Expert Systems With Applications 165 (2021) 113856

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Fuzzy C-Means clustering algorithm for data with unequal cluster sizes and contaminated with noise and outliers: Review and development

Salar Askari

Mechanical Engineering Department, Amirkabir University of Technology (Tehran Polytechnic), 424 Hafez Avenue, Tehran 1591634311, Iran

ARTICLE INFO

Keywords:
Fuzzy C-Means
FCM
Clustering
Outlier
Noise
Unequal clusters

ABSTRACT

Clustering algorithms aim at finding dense regions of data based on similarities and dissimilarities of data points. Noise and outliers contribute to the computational procedure of the algorithms as well as the actual data points that leads to inaccurate and misplaced cluster centers. This problem also arises when sizes of the clusters are different that moves centers of small clusters towards large clusters. Mass of the data points is important as well as their location in engineering and physics where non-uniform mass distribution results displacement of the cluster centers towards heavier clusters even if sizes of the clusters are identical and the data are noise-free. Fuzzy C-Means (FCM) algorithm that suffers from these problems is the most popular fuzzy clustering algorithm and has been subject of numerous researches and developments though improvements are still marginal. This work revises the FCM algorithm to make it applicable to data with unequal cluster sizes, noise and outliers, and non-uniform mass distribution. Revised FCM (RFCM) algorithm employs adaptive exponential functions to eliminate impacts of noise and outliers on the cluster centers and modifies constraint of the FCM algorithm to prevent large or heavier clusters from attracting centers of small clusters. Several algorithms are reviewed and their mathematical structures are discussed in the paper including Possibilistic Fuzzy C-Means (PFCM), Possibilistic C-Means (PCM), Robust Fuzzy C-Means (FCM-σ), Noise Clustering (NC), Kernel Fuzzy C-Means (KFCM), Intuitionistic Fuzzy C-Means (IFCM), Robust Kernel Fuzzy C-Mean (KFCM-σ), Robust Intuitionistic Fuzzy C-Means (IFCM-σ), Kernel Intuitionistic Fuzzy C-Means (KIFCM), Robust Kernel Intuitionistic Fuzzy C-Means (KIFCM-σ), Credibilistic Fuzzy C-Means (CFCM), Size-insensitive integrity-based Fuzzy C-Means (siibFCM), Size-insensitive Fuzzy C-Means (csiFCM), Subtractive Clustering (SC), Density Based Spatial Clustering of Applications with Noise (DBSCAN), Gaussian Mixture Models (GMM), Spectral clustering, and Outlier Removal Clustering (ORC). Some of these algorithms are suitable for noisy data and some others are designed for data with unequal clusters. The study shows that the RFCM algorithm works for both cases and outperforms the both categories of the algorithms.

1. Introduction

Clustering algorithms aim at grouping the data based on similarities and dissimilarities of data points. There are several categories of clustering algorithms. Cluster-centric algorithms such as K-means and FCM and their variants calculate centers of groups of data points mostly based on distance as the measure of similarity and dissimilarity (Havens, Bezdek, Leckie, Hall, & Palaniswami, 2012; Lei, Zhu, Chen, Lin, & Yang, 2012). Density based algorithms such as DBSCAN try to label neighboring data points with a specific concentration to one cluster, which is based on predefined density and neighborhood thresholds (Yu, Chen, Yao, & Wang, 2019). When the data contain uncertain attributes, belief functions are employed to manage uncertainty (Hariz, Elouedi, & Mellouli, 2006). Dempster-Shafer theory is one of the basic mathematical frameworks for such functions, which is devised for reasoning under uncertainty (Hamidzadeh & Ghadamyari, 2020). The data are clustered using the belief functions to handle uncertainty and these methods are usually called evidential clustering. On the other hand, mixture models assume that the data are generated by several distribution functions and employ a mixture of these functions to partition the data into different clusters such that each cluster is represented by one of the distributions (He & Ho, 2019; Zeng & Cheung, 2014). Most of the cluster-centric algorithms are only suited for spherical and elliptical clusters and are not applicable to clusters of arbitrary shapes. Spectral clustering algorithm transform connected components with any shape to round and compact clusters that could be easily identified by cluster-centric algorithms (von

E-mail address: s_askari@aut.ac.ir.

https://doi.org/10.1016/j.eswa.2020.113856
Received 11 March 2020; Received in revised form 13 July 2020; Accepted 6 August 2020
Available online 14 August 2020

Luxburg, 2007). Moreover, Fuzzy C-Ordered Means (FCOM) algorithm is proposed as a solution to resolve noise sensitivity of the FCM algorithm. The FCM algorithm calculates the cluster centers as weighted mean where the weights are membership grades of the data points in the clusters. It is well-known that such approach is vulnerable to noise and outliers since the algorithm assigns weights to all points in the same way and cannot effectively distinguish noise and outliers from the actual data points. The Ordered Weighted Averaging (OWA) method is employed in the FCOM algorithm to remove this deficiency and it is shown that this algorithm is superior to several other algorithms in dealing with noise and outliers (Leski, 2016; Siminski, 2017).

The present work mainly focuses on fuzzy clustering algorithms to handle the data contaminated with noise and outliers. These algorithms are applied to various practical problems such as image processing (Beliakov, Li, Vu, & Wilkin, 2015; Liu, Xu, Zhang, & Chen, 2014; Makrogiannis, Economou, Fotopoulos, & Bourbakis, 2005; Ozdemir & Akarun, 2001; Tolias & Panas, 1998), fuzzy time series (Chen & Chen, 2015; Li & Cheng, 2010; Lee, Chang, & Lin, 2014; Chen, Chu, & Sheu, 2012; Askari et al., 2015a, 2015b; Chen, Manalu, Pan, & Liu, 2013; Askari & Montazerin, 2015; Chen & Wang, 2010), system identification (Askari et al., 2015a,2015b), fuzzy systems (Askari, 2017a; Askari, Montazerin, & Fazel Zarandi, 2020; Tung & Quek, 2004; Zhang, Hall, & Goldgof, 2002), oil reservoirs classification (Askari, 2017b), etc. These algorithms are unsupervised techniques and cluster the data based on similarities and dissimilarities that is measured by distances of the data points to the cluster centers. Distance is the key parameter in these algorithms since it determines degree of belongingness of each data point to different clusters. Cluster centers are then calculated as weighted averages of the data points. The way of computing these weights differs from one algorithm to another. Therefore, all data points regardless of being outlier or noise, equally contribute to the calculation of cluster centers and there is no filtration to reduce or eliminate impacts of noise and outliers. Such cluster centers are displaced from dense regions and misrepresent actual structures of the data. Since any post-clustering computation and inference deals with the cluster centers or the membership grades calculated from these centers, it is necessary to reduce impacts of noise and outliers to find more accurate clusters.

Beside noise and outliers, sometimes there are clusters with different sizes in the data. Most of the fuzzy clustering algorithms treat all clusters equally and cannot measure size of the clusters that causes displacement of the centers of small clusters towards large clusters. Moreover, in physics and engineering the data points may represent particles or objects that their masses are important as well as their locations and centroid of clusters of particles should be calculated. Non-uniform mass distribution displaces centers of small clusters towards heavier ones. These misplaced cluster centers do not represent actual structure of the data and the algorithms should be corrected to handle these types of data.

Fuzzy C-Means (FCM) is the most popular fuzzy clustering algorithm that is highly sensitive to noise and outliers and size of the clusters (Bezdek, Ehrlich, & Full, 1984; Chen, Chen, & Lu, 2011; Groll & Jakel, 2005; Hathaway & Bezdek, 2001; Havens et al., 2012; Maji & Pal, 2007; Zhu, Chung, & Wang, 2009). Many researches are carried out to overcome these problems. Possibilistic C-Means (PCM) is presented to deal with the data containing noise and outliers (Filippone, Masulli, & Rovetta, 2010; Koutroumbas, Xenaki, & Rontogiannis, 2018; Krishnapuram & Keller, 1993, 1996; Zhang & Leung, 2004; Zhang, Yang, Chen, & Xia, 2017). Sensitivity to initialization and generating coincident cluster centers are the main problems with this algorithm (Anderson, Bezdek, Popescu, & Keller, 2010; Pal, Pal, Keller, & Bezdek, 2005). In contrast to the PCM algorithm, FCM algorithm does not yield coincident clusters and is not sensitive to initialization (Pal et al., 2005). With the notion of utilizing advantages of the both algorithms and getting rid of their drawbacks, the FCM and PCM algorithms are combined to create Possibilistic Fuzzy C-Means (PFCM) algorithm that is supposed to handle noise and outliers by its possibilistic terms and avoid coincident clusters and sensitivity to initialization by its fuzzy terms (Pal et al., 2005). However, experiments show that the PFCM algorithm does not perform well even on data with a few outliers (Askari, Montazerin, & Fazel Zarandi, 2017; Askari, Montazerin, Zarandi, & Hakimi, 2017). Two other algorithms are presented in Askari, Montazerin, Zarandi, and Hakimi (2017) and Askari, Montazerin, and Fazel Zarandi (2017) for noisy data by modifying objective function of the PFCM algorithm. Although these algorithms are more accurate than the FCM, PCM, and PFCM algorithms, but they suffer from high running time and complexity.

The present work revises the FCM algorithm to handle data with noise and outliers, unequal clusters, and non-uniform mass distribution. This algorithm is not sensitive to initialization and does not provide coincident cluster centers. Rest of the paper is organized as follows. Several well-known algorithms are briefly reviewed in Section 2. The RFCM algorithm is formulated in Section 3. Performance of the algorithms is studied in Section 4 and concluding remarks are drawn in Section 5.

2. Brief review of the clustering algorithms

There are several clustering algorithms designed for different purposes to deal with various difficulties arise in data clustering. Some prominent and well-known algorithms mostly related to the FCM algorithm are discussed in the following subsections.

2.1. Fuzzy C-Means (FCM) algorithm

Most of the clustering algorithms are based on minimizing an objective function to get the most compact clusters placed in dense regions of data. Objective function of the FCM algorithm is as follows (Pal et al., 2005). Since this function sums the pairwise difference between data points and cluster centers, the most compact clusters are found by minimizing the function.

$$J = \sum_{j=1}^{n}\sum_{i=1}^{c} u_{ij}^{m} \| x_j - v_i \|_A^2, \sum_{k=1}^{c} u_{kj} = 1 \tag{1}$$

where n is the number of data points, $c \in [2, n]$ is the number of clusters, r is the number of variables or features, v_i $(r \times 1)$ is the ith cluster center (ith column of the cluster centers matrix $V_{r\times c}$), x_j $(r \times 1)$ is the jth data point (jth column of the data matrix $X_{r\times n}$), u_{ij} is membership grade of the jth data point in the ith cluster (element of the partition matrix $U_{c\times n}$), $m > 1$ is degree of fuzziness (typically $m = 2$), $\| x_j - v_i \|_A^2 = (x_j - v_i)' A (x_j - v_i)$ is the distance between x_j and v_i, where $A_{r\times r}$ is a norm matrix. The identity norm matrix $A_{r\times r} = I_r$ yields Euclidean distance and spherical clusters whereas the following covariance norm matrix results Mahalanobis distance and elliptical clusters.

$$A = \left(\frac{1}{n} \sum_{j=1}^{n} (x_j - \bar{v})(x_j - \bar{v})' \right)^{-1}, \quad \bar{v} = \frac{1}{n} \sum_{j=1}^{n} x_j \tag{2}$$

The following cluster centers and membership grades minimize Eq. (1) (Pal et al., 2005).

$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m}, u_{ij} = \left[\sum_{k=1}^{c}\left(\frac{\| x_j - v_i\|_A^2}{\| x_j - v_k\|_A^2}\right)^{\frac{1}{m-1}}\right]^{-1} \quad (3)$$

The FCM algorithm starts with $U = rand(c,n) = [u_{ij}]$ that generates a random $c \times n$ partition matrix U to initialize the algorithm. Cluster centers v_i and membership degrees u_{ij} and are then updated using (3) until convergence of the algorithm. In the following algorithm, τ is the maximum number of iterations. Since the FCM algorithm does not perform well on noisy data or data with unequal clusters, many improvements have been made on the algorithm, which are discussed in the following subsections.

FCM algorithm

Inputs : X, c, ε, m, τ

$U = \text{rand}(c, n)$

for $t = 1 : \tau$

$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m} \quad \forall i \in [1, c]$$

$$u_{ij} = \left[\sum_{k=1}^{c}\left(\frac{\|x_j - v_i\|_A^2}{\|x_j - v_k\|_A^2}\right)^{\frac{1}{m-1}}\right]^{-1} \quad \forall i \in [1, c], \forall j \in [1, n]$$

if $\|V^{(t+1)} - V^{(t)}\| \le \varepsilon$

Stop

else

end

end

Outputs : U, V

2.2. Possibilistic Fuzzy C-Means (PFCM) algorithm

The PFCM algorithm is proposed with the following objective function to overcome shortcomings of the FCM algorithm in dealing with noise and outliers (Pal et al., 2005).

$$J = \sum_{j=1}^{n}\sum_{i=1}^{c}\left(c_F u_{ij}^m + c_P t_{ij}^\eta\right)\| x_j - v_i\|_A^2 + \sum_{i=1}^{c}\gamma_i \sum_{j=1}^{n}(1 - t_{ij})^\eta, \sum_{k=1}^{c} u_{kj} = 1 \quad (4)$$

where c_F and c_P are user defined constants that represent relative importance of fuzzy and possibilistic terms of the algorithm (typically $c_F = c_P = 1$). Larger values of c_P makes the algorithm more accurate when noise and outliers are present in the data. Moreover, t_{ij} that is associated with the possibilistic terms of the PFCM algorithm is typicality of the jth data point in the ith cluster (element of the typicality matrix $T_{c\times n}$) that indicates its belongingness in that cluster, $\eta > 1$ is possibilistic exponent (typically $\eta = 2$), and $\gamma_i = K\left(\sum_{j=1}^{n} u_{ij}^m \| x_j - v_i\|_A^2\right) \Big/ \sum_{j=1}^{n} u_{ij}^m$ where K is a predefined constant (typically $K = 1$) (Pal et al., 2005). Minimizing objective function of the PFCM algorithm provides the following equations for typicalities t_{ij} and cluster centers v_i (Pal et al., 2005). The membership grades u_{ij} are again obtained as (3).

$$v_i = \frac{\sum_{j=1}^{n}\left(c_F u_{ij}^m + c_P t_{ij}^\eta\right) x_j}{\sum_{j=1}^{n}\left(c_F u_{ij}^m + c_P t_{ij}^\eta\right)}$$

$$t_{ij} = \left(1 + \left(\frac{c_P}{\gamma_i}\| x_j - v_i\|_A^2\right)^{\frac{1}{\eta-1}}\right)^{-1} \quad (5)$$

The PFCM algorithm initializes with cluster centers calculated by the FCM algorithm and then cluster centers v_i, membership degrees u_{ij}, and typicalities t_{ij} are updated using until convergence criteria are met. Note that PFCM algorithm with $c_F = 1$ and $c_P = 0$ converts to the FCM algorithm. The data points far away from the cluster centers obviously get lower typicalities and have less impact on the cluster centers. This is why this algorithm is more proper for handling noise and outliers. However, this algorithm suffers from sensitivity to initialization and coincident clusters especially when the data are highly noisy and the clusters are overlapped (Askari, Montazerin, & Fazel Zarandi, 2017; Askari, Montazerin, Zarandi, & Hakimi, 2017).

PFCM algorithm

Inputs : X, c, ε, m, η, τ, c_F, c_P, K

$[U, V] = \text{FCM}(X, c, \varepsilon, m, \tau)$

$$\gamma_i = K\frac{\sum_{j=1}^{n} u_{ij}^m \|x_j - v_i\|_A^2}{\sum_{j=1}^{n} u_{ij}^m} \quad \forall i \in [1, c]$$

for $t = 1 : \tau$

$$t_{ij} = \left(1 + \left(\frac{c_P}{\gamma_i}\|x_j - v_i\|_A^2\right)^{\frac{1}{\eta-1}}\right)^{-1} \quad \forall i \in [1, c], \forall j \in [1, n]$$

$$u_{ij} = \left[\sum_{k=1}^{c}\left(\frac{\|x_j - v_i\|_A^2}{\|x_j - v_k\|_A^2}\right)^{\frac{1}{m-1}}\right]^{-1} \quad \forall i \in [1, c], \forall j \in [1, n]$$

$$v_i = \frac{\sum_{j=1}^{n}\left(c_F u_{ij}^m + c_P t_{ij}^\eta\right) x_j}{\sum_{j=1}^{n}\left(c_F u_{ij}^m + c_P t_{ij}^\eta\right)} \quad \forall i \in [1, c]$$

if $\|V^{(t+1)} - V^{(t)}\| \le \varepsilon$

Stop

else

end

end

Outputs : U, V

2.3. Possibilistic C-Means (PCM) algorithm

Letting $c_F = 0$ and $c_P = 1$ in (4) and (5) yields the PCM algorithm with the following update equations (Koutroumbas et al., 2018; Krishnapuram & Keller, 1993, 1996; Pal et al., 2005).

$$v_i = \frac{\sum_{j=1}^{n} t_{ij}^{\eta} x_j}{\sum_{j=1}^{n} t_{ij}^{\eta}}, t_{ij} = \left(1 + \left(\frac{1}{\gamma_i} \| x_j - v_i \|_A^2\right)^{\frac{1}{\eta-1}}\right)^{-1} \tag{6}$$

The PCM algorithm starts with the cluster centers computed by the FCM algorithm and then v_i and t_{ij} are updated until convergence of the algorithm.

PCM algorithm
Inputs : X, c, ε, m, η, τ, K
$[U, V] = \mathrm{FCM}(X, c, \varepsilon, m, \tau)$
$$\gamma_i = K \frac{\sum_{j=1}^{n} u_{ij}^{m} \|x_j - v_i\|_A^2}{\sum_{j=1}^{n} u_{ij}^{m}} \forall i \in [1, c]$$
for t = 1 : τ
$$t_{ij} = \left(1 + \left(\frac{1}{\gamma_i} \|x_j - v_i\|_A^2\right)^{\frac{1}{\eta-1}}\right)^{-1} \quad \forall i \in [1, c], \forall j \in [1, n]$$
$$v_i = \frac{\sum_{j=1}^{n} t_{ij}^{\eta} x_j}{\sum_{j=1}^{n} t_{ij}^{\eta}} \forall i \in [1, c]$$
if $\|V^{(t+1)} - V^{(t)}\| \le \varepsilon$
Stop
else
end
end
Outputs : U, V

2.4. Robust Fuzzy C-Means (FCM-σ) algorithm

The FCM-σ algorithm is presented to improve performance of the FCM algorithm on data with clusters of uneven densities and non-hyperspherical shapes (Tsai & Lin, 2011). Objective function of this algorithm is the same as that of the FCM algorithm but with normalized distance as follows.

$$J = \sum_{j=1}^{n} \sum_{i=1}^{c} u_{ij}^{m} \frac{\| x_j - v_i \|_A^2}{\sigma_i}, \sum_{k=1}^{c} u_{kj} = 1$$
$$\sigma_i = \left(\frac{\sum_{j=1}^{n} u_{ij}^{m} \| x_j - v_i \|_A^2}{\sum_{j=1}^{n} u_{ij}^{m}}\right)^{\frac{1}{2}} \tag{7}$$

This algorithm normalizes the distance by spread of the data from each cluster center σ_i. Update equations of this algorithm are as follows where v_i is obtained as (3) (Tsai & Lin, 2011).

$$u_{ij} = \left[\sum_{k=1}^{c} \left(\frac{\| x_j - v_i \|_A^2 / \sigma_i}{\| x_j - v_k \|_A^2 / \sigma_k}\right)^{\frac{1}{m-1}}\right]^{-1} \tag{8}$$

The FCM-σ algorithm initializes with the random partition matrix $U = rand(c, n)$ and then in each iteration σ_i, u_{ij}, and v_i are updated until convergence.

FCM − σ algorithm
Inputs : X, c, ε, m, τ
U = rand(c, n)
for t = 1 : τ
$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^{m} x_j}{\sum_{j=1}^{n} u_{ij}^{m}} \forall i \in [1, c]$$
$$\sigma_i = \left(\frac{\sum_{j=1}^{n} u_{ij}^{m} \|x_j - v_i\|_A^2}{\sum_{j=1}^{n} u_{ij}^{m}}\right)^{\frac{1}{2}} \forall i \in [1, c]$$
$$u_{ij} = \left[\sum_{k=1}^{c} \left(\frac{\|x_j - v_i\|_A^2 / \sigma_i}{\|x_j - v_k\|_A^2 / \sigma_k}\right)^{\frac{1}{m-1}}\right]^{-1} \quad \forall i \in [1, c], \forall j \in [1, n]$$
if $\|V^{(t+1)} - V^{(t)}\| \le \varepsilon$
Stop
else
end
end
Outputs : U, V

2.5. Noise clustering (NC)

The NC algorithm allocates one cluster center v_N to all noise and outliers in the data and the distance δ between any data point x_j and v_N is assumed constant $\| x_j - v_N \|_A^2 = \delta^2 \ \forall j$. Therefore, all noise and outliers are treated as one separate cluster excluded from the actual data and for any data point x_j the constraint $\sum_{k=1}^{c} u_{kj} = 1 - u_{Nj}$ is held where u_{Nj} is membership grade of x_j in the noise cluster. Objective function of this algorithm is as follows (Dave & Krishnapuram, 1997; Dave, 1991).

$$J = \sum_{j=1}^{n} \sum_{i=1}^{c} u_{ij}^{m} \| x_j - v_i \|_A^2 + \sum_{j=1}^{n} \delta^2 \left(1 - \sum_{k=1}^{c} u_{kj}\right)^m \tag{9}$$

The membership grades u_{ij} are as follows and the cluster centers v_i are calculated as (3) (Dave & Krishnapuram, 1997; Dave, 1991).

$$u_{ij} = \left[\left(\frac{\| x_j - v_i \|_A^2}{\delta^2}\right)^{\frac{1}{m-1}} + \sum_{k=1}^{c} \left(\frac{\| x_j - v_i \|_A^2}{\| x_j - v_k \|_A^2}\right)^{\frac{1}{m-1}}\right]^{-1} \tag{10}$$

The following value of δ^2 is recommended in Dave (1991).

$$\delta^2 = \frac{\lambda}{nc} \sum_{j=1}^{n} \sum_{i=1}^{c} \| x_j - v_i \|_A^2 \tag{11}$$

Clustering results of the NC algorithm are not very sensitive to the constant λ, which is chosen from the interval $[0.005, 0.5]$ (Dave, 1991). The NC algorithm initializes with the random partition matrix $U = rand(c, n)$ and then in each iteration, δ^2 is computed from (11) and u_{ij} and v_i are updated until convergence. Note that the algorithm does not calculate noise and outliers cluster center v_N and this cluster is just excluded from the data, which is clearly demonstrated by the constraint $\sum_{k=1}^{c} u_{kj} = 1 - u_{Nj}$.

NC algorithm

Inputs : X, c, ε, m, τ, λ

$U = \mathrm{rand}(c,n)$

for t = 1 : τ

$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^{m} x_j}{\sum_{j=1}^{n} u_{ij}^{m}} \; \forall i \in [1,c]$$

$$\delta^2 = \frac{\lambda}{nc} \sum_{j=1}^{n} \sum_{i=1}^{c} \left\| x_j - v_i \right\|_A^2$$

$$u_{ij} = \left[\left(\frac{\left\| x_j - v_i \right\|_A^2}{\delta^2} \right)^{\frac{1}{m-1}} + \sum_{k=1}^{c} \left(\frac{\left\| x_j - v_i \right\|_A^2}{\left\| x_j - v_k \right\|_A^2} \right)^{\frac{1}{m-1}} \right]^{-1} \; \forall i \in [1,c], \forall j \in [1,n]$$

if $\left\| V^{(t+1)} - V^{(t)} \right\| \le \varepsilon$

Stop

else

end

end

Outputs : U, V

2.6. Kernel Fuzzy C-Means (KFCM)

The FCM algorithm is only suitable for data with linearly separable structures. The data are transformed to a higher dimensional feature space by nonlinear mapping using kernel function $\Phi(x)$ to provide nonlinear separation of data points using the KFCM algorithm. Objective function of this algorithm is as follows (Tsai & Lin, 2011).

$$\begin{aligned} &J = \sum_{j=1}^{n} \sum_{i=1}^{c} u_{ij}^{m} \| \Phi(x_j) - \Phi(v_i) \|_A^2, \sum_{k=1}^{c} u_{kj} = 1 \\ &\| \Phi(x_j) - \Phi(v_i) \|_A^2 = K(x_j, x_j) + K(v_i, v_i) - 2K(x_j, v_i) \end{aligned} \tag{12}$$

Using the Gaussian kernel $K(x,y) = exp\left(- \| x - y \|_A^2 / \sigma^2 \right)$ simply results $\| \Phi(x_j) - \Phi(v_i) \|_A^2 = 2 - 2K(x_j, v_i)$. Update equations of the KFCM algorithm with the Gaussian kernel are as follows (Tsai & Lin, 2011).

$$\begin{aligned} u_{ij} &= \left[\sum_{k=1}^{c} \left(\frac{1 - K(x_j, v_i)}{1 - K(x_j, v_k)} \right)^{\frac{1}{m-1}} \right]^{-1} \\ v_i &= \frac{\sum_{j=1}^{n} u_{ij}^{m} K(x_j, v_i) x_j}{\sum_{j=1}^{n} u_{ij}^{m} K(x_j, v_i)} \end{aligned} \tag{13}$$

Bandwidth of the Gaussian kernel σ is calculated as the variance of distances of all data points from the centroid of the entire data $\bar{v}$ (Tsai & Lin, 2011).

$$\begin{aligned} &\bar{v} = \frac{1}{n} \sum_{j=1}^{n} x_j, \bar{d} = \frac{1}{n} \sum_{j=1}^{n} \sqrt{\| x_j - \bar{v} \|_A^2} \\ &\sigma^2 = \frac{1}{n-1} \sum_{j=1}^{n} \left(\sqrt{\| x_j - \bar{v} \|_A^2} - \bar{d} \right)^2 \end{aligned} \tag{14}$$

The bandwidth σ is calculated from (14) and the KFCM algorithm starts with the random partition matrix $U = rand(c,n)$ from which the initial cluster centers are calculated. Membership degrees u_{ij} and cluster centers v_i are then updated until convergence of the algorithm.

KFCM algorithm

Inputs : X, c, ε, m, τ

$$\bar{v} = \frac{1}{n} \sum_{j=1}^{n} x_j, \bar{d} = \frac{1}{n} \sum_{j=1}^{n} \sqrt{\left\| x_j - \bar{v} \right\|_A^2}$$

$$\sigma^2 = \frac{1}{n-1} \sum_{j=1}^{n} \left(\sqrt{\left\| x_j - \bar{v} \right\|_A^2} - \bar{d} \right)^2$$

$U = \mathrm{rand}(c,n)$

$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^{m} x_j}{\sum_{j=1}^{n} u_{ij}^{m}} \; \forall i \in [1,c]$$

for t = 1 : τ

$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^{m} K(x_j, v_i) x_j}{\sum_{j=1}^{n} u_{ij}^{m} K(x_j, v_i)} \; \forall i \in [1,c]$$

$$u_{ij} = \left[\sum_{k=1}^{c} \left(\frac{1 - K(x_j, v_i)}{1 - K(x_j, v_k)} \right)^{\frac{1}{m-1}} \right]^{-1} \; \forall i \in [1,c], \forall j \in [1,n]$$

if $\left\| V^{(t+1)} - V^{(t)} \right\| \le \varepsilon$

Stop

else

end

end

Outputs : U, V

2.7. Intuitionistic Fuzzy C-Means (IFCM)

Consider the membership function $\mu_A(x)$ defined on the universe of discourse of the fuzzy set A. Since $\mu_A(x)$ represents membership degree of x in A; $1 - \mu_A(x)$ indicates non-membership degree of x in A that is called complement $\nu_A(x)$. However, in an intuitionistic fuzzy set A, non-membership degree of x is unknown and generally different from $1 - \mu_A(x)$. The hesitation degree defined as $\pi_A(x) = 1 - \mu_A(x) - \nu_A(x)$ represents lack of knowledge in defining the membership function $\mu_A(x)$ (Chaira, 2011). Since intuitionistic fuzzy complement is given by $\nu_A(x) = [1 - (\mu_A(x))^{\rho}]^{1/\rho}$, the hesitation degree is calculated as $\pi_A(x) = 1 - \mu_A(x) - [1 - (\mu_A(x))^{\rho}]^{1/\rho}$ where ρ is a user defined constant (typically $\rho = 0.85$) (Chaira, 2011). Objective function of the IFCM algorithm is as follows (Chaira, 2011).

$$J = \sum_{j=1}^{n} \sum_{i=1}^{c} w_{ij}^{m} \| x_j - v_i \|_A^2 + \sum_{i=1}^{c} \pi_i e^{1-\pi_i}, \pi_i = \frac{1}{n} \sum_{j=1}^{n} \pi_{ij} \tag{15}$$

It is shown that update equations of the IFCM algorithm are as follows where u_{ij} is as (3) (Chaira, 2011).

$$\pi_{ij} = 1 - u_{ij} - \left(1 - u_{ij}^{\rho}\right)^{1/\rho}, w_{ij} = u_{ij} + \pi_{ij}$$
$$v_i = \frac{\sum_{j=1}^{n} w_{ij}^{m} x_j}{\sum_{j=1}^{n} w_{ij}^{m}} \tag{16}$$

where the intuitionistic fuzzy membership degree w_{ij} is calculated by adding hesitation degree π_{ij} to the fuzzy membership degree u_{ij}.

The IFCM algorithm initializes by the random partition matrix $U = rand(c, n)$ and then u_{ij}, π_{ij}, w_{ij}, and v_i are updated until the convergence criteria are met.

IFCM algorithm

Inputs : X, c, ε, m, τ, ρ

$U = \text{rand}(c, n)$

for t = 1 : τ

$$\pi_{ij} = 1 - u_{ij} - \left(1 - u_{ij}^{\rho}\right)^{1/\rho}, w_{ij} = u_{ij} + \pi_{ij} \quad \forall i \in [1, c], \forall j \in [1, n]$$

$$v_i = \frac{\sum_{j=1}^{n} w_{ij}^{m} x_j}{\sum_{j=1}^{n} w_{ij}^{m}} \quad \forall i \in [1, c]$$

$$u_{ij} = \left[\sum_{k=1}^{c} \left(\frac{\|x_j - v_i\|_A^2}{\|x_j - v_k\|_A^2}\right)^{\frac{1}{m-1}}\right]^{-1} \quad \forall i \in [1, c], \forall j \in [1, n]$$

if $\|V^{(t+1)} - V^{(t)}\| \le \varepsilon$

Stop

else

end

end

Outputs : U, V

2.8. Robust Kernel Fuzzy C-Mean (KFCM-σ)

Normalizing the distance in the objective function of the KFCM algorithm yields the following objective function for the KFCM-σ algorithm (Gosain & Dahiya, 2016; Tsai & Lin, 2011).

$$J = \sum_{j=1}^{n}\sum_{i=1}^{c} u_{ij}^{m} \frac{\| \Phi(x_j) - \Phi(v_i) \|_A^2}{\Phi_{\sigma_i}}, \sum_{k=1}^{c} u_{kj} = 1$$
$$\Phi_{\sigma_i} = \left(\frac{\sum_{j=1}^{n} u_{ij}^{m} \| \Phi(x_j) - \Phi(v_i) \|_A^2}{\sum_{j=1}^{n} u_{ij}^{m}}\right)^{\frac{1}{2}} \tag{17}$$

Update equation of the KFCM-σ algorithm with the Gaussian kernel are as follows where bandwidth of the kernel σ is calculated from (14) and v_i is computed from (13) (Tsai & Lin, 2011).

$$u_{ij} = \left[\sum_{k=1}^{c} \left(\frac{(1 - K(x_j, v_i))/\Phi_{\sigma_i}}{(1 - K(x_j, v_k))/\Phi_{\sigma_k}}\right)^{\frac{1}{m-1}}\right]^{-1} \tag{18}$$

The KFCM-σ algorithm starts with the random partition matrix $U = rand(c, n)$ from which the initial cluster centers are calculated. The normalization factors Φ_{σ_i}, membership degrees u_{ij}, and cluster centers v_i are then updated until convergence of the algorithm.

KFCM - σ algorithm

Inputs : X, c, ε, m, τ

$$\bar{v} = \frac{1}{n}\sum_{j=1}^{n} x_j, \bar{d} = \frac{1}{n}\sum_{j=1}^{n} \|x_j - \bar{v}\|_A^2$$

$$\sigma^2 = \frac{1}{n-1}\sum_{j=1}^{n}\left(\|x_j - \bar{v}\|_A^2 - \bar{d}\right)^2$$

$U = \text{rand}(c, n)$

$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^{m} x_j}{\sum_{j=1}^{n} u_{ij}^{m}} \quad \forall i \in [1, c]$$

for t = 1 : τ

$$\Phi_{\sigma_i} = \left(\frac{2\sum_{j=1}^{n} u_{ij}^{m}(1 - K(x_j, v_i))}{\sum_{j=1}^{n} u_{ij}^{m}}\right)^{\frac{1}{2}} \quad \forall i \in [1, c]$$

$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^{m} K(x_j, v_i) x_j}{\sum_{j=1}^{n} u_{ij}^{m} K(x_j, v_i)} \quad \forall i \in [1, c]$$

$$u_{ij} = \left[\sum_{k=1}^{c} \left(\frac{(1 - K(x_j, v_i))/\Phi_{\sigma_i}}{(1 - K(x_j, v_k))/\Phi_{\sigma_k}}\right)^{\frac{1}{m-1}}\right]^{-1} \quad \forall i \in [1, c], \forall j \in [1, n]$$

if $\|V^{(t+1)} - V^{(t)}\| \le \varepsilon$

Stop

else

end

end

Outputs : U, V

2.9. Robust Intuitionistic Fuzzy C-Means (IFCM-σ)

Normalizing the distance in the objective function of the IFCM algorithm yields the following objective function for the IFCM-σ algorithm (Kaur, Soni, & Gosain, 2012).

$$J = \sum_{j=1}^{n}\sum_{i=1}^{c} w_{ij}^{m} \frac{\| x_j - v_i \|_A^2}{\sigma_i} + \sum_{i=1}^{c} \pi_i e^{1-\pi_i} \tag{19}$$

$$\pi_i = \frac{1}{n}\sum_{j=1}^{n}\pi_{ij} \tag{20}$$

where σ_i and u_{ij} are calculated from (7) and (8), respectively and π_{ij}, w_{ij}, and v_i are computed from (16).

The IFCM-σ algorithm initializes by the random partition matrix $U = rand(c,n)$ and then σ_i, u_{ij}, π_{ij}, w_{ij}, and v_i are updated until the algorithm converges.

IFCM - σ algorithm

Inputs : X, c, ε, m, τ, ρ

$U = \text{rand}(c, n)$

for t = 1 : τ

$$\pi_{ij} = 1 - u_{ij} - \left(1 - u_{ij}^{\rho}\right)^{1/\rho},\ w_{ij} = u_{ij} + \pi_{ij}\ \forall i \in [1,c], \forall j \in [1,n]$$

$$v_i = \frac{\sum_{j=1}^{n} w_{ij}^{m} x_j}{\sum_{j=1}^{n} w_{ij}^{m}}\ \forall i \in [1,c]$$

$$\sigma_i = \left(\frac{\sum_{j=1}^{n} u_{ij}^{m}\left\|x_j - v_i\right\|_A^2}{\sum_{j=1}^{n} u_{ij}^{m}}\right)^{\frac{1}{2}}\ \forall i \in [1,c]$$

$$u_{ij} = \left[\sum_{k=1}^{n}\left(\frac{\left\|x_j - v_i\right\|_A^2 / \sigma_i}{\left\|x_j - v_k\right\|_A^2 / \sigma_k}\right)^{\frac{1}{m-1}}\right]^{-1}\ \forall i \in [1,c], \forall j \in [1,n]$$

if $\left\|V^{(t+1)} - V^{(t)}\right\| \le \varepsilon$

Stop

else

end

end

Outputs : U, V

2.10. Kernel Intuitionistic Fuzzy C-Means (KIFCM)

Objective function of the KIFCM algorithm is as follows (Kaur et al., 2012).

$$J = \sum_{j=1}^{n}\sum_{i=1}^{c} w_{ij}^{m} \| \Phi(x_j) - \Phi(v_i) \|_A^2 + \sum_{i=1}^{c}\pi_i e^{1-\pi_i}$$
$$\pi_i = \frac{1}{n}\sum_{j=1}^{n}\pi_{ij} \tag{21}$$

It is shown that for minimizing this function, u_{ij} is calculated from (13) and π_{ij} and w_{ij} are computed from (16) and v_i is as follows (Kaur et al., 2012).

$$v_i = \frac{\sum_{j=1}^{n} w_{ij}^{m} K(x_j, v_i) x_j}{\sum_{j=1}^{n} w_{ij}^{m} K(x_j, v_i)} \tag{22}$$

Bandwidth of the kernel σ is calculated from (14). The KIFCM algorithm starts with the random partition matrix $U = rand(c,n)$ and then u_{ij}, π_{ij}, w_{ij}, and v_i are updated until convergence of the algorithm.

KIFCM algorithm

Inputs : X, c, ε, m, τ, ρ

$$\bar{v} = \frac{1}{n}\sum_{j=1}^{n} x_j,\ \bar{d} = \frac{1}{n}\sum_{j=1}^{n}\left\|x_j - \bar{v}\right\|_A^2$$

$$\sigma^2 = \frac{1}{n-1}\sum_{j=1}^{n}\left(\left\|x_j - \bar{v}\right\|_A^2 - \bar{d}\right)^2$$

$U = \text{rand}(c, n)$

for t = 1 : τ

$$\pi_{ij} = 1 - u_{ij} - \left(1 - u_{ij}^{\rho}\right)^{1/\rho},\ w_{ij} = u_{ij} + \pi_{ij}\ \forall i \in [1,c], \forall j \in [1,n]$$

$$v_i = \frac{\sum_{j=1}^{n} w_{ij}^{m} K(x_j, v_i) x_j}{\sum_{j=1}^{n} w_{ij}^{m} K(x_j, v_i)}\ \forall i \in [1,c]$$

$$u_{ij} = \left[\sum_{k=1}^{c}\left(\frac{1 - K(x_j, v_i)}{1 - K(x_j, v_k)}\right)^{\frac{1}{m-1}}\right]^{-1}\ \forall i \in [1,c], \forall j \in [1,n]$$

if $\left\|V^{(t+1)} - V^{(t)}\right\| \le \varepsilon$

Stop

else

end

end

Outputs : U, V

2.11. Robust Kernel Intuitionistic Fuzzy C-Means (KIFCM-σ)

Objective function of the KIFCM-σ algorithm is as follows (Kaur et al., 2012).

$$J = \sum_{j=1}^{n}\sum_{i=1}^{c} w_{ij}^{m}\frac{\| \Phi(x_j) - \Phi(v_i) \|_A^2}{\Phi_{\sigma_i}} + \sum_{i=1}^{c}\pi_i e^{1-\pi_i}$$
$$\pi_i = \frac{1}{n}\sum_{j=1}^{n}\pi_{ij} \tag{23}$$

where Φ_{σ_i} is computed from (17), π_{ij} and w_{ij} are calculated from (16), v_i is computed from (22), bandwidth of the kernel σ is calculated from (14), and u_{ij} is as follows (Kaur et al., 2012).

$$u_{ij} = \left[\sum_{k=1}^{c}\left(\frac{(1 - K(x_j, v_i))/\Phi_{\sigma_i}}{(1 - K(x_j, v_k))/\Phi_{\sigma_k}}\right)^{\frac{1}{m-1}}\right]^{-1} \tag{24}$$

The KIFCM-σ algorithm starts with the random partition matrix $U = rand(c,n)$ from which the initial cluster centers are calculated. Φ_{σ_i}, u_{ij}, π_{ij}, w_{ij}, and v_i are then updated until the algorithm converges.

KIFCM - σ algorithm

Inputs : X, c, ε, m, τ, ρ

$$\bar{v} = \frac{1}{n}\sum_{j=1}^{n} x_j, \bar{d} = \frac{1}{n}\sum_{j=1}^{n}\left\|x_j - \bar{v}\right\|_A^2$$

$$\sigma^2 = \frac{1}{n-1}\sum_{j=1}^{n}\left(\left\|x_j - \bar{v}\right\|_A^2 - \bar{d}\right)^2$$

$$U = \mathrm{rand}(c, n)$$

$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m} \forall i \in [1, c]$$

for t = 1 : τ

$$\Phi_{\sigma_i} = \left(\frac{2\sum_{j=1}^{n} u_{ij}^m \left(1 - K\left(x_j, v_i\right)\right)}{\sum_{j=1}^{n} u_{ij}^m}\right)^{\frac{1}{2}}$$

$$\pi_{ij} = 1 - u_{ij} - \left(1 - u_{ij}^{\rho}\right)^{1/\rho}, w_{ij} = u_{ij} + \pi_{ij} \ \forall i \in [1, c], \forall j \in [1, n]$$

$$v_i = \frac{\sum_{j=1}^{n} w_{ij}^m K\left(x_j, v_i\right) x_j}{\sum_{j=1}^{n} w_{ij}^m K\left(x_j, v_i\right)} \forall i \in [1, c]$$

$$u_{ij} = \left[\sum_{k=1}^{c}\left(\frac{\left(1 - K\left(x_j, v_i\right)\right)/\Phi_{\sigma_i}}{\left(1 - K\left(x_j, v_k\right)\right)/\Phi_{\sigma_k}}\right)^{\frac{1}{m-1}}\right]^{-1} \quad \forall i \in [1, c], \forall j \in [1, n]$$

if $\left\|V^{(t+1)} - V^{(t)}\right\| \le \varepsilon$

Stop

else

end

end

Outputs : U, V

2.12. Credibilistic Fuzzy C-Means (CFCM)

The FCM algorithm assigns high membership degrees to noise and outliers because of the constraint in (1) that displaces the cluster centers. The CFCM algorithm is proposed to overcome this deficiency with the following objective function (Chintalapudi & Kam, 1998).

$$J = \sum_{j=1}^{n}\sum_{i=1}^{c} u_{ij}^m \| x_j - v_i \|_A^2, \sum_{k=1}^{c} u_{kj} = \rho_j(U, V, X) \tag{25}$$

For any data point x_j; ρ_j generally depends on the partition matrix U, cluster centers matrix V, and data matrix X. To simplify derivation of the update equations it is assumed that $\rho_j = f(X)$ that results $\partial\rho_j/\partial u_{ij} = \partial\rho_j/\partial v_i = 0$ (Chintalapudi & Kam, 1998). Therefore, zeroing derivatives of the objective function with respect to the membership degrees u_{ij} and cluster centers v_i provides the following update equations for the CFCM algorithm where v_i is as (3).

$$u_{ij} = \rho_j\left[\sum_{k=1}^{c}\left(\frac{\| x_j - v_i \|_A^2}{\| x_j - v_k \|_A^2}\right)^{\frac{1}{m-1}}\right]^{-1} \tag{26}$$

Let $Y_j = \left\{y_j^1, y_j^2, \ldots, y_j^q\right\}$ be the q-nearest neighbors of x_j where $q = [\gamma(n/c)]$. Results of the CFCM algorithm are relatively insensitive to the user defined constant $\gamma \in (0, 1)$. The parameter ρ_j that is introduced to deal with noise and outliers is calculated using q-nearest neighbors of x_j as follows (Chintalapudi & Kam, 1998).

$$\rho_j = 1 - \frac{\theta_j - \min\limits_{p\in[1,n]}(\theta_p)}{\max\limits_{p\in[1,n]}(\theta_p) - \min\limits_{p\in[1,n]}(\theta_p)}, \theta_j = \frac{\sum_{z=1}^{q} \| y_j^z - x_j \|_A^2}{q} \tag{27}$$

where θ_j represents mean distance of x_j from its q-nearest neighbors. An outlier is supposed to have larger θ_j that is equivalent to smaller credibility ρ_j since its q-nearest neighbors are far away from it as compared to the data points in dense regions of the data.

To apply the CFCM algorithm to the data, θ_j and ρ_j are calculated from (27) and then the algorithm starts with the random partition matrix $U = rand(c, n)$. Finally, membership degrees u_{ij} and cluster centers v_i are updated until convergence of the algorithm.

CFCM algorithm

Inputs : X, c, ε, m, τ, γ

$$q = \left[\gamma\left(n / c\right)\right]$$

$$\theta_j = \frac{\sum_{z=1}^{q}\left\|y_j^z - x_j\right\|_A^2}{q} \forall j \in [1, n]$$

$$\rho_j = 1 - \frac{\theta_j - \min\limits_{p\in[1,n]}\left(\theta_p\right)}{\max\limits_{p\in[1,n]}\left(\theta_p\right) - \min\limits_{p\in[1,n]}\left(\theta_p\right)} \forall j \in [1, n]$$

$$U = \mathrm{rand}\ (c, n)$$

for t = 1 : τ

$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m} \forall i \in [1, c]$$

$$u_{ij} = \rho_j\left[\sum_{k=1}^{c}\left(\frac{\left\|x_j - v_i\right\|_A^2}{\left\|x_j - v_k\right\|_A^2}\right)^{\frac{1}{m-1}}\right]^{-1} \quad \forall i \in [1, c], \forall j \in [1, n]$$

if $\left\|V^{(t+1)} - V^{(t)}\right\| \le \varepsilon$

Stop

else

end

end

Outputs : U, V

2.13. Size-insensitive integrity-based Fuzzy C-Means (siibFCM)

If there are unequal clusters in the data, large clusters pull centers of small clusters as demonstrated by the update equation of cluster centers in the aforementioned algorithms. The siibFCM algorithm is proposed to improve the FCM algorithm to cope with this problem (Lin, Huang, Kuo, & Lai, 2014). Objective function and update equations of the siibFCM algorithm are basically the same as that of the CFCM algorithm ((25) and (26)) however the term ρ_j is calculated and interpreted differently (Lin et al., 2014). There are some preliminary concepts to calculate $\rho_j(X, U)$ that is only a function of the data X and the partition matrix U. A cluster A_i with center v_i is defined as a set of data points with the highest membership degrees in that cluster according to the partition matrix U. Size of this cluster is denoted by $|A_i|$ and its compactness is defined as follows (Lin et al., 2014).

$$C_i = 1 - \sqrt{\frac{1}{|A_i|}\sum_{x_j \in A_i}\left(\sqrt{\| x_j - v_i\|_A^2} - \mu_i\right)^2} \tag{28}$$

$$A_i = \left\{x_j \in X | u_{ij} > u_{kj}\ \forall k \in [1,c]; k \neq i\right\}$$

where μ_i represents average spread of the data points associated with the cluster A_i with respect to its center v_i (Lin et al., 2014).

$$\mu_i = \frac{1}{|A_i|}\sum_{x_j \in A_i}\sqrt{\| x_j - v_i\|_A^2} \tag{29}$$

Purity of the cluster A_i is defined as follows (Lin et al., 2014).

$$P_i = \frac{1}{|A_i|}\sum_{x_j \in A_i} p_{ij} \tag{30}$$

where purity p_{ij} of the data point x_j with respect to A_i is defined as follows (Lin et al., 2014).

$$p_{ij} = \frac{\left|\sqrt{\| x_j - v_i\|_A^2} - \sqrt{\| x_j - v_k\|_A^2}\right|}{\sqrt{\| v_i - v_k\|_A^2}}$$
$$k = arg\min_{r \in [1,c], r \neq i}\left(\sqrt{\| v_i - v_r\|_A^2}\right) \tag{31}$$

Finally, integrtirty I_i and normalized integrity I_i^* of the cluster A_i are defined in terms of compactness C_i and purity P_i of this cluster as follows (Lin et al., 2014).

$$I_i = \frac{1}{2}(C_i + P_i)$$
$$I_i^* = \frac{I_i - \min_{r \in [1,c]}(I_r)}{\max_{r \in [1,c]}(I_r) - \min_{r \in [1,c]}(I_r)}, 0 \leqslant I_i^* \leqslant 1 \tag{32}$$

Exponetially boosted purity p_{ij}^* is then defined as follows (Lin et al., 2014).

$$p_{ij}^* = e^{(1-I_i^*)p_{ij}} \tag{33}$$

A condition value ρ_j is attributed to the cluster A_i as follows where $|X|$ designates size of the entire data set (Lin et al., 2014). Note that number of data points in A_i and the data matrix X can be considered as the size of the cluster $|A_i|$ and size of the data $|X|$, respectively.

$$\rho_j = \frac{1 - S_i}{\max_{r \in [1,c]}(1 - S_r)}, i = arg\max_{r \in [1,c]}(u_{rj})$$
$$S_i = \frac{|A_i|}{|X|}, S_r = \frac{|A_r|}{|X|} \tag{34}$$

where S_i is relative size of the cluster A_i as compared to the size of the data matrix X. Finally, by the weights w_{ij} that are calculated using the condition value ρ_j and exponentially boosted purity p_{ij}^*, the following update equations are provided for this algorithm where the cluster centers v_i are computed from (3) (Lin et al., 2014).

$$w_{ij} = \rho_j p_{ij}^*$$
$$u_{ij} = w_{ij}\left[\sum_{k=1}^{c}\left(\frac{\| x_j - v_i\|_A^2}{\| x_j - v_k\|_A^2}\right)^{\frac{1}{m-1}}\right]^{-1} \tag{35}$$

The partition matrix U and cluster centers matrix V are calculated by the FCM algorithm to initialize the siibFCM algorithm. Then, in each iteration of the siibFCM algorithm, A_i are determined using (28). Normalized integrity I_i^* are then calculated from (28)–(32). The condition value ρ_j and the weights w_{ij} are calculated from (33)–(35) and finally the membership degrees u_{ij} and cluster centers v_i are updated using (35) and (3).

siibFCM algorithm
Inputs : X, c, ε, m, τ
$[U,V] = FCM(X, c, \varepsilon, m, \tau)$
for $t = 1:\tau$
$A_i = \left\{x_j \in X \mid u_{ij} > u_{kj} \forall k \in [1,c], k \neq i\right\} \forall i \in [1,c]$
$\mu_i = \frac{1}{|A_i|}\sum_{x_j \in A_i}\sqrt{\|x_j - v_i\|_A^2} \ \forall i \in [1,c]$
$C_i = 1 - \sqrt{\frac{1}{|A_i|}\sum_{x_j \in A_i}\left(\sqrt{\|x_j - v_i\|_A^2} - \mu_i\right)^2} \ \forall i \in [1,c]$
$k = \arg\min_{r \in [1,c], r \neq i}\left(\sqrt{\|v_i - v_r\|_A^2}\right)$
$p_{ij} = \frac{\left|\sqrt{\|x_j - v_i\|_A^2} - \sqrt{\|x_j - v_k\|_A^2}\right|}{\sqrt{\|v_i - v_k\|_A^2}} \forall i \in [1,c], \forall j \in [1,n]$
$P_i = \frac{1}{|A_i|}\sum_{x_j \in A_i} p_{ij} \ \forall i \in [1,c]$
$I_i = \frac{1}{2}(C_i + P_i) \forall i \in [1,c], I_i^* = \frac{I_i - \min_{r \in [1,c]}(I_r)}{\max_{r \in [1,c]}(I_r) - \min_{r \in [1,c]}(I_r)}$
$p_{ij}^* = e^{(1-I_i^*)p_{ij}} \ \forall i \in [1,c], \forall j \in [1,n]$
$S_i = \frac{|A_i|}{|X|}, S_r = \frac{|A_r|}{|X|} \forall i, r \in [1,c]$
$i = \arg\max_{r \in [1,c]}(u_{rj}) \forall j \in [1,n], \rho_j = \frac{1 - S_i}{\max_{r \in [1,c]}(1 - S_r)} \forall j \in [1,n]$
$w_{ij} = \rho_j p_{ij}^* \ \forall i \in [1,c], \forall j \in [1,n]$
$v_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m} \forall i \in [1,c]$
$u_{ij} = w_{ij}\left[\sum_{k=1}^{c}\left(\frac{\|x_j - v_i\|_A^2}{\|x_j - v_k\|_A^2}\right)^{\frac{1}{m-1}}\right]^{-1} \forall i \in [1,c], \forall j \in [1,n]$
if $\left\|V^{(t+1)} - V^{(t)}\right\| \leq \varepsilon$
Stop
else
end
end
Outputs : U, V

2.14. Size-insensitive Fuzzy C-Means (csiFCM)

The csiFCM algorithm is proposed to handle data with unequal clusters by introducing the term $\rho_j(X,U)$ in the constraint of the FCM algorithm $\sum_{k=1}^{c} u_{kj} = \rho_j(X,U)$ (Noordam, van den Broek, & Buydens, 2002). Objective function of this algorithm is the same as that of the CFCM algorithm however the condition value $\rho_j(X,U)$ is calculated from (34) instead of (27). Update equations of this algorithm are as (26). The siibFCM algorithm is derived from the csiFCM algorithm by changing the way $\rho_j(X,U)$ is computed (Lin et al., 2014). The csiFCM algorithm

initializes by the random partition matrix $U = rand(c, n)$ and then ρ_j, u_{ij}, and v_i are updated using (34) and (26) until the convergence criteria are met. Using (25) and the Lagrange Multipliers method, objective function of the csiFCM algorithm is written as follows.

$$J = \sum_{j=1}^{n}\sum_{i=1}^{c} u_{ij}^{m} \| x_j - v_i \|_A^2 + \sum_{j=1}^{n} \lambda_j \left(\sum_{k=1}^{c} u_{kj} - \rho_j \right)$$

where λ_j are the Lagrange Multipliers. Since ρ_j is a function of X and U according to (34), when zeroing derivative of J with respect to u_{ij} to calculate u_{ij}; $\partial\rho_j/\partial u_{ij}$ is not zero but it is mistakenly taken zero in this algorithm. The same mistake is repeated in (Lin et al., 2014) when deriving siibFCM algorithm from the csiFCM algorithm.

csiFCM algorithm

Inputs: $X, c, \varepsilon, m, \tau$

$U = \text{rand}(c, n)$

for $t = 1 : \tau$

$A_i = \left\{ x_j \in X \mid u_{ij} > u_{kj} \forall k \in [1,c]; k \neq i \right\} \forall i \in [1,c]$

$S_i = \frac{|A_i|}{|X|}, S_r = \frac{|A_r|}{|X|} \; \forall i, r \in [1,c]$

$i = \arg\max_{r \in [1,c]} (u_{rj}) \; \forall j \in [1,n]$

$\rho_j = \frac{1 - S_i}{\max_{r \in [1,c]} (1 - S_r)} \; \forall j \in [1,n]$

$v_i = \frac{\sum_{j=1}^{n} u_{ij}^{m} x_j}{\sum_{j=1}^{n} u_{ij}^{m}} \; \forall i \in [1,c]$

$u_{ij} = \rho_j \left[\sum_{k=1}^{c} \left(\frac{\|x_j - v_i\|_A^2}{\|x_j - v_k\|_A^2} \right)^{\frac{1}{m-1}} \right]^{-1} \; \forall i \in [1,c], \forall j \in [1,n]$

if $\left\| V^{(t+1)} - V^{(t)} \right\| \leq \varepsilon$

Stop

else

end

end

Outputs: U, V

2.15. Subtractive clustering (SC)

Number of clusters is known a priori in the aforementioned algorithms. The SC algorithm is proposed to calculate both number of clusters and the cluster centers (Chiu, 1994). This algorithm is inspired by the Mountain Method in which the data domain is divided into some grids with equal or unequal sizes and a potential is assigned to each vertex point of the grid based on its distances to other data points (Yager & Filev, 1994). The potential function represents density of the region over the vertex point. Both the number of clusters and the cluster centers are determined in the Mountain Method. A vertex point with many data points nearby has higher potential. The vertex point with the highest potential is chosen as the first cluster center and then potentials of all vertex points are reduced according to their distance to the first cluster center. Therefore, the vertex point near the first cluster center will have low potential to be the next cluster center. The vertex point with the highest potential is chosen as the next cluster center and potentials of all vertex points are again reduced according to their distances from this new cluster center. This procedure continues until a termination criteria are met (Yager & Filev, 1994). However, the SC algorithm assigns a potential to each data point instead of the vertex points but the same procedure is used to find the number of clusters and the cluster centers. Potential of the data point x_j is defined as follows (Chiu, 1994).

$$P_{x_j}^1 = \sum_{q=1}^{n} exp\left(- \frac{\| x_j - x_q \|_A^2}{r_1^2} \right), r_1 = \sigma_1 R \tag{36}$$

where R represents size of the data space and can be chosen differently for different dimensions (features) of the data. It can be taken unity for normalized data. The influence range σ_1 (typically $\sigma_1 \in [0.2, 0.5]$) defines the cluster radius to determine a neighborhood that the data points outside this region have little contributions to the potential of x_j. Note that different influence ranges may be used for different dimensions (features) of the data. Large σ_1 provides a few large clusters and small σ_1 results many small clusters. A data point with many points nearby has larger potential according to (36). Now the data point with the highest potential is adopted as the first cluster center v_1 and its potential is subtracted from the potentials of all data points as follows (Chiu, 1994). This is why the algorithm is called subtractive clustering.

$$P_{x_j}^2 = P_{x_j}^1 - P_{v_1}^1 exp\left(- \frac{\| x_j - v_1 \|_A^2}{r_2^2} \right), r_2 = \sigma_2 r_1 \tag{37}$$

where the quash factor $\sigma_2 > 1$ (typically $\sigma_2 = 1.25$) is to set r_2 greater than r_1 to avoid selecting cluster centers too close to each other. The data point with the highest potential is selected as the next cluster center v_2. It is clear that a data point close to the first cluster center has low potential and is less likely to be chosen as the next cluster center. Similarly, the ith cluster center v_i is the data point with the highest potential computed from the following function (Chiu, 1994).

$$P_{x_j}^i = P_{x_j}^{i-1} - P_{v_{i-1}}^{i-1} exp\left(- \frac{\| x_j - v_{i-1} \|_A^2}{r_2^2} \right), \quad r_2 = \sigma_2 r_1 \tag{38}$$

To stop the procedure and get the last cluster center v_c where c is the number of clusters, the termination criterion $P_{v_c}^c / P_{v_1}^1 < \varepsilon$ is used where ε is a user defined constant. This criterion is the main flaw of the SC algorithm since there is no standard method to calculate ε for a given data set and different values of ε provide different number of clusters that undermines the first objective of the algorithm, which is determination of the actual number of clusters. This problem has led to additional restrictions to make the algorithm more practicable and useful. If $P_{v_i}^i / P_{v_1}^1 > \varepsilon_a$ then v_i is definitely accepted as a cluster center where ε_a is the accept ratio (typically $\varepsilon_a = 0.5$). If $P_{v_i}^i / P_{v_1}^1 < \varepsilon_r$ then v_i is definitely rejected as a cluster center where ε_r is the reject ratio (typically $\varepsilon_r = 0.15$). If $\varepsilon_r < P_{v_i}^i / P_{v_1}^1 < \varepsilon_a$, more considerations should be made to decide on the status of v_i as a cluster center. Let d_{min} be the least distance between v_i and the previously found cluster centers. If $(d_{min}/r_1) + \left(P_{v_i}^i / P_{v_1}^1 \right) \geqslant 1$; v_i is accepted as a cluster center and otherwise it is rejected and $P_{v_i}^i$ is set to 0 and the data point with the next highest potential is adopted as the next cluster center and is examined and so on (Chiu, 1994).

To apply the SC algorithm to data, potential of all data points are calculated using (36) and the data point with the highest potential is adopted as the first cluster center. Potential of this cluster center is then subtracted from the potentials of all data points to find the subsequent cluster center such that for each data point x_j; $P_{x_j}^i$ is calculated from (38) and the data point with the highest potential is chosen as the next cluster center $v_i = x_j$ and then the criteria $P_{v_i}^i / P_{v_1}^1 > \varepsilon_a$; $P_{v_i}^i / P_{v_1}^1 < \varepsilon_r$; $\varepsilon_r < P_{v_i}^i / P_{v_1}^1 < \varepsilon_a$; and $(d_{min}/r_1) + \left(P_{v_i}^i / P_{v_1}^1 \right) \geqslant 1$ are examined as discussed above. If v_i falls within boundaries of the criteria, it is selected as a cluster center and otherwise it is rejected. The procedure continues until no more data points meet the criteria. Since the SC algorithm provides different results

for different sets of the parameters and there are no definite values for these parameters, results of the algorithm are quite controversial. In the following algorithm, C_{max} is the maximum number of clusters, which is determined by the user.

SC algorithm

Inputs : $X, C_{max}, \sigma_1, \sigma_2, \varepsilon_a, \varepsilon_r$

for $q = 1 : r$

$R_{min}(q,1) = \min(X(q,:))$

$R_{max}(q,1) = \max(X(q,:))$

end

$\vec{R}_{min} = [R_{min}(1,1) \quad R_{min}(2,1) \quad \cdots \quad R_{min}(r,1)]'$

$\vec{R}_{max} = [R_{max}(1,1) \quad R_{max}(2,1) \quad \cdots \quad R_{max}(r,1)]'$

$R = 0.5\sqrt{\left\|\vec{R}_{max} - \vec{R}_{min}\right\|_A^2}$

$r_1 = \sigma_1 R$

$r_2 = \sigma_2 r_1$

$$P_{x_j}^1 = \sum_{l=1}^{n} \exp\left(-\frac{\left\|x_j - x_1\right\|_A^2}{r_1^2}\right) \forall j \in [1,n]$$

$$k = \arg\left(\max_j \left\{P_{x_j}^1\right\}\right)$$

$v_1 = x_k$

for $i = 2 : C_{max}$

$$P_{x_j}^i = P_{x_j}^{i-1} - P_{v_{i-1}}^{i-1} \exp\left(-\frac{\left\|x_j - v_{i-1}\right\|_A^2}{r_2^2}\right) \forall j \in [1,n]$$

for $j = 1 : n$

if $P_{v_i}^i / P_{v_1}^1 > \varepsilon_a$

Accept v_i as a cluster center.

else

end

if $\varepsilon_r < P_{v_i}^i / P_{v_1}^1 < \varepsilon_a$

$d_{min} = \min_{j, j \neq i}\left\{\left\|v_j - v_i\right\|_A^2\right\}$

if $(d_{min}/r_1) + \left(P_{v_i}^i / P_{v_1}^1\right) \geq 1$

Accept v_i as a cluster center.

break

else

$P_{v_i}^i := 0$

end

else

end

end

end

Outputs : V

2.16. Density based spatial clustering of applications with noise (DBSCAN) algorithm

Density Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm is the most popular density based clustering algorithm that finds both clusters with arbitrary shapes in the data with noise and outliers, and

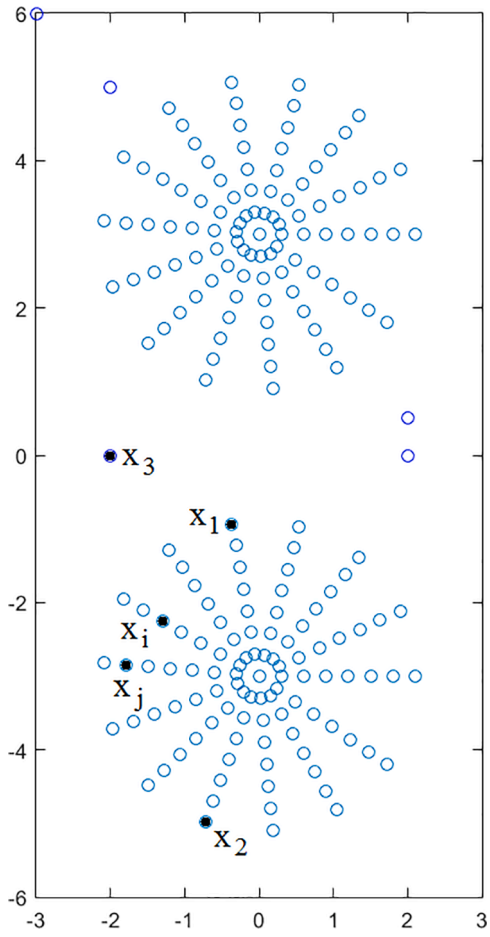

Fig. 1. Definition of border point, core point, and noise point in the DBSCAN algorithm.

number of clusters within the data (Yu et al., 2019). The DBSCAN algorithm is applicable to high dimensional data and is nonparametric since makes no model assumptions (Xu, Wang, Zhuang, & Gao, 2019). The distance between data points x_i and x_j is defined as d_{ij}, which is any arbitrary distance, e.g., Euclidean distance, Manhattan distance, Mahalanobis distance, etc. The ε-neighborhood of the point x_j denoted by $N_\varepsilon(x_j)$ includes all data points x_i whose distance from x_j is less than ε, which is defined as $N_\varepsilon(x_j) = \{x_i | d_{ij} \leq \varepsilon\}$. Density of x_j is cardinality of the set $N_\varepsilon(x_j)$, which is defined as $\rho(x_j) = |N_\varepsilon(x_j)|$. Different types of data points are recognized by the DBSCAN algorithm as shown in Fig. 1. If density of x_j is greater than or equal to a threshold ρ_{min} $(\rho(x_j) \geqslant \rho_{min})$ it is a core point (located inside the cluster, e.g., x_i and x_j in Fig. 1); and it is a border point if its density is less than ρ_{min} $(\rho(x_j) < \rho_{min})$ but there is at least one core point in its ε-neighborhood (located at the border line of the cluster, e.g., x_1 and x_2 in Fig. 1; and it is a noise point if its density is less than ρ_{min} $(\rho(x_j) < \rho_{min})$ and there is no core point in its ε-neighborhood, i.e., x_3 in Fig. 1. A cluster is a set of core and border points, and the data points that belong to no cluster are noise (Boonchoo et al., 2019). The DBSCAN algorithm is as follows where all data points are initially labeled with zero, which means they are not processed (Luchi, Loureiros Rodrigues, & Miguel Varejão, 2019). Noise and outliers are labeled with -1. The data points belonging to the cluster C are labeled with C. Note that, contrary to the previous algorithms, the DBSCAN algorithm does not calculate cluster centers and instead labels the data points as the members of different clusters.

DBSCAN algorithm
Inputs : $X, \rho_{min}, \varepsilon$
$C := 0$
for $j = 1 : n$
  $label(x_j) := 0$
end
for $j = 1 : n$
  if $label(x_j) = 0$
    $N = N_\varepsilon(x_j)$
    if $\rho(x_j) < \rho_{min}$
      $label(x_j) = -1 \therefore x_j$ is noise or outlier.
    else
      $C := C + 1$
      $label(x_j) = C$
      $S = N - \{x_j\}$; Removing x_j from N.
      Delete N
      for each $x_q \in S$
        if $label(x_q) = -1$
          $label(x_q) = C$
        elseif $label(x_q) = 0$
          $label(x_q) = C$
          $N = N_\varepsilon(x_q)$
          if $\rho(x_q) \geq \rho_{min}$
            $S := S \cup N$; Union S and N.
          end
        end
        Delete N
      end
      Delete S
    end
  end
end
Outputs : $label(x_j) \forall j \in [1, n]$

The main problem with this algorithm is determination of the input parameters ε and ρ_{min} for which the k-dist method is proposed in (Ester, Kriegel, Sander, & Xu, 1996). In this method, for each data point x_j, its k-nearest distance from its neighboring data points is calculated, which is called k-dist. These k-dist values are then sorted in descending order and are plotted versus the data points. It is performed for different values of $k = 1, 2, 3, \ldots$. Stable value of k (k_{st}) is reached when the curve is stable (Xu et al., 2019) from which $\rho_{min} = 1 + k_{st}$. The value of k-dist in the stable curve at the first break point of the curve (which is called valley) is chosen as ε as shown in Fig. 2. This procedure is much similar to the well-known elbow method, which is a heuristic approach to determine number of clusters in a data set.

2.17. Gaussian mixture models (GMM)

The ith cluster in the GMM algorithm is represented by a parametric distribution $p(x|v_i, S_i)$ and the entire data is simulated by a mixture of these distributions $P(x|\Theta)$ where μ_i and S_i are center and covariance matrix of the ith cluster (Li, Zhang, He, Tian, & Wei, 2019). The distributions and the mixture are as follows (He & Ho, 2019; Zeng & Cheung, 2014).

$$p(x|v_i, S_i) = \frac{1}{\sqrt{(2\pi)^r |S_i|}} exp\left(-\frac{1}{2}(x - v_i)' S_i^{-1} (x - v_i) \right) \tag{39}$$

$$P(x|\Theta) = \sum_{i=1}^{c} \mu_i p(x|v_i, S_i), \sum_{i=1}^{c} \mu_i = 1, \mu_i \in [0, 1] \tag{40}$$

where μ_i are the mixing coefficients and $\Theta = \{\mu_i, v_i, S_i, i \in [1, c]\}$ are parameters of the mixture that should be estimated and c is the number of clusters, which is the same as the number of distributions (Li et al., 2019). Parameters of the GMM algorithm are calculated using the Maximum Likelihood Estimation (MLE) method. Log-likelihood of the entire observed data X is as follows (Gebru, Alameda-Pineda, Forbes, & Horaud, 2016).

$$L(X|\Theta) = \sum_{j=1}^{n} ln\left(\sum_{i=1}^{c} \mu_i p(x_j|v_i, S_i) \right), \sum_{i=1}^{c} \mu_i = 1 \tag{41}$$

Zeroing derivative of (41) with respect to v_i leads to the flowing equation from which v_i is calculated.

$$\sum_{j=1}^{n} \frac{\mu_i p(x_j|v_i, S_i)}{\sum_{k=1}^{c} \mu_k p(x_j|v_k, S_k)} S_i^{-1}(x_j - v_i) = 0 \Rightarrow \sum_{j=1}^{n} G_{ij}(x_j - v_i) = 0 \Rightarrow$$

$$v_i = \frac{\sum_{j=1}^{n} G_{ij} x_j}{\sum_{j=1}^{n} G_{ij}} \tag{42}$$

where

$$G_{ij} = \frac{\mu_i p(x_j|v_i, S_i)}{\sum_{k=1}^{c} \mu_k p(x_j|v_k, S_k)} \tag{43}$$

Using the Lagrange Multipliers method for the constraint $\sum_{i=1}^{c} \mu_i = 1$, other parameters of the GMM algorithm are calculated as follows (Gebru et al., 2016).

$$S_i = \frac{\sum_{j=1}^{n} G_{ij}(x_j - v_i)(x_j - v_i)'}{\sum_{j=1}^{n} G_{ij}} \tag{44}$$

$$\mu_i = \frac{\sum_{j=1}^{n} G_{ij}}{n} \tag{45}$$

The algorithm consists of two steps. The E-step (Expectation-step) where G_{ij} are calculated from (43) and the M-step (Maximization-step) where the likelihood is maximized by calculating v_i, μ_i and S_i, respectively from (42), (45), and (44). Therefore, the entire GMM algorithm is an Expectation Maximization (EM) algorithm. Since the GMM algorithm assumes that the data are generated by several distributions $p(x|v_i, S_i)$,

the EM algorithm aims at estimating parameters of the distributions according to the available data X to make the distributions more likely to generate the data. Parameters of the algorithm are repeatedly updated by equations (42)–(45) until convergence. The algorithm can be initialized by choosing G_{ij} randomly for the first iteration.

The GMM algorithm can be used for both spherical and non-spherical clusters and provides clusters with different probabilities whereas the K-Means algorithm is only suitable for spherical clusters with identical probabilities. Lacking a standard method of determining type of the distributions and overfitting the data are the main drawbacks of the GMM algorithm.

We applied the GMM algorithm to several data sets and found that it is very sensitive to initialization and provides coincident clusters in many cases. Our study shows that performance of the GMM algorithm improves if the FCM algorithm is first applied to the data and G_{ij} is initialized with the partition matrix U of the FCM algorithm $G_{ij} = u_{ij}$.

GMM algorithm

Inputs : $X, c, \varepsilon, m, \tau$

$[U, V] = \mathrm{FCM}(X, c, \varepsilon, m, \tau)$

$G = U$

for $t = 1 : \tau$

$$v_i = \frac{\sum_{j=1}^{n} G_{ij} x_j}{\sum_{j=1}^{n} G_{ij}} \quad \forall i \in [1, c]$$

$$\mu_i = \frac{\sum_{j=1}^{n} G_{ij}}{n} \quad \forall i \in [1, c]$$

$$S_i = \frac{\sum_{j=1}^{n} G_{ij} (x_j - v_i)(x_j - v_i)'}{\sum_{j=1}^{n} G_{ij}} \quad \forall i \in [1, c]$$

$$p(x_j | v_i, S_i) = \frac{1}{\sqrt{(2\pi)^r |S_i|}} \exp\left(-\frac{1}{2} (x_j - v_i)' S_i^{-1} (x_j - v_i) \right) \quad \forall i \in [1, c], \forall j \in [1, n]$$

$$G_{ij} = \frac{\mu_i p(x_j | v_i, S_i)}{\sum_{k=1}^{c} \mu_k p(x_j | v_k, S_k)} \quad \forall i \in [1, c], \forall j \in [1, n]$$

if $\left\| V^{(t+1)} - V^{(t)} \right\| \le \varepsilon$

Stop

else

end

end

Outputs : V

2.18. Spectral clustering algorithm

Clustering algorithms as the basic tools for computer vision, machine learning, signal processing, image processing, data mining, etc., are divided into two main groups. Hierarchical clustering that includes single link and complete link algorithms and partitional clustering that includes square error algorithms, e.g., FCM and K-Means algorithms, graph theoretic algorithms, i.e., Spectral clustering, mixture models, i.e., GMM algorithm, and density based algorithms, i.e., DBSCAN algorithm. As mentioned above, the main objective of the clustering algorithms is to partition the data into some groups based on similarities and dissimilarities of the data points by maximizing within-group similarities and minimizing between-group similarities. As a graph theoretic algorithm, the spectral clustering carries out this process by defining an undirected weighted similarity graph where the data points are vertices of the graph and weight of the edge connecting two vertices implies similarities of their corresponding data points. The spectral clustering algorithm aims at maximizing within-group connections and minimizing between-group connections to achieve the basic clustering goals (von Luxburg, 2007). The similarity graph is defined as $G = (X, W)$ where $X = \{x_j \in X \forall j \in [1, n]\}$ is the matrix of graph vertices and $W = [w_{ij}] \forall i, j \in [1, n]$ where $w_{ij} \geqslant 0$ is weight of the edge connecting vertices x_i and x_j, and n is the number of data points. Therefore, the similarity matrix W should be calculated to maximize within-group weights and minimize between-group weights for which several methods are used (von Luxburg, 2007). If $w_{ij} \neq 0; x_i$ and x_j are connected and if $w_{ij} = 0; x_i$ and x_j are not connected. Note that diagonal elements of the similarity matrix are usually set to zero $w_{ii} = 0 \ \forall i \in [1, n]$ and this matrix is symmetric $w_{ij} = w_{ji} \ \forall i.j \in [1, n]$.

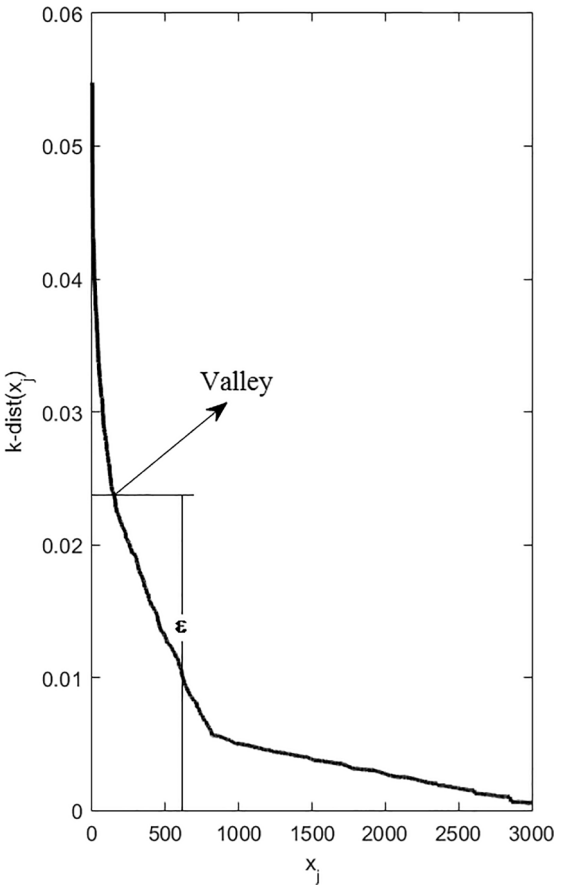

Fig. 2. Determination of ε from k-dist graph in the DBSCAN algorithm.

Different similarity matrices are used for spectral clustering. The ε-neighborhood method connects every pair of data points with pairwise distance of less than ε, that obviously leads to an undirected graph. The k-nearest neighbors approach connects vertex x_j to vertex x_i if x_i is among the k-nearest neighbors of x_j. Since x_j may not be among k-nearest neighbors of x_i, the resulting similarity graph is generally directed. Two methods are used to make this graph undirected. In the

first method, x_i and x_j are connected if x_i is among k-nearest neighbors of x_j or x_j is among k-nearest neighbors of x_i. Such a graph is called the k-nearest neighbors graph. In the second method, x_j and x_j are connected if x_i is among k-nearest neighbors of x_j and x_j is among k-nearest neighbors of x_i. Such a graph is called the mutual k-nearest neighbors graph. In both methods, weights of the edges between pairwise connected vertices is calculated according to their similarity. The fully connected graph connects all vertices and the similarity matrix is calculated by the Gaussian function $w_{ij} = exp\left(- \| x_i - x_j \|_A^2 / 2\sigma^2 \right)$ where σ controls width of the Gaussian function that may be calculated as the average pairwise distance between all data points $\sigma^2 = \frac{1}{2n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\| x_i - x_j \|_A^2$ (Li, Hu, Stojmenovic, Liu, & Liu, 2020). In some cases, cosine similarity measure is used instead of the Gaussian function (Janani & Vijayarani, 2019). Experiments show that the k-nearest neighbors graph is a better choice and it is recommend to try this one first when implementing the algorithm (von Luxburg, 2007). Then main problem with these approaches is selection of proper values of the parameters $\varepsilon; k$ and σ. If proper value of k is determined by any kind of investigations, σ can be calculated as the mean distance of a point to its k-nearest neighbor; and if ε is determined by any means, $\sigma = \varepsilon$ is a good choice for fully connected graph (von Luxburg, 2007).

Degree matrix of the similarity graph is defined as follows, which is diagonal and plays an important role in the spectral clustering algorithms (Hu et al., 2020).

$$\begin{cases} d_{ii} = \sum_{j=1}^{n} w_{ij} \ \forall i \in [1,n] \\ d_{ij} = 0 \ \forall i \neq j \in [1,n] \end{cases} \tag{46}$$

Laplacian matrix is the dominant factor in spectral clustering. There are three types of Laplacian matrix including unnormalized Laplacian matrix, symmetric Laplacian matrix, and random walk Laplacian matrix that the latter two matrices are normalized (von Luxburg, 2007). The unnormalized Laplacian matrix L_{un} is defined as follows.

$$L_{un} = D - W \tag{47}$$

The normalized Laplacian matrix including the symmetric Laplacian matrix L_{sym} and the random walk Laplacian matrix L_{rw} are defined as follows (von Luxburg, 2007).

$$L_{sym} = D^{-\frac{1}{2}} L_{un} D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}} \tag{48}$$

$$L_{rw} = D^{-1} L_{un} = I - D^{-1} W \tag{49}$$

All three types of the Laplacian matrix are positive semi-definite and have n real and positive eigenvalues and their smallest eigenvalue is zero that corresponds to an eigenvector with identical elements (von Luxburg, 2007).

If the similarity matrix is calculated by k-nearest neighbors approach, multiplicity of zero eigenvalues in both unnormalized and normalized Laplacian matrices indicates number of connected components or clusters in the graph. For example, c zero eigenvalues indicates that there are c connected components (clusters) in the graph. On the other hand, if the similarity matrix is computed by the fully connected graph approach, e.g., Gaussian or cosine function, when the eigenvalues are sorted, there will be an obvious difference (namely eigengap) between the eigenvalues λ_c and λ_{c+1} where c is the number of connected components (clusters). In fact, value of i that maximizes the eigengap $\Delta_i = \lambda_{i+1} - \lambda_i$ of the sorted eigenvalues is a good choice for the number of clusters. It should be noted that the multiplicity of zero eigenvalues and the eigengap disappears when the clusters are not well-separated and have some degree of overlapping (von Luxburg, 2007). In the spectral clustering the similarity graph G is partitioned into c disjoint subgraphs (clusters) $A_i \subset G \ \forall i \in [1,c]$ such that $\cup_{i=1}^{c} A_i = G$ and $A_i \cap A_j = \emptyset \ \forall i \neq j \in [1,c]$. Complement of the subgraph A_i is defined as $\overline{A}_i = G - A_i$. A cut is defined as the sum of the weights of the edges connecting a subgraph to its complement. Lower value of a cut indicates that the subgraph is more disjoint from its complement and can be considered as a cluster. This is why the spectral clustering algorithm attempts to minimize the cut.

$$cut\left(A_i, \overline{A}_i\right) = \sum_{x_i \in A_i, x_j \in \overline{A}_i} w_{ij} \tag{50}$$

Therefore, objective function of the spectral clustering algorithm is defined as follows that should be minimized to find disjoint subgraphs (clusters) (von Luxburg, 2007).

$$J = \sum_{i=1}^{c} cut\left(A_i, \overline{A}_i\right) \tag{51}$$

A serious problem with (51) is that in many cases it separates an individual data point (vertex) from the graph as a cluster whereas a cluster should include relatively large number of data points. To avoid this problem, the objective function is defined based on ratio cut (J_{Rcut}) and normalized cut (J_{Ncut}) as follows (von Luxburg, 2007). Each of the objective functions leads to a different spectral clustering algorithm depending on the type of the Laplacian matrix.

$$J_{Rcut} = \sum_{i=1}^{c} \frac{cut\left(A_i, \overline{A}_i\right)}{|A_i|} \tag{52}$$

$$J_{Ncut} = \sum_{i=1}^{c} \frac{cut\left(A_i, \overline{A}_i\right)}{vol(A_i)} \tag{53}$$

where $|A_i|$ is the number of vertices in the subgraph A_i that measures size of A_i with the number of data points in this subgraph and $vol(A_i) = \sum_{x_i \in A_i} d_{ii} = \sum_{x_i \in A_i} \sum_{j=1}^{n} w_{ij}$ that measures size of A_i by summing weights of all edges attached to this subgraph. Ratio cut and normalized cut makes the spectral clustering algorithms more robust when dealing with outliers.

Minimizing these objective functions is a NP-hard problem. Therefore, relaxed versions of these problems are solved in spectral clustering. Relaxing J_{Rcut} leads to unnormalized spectral clustering and relaxing J_{Ncut} results normalized spectral clustering (von Luxburg, 2007). It is shown that J_{Rcut} can be relaxed as the following optimization problem (von Luxburg, 2007).

$$J_{Rcut} = tr(H^{'} L_{un} H) \text{ subject to } H^{'} H = I \tag{54}$$

According to the Rayleigh-Ritz theorem, solution of this equation is the spectrum matrix H whose columns are the first c eigenvectors of the unnormalized Laplacian matrix L_{un} ordered according to the size of the their corresponding eigenvalues $\lambda_1 \leqslant \lambda_2 \leqslant \ldots \leqslant \lambda_c$, where c is the number connected components or clusters. So, the UNnormalized Spectral Clustering algorithm (UNSC) is as follows (von Luxburg, 2007).

UNSC algorithm

Inputs: X, c

1. Construct a similarity graph by the similarity matrix W using one of the methods discussed above.
2. Compute the unnormalized Laplacian matrix L_{un}.
3. Compute the first c eigenvectors of L_{un} : s_1, s_2, $\cdots$, s_c.
4. Let the spectrum matrix $S \in R^{n \times c}$ be the matrix containing the eigenvectors s_1, s_2, $\cdots$, s_c as columns.
5. Consider each row of the S matrix as a data point and cluster this matrix into c clusters by an appropriate clustering algorithm such as K-Means, GMM, FCM, etc.
6. Assign the data point x_j to the ith cluster if the jth row of S is assigned to this cluster.
7. Outputs: U, V

   where U and V are partition matrix and cluster centers matrix, respectively.

It is shown that J_{Ncut} can be relaxed as follows with L_{rw} as the Laplacian matrix (von Luxburg, 2007).

$$J_{Ncut} = tr(H^{'} L_{rw} H) \text{ subject to } H^{'} DH = I \tag{55}$$

According to the Rayleigh-Ritz theorem, solving this equation yields the spectrum matrix H that contains first c eigenvectors of the Laplacian matrix L_{rw} as columns ordered according to the size of the their corresponding eigenvalues $\lambda_1 \leqslant \lambda_2 \leqslant \ldots \leqslant \lambda_c$, where c is the number of clusters. Using the symmetric Laplacian matrix L_{sym}, the problem is relaxed as follows where $T = D^{\frac{1}{2}} H$ (von Luxburg, 2007).

$$J_{Ncut} = tr\left(T^{'} D^{-\frac{1}{2}} L_{sym} D^{-\frac{1}{2}} T \right) \text{ subject to } T^{'} T = I \tag{56}$$

According to the Rayleigh-Ritz theorem, solving this equation yields the spectrum matrix $T^{n \times c}$ whose columns are the first c eigenvectors of the Laplacian matrix L_{sym} as columns ordered according to the size of the their corresponding eigenvalues $\lambda_1 \leqslant \lambda_2 \leqslant \ldots \leqslant \lambda_c$, where c is the number of clusters. So, the Normalized Spectral Clustering algorithm (NSC) is as follows (von Luxburg, 2007).

NSC algorithm with L_{rw}

Inputs: X, c

1. Construct a similarity graph by the similarity matrix W using one of the methods discussed above.
2. Compute the normalized Laplacian matrix L_{rw}.
3. Compute the first c eigenvectors of L_{rw} : $s_1, s_2, \cdots, s_c$.
4. Let the spectrum matrix $S \in R^{n \times c}$ be the matrix containing the eigenvectors $s_1, s_2, \cdots, s_c$ as columns.
5. Consider each row of the S matrix as a data point and cluster this matrix into c clusters by an appropriate clustering algorithm such as K-Means, GMM, FCM, etc.
6. Assign the data point x_j to the ith cluster if the jth row of S is assigned to this cluster.
7. Outputs: U, V

   where U and V are partition matrix and cluster centers matrix, respectively.

NSC algorithm with L_{sym}

Inputs: X, c

1. Construct a similarity graph by the similarity matrix W using one of the methods discussed above.
2. Compute the normalized Laplacian matrix L_{sym}.
3. Compute the first c eigenvectors of L_{sym} : $s_1, s_2, \cdots, s_c$.
4. Let the spectrum matrix $S \in R^{n \times c}$ be the matrix containing the eigenvectors $s_1, s_2, \cdots, s_c$ as columns.
5. Normalize rows of the matrix $S \in R^{n \times c}$ to $T \in R^{n \times c}$ by the equation $t_{ij} = \frac{s_{ij}}{\sqrt{\sum_{j=1}^{c} s_{ij}^2}} \; \forall i \in [1,n], j \in [1,c]$.
6. Consider each row of the normalized spectrum matrix T as a data point and cluster T into c clusters by an appropriate clustering algorithm such as K-Means, GMM, FCM, etc.
7. Assign the data point x_j to the ith cluster if the jth row of T is assigned to this cluster.
8. Outputs: U, V

   where U and V are partition matrix and cluster centers matrix, respectively.

So, unlike the other algorithms, the normalized algorithm with L_{sym} needs an additional row normalization step. Experiments show that the normalized algorithms perform better than the unnormalized algorithm and the normalized algorithm with random walk Laplacian matrix L_{rw} is superior to the one with symmetric Laplacian matrix L_{sym} (von Luxburg, 2007). The normalized algorithms minimize between-cluster similarities and maximize within-cluster similarities whereas the unnormalized algorithm only minimizes between-cluster similarities.

The K-Means, GMM, FCM and many other algorithms are suited for round clusters but the spectral clustering algorithm is suitable for both round clusters and clusters with arbitrary shapes, sizes and densities. The spectral clustering algorithm transforms the clusters of general shapes into compact and round clusters in the spectrum space (eigenvectors) where K-Means and other algorithms work properly. Because of this, first the spectral algorithm is applied to the data and the resulting spectrum is then clustered by the other algorithms.

In the present work, the FCM algorithm is employed to cluster the spectrum matrix S (or T) that yields the partition matrix U. The final cluster centers are then calculated using the partition matrix $U = [u_{ij}]$ and the data matrix X from (3) as $v_i = \left(\sum_{j=1}^{n} u_{ij}^m x_j / \sum_{j=1}^{n} u_{ij}^m \right) \; \forall i \in [1,c]$.

2.19. Outlier Removal Clustering (ORC) algorithm

K-Means algorithm is hard version of the FCM algorithm where each data point belongs only to one cluster whereas in the FCM algorithm each data points belongs to all clusters to some degree, which is represented by the partition matrix U. Objective function of the K-Means algorithm is as follows (Lei et al., 2012).

$$J = \sum_{j=1}^{n} \sum_{i=1}^{c} u_{ij} \| x_j - v_i \|_A^2 \tag{57}$$

where u_{ij} is one when the data point x_j belongs to the ith cluster and is zero when it does not belong to this cluster. Therefore, the partition matrix and cluster centers of the K-Means algorithm are as follows (Lei et al., 2012).

$$u_{ij} = \begin{cases} 1 & \| x_j - v_i \|_A^2 \leqslant \| x_j - v_k \|_A^2 \; \forall k \neq j \in [1,c] \\ 0 & otherwise \end{cases} \qquad v_i = \frac{\sum_{j=1}^{n} u_{ij} x_j}{\sum_{j=1}^{n} u_{ij}} \tag{58}$$

K-Means algorithm starts with the random partition matrix $U = rand(c,n)$ and then (58) is repeatedly updated until convergence of the algorithm.

K - Means algorithm

Inputs : X, c, τ, ε

$U = \text{round}(\text{rand}(c,n))$

for $t = 1 : \tau$

$$v_i = \frac{\sum_{j=1}^{n} u_{ij} x_j}{\sum_{j=1}^{n} u_{ij}} \; \forall i \in [1,c]$$

$$u_{ij} = \begin{cases} 1 & \left\| x_j - v_i \right\|_A^2 \leq \left\| x_j - v_k \right\|_A^2 \; \forall k \neq i \in [1,c] \\ 0 & \text{otherwise} \end{cases} \quad \forall i \in [1,c], \forall j \in [1,n]$$

if $\left\| V^{(t+1)} - V^{(t)} \right\| \leq \varepsilon$

Stop

else

end

end

Outputs : U, V

K-Means algorithm is highly sensitive to noise and outliers. Outlier Removal Clustering (ORC) algorithm is proposed to overcome this shortcoming (Hautamaki, Cherednichenko, Karkkainen, Kinnunen, & Franti, 2005). In this algorithm, the data are first clustered by the K-Means algorithm and then degree of outlyingness o_j of data point x_j is calculated by the maximum distance of all data points to the cluster centers d_{max}. Note that

$u_{ij} = 1$ indicates that x_j belongs to the ith cluster A_i.

$$o_j = \frac{\| x_j - v_i \|_A^2}{d_{max}} \Big| x_j \in A_i$$
$$d_{max} = \max_j \left\{ \| x_j - v_i \|_A^2 \Big| x_j \in A_i \right\} \quad (59)$$

Data points with outlyingness larger than a user defined threshold o_{max} are recognized as outlier and are removed from the data. The outlier removal is carried out multiple times T as shown in the following algorithm. The hyperparameters o_{max} and T significantly affect results of the algorithm and the main drawbacks with the ORC algorithm is lack of a standard method for determination of o_{max} and T.

ORC algorithm
Inputs : X, c, τ, ε, T, o_{max}
[U, V] = K - Means(X, c, τ, ε)
for t = 1 : T
  $d_{max} = \max_j \left\{ \|x_j - v_i\|_A^2 \Big| x_j \in A_i \right\}$
  for j = 1 : n
    $o_j = \frac{\|x_j - v_i\|_A^2}{d_{max}} \Big| x_j \in A$
    if $o_j > o_{max}$
      $X := X - \{x_j\}$
    else
    end
  end
  [U, V] = K - Means(X, c, τ, ε)
end
Outputs : U, V

3. Revised Fuzzy C-Means (RFCM)

The RFCM algorithm aims at two objectives; preventing large clusters from attracting centers of small clusters and eliminating impacts of noise and outliers on the cluster centers. To this purpose, two objective functions are used one for determining initial cluster centers by a size-insensitive approach and the other for calculating final cluster centers with a noise-resistant approach. Minimizing the first objective function provides cluster centers by which the second part of the algorithm starts. The following objective function is used for size-insensitive part of the RFCM algorithm.

$$J = \sum_{j=1}^{n} \sum_{i=1}^{c} u_{ij}^m \| x_j - v_i \|_A^2, \sum_{k=1}^{c} u_{kj} = \rho_j(X, U) \quad (60)$$

where ρ_j is used to reduce the interactions caused by inequality of size of the clusters and is calculated as follows.

$$\rho_j = 1 - S_i, i = \underset{r \in [1,c]}{arg\,max} (u_{rj}) \quad (61)$$

The equation $i = \underset{r \in [1,c]}{argmax}(u_{rj})$ determines i, which is index of the cluster where the data point x_j attains its maximum membership grade and hence belongs to this cluster. Relative size of the ith cluster S_i is defined as follows.

$$S_i = \frac{1}{|X|} \sum_{x_j \in A_i} \left(1 + \frac{u_{ij}}{|X|^p} \right) \quad (62)$$

where p is a sufficiently large positive number. Note that the relative size S_i defined in (62) tends to the relative size defined in (34) for large values of p but they are basically different since the one defined in (62) is differentiable with respect to u_{ij} in contrast to the one defined in (34).

According to (61), ρ_j depends on the relative size S_i of the cluster that the data point x_j belongs to. Index of this cluster is determined by $i = arg\,\underset{r \in [1,c]}{max}(u_{rj})$ and according to (62), S_i depends on the data matrix X and size of the cluster A_i, which is the number of data points that attain their maximum membership grade in this cluster and is taken into account by the partition matrix $U = [u_{ij}]$. So, ρ_j is a function of X and U, which is mentioned in (60) by $\rho_j(X, U)$. Using the Lagrange Multipliers method (39) is written as follows.

$$J = \sum_{j=1}^{n} \sum_{i=1}^{c} u_{ij}^m \| x_j - v_i \|_A^2 + \sum_{j=1}^{n} \lambda_j \left(\sum_{k=1}^{c} u_{kj} - \rho_j \right)$$

where λ_j are the Lagrange multipliers. The initial cluster centers are computed as follows by zeroing derivative of this function with respect to v_i.

$$\frac{\partial J}{\partial v_i} = \sum_{j=1}^{n} u_{ij}^m (A + A')(x_j - v_i) = 0 \Rightarrow$$
$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m} \quad (63)$$

Zeroing derivative of this function with respect to u_{ij} results the following membership grades.

$$\frac{\partial J}{\partial u_{ij}} = m u_{ij}^{m-1} \| x_j - v_i \|_A^2 + \lambda_j \left(1 - \frac{\partial \rho_j}{\partial u_{ij}} \right) = 0 \Rightarrow$$
$$u_{ij} = \left(\frac{-\lambda_j \left(1 - \frac{\partial \rho_j}{\partial u_{ij}} \right)}{m f \left(\| x_j - v_i \|_A^2 \right)} \right)^{\frac{1}{m-1}}, \sum_{k=1}^{c} u_{kj} = \rho_j \Rightarrow$$
$$u_{ij} = \rho_j \left[\sum_{k=1}^{c} \left(\frac{\left(1 - \frac{\partial \rho_j}{\partial u_{kj}} \right) \| x_j - v_i \|_A^2}{\left(1 - \frac{\partial \rho_j}{\partial u_{ij}} \right) \| x_j - v_k \|_A^2} \right)^{\frac{1}{m-1}} \right]^{-1} \quad (64)$$

Since ρ_j depends on the relative size of the clusters S_i that is ratio of the number of data points which gain their maximum membership grades u_{ij} in that cluster to the total number of data points $|X|$; ρ_j depends on the partition matrix U however $\partial \rho_j / \partial u_{ij}$ is mathematically difficult since the relationship cannot be expressed easily. However, defining size of the clusters as (62) and adopting a large value of P, $(u_{ij}/|X|^p) \to 0$ and $\sum_{x_j \in A_i} (1 + u_{ij}/|X|^p)$ practically converts to the number of data points in the ith clusters. The term $\partial \rho_j / \partial u_{ij}$ is mistakenly set to zero in the siibFCM and csiFCM algorithms because of lacking explicit representation of ρ_j in terms of X and U whereas ρ_j is a function of $U = [u_{ij}]$ and X. However, using (62) $\partial \rho_j / \partial u_{ij}$ is calculated as follows.

$$\frac{\partial \rho_j}{\partial u_{ij}} = -\frac{\partial S_i}{\partial u_{ij}} = \begin{cases} -\frac{1}{|X|^{p+1}} & if\ i = \underset{r \in [1,c]}{arg\,max}(u_{rj}) \\ 0 & Otherwise \end{cases} \quad (65)$$

For large values of P, $\partial\rho_j/\partial u_{ij}$ tends to zero and (64) becomes the same as the update equation used for u_{ij} in the siibFCM and csiFCM algorithms. Note that the normalizing term $\max_{r\in[1,c]}(1-S_r)$ in the denominator of ρ_j in (34) is removed because this term makes $\partial\rho_j/\partial u_{ij}$ very complicated.

When update equations of the size-insensitive part of the RFCM algorithm converges the resulting cluster centers are employed to initialize the noise-resistant part of the algorithm in which a function of distance $f\left(\| x_j - v_i\|_A^2\right)$ is used instead of the distance itself $\| x_j - v_i\|_A^2$ to control impacts of noise and outliers on the cluster centers. Therefore, objective function of the noise-resistant part of the RFCM algorithm is defined as follows.

$$J = \sum_{j=1}^{n}\sum_{i=1}^{c} u_{ij}^m f\left(\| x_j - v_i\|_A^2\right), \sum_{k=1}^{c} u_{kj} = \phi_j(X,U) \tag{66}$$

The parameter $\phi_j(X,U)$ is calculated depending on the user choice. Cluster centers matrix and partition matrix of (66) are calculated using the Lagrange Multipliers method.

$$J = \sum_{j=1}^{n}\sum_{i=1}^{c} u_{ij}^m f\left(\| x_j - v_i\|_A^2\right) + \sum_{j=1}^{n}\lambda_j\left(\sum_{k=1}^{c} u_{kj} - \phi_j\right)$$

Zeroing derivative of J with respect to v_i yields the following update equation for the cluster centers.

$$\frac{\partial J}{\partial v_i} = \sum_{j=1}^{n} u_{ij}^m f'\left(\| x_j - v_i\|_A^2\right)(A + A')(x_j - v_i) = 0 \Rightarrow$$

$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^m f'\left(\| x_j - v_i\|_A^2\right) x_j}{\sum_{j=1}^{n} u_{ij}^m f'\left(\| x_j - v_i\|_A^2\right)} \tag{67}$$

Zeroing derivative of J with respect to u_{ij} results the following equation for the membership grades.

$$\frac{\partial J}{\partial u_{ij}} = 0 \Rightarrow m u_{ij}^{m-1} f\left(\| x_j - v_i\|_A^2\right) + \lambda_j\left(1 - \frac{\partial\phi_j}{\partial u_{ij}}\right) = 0 \Rightarrow$$

$$u_{ij} = \left(\frac{-\lambda_j\left(1 - \frac{\partial\phi_j}{\partial u_{ij}}\right)}{m f\left(\| x_j - v_i\|_A^2\right)}\right)^{\frac{1}{m-1}}, \sum_{k=1}^{c} u_{kj} = \phi_j \Rightarrow$$

$$u_{ij} = \phi_j\left[\sum_{k=1}^{c}\left(\frac{\left(1 - \frac{\partial\phi_j}{\partial u_{kj}}\right) f\left(\| x_j - v_i\|_A^2\right)}{\left(1 - \frac{\partial\phi_j}{\partial u_{ij}}\right) f\left(\| x_j - v_k\|_A^2\right)}\right)^{\frac{1}{m-1}}\right]^{-1} \tag{68}$$

Note that in this paper $\phi_j = 1\ \forall j \in [1,n]$ and consequently $\partial\phi_j/\partial u_{ij} = \partial\phi_j/\partial u_{kj} = 0\ \forall j \in [1,n]$. However, the update equation is given in general form for possible future developments.

It is obvious that the RFCM algorithm converts to the FCM algorithm with $f(x) = x$. Since f' is used in the update equation of the cluster centers (67), the following exponential function f with the exponential derivative f' is employed to damp impacts of noise and outliers on the cluster centers.

$$f\left(\| x_j - v_i\|_A^2\right) = 1 - exp\left(-\frac{\| x_j - v_i\|_A^2}{\omega_i^2}\right) \tag{69}$$

$$f'\left(\| x_j - v_i\|_A^2\right) = \frac{1}{\omega_i^2} exp\left(-\frac{\| x_j - v_i\|_A^2}{\omega_i^2}\right) \tag{70}$$

where ω_i is bandwidth of the exponential function. Since the outliers and most of the noise points are far away from the cluster centers, the distance $\| x_j - v_i\|_A^2$ is relatively larger for such points as compared to the data points close to the cluster centers and $f'\left(\| x_j - v_i\|_A^2\right)$ in (67) becomes negligible according to (70) and consequently impacts of these points on the cluster centers are eliminated. However, according to (3) that represents update equation of the cluster centers in the FCM algorithm, contributions of noise and outliers are not controlled and they are treated the same as the actual data points that results misplaced cluster centers. One may use any other proper function with strong damping strength instead of the above exponential function such as $f(x) = 1/(1+x), tanh(x), etc.$ Bandwidth of the exponential function is calculated as follows.

$$\omega_i^2 = \frac{\sum_{j=1}^{n} s_{ij}^m \| x_j - v_i\|_A^2}{\alpha\sum_{j=1}^{n} s_{ij}^m} \tag{71}$$

$$s_{ij} = \left[\sum_{k=1}^{c}\left(\frac{\| x_j - v_i\|_A^2}{\| x_j - v_k\|_A^2}\right)^{\frac{1}{m-1}}\right]^{-1} \tag{72}$$

where $\alpha > 0$ controls ω_i by limiting number of data points with high membership grades u_{ij} in the ith cluster center v_i. Note that s_{ij} is the same as u_{ij} of the FCM algorithm given in (3) and differs from the membership grades u_{ij} calculated by (68).

Degree of fuzziness m, adjusting parameter of bandwidth α, convergence criterion ε, number of clusters c, and the data matrix X are inputs to the RFCM algorithm. The size-insensitive part of the algorithm initializes by the random partition matrix $U = rand(c,n)$. When this part of the algorithm converges or iterates up to some number, the resulting cluster centers are employed to initialize the noise-resistant part of the algorithm, which yields the final cluster centers after convergence.

In most of the real world cases, features of the data have different physical units and variances and should be normalized to make the Euclidean distance meaningful and let all features to contribute effectively in the distance. Several methods are used for normalization. For instance, in standard normalization features are normalized to zero mean and unit variance and in domain standardization they are scaled to the interval $[0,1]$. Domain standardization is used in this paper where jth value of the qth feature x_{qj} is transformed to x_{qj}^* as follows (Linn et al., 2016). This method was previously used in (32) to normalize integrity of the clusters.

$$x_{qj}^* = \frac{x_{qj} - \min_{j\in[1,n]}\left(x_{qj}\right)}{\max_{j\in[1,n]}\left(x_{qj}\right) - \min_{j\in[1,n]}\left(x_{qj}\right)}\ \forall q \in [1,r] \tag{73}$$

The RFCM algorithm is applied through the following steps.

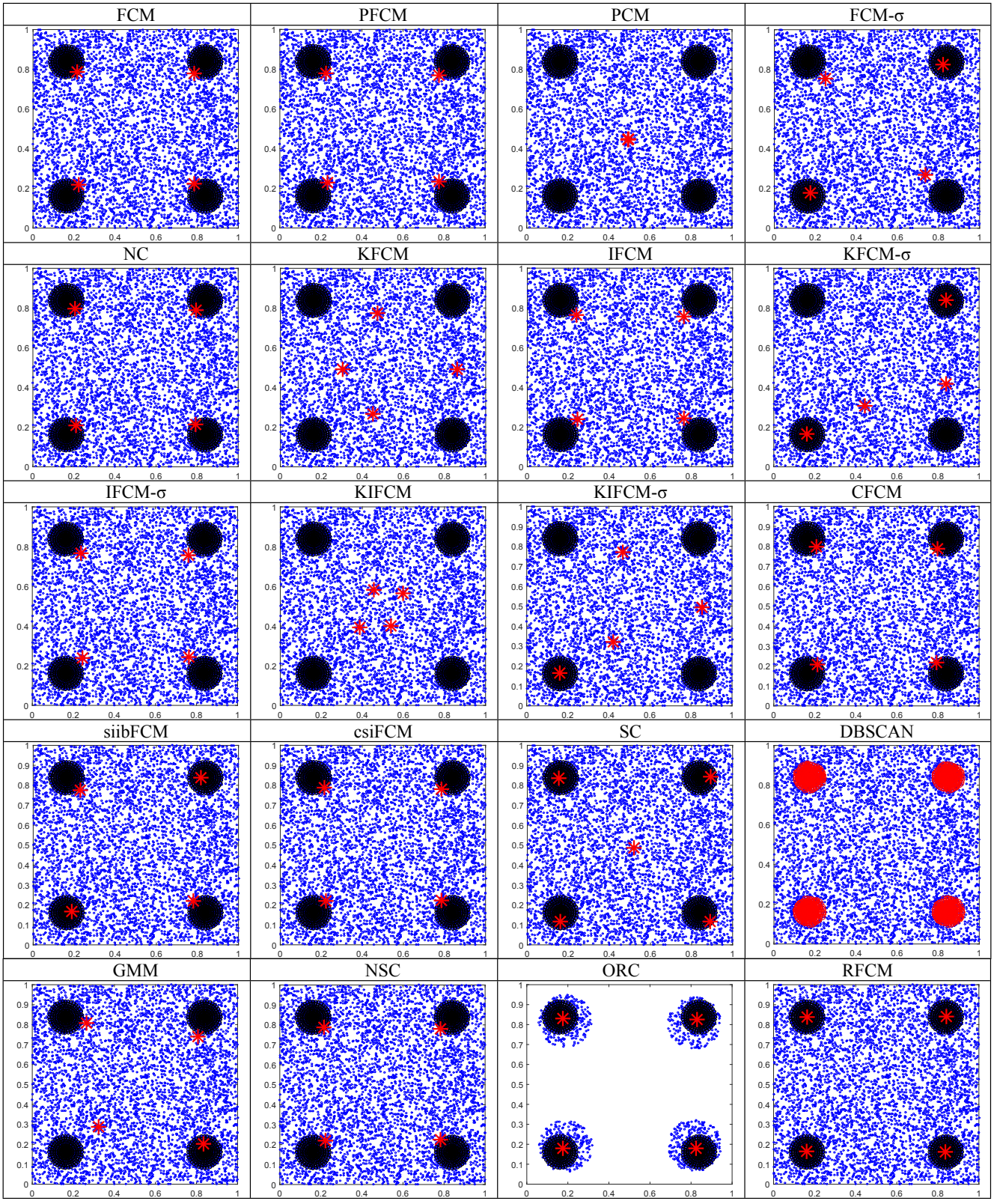

Fig. 3. Clustering noisy data with four identical clusters by different algorithms.

1) The input data are normalized using (73) or preprocessed with any other method or used unprocessed; and constants of the algorithm (α, c, m, p, and ε) are set based on the user discretion.
2) The algorithm starts with the random partition matrix $U = rand(c, n)$ and Eqs. (63) and (64) are updated until convergence that yields cluster centers of the size-insensitive part of the algorithm. Note that it is not necessary to let this part of the algorithm to converge completely. It can be terminated after 40–50 iterations to calculate initial cluster centers since the final cluster centers are determined by the noise-resistant part of the algorithm. Note that S_i, ρ_j, $\partial\rho_j/\partial u_{ij}$, and $\partial\rho_j/\partial u_{kj}$ are updated in every iteration.
3) Finally, the noise-resistant part of the algorithm initializes with the cluster centers computed in step 2 and Eqs. (67) and (68) are updated until convergence that results the final cluster centers. Note that ω_i, s_{ij}, ϕ_j, $\partial\phi_j/\partial u_{ij}$, and $\partial\phi_j/\partial u_{kj}$ are updated in each iteration.

In summary, the size-insensitive part of the RFCM algorithm (siRFCM) is applied to the data as follows.

siRFCM algorithm

Inputs : $X, c, \varepsilon, m, \tau, p$

$U = \text{rand}(c, n)$

for $t = 1 : \tau$

$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^{m} x_j}{\sum_{j=1}^{n} u_{ij}^{m}} \quad \forall i \in [1, c]$$

$$S_i = \frac{1}{|X|} \sum_{x_j \in A_i} \left(1 + \frac{u_{ij}}{|X|^p}\right) \quad \forall i \in [1, c]$$

$$i = \arg\max_{r \in [1,c]} \left(u_{rj}\right) \quad \forall j \in [1, n]$$

$$\rho_j = 1 - S_i \quad \forall j \in [1, n]$$

$$\frac{\partial \rho_j}{\partial u_{ij}} = \begin{cases} -\dfrac{1}{|X|^{p+1}} & \text{if } i = \arg\max_{r \in [1,c]} \left(u_{rj}\right) \\ 0 & \text{Otherwise} \end{cases} \quad \forall i \in [1, c], \forall j \in [1, n]$$

$$u_{ij} = \rho_j \left[\sum_{k=1}^{c} \left(\frac{\left(1 - \frac{\partial \rho_j}{\partial u_{kj}}\right) \left\| x_j - v_i \right\|_A^2}{\left(1 - \frac{\partial \rho_j}{\partial u_{ij}}\right) \left\| x_j - v_k \right\|_A^2} \right)^{\frac{1}{m-1}} \right]^{-1} \quad \forall i \in [1, c], \forall j \in [1, n]$$

if $\left\| V^{(t+1)} - V^{(t)} \right\| \le \varepsilon$

Stop

else

end

end

Outputs : U, V

Output of the size-insensitive part of the algorithm is used to initialize the noise-resistant part of the RFCM algorithm (nrRFCM) as follows.

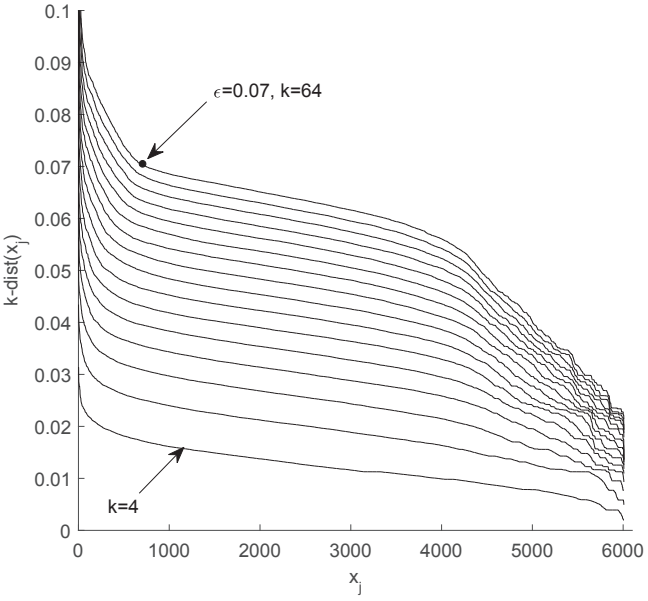

Fig. 4. The *k*-dist graph for noisy data with four clusters.

nrRFCM algorithm

Inputs : $X, c, \varepsilon, m, \alpha, \tau, p$

$[U, V] = \text{siRFCM}(X, c, \varepsilon, m, \tau, p)$

for $t = 1 : \tau$

$$s_{ij} = \left[\sum_{k=1}^{c} \left(\frac{\left\| x_j - v_i \right\|_A^2}{\left\| x_j - v_k \right\|_A^2} \right)^{\frac{1}{m-1}} \right]^{-1} \quad \forall i \in [1, c], \forall j \in [1, n]$$

$$\omega_i^2 = \frac{\sum_{j=1}^{n} s_{ij}^{m} \left\| x_j - v_i \right\|_A^2}{\alpha \sum_{j=1}^{n} s_{ij}^{m}} \quad \forall i \in [1, c]$$

$$u_{ij} = \varphi_j \left[\sum_{k=1}^{c} \left(\frac{\left(1 - \frac{\partial \varphi_j}{\partial u_{kj}}\right) f\left(\left\| x_j - v_i \right\|_A^2\right)}{\left(1 - \frac{\partial \varphi_j}{\partial u_{ij}}\right) f\left(\left\| x_j - v_k \right\|_A^2\right)} \right)^{\frac{1}{m-1}} \right]^{-1} \quad \forall i \in [1, c], \forall j \in [1, n]$$

$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^{m} f'\left(\left\| x_j - v_i \right\|_A^2\right) x_j}{\sum_{j=1}^{n} u_{ij}^{m} f'\left(\left\| x_j - v_i \right\|_A^2\right)} \quad \forall i \in [1, c]$$

if $\left\| V^{(t+1)} - V^{(t)} \right\| \le \varepsilon$

Stop

else

end

end

Outputs : U, V

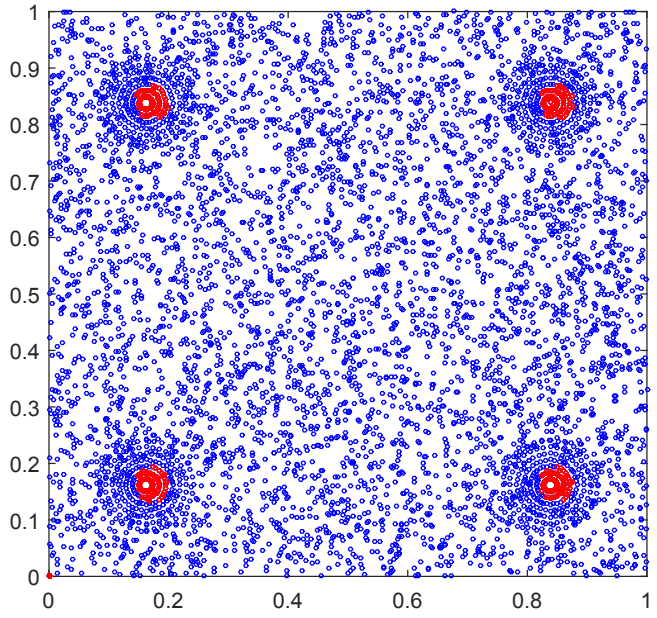

Fig. 5. Clustering results of the DBSCAN algorithm for $k = 8$ and $\varepsilon = 0.03$.

4. Results and discussions

The main objectives of fuzzy clustering algorithms are identifying dense and connected regions (clusters) in the data and in most cases finding centers of these clusters. Cluster centers of real data are not known a priori and are determined by the clustering algorithms. Therefore, two synthetic data sets with known cluster centers are used to study performance of the clustering algorithms discussed above. The algorithms are then applied to some real world data sets to study their performance. Since non-uniform mass distribution displaces cluster centers towards heavier clusters, performance of the RFCM algorithm is studied on two data sets with non-uniform mass distribution one with unequal clusters and the other with equal clusters.

There are some hyperparameters in each of the clustering algorithms discussed above that cannot be determined by conventional optimization techniques such as genetic algorithms, gradient descent algorithms, particle swarm optimization, etc. They are usually set by trial and error or via some specific measures and criteria proposed in the literature. In the present work, settings of the hyperparameters are carried out according to the typical values recommended by the literature. For all algorithms, degree of fuzziness $m = 2$, Euclidean distance with $A_{r\times r} = I_r$, termination criterion $\varepsilon = 0.00001$, and maximum number of iterations $\tau = 1000$ are used. For the PFCM and PCM algorithms $K = c_F = c_P = 1, \eta = 2$. For the NC algorithm $\lambda = 0.2$ and for the IFCM, IFCM-σ and KIFCM-σ algorithms $\rho = 0.85$ is used to calculate the hesitation degree π_{ij}. For the CFCM algorithm $\gamma = 0.1$ and for the SC algorithm the following values are used for influence range, quash factor, accept ratio, and reject ratio $\sigma_1 = 0.5,\ \sigma_2 = 1.25,\ \varepsilon_a = 0.5,\ \varepsilon_r = 0.15$. Hyperparameters of the DBSCAN algorithm ρ_{min} and ε are calculated by the k-dist graph as discussed above. Fully connected graph and similarity matrix with Gaussian weights are used in the spectral clustering algorithms. Degree of outlyingness o_{min} and number of denoising iterations T are set manually for the ORC algorithm to get the best clustering results. For the RFCM algorithm $\alpha = 4,\ p = 10,\ \phi_j = 1\ \forall j \in [1, n]$ are used.

4.1. Data with known cluster centers

There are four identical clusters in the noisy data shown in Fig. 3 each with 252 data points; and 5000 noise points are randomly distributed in the data. Clustering results of the abovementioned algorithms are shown in Fig. 3. The FCM algorithm does not identify actual cluster centers. The PFCM algorithm that is proposed by introducing the possibilistic terms into the FCM algorithm to improve its performance on noisy data shows no better performance. The PCM algorithm, which is specifically designed for noisy data, generates coincident clusters that is one of the main drawbacks of this algorithm as discussed above. Cluster centers calculated by the FCM-σ, KFCM, KFCM-σ, KIFCM, GMM, and KIFCM-σ algorithms are highly displaced from their actual locations and no improvements are made by these algorithms. However, comparing the IFCM and IFCM-σ algorithms to the KIFCM and KIFCM-σ algorithms indicates that kernelizing these algorithms deteriorates their performance. The NC, CFCM, NSC and csiFCM algorithms find inaccurate cluster centers but the siibFCM algorithm determines a couple of cluster centers more precisely. The SC algorithm specifically proposed to calculate both number of clusters and cluster centers determines five clusters, which is not true and the cluster centers are displaced due to noise impacts.

Applying the DBSCAN algorithm to the data requires determining the hyperparameters k and ε, which are calculated using the k-dist graph as shown in Fig. 2. As shown in Fig. 4, no stable value of k is identified from the k-dist graph and consequently value of ε cannot be clearly estimated. After several trials and errors the best values of these hyperparameters are found as $k = 64$ and $\varepsilon = 0.07$ for which results of the DBSCAN algorithm are shown in Fig. 3 that demonstrates good performance of the algorithm. Since DBSCAN is a density based algorithm, it does not calculate cluster centers but instead assigns the data points to the clusters. The data points indicated by red color are assigned to the first four clusters. It should be noted that deciding on optimal values of k and ε is seriously challenging since performance of the algorithm is highly sensitive to these hyperparameters. If the k-dist approach fails; it is not possible to find proper values of these hyperparameters in high dimensional data because such data cannot be visualized. For example, clustering results of the DBSCAN algorithm for $k = 8$ and $\varepsilon = 0.03$ are illustrated in Fig. 5 where the incomplete clusters clearly shows high sensitivity of the algorithm to these hyperparameters.

Results of clustering the data by the ORC algorithm are shown in Fig. 3 for $T = 4$ and $o_{max} = 0.5$, which are more accurate than those of many other algorithms studied in this work. As shown in Fig. 6, performance of this algorithm highly depends on T and o_{max} but there is no standard method to determine these hyperparameters for a given data set. It is observed that for $T = 1$ and $o_{max} = 0.5$ a large amount of noise is recognized as the actual data whereas for $T = 3$ and $o_{max} = 0.2$ the actual data are totally removed as noise. This is a serious drawback with this algorithm since it is not possible to visualize high dimensional data.

Finally, the RFCM algorithm determines the cluster centers correctly thanks to the exponential functions with adaptive bandwidth ω_i devised in the objective function of the algorithm to damp impacts of noise and outliers on the cluster centers.

As the RFCM algorithm converges, bandwidth of the exponential function ω_i gets more stable and during the last iterations remains unchanged as shown in Fig. 7 for each of the four clusters of the above data. Although four clusters of these data are identical but because of different noise intensities in their vicinities, different values are calculated for the bandwidth by the RFCM algorithm.

In order to study performance of the algorithms on noisy data with unequal clusters, the data shown in Fig. 8 are used where the clusters include 66, 72, 108, and 2160 data points. A total of 700 noise points are randomly distributed over these clusters. The FCM algorithm cannot find the actual cluster centers due to the noise and interactions between the unequal clusters. The PFCM and PCM algorithms calculate coincident clusters because of the possibilistic terms. In the FCM-σ, NC, KFCM, KFCM-σ, KIFCM, and KIFCM-σ algorithms the larger cluster attracts all cluster centers and the algorithms are unable to detect the small clusters. The IFCM, IFCM-σ, CFCM, and NSC algorithms show better performance

since only three cluster centers are captured by the large cluster though one of the clusters detected by the NSC algorithm is far away from the actual clusters. The GMM and ORC ($T = 3$ and $o_{max} = 0.4$) algorithms find two of the three smaller clusters. The DBSCAN algorithm ($k = 35$ and $\varepsilon = 0.14$) is significantly more accurate than the other algorithms though considers some of the noise as actual data. The SC algorithm finds only one cluster center and does not recognize presence of the other three clusters in the data. However, the siibFCM and csiFCM algorithms that are specifically designed for data with unequal clusters, perform fairly well although the cluster centers computed by these algorithms are inaccurate due to the noise impacts. The RFCM algorithm shows the best performance because incorporating size of the clusters in the algorithm using (62), reduces its sensitivity to the size of the clusters and the exponential functions eliminate impacts of noise and outliers on the cluster centers.

The siibFCM algorithm does not converge for the both data sets studied here and is terminated manually after large number of iterations. The same problem appears when it is applied to most of the real data discussed later.

4.2. Finding dense regions of data

4.2.1. Quality of clustering

Since cluster centers of real data are unknown, quality of clustering is an appropriate measure for evaluating performance of the clustering algorithms on such data, which is defined as the ratio of separation of the cluster centers (between-groups dissimilarity) to the compactness of the clusters (within-group similarity) (Askari, Montazerin, & Fazel Zarandi, 2017; Kwon, 1998; Pal & Bezdek, 1995; Xie & Beni, 1991). The former indicates how separate the cluster centers are and higher separation means better clustering. The latter demonstrates how compact each cluster is and larger compactness indicates better clustering. If the cluster centers are well-separated, the mutual differences between the membership grades u_{ij} of the data points in those clusters are larger because u_{ij} depends on the position of the cluster centers according to (3). Therefore, separation of the clusters is calculated as follows (Askari, Montazerin, & Fazel Zarandi, 2017).

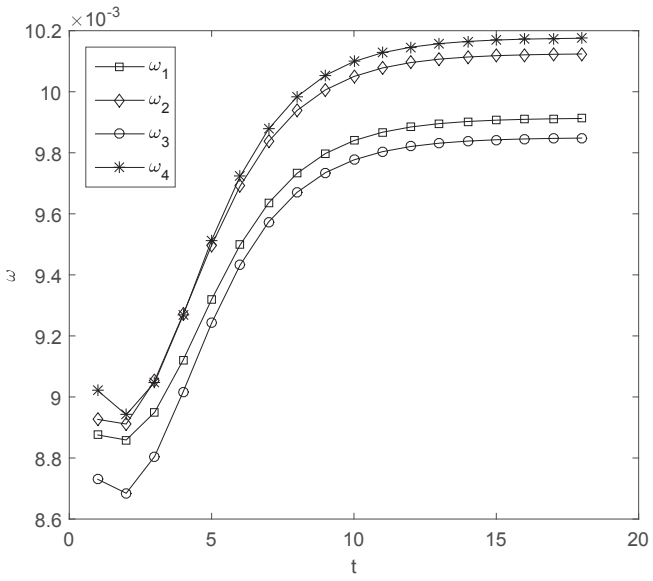

Fig. 7. Convergence of bandwidth of the exponential function used in the RFCM algorithm for four clusters.

$$S = \sum_{k=1}^{c}\sum_{i=1}^{c}\sum_{j=1}^{n}\left|u_{kj} - u_{ij}\right|^{m} \tag{74}$$

On the other hand, if a cluster is more compact, membership grades u_{ij} of its nearby data points in that cluster are larger and their complements (or non-membership degree) $\overline{u}_{ij} = 1 - u_{ij}$ is smaller. So, compactness of the clusters is defined as follows.

$$C = \sum_{j=1}^{n}\sum_{i=1}^{c}\overline{u}_{ij}^{m}, \overline{u}_{ij} = 1 - u_{ij} \tag{75}$$

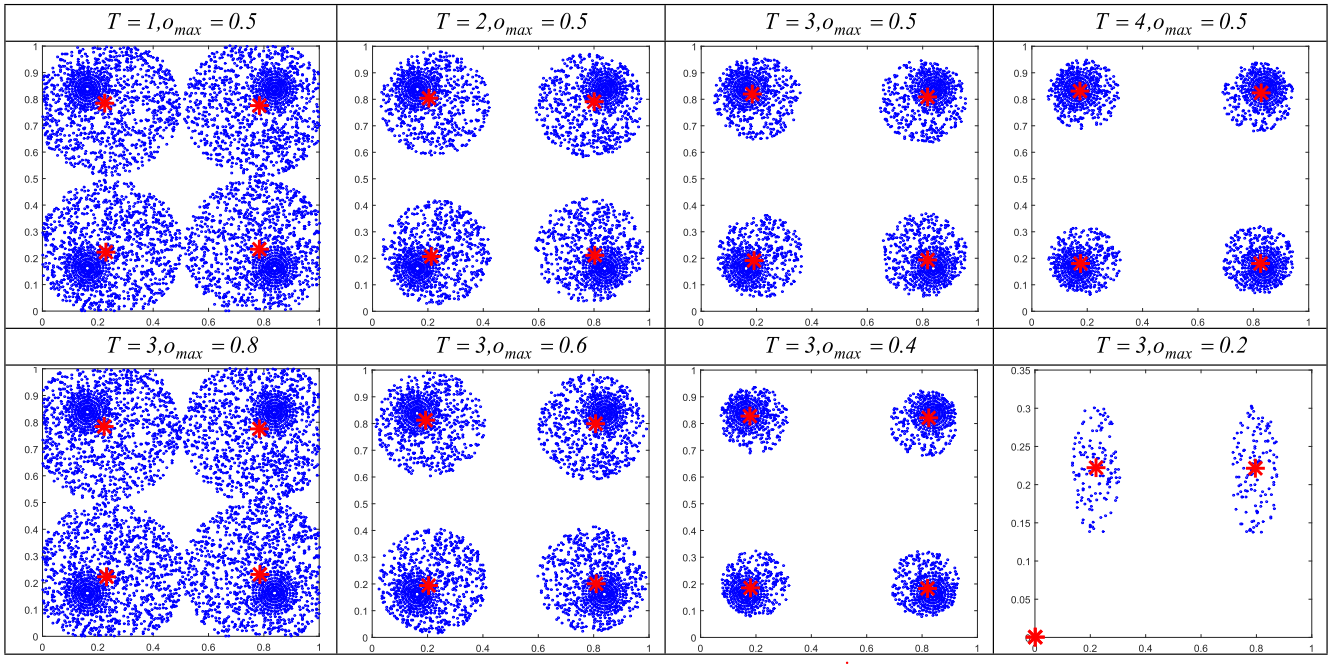

Fig. 6. Clustering results of the ORC algorithm for different values of its hyperparameters T and o_{max}.

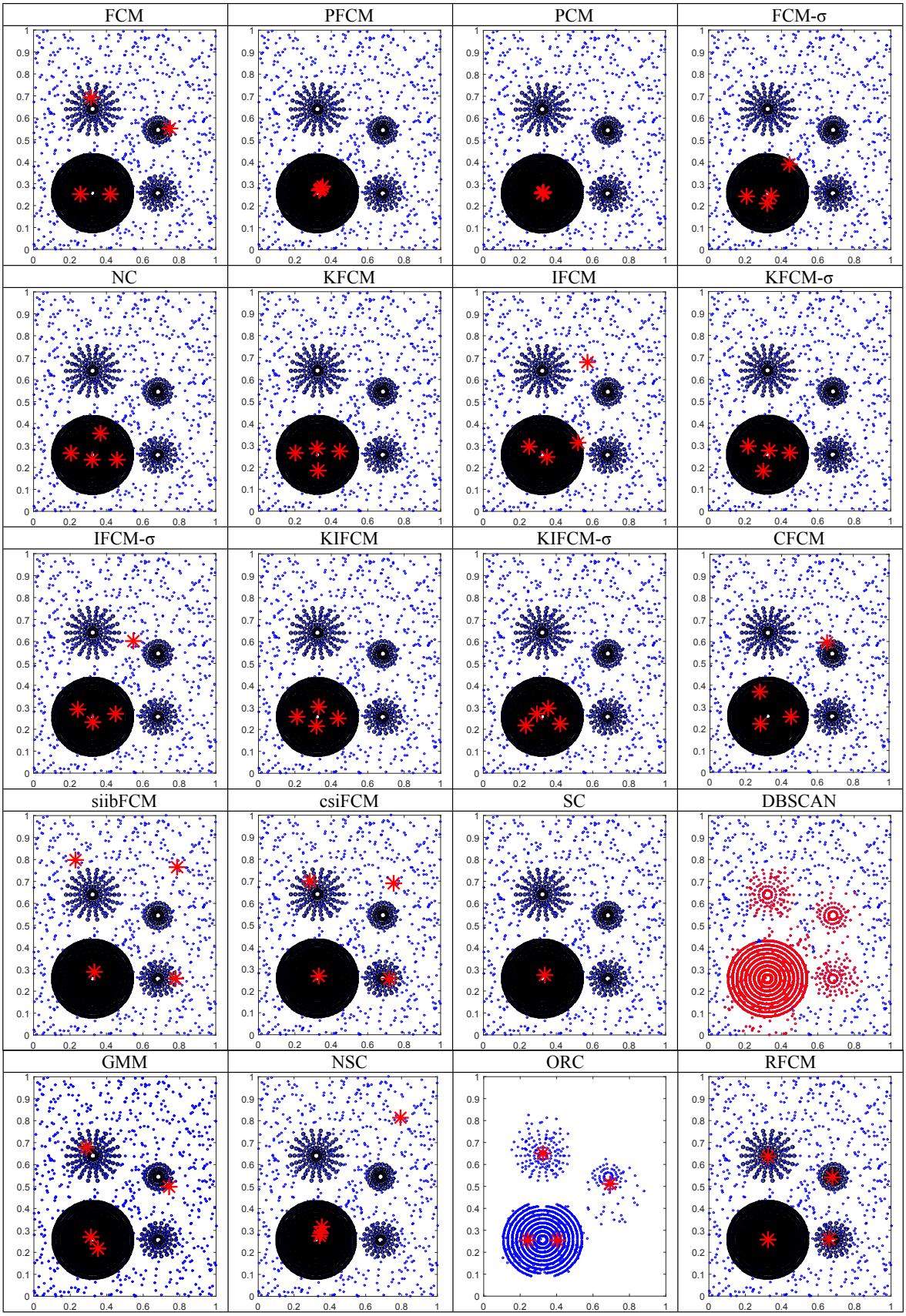

Fig. 8. Clustering noisy data with four unequal clusters by different clustering algorithms.

Since less value of C indicates higher compactness, the ratio of separation to compactness represents quality of clustering, which is defined as follows where u_{ij} and u_{kj} are calculated from (3) but with the cluster centers computed by the algorithm used for the clustering task. Therefore, higher value of $Q = S/C$ indicates better clustering.

$$Q = \frac{\sum_{k=1}^{c}\sum_{i=1}^{c}\sum_{j=1}^{n}\left|u_{kj} - u_{ij}\right|^{m}}{\sum_{j=1}^{n}\sum_{i=1}^{c}\overline{u}_{ij}^{m}}, \overline{u}_{ij} = 1 - u_{ij} \tag{76}$$

4.2.2. Xie-Beni index (XBI)

The Xie-Beni index *XBI* is defined as the ratio of compactness of the clusters to their separation. Therefore, less value of this index indicates more informative and better clusters. This index is defined as follows (Askari, Montazerin, Zarandi, & Hakimi, 2017).

$$XBI = \frac{\sum_{j=1}^{n}\sum_{i=1}^{c}u_{ij}^{m}\left\| x_j - v_i\right\|_A^2}{n \min\limits_{\substack{i,k\in[1,c]\\ i\neq k}}\left\{\left\| v_i - v_k\right\|_A^2\right\}} \tag{77}$$

4.2.3. Davies-Bouldin index (DBI)

The Davies-Bouldin Index *DBI* is widely used as a measure of quality of clustering results, which is defined as follows (Askari, Montazerin, Zarandi, & Hakimi, 2017).

$$\begin{aligned} &S_i = \frac{1}{n}\sum_{j=1}^{n}u_{ij}^{m}\left\| x_j - v_i\right\|_A^2, M_{ik} = \left\| v_k - v_i\right\|_A^2 \\ &R_{ik} = \frac{S_i + S_k}{M_{ik}} \ \forall i \neq k \in [1,c] \\ &D_i = \max_{\substack{k\in[1,c]\\ i\neq k}}\{R_{ik}\} \\ &DBI = \frac{1}{c}\sum_{i=1}^{c}D_i \end{aligned} \tag{78}$$

Less value of this index indicates better clustering results (Askari, Montazerin, Zarandi, & Hakimi, 2017).

The following data sets from the UCI machine learning repository are employed to examine performance of different algorithms on real data with different numbers of clusters, data points, and features. Glass Identification (GI), IRIS, Wine (W), Balance Scale (BS), Vehicle Silhouette (VS), Audit Data (AD), Statlog Image Segmentation (SIM), Libras (L), SCADI, Breast Cancer Coimbra (BCC), Anuran Calls (AC), and Parkinson (P)

Details of these data sets are given in Table 1 where c, n, and r indicate number of clusters (classes), number of samples, and number of features. Note that the data sets SCADI and Parkinson are not normalized since the algorithms yield coincident clusters if they are normalized.

The siibFCM algorithm suffers from convergence problems; the SC algorithm does not find the actual number of clusters in most cases, and the CFCM algorithm demands long running time because of considering q neighbors of each data point. The cases designated by (-) indicate that the algorithm does not converge or fails to provide any proper solution.

Quality of clustering these data sets with the algorithms discussed above is given in Table 1. When an algorithm generates well-separated clusters and places the cluster centers in dense regions of the data, the results are more qualified, which is indicated by larger Q. The PCM algorithm fails to cluster the data properly in most cases because of yielding coincident clusters. The same thing happens to the PFCM algorithm when the possibilistic terms are dominant as shown for some data sets in Table 1. It is observed that the RFCM algorithm provides the best clustering results in most cases when applied to these real data. The Xie-Beni and Davies-Bouldin indices for each of the algorithms are provided in Tables 2 and 3, respectively. It is observed that in most cases RFCM outperforms the other algorithms in terms of Xie-Beni and Davies-Bouldin indices

Running time of the algorithms for the data set shown in Fig. 3 is provided in Table 4. In terms of running time, the FCM and CFCM algorithms are the fastest and the slowest ones, respectively. The CFCM algorithm demands high running time because finding the q-nearest neighbors of each data point is a very time consuming process. Moreover, kernelized algorithms along with the RFCM algorithm takes long time because of nonlinear update equations for cluster centers. Density based algorithms such as SC and DBSCAN demand high running time because of wide search in the feature space. The NSC algorithm incurs high running time because calculating the similarity matrix $W_{n\times n}$ and the eigenvectors (spectra) is a time consuming task. The siibFCM algorithm needs large number of iterations to converge that needs high running time.

Table 1
Quality of clustering the real data sets by different algorithms.

	GI	IRIS	W	BS	VS	AD	SIM	L	SCADI	BCC	AC	P
C	6	3	3	3	4	2	7	15	7	2	4	2
N	214	150	178	625	846	776	2310	360	70	116	7195	195
R	9	4	13	4	18	17	18	90	205	9	22	22
FCM	0.6856	1.4076	0.6777	0.0054	0.5220	1.1234	0.6060	0.0002	–	0.3947	0.4469	1.3890
PFCM	0.0050	1.3578	0.1992	0.0000	0.3215	0.8125	0.2096	0.0000	–	0.0000	0.0266	1.1259
PCM	0.0037	0.9397	0.0000	0.0000	0.0002	0.0094	0.0000	0.0000	–	0.0000	0.0225	0.5374
FCM-σ	0.6448	1.4066	0.6789	0.0005	**0.5657**	1.0168	0.6130	0.0000	–	0.4016	0.3866	1.3953
NC	0.6219	1.3886	0.7081	0.0047	0.5151	1.0622	0.6322	0.0318	0.0140	0.4163	0.4561	1.3197
KFCM	0.6325	1.3872	0.3610	0.2554	0.5431	1.0553	0.3188	0.1951	–	0.4522	0.4606	1.3962
IFCM	0.5361	1.3789	0.3072	0.0002	0.4308	0.9353	0.4329	0.0000	–	0.0189	0.3935	1.3385
KFCM-σ	0.5666	1.3867	0.3594	0.2254	0.5363	1.0566	0.3192	0.1900	0.1690	0.4543	0.4213	1.3731
IFCM-σ	0.3546	1.3792	0.4650	0.0002	0.4372	0.8214	0.4274	0.0000	–	0.0081	0.2999	1.3751
KIFCM	0.4401	1.3623	0.3394	0.1850	0.4863	1.0217	0.1896	0.1027	–	0.0717	0.3203	1.3569
KIFCM-σ	0.3841	1.3627	0.3369	0.1850	0.4911	1.0246	0.2070	0.0981	0.0717	0.2253	0.4035	1.3479
CFCM	0.6797	1.3925	0.7226	–	0.5280	1.1000	0.6098	0.0446	0.1448	0.0752	0.0004	1.4113
siibFCM	–	–	–	–	–	1.0310	–	–	–	**0.5445**	–	–
csiFCM	0.8831	1.4136	0.6805	0.1850	0.5368	1.0181	0.6187	0.0000	0.0787	0.3703	0.4851	–
SC	–	1.3712	–	–	–	1.2581	–	–	–	–	–	–
DBSCAN	–	–	–	–	–	–	–	–	–	–	–	–
GMM	–	1.3804	0.7687	0.0012	0.5501	–	–	–	–	0.5262	0.0111	1.3874
NSC	–	–	–	–	–	–	–	–	–	–	–	–
ORC	–	–	–	–	–	–	–	–	–	–	–	–
RFCM	**1.0025**	**1.4147**	**0.8121**	**0.3530**	0.5452	**1.2611**	**0.6895**	**0.2015**	**0.3208**	0.5356	**0.5701**	**1.7696**

Table 2
The Xie-Beni index for different algorithms applied to real data sets.

	GI	IRIS	W	BS	VS	AD	SIM	L	SCADI	BCC	AC	P
C	6	3	3	3	4	2	7	15	7	2	4	2
N	214	150	178	625	846	776	2310	360	70	116	7195	195
R	9	4	13	4	18	17	18	90	205	9	22	22
FCM	1.3849	0.1757	0.4398	330.87	1.2208	0.2988	0.5071	20,546	–	76.04	5079	0.2628
PFCM	61,814	0.2821	2.8276	8.3e5	858.15	8.5e5	2.9e6	–	–	5.4e4	58,318	1.1758
PCM	49,201	7730	1682	1.2e7	20,480	82.46	1.4e6	–	–	3.7e5	8.0e6	13.06
FCM-σ	–	0.1692	0.4072	528.41	**0.5431**	0.7866	0.5910	7.1e5	–	1.5389	7792	0.6733
NC	7.6729	1.0278	0.3868	828.16	11.47	**0.1860**	**0.4442**	1.4e5	37,226	1.8140	33.65	0.6622
KFCM	8.1785	1.1592	1128	2.5012	5.1345	0.8811	0.9916	1451	–	1.4678	283.19	0.3843
IFCM	7.6464	0.2282	229.61	9467	7.6663	0.4908	6.6e6	73,440	–	129.74	14,883	0.3156
KFCM-σ	5.9258	1.1630	175.96	0.8507	5.1174	0.8762	1.0617	43.97	1.6e5	32.06	19.21	0.4427
IFCM-σ	1373	0.2256	0.8041	2876	42.17	4282	23.86	81,346	–	1617	36,366	0.3180
KIFCM	4001	1.2798	7063	24.68	12.14	0.9444	88.41	1.9e6	–	3.8494	7814	0.4581
KIFCM-σ	29,586	1.2792	6700	10.28	7097	0.9363	395.41	14,275	4.3e6	3.8432	3.6e5	0.5116
CFCM	5.4521	0.2046	0.3483	–	1.3354	0.3121	0.4796	2822	1023	1.4774	5399	0.3081
siibFCM	–	–	–	–	–	0.4397		–	–	0.9504	–	–
csiFCM	3.3887	0.1723	0.4091	411.50	1.5619	0.4526	0.5033	6.9e6	127.18	**0.6089**	46,315	–
SC	–	–	–	–	–	–	–	–	–	–	–	–
DBSCAN	–	–	–	–	–	–	–	–	–	–	–	–
GMM	–	0.2178	0.2975	3221	0.8967	–	–	–	–	0.8008	533.37	0.2357
NSC	–	–	–	–	–	–	–	–	–	–	–	–
ORC	–	–	–	–	–	–	–	–	–	–	–	–
RFCM	**0.8402**	**0.1307**	**0.2838**	**0.5318**	2.8343	**0.1860**	0.4883	**2.5525**	**0.5869**	1.3062	**6.5028**	**0.0409**

Table 3
The Davies-Bouldin index for different algorithms applied to real data sets.

	GI	IRIS	W	BS	VS	AD	SIM	L	SCADI	BCC	AC	P
C	6	3	3	3	4	2	7	15	7	2	4	2
N	214	150	178	625	846	776	2310	360	70	116	7195	195
R	9	4	13	4	18	17	18	90	205	9	22	22
FCM	0.4586	0.0946	0.2968	206.34	0.4616	0.2988	0.1102	1339	–	76.04	1617	0.2628
PFCM	20,601	0.1972	1.9127	5.5e4	313.79	857,737	695,566	–	–	54,960	29,193	1.1758
PCM	12,706	3782	1126	6.8e5	10,160	82.46	424,661	–	–	3.7e4	3.02e6	13.06
FCM-σ	–	0.1197	0.2721	283.60	**0.2759**	0.7866	0.1586	94,672	–	1.5389	3414	0.6733
NC	1.9828	0.8022	0.2607	552.11	5.5267	**0.1860**	0.1273	14,631	9086	1.8140	14.46	0.6622
KFCM	2.3444	0.9099	659.95	1.5285	2.4901	0.8811	0.2543	168.92	–	1.4678	60.28	0.3843
IFCM	2.4204	0.1584	141.86	4345	2.7512	0.4908	1.8e5	5612	–	129.74	7441	0.3156
KFCM-σ	1.9755	0.9127	121.39	0.5698	1.9238	0.8762	0.2776	2.41	35,428	32.06	6.3636	0.4427
IFCM-σ	417.70	0.1182	0.5333	1917	19.97	4282	6.8138	10,300	–	1617	18,179	0.3180
KIFCM	1296	0.9962	2752	14.64	5.7517	0.9444	15.45	2.6e4	–	3.8494	3330	0.4581
KIFCM-σ	4827	0.9957	2611	6.8597	3385	1.0180	59.94	724.08	957,047	3.8432	1.5e4	0.5116
CFCM	1.2006	0.1434	0.2366	–	0.3897	0.3121	0.1214	353.92	214.73	1.4774	2699	0.3081
siibFCM	–	–	–	–	–	0.4397	–	–	–	0.9504	–	–
csiFCM	0.8116	0.0925	0.2796	263.16	0.6053	0.4526	0.1276	871,927	37.53	**0.6089**	10,881	–
SC	–	–	–	–	–	–	–	–	–	–	–	–
DBSCAN	–	–	–	–	–	–	–	–	–	–	–	–
GMM	–	0.1542	0.2031	1657	0.4112	–	–	–	–	0.8008	265.90	0.2357
NSC	–	–	–	–	–	–	–	–	–	–	–	–
ORC	–	–	–	–	–	–	–	–	–	–	–	–
RFCM	**0.1832**	**0.0775**	**0.1950**	**0.3313**	1.1814	**0.1860**	**0.0959**	**0.2876**	**0.1631**	1.3062	**2.4015**	**0.0409**

Table 4
Running time (seconds) of different algorithms applied to the noisy data set with four clusters.

Algorithm	Running time	Algorithm	Running time	Algorithm	Running time	Algorithm	Running time
FCM	15.78	KFCM	110.78	KIFCM-σ	165.04	DBSCAN	598.64
PFCM	28.89	IFCM	17.84	CFCM	13,087	GMM	51.09
PCM	43.63	KFCM-σ	184.19	siibFCM	433.82	NSC	724.72
FCM-σ	82.43	IFCM-σ	31.95	csiFCM	24.41	ORC	55.00
NC	25.46	KIFCM	193.20	SC	118.97	RFCM	59.36

4.3. Applications to physics and engineering

The data points in the previous examples are massless whereas in physics and engineering they may represent particles with nonzero mass. Centroid of distributed particles is interested in many applications. Consider the single cluster of particles shown in Fig. 9. Center of this cluster v_i is calculated by equating moment of total mass of the cluster concentrated at v_i with respect to an arbitrary point O to the sum of moments of all particles of the cluster with position vector x_j and mass m_j with respect to O.

$$\left(\sum_{j=1}^{n} m_j\right) v_i = \sum_{j=1}^{n} m_j x_j \Rightarrow$$

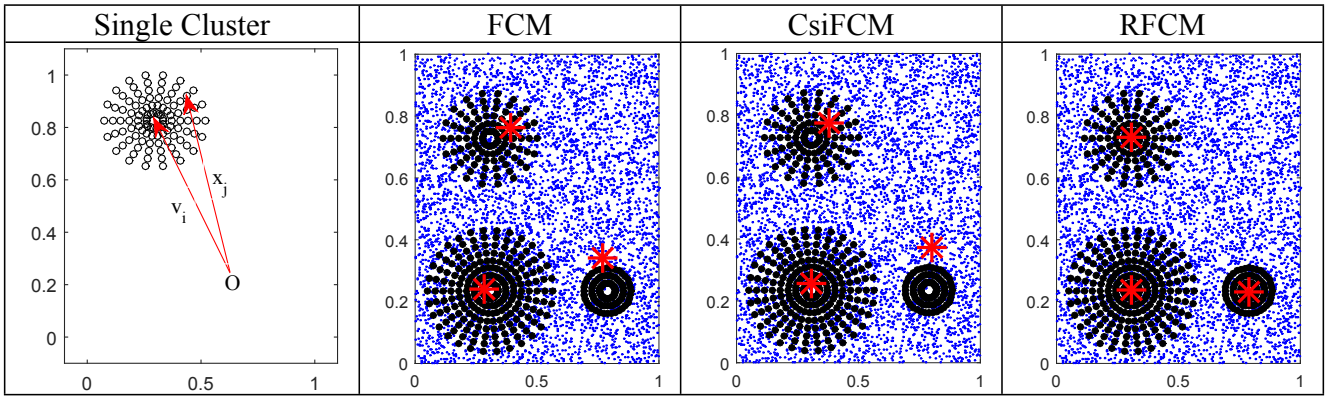

Fig. 9. Clustering noisy data with three unequal clusters and non-uniform mass distribution.

$$v_i = \frac{\sum_{j=1}^{n} m_j x_j}{\sum_{j=1}^{n} m_j} \tag{79}$$

This equation works only for one cluster. In presence of multiple clusters as shown in Fig. 9, a human being visually recognizes different clusters and calculates center of each cluster using (79). However, an intelligent machine has not such vision and should be equipped with proper algorithms to do the task correctly for which clustering algorithms can be used. The following modified form of objective function of the FCM algorithm works for these problems where mass of the particles m_j is included in the function.

$$J = \sum_{j=1}^{n}\sum_{i=1}^{c} m_j u_{ij}^{m} \| x_j - v_i \|_A^2, \sum_{k=1}^{c} u_{kj} = 1 \tag{80}$$

Using the Lagrange Multipliers method and zeroing derivatives of this function with respect to v_i and u_{ij} results the following equation for the cluster centers, which depends on the mass distribution as well as the location of the particles. However, Eq. (3) is derived again for the membership grades u_{ij}.

$$v_i = \frac{\sum_{j=1}^{n} m_j u_{ij}^{m} x_j}{\sum_{j=1}^{n} m_j u_{ij}^{m}} \tag{81}$$

Even if the data are noise-free and size of the clusters are not much different, uneven mass distribution may cause displacement of the cluster centers, which is clearly demonstrated by (81). Since the RFCM algorithm is less sensitive to the size of clusters and noise and outliers, it is supposed to work properly for such problems. Objective functions of the size-insensitive and noise-resistant parts of this algorithm are modified as follows to take mass of the particles into account.

$$J = \sum_{j=1}^{n}\sum_{i=1}^{c} m_j u_{ij}^{m} \| x_j - v_i \|_A^2, \sum_{k=1}^{c} u_{kj} = \rho_j(X, U) \tag{82}$$

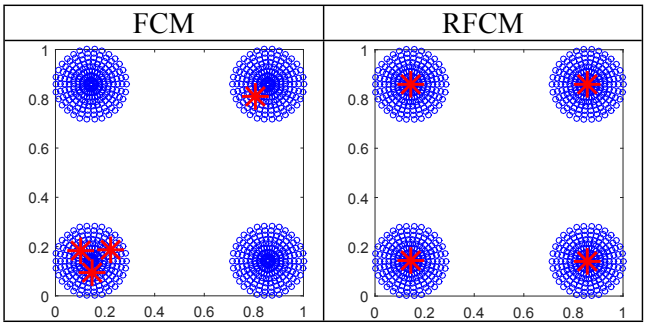

Fig. 10. Clustering data with four identical clusters but non-uniform mass distribution.

$$J = \sum_{j=1}^{n}\sum_{i=1}^{c} m_j u_{ij}^{m} f\left(\| x_j - v_i \|_A^2\right), \sum_{k=1}^{c} u_{kj} = \phi_j(X, U) \tag{83}$$

Similarly, by the procedure used for the RFCM algorithm it is shown that the following cluster centers are calculated by the size-insensitive and noise-resistant parts of the algorithm whereas the membership grades remain unaltered and are still calculated from (64) and (68), respectively.

Cluster centers calculated by the size-insensitive part of the RFCM algorithm:

$$v_i = \frac{\sum_{j=1}^{n} m_j u_{ij}^{m} x_j}{\sum_{j=1}^{n} m_j u_{ij}^{m}} \tag{84}$$

Cluster centers calculated by the noise-resistant part of the RFCM algorithm:

$$v_i = \frac{\sum_{j=1}^{n} m_j u_{ij}^{m} f'\left(\| x_j - v_i \|_A^2\right) x_j}{\sum_{j=1}^{n} m_j u_{ij}^{m} f'\left(\| x_j - v_i \|_A^2\right)} \tag{85}$$

Relative size of the ith cluster S_i is defined as follows by introducing mass of the particles into Eq. (62).

$$S_i = \frac{1}{|X|}\sum_{x_j \in A_i} m_j \left(1 + \frac{u_{ij}}{|X|^p}\right) \tag{86}$$

where $|X| = \sum_{j=1}^{n} m_j$ is total mass of the data and ρ_j is computed from (40) and $\partial\rho_j/\partial u_{ij}$ is calculated as follows.

$$\frac{\partial \rho_j}{\partial u_{ij}} = -\frac{\partial S_i}{\partial u_{ij}} = \begin{cases} -\dfrac{m_j}{|X|^{p+1}} & \text{if } i = \underset{r\in[1,c]}{arg\,max}\left(u_{rj}\right) \\ 0 & \text{Otherwise} \end{cases} \tag{87}$$

Note that relative size of the ith cluster is calculated as $S_i = |A_i|/|X|$ where $A_i = \sum_{x_j \in A_i} m_j$ and $|X| = \sum_{j=1}^{n} m_j$ for the siibFCM and csiFCM algorithms. The bandwidth ω_i of the exponential functions used in the RFCM algorithm is calculated as follows where s_{ij} is computed from (72).

$$\omega_i^2 = \frac{\sum_{j=1}^{n} m_j s_{ij}^{m} \| x_j - v_i \|_A^2}{\alpha \sum_{j=1}^{n} m_j s_{ij}^{m}} \tag{88}$$

The FCM, csiFCM, and RFCM algorithms are applied to the data with three unequal clusters and non-uniform mass distribution as shown in Fig. 9 where the large, medium, and small clusters contain 240, 108, 108 particles but the small cluster is more compact and dense than the medium one. Particles of each cluster have the same mass. Masse of particles of these clusters are 12, 3, and 2, respectively. Moreover, 5000 noise particles with unit mass are randomly distributed within the data. Results of clustering these data using the FCM, csiFCM, and RFCM

algorithms are shown in Fig. 9. The RFCM algorithm clearly shows better performance when dealing with noisy data with unequal clusters and non-uniform mass distribution.

A noise-free data set with four clusters is shown in Fig. 10. There are 240 data points in each cluster. Mass of each particle of the bottom-left cluster is 400 and that of the other three clusters is 10. Applying the FCM algorithm to these data results displacement of three clusters centers towards the heavier cluster because of non-uniform mass distribution according to (79) although these clusters are geometrically identical. However, the RFCM algorithm eliminates interactions among the clusters thanks to its size-insensitivity and the exponential functions that results accurate cluster centers.

5. Conclusions

Noise and outliers, unequal clusters, and non-uniform mass distribution impair accuracy of clustering algorithms and result inaccurate clustering results. Revised version of the FCM algorithm (RFCM) is presented in this paper for clustering data with such characteristics. RFCM algorithm eliminates contributions of noise and outliers by introducing an adaptive exponential function into the objective function of the FCM algorithm and prevents large or heavier clusters from attracting centers of small clusters by a size-insensitive mechanism. Performance of the proposed algorithm is compared to that of several well-known algorithms including Possibilistic Fuzzy C-Means (PFCM), Possibilistic C-Means (PCM), Robust Fuzzy C-Means (FCM-σ), Noise Clustering (NC), Kernel Fuzzy C-Means (KFCM), Intuitionistic Fuzzy C-Means (IFCM), Robust Kernel Fuzzy C-Mean (KFCM-σ), Robust Intuitionistic Fuzzy C-Means (IFCM-σ), Kernel Intuitionistic Fuzzy C-Means (KIFCM), Robust Kernel Intuitionistic Fuzzy C-Means (KIFCM-σ), Credibilistic Fuzzy C-Means (CFCM), Size-insensitive integrity-based Fuzzy C-Means (siibFCM), Size-insensitive Fuzzy C-Means (csiFCM), Subtractive Clustering (SC), Density Based Spatial Clustering of Applications with Noise (DBSCAN), Gaussian Mixture Models (GMM), Spectral clustering algorithm, and Outlier Removal Clustering (ORC). The algorithms are first applied to noisy data sets with equal or unequal cluster sizes and non-uniform mass distribution but known cluster centers. The experiments indicate superiority of the RFCM algorithm over the other ones. The algorithms are also applied to real data sets but since cluster centers of such data are unknown, quality of clustering, Xie-Beni index, and Davies-Bouldin index are employed to assess performance of the algorithms. The study shows that the RFCM algorithm provides the best results in terms of these measures in most cases.

CRediT authorship contribution statement

Salar Askari: Conceptualization, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing - original draft, Writing - review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Anderson, D. T., Bezdek, J. C., Popescu, M., & Keller, J. M. (2010). Comparing fuzzy, probabilistic, and possibilistic partitions. *IEEE Transactions on Fuzzy Systems, 18*(5), 906–918.

Askari, S. (2017a). A novel and fast MIMO fuzzy inference system based on a class of fuzzy clustering algorithms with interpretability and complexity analysis. *Expert Systems with Applications, 84*, 301–322.

Askari, S. (2017b). Oil reservoirs classification using fuzzy clustering. *International Journal of Engineering, Transactions C: Aspects, 30*, 1391–1400.

Askari, S., & Montazerin, N. (2015). A high-order multi-variable Fuzzy Time Series forecasting algorithm based on fuzzy clustering. *Expert Systems with Applications, 42*(4), 2121–2135.

Askari, S., Montazerin, N., & Fazel Zarandi, M. H. (2017). Generalized Possibilistic Fuzzy C-Means with novel cluster validity indices for clustering noisy data. *Applied Soft Computing, 53*, 262–283.

Askari, S., Montazerin, N., & Fazel Zarandi, M. H. (2020). Modeling energy flow in natural gas networks using time series disaggregation and fuzzy systems tuned by particle swarm optimization. *Applied Soft Computing, 92*, 106332.

Askari, S., Montazerin, N., & Zarandi, M. H. F. (2015a). A clustering based forecasting algorithm for multivariable fuzzy time series using linear combinations of independent variables. *Applied Soft Computing, 35*, 151–160.

Askari, S., Montazerin, N., & Zarandi, M. H. F. (2015b). Forecasting semi-dynamic response of natural gas networks to nodal gas consumptions using genetic fuzzy systems. *Energy, 83*, 252–266.

Askari, S., Montazerin, N., Zarandi, M. H. F., & Hakimi, E. (2017). Generalized entropy based possibilistic fuzzy C-Means for clustering noisy data and its convergence proof. *Neurocomputing, 219*, 186–202.

Beliakov, G., Li, G., Vu, H. Q., & Wilkin, T. (2015). Characterizing compactness of geometrical clusters using fuzzy measures. *IEEE Transactions on Fuzzy Systems, 23*(4), 1030–1043.

Bezdek, J. C., Ehrlich, R., & Full, W. (1984). FCM: The fuzzy c-means clustering algorithm. *Computers & Geosciences, 10*(2-3), 191–203.

Boonchoo, T., Ao, X., Liu, Y., Zhao, W., Zhuang, F., & He, Q. (2019). Grid-based DBSCAN: Indexing and inference. *Pattern Recognition, 90*, 271–284.

Chaira, T. (2011). A novel intuitionistic fuzzy C means clustering algorithm and its application to medical images. *Applied Soft Computing, 11*(2), 1711–1717.

Chen, S.-M., & Chen, S.-W. (2015). Fuzzy forecasting based on two-factors second-order fuzzy-trend logical relationship groups and the probabilities of trends of fuzzy logical relationships. *IEEE Transactions on Cybernetics, 45*(3), 391–403.

Chen, L., Chen, C. L. P., & Lu, M. (2011). A multiple-kernel fuzzy C-means algorithm for image segmentation. *IEEE Transactions on Systems, Man, and Cybernetics, 41*(5), 1263–1274.

Chen, S.-M., Chu, H.-P., & Sheu, T.-W. (2012). TAIEX forecasting using fuzzy time series and automatically generated weights of multiple factors. *IEEE Transactions on Systems, Man, and Cybernetics, 42*(6), 1485–1495.

Chen, S. M., Manalu, G. M. T., Pan, J. S., & Liu, H. C. (2013). Fuzzy forecasting based on two-factors second-order fuzzy-trend logical relationship groups and particle swarm optimization techniques. *IEEE Transactions on Cybernetics, 43*, 1102–1117.

Chen, S.-M., & Wang, N.-Y. (2010). Fuzzy forecasting based on fuzzy-trend logical relationship groups. *IEEE Transactions on Systems, Man, and Cybernetics, 40*(5), 1343–1358.

Chintalapudi, K. K., & Kam, M. (1998). The credibilistic fuzzy C-means clustering algorithm. *IEEE International Conference on Systems, Man, and Cybernetics*, 2034–2039.

Chiu, S. L. (1994). Fuzzy model identification based on cluster estimation. *Journal of Intelligent and Fuzzy Systems, 2*, 267–278.

Dave, R. N. (1991). Characterization and detection of noise in clustering. *Pattern Recognition Letters, 12*(11), 657–664.

Dave, R. N., & Krishnapuram, R. (1997). Robust clustering methods: A unified view. *IEEE Transactions on Fuzzy Systems, 5*, 270–293.

Ester, M., Kriegel, H. P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In C. A. Menlo Park (Ed.), *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* (pp. 226–231).

Filippone, M., Masulli, F., & Rovetta, S. (2010). Applying the possibilistic c-means algorithm in kernel-induced spaces. *IEEE Transactions on Fuzzy Systems, 18*(3), 572–584.

Gebru, I. D., Alameda-Pineda, X., Forbes, F., & Horaud, R. (2016). EM algorithms for weighted-data clustering with application to audio-visual scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 38*(12), 2402–2415.

Gosain, A., & Dahiya, S. (2016). Performance analysis of various fuzzy clustering algorithms: A review. *Procedia Computer Science, 79*, 100–111.

Groll, L., & Jakel, J. (2005). A new convergence proof of fuzzy c-means. *IEEE Transactions on Fuzzy Systems, 13*(5), 717–720.

Hamidzadeh, J., & Ghadamyari, R. (2020). Clustering data stream with uncertainty using belief function theory and fading function. *Soft Computing, 24*(12), 8955–8974.

Hariz, S. B., Elouedi, Z., & Mellouli, K. (2006). Clustering approach using belief function theory. *International Conference on Artificial Intelligence: Methodology, Systems, and Applications*, 162–171.

Hathaway, R. J., & Bezdek, J. C. (2001). Fuzzy c-means clustering of incomplete data. *IEEE Transactions on Systems, Man, and Cybernetics, 31*, 735–744.

Hautamaki, V., Cherednichenko, S., Karkkainen, I., Kinnunen, T., & Franti, P. (2005). Improving K-means by outlier removal. *Scandinavian Conference on Image Analysis*, 978–987.

Havens, T. C., Bezdek, J. C., Leckie, C., Hall, L. O., & Palaniswami, M. (2012). Fuzzy c-means algorithms for very large data. *IEEE Transactions on Fuzzy Systems, 20*(6), 1130–1146.

He, Z., & Ho, C.-H. (2019). An improved clustering algorithm based on finite Gaussian mixture model. *Multimedia Tools and Applications, 78*(17), 24285–24299.

Hu, Z., Nie, F., Chang, W., Hao, S., Wang, R., & Li, X. (2020). Multi-view spectral clustering via sparse graph learning. *Neurocomputing, 384*, 1–10.

Janani, R., & Vijayarani, S. (2019). Text document clustering using Spectral Clustering algorithm with Particle Swarm Optimization. *Expert Systems with Applications, 134*, 192–200.

Kaur, P., Soni, A. K., & Gosain, A. (2012). Novel intuitionistic fuzzy c means clustering for linearly and nonlinearly separable data. *WSEAS Transactions on Computers, 11*, 65–76.
Koutroumbas, K. D., Xenaki, S. D., & Rontogiannis, A. A. (2018). On the convergence of the sparse possibilistic c-means algorithm. *IEEE Transactions on Fuzzy Systems, 26*(1), 324–337.
Krishnapuram, R., & Keller, J. (1993). A possibilistic approach to clustering. *IEEE Transactions on Fuzzy Systems, 1*, 98–110.
Krishnapuram, R., & Keller, J. (1996). The possibilistic c–means algorithm: Insights and recommendations. *IEEE Transactions on Fuzzy Systems, 4*, 385–393.
Kwon, S. H. (1998). Cluster validity index for fuzzy clustering. *Electronics Letters, 34*(22), 2176. https://doi.org/10.1049/el:19981523.
Lee, C.-H., Chang, F.-Y., & Lin, C.-M. (2014). An efficient interval type-2 fuzzy CMAC for chaos time-series prediction and synchronization. *IEEE Transactions on Cybernetics, 44*(3), 329–341.
Lei, D., Zhu, Q., Chen, J., Lin, H., & Yang, P. (2012). Automatic K-means clustering algorithm for outlier detection. *Information Engineering and Applications, 154*, 363–372.
Leski, J. M. (2016). Fuzzy c -ordered-means clustering. *Fuzzy Sets and Systems, 286*, 114–133.
Li, S. T., & Cheng, Y. C. (2010). A stochastic HMM-based forecasting model for fuzzy time series. *IEEE Transactions on Systems, Man, and Cybernetics, 40*, 1255–1266.
Li, Z., Hu, J., Stojmenovic, M., Liu, Z., & Liu, W. (2020). Revisiting spectral clustering for near-convex decomposition of 2D shape. *Pattern Recognition, 105*, Article 107371.
Li, Y., Zhang, J., He, R., Tian, L., & Wei, H. (2019). Hybrid DE-EM algorithm for gaussian mixture model-based wireless channel multipath clustering. *International Journal of Antennas and Propagation, 2019*, 1–10.
Lin, P.-L., Huang, P.-W., Kuo, C. H., & Lai, Y. H. (2014). A size-insensitive integrity-based fuzzy c-means method for data clustering. *Pattern Recognition, 47*(5), 2042–2056.
Linn, K. A., Gaonkar, B., Satterthwaite, T. D., Doshi, J., Davatzikos, C., & Shinohara, R. T. (2016). Control-group feature normalization for multivariate pattern analysis of structural MRI data using the support vector machine. *NeuroImage, 132*, 157–166.
Liu, Z., Xu, S., Zhang, Y., & Chen, C. L. P. (2014). A multiple-feature and multiple-kernel scene segmentation algorithm for humanoid robot. *IEEE Transactions on Cybernetics, 44*, 2232–2240.
Luchi, D., Loureiros Rodrigues, A., & Miguel Varejão, F. (2019). Sampling approaches for applying DBSCAN to large datasets. *Pattern Recognition Letters, 117*, 90–96.
Maji, P., & Pal, S. K. (2007). Rough set based generalized fuzzy c-means algorithm and quantitative indices. *IEEE Transactions on Systems, Man, and Cybernetics, 37*(6), 1529–1540.
Makrogiannis, S., Economou, G., Fotopoulos, S., & Bourbakis, N. G. (2005). Segmentation of color images using multiscale clustering and graph theoretic region synthesis. *IEEE Transactions on Systems, Man, and Cybernetics, 35*(2), 224–238.
Noordam, J. C., van den Broek, W. H. A. M., & Buydens, L. M. C. (2002). Multivariate image segmentation with cluster size insensitive Fuzzy C-means. *Chemometrics and Intelligent Laboratory Systems, 64*(1), 65–78.
Ozdemir, D., & Akarun, L. (2001). Fuzzy algorithms for combined quantization and dithering. *IEEE Transactions on Image Processing, 10*, 923–931.
Pal, N. R., & Bezdek, J. C. (1995). On cluster validity for the Fuzzy C-Means model. *IEEE Transactions on Fuzzy Systems, 3*, 370–379.
Pal, N. R., Pal, K., Keller, J. M., & Bezdek, J. C. (2005). A possibilistic fuzzy c-means clustering algorithm. *IEEE Transactions on Fuzzy Systems, 13*(4), 517–530.
Siminski, K. (2017). Fuzzy weighted C-ordered means clustering algorithm. *Fuzzy Sets and Systems, 318*, 1–33.
Tolias, Y. A., & Panas, S. M. (1998). Image segmentation by a fuzzy clustering algorithm using adaptive spatially constrained membership functions. *IEEE Transactions on Systems, Man, and Cybernetics, 28*, 359–369.
Tsai, D.-M., & Lin, C.-C. (2011). Fuzzy C-means based clustering for linearly and nonlinearly separable data. *Pattern Recognition, 44*(8), 1750–1760.
Tung, W. L., & Quek, C. (2004). Falcon: Neural fuzzy control and decision systems using FKP and PFKP clustering algorithms. *IEEE Transactions on Systems, Man, and Cybernetics, 34*(1), 686–695.
von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and Computing, 17*(4), 395–416.
Xie, X. L., & Beni, G. A. (1991). Validity measure for fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 3*, 841–846.
Xu, S. K., Wang, C., Zhuang, L. H., & Gao, X. H. (2019). DBSCAN clustering algorithm for the detection of nearby open clusters based on Gaia-DR2. *Chinese Astronomy and Astrophysics, 43*, 225–236.
Yager, R. R., & Filev, D. P. (1994). Approximate clustering via the mountain method. *IEEE Transactions on Systems, Man, and Cybernetics, 24*, 1279–1284.
Yu, H., Chen, L. Y., Yao, J. T., & Wang, X. N. (2019). A three-way clustering method based on an improved DBSCAN algorithm. *Physica A, 535*, 122289.
Zeng, H., & Cheung, Y. M. (2014). Learning a mixture model for clustering with the completed likelihood minimum message length criterion. *Pattern Recognition, 47*, 2011–2030.
Zhang, M., Hall, L. O., & Goldgof, D. B. (2002). A generic knowledge-guided image segmentation and labeling system using fuzzy clustering algorithms. *IEEE Transactions on Systems, Man, and Cybernetics, 32*(5), 571–582.
Zhang, J.-S., & Leung, Y.-W. (2004). Improved possibilistic C-means clustering algorithms. *IEEE Transactions on Fuzzy Systems, 12*(2), 209–217.
Zhang, Q., Yang, L. T., Chen, Z., & Xia, F. (2017). A high-order possibilistic c-means algorithm for clustering incomplete multimedia data. *IEEE Systems Journal, 11*, 2160–2169.
Zhu, L., Chung, F. L., & Wang, S. (2009). Generalized fuzzy c-means clustering algorithm with improved fuzzy partitions. *IEEE Transactions on Systems, Man, and Cybernetics, 39*, 578–591.