper-parameter [8]. This parameter is tuned once and for all, and it is the same for all variables. Thus, implicitly the conventional Gaussian kernel 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 are irrelevant while others have different degrees of relevance to the clustering task. Moreover, each cluster may have its own different set of relevant variables [10,15,20,21,28,46,51,54].

Recently, Ref. [14] proposed a Gaussian kernel c-means hard clustering algorithm with kernelization of the metric, where each variable has its own hyper-parameter that is iteratively computed during the processing of the algorithm. This paper extends Ref. [14]; its main contribution is to provide Gaussian kernel c-means clustering algorithms, with both kernelization of the metric and in the feature space, and with automated computation of the width hyper-parameters using an adaptive Gaussian kernel. In these kernel-based clustering algorithms, the hyper-parameters change at each algorithm iteration, they differ from variable to variable, and can differ from cluster to cluster. 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 reviews the conventional kernel based c-means hard clustering algorithms. Section 3 provides the Gaussian kernel c-means hard clustering algorithms with automated computation of the width hyper-parameters. In Section 4, experiments using synthetic data sets and using UCI machine learning repository data sets corroborate the usefulness of the proposed algorithms. Section 5 provides the final remarks of the paper.

2. Conventional kernel c-means hard clustering algorithms

This section briefly reviews the basic concepts of kernel functions and the conventional Gaussian kernel c-means hard clustering algorithms.

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 $\mathbf{x}_k = (x_{k1}, \ldots, x_{kp}) \in \mathrm{R}^p$.

Let $\Phi : \mathcal{D} \rightarrow \mathcal{F}$ be a nonlinear mapping from the input space $\mathcal{D}$ to a high dimensional feature space $\mathcal{F}$. By applying the mapping Φ, the dot 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. In the kernel-based approaches the nonlinear mapping Φ does not need to be explicitly specified because each Mercer kernel can be expressed as $\mathcal{K}(\mathbf{x}_l, \mathbf{x}_k) = \Phi(\mathbf{x}_l)^T \Phi(\mathbf{x}_k)$ [47,50].

One the most relevant aspects in clustering and other tasks lies in the possibility to compute Euclidean distances in $\mathcal{F}$ without explicitly knowing Φ, by using the so-called distance kernel trick [22,26,47,50]:

$$\begin{aligned} &||\Phi(\mathbf{x}_l) - \Phi(\mathbf{x}_k)||^2 \\ &= (\Phi(\mathbf{x}_l) - \Phi(\mathbf{x}_k)^T (\Phi(\mathbf{x}_l) - \Phi(\mathbf{x}_k) \\ &= \Phi(\mathbf{x}_l)^T \Phi(\mathbf{x}_l) - 2\Phi(\mathbf{x}_l)^T \Phi(\mathbf{x}_k) + \Phi(\mathbf{x}_k)^T \Phi(\mathbf{x}_k) \\ &= \mathcal{K}(\mathbf{x}_l, \mathbf{x}_l) - 2\mathcal{K}(\mathbf{x}_l, \mathbf{x}_k) + \mathcal{K}(\mathbf{x}_k, \mathbf{x}_k). \end{aligned} \quad (1)$$

There are two main variants of kernel c-means clustering methods: kernelization of the metric [62] and clustering in feature space [24]. Variants based on the kernelization of the metric [62] search the cluster representatives in the original input space and the distances between objects and representatives are obtained by means of kernels. Clustering in feature space [24] then occurs by mapping each object through a non-linear function Φ and then obtaining the distances between the objects and the cluster representatives in the high dimensional feature space $\mathcal{F}$ by means of the kernel trick. Because the cluster representatives are not computed directly, clustering in feature space is often faster than the variants based on the kernelization of the metric.

2.1. Gaussian kernel c-means with kernelization of the metric

Starting from an initial solution, the kernel c-means with kernelization of the metric (henceforth named KCM-K) iteratively provides a partition $\mathcal{P}$ of E into c clusters $P_i \subset E (1 \leq i \leq c)$ in two steps, and c representatives (henceforth named prototypes) $\mathbf{g}_i = (g_{i1}, \ldots, g_{ip}) \in \mathrm{R}^p$ $(1 \leq i \leq c)$ of the clusters, by the minimization of a suitable objective function, denoted below as J_{KCM-K}, which gives the total heterogeneity of the partition computed as the sum of the heterogeneity in each cluster:

$$\begin{aligned} J_{KCM-K} &= \sum_{i=1}^{c} \sum_{e_k \in P_i} ||\Phi(\mathbf{x}_k) - \Phi(\mathbf{g}_i)||^2 \\ &= \sum_{i=1}^{c} \sum_{e_k \in P_i} \{\mathcal{K}(\mathbf{x}_k, \mathbf{x}_k) - 2\mathcal{K}(\mathbf{x}_k, \mathbf{g}_i) + \mathcal{K}(\mathbf{g}_i, \mathbf{g}_i)\} \end{aligned} \quad (2)$$

Next, the Gaussian kernel, the most commonly used in the literature, is considered:

$$\mathcal{K}(\mathbf{x}_l, \mathbf{x}_k) = \exp\left\{-\frac{||\mathbf{x}_l - \mathbf{x}_k||^2}{2\sigma^2}\right\} = \exp\left\{-\frac{1}{2}\sum_{j=1}^{p} \frac{1}{\sigma^2}(x_{lj} - x_{kj})^2\right\}, \quad (3)$$

where σ^2 is the width hyper-parameter of the Gaussian kernel. Note that σ^2 is the same for all the variables and it is tuned once and for all, for example, such that $2\sigma^2$ is the average of the 0.1 and 0.9 quantiles of $||\mathbf{x}_i - \mathbf{x}_k||^2$, $i, k = 1, \ldots, p$, $i \neq k$ [8]. Therefore, the objective function J_{KMC-K} becomes:

$$J_{KCM-K} = \sum_{i=1}^{c} \sum_{e_k \in P_i} 2\,(1 - \mathcal{K}(\mathbf{x}_k, \mathbf{g}_i)) \quad (4)$$

The representation step provides the cluster prototypes $\mathbf{g}_i (1 \leq i \leq c)$, which are computed as follows:

$$\mathbf{g}_i = \frac{\sum_{e_k \in P_i} \mathcal{K}(\mathbf{x}_k, \mathbf{g}_i)\mathbf{x}_k}{\sum_{e_k \in P_i} \mathcal{K}(\mathbf{x}_k, \mathbf{g}_i)}. \quad (5)$$

In the allocation step, the clusters $P_i (1 \leq i \leq c)$ are updated according to the following allocation rule:

$$P_i = \left\{e_k \in E : (1 - \mathcal{K}(\mathbf{x}_k, \mathbf{g}_i)) = \min_{h=1}^{c} (1 - \mathcal{K}(\mathbf{x}_k, \mathbf{g}_h))\right\} \quad (6)$$

These two steps are repeated up to the convergence of the KCM-K algorithm.

2.2. Gaussian kernel c-means in the feature space

The particularity of the kernel c-means algorithm in the feature space (henceforth named KCM-F) is that the representatives g_i^{Φ} of the clusters $P_i\,(1 \le i \le c)$ are not in the original space but instead in a high dimensional space $\mathcal{F}$. Therefore, they cannot be provided in the original space by the KCM-F algorithm.

Starting from an initial solution, the partition $\mathcal{P}$ is obtained iteratively in two steps (representation and allocation) by the minimization of a suitable objective function, denoted below as J_{KCM-F}, which gives the total heterogeneity of the partition computed as the sum of the heterogeneity in each cluster:

$$J_{KCM-F} = \sum_{i=1}^{c} \sum_{e_k \in P_i} ||\Phi(\mathbf{x}_k) - \mathbf{g}_i^{\Phi}||^2 \tag{7}$$

The representation step does not provide the cluster representatives, because they cannot be provided in the original space by the KCM-F algorithm, but it does provide the distances between $\Phi(\mathbf{x}_k)$ and $\mathbf{g}_i^{\Phi}$ $(1 \le k \le n;\ 1 \le i \le c)$ in the feature space through the kernel in the original space, as follows:

$$\begin{aligned} ||\Phi(\mathbf{x}_k) - \mathbf{g}_i^{\Phi}||^2 &= (\Phi(\mathbf{x}_k))^T \Phi(\mathbf{x}_k) - 2(\Phi(\mathbf{x}_k))^T \mathbf{g}_i^{\Phi} + (\mathbf{g}_i^{\Phi})^T \mathbf{g}_i^{\Phi} \\ &= (\Phi(\mathbf{x}_k))^T \Phi(\mathbf{x}_k) - \frac{2\sum_{e_l \in P_i} (\Phi(\mathbf{x}_k))^T \Phi(\mathbf{x}_l)}{|P_i|} \\ &\quad + \frac{\sum_{e_r \in P_i} \sum_{e_s \in P_i} (\Phi(\mathbf{x}_r))^T \Phi(\mathbf{x}_s)}{|P_i|^2} \\ &= \mathcal{K}(\mathbf{x}_k, \mathbf{x}_k) - \frac{2\sum_{e_l \in P_i} \mathcal{K}(\mathbf{x}_k, \mathbf{x}_l)}{|P_i|} \\ &\quad + \frac{\sum_{e_r \in P_i} \sum_{e_s \in P_i} \mathcal{K}(\mathbf{x}_r, \mathbf{x}_s)}{|P_i|^2} \end{aligned}$$

Here we consider once again the Gaussian kernel Eq. (3). Therefore:

$$||\Phi(\mathbf{x}_k) - \mathbf{g}_i^{\Phi}||^2 = 1 - \frac{2\sum_{e_l \in P_i} \mathcal{K}(\mathbf{x}_k, \mathbf{x}_l)}{|P_i|} + \frac{\sum_{e_r \in P_i} \sum_{e_s \in P_i} \mathcal{K}(\mathbf{x}_r, \mathbf{x}_s)}{|P_i|^2}$$

In the allocation step, the clusters $P_i\,(1 \le i \le c)$, are updated according to the following allocation rule:

$$P_i = \left\{ e_k \in E : ||\Phi(\mathbf{x}_k) - \mathbf{g}_i^{\Phi}||^2 = \min_{h=1}^{c} ||\Phi(\mathbf{x}_k) - \mathbf{g}_h^{\Phi}||^2 \right\} \tag{8}$$

These two steps are repeated until the convergence of KCM-F.

3. Gaussian kernel c-means hard clustering algorithms with automated computation of width hyper-parameters

This section presents the Gaussian kernel c-means hard clustering algorithms, with both kernelization of the metric and in the feature space, and with automated computation of width hyper-parameters. These hyper-parameters change at each iteration of the algorithms. They differ from variable to variable and can also differ from cluster to cluster. Therefore, these algorithms are able to rescale the variables differently and thus select the relevant ones for the clustering task.

3.1. Gaussian kernel c-means with kernelization of the metric and automated computation of width hyper-parameters

Starting from an initial solution, the kernel c-means with kernelization of the metric and automatic computation of width hyper-parameters hard clustering algorithm (henceforth named KCM-K-H) iteratively provides, in three steps (described in Section 3.1.1. of this paper), a partition $\mathcal{P}$ of E into c clusters $P_i\,(1 \le i \le c)$, width hyper-parameters for the variables, and a matrix of prototypes $\mathbf{G} = (\mathbf{g}_1, \ldots, \mathbf{g}_c)$ of the clusters in the partition $\mathcal{P}$, by the minimization of a suitable objective function, denoted below as $J_{KCM-K-H}$, which provides the total heterogeneity of the partition computed as the sum of the heterogeneity in each cluster:

Next, we consider two variants of the $KCM-K-H$ algorithm provided by two Gaussian kernel functions:

(a) the KCM-K-GH variant was introduced in Ref. [14]. This variant is based on a Gaussian kernel function with a global vector of width hyper-parameters $\mathbf{s} = (s_1^2, \ldots, s_p^2)$:

$$\mathcal{K}^{(\mathbf{s})}(\mathbf{x}_l, \mathbf{x}_k) = \exp\left\{ -\frac{1}{2} \sum_{j=1}^{p} \frac{1}{s_j^2} (x_{lj} - x_{kj})^2 \right\} \tag{9}$$

Note that each variable has its own hyper-parameter $s_j^2\ (1 \le j \le p)$. In this case:

$$||\Phi(\mathbf{x}_k) - \Phi(\mathbf{g}_i)||^2 = \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{x}_k) - 2\mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i) + \mathcal{K}^{(\mathbf{s})}(\mathbf{g}_i, \mathbf{g}_i) \tag{10}$$

Therefore, the objective function becomes:

$$J_{KCM-K-GH} = \sum_{i=1}^{c} \sum_{e_k \in P_i} 2\,(1 - \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)) \tag{11}$$

(b) the KCM-K-LH variant is based on a Gaussian kernel function with a local vector of width hyper-parameters $\mathbf{s}_i = (s_{i1}^2, \ldots, s_{ip}^2),\ 1 \le i \le c$,

$$\mathcal{K}^{(\mathbf{s}_i)}(\mathbf{x}_l, \mathbf{x}_k) = \exp\left\{ -\frac{1}{2} \sum_{j=1}^{p} \frac{1}{s_{ij}^2} (x_{lj} - x_{kj})^2 \right\} \tag{12}$$

Note that each variable has its own hyper-parameter in each cluster $s_{ij}^2\ (1 \le i \le c;\ 1 \le j \le p)$. In this case, the objective function becomes:

$$J_{KCM-K-LH} = \sum_{i=1}^{c} \sum_{e_k \in P_i} 2\,(1 - \mathcal{K}^{(\mathbf{s}_i)}(\mathbf{x}_k, \mathbf{g}_i)) \tag{13}$$

3.1.1. The KCM-K-GH and KCM-K-LH algorithms

This section describes the three respective steps of the $KCM-K-GH$ and $KCM-K-LH$ algorithms.

3.1.1.1. Representation step. This step provides the optimal solution to the computation of the cluster representatives.

In the representation step of the KCM-K-GH and KCM-K-LH algorithms, the partition $\mathcal{P}$ and, respectively, the global vector of width hyper-parameters $\mathbf{s}$ and the local vectors of width hyper-parameters $\mathbf{s}_1, \ldots, \mathbf{s}_c$, are kept fixed. The objective functions $J_{KMC-K-GH}$ and $J_{KMC-K-LH}$ are optimized with respect to the prototypes.

Proposition 1.

(i) *The objective function of the KCM-K-GH algorithm is minimized with respect to the cluster prototypes* $\mathbf{g}_i\ (i = 1, \ldots, c)$ *when they are computed as follows*

$$\mathbf{g}_i = \frac{\sum_{e_k \in P_i} \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i) \mathbf{x}_k}{\sum_{e_k \in P_i} \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)},\ 1 \le i \le c. \tag{14}$$

(ii) *The objective function of the KCM-K-LH algorithm is minimized with respect to the cluster prototypes* $\mathbf{g}_i\ (i = 1, \ldots, c)$ *when they are computed as follows*

$$\mathbf{g}_i = \frac{\sum_{e_k \in P_i} \mathcal{K}^{(\mathbf{s}_i)}(\mathbf{x}_k, \mathbf{g}_i) \mathbf{x}_k}{\sum_{e_k \in P_i} \mathcal{K}^{(\mathbf{s}_i)}(\mathbf{x}_k, \mathbf{g}_i)},\ 1 \le i \le c. \tag{15}$$

Proof. From $\frac{\partial J_{KMC-K-GH}}{\partial \mathbf{g}_i} = 0$ and after some algebra, the cluster prototypes provided by the KCM-K-GH algorithm are obtained according to Eq. (14). In a similar way, the cluster prototypes provided by the KCM-K-LH algorithm are obtained by using Eq. (15). □

3.1.1.2. Computation of the width hyper-parameters step. This step provides the optimal solution to the computation of the width hyper-parameters.

During the computation of the width hyper-parameters step of algorithms KCM-K-GH and KCM-K-LH, the matrix of prototypes $\mathbf{G}$ and the partition $\mathcal{P}$ are kept fixed.

Proposition 2.

(i) *The objective function of the KCM-K-GH algorithm is minimized with respect to the global vector of hyper-parameters* $\mathbf{s}$ *when its components* s_j^2 $(j = 1, \ldots, p)$ *are computed as follows*

$$\frac{1}{s_j^2} = \frac{\gamma^{\frac{1}{p}} \left\{ \prod_{h=1}^{p} \left[\sum_{i=1}^{c} \sum_{e_k \in P_i} \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)(x_{kh} - g_{ih})^2 \right] \right\}^{\frac{1}{p}}}{\sum_{i=1}^{c} \sum_{e_k \in P_i} \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)(x_{kj} - g_{ij})^2}. \quad (16)$$

(ii) *The objective function of the KCM-K-LH algorithm is minimized with respect to the local vectors of hyper-parameters* $\mathbf{s}_i$ $(i = 1, \ldots, c)$ *when their components* s_{ij} $(j = 1, \ldots, p)$ *are computed as follows*

$$\frac{1}{s_{ij}^2} = \frac{\gamma^{\frac{1}{p}} \left\{ \prod_{h=1}^{p} \left[\sum_{e_k \in P_i} \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)(x_{kh} - g_{ih})^2 \right] \right\}^{\frac{1}{p}}}{\sum_{e_k \in P_i} \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)(x_{kj} - g_{ij})^2}. \quad (17)$$

Proof. For the KCM-K-GH algorithm, first, we use the method of Lagrange multipliers with the restriction [15] that $\prod_{j=1}^{p} (\frac{1}{s_j^2}) = \gamma$, where γ is a suitable parameter, and obtain

$$L_{KCM-K-GH} = J_{KCM-K-GH} - \omega \left(\prod_{j=1}^{p} \frac{1}{s_j^2} - \gamma \right)$$
$$= \sum_{i=1}^{c} \sum_{e_k \in P_i} 2\left(1 - \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)\right) - \omega \left(\prod_{j=1}^{p} \frac{1}{s_j^2} - \gamma \right).$$

Then, the partial derivatives of $L_{KCM-K-GH}$ w.r.t ω and $\frac{1}{s_j^2}$ $(j = 1, \ldots, p)$ are computed, and, by setting the partial derivatives to zero and after some algebra, the following is obtained:

$$\frac{1}{s_j^2} = \frac{\gamma^{\frac{1}{p}} \left\{ \prod_{h=1}^{p} \left[\sum_{i=1}^{c} \sum_{e_k \in P_i} \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)(x_{kh} - g_{ih})^2 \right] \right\}^{\frac{1}{p}}}{\sum_{i=1}^{c} \sum_{e_k \in P_i} \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)(x_{kj} - g_{ij})^2}.$$

Note that another useful restriction [28] is to impose $\sum_{j=1}^{p} (\frac{1}{s_j^2}) = \gamma$, however, this latter method requires the tuning of an additional parameter in this optimization step and, for the sake of brevity, will not be considered here.

Similarly, for the KCM-K-LH algorithm, the Lagrange multipliers method is used with the restriction that $\prod_{j=1}^{p} (\frac{1}{s_{ij}^2}) = \gamma$, where γ is a suitable parameter, which results in:

$$L_{KCM-K-LH} = J_{KCM-K-LH} - \sum_{i=1}^{c} \omega_i \left(\prod_{j=1}^{p} \frac{1}{s_{ij}^2} - \gamma \right)$$
$$= \sum_{i=1}^{c} \sum_{e_k \in P_i} 2\left(1 - \mathcal{K}^{(\mathbf{s}_i)}(\mathbf{x}_k, \mathbf{g}_i)\right) - \sum_{i=1}^{c} \omega_i \left(\prod_{j=1}^{p} \frac{1}{s_{ij}^2} - \gamma \right).$$

Then, the partial derivatives of $L_{KCM-K-LH}$ w.r.t ω_i and $\frac{1}{s_{ij}^2}$ $(i = 1, \ldots, c; j = 1, \ldots, p)$ are computed and by setting the partial derivatives to zero, and after some algebra, the following is obtained:

$$\frac{1}{s_{ij}^2} = \frac{\gamma^{\frac{1}{p}} \left\{ \prod_{h=1}^{p} \left[\sum_{e_k \in P_i} \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)(x_{kh} - g_{ih})^2 \right] \right\}^{\frac{1}{p}}}{\sum_{e_k \in P_i} \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)(x_{kj} - g_{ij})^2}.$$

□

Note that according to Eq. (16), the closer the objects are to the set of cluster representatives for a variable j, the higher is $\frac{1}{s_j^2}$ and, therefore, the lower is the width hyper-parameter s_j. Moreover, according to Eq. (17), the closer the objects are to the representative of a cluster i for a variable j, the higher is $\frac{1}{s_{ij}^2}$, and therefore, the lower is the width hyper-parameter s_{ij}.

3.1.1.3. Allocation step. This step provides the optimal solution for the cluster partition.

During the allocation step of the KCM-K-GH and KCM-K-LH algorithms, the cluster prototypes $\mathbf{g}_1, \ldots, \mathbf{g}_c$ and, respectively, the global vector of width hyper-parameters $\mathbf{s}$ and the local vectors of width hyper-parameters $\mathbf{s}_1, \ldots, \mathbf{s}_c$, are kept fixed.

Proposition 3.

(i) *The objective function of the KCM-K-GH algorithm is minimized with respect to the partition* $\mathcal{P}$ *when the clusters* P_i $(i = 1, \ldots, c)$ *are updated according to the following allocation rule:*

$$P_i = \left\{ e_k \in E : 2\left(1 - \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)\right) = \min_{h=1}^{c} 2\left(1 - \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_h)\right) \right\} \quad (18)$$

(ii) *The objective function of the KCM-K-LH algorithm is minimized with respect to the partition* $\mathcal{P}$ *when the clusters* P_i $(i = 1, \ldots, c)$ *are updated according to the following allocation rule:*

$$P_i = \left\{ e_k \in E : 2\left(1 - \mathcal{K}^{(\mathbf{s}_i)}(\mathbf{x}_k, \mathbf{g}_i)\right) = \min_{h=1}^{c} 2\left(1 - \mathcal{K}^{(\mathbf{s}_h)}(\mathbf{x}_k, \mathbf{g}_h)\right) \right\} \quad (19)$$

Proof. The aim is to minimize $J_{KCM-K-GH}$ with respect to the partition $\mathcal{P}$. As the cluster prototypes $\mathbf{g}_1, \ldots, \mathbf{g}_c$ and the global vector of width hyper-parameters $\mathbf{s}$ are kept fixed, we can rewrite $J_{KCM-K-GH}$ as

$$J_{KCM-K-GH} = \sum_{k=1}^{n} 2\left(1 - \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_{f(\mathbf{x}_k)})\right)$$

where $f : \mathcal{D} \to \{1, \ldots, c\}$ is the allocation function that associates each data unit $\mathbf{x}_k$ to an index $h = f(\mathbf{x}_k)$ of the index set $\{1, \ldots, c\}$.

The objective function $J_{KCM-K-GH}$ is minimized if, for each $\mathbf{x}_k \in \mathcal{D}$, $2\left(1 - \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_{f(\mathbf{x}_k)})\right)$ is minimized. For fixed cluster prototypes $\mathbf{g}_1, \ldots, \mathbf{g}_c$ and global vector of width hyper-parameters $\mathbf{s}$, $\mathbf{x}_k \in \mathcal{D}$, $2\left(1 - \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_{f(\mathbf{x}_k)})\right)$ is minimized if $f(\mathbf{x}_k) = i = arg\min_{1 \le h \le c} 2\left(1 - \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_h)\right)$, i.e., if $P_i = \{e_k \in E : 2\left(1 - \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)\right) = \min_{h=1}^{c} 2\left(1 - \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_h)\right)\}$.

Similarly, the objective function $J_{KCM-K-LH}$ is minimized with respect to partition $\mathcal{P}$ if the clusters $P_i (1 \le i \le c)$ are updated according to $P_i = \{e_k \in E : 2\left(1 - \mathcal{K}^{(\mathbf{s}_i)}(\mathbf{x}_k, \mathbf{g}_i)\right) = \min_{h=1}^{c} 2\left(1 - \mathcal{K}^{(\mathbf{s}_h)}(\mathbf{x}_k, \mathbf{g}_h)\right)\}$. □

Note that, according to Proposition 3, the assignment of the objects to the clusters is based on the computation of the dissimilarity between the objects and cluster representatives. A variable that has a small (big) width hyper-parameter contributes strongly (weekly) to the computation of the dissimilarity between objects and cluster representatives. It is for this reason, in this sense, that a variable that is rescaled by a small width hyper-parameter is more relevant to the clustering task than a variable which is rescaled by a big width hyper-parameter. Therefore, KCM-K-GH is able to select the important variable for the whole partition whereas KCM-K-LH is able to select the important variables for a specific cluster.

3.1.1.4. KCM-K-GH and KCM-K-LH algorithms. These three steps are repeated until the convergence of the KCM-K-GH and KCM-K-LH clustering algorithms. The Algorithm 1 summarizes these steps.

Algorithm 1 KCM-K-GH and KCM-K-LH algorithms.

input
  $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ (the data set); c (the number of clusters); $\gamma > 0$ (a suitable parameter);
Initialization
  Randomly select c distinct prototypes $g_i \in \mathcal{D}$ $(1 \leq i \leq c)$;
  KCM-K-GH: set $\frac{1}{s_j^2} = (\gamma)^{\frac{1}{p}}$ $(1 \leq j \leq p)$ or randomly set $\frac{1}{s_j^2}$ such that $\frac{1}{s_j^2} > 0$ and $\prod_{j=1}^{p} \frac{1}{s_j^2} = \gamma$;
  KCM-K-LH: set $\frac{1}{s_{ij}^2} = (\gamma)^{\frac{1}{p}}$ $(1 \leq i \leq c; 1 \leq j \leq p)$ or randomly set $\frac{1}{s_{ij}^2}$ such that $\frac{1}{s_{ij}^2} > 0$ and $\prod_{j=1}^{p} \frac{1}{s_{ij}^2} = \gamma$;
  KCM-K-GH: assign the object e_k $(1 \leq k \leq n)$ to the cluster P_i $(1 \leq i \leq c)$ if $2(1 - \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i)) = \min_{h=1}^{c} 2(1 - \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_h))$
  KCM-K-LH: assign the object e_k $(1 \leq k \leq n)$ to the cluster P_i $(1 \leq i \leq c)$ if $2(1 - \mathcal{K}^{(\mathbf{s}_i)}(\mathbf{x}_k, \mathbf{g}_i)) = \min_{h=1}^{c} 2(1 - \mathcal{K}^{(\mathbf{s}_h)}(\mathbf{x}_k, \mathbf{g}_h))$
repeat
  Step 1: representation.
    KCM-K-GH: compute the cluster representatives $\mathbf{g}_1, \ldots, \mathbf{g}_c$ using equation (14);
    KCM-K-LH: compute the cluster representatives $\mathbf{g}_1, \ldots, \mathbf{g}_c$ using equation (15);
  Step 2: computation of the width hyper-parameter
    KMC-K-GH: compute the global vector of width hyper-parameters $\mathbf{s}$ using equation (16);
    KCM-K-LH: compute the local vectors of width hyper-parameters $\mathbf{s}_1, \ldots, \mathbf{s}_c$ using equation (17);
  Step 3: allocation
  $test \leftarrow 0$;
  for $k = 1$ to n **do**
    KCM-K-GH: compute the winning cluster P_h such that $2(1 - \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_l)) = \min_{h=1}^{c} 2(1 - \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_h))$
    KCM-K-LH: compute the winning cluster P_h such that $2(1 - \mathcal{K}^{(\mathbf{s}_h)}(\mathbf{x}_k, \mathbf{g}_l)) = \min_{h=1}^{c} 2(1 - \mathcal{K}^{(\mathbf{s}_l)}(\mathbf{x}_k, \mathbf{g}_h))$
    if $\mathbf{x}_k \in P_i$ and $l \neq i$ **then**
      $test \leftarrow 1$; $P_l = P_l \cup \{\mathbf{x}_k\}$; $P_i = P_i \setminus \{\mathbf{x}_k\}$;
    end if
  end for
until $test = 0$
Output
  the cluster representatives $\mathbf{g}_1, \ldots, \mathbf{g}_c$;
  KCM-K-GH: the global vector of hyper-parameters $\mathbf{s}$;
  KCM-K-LH: the local vectors of hyper-parameters $\mathbf{s}_1, \ldots, \mathbf{s}_c$;
  Both KCM-K-GH and KCM-K-LH : the partition $\mathcal{P}$.

3.2. Gaussian kernel c-means in the feature space and automated computation of width hyper-parameters

The kernel c-means in the feature space and automated computation of width hyper-parameters hard clustering algorithm (henceforth named KCM-F-H) provides a partition $\mathcal{P}$ of E into c clusters P_i $(1 \leq i \leq c)$ and width hyper-parameters for the variables. As in the conventional kernel c-means in feature space (KCM-F), the representatives g_i^{Φ} of the clusters P_i $(1 \leq i \leq c)$ are not in the original space but in a high dimensional space $\mathcal{F}$. Therefore, they are not provided by the KCM-F-H algorithm.

Starting from an initial solution, the width hyper-parameters for the variables and the partition $\mathcal{P}$ are obtained interactively in three steps (representation, computation of the width hyper-parameters and allocation) by the minimization of a suitable objective function, here-below denoted as $J_{KCM-F-H}$, that provides the total heterogeneity of the partition computed as the sum of the heterogeneity in each cluster:

$$J_{KCM-F-H} = \sum_{i=1}^{c} \sum_{e_k \in P_i} ||\Phi(\mathbf{x}_k) - \mathbf{g}_i^{\Phi}||^2 \tag{20}$$

In what follows, we consider two variants of the KCM-F-H algorithm provided by two Gaussian kernel functions:

(a) The KCM-F-GH variant based on a Gaussian kernel function with a global vector of width hyper-parameters (one for each variable) $\mathbf{s} = (s_1^2, \ldots, s_p^2)$ (see Eq. (9)) and;
(b) The KCM-F-LH variant based on a Gaussian kernel function with a local vector of width hyper-parameters (one for each variable in a given cluster) $\mathbf{s}_i = (s_{i1}^2, \ldots, s_{ip}^2)$ $(1 \leq i \leq c)$ (see Eq. (12)).

3.2.1. The KCM-F-GH and KCM-F-LH algorithms

This section describes the three respective steps of the KCM-F-GH and KCM-F-LH algorithms.

3.2.1.1. Representation step. As in the conventional KCM-F algorithm, this representation step cannot provide the cluster representatives because they are not in the original space but in a high dimensional space $\mathcal{F}$ and the non-linear mapping Φ is not explicitly known. Instead, the representation step of the KCM-F-GH and of the KCM-F-LH algorithms provides the optimal distances between $\Phi(\mathbf{x}_k)$ and $\mathbf{g}_i^{\Phi}$ $(1 \leq k \leq n; 1 \leq i \leq c)$ in the feature space through the kernel in the original space.

During the representation step of the KCM-F-GH and KCM-F-LH algorithms, the partition $\mathcal{P}$ and, respectively, the global vector of width hyper-parameters $\mathbf{s}$ and the local vectors of width hyper-parameters $\mathbf{s}_1, \ldots, \mathbf{s}_c$, are kept fixed.

Proposition 4.

(i) *The objective function of the KCM-F-GH algorithm is minimized with respect to the distances between* $\Phi(\mathbf{x}_k)$ *and* $\mathbf{g}_i^{\Phi}$ $(1 \leq k \leq n; 1 \leq i \leq c)$ *in the feature space when they are computed as follows:*

$$||\Phi(\mathbf{x}_k) - \mathbf{g}_i^{\Phi}||^2 = 1 - \frac{2\sum_{e_l \in P_i} \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{x}_l)}{|P_i|} + \frac{\sum_{e_r \in P_i} \sum_{e_s \in P_i} \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_r, \mathbf{x}_s)}{|P_i|^2}; \tag{21}$$

(ii) *The objective function of the KCM-F-LH algorithm is minimized with respect to the distances between* $\Phi(\mathbf{x}_k)$ *and* $\mathbf{g}_i^{\Phi}$ $(1 \leq k \leq n; 1 \leq i \leq c)$ *in the feature space when they are computed as follows:*

$$||\Phi(\mathbf{x}_k) - \mathbf{g}_i^{\Phi}||^2 = 1 - \frac{2\sum_{e_l \in P_i} \mathcal{K}^{(\mathbf{s}_i)}(\mathbf{x}_k, \mathbf{x}_l)}{|P_i|} + \frac{\sum_{e_r \in P_i} \sum_{e_s \in P_i} \mathcal{K}^{(\mathbf{s}_i)}(\mathbf{x}_r, \mathbf{x}_s)}{|P_i|^2}. \tag{22}$$

Proof. First, from $\frac{\partial J_{KMC-F-GH}}{\partial \mathbf{g}_i} = 0$, and from $\frac{\partial J_{KMC-F-LH}}{\partial \mathbf{g}_i} = 0$, and after some algebra, the optimal cluster prototypes in the feature space are obtained as follows:

$$\mathbf{g}_i^{\Phi} = \frac{1}{|P_i|} \sum_{e_k \in P_i} \Phi(\mathbf{x}_k), \ (1 \leq i \leq c). \tag{23}$$

where $|P_i|$ is the cardinality of cluster P_i.

Then, for the KCM-F-GH algorithm, the optimal distances between $\Phi(\mathbf{x}_k)$ and $\mathbf{g}_i^{\Phi}$ $(k = 1, \ldots, n; i = 1, \ldots, c)$ are obtained using the distance kernel trick $||\Phi(\mathbf{x}_k) - \mathbf{g}_i^{\Phi}||^2 = (\Phi(\mathbf{x}_k))^T \Phi(\mathbf{x}_k) - 2(\Phi(\mathbf{x}_k))^T \mathbf{g}_i^{\Phi} + (\mathbf{g}_i^{\Phi})^T \mathbf{g}_i^{\Phi}$, the Gaussian kernel function with a global vector of width hyper-parameters Eq. (9) and the optimal cluster prototypes in the feature space Eq. (23), as follows:

$||\Phi(\mathbf{x}_k) - \mathbf{g}_i^{\Phi}||^2$

$$= (\Phi(\mathbf{x}_k))^T\Phi(\mathbf{x}_k) - 2(\Phi(\mathbf{x}_k))^T\mathbf{g}_i^\Phi + (\mathbf{g}_i^\Phi)^T\mathbf{g}_i^\Phi$$
$$= 1 - \frac{2\sum_{e_l\in P_i}\mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k,\mathbf{x}_l)}{|P_i|} + \frac{\sum_{e_r\in P_i}\sum_{e_s\in P_i}\mathcal{K}^{(\mathbf{s})}(\mathbf{x}_r,\mathbf{x}_s)}{|P_i|^2}$$

Similarly, the representation step of KCM-F-LH algorithm provides the optimal distances between $\Phi(\mathbf{x}_k)$ and $\mathbf{g}_i^\Phi$ $(k = 1,\ldots,n; i = 1,\ldots,c)$ in the feature space through the kernel in the original space, as follows:

$$||\Phi(\mathbf{x}_k) - \mathbf{g}_i^\Phi||^2$$
$$= (\Phi(\mathbf{x}_k))^T\Phi(\mathbf{x}_k) - 2(\Phi(\mathbf{x}_k))^T\mathbf{g}_i^\Phi + (\mathbf{g}_i^\Phi)^T\mathbf{g}_i^\Phi$$
$$= 1 - \frac{2\sum_{e_l\in P_i}\mathcal{K}^{(\mathbf{s}_i)}(\mathbf{x}_k,\mathbf{x}_l)}{|P_i|} + \frac{\sum_{e_r\in P_i}\sum_{e_s\in P_i}\mathcal{K}^{(\mathbf{s}_i)}(\mathbf{x}_r,\mathbf{x}_s)}{|P_i|^2}$$

□

3.2.1.2. Computation of the width hyper-parameters step. This step provides the optimal solution to the computation of the width hyper-parameters.

During the computation of the width hyper-parameters step of the KCM-F-GH and KCM-F-LH algorithms, the partition $\mathcal{P}$ and, respectively, the distances $||\Phi(\mathbf{x}_k) - \mathbf{g}_i^\Phi||^2$ (Eq. (21)) and $||\Phi(\mathbf{x}_k) - \mathbf{g}_i^\Phi||^2$ (Eq. (22)) between $\Phi(\mathbf{x}_k)$ and $\mathbf{g}_i^\Phi$ $(1 \le k \le n; 1 \le i \le c)$, are kept fixed.

Proposition 5.

(i) *The objective function of the KCM-F-GH algorithm is minimized with respect to the global vector of hyper-parameters* $\mathbf{s}$ *when its components* s_j^2 $(j = 1,\ldots,p)$ *are computed as follows*

$$\frac{1}{s_j^2} = \frac{\gamma^{\frac{1}{p}}\left\{\prod_{h=1}^p \pi_h\right\}^{\frac{1}{p}}}{\pi_j}. \tag{24}$$

where

$$\pi_h = \sum_{i=1}^c\sum_{e_k\in P_i}\left\{-\frac{2\sum_{e_l\in P_i}\mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k,\mathbf{x}_l)(x_{kh}-x_{lh})^2}{|P_i|} + \frac{\sum_{e_r\in P_i}\sum_{e_s\in P_i}\mathcal{K}^{(\mathbf{s})}(\mathbf{x}_r,\mathbf{x}_s)(x_{rh}-x_{sh})^2}{|P_i|^2}\right\} \quad (1 \le h \le p)$$

(ii) *The objective function of the KCM-F-LH algorithm is minimized with respect to the local vectors of hyper-parameters* $\mathbf{s}_i$ $(i = 1,\ldots,c)$ *when their components* s_{ij} $(j = 1,\ldots,p)$ *are computed as follows*

$$\frac{1}{s_{ij}^2} = \frac{\gamma^{\frac{1}{p}}\left\{\prod_{h=1}^p \pi_h\right\}^{\frac{1}{p}}}{\pi_j}. \tag{25}$$

where

$$\pi_h = \sum_{e_k\in P_i}\left\{-\frac{2\sum_{e_l\in P_i}\mathcal{K}^{(\mathbf{s}_i)}(\mathbf{x}_k,\mathbf{x}_l)(x_{kh}-x_{lh})^2}{|P_i|} + \frac{\sum_{e_r\in P_i}\sum_{e_s\in P_i}\mathcal{K}^{(\mathbf{s}_i)}(\mathbf{x}_r,\mathbf{x}_s)(x_{rh}-x_{sh})^2}{|P_i|^2}\right\} \quad (1 \le h \le p)$$

Proof. Concerning the KCM-F-GH algorithm, we first use the method of Lagrange multipliers with the restriction that $\prod_{j=1}^p(\frac{1}{s_j^2}) = \gamma$, where γ is a suitable parameter, and obtain

$$L_{KCM-F-GH} = J_{KCM-F-GH} - \omega\left(\prod_{j=1}^p\frac{1}{s_j^2} - \gamma\right)$$
$$= \sum_{i=1}^c\sum_{e_k\in P_i}\left\{1 - \frac{2\sum_{e_l\in P_i}\mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k,\mathbf{x}_l)}{|P_i|} + \frac{\sum_{e_r\in P_i}\sum_{e_s\in P_i}\mathcal{K}^{(\mathbf{s})}(\mathbf{x}_r,\mathbf{x}_s)}{|P_i|^2}\right\} - \omega\left(\prod_{j=1}^p\frac{1}{s_j^2} - \gamma\right).$$

Then, we compute the partial derivatives of $L_{KCM-F-GH}$ w.r.t ω and $\frac{1}{s_j^2}$ $(j = 1,\ldots,p)$, and, by setting the partial derivatives to zero, and after some algebra we obtain:

$$\frac{1}{s_j^2} = \frac{\gamma^{\frac{1}{p}}\left\{\prod_{h=1}^p \pi_h\right\}^{\frac{1}{p}}}{\pi_j}.$$

where

$$\pi_h = \sum_{i=1}^c\sum_{e_k\in P_i}\left\{-\frac{2\sum_{e_l\in P_i}\mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k,\mathbf{x}_l)(x_{kh}-x_{lh})^2}{|P_i|} + \frac{\sum_{e_r\in P_i}\sum_{e_s\in P_i}\mathcal{K}^{(\mathbf{s})}(\mathbf{x}_r,\mathbf{x}_s)(x_{rh}-x_{sh})^2}{|P_i|^2}\right\} \quad (h = 1,\ldots,p)$$

Concerning the KCM-F-LH algorithm, we first use the method of Lagrange multipliers with the restriction that $\prod_{j=1}^p(\frac{1}{s_{ij}^2}) = \gamma$, where γ is a suitable parameter, and obtain

$$L_{KCM-F-LH} = J_{KCM-F-LH} - \sum_{i=1}^c\omega_i\left(\prod_{j=1}^p\frac{1}{s_{ij}^2} - \gamma\right)$$
$$= \sum_{i=1}^c\sum_{e_k\in P_i}\left\{1 - \frac{2\sum_{e_l\in P_i}\mathcal{K}^{(\mathbf{s}_i)}(\mathbf{x}_k,\mathbf{x}_l)}{|P_i|} + \frac{\sum_{e_r\in P_i}\sum_{e_s\in P_i}\mathcal{K}^{(\mathbf{s}_i)}(\mathbf{x}_r,\mathbf{x}_s)}{|P_i|^2}\right\} - \sum_{i=1}^c\omega_i\left(\prod_{j=1}^p\frac{1}{s_{ij}^2} - \gamma\right).$$

Then, we compute the partial derivatives of $L_{KCM-K-LH}$ w.r.t ω_i and $\frac{1}{s_{ij}^2}$ $(i = 1,\ldots,c; j = 1,\ldots,p)$, and, by setting the partial derivatives to zero, and after some algebra we obtain:

$$\frac{1}{s_{ij}^2} = \frac{\gamma^{\frac{1}{p}}\left\{\prod_{h=1}^p \pi_h\right\}^{\frac{1}{p}}}{\pi_j}.$$

where

$$\pi_h = \sum_{e_k\in P_i}\left\{-\frac{2\sum_{e_l\in P_i}\mathcal{K}^{(\mathbf{s}_i)}(\mathbf{x}_k,\mathbf{x}_l)(x_{kh}-x_{lh})^2}{|P_i|} + \frac{\sum_{e_r\in P_i}\sum_{e_s\in P_i}\mathcal{K}^{(\mathbf{s}_i)}(\mathbf{x}_r,\mathbf{x}_s)(x_{rh}-x_{sh})^2}{|P_i|^2}\right\} \quad (h = 1,\ldots,p)$$

□

3.2.1.3. Allocation step. This step provides the optimal solution for the cluster partition.

During the allocation step of the KCM-F-GH and KCM-F-LH algorithms, the distances $||\Phi(\mathbf{x}_k) - \mathbf{g}_i^\Phi||^2$ (Eq. (21)) and $||\Phi(\mathbf{x}_k) - \mathbf{g}_i^\Phi||^2$ (Eq. (22)) between $\Phi(\mathbf{x}_k)$ and $\mathbf{g}_i^\Phi$ $(1 \le k \le n; 1 \le i \le c)$, the global vector of width hyper-parameters $\mathbf{s}$, and the local vectors of width hyper-parameters $\mathbf{s}_1,\ldots,\mathbf{s}_c$, are kept fixed.

Proposition 6.

(i) *The objective function of the KCM-F-GH algorithm is minimized with respect to the partition* $\mathcal{P}$ *when the clusters* P_i $(1 \le i \le c)$ *are*

updated according to the following allocation rule:

$$P_i = \{e_k \in E : ||\Phi(\mathbf{x}_k) - \mathbf{g}_i^{\Phi}||^2 = \min_{h=1}^{c} ||\Phi(\mathbf{x}_k) - \mathbf{g}_h^{\Phi}||^2\} \quad (26)$$

where $||\Phi(\mathbf{x}_k) - \mathbf{g}_h^{\Phi}||^2$ $(1 \le h \le c)$ *is computed according to Eq. (21);*

(ii) *The objective function of the KCM-F-LH algorithm is minimized with respect to the partition* $\mathcal{P}$ *when the clusters* $P_i\,(1 \le i \le c)$ *are updated according to the following allocation rule:*

$$P_i = \{e_k \in E : ||\Phi(\mathbf{x}_k) - \mathbf{g}_i^{\Phi}||^2 = \min_{h=1}^{c} ||\Phi(\mathbf{x}_k) - \mathbf{g}_h^{\Phi}||^2\} \quad (27)$$

where $||\Phi(\mathbf{x}_k) - \mathbf{g}_h^{\Phi}||^2$ $(1 \le h \le c)$ *is computed according to Eq. (22).*

Proof. The proof proceeds similarly to the proof of Proposition 3. □

3.2.1.4. KCM-F-GH and KCM-F-LH algorithms. These three steps are repeated until there is convergence of the KCM-F-GH and KCM-F-LH clustering algorithms. Algorithm 2 summarizes these steps.

3.3. Properties of the KCM-K-GH, KCM-K-LH, KCM-F-GH and KCM-F-LH algorithms

This section discusses the convergence properties and the time complexity of the KCM-K-GH, KCM-K-LH, KCM-F-GH and KCM-F-LH algorithms.

3.3.1. Convergence properties

KCM-K-GH and KCM-K-LH algorithms provide a partition $\mathcal{P}^* = \{P_1^*, \ldots, P_c^*\}$ of E into c non-empty clusters, a vector of representatives $\mathbf{G}^* = (\mathbf{g}_1^*, \ldots, \mathbf{g}_c^*)$ of the clusters in the partition $\mathcal{P}^*$ and, respectively, a vector of width hyper-parameters $\mathbf{s}^* = ((s_1^2)^*, \ldots, (s_p^2)^*)$ and a vector of c vectors of width hyper-parameters $\mathbf{S}^* = (\mathbf{s}_1^*, \ldots, \mathbf{s}_c^*)$ with $\mathbf{s}_i^* = ((s_{i1}^2)^*, \ldots, (s_{ip}^2)^*)$, $1 \le i \le c$, such that

- $J_{KCM-K-GH}(\mathbf{G}^*, \mathbf{s}^*, \mathcal{P}^*) = \min\{J_{KCM-K-GH}(\mathbf{G}, \mathbf{s}, \mathcal{P}) : \mathbf{G} \in \mathbb{L}^c, \mathbf{s} \in \mathbf{\Lambda}, \mathcal{P} \in \mathbb{P}_c\}$;
- $J_{KCM-K-LH}(\mathbf{G}^*, \mathbf{S}^*, \mathcal{P}^*) = \min\{J_{KCM-K-LH}(\mathbf{G}, \mathbf{s}_i, \mathcal{P}) : \mathbf{G} \in \mathbb{L}^c, \mathbf{S} \in \mathbf{\Lambda}^c, \mathcal{P} \in \mathbb{P}_c\}$;

where

- $\mathbb{L}$ is the representation space of the prototypes such that $\mathbf{g}_i \in \mathbb{L}$ and $\mathbf{G} \in \mathbb{L}^c$. In this paper, $\mathbb{L} = \mathbb{R}^p$.
- $\mathbf{\Lambda}$ is the space of width hyper-parameters such that $\mathbf{s} \in \mathbf{\Lambda}$ and $\mathbf{S} \in \mathbf{\Lambda}^c$. In this paper, $\mathbf{\Lambda} = \{\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_p) \in \mathbb{R}^p : \lambda_j > 0 \text{ and } \prod_{j=1}^{p} \lambda_j = 1\}$;
- $\mathbb{P}_c$ is the set of all partitions of E in c non-empty clusters such that $P_i \in (\mathfrak{p}(E) - \emptyset)$ ($\mathfrak{p}(E)$ is the power set of E) and $\mathcal{P} \in \mathbb{P}_c$;

KCM-F-GH and KCM-F-LH algorithms provide a partition $\mathcal{P}^* = \{P_1^*, \ldots, P_c^*\}$ of E into c non-empty clusters, a matrix $\mathbf{D}^* = [(||\Phi(\mathbf{x}_k) - \mathbf{g}_i^{\Phi}||^2)^*]_{\substack{1 \le k \le n \\ 1 \le i \le c}}$ of optimal distances between $\Phi(\mathbf{x}_k)$ and $\mathbf{g}_i^{\Phi}$ $(k = 1, \ldots, n; i = 1, \ldots, c)$ and, respectively, a vector of width hyper-parameters $\mathbf{s}^* = ((s_1^2)^*, \ldots, (s_p^2)^*)$ and a vector of c vectors of width hyper-parameters $\mathbf{S}^* = (\mathbf{s}_1^*, \ldots, \mathbf{s}_c^*)$ with $\mathbf{s}_i^* = ((s_{i1}^2)^*, \ldots, (s_{ip}^2)^*)$, $1 \le i \le c$, such that

- $J_{KCM-F-GH}(\mathbf{D}^*, \mathbf{s}^*, \mathcal{P}^*) = \min\{J_{KCM-F-GH}(\mathbf{D}, \mathbf{s}, \mathcal{P}) : \mathbf{D} \in (\mathbb{R}^+)^{n \times c}, \mathbf{s} \in \mathbf{\Lambda}, \mathcal{P} \in \mathbb{P}_c\}$;
- $J_{KCM-F-LH}(\mathbf{D}^*, \mathbf{S}^*, \mathcal{P}^*) = \min\{J_{KCM-F-LH}(\mathbf{D}, \mathbf{s}_i, \mathcal{P}) : \mathbf{D} \in (\mathbb{R}^+)^{n \times c}, \mathbf{S} \in \mathbf{\Lambda}^c, \mathcal{P} \in \mathbb{P}_c\}$;

Algorithm 2 KCM-F-GH and KCM-F-LH algorithms.

Input
  $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ (the data set); c (the number of clusters); $\gamma > 0$ (a suitable parameter)
Initialization
  Randomly select c distinct prototypes such that $g_i^{\Phi} = \Phi(\mathbf{x}_l)$, where $\mathbf{x}_l \in \mathcal{D}$ $(1 \le i \le c)$;
  KCM-K-GH: set $\frac{1}{s_j^2} = (\gamma)^{\frac{1}{p}}$ $(1 \le j \le p)$ or randomly set $\frac{1}{s_j^2}$ such that $\frac{1}{s_j^2} > 0$ and $\prod_{j=1}^{p} \frac{1}{s_j^2} = \gamma$;
  KCM-K-LH: set $\frac{1}{s_{ij}^2} = (\gamma)^{\frac{1}{p}}$ $(1 \le i \le c; 1 \le j \le p)$ or randomly set $\frac{1}{s_{ij}^2}$ such that $\frac{1}{s_{ij}^2} > 0$ and $\prod_{j=1}^{p} \frac{1}{s_{ij}^2} = \gamma$;
  KCM-F-GH: assign the object e_k $(1 \le k \le n)$ to the cluster P_i $(1 \le i \le c)$ if $||\Phi(\mathbf{x}_k) - \mathbf{g}_i^{\Phi}||^2 = \min_{h=1}^{c} ||\Phi(\mathbf{x}_k) - \mathbf{g}_h^{\Phi}||^2$ ($||\Phi(\mathbf{x}_k) - \mathbf{g}_h^{\Phi}||^2$ is provided by equation (21);
  KCM-F-LH: assign the object e_k $(1 \le k \le n)$ to the cluster P_i $(1 \le i \le c)$ if $||\Phi(\mathbf{x}_k) - \mathbf{g}_i^{\Phi}||^2 = \min_{h=1}^{c} ||\Phi(\mathbf{x}_k) - \mathbf{g}_h^{\Phi}||^2$ ($||\Phi(\mathbf{x}_k) - \mathbf{g}_h^{\Phi}||^2$ is provided by equation 22);
repeat
  Sep 1: representation.
    KCM-F-GH: compute the distances $||\Phi(\mathbf{x}_k) - \mathbf{g}_i^{\Phi}||^2$ between $\Phi(\mathbf{x}_k)$ and $\mathbf{g}_i^{\Phi}$ $(1 \le k \le n; 1 \le i \le c)$ using equation (21);
    KCM-F-LH: compute the distances $||\Phi(\mathbf{x}_k) - \mathbf{g}_i^{\Phi}||^2$ between $\Phi(\mathbf{x}_k)$ and $\mathbf{g}_i^{\Phi}$ $(1 \le k \le n; 1 \le i \le c)$ using equation (22);
  Step 2: computation of the width hyper-parameters
    KMC-F-GH: compute the global vector of width hyper-parameters $\mathbf{s}$ using equation (24);
    KCM-F-LH: compute the local vectors of width hyper-parameters $\mathbf{s}_1, \ldots, \mathbf{s}_c$ using equation (25);
  Step 3: allocation
  $test \leftarrow 0$;
  for $k = 1$ to n **do**
    KCM-F-GH: compute the winning cluster P_l such that $||\Phi(\mathbf{x}_k) - \mathbf{g}_l^{\Phi}||^2 = \min_{h=1}^{c} ||\Phi(\mathbf{x}_k) - \mathbf{g}_h^{\Phi}||^2$, where $||\Phi(\mathbf{x}_k) - \mathbf{g}_h^{\Phi}||^2$ is provided by equation (21);
    KCM-F-LH: compute the winning cluster P_l such that $||\Phi(\mathbf{x}_k) - \mathbf{g}_l^{\Phi}||^2 = \min_{h=1}^{c} ||\Phi(\mathbf{x}_k) - \mathbf{g}_h^{\Phi}||^2$, where $||\Phi(\mathbf{x}_k) - \mathbf{g}_h^{\Phi}||^2$ is provided by equation (22);
    if $\mathbf{x}_k \in P_i$ and $l \ne i$ **then**
      $test \leftarrow 1$; $P_l = P_l \cup \{\mathbf{x}_k\}$; $P_i = P_i \setminus \{\mathbf{x}_k\}$;
    end if
  end for
until $test = 0$
OUTPUT:
  KCM-F-GH: the global vector of hyper-parameters $\mathbf{s}$;
  KCM-F-LH: the local vectors of hyper-parameters $\mathbf{s}_1, \ldots, \mathbf{s}_c$;
  Both KCM-F-GH and KCM-F-LH : the partition $\mathcal{P}$.

where $(\mathbb{R}^+)^{n \times c}$ is the space of matrices of positive real values and $\mathbf{\Lambda}$ and $\mathbb{P}_c$ are as previously defined.

According to Ref. [16], the properties of convergence of the algorithms KCM-K-GH, KCM-K-LH, KCM-F-GH, and KCM-F-LH can be studied from the series:

- $v_{KCM-K-GH}^{(t)} = (\mathbf{G}^{(t)}, \mathbf{s}^{(t)}, \mathcal{P}^{(t)}) \in \mathbb{L}^c \times \Lambda \times \mathbb{P}_c$, $t = 0, 1, \ldots$;
- $u_{KCM-K-GH}^{(t)} = J_{KCM-K-GH}(v_{KCM-K-GH}^{(t)}) = J_{KCM-K-GH}(\mathbf{G}^{(t)}, \mathbf{s}^{(t)}, \mathcal{P}^{(t)})$, $t = 0, 1, \ldots$;
- $v_{KCM-K-LH}^{(t)} = (\mathbf{G}^{(t)}, \mathbf{S}^{(t)}, \mathcal{P}^{(t)}) \in \mathbb{L}^c \times \Lambda^c \times \mathbb{P}_c$, $t = 0, 1, \ldots$;

- $u^{(t)}_{KCM-K-LH} = J_{KCM-K-LH}(v^{(t)}_{KCM-K-LH}) = J_{KCM-K-LH}(\mathbf{G}^{(t)}, \mathbf{S}^{(t)}, \mathcal{P}^{(t)})$, $t = 0, 1, \ldots$;
- $v^{(t)}_{KCM-F-GH} = (\mathbf{D}^{(t)}, \mathbf{s}^{(t)}, \mathcal{P}^{(t)}) \in (\mathbb{R}^+)^{n\times c} \times \Lambda \times \mathbb{P}_c$, $t = 0, 1, \ldots$;
- $u^{(t)}_{KCM-F-GH} = J_{KCM-F-GH}(v^{(t)}_{KCM-F-GH}) = J_{KCM-F-GH}(\mathbf{D}^{(t)}, \mathbf{s}^{(t)}, \mathcal{P}^{(t)})$, $t = 0, 1, \ldots$;
- $v^{(t)}_{KCM-F-LH} = (\mathbf{D}^{(t)}, \mathbf{S}^{(t)}, \mathcal{P}^{(t)}) \in (\mathbb{R}^+)^{n\times c} \times \Lambda^c \times \mathbb{P}_c$, $t = 0, 1, \ldots$;
- $u^{(t)}_{KCM-F-LH} = J_{KCM-F-LH}(v^{(t)}_{KCM-F-LH}) = J_{KCM-F-LH}(\mathbf{D}^{(t)}, \mathbf{S}^{(t)}, \mathcal{P}^{(t)})$, $t = 0, 1, \ldots$.

From the initial terms $v^{(0)}_{KCM-K-GH} = (\mathbf{G}^{(0)}, \mathbf{s}^{(0)}, \mathcal{P}^{(0)})$, $v^{(0)}_{KCM-K-LH} = (\mathbf{G}^{(0)}, \mathbf{S}^{(0)}, \mathcal{P}^{(0)})$, $v^{(0)}_{KCM-F-GH} = (\mathbf{D}^{(0)}, \mathbf{s}^{(0)}, \mathcal{P}^{(0)})$ and $v^{(0)}_{KCM-F-LH} = (\mathbf{D}^{(0)}, \mathbf{S}^{(0)}, \mathcal{P}^{(0)})$, the algorithms KCM-K-GH, KCM-K-LH, KCM-F-GH and KCM-F-LH compute the different terms of the series respectively, $v^{(t)}_{KCM-K-GH}$, $v^{(t)}_{KCM-K-LH}$, $v^{(t)}_{KCM-F-GH}$, $v^{(t)}_{KCM-F-LH}$, until the respective convergence (to be shown) when the objective functions $J_{KCM-K-GH}$, $J_{KCM-K-LH}$, $J_{KCM-F-GH}$ and $J_{KCM-F-LH}$ reach stationary values.

Proposition 7.

(i) *The series* $u^{(t)}_{KCM-K-GH} = J_{KCM-K-GH}(v^{(t)}_{KCM-K-GH}) = J_{KCM-K-GH}(\mathbf{G}^{(t)}, \mathbf{s}^{(t)}, \mathcal{P}^{(t)})$, $t = 0, 1, \ldots$, *decreases at each iteration and converges;*

(ii) *The series* $u^{(t)}_{KCM-K-LH} = J_{KCM-K-LH}(v^{(t)}_{KCM-K-LH}) = J_{KCM-K-LH}(\mathbf{G}^{(t)}, \mathbf{S}^{(t)}, \mathcal{P}^{(t)})$, $t = 0, 1, \ldots$, *decreases at each iteration and converges.*

(iii) *The series* $u^{(t)}_{KCM-F-GH} = J_{KCM-F-GH}(v^{(t)}_{KCM-F-GH}) = J_{KCM-F-GH}(\mathbf{D}^{(t)}, \mathbf{s}^{(t)}, \mathcal{P}^{(t)})$, $t = 0, 1, \ldots$, *decreases at each iteration and converges;*

(iv) *The series* $u^{(t)}_{KCM-F-LH} = J_{KCM-F-LH}(v^{(t)}_{KCM-F-LH}) = J_{KCM-F-LH}(\mathbf{D}^{(t)}, \mathbf{S}^{(t)}, \mathcal{P}^{(t)})$, $t = 0, 1, \ldots$, *decreases at each iteration and converges.*

Proof.

(i) The series $u^{(t)}_{KCM-K-GH} = J_{KCM-K-GH}(v^{(t)}_{KCM-K-GH}) = J_{KCM-K-GH}(\mathbf{G}^{(t)}, \mathbf{s}^{(t)}, \mathcal{P}^{(t)})$, $t = 0, 1, \ldots$ decreases at each iteration and converges.

The objective function $J_{KCM-K-GH}$ measures the heterogeneity of the partition as the sum of the heterogeneities in each cluster. We will first show that the inequalities (I), (II) and (III) below hold (i.e., the series decreases at each iteration).

$$\underbrace{J_{KCM-K-GH}(\mathbf{G}^{(t)}, \mathbf{s}^{(t)}, \mathcal{P}^{(t)})}_{u^{(t)}_{KCM-K-GH}} \overbrace{\geq}^{(I)} J_{KCM-K-GH}(\mathbf{G}^{(t+1)}, \mathbf{s}^{(t)}, \mathcal{P}^{(t)}) \overbrace{\geq}^{(II)} J_{KCM-K-GH}(\mathbf{G}^{(t+1)}, \mathbf{s}^{(t+1)}, \mathcal{P}^{(t)}) \overbrace{\geq}^{(III)} \underbrace{J_{KCM-K-GH}(\mathbf{G}^{(t+1)}, \mathbf{s}^{(t+1)}, \mathcal{P}^{(t+1)})}_{u^{(t+1)}_{KCM-K-GH}}$$

The inequality (I) holds because $J_{KCM-K-GH}(\mathbf{G}^{(t)}, \mathbf{s}^{(t)}, \mathcal{P}^{(t)}) = \sum_{i=1}^{c}\sum_{e_k\in P_i^{(t)}} 2(1 - \mathcal{K}^{(\mathbf{s}^{(t)})}(\mathbf{x}_k, \mathbf{g}_i^{(t)}))$, and $J_{KCM-K-GH}(\mathbf{G}^{(t+1)}, \mathbf{s}^{(t)}, \mathcal{P}^{(t)}) = \sum_{i=1}^{c}\sum_{e_k\in P_i^{(t)}} 2(1 - \mathcal{K}^{(\mathbf{s}^{(t)})}(\mathbf{x}_k, \mathbf{g}_i^{(t+1)}))$, and according to Proposition 15,

$$\mathbf{G}^{(t+1)} = (\mathbf{g}_1^{(t+1)}, \ldots, \mathbf{g}_c^{(t+1)}) = \underset{\mathbf{G}=(\mathbf{g}_1\ldots\mathbf{g}_c)\in\mathbb{L}^c}{\operatorname{argmin}} \sum_{i=1}^{c}\sum_{e_k\in P_i^{(t)}} 2(1 - \mathcal{K}^{(\mathbf{s}^{(t)})}(\mathbf{x}_k, \mathbf{g}_i)).$$

Moreover, inequality (II) holds because $J_{KCM-K-GH}(\mathbf{G}^{(t+1)}, \mathbf{s}^{(t+1)}, \mathcal{P}^{(t)}) = \sum_{i=1}^{c}\sum_{e_k\in P_i^{(t)}} 2(1 - \mathcal{K}^{(\mathbf{s}^{(t+1)})}(\mathbf{x}_k, \mathbf{g}_i^{(t+1)}))$, and according to Proposition 2,

$$\mathbf{s}^{(t+1)} = ((s_1^2)^{(t+1)}, \ldots, (s_p^2)^{(t+1)}) = \underset{\mathbf{s}=(s_1^2,\ldots,s_p^2)\in\Lambda}{\operatorname{argmin}} \sum_{i=1}^{c}\sum_{e_k\in P_i^{(t)}} 2(1 - \mathcal{K}^{(\mathbf{s})}(\mathbf{x}_k, \mathbf{g}_i^{(t+1)})).$$

The inequality (III) also holds because $J_{KCM-K-GH}(\mathbf{G}^{(t+1)}, \mathbf{s}^{(t+1)}, \mathcal{P}^{(t+1)}) = \sum_{i=1}^{c}\sum_{e_k\in P_i^{(t+1)}} 2(1 - \mathcal{K}^{(\mathbf{s}^{(t+1)})}(\mathbf{x}_k, \mathbf{g}_i^{(t+1)}))$, and according to Proposition 3,

$$\mathcal{P}^{(t+1)} = (P_1^{(t+1)}, \ldots, P_c^{(t+1)}) = \underset{\mathcal{P}=(P_1,\ldots,P_c)\in\mathbb{P}_c}{\operatorname{argmin}} \sum_{i=1}^{c}\sum_{e_k\in P_i} 2(1 - \mathcal{K}^{(\mathbf{s}^{(t+1)})}(\mathbf{x}_k, \mathbf{g}_i^{(t+1)})).$$

Finally, because the series $u^{(t)}_{KCM-K-GH} = J_{KCM-K-GH}(v^{(t)}_{KCM-K-GH})$ decreases and it is bounded ($J(v^{(t)}_{KCM-K-GH}) \geq 0$), it converges.

The proof of the convergence of the series $u^{(t)}_{KCM-K-LH}$, $t = 0, 1, \ldots$, $u^{(t)}_{KCM-F-GH}$, $t = 0, 1, \ldots$ and $u^{(t)}_{KCM-F-LH}$, $t = 0, 1, \ldots$, proceeds similarly to the proof of the convergence of the series $u^{(t)}_{KCM-K-GH}$, $t = 0, 1, \ldots$, presented above. □

Proposition 8.

(i) *The series* $v^{(t)}_{KCM-K-GH} = (\mathbf{G}^{(t)}, \mathbf{s}^{(t)}, \mathcal{P}^{(t)})$, $t = 0, 1, \ldots$, *converges;*

(ii) *The series* $v^{(t)}_{KCM-K-LH} = (\mathbf{G}^{(t)}, \mathbf{S}^{(t)}, \mathcal{P}^{(t)})$, $t = 0, 1, \ldots$, *converges.*

(iii) *The series* $v^{(t)}_{KCM-F-GH} = (\mathbf{D}^{(t)}, \mathbf{s}^{(t)}, \mathcal{P}^{(t)})$, $t = 0, 1, \ldots$, *converges;*

(iv) *The series* $v^{(t)}_{KCM-F-LH} = (\mathbf{D}^{(t)}, \mathbf{S}^{(t)}, \mathcal{P}^{(t)})$, $t = 0, 1, \ldots$, *converges.*

Proof.

(i) The series $v^{(t)}_{KCM-K-GH} = (\mathbf{G}^{(t)}, \mathbf{s}^{(t)}, \mathcal{P}^{(t)})$, $t = 0, 1, \ldots$ converges.

Assuming that the stationarity of the series $u^{(t)}_{KCM-K-GH}$ is achieved in the iteration t=T, then, we have $u^{(T)}_{KCM-K-GH} = u^{(T+1)}_{KCM-K-GH}$ and then $J_{KCM-K-GH}(v^{(T)}_{KCM-K-GH}) = J_{KCM-K-GH}(v^{(T+1)}_{KCM-K-GH})$.

From $J_{KCM-K-GH}(v^{(T)}_{KCM-K-GH}) = J_{KCM-K-GH}(v^{(T+1)}_{KCM-K-GH}))$, we arrive at $J_{KCM-K-GH}(\mathbf{G}^{(T)}, \mathbf{s}^{(T)}, \mathcal{P}^{(T)}) = J_{KCM-K-GH}(\mathbf{G}^{(T+1)}, \mathbf{s}^{(T+1)}, \mathcal{P}^{(T+1)})$. This equality, according to Proposition 7, can be rewritten as the equalities (I)-(III):

$$\underbrace{J_{KCM-K-GH}(\mathbf{G}^{(T)}, \mathbf{s}^{(T)}, \mathcal{P}^{(T)})}_{u^{(T)}_{KCM-K-GH}} \overbrace{=}^{(I)} J_{KCM-K-GH}(\mathbf{G}^{(T+1)}, \mathbf{s}^{(T)}, \mathcal{P}^{(T)}) \overbrace{=}^{(II)} J_{KCM-K-GH}(\mathbf{G}^{(T+1)}, \mathbf{s}^{(T+1)}, \mathcal{P}^{(T)}) \overbrace{=}^{(III)} \underbrace{J_{KCM-K-GH}(\mathbf{G}^{(T+1)}, \mathbf{s}^{(T+1)}, \mathcal{P}^{(T+1)})}_{u^{(T+1)}_{KCM-K-GH}}$$

From the first equality (I), we have that $\mathbf{G}^{(T)} = \mathbf{G}^{(T+1)}$, because $\mathbf{G}$ is unique, minimizing $J_{KCM-K-GH}$, when the partition represented by $\mathcal{P}^{(T)}$ and the vector of width hyper-parameters $\mathbf{s}^{(T)}$ are kept fixed. From the second equality (II), we have that $\mathbf{s}^{(T)} = \mathbf{s}^{(T+1)}$ because $\mathbf{s}$ is unique, minimizing $J_{KCM-K-GH}$, when the partition represented

Table 1
Time complexity of the clustering algorithms.

Algorithms					
KCM-K	KCM-K-GH	KCM-K-LH	KCM-F	KCM-F-GH	KCM-F-LH
$\mathcal{O}(cnp)$	$\mathcal{O}(cnp)$	$\mathcal{O}(cnp)$	$\mathcal{O}(n^2p)$	$\mathcal{O}(n^2p)$	$\mathcal{O}(n^2p)$

by $\mathcal{P}^{(T)}$ and the matrix of prototypes $\mathbf{G}^{(T+1)}$ are kept fixed. Furthermore, from the third equality (III), we have that $\mathcal{P}^{(T)} = \mathcal{P}^{(T+1)}$ because $\mathcal{P}$ is unique, minimizing $J_{KCM-K-GH}$, when the matrix of prototypes $\mathbf{G}^{(T+1)}$ and the vector of width hyper-parameters $\mathbf{s}^{(T+1)}$ are kept fixed.

Therefore, it can be concluded that $v^{(T)}_{KCM-K-GH} = v^{(T+1)}_{KCM-K-GH}$. This conclusion stands for all $t \geq T$ and $v^{(t)}_{KCM-K-GH} = v^{(T)}_{KCM-K-GH}, \forall t \geq T$ and it follows that the series $v^{(t)}_{KCM-K-GH}$ converges.

The proof of the convergence of the series $v^{(t)}_{KCM-K-LH}$, $t = 0, 1, \ldots,$ $v^{(t)}_{KCM-F-GH}, t = 0, 1, \ldots,$ and $v^{(t)}_{KCM-F-LH}$, $t = 0, 1, \ldots$ proceeds similarly to the proof of the convergence of the series $v^{(t)}_{KCM-K-GH}$, $t = 0, 1, \ldots,$ presented above. □

3.4. Complexity analysis

Table 1 provides the time complexity of KCM-K, KCM-K-GH, KCM-K-LH, KCM-F, KCM-F-GH and KCM-F-LH clustering algorithms at each iteration, where c is the number of clusters, n is the number of objects, and p is the number of variables.

The time complexity of KCM-K and KCM-F clustering algorithms for a single iteration is, respectively $\mathcal{O}(cnp)$ and $\mathcal{O}(n^2p)$. For the KCM-F algorithm, it is assumed that the kernel matrix is computed only once at the initialization of the algorithm.

For KCM-K-GH and KCM-K-LH, the kernel matrix between patterns and prototypes is computed at the beginning of each iteration with a cost of $\mathcal{O}(cnp)$. Each prototype is computed with a cost of $\mathcal{O}(np)$, since the kernel matrix is already computed. The cost of the width hyper-parameters computation is $\mathcal{O}(n+p)$ for the KCM-K-GH variant, and $\mathcal{O}(n+cp)$ for the KCM-K-LH variant. Finally, every object is then allocated to the closest prototype with a cost of $\mathcal{O}(cn)$. Thus the computational complexity of the KCM-K-H variants is $\mathcal{O}(cnp)$ for a single iteration.

For KCM-F-GH and KCM-F-LH, the kernel matrix is computed at the beginning of each iteration with a cost of $\mathcal{O}(n^2p)$. The KCM-F variants lack the step in which cluster prototypes are updated but the distances between objects and cluster prototypes are computed with a cost of $\mathcal{O}(cnp)$, since the kernel matrix is already computed. The cost of the width hyper-parameters computation is $\mathcal{O}(n^2p)$. Finally, every object is then allocated to the closest prototype with a cost of $\mathcal{O}(cn)$. Thus the computational complexity of the KCM-F-H variants is $\mathcal{O}(n^2p)$ for a single iteration.

4. Empirical results

This section evaluates the performance of the proposed methods, in comparison to the conventional KCM-K and KCM-F algorithms, through applications with synthetic data sets as well as benchmark data sets selected from the UCI Machine Learning Repository.

4.1. Clustering assessment

To compare the quality of the partitions provided by these algorithms, the Error of classification (ER) [6], the external adjusted Rand index (ARI) [29], and the F-measure [43] were computed for the best results selected according to the clustering objective function.

Let $P = \{P_1, \ldots, P_i, \ldots, P_m\}$ be the *a priori* partition into m classes and $Q = \{Q_1, \ldots, Q_j, \ldots, Q_K\}$ be the partition into K clusters provided by a clustering algorithm. Table 2 provides the confusion matrix:

The adjusted Rand index is:

$$ARI = \frac{\sum_{i=1}^{m}\sum_{j=1}^{K}\binom{n_{ij}}{2} - \binom{n}{2}^{-1}\sum_{i=1}^{m}\binom{n_{i\bullet}}{2}\sum_{j=1}^{K}\binom{n_{\bullet j}}{2}}{\frac{1}{2}\left[\sum_{i=1}^{m}\binom{n_{i\bullet}}{2} + \sum_{j=1}^{K}\binom{n_{\bullet j}}{2}\right] - \binom{n}{2}^{-1}\sum_{i=1}^{m}\binom{n_{i\bullet}}{2}\sum_{j=1}^{K}\binom{n_{\bullet j}}{2}} \quad (28)$$

where $\binom{n}{2} = \frac{n(n-1)}{2}$ and n_{ij} represents the number of objects that are in class P_i and cluster Q_j; $n_{i\bullet}$ indicates the number of objects in class P_i; $n_{\bullet j}$ indicates the number of objects in cluster Q_j; and n is the total number of objects in the data set.

The ARI index assesses the degree of agreement (similarity) between an a priori partition and a partition provided by the clustering algorithm. Moreover, the ARI index is not sensitive to the number of classes in the partitions or the distribution of the items in the clusters. Finally, the ARI index takes its values from the interval [-1,1], in which the value 1 indicates perfect agreement between partitions, whereas values near 0 (or negatives) correspond to cluster agreement found by chance [45].

The traditional $F-measure$ between class P_i $(i = 1, \ldots, m)$ and cluster Q_j $(j = 1, \ldots, K)$ is the harmonic mean of precision and recall:

$$F-measure(P_i, Q_j) = 2\frac{Precision(P_i, Q_j) \times Recall(P_i, Q_j)}{Precision(P_i, Q_j) + Recall(P_i, Q_j)} \quad (29)$$

The *Precision* between class P_i $(i = 1, \ldots, m)$ and cluster Q_j $(j = 1, \ldots, K)$ is defined as the ratio between the number of objects that are in class P_i and cluster Q_j and the number of objects in cluster Q_j:

$$Precision(P_i, Q_j) = \frac{n_{ij}}{n_{\bullet j}} = \frac{n_{ij}}{\sum_{i=1}^{m} n_{ij}} \quad (30)$$

The *Recall* between class P_i $(i = 1, \ldots, m)$ and cluster Q_j $(j = 1, \ldots, K)$ is defined as the ratio between the number of objects that are in class P_i and cluster Q_j and the number of objects in class P_i:

$$Recall(P_i, Q_j) = \frac{n_{ij}}{n_{i\bullet}} = \frac{n_{ij}}{\sum_{j=1}^{K} n_{ij}} \quad (31)$$

The $F-measure$ between the *a priori* partition $P = \{P_1, \ldots, P_i, \ldots, P_m\}$ and the hard partition $Q = \{Q_1, \ldots, Q_j, \ldots, Q_K\}$ given by a cluster algorithm is defined as:

$$F-measure(P, Q) = \frac{1}{n}\sum_{i=1}^{m} n_{i\bullet} max_{1 \leq j \leq K} F-measure(P_i, Q_j) \quad (32)$$

The $F-measure$ index takes its values from the interval [0,1], in which value 1 indicates perfect agreement between partitions.

Finally, the ER index aims to measure the ability of a clustering algorithm to find out the a priori classes present in a data set and takes its values in the range [0, 1], in which lower ER values indicate better clustering results. It is computed as:

$$OERC = \sum_{j=1}^{K}\frac{n_{\bullet j}}{n}\left(1 - \max_{1 \leq i \leq m} n_{ij}/n_{\bullet j}\right) = 1 - \frac{\sum_{j=1}^{K}\max_{1 \leq i \leq m} n_{ij}}{n} \quad (33)$$

4.2. Initialization

The conventional, as well as the kernel c-means algorithms proposed in this paper, start from an initial solution and iteratively alternate two or three steps until the convergence on a final solution, when the objective function that measures the intra-cluster heterogeneity reaches a local minimum. It is well-known that the quality of the final solution provided by this kind of algorithm

Table 2
Confusion matrix.

Classes	Clusters					
	Q_1	...	Q_j	...	Q_K	Σ
P_1	n_{11}	...	n_{1j}	...	n_{1K}	$n_{1\bullet} = \sum_{j=1}^{K} n_{1j}$
⋮	⋮	...	⋮	...	⋮	⋮
P_i	n_{i1}	...	n_{ij}	...	n_{iK}	$n_{i\bullet} = \sum_{j=1}^{K} n_{ij}$
⋮	⋮	...	⋮	...	⋮	
P_m	n_{m1}	...	n_{mj}	...	n_{mK}	$n_{m\bullet} = \sum_{j=1}^{K} n_{mj}$
Σ	$n_{\bullet 1} = \sum_{i=1}^{m} n_{i1}$	...	$n_{\bullet j} = \sum_{i=1}^{m} n_{ij}$	...	$n_{\bullet K} = \sum_{i=1}^{m} n_{iK}$	$n = \sum_{i=1}^{m}\sum_{j=1}^{K} n_{ij}$

(k-means like algorithms) crucially depends on the initialization [3,48]

In this paper, conventional KCM-K and KCM-F algorithms start from an initial solution where the representatives are initialized by randomly choosing c objects of the data set and assigning the rest of the objects to the cluster represented by the nearest representative (Forgy approach [23]) to produce the initial partition.

KCM-K-GH, KCM-K-LH, KCM-F-GH and KCM-F-LH algorithms start from an initial solution where, first, the representatives are initialized by randomly choosing c objects of the data set and then, the width hyper-parameters are also initialized as described hereafter. Finally, using the initialization of the cluster representatives and the initialization of the width hyper-parameters, the rest of the objects are assigned to the cluster represented by the nearest representative to produce the initial partition.

The performances of the conventional KCM-K and KCM-F clustering algorithms are heavily dependent on the tuning of the width hyper-parameter σ^2 of the Gaussian kernel function Eq. (3). In this paper, the recommendation provided by Ref. [8] was adopted, which, based on experimental evidence, proposes the average of the 0.1 and 0.9 quantiles of $||\mathbf{x}_l - \mathbf{x}_k||^2$, $l \neq k$ as a suitable value for this parameter. The quantiles of $||\mathbf{x}_l - \mathbf{x}_k||^2$, $l \neq k$ have been previously computed and only once, as a preliminary step, before running these algorithms.

Moreover, based on experimentations, we observed that the performance of the KCM-K-GH and KCM-F-GH algorithms as well as the performance of the KCM-K-LH and KCM-F-LH algorithms are, respectively, dependent on the *initial value* of the width hyper-parameters s_j and s_{ij} $(1 \leq j \leq p;\ 1 \leq i \leq c)$.

We observed that if, following Ref. [8], the initial values for the width hyper-parameters s_j and s_{ij} are around the average of the 0.1 and 0.9 quantiles of $||\mathbf{x}_l - \mathbf{x}_k||^2$, $l \neq k$, often these algorithms performed better than the conventional KCM-K and KCM-F algorithms. Therefore, in this paper, we set initial values for the width hyper-parameters s_j and s_{ij} according two different options. We observed that the results in the terms of the quality of the partition provided by KCM-K-GH, KCM-K-LH, KCM-F-GH and KCM-F-LH algorithms were similar with both initialization options.

The first option, henceforth named the fixed initialization option, sets the width hyper-parameters as $s_j = \sigma$ and as $s_{ij} = \sigma$, where σ^2 is the average of the 0.1 and 0.9 quantiles of $||\mathbf{x}_l - \mathbf{x}_k||^2$, $l \neq k$. Consequently, the parameter γ of KCM-K-GH and KCM-K-LH (see Algorithm 1) and of KCM-F-GH and KCM-F-LH (see Algorithm 2) is set as $\gamma = (\frac{1}{\sigma^2})^p$.

The second option, henceforth named the random initialization option, sets the width hyper-parameters randomly but with the parameter γ also set as $\gamma = (\frac{1}{\sigma^2})^p$, as in the fixed initialization option. Thus, the majority of the initial values of these width hyper-parameters are around the average of the 0.1 and 0.9 quantiles of $||\mathbf{x}_l - \mathbf{x}_k||^2$, $l \neq k$ and the proposed algorithms of this paper often performed better than the conventional KCM-K and KCM-F algorithms.

Table 3
Configuration of the synthetic data set 1.

$\boldsymbol{\mu}$	Group 1	Group 2	Group 3	Group 4
μ_1	−0,5	0, 5	0, 0	0, 0
μ_2	−0,5	−0,5	0, 5	0, 5
μ_3	0, 0	0, 0	−0,5	0, 5
$\boldsymbol{\Sigma}$	Group 1	Group 2	Group 3	Group 4
σ_1^2	0, 04	0, 04	1, 00	1, 00
σ_2^2	0, 04	0, 04	0, 04	0, 04
σ_3^2	1, 00	1, 00	0, 04	0, 04

4.3. Synthetic data sets

To illustrate the ability of the KCM-K-GH, KCM-K-LH, KCM-F-GH, and KCM-F-LH clustering methods to automatically learn their corresponding width-parameter, and thus automatically select the relevant variables in the clustering process, we considered two configurations of three-dimensional real-valued data with four and three a priori classes. Three sample sizes were adopted for each class: 100, 250 and 500 objects. Thus, in configuration 1, we considered data sets with 400, 1000 and 2000 objects, and, in configuration 2, data sets with 300, 750 and 1500 objects.

For each synthetic data set, the ARI index, F-measure and ER index were computed in the framework of a Monte Carlo simulation with 100 replications. In each replication, the clustering algorithms were run (until convergence to a stationary value of the objective function) 100 times and the best result for each method was selected according to the objective function. The average and the standard deviation of these measures based on 100 Monte Carlo replications were computed. Moreover, the fixed initialization of the width hyper-parameters of KCM-K-GH, KCM-K-LH, KCM-F-GH, and KCM-F-LH algorithms was used with these synthetic data sets.

4.3.1. Synthetic data set 1

The synthetic data set 1 has $n \in \{400, 1000, 2000\}$ objects in three dimensions and is divided in $c = 4$ a priori classes of $n_i \in \{100, 250, 500\}$ $(i = 1, 2, 3, 4)$ objects drawn from 3-dimensional Gaussian distributions with a specific mean vector and a specific covariance matrix for each class. The mean vector and the variance structure of each class are shown in Table 3. The correlation between the variables was set as equal to zero in all classes.

It can be observed from Table 3 that the variables 1 and 2 are relevant to defining the classes 1 and 2 (they have a small variance in comparison with variable 3), whereas classes 3 and 4 are defined by the variables 2 and 3 (they have a small variance in comparison with variable 1).

Table 4
Performance of the algorithms on the synthetic data set 1: average and standard deviation (in parentheses) of the ARI index, F-measure and ER index.

n_i	Algorithm	ARI	F-measure	ER
100	KCM-K	0.2572 (0.0258)	0.5112 (0.0239)	0.4911 (0.0239)
	KCM-K-GH	0.3451 (0.0697)	0.5517 (0.0520)	0.4522 (0.0537)
	KCM-K-LH	0.9702 (0.0138)	0.9888 (0.0052)	0.0112 (0.0052)
	KCM-F	0.2761 (0.0246)	0.5210 (0.0242)	0.4816 (0.0251)
	KCM-F-GH	0.3713 (0.1085)	0.5707 (0.0774)	0.4331 (0.0787)
	KCM-F-LH	0.9705 (0.0130)	0.9889 (0.0049)	0.0111 (0.0049)
250	KCM-K	0.2537 (0.0135)	0.4937 (0.0127)	0.5078 (0.0133)
	KCM-K-GH	0.3281 (0.0040)	0.5272 (0.0115)	0.4760 (0.0119)
	KCM-K-LH	0.9697 (0.0334)	0.9875 (0.0232)	0.0125 (0.0231)
	KCM-F	0.2737 (0.0123)	0.5043 (0.0129)	0.4974 (0.0134)
	KCM-F-GH	0.3476 (0.0772)	0.5413 (0.0546)	0.4615 (0.0552)
	KCM-F-LH	0.9730 (0.0090)	0.9898 (0.0034)	0.0102 (0.0034)
500	KCM-K	0.2546 (0.0091)	0.4873 (0.0104)	0.5133 (0.0105)
	KCM-K-GH	0.3271 (0.0024)	0.5181 (0.0080)	0.4844 (0.0090)
	KCM-K-LH	0.9739 (0.0065)	0.9901 (0.0025)	0.0099 (0.0025)
	KCM-F	0.2742 (0.0081)	0.4973 (0.0101)	0.5039 (0.0104)
	KCM-F-GH	0.3272 (0.0024)	0.5185 (0.0083)	0.4839 (0.0093)
	KCM-F-LH	0.9738 (0.0064)	0.9901 (0.0024)	0.0099 (0.0024)

Table 5
Configuration of the synthetic data set 2.

$\boldsymbol{\mu}$	Class 1	Class 2	Class 3
μ_1	0, 0	0, 1	0, 9
μ_2	0, 0	0, 7	0, 1
$\boldsymbol{\Sigma}$	Class 1	Class 2	Class 3
σ_1^2	0, 04	0, 04	0, 04
σ_2^2	0, 04	0, 04	0, 04

Because there is a specific set of relevant variables to each cluster, it is expected that KCM-K-LH and KCM-F-LH algorithms have the best performance from the data set drawn according to the synthetic data set 1.

For each sample size, the algorithms were applied to the synthetic dataset 1 to obtain a 4-cluster partition. The 4-cluster partitions provided by the clustering algorithms were compared to the known a priori 4-class partition. Table 4 shows the performance of the considered algorithms on the synthetic dataset 1 according to the ARI index, F-measure, and ER index. As expected, the superiority of the KCM-K-LH and KCM-F-LH, that are Gaussian kernel c-means clustering algorithms with automated computation of a local vector of width hyper-parameter (i.e, where each variable has its own hyper-parameter in each cluster), can be observed. It can also be noticed that the sample size does not significantly affect the performance of the algorithms. For instance, the values of the ARI index, F-measure, and ER index for the KCM-F-LH algorithm were, equal to 0.9705, 0.9889 and 0.0111, respectively, when considering classes with sample sizes equal to 100, whereas these measures were equal to 0.9738, 0.9901*and*0.0099, respectively, when considering classes with sample sizes equal to 500.

4.3.2. Synthetic data set 2

The synthetic data set 2 has $n \in \{300, 750, 1500\}$ objects in three dimensions and is divided in $c = 3$ a priori classes of $n_i \in \{100, 250, 500\}$ $(i = 1, 2, 3)$ objects.

Variables 1 and 2 were drawn from bivariate Gaussian distributions with a specific mean vector and a specific covariance matrix for each class. Variable 3 was obtained as a linear combination of the variables 1 and 2 plus a noise drawn from a standard Gaussian distribution, i.e., $x_{3i} = 2x_{1i} - 1.5x_{2i} + u_i$, where $u_i \sim \mathcal{N}(0, 1), i = 1, \ldots, 300$. The mean vector and the variance structure of each class are shown in Table 5. The correlation between the variables 1 and 2 was set as equal to zero in all clusters. Variables 1 and 3 are positively correlated on each cluster, whereas variables 2 and 3 are negatively correlated on each cluster.

It can be observed from Table 5 that the variables 1 and 2 are relevant to defining the classes 1, 2 and 3 (they have a small variance in comparison with variable 3), i.e., the set of relevant variables is the same for all clusters.

Because the set of relevant variables was the same for all clusters, it is expected that KCM-K-GH and KCM-F-GH algorithms present the best performance on the data set drawn according to the synthetic data set 2.

For each sample size, the algorithms were applied to the synthetic data set 2 to obtain a 3-cluster partition. The 3-cluster partitions provided by the clustering algorithms were compared to the known a priori 3-class partition. Table 6 shows the performance of the considered algorithms on the synthetic data set 2, according to the ARI index, F-measure, and ER index. As expected, the superiority of the KCM-K-GH and KCM-F-GH, that are Gaussian kernel c-means clustering algorithms with automated computation of a global vector of width hyper-parameter (i.e, where each variable has its own hyper-parameter), can be observed. The very good performance of the KCM-K-LH and KCM-F-LH algorithms can also be observed. As was observed with the synthetic data set 1, here it is possible to notice that the sample size does not significantly affect the performance of the clustering algorithms.

Table 6
Performance of the algorithms in the synthetic data set 2: average and standard deviation (in parentheses) of the ARI index, F-measure and ER index.

n_i	Algorithm	ARI	F-measure	ER
100	KCM-K	0.2415 (0.0341)	0.6342 (0.0276)	0.3715 (0.0283)
	KCM-K-GH	0.8733 (0.0333)	0.9560 (0.0122)	0.0440 (0.0121)
	KCM-K-LH	0.8710 (0.0353)	0.9551 (0.0131)	0.0449 (0.0130)
	KCM-F	0.2446 (0.0350)	0.6354 (0.0277)	0.3699 (0.0282)
	KCM-F-GH	0.8729 (0.0332)	0.9558 (0.0121)	0.0442 (0.0121)
	KCM-F-LH	0.8709 (0.0359)	0.9550 (0.0133)	0.0449 (0.0132)
250	KCM-K	0.2438 (0.0233)	0.6368 (0.0190)	0.3698 (0.0197)
	KCM-K-GH	0.8792 (0.0225)	0.9580 (0.0082)	0.0419 (0.0082)
	KCM-K-LH	0.8778 (0.0220)	0.9575 (0.0080)	0.0425 (0.0080)
	KCM-F	0.2472 (0.0233)	0.6381 (0.0191)	0.3682 (0.0198)
	KCM-F-GH	0.8792 (0.0228)	0.9580 (0.0083)	0.0420 (0.0083)
	KCM-F-LH	0.8779 (0.0227)	0.9575 (0.0083)	0.0424 (0.0082)
500	KCM-K	0.2414 (0.0146)	0.6378 (0.0127)	0.3695 (0.0129)
	KCM-K-GH	0.8831 (0.0163)	0.9595 (0.0059)	0.0405 (0.0059)
	KCM-K-LH	0.8823 (0.0163)	0.9592 (0.0059)	0.0408 (0.0059)
	KCM-F	0.2455 (0.0155)	0.6393 (0.0130)	0.3675 (0.0132)
	KCM-F-GH	0.8835 (0.0163)	0.9596 (0.0059)	0.0404 (0.0059)
	KCM-F-LH	0.8830 (0.0159)	0.9595 (0.0058)	0.0405 (0.0058)

4.3.3. Further evaluation

To access the performance of the clustering algorithms in the case where the number of irrelevant variables is greater than the number of important variables, we carried out another Monte Carlo experiment considering again synthetic data sets 1 and 2 described in the previous section, but including, in each configuration, 10 and 100 additional nuisance variables. These additional nuisance variables were independently drawn from the standard Gaussian distribution and did not contain any information about the a priori classes. Since the sample size did not affect the clustering performance, we only considered data sets with 100 objects in each class, so that synthetic data set 1 had 400 objects whereas synthetic data set 2 had 300 objects. We considered the same experimental schema used before, i.e., 100 Monte Carlo replications, where in each replication the algorithms were run (until convergence to a stationary value of the objective function) 100 times and the best solution for each method was selected according to the value of the objective function.

Table 7 displays the performance, according to the ARI index, F-measure, and ER index, of the clustering algorithms in the syn-

Table 7
Performance of the algorithms in the synthetic data set 1 with 10 and 100 additional irrelevant variables: average and standard deviation (in parentheses) of the ARI index, F-measure, and ER index.

p	Algorithm	ARI	F-measure	ER
10	KCM-K	0.0053 (0.0061)	0.3083 (0.0148)	0.6927 (0.0150)
	KCM-K-GH	0.3340 (0.0320)	0.5444 (0.0286)	0.4596 (0.0299)
	KCM-K-LH	0.6709 (0.1236)	0.7880 (0.0843)	0.2200 (0.0882)
	KCM-F	0.0063 (0.0061)	0.3121 (0.0148)	0.6889 (0.0158)
	KCM-F-GH	0.3224 (0.0641)	0.5413 (0.0492)	0.4639 (0.0508)
	KCM-F-LH	0.5108 (0.1697)	0.6850 (0.1220)	0.3297 (0.1229)
100	KCM-K	0.0022 (0.0042)	0.3099 (0.0124)	0.6990 (0.0129)
	KCM-K-GH	0.0079 (0.0088)	0.3207 (0.0198)	0.6876 (0.0187)
	KCM-K-LH	0.0098 (0.0086)	0.3274 (0.0205)	0.6835 (0.0180)
	KCM-F	0.0018 (0.0040)	0.3121 (0.0129)	0.7002 (0.0127)
	KCM-F-GH	0.0055 (0.0071)	0.3188 (0.0174)	0.6918 (0.0173)
	KCM-F-LH	0.0052 (0.0068)	0.3186 (0.0170)	0.6921 (0.0174)

Table 8
Performance of the algorithms in the synthetic data set 2 with 10 and 100 additional irrelevant variables: average and standard deviation (in parentheses) of the ARI index, F-measure, and ER index.

p	Algorithm	ARI	F-measure	ER
10	KCM-K	0.2166 (0.0320)	0.5820 (0.0260)	0.4165 (0.0269)
	KCM-K-GH	0.8759 (0.0330)	0.9569 (0.0119)	0.0431 (0.0119)
	KCM-K-LH	0.8616 (0.0376)	0.9516 (0.0137)	0.0484 (0.0137)
	KCM-F	0.2203 (0.0343)	0.5842 (0.0267)	0.4124 (0.0273)
	KCM-F-GH	0.8754 (0.0318)	0.9567 (0.0115)	0.0432 (0.0114)
	KCM-F-LH	0.8537 (0.0545)	0.9480 (0.0232)	0.0519 (0.0232)
100	KCM-K	0.1386 (0.0376)	0.5305 (0.0314)	0.4699 (0.0291)
	KCM-K-GH	0.4829 (0.1048)	0.7300 (0.0749)	0.2700 (0.0755)
	KCM-K-LH	0.2738 (0.0801)	0.6063 (0.0479)	0.3997 (0.0402)
	KCM-F	0.1034 (0.0415)	0.5047 (0.0370)	0.4967 (0.0354)
	KCM-F-GH	0.3408 (0.0933)	0.6475 (0.0533)	0.3695 (0.0432)
	KCM-F-LH	0.2068 (0.0754)	0.5732 (0.0478)	0.4348 (0.0403)

thetic dataset 1 with 10 and 100 additional nuisance variables, respectively. Although we can observe that the KCM-K-LH and KCM-F-LH methods perform better than the other methods in the data set 1 with 10 additional irrelevant variables, a degradation of the clustering results in comparison with the results presented previously in Table 4 can also be noticed. In the case where we considered 100 additional nuisance variables, all clustering algorithms presented a poor performance.

Table 8 displays the performance, according to the ARI index, F-measure and ER index, of the clustering algorithms on the synthetic data set 2 with 10 and 100 additional nuisance variables, respectively. It can be noticed that the KCM-K-GH and KCM-F-GH methods perform better than the other methods on the data set 2 whether we consider 10 or 100 additional nuisance variables or not. It can also be observed, in comparison with the results presented previously in Table 6, that the effect of the inclusion of the nuisance variables in degrading the clustering results was lower than that verified in data set 1. Moreover, considering only 10 additional nuisance variables, the KCM-K-LH and KCM-F-LH algorithms also presented satisfactory results.

4.4. Benchmark data sets

Sixteen data sets from the UCI Machine Learning Repository [4], namely, Banknote authentication, Breast tissue, Ecoli, Glass identification, Image segmentation, Ionosphere, Iris plants, Leaf, Libras Movement, Pima Indians diabetes, Seeds, Sonar, Thyroid gland, Breast Cancer Wisconsin (diagnostic), Wine and Yeast, with different number of objects, variables, and a priori classes, were considered in this study. Table 9 (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. Also considered in this study is a standardized version of each data set of Table 9, where each variable has an average equal to zero and standard deviation equal to one.

KCM-K, KCM-F, KCM-K-GH, KCM-K-LH, KCM-F-GH, and KCM-F-LH algorithms were run on the original and standardized versions of the data sets of Table 9 until convergence was achieved 100 times. For each algorithm, the best (the most homogeneous) solution was selected, i.e., the solution (among 100) that corresponded to the minimum value of the respective objective function. Moreover, the random initialization of the width hyper-parameters of KCM-K-GH, KCM-K-LH, KCM-F-GH, and KCM-F-LH algorithms was used to run these algorithms on these data sets.

Tables 10–12 show, respectively, the ARI, the F-measure and the ER indexes computed for the best solution provided by the KCM-K, KCM-F, KCM-K-GH, KCM-K-LH, KCM-F-GH, and KCM-F-LH algorithms on the original and standardized version of the data sets of Table 9, with random initialization. Finally, Table 13 summarizes the performance of these algorithms by averaging their ranking position in terms of Accuracy (ERI index), F-measure and ARI for all data sets (original and standardized versions) of Table 9.

From these Tables, the following can be observed:

- For the original data sets of Table 9, KCM-F-GH was the best in terms of accuracy and F-measure, and the second best in terms of ARI index. KCM-K-GH was the best concerning ARI, and the second best in terms of accuracy and F-measure. Moreover, whatever the index considered, KCM-F was the worst and KCM-K was the second worst.
- For the standardized version of the data sets of Table 9, KCM-K-GH was the best and KCM-F-LH was the second best in terms of accuracy. KCM-K-LH was the best concerning the F-measure and the second best concerning ARI index. KCM-F-GH was the best in terms of ARI index and the second best in terms of F-measure. Moreover, whatever the index considered, also in this case, KCM-F was the worst and KCM-K was the second worst.
- Finally, standardization of the data sets improved the absolute value of the performance rank of the conventional KCM-K and KCM-F Gaussian kernel c-means algorithms. Despite that, for both original and standardized data sets, and whatever the index considered, they were outperformed by KCM-K-GH, KCM-K-LH, KCM-F-GH, and KCM-F-LH Gaussian kernel c-means algorithms with automated computation of the width hyper-parameters on the majority of the data sets of Table 9.

Table 9
Summary of the data sets.

Data sets	n	p	K	Data sets	n	p	K
Banknote authentication	1372	4	2	Leaf	310	14	36
Brest cancer winsconsin	569	30	2	Libras Movement	360	90	15
Breast tissue	106	9	6	Pima Indians diabetes	768	8	2
Ecoli	336	7	8	Seeds	210	7	3
Glass identification	214	9	6	Sonar	208	60	2
Image segmentation	2100	19	7	Thyroid gland	215	5	3
Ionosphere	351	32	2	Wine	178	13	3
Iris	150	4	3	Yeast	1484	8	10

Table 10
Performance of the algorithms according to the ARI index.

Data sets	Algorithms						
	Standardization	KCM-K	KCM-K-GH	KCM-K-LH	KCM-F	KCM-F-GH	KCM-F-LH
Banknote	No	0.0942	0.0114	0.0160	0.1637	0.0135	0.0030
authentication	Yes	0.0141	0.0142	0.0171	0.0074	0.0135	0.0138
Brest cancer	No	0.0029	0.7297	0.7857	0.0062	0.7358	0.7794
winsconsin	Yes	0.7612	0.6721	0.7800	0.2264	0.2844	0.3881
Breast	No	0.1501	0.3122	0.3341	0.2245	0.3084	0.3351
tissue	Yes	0.2759	0.3310	0.3633	0.3328	0.3692	0.3755
Ecoli	No	0.4227	0.4987	0.5777	0.4231	0.4913	0.6071
	Yes	0.3390	0.4685	0.6115	0.4070	0.5691	0.4286
Glass	No	0.1455	0.2286	0.1792	0.0756	0.1969	0.2294
identification	Yes	0.0993	0.2587	0.1837	0.1744	0.2435	0.2077
Image	No	0.3705	0.4390	0.5005	0.3903	0.4969	0.4967
segmentation	Yes	0.5163	0.5320	0.4946	0.4330	0.5070	0.4699
Ionosphere	No	0.1456	0.1456	0.1589	0.0467	0.0542	0.0542
	Yes	0.1286	0.1456	0.1585	0.0508	0.0612	0.0577
Iris	No	0.7436	0.8856	0.8680	0.7583	0.8856	0.8856
	Yes	0.6201	0.8856	0.8856	0.6410	0.8856	0.8856
Leaf	No	0.2693	0.4140	0.3948	0.3037	0.3830	0.3829
	Yes	0.3921	0.4063	0.3890	0.3995	0.4004	0.4016
Libras	No	0.3266	0.3172	0.2773	0.3581	0.2816	0.2572
Movement	Yes	0.3165	0.3259	0.3237	0.2946	0.3437	0.3149
Pima Indians	No	-0.0003	0.0973	0.0009	-0.0091	0.1243	0.0009
diabetes	Yes	0.1035	0.1001	0.0938	0.0985	0.1034	0.0961
Seeds	No	0.7056	0.6789	0.6789	0.4073	0.6789	0.6789
	Yes	0.7732	0.7607	0.7840	0.7979	0.7845	0.7507
Sonar	No	-0.0033	0.0085	0.0064	-0.0047	0.0064	0.0108
	Yes	0.0133	0.0085	0.0135	0.0109	0.0109	0.0161
Thyroid	No	0.1652	0.6263	0.8625	0.2139	0.8656	0.8625
gland	Yes	0.2048	0.3242	0.8625	0.3954	0.3478	0.8628
Wine	No	0.3711	0.8348	0.8150	0.3711	0.8348	0.8150
	Yes	0.8974	0.8332	0.8974	0.9485	0.9485	0.9485
Yeast	No	0.1303	0.1469	0.0761	0.1155	0.1440	0.1121
	Yes	0.1385	0.1652	0.1485	0.1155	0.1583	0.1469

Table 11
Performance of the algorithms according to the F-measure index.

Data sets	Algorithms						
	Standardization	KCM-K	KCM-K-GH	KCM-K-LH	KCM-F	KCM-F-GH	KCM-F-LH
Banknote	No	0.6509	0.5559	0.5652	0.6872	0.5607	0.5320
authentication	Yes	0.5613	0.5619	0.5676	0.5463	0.5605	0.5611
Brest cancer	No	0.5646	0.9264	0.9429	0.6034	0.9283	0.9411
winsconsin	Yes	0.9364	0.9108	0.9417	0.7408	0.7695	0.8148
Breast	No	0.3961	0.6204	0.6223	0.4580	0.5566	0.6229
tissue	Yes	0.5582	0.5985	0.6139	0.5874	0.6182	0.6201
Ecoli	No	0.6808	0.6580	0.7103	0.6568	0.6691	0.7648
	Yes	0.6044	0.6790	0.7494	0.6533	0.5738	0.6736
Glass	No	0.4782	0.5284	0.5042	0.4579	0.5054	0.5559
identification	Yes	0.3906	0.5230	0.4860	0.4734	0.5533	0.5398
Image	No	0.5668	0.6074	0.6483	0.6203	0.6523	0.6454
segmentation	Yes	0.6646	0.6553	0.6621	0.6331	0.6562	0.6224
Ionosphere	No	0.6985	0.6985	0.7067	0.6149	0.6227	0.6227
	Yes	0.6873	0.6985	0.7063	0.6193	0.6294	0.6261
Iris	No	0.8987	0.9599	0.9599	0.9053	0.9600	0.9599
	Yes	0.8331	0.9599	0.9599	0.8466	0.9599	0.9599
Leaf	No	0.4773	0.6006	0.5636	0.4989	0.5792	0.5417
	Yes	0.5459	0.5796	0.5729	0.5943	0.5820	0.5773
Libras	No	0.5230	0.5013	0.4888	0.5303	0.4924	0.4620
Movement	Yes	0.5152	0.5093	0.5491	0.4840	0.5091	0.4685
Pima Indians	No	0.5367	0.6626	0.5345	0.5747	0.6845	0.5345
diabetes	Yes	0.6683	0.6651	0.6619	0.6648	0.6690	0.6632
Seeds	No	0.8903	0.8747	0.8747	0.7632	0.8747	0.8747
	Yes	0.9193	0.9147	0.9243	0.9287	0.9243	0.9097
Sonar	No	0.5535	0.5573	0.5532	0.6109	0.5530	0.5629
	Yes	0.5675	0.5573	0.6281	0.5930	0.5930	0.5901
Thyroid	No	0.6257	0.8786	0.9570	0.5971	0.9520	0.9570
gland	Yes	0.6262	0.7387	0.8625	0.7013	0.6900	0.9578
Wine	No	0.7147	0.9430	0.9372	0.7147	0.9430	0.9371
	Yes	0.9660	0.9429	0.9660	0.9831	0.9831	0.9831
Yeast	No	0.4146	0.4135	0.3739	0.3844	0.4126	0.4048
	Yes	0.3924	0.4590	0.4701	0.3869	0.4422	0.4459

Table 12
Performance of the algorithms according to the ER index.

Data sets	Algorithms						
	Standardization	KCM-K	KCM-K-GH	KCM-K-LH	KCM-F	KCM-F-GH	KCM-F-LH
Banknote	No	0.3454	0.4446	0.4351	0.2959	0.4402	0.4446
authentication	Yes	0.4387	0.4387	0.4329	0.4446	0.4402	0.4395
Brest cancer	No	0.3725	0.0720	0.0562	0.3725	0.0702	0.0579
winsconsin	Yes	0.0632	0.0896	0.0579	0.2601	0.2319	0.1880
Breast	No	0.5660	0.4056	0.3962	0.5188	0.4528	0.4339
tissue	Yes	0.4245	0.3867	0.4056	0.3867	0.3773	0.3773
Ecoli	No	0.1666	0.1636	0.2410	0.1607	0.1398	0.2023
	Yes	0.2142	0.1845	0.2202	0.2113	0.2142	0.1904
Glass	No	0.4112	0.4719	0.4485	0.4579	0.3831	0.3598
identification	Yes	0.5000	0.4439	0.4532	0.4672	0.3785	0.3738
Image	No	0.4080	0.3738	0.3566	0.3995	0.3628	0.3604
segmentation	Yes	0.3490	0.3228	0.3447	0.3957	0.3504	0.3923
Ionosphere	No	0.3076	0.3076	0.2991	0.3589	0.3589	0.3589
	Yes	0.3190	0.3076	0.2991	0.3589	0.3589	0.3589
Iris	No	0.1000	0.0400	0.0466	0.0933	0.0400	0.0400
	Yes	0.1666	0.0400	0.0400	0.1533	0.0400	0.0400
Leaf	No	0.4911	0.3794	0.3948	0.4617	0.3911	0.4323
	Yes	0.4264	0.3970	0.4000	0.3882	0.3941	0.3941
Libras	No	0.4916	0.5166	0.5277	0.4805	0.5527	0.5444
Movement	Yes	0.5111	0.5166	0.5491	0.5416	0.5250	0.5416
Pima Indians	No	0.3489	0.3372	0.3489	0.3489	0.3229	0.3489
diabetes	Yes	0.3346	0.3359	0.3450	0.3398	0.3372	0.3424
Seeds	No	0.1095	0.1238	0.1238	0.2380	0.1238	0.1238
	Yes	0.0809	0.0857	0.0761	0.0714	0.0761	0.0904
Sonar	No	0.4663	0.4423	0.4471	0.4663	0.4471	0.4375
	Yes	0.4326	0.4423	0.4326	0.4375	0.4375	0.4278
Thyroid	No	0.2465	0.1116	0.0418	0.2418	0.0465	0.0418
gland	Yes	0.2093	0.1767	0.0418	0.1534	0.1860	0.0418
Wine	No	0.2977	0.0561	0.0617	0.2977	0.0561	0.0617
	Yes	0.0337	0.0561	0.0337	0.0168	0.0168	0.0168
Yeast	No	0.4568	0.4602	0.5498	0.4730	0.4838	0.5357
	Yes	0.4346	0.4595	0.4615	0.4764	0.4669	0.4656

Table 13
Average performance ranking.

Metrics	Standardization	Algorithms					
		KCM-K	KCM-K-GH	KCM-K-LH	KCM-F	KCM-F-GH	KCM-F-LH
ER	No	4.09	3.00	3.15	4.37	2.96	3.40
	Yes	3.68	3.09	3.34	4.09	3.56	3.21
F-measure	No	4.00	3.09	3.15	4.46	3.03	3.25
	Yes	4.21	3.34	2.40	4.21	3.15	3.71
ARI	No	4.62	2.50	3.00	4.78	3.00	3.06
	R	4.15	3.03	3.00	4.59	2.87	3.34

4.5. Iris data set

This section considers the results provided by the KCM-K-LH clustering algorithm when applied to the Iris data set from Table 9, aiming to show its usefulness. The Iris data set describes 150 objects defined by four real-valued variables, namely *sepal length, sepal width, petal length* and *petal width*. The objects are partitioned according to an a priori classification in three classes, namely *Iris Setosa, Iris Versicolour* and *Iris Virginica*. Each a priori class has 50 objects.

The KCM-K-LH algorithm was run on the original iris data set until convergence was achieved 100 times. Here we consider the best solution (among 100), according to the minimum value of the corresponding objective function Eq. (11). Table 14 shows the confusion matrix between the a priori partition of this data set and the hard cluster partition provided by the KCM-K-LH algorithm. The good performance of this algorithm on this data set can be observed.

Table 15 provides the representatives of the clusters. It can be observed that they are differentiated mainly by the real-valued variables *petal length* and *petal width*.

Table 14
Iris data set: confusion matrix.

A priori classes	Clusters		
	1	2	3
Iris Setosa	50	0	0
Iris Versicolour	0	3	47
Iris Virginica	0	46	4

Table 15
Iris data set: matrix of the cluster prototypes.

Real-valued	Clusters		
variables	1	2	3
sepal length	5.0009	6.5460	5.9292
sepal width	3.4064	3.0029	2.7693
petal length	1.4645	5.4448	4.3113
petal width	0.2373	2.0387	1.3300

Table 16 shows the four-components vector of width hyper-parameters of the real-valued variables on each cluster provided by the KCM-K-LH algorithm.

Table 16
Iris data set: width hyper-parameters of the real-valued variables on the clusters.

Real-valued	Clusters		
variables	1	2	3
sepal length	4.5629	3.6614	3.9566
sepal width	5.1506	0.8794	1.5084
petal length	1.0750	3.0820	3.3840
petal width	0.4108	1.0460	0.5140

As it has been pointed out, the closer the objects are to the representative of a given cluster concerning a given real-valued variable, the lower the width hyper-parameter of this variable is on this cluster. All the objects of cluster 1 belong to the a priori class 1 *Iris Setosa*. In this cluster, the variable *petal width* has the lowest width hyper-parameter, whereas the variable *sepal width* has the highest width hyper-parameter. The variable *petal width* is more relevant for cluster 1 than the variable *sepal width* because a small difference between an object and the representative of cluster 1 is amplified on the former variable, whereas an important difference between an object and the representative of cluster 1 is reduced on the latter variable.

5. Final remarks and conclusions

Gaussian kernel functions have an important parameter, the width hyper-parameter, that needs to be tuned. Usually this parameter is tuned once and for all, and it is the same for all variables. Therefore, implicitly, the variables have the same importance on the clustering task.

This paper presented Gaussian kernel c-means hard clustering algorithms, with both kernelization of the metric and in the feature space, and with automated computation of the width hyper-parameters. In the proposed kernel-based clustering algorithms, the hyper-parameters change at each iteration of the algorithm, they differ from variable to variable and can differ from cluster to cluster. Thus, the proposed algorithms are able to rescale the variables differently and, therefore, select the important variables for the clustering task.

Experiments with synthetic data sets were performed by considering different sample sizes as well as different numbers of irrelevant or nuisance variables aiming to demonstrate the ability of the proposed methods to automatically learn the hyper-parameter of the Gaussian kernel functions and select the relevant variables for the clustering task. In particular, the superiority of the Gaussian kernel c-means hard clustering algorithms with automated computation of a width hyper-parameter for each variable in each cluster on the synthetic data sets with a specific set of relevant variables to each cluster was observed, regardless of the sample size considered. Moreover, for the synthetic data sets with the same set of relevant variables for all clusters, the Gaussian kernel c-means clustering hard algorithms with automated computation of a width hyper-parameter globally for each variable were the best. When the number of irrelevant variables is greater than the number of important variables, it was noticed that, with 10 additional nuisance variables, the Gaussian kernel c-means hard clustering algorithms with automated computation of a width hyper-parameter for each variable in each cluster on the synthetic data sets with a specific set of relevant variables to each cluster still have the best performance. A similar conclusion can be drawn from the Gaussian kernel c-means clustering hard algorithms with automated computation of a width hyper-parameter globally for each variable on the synthetic data sets with the same set of relevant variables for all clusters. Finally, in the cases where we considered 100 additional nuisance variables, all clustering algorithms presented a poor performance.

Additionally, experiments with sixteen data sets from the UCI machine learning repository, with a different number of objects, variables and a priori classes, showed the performance of the proposed algorithm. Based on the computation of the ARI index, F-measure, and ER index, it can be concluded that, in most the data sets considered in this study, the proposed Gaussian kernel c-means hard clustering algorithms with automated computation of the hyper-parameters were superior to the conventional Gaussian kernel c-means hard clustering algorithms.

Finally, the proposed algorithms provide the possibility of different scales across different variables and across different clusters. However, they are not able to address correlated variables and clusters oriented in different directions [9], and this constitutes a limitation of the algorithms proposed in this paper. For future work, we intend to overcome these limitations by considering Gaussian kernel functions with a matrix of width hyper-parameters.

Acknowledgments

The authors are grateful to the anonymous referees for their careful revision, valuable suggestions, and comments which improved this paper. The authors would like to thank FACEPE, Research Agency from the State of Pernambuco, Brazil (APQ-1529-1.03/15), and CNPq, National Council for Scientific and Technological Development, Brazil (303187/2013-1), for their financial support.

References

[1] A.A. Abin, H. Beigy, Active constrained fuzzy clustering: a multiple kernels learning approach, Pattern Recognit. 48 (2015) 953–967.
[2] J.C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Advanced applications in pattern recognition, Plenum Press, New York, 1981.
[3] M. Bicego, M.A.T. Figueiredo, Soft clustering using weighted one-class support vector machines, Pattern Recognit. 42 (2009) 27–32.
[4] C. Black, C. Merz, UCI Repository of machine learning databases, 1998, (http://www.ics.uci.edu/mlearn/MLRepository.html).
[5] S. Borer, W. Gerstner, A new kernel clustering algorithm, in: Proceedings of the 9th International Conference on Neural Information Processing. ICONIP'02, 5, 2002, pp. 2527–2531.
[6] L. Breiman, J. Friedman, C. Stone, R. Olshen, Classification and Regression Trees, Chapman and Hall/CRC, Boca Raton, 1984.
[7] F. Camastra, A. Verri, A novel kernel method for clustering, IEEE Trans. Neural Netw. 27 (2005) 801–804.
[8] B. Caputo, K. Sim, F. Furesjo, A. Smola, Appearence-based object recognition using svms: wich kernel should i use? in: Proceedings of NIPS Workshop on Statistical Methods for Computational Experiments in Visual Processing and Computer vision, 2002.
[9] J.E. Chacn, T. Duong, Data-driven density derivative estimation, with applications to nonparametric clustering and bump hunting, Electron. J. Stat. 7 (2013) 499–532.
[10] E.Y. Chan, W.K. Ching, M.K. Ng, J.Z. Huang, An optimization algorithm for clustering using weighted dissimilarity measures, Pattern Recognit. 37 (5) (2004) 943–952.
[11] S.C. Chen, D.Q. Zhang, Robust image segmentation using fcm with spatial constraints based on new kernel-induced distance measure, IEEE Trans. Syst. Man Cybern. 34 (4) (2004) 1907–1916.
[12] R. Chitta, R. Jin, T.C. Havens, A.K. Jain, Scalable kernel clustering: approximate kernel k-means, CoRR (2014). arXiv:1402.3849.
[13] G. Cleuziou, J.G. Moreno, Kernel methods for point symmetry-based clustering, Pattern Recognit. 48 (2015) 2812–2830.
[14] F.A.T. de Carvalho, M.R.P. Ferreira, E.C.S. oes, A Gaussian kernel-based clustering algorithm with automatic hyper-parameters computation, in: L. Cheng, Q. Liu, A. Ronzhin (Eds.), Lecture Notes in Computer Science, Advances in Neural Networks-ISNN 2016, Springer, Heidelberg, 2016, pp. 393–400.
[15] E. Diday, G. Govaert, Classification automatique avec distances adaptatives, R.A.I.R.O. Inf. Comput. Sci. 11 (4) (1977) 329–349.
[16] E. Diday, J.C. Simon, Clustering analysis, in: K. Fu (Ed.), Digital Pattern Classification, Springer, Berlin, 1976, pp. 47–94.
[17] S. Ding, X. Xu, S. Fan, Y. Xue, Locally adaptive multiple kernel k-means algorithm based on shared nearest neighbors, Soft Comput. (2017). https://doi.org/10.1007/s00500-017-2640-5.
[18] Y. Ding, X. Fu, Kernel-based fuzzy c-means clustering algorithm based on genetic algorithm, Neurocomputing 188 (2016) 233–238.

[19] M. Fauvel, J. Chanussot, J. Benediktsson, A. Villa, Parsimonious mahalanobis kernel for the classification of high dimensional data, Pattern Recognit. 46 (2013) 845–854.
[20] M.R.P. Ferreira, F.A.T. de Carvalho, Kernel-based hard clustering methods in the feature space with automatic variable weighting, Pattern Recognit. 47 (2014) 3082–3095.
[21] M.R.P. Ferreira, F.A.T. de Carvalho, E.C.S. oes, Kernel-based hard clustering methods with kernelization of the metric and automatic weighting of the variables, Pattern Recognit. 51 (2016) 310–321.
[22] M. Filippone, F. Camastra, F. Masulli, S. Rovetta, A survey of kernel and spectral methods for clustering, Pattern Recognit. 41 (2008) 176–190.
[23] E. Forgy, Cluster analysis of multivariate data: efficiency vs. interpretability of classifications, Biometrics 21 (1965) 768–769.
[24] M. Girolami, Mercer kernel-based clustering in feature space, IEEE Trans. Neural Netw. 13 (2002) 780–784.
[25] A.D. Gordon, Classification, 2, Chapman & Hall, Boca Raton, 1999.
[26] D. Graves, W. Pedrycz, Kernel-based fuzzy clustering and fuzzy clustering: a comparative experimental study, Fuzzy Sets Syst. 161 (2010) 522–543.
[27] H.-C. Huang, Y.-Y. Chuang, C.-S. Chen, Multiple kernel fuzzy clustering, IEEE Trans. Fuzzy Syst. 20 (1) (2012) 120–134.
[28] J.Z. Huang, M.K. Ng, H. Rong, Z. Li, Automated variable weighting in k-means type clustering, Pattern Anal. Mach. Intell. IEEE Trans. 27 (5) (2005) 657–668.
[29] L. Hubert, P. Arabie, Comparing partitions, J. Classif. 2 (1985) 193–218.
[30] R. Inokuchi, S. Miyamoto, Lvq clustering and som using a kernel function, in: Proceedings of IEEE International Conference on Fuzzy Systems, 3, 2004, pp. 1497–1500.
[31] A.K. Jain, Data clustering: 50 years beyond k-means, Pattern Recognit. Lett. 31 (2010) 651–666.
[32] A.K. Jain, M.N. Murty, P.J. Flynn, Data clustering: a review, ACM Comput. Surv. 31 (3) (1999). 264–233
[33] D.W. Kim, K.Y. Lee, D. Lee, K.H. Lee, Evaluation of the performance of clustering algorithms in kernel-induced feature space, Pattern Recognit. 38 (4) (2005) 607–611.
[34] D.W. Kim, K.Y. Lee, D. Lee, K.H. Lee, Evaluation of the performance of clustering algorithms in kernel-induced feature space, Pattern Recognit. Lett. 26 (2005) 879–891.
[35] D.W. Kim, K.Y. Lee, D. Lee, K.H. Lee, A kernel-based subtractive clustering method, Pattern Recognit. Lett. 26 (2005) 879–891.
[36] T. Kohonen, Self-organized formation of topologically correct feature maps, Biol. Cybern. 43 (1) (1982) 59–69.
[37] T. Kohonen, The self-organizing map, in: Proceedings of the IEEE, 78, 1990, pp. 1464–1480.
[38] T. Kohonen, Self-Organizing Maps, Springer, New York, 2001.
[39] T. Kohonen, Essentials of the self-organizing map, Neural Netw. 37 (2013) 52–65.
[40] L. Kong, L. Chen, Approximate fuzzy kernel clustering with random feature mapping and dimension reduction, in: 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery, ICNC-FSKD 2016, Changsha, China, August 13–15, 2016, 2016, pp. 960–965.
[41] Y. Lu, L. Wang, J. Lu, J. Yang, C. Shen, Multiple kernel clustering based on centered kernel alignment, Pattern Recognit. 47 (11) (2014) 3656–3664.
[42] D. Macdonald, C. Fyfe, The kernel self-organizing map, in: Fourth International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies, 1, 2000, pp. 317–320.
[43] C.D. Manning, P. Raghavan, H. Schuetze, Introduction to Information Retrieval, Cambridge University Press, Cambridge, UK, 2008.
[44] T.M. Martinetz, S.G. Berkovich, K.J. Schulten, 'Neural gas' network for vector quantization and its application to time-series prediction, IEEE Trans. Neural Netw. 4 (4) (1993) 558–569.
[45] G.W. Milligan, Clustering validation: results and implications for applied analysis, in: P. Arabie, L.J. Hubert, G.D. Soete (Eds.), Clustering and Classification, World Scientific, River Edge, NJ, 1996, pp. 341–375.
[46] D.S. Modha, W.S. Spangler, Feature weighting in k-means clustering, Mach. Learn. 52 (3) (2003) 217–237.
[47] K.R. Müller, S. Mika, G. Rätsch, K. Tsuda, B. Schölkopf, An introduction to kernel-based learning algorithms, IEEE Trans. Neural Netw. 12 (2) (2001) 181–202.
[48] J.M. Pena, J.A. Lozano, P. Larranaga, An empirical comparison of four initialization methods for the k-means algorithm, Pattern Recognit. Lett. 20 (1999) 1027–1040.
[49] A.K. Qinand, P.N. Suganthan, Kernel neural gas algorithms with application to cluster analysis, in: ICPR – 17th International Conference on Pattern Recognition (ICPR'04), 4, 2004, pp. 617–620.
[50] B. Schölkopf, A.J. Smola, K.R. Müller, Nonlinear component analysis as a kernel eigenvalue problem, Neural Comput. 10 (5) (1998) 1299–1319.
[51] C.Y. Tsai, C.C. Chiu, Developing a feature weight self-adjustment mechanism for a k-means clustering algorithm, Comput. Stat. Data Anal. 52 (2008) 4658–4672.
[52] N. Tsapanos, A. Tefas, N. Nikolaidis, I. Pitas, A distributed framework for trimmed kernel k-means clustering, Pattern Recognit. 48 (2015) 2685–2698.
[53] J. Wang, Z. Deng, K.-S. Choi, Y. Jiang, X. Luo, F.-L. Chung, S. Wang, Distance metric learning for soft subspace clustering in composite kernel space, Pattern Recognit. 52 (2016) 113–134.
[54] Q. Wang, Y. Ye, J.Z. Huang, Fuzzy k-means with variable weighting in high dimensional data analysis, in: Proceedings of the Ninth International Conference on Web-Age Information Managemen - WAIM'08, 2008, pp. 365–372.
[55] S. Wang, A. Gittens, M.W. Mahoney, Scalable kernel k-means clustering with nystrom approximation: relative-error bounds, CoRR (2017). arXiv:1706.02803.
[56] D. Xiang, T. Tang, C. Hu, Y. Li, Y. Su, A kernel clustering algorithm with fuzzy factor: application to SAR image segmentation, IEEE Geosci. Remote Sens. Lett. 11 (7) (2014) 1290–1294.
[57] R. Xu, D.I.I. Wunusch, Survey of clustering algorithms, IEEE Trans. Neural Netw. 16 (3) (2005) 645–678.
[58] R.R. Yager, D.P. Filev, Approximate clustering via the mountain method, IEEE Trans. Syst. Man, Cybern. 24 (8) (1994) 1279–1284.
[59] B. Yan, P. Sarkar, On robustness of kernel clustering, in: Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5–10, 2016, Barcelona, Spain, 2016, pp. 3090–3098.
[60] D. Yang, R. Fei, J. Yao, M. Gong, Two-stage SAR image segmentation framework with an efficient union filter and multi-objective kernel clustering, Appl. Soft Comput. 44 (2016) 30–44.
[61] X. Yin, S. Chen, E. Hu, D. Zhang, Semi-supervised clustering with metric learning: an adaptive kernel method, Pattern Recognit. 43 (2010) 1320–1333.
[62] D.Q. Zhang, S.C. Chen, Fuzzy clustering using kernel method, in: Proceedings of the 2002 International Conference on Control and Automation - ICCA 2002, 2002, pp. 123–127.
[63] L. Zhang, X. Hu, A novel multiple kernel clustering method, in: Emerging Intelligent Computing Technology and Applications - 8th International Conference, ICIC 2012, Huangshan, China, July 25–29, 2012. Proceedings, 2012, pp. 87–92.
[64] L. Zhang, X. Hu, Locally adaptive multiple kernel clustering, Neurocomputing 137 (2014) 192–197.

Francisco de A.T. de Carvalho received the Ph.D. degree in Computer Science in 1992 from Institut National de Recherche en Informatique et en Automatique (INRIA) and Université Paris-IX Dauphine, France. From 1992 to 1998, he was a lecturer at Statistical Department at Universidade Federal de Pernambuco, Brazil. He joined the Center of Informatics at Universidade Federal de Pernambuco in 1999, where he is currently Full Professor. He held visiting posts in several leading universities and research centers in Europe. With main research interests in symbolic data analysis, clustering analysis and machine learning he has authored over 200 technical papers in international journals and conferences. He has served as Coordinator (2005–2009) of the post-graduate program of computer science of the CIn/UFPE. He has been involved in program committees of many Brazilian and international conferences. He has also served as review of many international journals and conferences. He was member for the council (2009–2013) of the International Association for Statistical Computing (IASC).

Eduardo C. Simões currently is an undergraduate student in computer engineering at Universidade Federal de Pernambuco, Brazil.

Lucas V.C. Santana currently is an undergraduate student in computer engineering at Universidade Federal de Pernambuco, Brazil.

Marcelo R.P. Ferreira received a Ph.D. degree in Computer Science in 2013 from the Center of Informatics at Federal University of Pernambuco, Brazil. He earned both his B.Sc. and M.Sc. degrees in Statistics from the Department of Statistics at Federal University of Pernambuco, Brazil. He joined the Department of Statistics at Federal University of Paraiba, Brazil, in 2009, where he is currently an Assistant Professor. With research interests in clustering methods, statistical pattern recognition, bootstrap, nonparametric methods and symbolic data analysis, he has authored technical papers in international journals and conferences. He has also served as a reviewer of various international journals.