Fuzzy and Crisp Gaussian Kernel-Based Co-clustering with Automatic Bandwidth Computation

José Nataniel A. de Sá [1] **Marcelo R. P. Ferreira** [2]
Francisco de A.T. de Carvalho [1]

Centro de Informática, Universidade Federal de Pernambuco [1]
Departamento de Estatıstica, Universidade Federal da Paraíba [2]

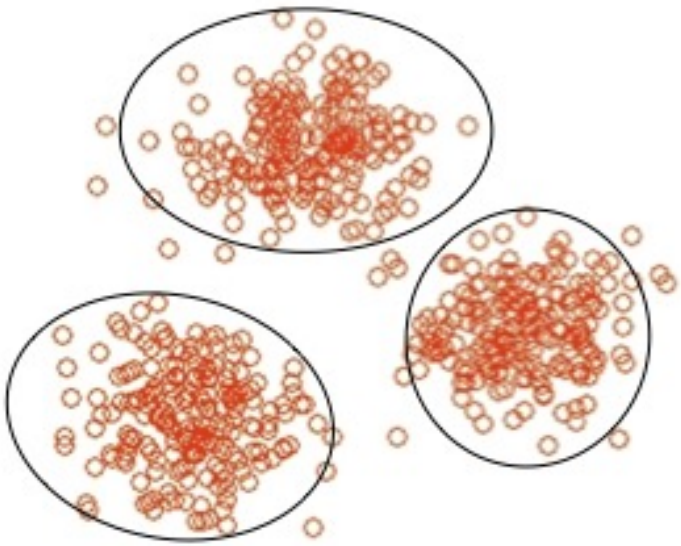
Conservatoire des Arts et Métiers - CNAM
Paris, France, February, 2026

Clustering

- Clustering: aims to divide a data set into clusters such that
  - objects belonging to the same cluster are similar to each other and dissimilar to objects belonging to other clusters
- Clustering is successfully applied in different fields:
  - bioinformatics, image processing, information retrieval, etc
- the most popular cluster structures produced by clustering algorithms:
  - hierarchy and partition
- there is many approaches and algorithms to data clustering
  - however, k-means is still the most popular

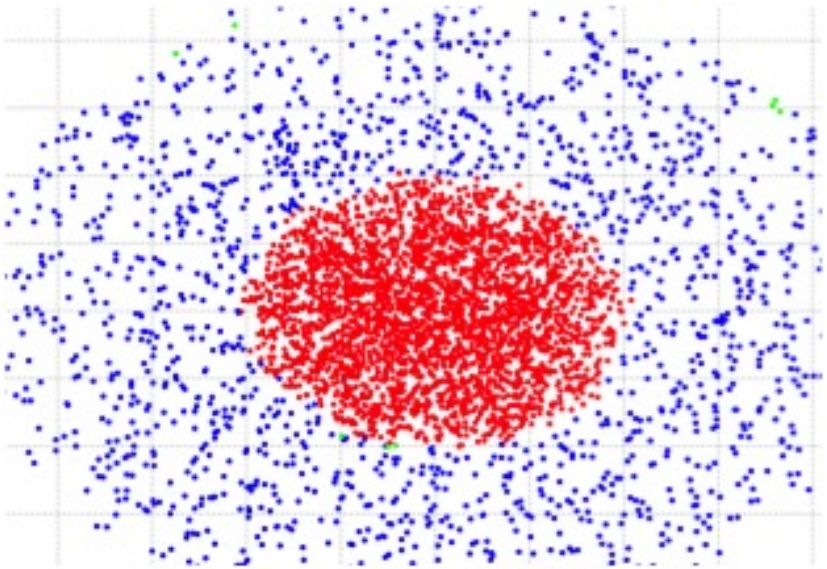
K-means

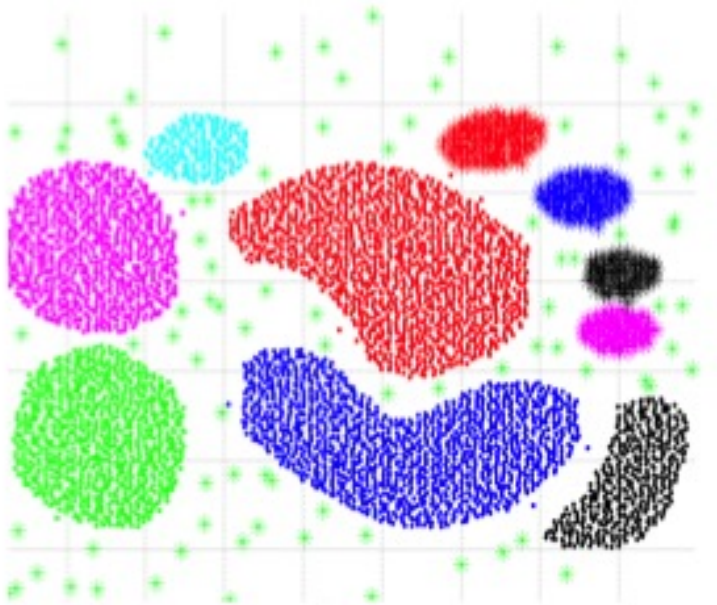
- k-means works well when the clusters are compact, linearly separable



K-means drawbacks

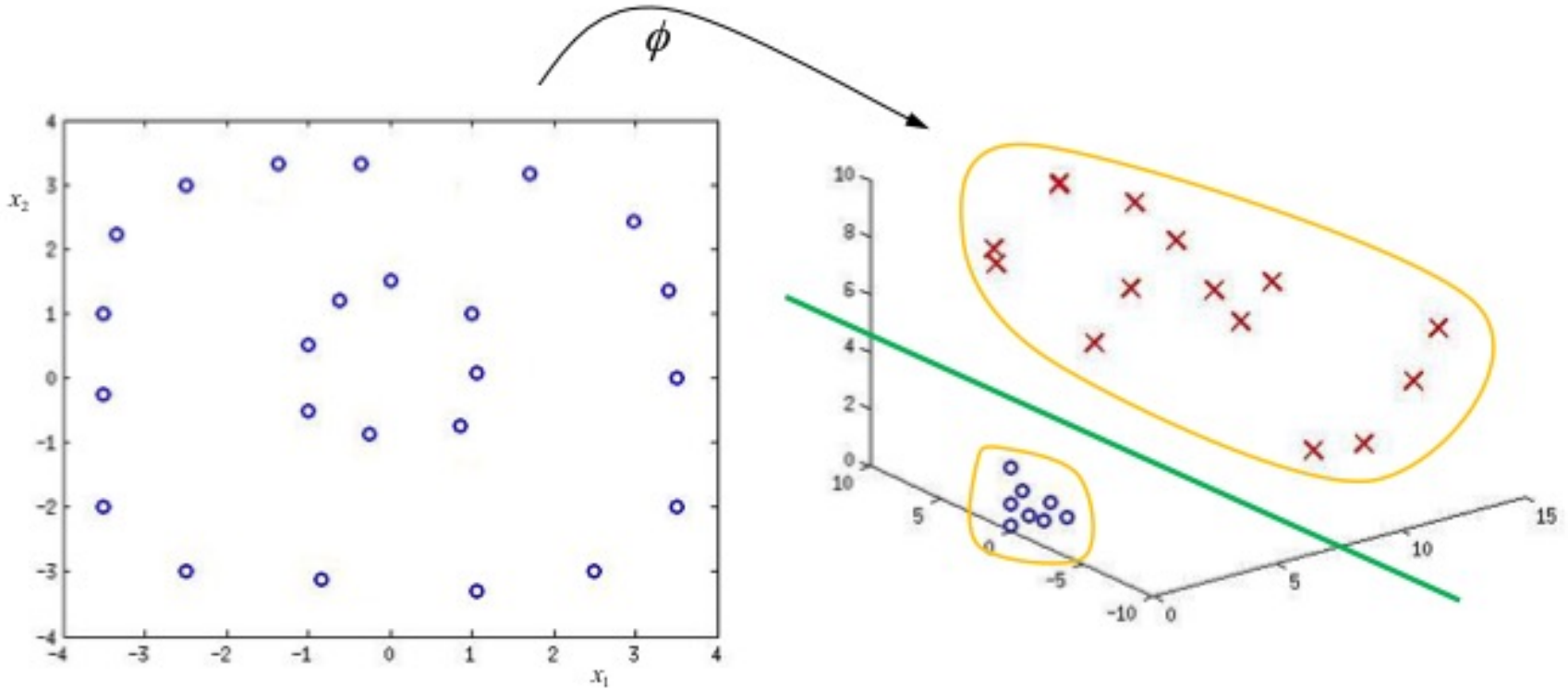
- k-means fails when the clusters are arbitrarily shaped, of different densities, not linearly separable



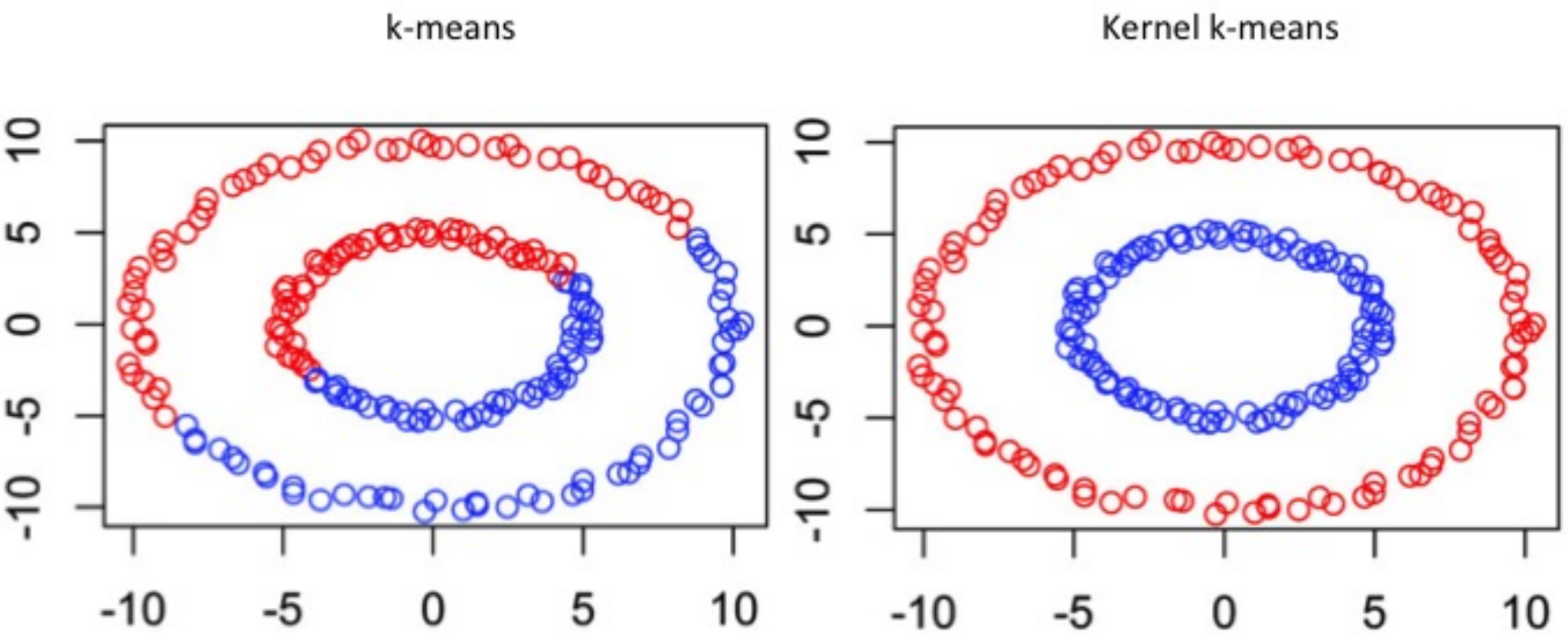


Kernel based clustering

- Kernel based clustering: the basic idea [10]
  - to project the data into a higher dimensional space through some (nonlinear) mapping
  - expectation: in high-dimensional spaces it will be obtained well defined and linearly separable groups



K-means versus Kernel based k-means clustering



Mercer Kernel

- Applied successfully on regression, classification and clustering [24], [25]
- $E = \{e_1, \ldots, e_n\}$: a set of n objects described by p real-valued variables;
- $\mathbf{x}_i = (x_{i1}, \ldots, x_{iP}) \in \mathbb{R}^p$: the description of the k^{th} object e_i;
- $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$: data set
- $\mathcal{K} : \mathcal{D} \times \mathcal{D} \rightarrow \mathbb{R}$ is a positive definite kernel, Mercer kernel [11]:
  - $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_k) = \mathcal{K}(\mathbf{x}_k, \mathbf{x}_i)$ (symmetry)
  - $\sum_{i=1}^{n} \sum_{k=1}^{n} c_i c_k \mathcal{K}(\mathbf{x}_i, \mathbf{x}_k) \geq 0, \ \forall n \geq 2, \ c_i, c_k \in \mathbb{R}$

Mercer Kernel

- $\Phi : \mathcal{D} \rightarrow \mathcal{F}$: a nonlinear mapping from $\mathcal{D}$ to a high dimensional feature space $\mathcal{F}$;
- Key idea: Φ does not need to be explicitly specified because each Mercer kernel can be expressed as $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_k) = \Phi(\mathbf{x}_i)^\top \Phi(\mathbf{x}_k)$
- Application: compute Euclidean distances in $\mathcal{F}$ without knowing explicitly Φ (kernel trick):

$$\|\Phi(\mathbf{x}_l) - \Phi(\mathbf{x}_k)\|^2 = (\Phi(\mathbf{x}_l) - \Phi(\mathbf{x}_k))^\top (\Phi(\mathbf{x}_l) - \Phi(\mathbf{x}_k)) = \mathcal{K}(\mathbf{x}_l, \mathbf{x}_l) - 2\mathcal{K}(\mathbf{x}_l, \mathbf{x}_k) + \mathcal{K}(\mathbf{x}_k, \mathbf{x}_k)$$

Gaussian Kernel K-means with prototypes in the original input space

- The cluster representatives are in the original input space
- It provides a crisp partition of the objects into K clusters as well as the cluster representatives (prototypes)
- based on a suitable objective function
- Let:
  - $E = \{e_1, \ldots, e_N\}$: the set of examples, individuals, objects, etc.
  - $\mathbf{x}_i = (x_{i1}, \ldots, x_{ij}, \ldots, x_{iP})$: description of object e_i $(1 \leq i \leq N)$
  - $\mathbf{X} = (x_{ij})_{\substack{1 \leq i \leq N \\ 1 \leq j \leq P}}$: data matrix
  - $\mathcal{P} = \{P_1, \ldots, P_K\}$: crisp partition of E in K clusters
  - $\mathbf{g}_k = (g_{k1}, \ldots, g_{kj}, \ldots, g_{kP})$: cluster representative (prototype) of cluster P_k $(1 \leq k \leq K)$
  - $\mathbf{G} = (g_{kj})_{\substack{1 \leq k \leq K \\ 1 \leq j \leq P}}$: matrix of prototypes
  - $\mathbf{U} = [u_{ik}]_{\substack{1 \leq i \leq N \\ 1 \leq k \leq K}}$ is the crisp membership matrix of the objects, where u_{ik} denotes the membership degree of the object i into k^{th} object cluster, s.t. $u_{ik} \in \{0, 1\}$ and $\sum_{k=1}^{K} u_{ik} = 1$;

Gaussian Kernel K-means with prototypes in the original input space

- Objective Function I

$$J(\mathbf{G}, \mathbf{U}) = \sum_{k=1}^{K} \sum_{i=1}^{n} (u_{ik}) \|\Phi(\mathbf{x}_i) - \Phi(\mathbf{g}_k)\|^2$$

- Kernel trick

$$\begin{aligned} \|\Phi(\mathbf{x}_i) - \Phi(\mathbf{g}_k)\|^2 &= (\Phi(\mathbf{x}_i) - \Phi(\mathbf{g}_k))^\top (\Phi(\mathbf{x}_i) - \Phi(\mathbf{g}_k)) \\ &= \mathcal{K}(\mathbf{x}_i, \mathbf{x}_i) - 2\mathcal{K}(\mathbf{x}_i, \mathbf{g}_k) + \mathcal{K}(\mathbf{g}_k, \mathbf{g}_k) \end{aligned}$$

- Objective Function II

$$J(\mathbf{G}, \mathbf{U}) = \sum_{k=1}^{K} \sum_{i=1}^{n} (u_{ik}) \left(\mathcal{K}(\mathbf{x}_i, \mathbf{x}_i) - 2\mathcal{K}(\mathbf{x}_i, \mathbf{g}_k) + \mathcal{K}(\mathbf{g}_k, \mathbf{g}_k)\right)$$

Gaussian Kernel K-means with prototypes in the original input space

- Gaussian kernel

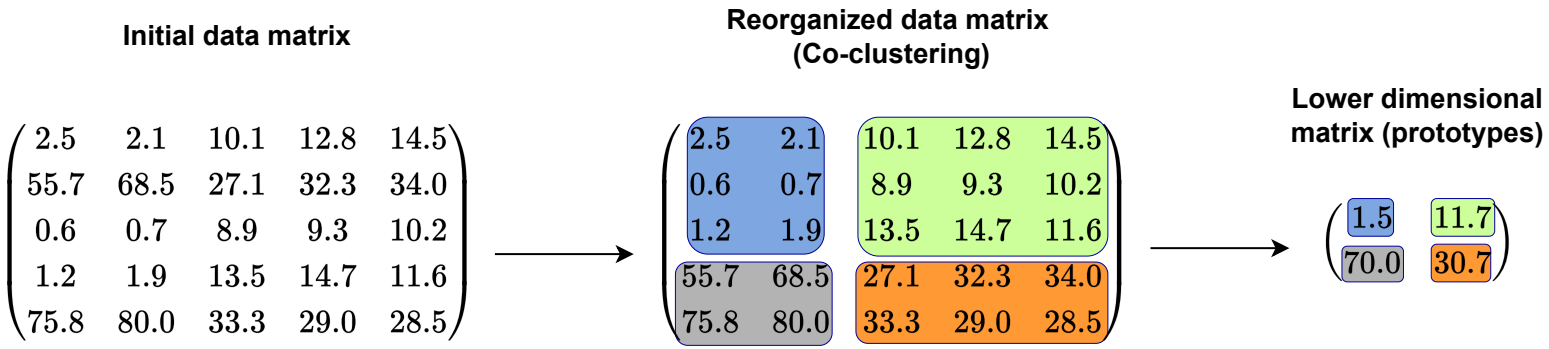
$$\mathcal{K}_{\mathcal{G}}(\mathbf{x}_i, \mathbf{g}_k) = \exp\left\{-\frac{\|\mathbf{x}_i - \mathbf{g}_k\|^2}{2\sigma^2}\right\} = \exp\left\{-\frac{1}{2}\sum_{j=1}^{p}\frac{1}{\sigma^2}(x_{ij} - g_{kj})^2\right\}$$

- σ^2 is the bandwidth hyper-parameter of the Gaussian kernel
- σ^2 is the same for all the variables and it is tuned once and for all
- Objective Function III

$$J(\mathbf{G}, \mathbf{U}) = \sum_{k=1}^{K}\sum_{i=1}^{n}(u_{ik})\,(2 - 2\,\mathcal{K}_{\mathcal{G}}(\mathbf{x}_i, \mathbf{g}_k))$$

Co-clustering (double k-means)

- Unlike traditional clustering, that peform one-side clustering, Co-clustering is a technique that simultaneously groups objects and variables in a data matrix;
- Co-clustering algoritms seek by blocks (co-cluster) of objects and variables whose elements are similar;
- Provide a lower-dimensional matrix consisting of representative values of each block.



Motivation

- Among the kernel based clustering algorithms, the conventional Gaussian kernel c-means is the most popular
  - it provides good results and requires the tunning of a single parameter, the bandwidth parameter
  - this parameter is tuned once and for all (using strategies such as cross-validation or empirical techniques) and it is the same for all variables
  - thus, the variables are equally rescaled and implicitly it is assumed that they have the same importance for the clustering task
- However [9, 14, 20]
  - some variables are irrelevant while others have different levels of relevance for the clustering task
  - each cluster may have its own different set of relevant variables

Proposal

- *Fuzzy (GKFDK) and Crisp (GKHDK) Gaussian Kernel-Based Co-clustering with Automatic Bandwidth Computation*;
  - We propose hard and fuzzy kernel co-clustering with learnable bandwidth parameters
  - Since co-clustering simultaneously groups objects and variables, we introduced bandwidth parameter structures for objects, variables or both
  - These bandwidth parameters can be the same for all clusters (global methods), or they can also vary across clusters (local methods)
  - In local methods, when the bandwidth parameter is defined in terms of objects, these are different for each variable cluster
  - Likewise, when the bandwidth parameter is defined in terms of variables, these are different for each object cluster
  - Main advantages
    - There is no need for an additional step to tune the bandwidth parameter
    - This parameter is not unique for the entire dataset, but different for objects, variables or both
    - The proposed methods can rescale the objects and variables separately, according to their distribution
    - In the local case, the proposed methods can rescale also according to the distribution in each variable cluster and object cluster

Definitions

- N is the number of objects, P is the number of variables, K ($K < N$) is the number of object clusters and H ($H < P$) is the number of variable clusters;
- $\mathbf{X} = [x_{ij}]_{\substack{1 \leq i \leq N \\ 1 \leq j \leq P}}$ ($x_{ij} \in \mathbb{R}$) is a data matrix (dataset);
- $\mathbf{G} = [g_{kh}]_{\substack{1 \leq k \leq K \\ 1 \leq h \leq H}}$ ($g_{kh} \in \mathbb{R}$) is the prototype matrix (representative of the co-clusters);
- $\mathbf{U} = [u_{ik}]_{\substack{1 \leq i \leq N \\ 1 \leq k \leq K}}$ is the (crisp, fuzzy) membership matrix of the objects, where u_{ik} denotes the membership degree of the object i into k^{th} object cluster, s.t. either $u_{ik} \in \{0, 1\}$ and $\sum_{k=1}^{K} u_{ik} = 1$ (crisp versions) OR $u_{ik} \in [0, 1]$ and $\sum_{k=1}^{K} u_{ik} = 1$ (fuzzy versions);
- $\mathbf{V} = [v_{jh}]_{\substack{1 \leq j \leq P \\ 1 \leq h \leq H}}$ is the (crisp, fuzzy) membership matrix of the variables, where v_{jh} denotes the membership degree of the variable j into h^{th} variable cluster, s.t. either $v_{jh} \in \{0, 1\}$ and $\sum_{h=1}^{H} v_{jh} = 1$ (crisp versions) OR $v_{jh} \in [0, 1]$ and $\sum_{h=1}^{H} v_{jh} = 1$ (fuzzy versions).

Gaussian Kernel functions according to the bandwidth parameter structure

Table 1: Gaussian kernel functions.

ID	Name	Equations		Constraints
1	Gaussian Kernel with Global Width Per Variable	$\mathcal{K}(x_{ij}, g_{kh}\|s_j) = \exp\left\{-\frac{1}{2}\frac{1}{s_j^2}(x_{ij}-g_{kh})^2\right\}$	(1)	$\mathbf{s} = [s_j]_{1\leq j\leq P}$, s.t $\prod_{j=1}^{P}\frac{1}{s_j^2} = 1$
2	Gaussian Kernel with Global Width Per Object	$\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_i) = \exp\left\{-\frac{1}{2}\frac{1}{\mathfrak{s}_i^2}(x_{ij}-g_{kh})^2\right\}$	(2)	$\boldsymbol{\mathfrak{s}} = [\mathfrak{s}_i]_{1\leq i\leq N}$, s.t $\prod_{i=1}^{N}\frac{1}{\mathfrak{s}_i^2} = 1$
3	Gaussian Kernel with Joint Global Width	$\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_i, s_j) = \exp\left\{-\frac{1}{2}\frac{1}{\mathfrak{s}_i^2}\frac{1}{s_j^2}(x_{ij}-g_{kh})^2\right\}$	(3)	$\boldsymbol{\mathfrak{s}} = [\mathfrak{s}_i]_{1\leq i\leq N}$, s.t $\prod_{i=1}^{N}\frac{1}{\mathfrak{s}_i^2} = 1$, and $\mathbf{s} = [s_j]_{1\leq j\leq P}$, s.t $\prod_{j=1}^{P}\frac{1}{s_j^2} = 1$
4	Gaussian Kernel with Local Width Per Variable	$\mathcal{K}(x_{ij}, g_{kh}\|s_{kj}) = \exp\left\{-\frac{1}{2}\frac{1}{s_{kj}^2}(x_{ij}-g_{kh})^2\right\}$	(4)	$\mathbf{S} = [s_{kj}]_{\substack{1\leq k\leq K\\ 1\leq j\leq P}}$, s.t $\prod_{j=1}^{P}\frac{1}{s_{kj}^2} = 1$ $(1\leq k\leq K)$
5	Gaussian Kernel with Local Width Per Object	$\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_{hi}) = \exp\left\{-\frac{1}{2}\frac{1}{\mathfrak{s}_{hi}^2}(x_{ij}-g_{kh})^2\right\}$	(5)	$\boldsymbol{\mathcal{S}} = [\mathfrak{s}_{hi}]_{\substack{1\leq h\leq H\\ 1\leq i\leq N}}$, s.t $\prod_{i=1}^{N}\frac{1}{\mathfrak{s}_{hi}^2} = 1$ $(1\leq h\leq H)$
6	Gaussian Kernel with Joint Local Width	$\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_{hi}, s_{kj}) = \exp\left\{-\frac{1}{2}\frac{1}{\mathfrak{s}_{hi}^2}\frac{1}{s_{kj}^2}(x_{ij}-g_{kh})^2\right\}$	(6)	$\boldsymbol{\mathcal{S}} = [\mathfrak{s}_{hi}]_{\substack{1\leq h\leq H\\ 1\leq i\leq N}}$, s.t $\prod_{i=1}^{N}\frac{1}{\mathfrak{s}_{hi}^2} = 1$ $(1\leq h\leq H)$, and $\mathbf{S} = [s_{kj}]_{\substack{1\leq k\leq K\\ 1\leq j\leq P}}$, s.t $\prod_{j=1}^{P}\frac{1}{s_{kj}^2} = 1$ $(1\leq k\leq K)$

Gaussian Kernel-based hard co-clustering with automatic bandwidth computation (GKHDK)

Table 2: Objective functions for the proposed hard methods.

$$\mathcal{J}_{\text{GKHDK-GWV}} = 2\sum_{k,h,i,j}(u_{ik})\,(v_{jh})\left(1-\mathcal{K}(x_{ij},g_{kh}|s_j^2)\right) \quad (7)$$

$$\mathcal{J}_{\text{GKHDK-GWO}} = 2\sum_{k,h,i,j}(u_{ik})\,(v_{jh})\left(1-\mathcal{K}(x_{ij},g_{kh}|\mathfrak{s}_i^2)\right) \quad (8)$$

$$\mathcal{J}_{\text{GKHDK-JGW}} = 2\sum_{k,h,i,j}(u_{ik})\,(v_{jh})\left(1-\mathcal{K}(x_{ij},g_{kh}|\mathfrak{s}_i^2,s_j^2)\right) \quad (9)$$

$$\mathcal{J}_{\text{GKHDK-LWV}} = 2\sum_{k,h,i,j}(u_{ik})\,(v_{jh})\left(1-\mathcal{K}(x_{ij},g_{kh}|s_{kj}^2)\right) \quad (10)$$

$$\mathcal{J}_{\text{GKHDK-LWO}} = 2\sum_{k,h,i,j}(u_{ik})\,(v_{jh})\left(1-\mathcal{K}(x_{ij},g_{kh}|\mathfrak{s}_{hi}^2)\right) \quad (11)$$

$$\mathcal{J}_{\text{GKHDK-JLW}} = 2\sum_{k,h,i,j}(u_{ik})\,(v_{jh})\left(1-\mathcal{K}(x_{ij},g_{kh}|\mathfrak{s}_{hi}^2,s_{kj}^2)\right) \quad (12)$$

Gaussian Kernel-based hard co-clustering with automatic bandwidth computation (GKHDK)

- It starts from a random initialization of the partitions **U** and **V**
- It iteratively minimizes its objective function with respect to
  - the bandwidth parameters **s** or/and $\mathfrak{s}$ (global methods) or **S** or/and $\boldsymbol{\mathcal{S}}$ (local methods)
  - the prototypes **G**
  - the hard partition of objects **U**;
  - the hard partition of variables **V**.
- until reach a stopping criterion
- During the optimization steps, one component is updated while the others are kept fixed.

Gaussian Kernel-based hard co-clustering with automatic bandwidth computation (GKHDK)

Table 3: Lagrangian functions for the proposed hard methods

$$\mathcal{L}_{\text{GKHDK-GWV}} = 2\sum_{k,h,i,j}(u_{ik})\,(v_{jh})\left(1-\mathcal{K}(x_{ij},g_{kh}|s_j^2)\right) - \rho\left(\prod_{j=1}^{P}\frac{1}{s_j^2}-1\right) \tag{13}$$

$$\mathcal{L}_{\text{GKHDK-GWO}} = 2\sum_{k,h,i,j}(u_{ik})\,(v_{jh})\left(1-\mathcal{K}(x_{ij},g_{kh}|\mathfrak{s}_i^2)\right) - \eta\left(\prod_{i=1}^{N}\frac{1}{\mathfrak{s}_i^2}-1\right) \tag{14}$$

$$\mathcal{L}_{\text{GKHDK-JGW}} = 2\sum_{k,h,i,j}(u_{ik})\,(v_{jh})\left(1-\mathcal{K}(x_{ij},g_{kh}|\mathfrak{s}_i^2,s_j^2)\right) - \rho\left(\prod_{j=1}^{P}\frac{1}{s_j^2}-1\right) - \eta\left(\prod_{i=1}^{N}\frac{1}{\mathfrak{s}_i^2}-1\right) \tag{15}$$

$$\mathcal{L}_{\text{GKHDK-LWV}} = 2\sum_{k,h,i,j}(u_{ik})\,(v_{jh})\left(1-\mathcal{K}(x_{ij},g_{kh}|s_{kj}^2)\right) - \textstyle\sum_k \rho_k\left(\prod_{j=1}^{P}\frac{1}{s_{kj}^2}-1\right) \tag{16}$$

$$\mathcal{L}_{\text{GKHDK-LWO}} = 2\sum_{k,h,i,j}(u_{ik})\,(v_{jh})\left(1-\mathcal{K}(x_{ij},g_{kh}|\mathfrak{s}_{hi}^2)\right) - \textstyle\sum_h \eta_h\left(\prod_{i=1}^{N}\frac{1}{\mathfrak{s}_{hi}^2}-1\right) \tag{17}$$

$$\mathcal{L}_{\text{GKHDK-JLW}} = 2\sum_{k,h,i,j}(u_{ik})\,(v_{jh})\left(1-\mathcal{K}(x_{ij},g_{kh}|\mathfrak{s}_{hi}^2,s_{kj}^2)\right) - \textstyle\sum_k \rho_k\left(\prod_{j=1}^{P}\frac{1}{s_{kj}^2}-1\right) - \sum_h \eta_h\left(\prod_{i=1}^{N}\frac{1}{\mathfrak{s}_{hi}^2}-1\right) \tag{18}$$

Step 1: Computation of the bandwidth

Table 4: Bandwidth hyperparameters for the proposed hard methods.

Algorithm	Width	
GKHDK-GWV	$\frac{1}{s_j^2} = \frac{\left(\prod_{r=1}^{P} D_r\right)^{\frac{1}{P}}}{D_j}$, where $D_j = \sum_{k,h,i}(u_{ik})(v_{jh})\mathcal{K}(x_{ij}, g_{kh}\|s_j^2)(x_{ij} - g_{kh})^2$	(19)
GKHDK-GWO	$\frac{1}{\mathfrak{s}_i^2} = \frac{\left(\prod_{l=1}^{N} D_l'\right)^{\frac{1}{N}}}{D_i'}$, where $D_i' = \sum_{k,h,j}(u_{ik})(v_{jh})\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_i^2)(x_{ij} - g_{kh})^2$	(20)
GKHDK-JGW	$\frac{1}{s_j^2} = \frac{\left(\prod_{r=1}^{P} D_r^2\right)^{\frac{1}{P}}}{D_j^2}$, where $D_j^2 = \sum_{k,h,i}(u_{ik})(v_{jh})\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_i^2, s_j^2)\frac{1}{\mathfrak{s}_i^2}(x_{ij} - g_{kh})^2$	(21)
	$\frac{1}{\mathfrak{s}_i^2} = \frac{\left(\prod_{l=1}^{N} D_l^2\right)^{\frac{1}{N}}}{D_i^2}$, where $D_i^2 = \sum_{k,h,j}(u_{ik})(v_{jh})\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_i^2, s_j^2)\frac{1}{s_j^2}(x_{ij} - g_{kh})^2$	(22)
GKHDK-LWV	$\frac{1}{s_{kj}^2} = \frac{\left(\prod_{r=1}^{P} D_{kr}^2\right)^{\frac{1}{P}}}{D_{kj}^2}$, where $D_{kj}^2 = \sum_{h,i}(u_{ik})(v_{jh})\mathcal{K}(x_{ij}, g_{kh}\|s_{kj}^2)(x_{ij} - g_{kh})^2$	(23)
GKHDK-LWO	$\frac{1}{\mathfrak{s}_{hi}^2} = \frac{\left(\prod_{l=1}^{N} D_{hl}^2\right)^{\frac{1}{N}}}{D_{hi}^2}$, where $D_{hi}^2 = \sum_{k,j}(u_{ik})(v_{jh})\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_{hi}^2)(x_{ij} - g_{kh})^2$	(24)
GKHDK-JLW	$\frac{1}{s_{kj}^2} = \frac{\left(\prod_{r=1}^{P} D_{kr}^2\right)^{\frac{1}{P}}}{D_{kj}^2}$, where $D_{kj}^2 = \sum_{h,i}(u_{ik})(v_{jh})\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_{hi}^2, s_{kj}^2)\frac{1}{\mathfrak{s}_{hi}^2}(x_{ij} - g_{kh})^2$	(25)
	$\frac{1}{\mathfrak{s}_{hi}^2} = \frac{\left(\prod_{l=1}^{N} D_{hl}^2\right)^{\frac{1}{N}}}{D_{hi}^2}$, where $D_{hi}^2 = \sum_{k,j}(u_{ik})(v_{jh})\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_{hi}^2, s_{kj}^2)\frac{1}{s_{kj}^2}(x_{ij} - g_{kh})^2$	(26)

Step 2: Representation

Table 5: Representative values for proposed hard methods.

Algorithm	Prototype	
GKHDK-GWV	$g_{kh} = \frac{\sum_{i,j}(u_{ik})(v_{jh})\frac{1}{s_j^2}\mathcal{K}(x_{ij}, g_{kh}\|s_j^2)x_{ij}}{\sum_{i,j}(u_{ik})(v_{jh})\frac{1}{s_j^2}\mathcal{K}(x_{ij}, g_{kh}\|s_j^2)}$	(27)
GKHDK-GWO	$g_{kh} = \frac{\sum_{i,j}(u_{ik})(v_{jh})\frac{1}{\mathfrak{s}_i^2}\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_i^2)x_{ij}}{\sum_{i,j}(u_{ik})(v_{jh})\frac{1}{\mathfrak{s}_i^2}\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_i^2)}$	(28)
GKHDK-JGW	$g_{kh} = \frac{\sum_{i,j}(u_{ik})(v_{jh})\frac{1}{\mathfrak{s}_i^2}\frac{1}{s_j^2}\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_i^2, s_j^2)x_{ij}}{\sum_{i,j}(u_{ik})(v_{jh})\frac{1}{\mathfrak{s}_i^2}\frac{1}{s_j^2}\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_i^2, s_j^2)}$	(29)
GKHDK-LWV	$g_{kh} = \frac{\sum_{i,j}(u_{ik})(v_{jh})\frac{1}{s_{kj}^2}\mathcal{K}(x_{ij}, g_{kh}\|s_{kj}^2)x_{ij}}{\sum_{i,j}(u_{ik})(v_{jh})\frac{1}{s_{kj}^2}\mathcal{K}(x_{ij}, g_{kh}\|s_{kj}^2)}$	(30)
GKHDK-LWO	$g_{kh} = \frac{\sum_{i,j}(u_{ik})(v_{jh})\frac{1}{\mathfrak{s}_{hi}^2}\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_{hi}^2)x_{ij}}{\sum_{i,j}(u_{ik})(v_{jh})\frac{1}{\mathfrak{s}_{hi}^2}\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_{hi}^2)}$	(31)
GKHDK-JLW	$g_{kh} = \frac{\sum_{i,j}(u_{ik})(v_{jh})\frac{1}{\mathfrak{s}_{hi}^2}\frac{1}{s_{kj}^2}\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_{hi}^2, s_{kj}^2)x_{ij}}{\sum_{i,j}(u_{ik})(v_{jh})\frac{1}{\mathfrak{s}_{hi}^2}\frac{1}{s_{kj}^2}\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_{hi}^2, s_{kj}^2)}$	(32)

Step 3: Object assignment

Table 6: Hard assignment rule of the objects for the proposed methods.

Algorithm	Assignment rule	
GKHDK-GWV	$u_{ik} = \begin{cases} 1, & \text{if } k = \underset{1 \le r \le K}{\operatorname{argmin}} \sum_{h,j} (v_{jh})(1 - \mathcal{K}(x_{ij}, g_{rh} \vert s_j^2)) \\ 0, & \text{otherwise} \end{cases}$	(33)
GKHDK-GWO	$u_{ik} = \begin{cases} 1, & \text{if } k = \underset{1 \le r \le K}{\operatorname{argmin}} \sum_{h,j} (v_{jh})(1 - \mathcal{K}(x_{ij}, g_{rh} \vert \mathfrak{s}_i^2)) \\ 0, & \text{otherwise} \end{cases}$	(34)
GKHDK-JGW	$u_{ik} = \begin{cases} 1, & \text{if } k = \underset{1 \le r \le K}{\operatorname{argmin}} \sum_{h,j} (v_{jh})(1 - \mathcal{K}(x_{ij}, g_{rh} \vert \mathfrak{s}_i^2, s_j^2)) \\ 0, & \text{otherwise} \end{cases}$	(35)
GKHDK-LWV	$u_{ik} = \begin{cases} 1, & \text{if } k = \underset{1 \le r \le K}{\operatorname{argmin}} \sum_{h,j} (v_{jh})(1 - \mathcal{K}(x_{ij}, g_{rh} \vert s_{kj}^2)) \\ 0, & \text{otherwise} \end{cases}$	(36)
GKHDK-LWO	$u_{ik} = \begin{cases} 1, & \text{if } k = \underset{1 \le r \le K}{\operatorname{argmin}} \sum_{h,j} (v_{jh})(1 - \mathcal{K}(x_{ij}, g_{rh} \vert \mathfrak{s}_{hi}^2)) \\ 0, & \text{otherwise} \end{cases}$	(37)
GKHDK-JLW	$u_{ik} = \begin{cases} 1, & \text{if } k = \underset{1 \le r \le K}{\operatorname{argmin}} \sum_{h,j} (v_{jh})(1 - \mathcal{K}(x_{ij}, g_{rh} \vert \mathfrak{s}_{hi}^2, s_{kj}^2)) \\ 0, & \text{otherwise} \end{cases}$	(38)

Step 4: Variable assignment

Table 7: Hard assignment rule of the variables for the prposed methods.

Algorithm	Assignment rule	
GKHDK-GWV	$v_{jh} = \begin{cases} 1, & \text{if } h = \underset{1 \le l \le H}{\operatorname{argmin}} \sum_{k,i} (u_{ik})(1 - \mathcal{K}(x_{ij}, g_{kl} \mid s_j^2)) \\ 0, & \text{otherwise} \end{cases}$	(39)
GKHDK-GWO	$v_{jh} = \begin{cases} 1, & \text{if } h = \underset{1 \le l \le H}{\operatorname{argmin}} \sum_{k,i} (u_{ik})(1 - \mathcal{K}(x_{ij}, g_{kl} \mid \mathfrak{s}_i^2)) \\ 0, & \text{otherwise} \end{cases}$	(40)
GKHDK-JGW	$v_{jh} = \begin{cases} 1, & \text{if } h = \underset{1 \le l \le H}{\operatorname{argmin}} \sum_{k,i} (u_{ik})(1 - \mathcal{K}(x_{ij}, g_{kl} \mid \mathfrak{s}_i^2, s_j^2)) \\ 0, & \text{otherwise} \end{cases}$	(41)
GKHDK-LWV	$v_{jh} = \begin{cases} 1, & \text{if } h = \underset{1 \le l \le H}{\operatorname{argmin}} \sum_{k,i} (u_{ik})(1 - \mathcal{K}(x_{ij}, g_{kl} \mid s_{kj}^2)) \\ 0, & \text{otherwise} \end{cases}$	(42)
GKHDK-LWO	$v_{jh} = \begin{cases} 1, & \text{if } h = \underset{1 \le l \le H}{\operatorname{argmin}} \sum_{k,i} (u_{ik})(1 - \mathcal{K}(x_{ij}, g_{kl} \mid \mathfrak{s}_{hi}^2)) \\ 0, & \text{otherwise} \end{cases}$	(43)
GKHDK-JLW	$v_{jh} = \begin{cases} 1, & \text{if } h = \underset{1 \le l \le H}{\operatorname{argmin}} \sum_{k,i} (u_{ik})(1 - \mathcal{K}(x_{ij}, g_{kl} \mid \mathfrak{s}_{hi}^2, s_{kj}^2)) \\ 0, & \text{otherwise} \end{cases}$	(44)

Algorithm 1 Hard co-clustering algorithms with automatic bandwidth computation

1: **Input**
2: **X** (the dataset); K (the number of object clusters); H (the number of variable clusters); T (maximum number of iterations).
3: **Output**
4: **U** (the hard partition of objects);
5: **V** (the hard partition of variables);
6: **G** (the matrix $K \times H$ with the prototypes of each block);
7: **s** or/and $\mathfrak{s}$ (the vector of bandwidth parameters for the global methods), and **S** or/and $\boldsymbol{\mathcal{S}}$ (the matrix of bandwidth parameters for the local methods).
8: **Initialization**
9: $t \leftarrow 0$
10: $\mathbf{s}^{(0)}$: set $\frac{1}{s_j^2} := [1]_{1 \leq j \leq P}$ or/and $\mathfrak{s}^{(0)}$: $\frac{1}{\mathfrak{s}_i^2} := [1]_{1 \leq i \leq N}$ or
$\mathbf{S}^{(0)}$: set $\frac{1}{s_{kj}^2} := [1]_{\substack{1 \leq k \leq K \\ 1 \leq j \leq P}}$ or/and $\boldsymbol{\mathcal{S}}^{(0)}$: $\frac{1}{\mathfrak{s}_{hi}^2} := [1]_{\substack{1 \leq h \leq H \\ 1 \leq i \leq N}}$;
11: $\mathbf{U}^{(0)}$: randomly assign each object i to a object cluster k $\forall k$;
12: $\mathbf{V}^{(0)}$: randomly assign each variable j to a variable cluster h $\forall h$;
13: $\mathbf{G}^{(0)}$: initialize the prototypes g_{kh} $\forall k, h$, as:

$$g_{kh}^{(0)} = \frac{\sum_{i,j} (u_{ik}^{(0)})(v_{jh}^{(0)}) x_{ij}}{\left(\sum_i u_{ik}^{(0)}\right)\left(\sum_j v_{jh}^{(0)}\right)},$$

repeat
  $t \leftarrow t + 1$

  Step 1: Computation of the bandwidth parameters

  Compute the global bandwidth parameters $\mathbf{s}^{(t)}$ and $\boldsymbol{\mathfrak{s}}^{(t)}$ or the local bandwidth hyperparameters $\mathbf{S}^{(t)}$ and $\boldsymbol{\mathcal{S}}^{(t)}$ according to the Table 4

  Step 2: Representation

  Compute the prototype matrix $\mathbf{G}^{(t)}$ according to the Table 5

  Step 3: Object assignment

  Update the hard membership matrix of the objects $\mathbf{U}^{(t)}$ according to the Table 6

  Step 4: Variable assignment

  Update the hard membership matrix of the variables $\mathbf{V}^{(t)}$ according to the Table 7

until $\mathbf{U}^{(t-1)} = \mathbf{U}^{(t)}$ and $\mathbf{V}^{(t-1)} = \mathbf{V}^{(t)}$ or $t > T$

Gaussian Kernel-based fuzzy co-clustering with automatic bandwidth computation (GKFDK)

Table 8: Objective functions for the proposed fuzzy methods.

$$\mathcal{J}_{\text{GKFDK-GWV}} = 2\sum_{k,h,i,j}(u_{ik})^m(v_{jh})^n\left(1-\mathcal{K}(x_{ij},g_{kh}|s_j^2)\right) \quad (45)$$

$$\mathcal{J}_{\text{GKFDK-GWO}} = 2\sum_{k,h,i,j}(u_{ik})^m(v_{jh})^n\left(1-\mathcal{K}(x_{ij},g_{kh}|\mathfrak{s}_i^2)\right) \quad (46)$$

$$\mathcal{J}_{\text{GKFDK-JGW}} = 2\sum_{k,h,i,j}(u_{ik})^m(v_{jh})^n\left(1-\mathcal{K}(x_{ij},g_{kh}|\mathfrak{s}_i^2,s_j^2)\right) \quad (47)$$

$$\mathcal{J}_{\text{GKFDK-LWV}} = 2\sum_{k,h,i,j}(u_{ik})^m(v_{jh})^n\left(1-\mathcal{K}(x_{ij},g_{kh}|s_{kj}^2)\right) \quad (48)$$

$$\mathcal{J}_{\text{GKFDK-LWO}} = 2\sum_{k,h,i,j}(u_{ik})^m(v_{jh})^n\left(1-\mathcal{K}(x_{ij},g_{kh}|\mathfrak{s}_{hi}^2)\right) \quad (49)$$

$$\mathcal{J}_{\text{GKFDK-JLW}} = 2\sum_{k,h,i,j}(u_{ik})^m(v_{jh})^n\left(1-\mathcal{K}(x_{ij},g_{kh}|\mathfrak{s}_{hi}^2,s_{kj}^2)\right) \quad (50)$$

Gaussian Kernel-based fuzzy co-clustering with automatic bandwidth computation (GKFDK)

- It starts from a random initialization of the partitions **U** and **V**
- It iteratively minimizes its objective function with respect to
  - the bandwidth parameters **s** or/and $\mathfrak{s}$ (global methods) or **S** or/and $\boldsymbol{\mathcal{S}}$ (local methods)
  - the prototypes **G**
  - the fuzzy partition of objects **U**;
  - the fuzzy partition of variables **V**.
- until reach a stopping criterion
- During the optimization steps, one component is updated while the others are kept fixed.

Gaussian Kernel-based fuzzy co-clustering with automatic bandwidth computation (GKFDK)

Table 9: Lagrangian functions for the proposed fuzzy methods

$$\mathcal{L}_{\text{GKHDK-GWV}} = 2\sum_{k,h,i,j}(u_{ik})^m\,(v_{jh})^n\left(1-\mathcal{K}(x_{ij},g_{kh}|s_j^2)\right) - \rho\left(\prod_{j=1}^{P}\frac{1}{s_j^2}-1\right) - \theta\left(\sum_k u_{ik}-1\right) - \vartheta\left(\sum_h v_{jh}-1\right) \quad (51)$$

$$\mathcal{L}_{\text{GKHDK-GWO}} = 2\sum_{k,h,i,j}(u_{ik})^m\,(v_{jh})^n\left(1-\mathcal{K}(x_{ij},g_{kh}|\mathfrak{s}_i^2)\right) - \eta\left(\prod_{i=1}^{N}\frac{1}{\mathfrak{s}_i^2}-1\right) - \theta\left(\sum_k u_{ik}-1\right) - \vartheta\left(\sum_h v_{jh}-1\right) \quad (52)$$

$$\mathcal{L}_{\text{GKHDK-JGW}} = 2\sum_{k,h,i,j}(u_{ik})^m\,(v_{jh})^n\left(1-\mathcal{K}(x_{ij},g_{kh}|\mathfrak{s}_i^2,s_j^2)\right) - \rho\left(\prod_{j=1}^{P}\frac{1}{s_j^2}-1\right) - \eta\left(\prod_{i=1}^{N}\frac{1}{\mathfrak{s}_i^2}-1\right) - \theta\left(\sum_k u_{ik}-1\right) - \vartheta\left(\sum_h v_{jh}-1\right) \quad (53)$$

$$\mathcal{L}_{\text{GKHDK-LWV}} = 2\sum_{k,h,i,j}(u_{ik})^m\,(v_{jh})^n\left(1-\mathcal{K}(x_{ij},g_{kh}|s_{kj}^2)\right) - \sum_k \rho_k\left(\prod_{j=1}^{P}\frac{1}{s_{kj}^2}-1\right) - \theta\left(\sum_k u_{ik}-1\right) - \vartheta\left(\sum_h v_{jh}-1\right) \quad (54)$$

$$\mathcal{L}_{\text{GKHDK-LWO}} = 2\sum_{k,h,i,j}(u_{ik})^m\,(v_{jh})^n\left(1-\mathcal{K}(x_{ij},g_{kh}|\mathfrak{s}_{hi}^2)\right) - \sum_h \eta_h\left(\prod_{i=1}^{N}\frac{1}{\mathfrak{s}_{hi}^2}-1\right) - \theta\left(\sum_k u_{ik}-1\right) - \vartheta\left(\sum_h v_{jh}-1\right) \quad (55)$$

$$\mathcal{L}_{\text{GKHDK-JLW}} = 2\sum_{k,h,i,j}(u_{ik})^m\,(v_{jh})^n\left(1-\mathcal{K}(x_{ij},g_{kh}|\mathfrak{s}_{hi}^2,s_{kj}^2)\right) - \sum_k \rho_k\left(\prod_{j=1}^{P}\frac{1}{s_{kj}^2}-1\right) - \sum_h \eta_h\left(\prod_{i=1}^{N}\frac{1}{\mathfrak{s}_{hi}^2}-1\right) - \theta\left(\sum_k u_{ik}-1\right) - \vartheta\left(\sum_h v_{jh}-1\right) \quad (56)$$

Step 1: Computation of the bandwidth

Table 10: Width hyperparameters for the proposed fuzzy methods.

Algorithm	Width	
GKFDK-GWV	$\frac{1}{s_j^2} = \frac{\left(\prod_{r=1}^{P} D_r^2\right)^{\frac{1}{P}}}{D_j^2}$, where $D_j^2 = \sum_{k,h,i} (u_{ik})^m (v_{jh})^n \mathcal{K}(x_{ij}, g_{kh} \vert s_j^2)(x_{ij} - g_{kh})^2$	(57)
GKFDK-GWO	$\frac{1}{\mathfrak{s}_i^2} = \frac{\left(\prod_{l=1}^{N} D_l^2\right)^{\frac{1}{N}}}{D_i^2}$, where $D_i^2 = \sum_{k,h,j} (u_{ik})^m (v_{jh})^n \mathcal{K}(x_{ij}, g_{kh} \vert \mathfrak{s}_i^2)(x_{ij} - g_{kh})^2$	(58)
GKFDK-JGW	$\frac{1}{s_j^2} = \frac{\left(\prod_{r=1}^{P} D_r^2\right)^{\frac{1}{P}}}{D_j^2}$, where $D_j^2 = \sum_{k,h,i} (u_{ik})^m (v_{jh})^n \mathcal{K}(x_{ij}, g_{kh} \vert \mathfrak{s}_i^2, s_j^2) \frac{1}{\mathfrak{s}_i^2}(x_{ij} - g_{kh})^2$	(59)
	$\frac{1}{\mathfrak{s}_i^2} = \frac{\left(\prod_{l=1}^{N} D_l^2\right)^{\frac{1}{N}}}{D_i^2}$, where $D_i^2 = \sum_{k,h,j} (u_{ik})^m (v_{jh})^n \mathcal{K}(x_{ij}, g_{kh} \vert \mathfrak{s}_i^2, s_j^2) \frac{1}{s_j^2}(x_{ij} - g_{kh})^2$	(60)
GKFDK-LWV	$\frac{1}{s_{kj}^2} = \frac{\left(\prod_{r=1}^{P} D_{kr}^2\right)^{\frac{1}{P}}}{D_{kj}^2}$, where $D_{kj}^2 = \sum_{h,i} (u_{ik})^m (v_{jh})^n \mathcal{K}(x_{ij}, g_{kh} \vert s_{kj}^2)(x_{ij} - g_{kh})^2$	(61)
GKFDK-LWO	$\frac{1}{\mathfrak{s}_{hi}^2} = \frac{\left(\prod_{l=1}^{N} D_{hl}^2\right)^{\frac{1}{N}}}{D_{hi}^2}$, where $D_{hi}^2 = \sum_{k,j} (u_{ik})^m (v_{jh})^n \mathcal{K}(x_{ij}, g_{kh} \vert \mathfrak{s}_{hi}^2)(x_{ij} - g_{kh})^2$	(62)
GKFDK-JLW	$\frac{1}{s_{kj}^2} = \frac{\left(\prod_{r=1}^{P} D_{kr}^2\right)^{\frac{1}{P}}}{D_{kj}^2}$, where $D_{kj}^2 = \sum_{h,i} (u_{ik})^m (v_{jh})^n \mathcal{K}(x_{ij}, g_{kh} \vert \mathfrak{s}_{hi}^2, s_{kj}^2) \frac{1}{\mathfrak{s}_{hi}^2}(x_{ij} - g_{kh})^2$	(63)
	$\frac{1}{\mathfrak{s}_{hi}^2} = \frac{\left(\prod_{l=1}^{N} D_{hl}^2\right)^{\frac{1}{N}}}{D_{hi}^2}$, where $D_{hi}^2 = \sum_{k,j} (u_{ik})^m (v_{jh})^n \mathcal{K}(x_{ij}, g_{kh} \vert \mathfrak{s}_{hi}^2, s_{kj}^2) \frac{1}{s_{kj}^2}(x_{ij} - g_{kh})^2$	(64)

Step 2: Representation

Table 11: Representative values for the proposed fuzzy methods.

Algorithm	Prototype	
GKFDK-GWV	$g_{kh} = \frac{\sum_{i,j}(u_{ik})^m (v_{jh})^n \frac{1}{s_j^2}\mathcal{K}(x_{ij}, g_{kh}\|s_j^2)x_{ij}}{\sum_{i,j}(u_{ik})^m (v_{jh})^n \frac{1}{s_j^2}\mathcal{K}(x_{ij}, g_{kh}\|s_j^2)}$	(65)
GKFDK-GWO	$g_{kh} = \frac{\sum_{i,j}(u_{ik})^m (v_{jh})^n \frac{1}{\mathfrak{s}_i^2}\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_i^2)x_{ij}}{\sum_{i,j}(u_{ik})^m (v_{jh})^n \frac{1}{\mathfrak{s}_i^2}\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_i^2)}$	(66)
GKFDK-JGW	$g_{kh} = \frac{\sum_{i,j}(u_{ik})^m (v_{jh})^n \frac{1}{\mathfrak{s}_i^2}\frac{1}{s_j^2}\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_i^2, s_j^2)x_{ij}}{\sum_{i,j}(u_{ik})^m (v_{jh})^n \frac{1}{\mathfrak{s}_i^2}\frac{1}{s_j^2}\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_i^2, s_j^2)}$	(67)
GKFDK-LWV	$g_{kh} = \frac{\sum_{i,j}(u_{ik})^m (v_{jh})^n \frac{1}{s_{kj}^2}\mathcal{K}(x_{ij}, g_{kh}\|s_{kj}^2)x_{ij}}{\sum_{i,j}(u_{ik})^m (v_{jh})^n \frac{1}{s_{kj}^2}\mathcal{K}(x_{ij}, g_{kh}\|s_{kj}^2)}$	(68)
GKFDK-LWO	$g_{kh} = \frac{\sum_{i,j}(u_{ik})^m (v_{jh})^n \frac{1}{\mathfrak{s}_{hi}^2}\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_{hi}^2)x_{ij}}{\sum_{i,j}(u_{ik})^m (v_{jh})^n \frac{1}{\mathfrak{s}_{hi}^2}\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_{hi}^2)}$	(69)
GKFDK-JGW	$g_{kh} = \frac{\sum_{i,j}(u_{ik})^m (v_{jh})^n \frac{1}{\mathfrak{s}_{hi}^2}\frac{1}{s_{kj}^2}\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_{hi}^2, s_{kj}^2)x_{ij}}{\sum_{i,j}(u_{ik})^m (v_{jh})^n \frac{1}{\mathfrak{s}_{hi}^2}\frac{1}{s_{kj}^2}\mathcal{K}(x_{ij}, g_{kh}\|\mathfrak{s}_{hi}^2, s_{kj}^2)}$	(70)

Step 3: Object assignment

The membership degree u_{ik} $(1 \leq i \leq N;\ 1 \leq k \leq K)$ is updated as follows

$$u_{ik} = \left[\sum_{r=1}^{K} \left(\frac{D_{ik}}{D_{ir}} \right)^{\frac{1}{m-1}} \right]^{-1}, \tag{71}$$

where D_{ik} is a distance term, computed according to the Table 12.

Table 12: Distance terms used to compute the fuzzy membership of the objects for the proposed methods.

Algorithm	Distance term	
GKFDK-GWV	$D_{ik} = \sum_{h,j} (v_{jh})^n (2 - 2\mathcal{K}(x_{ij}, g_{kh} \vert s_j^2))$	(72)
GKFDK-GWO	$D_{ik} = \sum_{h,j} (v_{jh})^n (2 - 2\mathcal{K}(x_{ij}, g_{kh} \vert \mathfrak{s}_i^2))$	(73)
GKFDK-JGW	$D_{ik} = \sum_{h,j} (v_{jh})^n (2 - 2\mathcal{K}(x_{ij}, g_{kh} \vert \mathfrak{s}_i^2, s_j^2))$	(74)
GKFDK-LWV	$D_{ik} = \sum_{h,j} (v_{jh})^n (2 - 2\mathcal{K}(x_{ij}, g_{kh} \vert s_{kj}^2))$	(75)
GKFDK-LWO	$D_{ik} = \sum_{h,j} (v_{jh})^n (2 - 2\mathcal{K}(x_{ij}, g_{kh} \vert \mathfrak{s}_{hi}^2))$	(76)
GKFDK-JLW	$D_{ik} = \sum_{h,j} (v_{jh})^n (2 - 2\mathcal{K}(x_{ij}, g_{kh} \vert \mathfrak{s}_{hi}^2, s_{kj}^2))$	(77)

Step 4: Variable assignment

The membership degree v_{jh} $(1 \leq j \leq P;\ 1 \leq h \leq H)$ is updated as follows

$$v_{jh} = \left[\sum_{l=1}^{H} \left(\frac{D'_{jh}}{D'_{jl}} \right)^{\frac{1}{n-1}} \right]^{-1}, \tag{78}$$

where D'_{jh} is a distance term, computed according to the Table 13.

Table 13: Distance terms used to compute the fuzzy membership of the variables for the proposed methods.

Algorithm	Distance term	
GKFDK-GWV	$D'_{jh} = \sum_{k,i} (u_{ik})^m (2 - 2\mathcal{K}(x_{ij}, g_{kh} \vert s_j^2))$	(79)
GKFDK-GWO	$D'_{jh} = \sum_{k,i} (u_{ik})^m (2 - 2\mathcal{K}(x_{ij}, g_{kh} \vert \mathfrak{s}_i^2))$	(80)
GKFDK-JGW	$D'_{jh} = \sum_{k,i} (u_{ik})^m (2 - 2\mathcal{K}(x_{ij}, g_{kh} \vert \mathfrak{s}_i^2, s_j^2))$	(81)
GKFDK-LWV	$D'_{jh} = \sum_{k,i} (u_{ik})^m (2 - 2\mathcal{K}(x_{ij}, g_{kh} \vert s_{kj}^2))$	(82)
GKFDK-LWO	$D'_{jh} = \sum_{k,i} (u_{ik})^m (2 - 2\mathcal{K}(x_{ij}, g_{kh} \vert \mathfrak{s}_{hi}^2))$	(83)
GKFDK-JLW	$D'_{jh} = \sum_{k,i} (u_{ik})^m (2 - 2\mathcal{K}(x_{ij}, g_{kh} \vert \mathfrak{s}_{hi}^2, s_{kj}^2))$	(84)

Algorithm 2 Fuzzy co-clustering algorithms with automatic bandwidth computation.

1: **Input**
2: **X** (the dataset); K (the number of object clusters); H (the number of variable clusters); m and n (fuzzifier parameters); T (maximum number of iterations); ε (threshold parameter).
3: **Output**
4: **U** (the fuzzy partition of objects);
5: **V** (the fuzzy partition of variables);
6: **G** (the matrix $K \times H$ with the prototypes of each block);
7: **s** or/and $\boldsymbol{\mathfrak{s}}$ (the vector of bandwidth parameters for the global methods), and **S** or/and $\boldsymbol{\mathcal{S}}$ (the matrix of bandwidth parameters for the local methods).
8: **Initialization**
9: $t \leftarrow 0$
10: $\mathbf{s}^{(0)}$: set $\frac{1}{s_j^2} := [1]_{1 \leq j \leq P}$ or/and $\boldsymbol{\mathfrak{s}}^{(0)}$: $\frac{1}{\mathfrak{s}_i^2} := [1]_{1 \leq i \leq N}$ or
$\mathbf{S}^{(0)}$: set $\frac{1}{s_{kj}^2} := [1]_{\substack{1 \leq k \leq K \\ 1 \leq j \leq P}}$ or/and $\boldsymbol{\mathcal{S}}^{(0)}$: $\frac{1}{\mathfrak{s}_{hi}^2} := [1]_{\substack{1 \leq h \leq H \\ 1 \leq i \leq N}}$;
11: $\mathbf{U}^{(0)}$: initialize randomly $\mathbf{U}^{(0)}$ according to the restrictions imposed.
12: $\mathbf{V}^{(0)}$: initialize randomly $\mathbf{V}^{(0)}$ according to the restrictions imposed.
13: $\mathbf{G}^{(0)}$: initialize the prototypes g_{kh} $\forall$ k, h, as:

$$g_{kh}^{(0)} = \frac{\sum_{i,j} (u_{ik}^{(0)})^m (v_{jh}^{(0)})^n x_{ij}}{\sum_{i,j} (u_{ik}^{(0)})^m (v_{jh}^{(0)})^n}$$

14: Compute $\mathcal{J}_{\text{NEW}}$ according to Table 8.

1: **repeat**
2: $t \leftarrow t + 1$
3:
4: **Step 1: Computation of the bandwidth hyperparameters**
5: Compute the global bandwidth parameters $\mathbf{s}^{(t)}$ and $\boldsymbol{\mathfrak{s}}^{(t)}$ or the local bandwidth hyperparameters $\mathbf{S}^{(t)}$ and $\boldsymbol{\mathcal{S}}^{(t)}$ according to the Table 10
6:
7: **Step 2: Representation**
8: Compute the prototype matrix $\mathbf{G}^{(t)}$ according to the Table 11
9:
10: **Step 3: Object assignment** Update the fuzzy membership matrix of the objects $\mathbf{U}^{(t)}$ according to the Table 12
11:
12: **Step 4: Variable assignment**
13: Update the fuzzy membership matrix of the variables $\mathbf{V}^{(t)}$ according to the Table 13
14: $\mathcal{J}_{\mathrm{OLD}} \leftarrow \mathcal{J}_{\mathrm{NEW}}$
Compute $\mathcal{J}_{\mathrm{NEW}}$ according to Table 8.
15:
16: **until** $|\mathcal{J}_{\mathrm{NEW}} - \mathcal{J}_{\mathrm{OLD}}| \leq \varepsilon$ or $t > T$.

Experimental evaluation

- Experimental trials with datasets from public repositories were performed;
- Metrics

Metric	Acronim	Type	Interval
Adjusted Rand Index [15]	ARI	Hard	[-1, 1]
Accuracy [29]	ACC	Hard	[0, 1]
Normalized Mutual Information [28]	NMI	Hard	[0, 1]
F-Measure [3]	FM	Hard	[0, 1]
Hullermeier Index [16]	HUL	Fuzzy	[0, 1]

- The higher the values of the metrics, the better the results
- From a fuzzy partition $\mathbf{U} = [u_{ik}]_{\substack{1 \leq i \leq N \\ 1 \leq k \leq K}}$, a crisp partition $\mathcal{Z} = \{\mathcal{Z}_1, ..., \mathcal{Z}_K\}$ can be obtained as follows:

$$\mathcal{Z}_k = \{i \in \{1, ..., N\} : u_{ik} = \max_{1 \leq r \leq K} u_{ir}\} \quad (1 \leq k \leq K)$$

State of art methods

Table 14: Characteristics of the state-of-the-art methods compared in this work.

Methods	Co-clustering ?	Output Partition	Kernel approach ?	Automatic width computation ?	Time complexity
KM [19]	No	Hard	No	No	$\mathcal{O}(TKNP)$
KKM [23]	No	Hard	Yes	No	$\mathcal{O}(TKNP)$
KCM-K-GH [7]	No	Hard	Yes	Yes	$\mathcal{O}(TKNP)$
KCM-K-LH [7]	No	Hard	Yes	Yes	$\mathcal{O}(TKNP)$
FCM [4]	No	Fuzzy	No	No	$\mathcal{O}(TKNP)$
KFCM [13]	No	Fuzzy	Yes	No	$\mathcal{O}(TKNP)$
KFCM-K-H [6]	No	Fuzzy	Yes	Yes	$\mathcal{O}(TKNP)$
KFCM-K-W.2 [27]	No	Fuzzy	Yes	Yes	$\mathcal{O}(TKNP)$
DK [21]	Yes	Hard	No	No	$\mathcal{O}(TKHNP)$
CCMOD [1]	Yes	Hard	No	No	$\mathcal{O}(TKNP)$
SCC [8]	Yes	Hard	No	No	$\mathcal{O}(TNP\ \mathrm{LOG}\{R\})$
SBC [17]	Yes	Hard	No	No	$\mathcal{O}(TNP\ \mathrm{LOG}\{R\})$
FDK [18]	Yes	Fuzzy	No	No	$\mathcal{O}(TKHNP)$
WFDK [18]	Yes	Fuzzy	No	No	$\mathcal{O}(TKHNP)$
GKFDK [22]	Yes	Fuzzy	Yes	No	$\mathcal{O}(TKHNP)$
WGKFDK [22]	Yes	Fuzzy	Yes	No	$\mathcal{O}(TKHNP)$
GKHDK-GWV (ours)	Yes	Hard	Yes	Yes	$\mathcal{O}(TKHNP)$
GKHDK-GWO (ours)	Yes	Hard	Yes	Yes	$\mathcal{O}(TKHNP)$
GKHDK-JGW (ours)	Yes	Hard	Yes	Yes	$\mathcal{O}(TKHNP)$
GKHDK-LWV (ours)	Yes	Hard	Yes	Yes	$\mathcal{O}(TKHNP)$
GKHDK-LWO (ours)	Yes	Hard	Yes	Yes	$\mathcal{O}(TKHNP)$
GKHDK-JLW (ours)	Yes	Hard	Yes	Yes	$\mathcal{O}(TKHNP)$
GKFDK-GWV (ours)	Yes	Fuzzy	Yes	Yes	$\mathcal{O}(TKHNP)$
GKFDK-GWO (ours)	Yes	Fuzzy	Yes	Yes	$\mathcal{O}(TKHNP)$
GKFDK-JGW (ours)	Yes	Fuzzy	Yes	Yes	$\mathcal{O}(TKHNP)$
GKFDK-LWV (ours)	Yes	Fuzzy	Yes	Yes	$\mathcal{O}(TKHNP)$
GKFDK-LWO (ours)	Yes	Fuzzy	Yes	Yes	$\mathcal{O}(TKHNP)$
GKFDK-JLW (ours)	Yes	Fuzzy	Yes	Yes	$\mathcal{O}(TKHNP)$

K, H, N, and P are defined as before, T represents the number of iterations to stabilize the algorithms, and R is the number of eigenvalues.

Hyperparameter setting

- the hyper parameter σ^2 of the algorithms KKM and KFCM was estimated following Caputo [5].
  - $\sigma^2 = (Q_{10} + Q_{90})/4$, where Q_{10} e Q_{90} are, respectively, 0.1 and 0.9 de $||\mathbf{x}_l - \mathbf{x}_r||^2$, $l \neq r$.
- the hyper parameter σ^2 of the algorithms GKFDK e WGKFDK was estimated following Caputo [5].
  - $\sigma^2 = (\mathcal{Q}_{10} + \mathcal{Q}_{90})/4$, where $\mathcal{Q}_{10}$ e $\mathcal{Q}_{90}$ are, respectively,the quantis 0.1 and 0.9 of $(x_{ab} - x_{cd})^2$, $a \neq c$ and $b \neq d$.
- the hyper parameter γ of the methods KCM-K-GH e KCM-K-LH were estimated following Caputo [5] as indicated by the authors.
- Regarding the methdos KFCM-K-H and KFCM-K-W.2, was defined $\gamma = 1$, following the authors.

Hyperparameter setting

- The parameters m, n (fuzzy algorithms) and γ (WFDK) were selected using a Bayesian optimization with the Optuna library [2];
  - The process explores promising regions using the Tree-Structured Parzen Estimator (TPE) to maximize expected improvement according to a pre-defined metric;
  - Metric used: minimum distance between the prototypes of the blocks.
- Number of evaluated configurations: 100;
- Clustering algorithms: $m \in [1.1, 3.0]$;
- Co-clustering algorithms: $m, n \in [1.001, 1.5]$;
- γ hyper parameter of the algorithm WFDK: $\gamma \in [1.001, 1.5]$.

Datasets

- 14 datasets;

Table 15: Summary of the datasets.

Datasets	N	P	C	Source	Datasets	N	P	C	Source
WDBC	569	30	2	UCI	Colposcopy	287	62	2	UCI
Breast Tissue	106	9	6	UCI	Urban	675	147	9	UCI
Vehicle	846	18	4	UCI	Yale	165	1024	15	Scikit-Feature
Abalone	4177	8	3	UCI	Prostate Ge	102	5966	2	Scikit-Feature
Algerian Forest	244	10	2	UCI	Glioma	50	4434	4	Scikit-Feature
Sonar	208	60	2	UCI	Fetal Health	2126	21	3	Kaggle
Vertebral Column	310	6	2	UCI	Rukiln	27	10	2	Ref. [26]

- All datasets were normalized using the average and standard deviation of the variables;
- Each algorithm was executed 100 times with different initializations.
- Two outputs:
  - Average case;
  - Best case (according to the value of the objective function).

Average solution analysis on real datasets

- The number of object clusters was set as the true number of classes (C);
- We took $K = H$;
- All data values were normalized based on the mean and standard deviation of the variables;
- Each algorithm was run 100 times with different initializations, and we report the average value over these 100 runs
- The maximum number of iterations was fixed as 100 ($T = 100$);
- The convergence hyper parameter ε of the algorithms CCMOD, SCC, SBC and of the fuzzy methods was fixed as 10^{-5}.

Average case - average ranking

	Method	ARI	ACC	HUL
State of art	KM	12.5 (10)	12.93 (10)	–
	KKM	12.43 (9)	13.14 (12)	–
	KCM-K-GH	18.50 (25)	17.29 (22)	–
	KCM-K-LH	17.00 (20)	16.79 (21)	–
	FCM	13.57 (13)	12.71 (9)	6.07 (3)
	KFCM	11.79 (8)	12.07 (7)	5.43 (2)
	KFCM-K-H	19.86 (27)	18.86 (26)	11.29 (13)
	KFCM-K-W.2	17.00 (20)	17.93 (23)	11.29 (13)
	DK	15.14 (18)	14.29 (15)	–
	CCMOD	17.86 (23)	18.79 (25)	–
	SCC	19.93 (28)	19.43 (28)	–
	SBC	19.21 (26)	18.93 (27)	–
	FDK	17.29 (22)	15.79 (18)	7.71 (11)
	WFDK	18.29 (24)	18.0 (24)	9.71 (12)
	GKFDK	15.07 (17)	15.36 (17)	6.86 (7)
	WGKFDK	10.93 (5)	8.71 (2)	6.36 (5)
Proposed algorithms	GKHDK-GWV	**7.86 (1)**	**7.93 (1)**	–
	GKHDK-GWO	11.64 (7)	13.71 (14)	–
	GKHDK-JGW	9.00 (2)	11.64 (6)	–
	GKHDK-LWV	9.79 (3)	9.07 (3)	–
	GKHDK-LWO	14.36 (16)	16.21 (20)	–
	GKHDK-JLW	13.50 (12)	13.14 (12)	–
	GKFDK-GWV	13.14 (11)	11.50 (5)	6.36 (5)
	GKFDK-GWO	13.57 (13)	14.93 (16)	7.00 (8)
	GKFDK-JGW	10.79 (4)	12.36 (8)	6.21 (4)
	GKFDK-LWV	13.57 (13)	13.07 (11)	7.14 (9)
	GKFDK-LWO	15.64 (19)	16.14 (19)	7.14 (9)
	GKFDK-JLW	11.00 (6)	11.36 (4)	**5.29 (1)**

- The propose method GKHDK-GWV outperformed all the others algorithms;
- The variants GKHDK-LWV, GKHDK-JGW, GKFDK-JGW e GKFDK-JLW had a good performance;
- The proposed methods outperformed the spectral co-clustering methods (CCMOD, SCC and SBC) and the algorithm WFDK.
- The proposed methods also outperformed the algorithms that are able to automatically compute the bandwidth (KCM-K-GH, KCM-K-LH, KFCM-K-H e KFCM-K-W.2).
- Regarding the metric HUL, the proposed method GKFDK-JLW presented the best result;
- The most competitive algorithm among the state of art methods was the WGKFDK.

Best solution analysis on real datasets

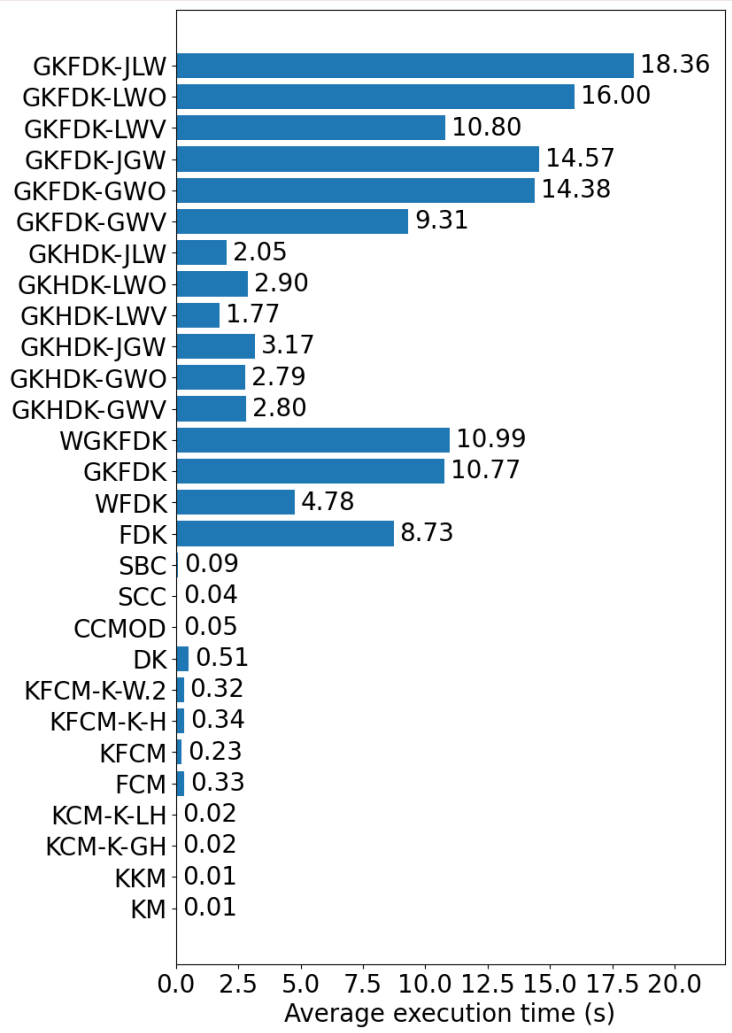
- The number of object clusters was set as the true number of classes (C);
- We took $K = H$;
- All data values were normalized based on the mean and standard deviation of the variables;
- Each algorithm was run 100 times and the best result was reported according to the objective function (lowest value)

Best case - average ranking

	Method	ARI	ACC	HUL
State of art	KM	15.64 (19)	13.5 (14)	–
	KKM	13.29 (11)	12.07 (11)	–
	KCM-K-GH	18.79 (27)	19.29 (28)	–
	KCM-K-LH	17.29 (24)	18.86 (27)	–
	FCM	13.71 (13)	12.14 (12)	6.93 (8)
	KFCM	12.29 (9)	11.21 (6)	**5.79 (1)**
	KFCM-K-H	20.00 (28)	18.43 (26)	11.5 (14)
	KFCM-K-W.2	16.64 (22)	17.86 (23)	10.93 (13)
	DK	15.86 (20)	15.29 (18)	–
	CCMOD	16.50 (21)	17.57 (21)	–
	SCC	18.29 (26)	18.29 (25)	–
	SBC	17.64 (25)	18.07 (24)	–
	FDK	15.43 (17)	13.86 (15)	7.57 (11)
	WFDK	16.86 (23)	17.57 (21)	8.43 (12)
	GKFDK	13.86 (15)	14.36 (16)	7.07 (9)
	WGKFDK	11.43 (5)	9.71 (3)	6.21 (4)
Proposed algorithms	GKHDK-GWV	**7.86 (1)**	**8.79 (1)**	–
	GKHDK-GWO	11.07 (4)	11.86 (10)	–
	GKHDK-JGW	8.79 (2)	10.93 (5)	–
	GKHDK-LWV	13.43 (12)	11.29 (7)	–
	GKHDK-LWO	14.79 (16)	15.64 (19)	–
	GKHDK-JLW	10.14 (3)	9.57 (2)	–
	GKFDK-GWV	12.14 (7)	9.71 (3)	5.86 (2)
	GKFDK-GWO	11.93 (6)	15.14 (17)	6.5 (5)
	GKFDK-JGW	12.21 (8)	13.14 (13)	5.93 (3)
	GKFDK-LWV	13.79 (14)	11.29 (7)	6.86 (7)
	GKFDK-LWO	15.50 (18)	16.50 (20)	7.5 (10)
	GKFDK-JLW	12.93 (10)	11.57 (9)	6.71 (6)

- The proposed algorithm GKHDK-GWV had the best performance.
- The variants GKHDK-JGW, GKFDK-GWV and GKHDK-JLW had good results.
- The proposed methods outperformed the spectral co-clustering methods (CCMOD, SCC and SBC) and the algorithm WFDK.
- The proposed methods also outperformed the algorithms that are able to automatically compute the bandwidth (KCM-K-GH, KCM-K-LH, KFCM-K-H e KFCM-K-W.2).
- The method KFCM presented the best performance regarding the metric HUL.

Average time of execution



- In terms of execution time, KM, KKM, KCM-K-GH, and KCM-K-LH outperformed the other algorithms.
- The proposed hard methods were much faster than the corresponding proposed fuzzy methods.
- The proposed variant GKFDK-JLW had the greatest execution time, followed by the algorithm GKFDK-LWO.
- The proposed method GKHDK-GWV was, on average, 4 times faster than the algorithm WGKFDK.

Scalability

- Assess the scalability of the algorithms with respect to execution time
  - Caso 1: $P = 1000$ and $N \in \{1000, 2000, ..., 10000\}$;
  - Caso 2: $N = 1000$ and $P \in \{1000, 2000, ..., 10000\}$;
  - Synthetic data ($K = H = 4$) using block latent model [12].
- The proposed hard variants demonstrated low execution times, even when applied to large datasets
- GKHDK-GWV scales more efficiently compared to the state-of-the-art WGKFDK algorithm
- The proposed fuzzy variants exhibited longer execution times;

WDBC dataset

- WDBC dataset contains 569 objects, described by 30 variables from digitized images of a breast mass, obtained through a common procedure in oncology to diagnose cancer
- These objects are labeled in two classes: benign, with 357 examples, and malignant, with 252 examples;
- This dataset was normalized based on the mean and standard deviation of the variables;
- Each algorithm was run 100 times and the best result was reported according to the objective function (lowest value)

Prototypes

Table 16: Prototypes provided by the proposed algorithms on WDBC dataset.

		GKHDK-GWV		GKHDK-GWO		GKHDK-JGW	
		Variables		Variables		Variables	
		Cluster 1	Cluster 2	Cluster 1	Cluster 2	Cluster 1	Cluster 2
Objects	Cluster 1	-0.567	-0.417	-0.375	-0.529	0.089	0.722
	Cluster 2	0.876	0.238	-0.175	0.520	-0.434	-0.563
		GKHDK-LWV		GKHDK-LWO		GKHDK-JLW	
		Variables		Variables		Variables	
		Cluster 1	Cluster 2	Cluster 1	Cluster 2	Cluster 1	Cluster 2
Objects	Cluster 1	-0.568	-0.416	0.533	0.134	-0.464	-0.587
	Cluster 2	0.852	0.234	-0.581	-0.464	0.220	0.754
		GKFDK-GWV		GKFDK-GWO		GKFDK-JGW	
		Variables		Variables		Variables	
		Cluster 1	Cluster 2	Cluster 1	Cluster 2	Cluster 1	Cluster 2
Objects	Cluster 1	0.223	0.856	0.533	-0.175	-0.433	-0.593
	Cluster 2	-0.415	-0.564	-0.525	-0.420	0.089	0.707
		GKFDK-LWV		GKFDK-LWO		GKFDK-JLW	
		Variables		Variables		Variables	
		Cluster 1	Cluster 2	Cluster 1	Cluster 2	Cluster 1	Cluster 2
Objects	Cluster 1	0.785	0.205	-0.255	-0.566	-0.168	-0.582
	Cluster 2	-0.582	-0.426	0.105	-0.119	0.199	0.773

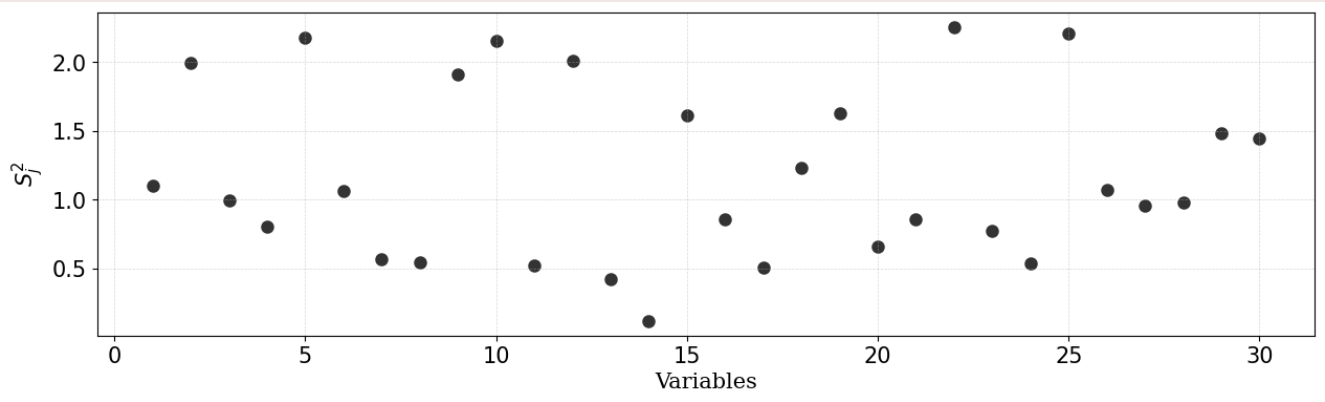
- The prototypes provided by the proposed algorithms are well separate
- The clustering obtained corresponds to the a priori classes

Bandwidth

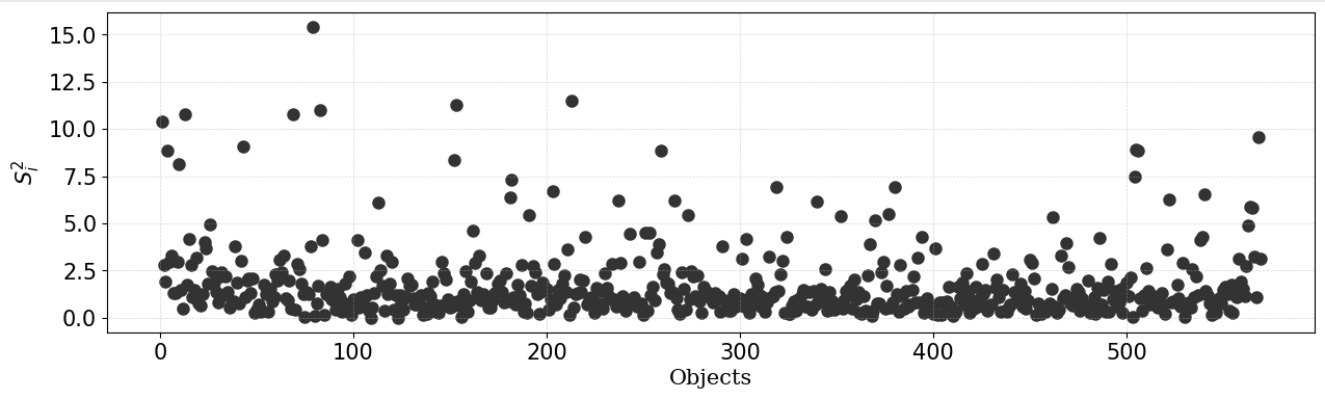
- The bandwidth hyperparameters are associated with the variability of objects or variables within data blocks
- When the bandwidth parameter is defined by variables, the closer the objects are to the block prototypes concerning a variable, the lower the value of the width hyperparameter for that variable;
- A small value indicates that within that block, the objects in that variable are more compact, and this variable is more relevant to the block to which it belongs;
- Likewise, if the bandwidth parameter is defined in terms of objects, the closer the variables are to the block prototypes concerning an object, the lower the value of the width hyperparameter associated with this object
- It means that a small value indicates that within that block, the variables in that object are more compact, and this object is more relevant to the block to which it belongs

Bandwidth

Figure 1: Width hyperparameter values of the variants hard global variants (GKHDK-GWV, GKHDK-GWO) on WDBC dataset.



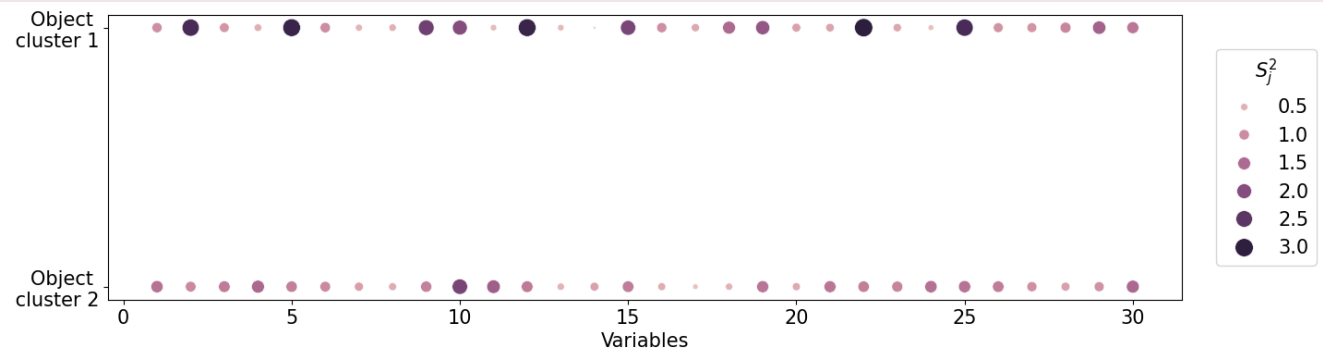
(a) GKHDK-GWV



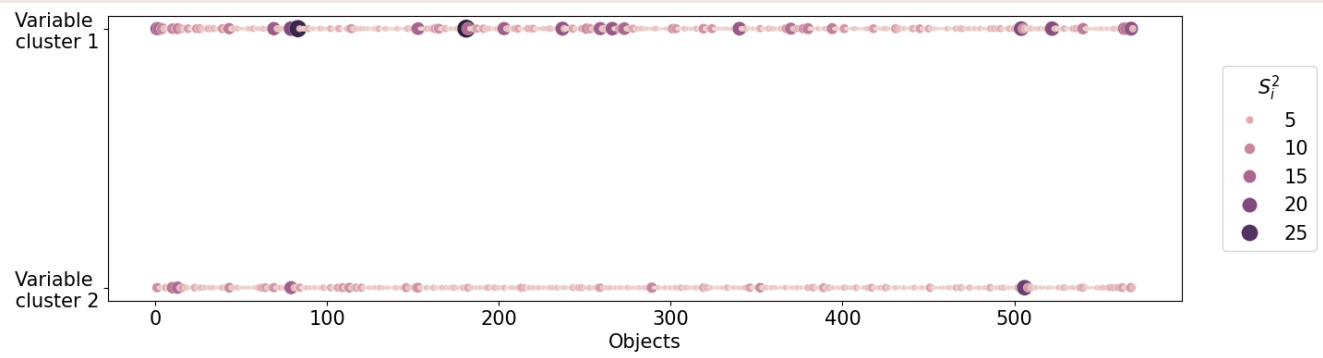
(b) GKHDK-GWO

Bandwidth

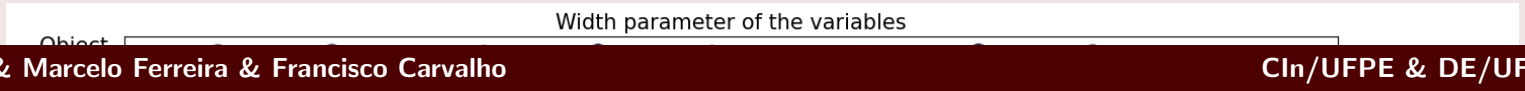
Figure 2: Width hyperparameter values of the variants hard global variants (GKHDK-GWV, GKHDK-GWO) on WDBC dataset.



(a) GKHDK-GWV



(b) GKHDK-GWO



Concluding Remarks

- This presentation showed hard and fuzzy co-clustering methods that automatically learn the bandwidth parameters of the Gaussian kernel
- We define the structure of the bandwidth parameter not only in terms of variables but also in terms of objects (the models group both simultaneously)
- These bandwidth parameters can be the same for all clusters, varying only with respect to objects or variables (global methods), or they can also vary across clusters (local methods)
- In local methods, when the bandwidth parameter is defined in terms of objects, these are different for each variable cluster
- Likewise, when the bandwidth parameter is defined in terms of variables, these are different for each object cluster
- Main advantage: they compute the value of the bandwidth parameters according to the data (no previous tunning step or heuristic-based estimates)
- Furthermore, this parameter is not unique for the entire dataset, but different for objects, variables, or both
- The methods can rescale the objects and variables separately, according to their distribution; in the local case, also according to the distribution in each variable cluster and object cluster

Concluding Remarks

- The results showed the effectiveness of the proposed algorithm;
  - When applied to real datasets, they achieved an overall satisfactory performance;
  - The variant GKHDK-GWV outperformed the other algorithms both in the analysis of the best solution and in the analysis of the average case
  - It was demonstrated the effectiveness of this model in practical applications, as well as its robustness about random initialization
  - the variants GKHDK-JGW and GKHDK-JLW obtained also good results
  - All proposed methods obtained better or competitive results regarding previous clustering algorithms that also compute the bandwidth hyperparameters
  - All the proposed methods outperformed the spectral co-clustering methods
- Future Research
  - the proposed fuzzy methods are very sensitive to the choice of the m and n parameters, One idea to tackle this problem is to develop methods based on entropy regularization
  - The proposed methods require prior knowledge of the number of clusters of objects and variables. In future work, we plan to study ways of choosing these parameters

Main Reference

Journals & Magazines > IEEE Transactions on Fuzzy Sy... > Volume: 33 Issue: 6

Fuzzy and Crisp Gaussian Kernel-Based Co-Clustering With Automatic Width Computation

Publisher: IEEE Cite This PDF

José Nataniel A. de Sá ; Marcelo R.P. Ferreira ; Francisco de A.T. de Carvalho **All Authors**

J. N. A. de Sá, M. R. P. Ferreira and F. d. A. T. de Carvalho, "Fuzzy and Crisp Gaussian Kernel-Based Co-Clustering With Automatic Width Computation," in *IEEE Transactions on Fuzzy Systems*, vol. 33, no. 6, pp. 1977-1991, June 2025, doi: 10.1109/TFUZZ.2025.3546802. keywords: {Kernel;Clustering algorithms;Fuzzy systems;Prototypes;Clustering methods;Training;Peer-to-peer computing;Image segmentation;Computational modeling;Gaussian processes;Automated width computation;co-clustering;Gaussian kernel functions},

List of References

[1] Melissa Ailem, François Role, and Mohamed Nadif. "Co-clustering document-term matrices by direct maximization of graph modularity". In: *Proceedings of the 24th ACM international on conference on information and knowledge management*. 2015, pp. 1807–1810.

[2] Takuya Akiba et al. "Optuna: A Next-generation Hyperparameter Optimization Framework". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2019.

[3] Ricardo Baeza-Yates, Berthier Ribeiro-Neto, et al. *Modern information retrieval*. Vol. 463. ACM press New York, 1999.

[4] James C Bezdek. *Pattern recognition with fuzzy objective function algorithms*. Springer Science & Business Media, 2013.

[5] Barbara Caputo et al. "Appearance-based object recognition using SVMs: which kernel should I use?" In: *Proc of NIPS workshop on Statistical methods for computational experiments in visual processing and computer vision, Whistler*. Vol. 2002. 2002.

List of References

[6] Francisco de AT de Carvalho, Lucas VC Santana, and Marcelo RP Ferreira. "Gaussian kernel-based fuzzy clustering with automatic bandwidth computation". In: *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part I 27*. Springer. 2018, pp. 685–694.

[7] Francisco de AT de Carvalho et al. "Gaussian kernel c-means hard clustering algorithms with automated computation of the width hyper-parameters". In: *Pattern recognition* 79 (2018), pp. 370–386.

[8] Inderjit S Dhillon. "Co-clustering documents and words using bipartite spectral graph partitioning". In: *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. 2001, pp. 269–274.

[9] E. Diday and G. Govaert. "Classification automatique avec distances adaptatives". In: *R.A.I.R.O. Informatique Computer Science* 11 (1977), pp. 329–349.

List of References

[10] Maurizio Filippone et al. "A survey of kernel and spectral methods for clustering". In: *Pattern recognition* 41.1 (2008), pp. 176–190.

[11] Mark A. Girolami. "Mercer kernel-based clustering in feature space". In: *IEEE Trans. Neural Networks* 13.3 (2002), pp. 780–784.

[12] Gérard Govaert and Mohamed Nadif. *Co-clustering: models, algorithms and applications*. John Wiley & Sons, 2013.

[13] Daniel Graves and Witold Pedrycz. "Kernel-based fuzzy clustering and fuzzy clustering: A comparative experimental study". In: *Fuzzy sets and systems* 161.4 (2010), pp. 522–543.

[14] Joshua Zhexue Huang et al. "Automated Variable Weighting in k-Means Type Clustering". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 27.5 (2005), pp. 657–668.

[15] Lawrence Hubert and Phipps Arabie. "Comparing partitions". In: *Journal of classification* 2.1 (1985), pp. 193–218.

List of References

[16] Eyke Hullermeier et al. "Comparing fuzzy partitions: A generalization of the rand index and related measures". In: *IEEE Transactions on Fuzzy Systems* 20.3 (2011), pp. 546–556.

[17] Yuval Kluger et al. "Spectral biclustering of microarray data: coclustering genes and conditions". In: *Genome research* 13.4 (2003), pp. 703–716.

[18] Charlotte Laclau, Francisco de AT de Carvalho, and Mohamed Nadif. "Fuzzy co-clustering with automated variable weighting". In: *2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE. 2015, pp. 1–8.

[19] J MacQueen. "Classification and analysis of multivariate observations". In: *5th Berkeley Symp. Math. Statist. Probability*. University of California Los Angeles LA USA. 1967, pp. 281–297.

[20] Dharmendra S. Modha and W. Scott Spangler. "Feature Weighting in *k*-Means Clustering". In: *Mach. Learn.* 52.3 (2003), pp. 217–237.

[21] Roberto Rocci and Maurizio Vichi. "Two-mode multi-partitioning". In: *Computational Statistics & Data Analysis* 52.4 (2008), pp. 1984–2003.

List of References

[22] José Nataniel A. de Sá, Marcelo R.P. Ferreira, and Francisco de A.T. de Carvalho. "Kernel-based Fuzzy Co-clustering in Feature Space with Automated Variable Weighting". In: *2022 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. 2022, pp. 1–8.

[23] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. "Nonlinear component analysis as a kernel eigenvalue problem". In: *Neural computation* 10.5 (1998), pp. 1299–1319.

[24] Bernhard Schölkopf and Alexander Johannes Smola. *Learning with Kernels: support vector machines, regularization, optimization, and beyond*. Adaptive computation and machine learning series. MIT Press, 2018.

[25] John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[26] Hongbin Shen et al. "Attribute weighted mercer kernel based fuzzy clustering algorithm for general non-spherical datasets". In: *Soft Computing* 10.11 (2006), pp. 1061–1073.

List of References

[27] Eduardo C Simões and Francisco de AT de Carvalho. "Gaussian Kernel Fuzzy C-Means with Width Parameter Computation and Regularization". In: *Pattern Recognition* (2023), p. 109749.

[28] Alexander Strehl and Joydeep Ghosh. "Cluster ensembles—a knowledge reuse framework for combining multiple partitions". In: *Journal of machine learning research* 3.Dec (2002), pp. 583–617.

[29] Wei Xu, Xin Liu, and Yihong Gong. "Document clustering based on non-negative matrix factorization". In: *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*. 2003, pp. 267–273.

Thank You!