A Novel Adaptive Possibilistic Clustering Algorithm

Spyridoula D. Xenaki\textsuperscript{1, a}, Konstantinos D. Koutroumbas\textsuperscript{1} and Athanasios A. Rontogiannis\textsuperscript{1}

\textsuperscript{1}Institute for Astronomy, Astrophysics, Space Applications and Remote Sensing (IAASARS), National Observatory of Athens, Penteli, GR-15236 Greece

Abstract

In this paper a novel possibilistic c-means clustering algorithm, called Adaptive Possibilistic c-means, is presented. Its main feature is that \textit{all} its parameters, after their initialization, are properly adapted during its execution. This increases the flexibility of the algorithm in following the variations in the formation of the clusters that occur from iteration to iteration. The fully adaptive nature of the proposed algorithm enables the removal of the clusters that gradually become obsolete. This makes the algorithm capable to determine, in principle, the true number of the clusters underlying the data set (a long-standing issue in the clustering literature), provided that it starts with a reasonable overestimate of the actual number of clusters. However, the algorithm necessitates an appropriate, but fully determined in most cases, scaling of the data set before its execution. Extensive simulation results on both synthetic and real data highlight the effectiveness of the proposed algorithm.

Index Terms

Possibilistic clustering, parameter adaptation, cluster elimination, data scaling

I. INTRODUCTION

Clustering is a well established data analysis methodology that has been extensively used in various fields of applications during the last decades, such as life sciences, medical sciences

\textsuperscript{a}Corresponding Author: ixenaki@noa.gr
and engineering [1]. Its aim is the identification of groups (clusters) formed by “similar” entities (e.g. [2], [3], [4], [5]). Usually, each entity is represented by a set of measurements, which forms its feature vector. This is also called data vector and the set of all these vectors forms the data set under study. The space where all these vectors live is called feature space. According to the way a data vector is associated with various clusters, three main philosophies have been developed: (a) hard clustering, where each vector belongs exclusively to a single cluster, (b) fuzzy clustering, where each vector may be shared among two or more clusters and (c) possibilistic clustering, where the association (degree of compatibility) of a data vector with a given cluster is independent of its association with any other cluster.

Most of the work on clustering has been focused on compact and hyperellipsoidally shaped clusters and the most well-known algorithms that deal with this case and follow one of the previous philosophies are the k-means (hard clustering), e.g. [6], the fuzzy c-means (FCM), e.g. [7], [8] and the possibilistic c-means (PCM), e.g. [2], [9], [10], [11], [12], [13], respectively. In all these algorithms the clusters are represented by vectors that lie in the feature space, called cluster representatives. The aim of all these algorithms is to move the representatives to the “centers” of the regions that are “dense in data points” (clusters). In this way, each such vector represents a cluster and their movement towards the center of the clusters is carried out via the minimization of appropriately defined cost functions.

Notwithstanding their popularity, both k-means and FCM have two shortcomings. First, they are vulnerable to noisy data and outliers [2], [11]. Second, they require prior knowledge of the number of clusters, \( m \), underlying in the data set (which, of course, is rarely known in practice). An additional characteristic that both of these algorithms share is that they impose a clustering structure on the data set, in the sense that they will return \( m \) clusters irrespectively of the fact that more or less than \( m \) clusters may actually underlie in the data set. Specifically, if \( m \) is less than the actual number of clusters, at least some representatives will fail to be moved to dense regions, while in the opposite case, some naturally formed clusters will split into more than one pieces.

As far as the PCM is concerned, the cluster representatives are updated, based on the

\(^1\)A method for facing this problem with FCM is discussed in [14].
degree of compatibility of a data vector with a given cluster. Contrastingly to the FCM, in PCM the degrees of compatibility of a data vector with the various clusters are independent to each other and no sum-to-one constraint is imposed on them. A consequence of this fact is that even if the number of clusters is overestimated, in principle, all representatives will be driven to dense regions, making thus feasible the uncovering of the true clusters. However, in this case, the scenario where two or more cluster representatives are led to the same dense in data region, may arise [15], [16]. In addition, although PCM deals well with noisy data points and outliers, compared to k-means and FCM, it involves additional parameters, usually denoted by $\eta$, and requires good estimates for them, which, once they have been estimated, they are kept fixed during its execution. Poor initial estimation of these parameters often leads to poor clustering performance, especially in more demanding data sets. Variants of PCM that try to address these issues have been proposed in [11], [12], [13] and [15].

In the present work, we focus on PCM. More specifically, we extend the classical PCM algorithm, proposed in [10], by modifying the way the parameters $\eta$ are treated, giving rise to a new algorithm called Adaptive Possibilistic c-means (APCM)\(^2\). In APCM the parameters $\eta$, after their initialization, are properly adapted as the algorithm evolves. In particular, for each specific cluster, we propose to estimate its parameter $\eta$ as a measure of the mean absolute deviation of the data vectors that are most compatible with this cluster. Note that this is in contrast to the usual policy of estimating $\eta$'s, as a measure of variance. However, due to this modification, it is shown that an appropriate (but fully specified in most practical cases) scaling of the data should be conducted, before the execution of the algorithm.

The adaptation of $\eta$'s renders the algorithm more flexible in uncovering the underlying clustering structure, compared to other related possibilistic algorithms. In addition, as a direct consequence of this adaptation, the algorithm has the ability to estimate the (unknown in most cases in practice) true number of natural clusters. More specifically, if the algorithm starts with a crude overestimation of the number of natural clusters, it gradually reduces this number, as it progresses, and, finally, provides the actual number of clusters, i.e., the algorithm is able to estimate by itself the number of clusters, which is a long-standing issue in the clustering

\(^2\)A preliminary version of APCM has been presented in [17].
framework. Extensive simulation results on both synthetic and real data, corroborate our theoretical analysis and show that APCM offers in general superior clustering performance compared to relative state-of-the-art clustering algorithms.

The rest of the paper is organized as follows. In Section 2, a brief description of PCM algorithms is given, as well as previous attempts for dealing with their shortcomings. In Section 3, the proposed Adaptive PCM (APCM) clustering algorithm is fully presented and the need for data scaling, as well as its consequences, are particularly emphasized. In Section 4, the performance of APCM is tested against several related state-of-the-art algorithms. Finally, concluding remarks are provided in Section 5.

II. A REVIEW OF PCM, ISSUES AND POTENTIAL SOLUTIONS

In this section, the PCM clustering algorithm is reviewed and its main features are discussed. Also, possible solutions from the literature are considered that try to deal with its weak points.

A. PCM review

Let

\[ X = \{ x_i \in \mathbb{R}^\ell, i = 1, \ldots, N \} \]

be a set of \( N \), \( \ell \)-dimensional data vectors and

\[ \Theta = \{ \theta_j \in \mathbb{R}^\ell, j = 1, \ldots, m \} \]

be a set of \( m \) vectors that will be used for the representation of the clusters formed by the points in \( X \). Let \( U = [u_{ij}], i = 1, \ldots, N, j = 1, \ldots, m \) be an \( N \times m \) matrix whose \((i, j)\) entry stands for the so-called degree of compatibility of \( x_i \) with the \( j \)th cluster, denoted by \( C_j \), and represented by the vector \( \theta_j \). In what follows we consider only Euclidean norms, denoted by \( \| \cdot \| \).

Unlike fuzzy clustering algorithms, the sum-to-one constraint is not imposed on the rows of \( U \) in possibilistic clustering algorithms, i.e. the term \( \sum_{j=1}^{m} u_{ij} \) is not necessarily equal to 1.
for each \( x_i \). According to [9], [10], the \( u_{ij} \)'s should satisfy the conditions,

\[
(C1) \ u_{ij} \in [0, 1], \quad (C2) \ \max_{j=1,...,m} u_{ij} > 0 \quad \text{and} \quad (C3) \ 0 < \sum_{i=1}^{N} u_{ij} < N
\]  

(1)

In words, (C2) means that no vector is allowed to be totally incompatible with all clusters, whereas (C3) means that for a given cluster, there is at least one data point that is not totally incompatible with it. Loosely speaking, each data point should “belong” to at least one cluster (C2), whereas no cluster is allowed to be “empty” (C3). The aim of a possibilistic algorithm is to move \( \theta_j \)'s to regions that are dense in data points of \( X \). This is carried out via the minimization of, among others, the following objective function [10]:

\[
J_{PCM}(\Theta, U) = \sum_{j=1}^{m} J_j \equiv \sum_{j=1}^{m} \left[ \sum_{i=1}^{N} u_{ij} \| x_i - \theta_j(t) \|^2 + \eta_j \sum_{i=1}^{N} (u_{ij} \ln u_{ij} - u_{ij}) \right]
\]  

(2)

with respect to \( \theta_j \)'s and \( u_{ij} \)'s, while \( \eta_j \)'s are fixed user-defined parameters. Note that the second term in the bracketed expression in the right hand side of eq. (2) prevents the algorithm from ending up with the trivial zero solution for \( u_{ij} \)'s.

Proceeding with the minimization of \( J_{PCM}(\Theta, U) \) with respect to \( u_{ij} \) and \( \theta_j \), we end up with the following PCM updating equations,

\[
u_{ij}(t) = \exp \left( -\frac{\| x_i - \theta_j(t) \|^2}{\eta_j} \right)
\]  

(3)

\[
\theta_j(t + 1) = \frac{\sum_{i=1}^{N} u_{ij}(t) x_i}{\sum_{i=1}^{N} u_{ij}(t)}
\]  

(4)

for \( t = 0, 1, 2, \ldots \), with the iterations being started after the initialization of \( \theta_j \)'s to \( \theta_j(0) \)'s, \( j = 1, \ldots, m \). Iterations are performed until a specific termination criterion is met (e.g., no significant change occurs on \( \theta_j \)'s between two successive iterations). Note from the updating equation (3) that \( u_{ij} \) decreases exponentially fast as \( x_i \) moves away from \( \theta_j \). Also, from eq. (4), it follows that all data vectors contribute to the estimation of the next location of each cluster.

---

1We use this cost function, instead of the one given in the seminal paper [9], since the proposed scheme, to be presented in the next section, is based on it. However, for reasons of thoroughness, we give also the cost function of [9], which is

\[
J_{PCM}^\prime(\Theta, U) = \sum_{j=1}^{m} J_j' \equiv \sum_{j=1}^{m} \left[ \sum_{i=1}^{N} u_{ij}' \| x_i - \theta_j \|^2 + \eta_j \sum_{i=1}^{N} (1 - u_{ij})^q \right]
\]

where \( q \) is a parameter that “resembles” to the fuzzifier in FCM (such a parameter does not appear in \( J_{PCM} \), see eq. (2)).
one of the representatives. However, the farthest ones from the current location of a specific \( \theta_j \) contribute less to the determination of its new location, since the corresponding \( u_{ij} \)’s are smaller for these vectors, as eq. \((3)\) indicates.

Let us comment now on the parameters \( \eta_j \), \( j = 1, \ldots, m \). These are a priori estimated and kept fixed during the execution of the algorithm. A common strategy for their estimation is to run the FCM algorithm first and set

\[
\eta_j = K \frac{\sum_{i=1}^{N} u_{ij}^{FCM} \|x_i - \theta_j\|^2}{\sum_{i=1}^{N} u_{ij}^{FCM}}, \quad j = 1, \ldots, m
\]

(5)

where \( \theta_j \)’s and \( u_{ij}^{FCM} \)’s are the final FCM estimates for cluster representatives and \( u_{ij} \) coefficients, respectively. Parameter \( K \) is user-defined and is usually set equal to 15. Let us comment on the rationale of the above choice of \( \eta_j \). According to eq. \((3)\), the value of \( \eta_j \) determines the squared Euclidean distance from the cluster representative \( \theta_j \), at which the degree of compatibility of a data point becomes equal to \( \exp(-1) \approx 0.37 \). This will be used as an “anchor” value, in order to have a quantification of “large” and “small” for the values of \( u_{ij} \). If \( \eta_j \) were selected to be “very large”, the \( u_{ij} \)’s would be “large”, even for points that lie not “too close” to the cluster representative \( \theta_j \). Thus, these distant points would significantly affect the adjustment of \( \theta_j \) and the latter would most likely fail to reach the center of a physical cluster. On the other hand, if \( \eta_j \) were too small, then the \( u_{ij} \)’s would be “too small” even for data points that are “too close” to \( \theta_j \). This, would impose a serious obstacle in the movement of \( \theta_j \) towards a dense in data region, since only very few points would practically contribute to the determination of the new position of \( \theta_j \). Thus, assuming that \( \theta_j \) is initialized close to a dense in data region (true cluster), \( \eta_j \) should be chosen so that to make \( u_{ij} \) “large” for data points in this region, and “small” for all other points. Having this in mind, a reasonable choice for \( \eta_j \) is to define it as a measure of variance of the true cluster. Thus, for points with \( \|x_i - \theta_j\|^2 < \eta_j \) (i.e., that lie close to \( \theta_j \)), the ratio \( \|x_i - \theta_j\|^2/\eta_j \) will become less than 1 and \( u_{ij} \) will become “large” (greater than \( \exp(-1) \approx 0.37 \)). On the other

\footnote{The version of eq. \((5)\) proposed in \cite{9} for the cost function \( J_{PCM}^r \) (see footnote \[3\]), raises \( u_{ij}^{FCM} \)’s to the \( q \)th power. However, in \( J_{PCM} \) no parameter \( q \) is involved.}

\footnote{An alternative choice for \( \eta_j \)’s, given in \cite{9} is \( \eta_j = \frac{\sum_{u_{ij} > \gamma} \|x_i - \theta_j\|^2}{\sum_{u_{ij} > \gamma} 1} \), where \( \gamma \) is an appropriate threshold.}
hand, for points with $\|x_i - \theta_j\|^2/\eta_j > 1$ (i.e. that lie away from $\theta_j$) and as this ratio increases, $u_{ij}$ will decrease exponentially fast, as eq. (3) suggests (e.g., for $\|x_i - \theta_j\|^2/\eta_j = 3$, $u_{ij}$ will become “small” ($\approx 0.05$)). This justifies the choice of eq. (5). Of course, one should keep in mind that information about how the true clusters are spread in the data space is, in general, not known.

It is worth noting that, due to the independence between $u_{ij}$’s, $j = 1, \ldots, m$, for a specific $x_i$, the optimization problem solved by PCM can be decomposed into $m$ sub-problems, each one optimizing a specific $J_j$ function (see eq. (2)). Considering the representative $\theta_j$ associated with a given $J_j$, we have from eq. (5) that points that lie closer to the cluster representative will have larger degrees of compatibility with $C_j$. On the other hand, eq. (4) implies that the new position of $\theta_j$ is mainly specified by the data points that are most compatible with $C_j$. It is not difficult to see that such a coupled iteration leads the representative $\theta_j$ towards the center of the “dense in data” region that lies closer to its initial position. Thus, in principle and for appropriate choices of $\eta_j$’s, PCM leads the cluster representatives to regions that are “dense in data points”.

B. PCM issues and potential solutions

Having described the main characteristics of the algorithm and the rationale behind them, let us focus now on some issues that a user faces with PCM. The first one concerns the $m$ parameters $\eta_j$’s. An improper choice of $\eta_j$’s may lead PCM to failure in identifying a sparse cluster that is located very close to a denser cluster (see experiment 1, in section IV-A), or it may even lead the algorithm to recover the whole data set as a single cluster [16]. Referring to eq. (5), the $u_{ij}$’s produced by the FCM ($u_{ij}^{FCM}$’s), are not always accurate (e.g. in the presence of noise, [10]). In addition, the choice of the parameter $K$ is clearly data-dependent and there is no general clue on how to select it. In order to deal with this problem, [12] proposes the replacement of all $\eta_j$’s by a single quantity that is controlled by only two parameters: (a) the number of clusters and (b) a parameter that plays a “fuzzifier” role.

An additional source of inconveniences concerning $\eta_j$’s is the fact that, once they have been set, they remain fixed during the execution of PCM. This reduces the ability of the algorithm to track the variations in the clusters formation during its evolution. A way out of
this problem is to allow $\eta_j$’s to vary during the execution of the algorithm. A hint on this issue has been given in [9], but, to the best of our knowledge, no further work has been done towards this direction.

The second issue, which is related with the first one, is that of coincident clusters. As stated before, with a proper choice of $\eta_j$’s, PCM drives, in principle, the cluster representatives towards the dense in data regions that are closer to their initial positions. Therefore, if two or more representatives are initialized close to the same dense region, they will move towards its center, i.e., all of them will represent the same cluster. Alternatively, one could say that the clusters represented by these representatives are coincident\(^6\). This situation arises due to the absence of dependence between the coefficients $u_{ij}$, $j = 1, \ldots, m$, associated with a specific $x_i$ (see eq. (3)), which, as an indirect consequence, leads the representatives to move independently from each other (see eq. (4)). Note that such an issue does not arise in FCM due to the sum-to-one constraint imposed on the $u_{ij}$’s associated with each $x_i$. Several ways to deal with this problem have been proposed in the literature. More specifically, in [11], a variation of PCM is proposed, named PFCM, which combines concepts from PCM and FCM. Relative approaches are discussed in [15], [18], [19], while other approaches are proposed in [13], [20].

A common feature in all the previously mentioned works, is that condition (C3), which basically requires all clusters to be non-empty, is respected. Thus, in all the algorithms, the true number of clusters $m$ is implicitly required, in order to give them the ability to recover all clusters, without, hopefully, return coincident clusters. Thus, the requirement of the knowledge of the number of clusters is still here in disguise. A conceptually simple solution to address this requirement, while respecting condition (C3), comes from the PCM itself. Specifically, one could run the original PCM with an overestimated number of cluster representatives which will be initialized appropriately (at least one representative should lie at each dense in data region). Then, after a proper selection of $\eta_j$’s, PCM will (hopefully) recover the physical clusters, that is, it will move at least one representative to each dense region. Then, an additional step is required in order to identify coincident clusters and remove duplicates.

\(^6\)This point of view explains the term “coincident clusters”.

December 12, 2014 DRAFT
This idea has been partially discussed in [10], without, however, proposing explicitly to run the algorithm with an overdetermined number of clusters. However, in this case a reliable method for identifying duplicate clusters should be invented.

In this work an alternative solution for facing the previous issues is proposed, that is, to remove completely condition (C3), i.e., to allow some clusters to become “empty”. Doing so, one can initialize PCM with an overestimated number of (appropriately initialized) cluster representatives and (after some suitable modifications) as the algorithm evolves, some clusters will become empty and will be removed during its execution. Thus, the algorithm is likely to end up with the correct number of clusters and, of course, with no coincident clusters.

Having all the previous discussion in mind, in the present work, we propose a new PCM algorithm, called *Adaptive PCM (APCM)* with the following features:

- Each parameter $\eta_j$ is adjusted properly at each iteration of the algorithm. This is done by taking into account at each iteration only the data points that are most compatible with $C_j$. This gives the algorithm increased flexibility to track more accurately the variations in the formation of the clusters, as it evolves. In addition, in the proposed definition for $\eta_j$’s, each one of them is computed as a mean absolute deviation measure, and not as a variance measure.

- APCM is executed on an appropriately scaled version of the data set, which is specified by the data themselves and a certain parameter which can be easily determined. As we will see later, this scaling is a consequence of the fact that the definition of $\eta_j$’s is based on Euclidean and not squared Euclidean distances.

- Due to the definition and the adaptation of $\eta_j$’s, the algorithm allows the possibility for a cluster to become empty, by reducing its corresponding $\eta_j$ towards zero. This allows us to start with an overestimated number of clusters and end up with the true one. Thus, in APCM the number of clusters is also considered to be a time-varying quantity.

In the sequel, we proceed with the detailed description of the proposed APCM algorithm.

### III. THE ADAPTIVE PCM (APCM)

In this section, we describe in detail the various stages of the algorithm. Specifically, we first describe the way its parameters are initialized. Next, we comment on the updating of
its parameters \((u_{ij}`s, \theta_j`s, \eta_j`s, m)\) and we discuss in detail, how the initial estimate of the number of clusters can be reduced to the true one, by exploiting the adaptation of \(\eta_j`s\).

A. Initialization in APCM

As mentioned previously, first, we make an overestimation, denoted by \(m_{ini}\), of the true number of clusters \(m\), underlying in the data set. Regarding \(\theta_j`s\) and \(\eta_j`s\), their initialization drastically affects the final clustering result in PCM. Thus, a good starting point for them is of crucial importance. To this end, the initialization of \(\theta_j`s\) is carried out using the final cluster representatives obtained from the FCM algorithm, when the latter is executed with \(m_{ini}\) clusters. Taking into account that FCM is likely to drive the representatives to “dense in data” regions (since \(m_{ini} > m\)), the probability of at least one of the initial \(\theta_j`s\) to be placed in each dense region (cluster) of the data set, increases with \(m_{ini}\).

After the initialization of \(\theta_j`s\), we propose to initialize \(\eta_j`s\) as follows:

\[
\eta_j = \frac{\sum_{i=1}^{N} u_{ij}^{FCM} \|x_i - \theta_j\|}{\sum_{i=1}^{N} u_{ij}^{FCM}}, \quad j = 1, \ldots, m_{ini}
\]

where \(\theta_j`s\) and \(u_{ij}^{FCM}`s\) in eq. (6) are the final parameter estimates obtained by FCM.\(^7\)

It is worth noting that the above initialization of \(\eta_j`s\) involves Euclidean instead of squared Euclidean distances, as is the case with the classical PCM algorithm. This gives the algorithm the agility to deal well with closely located clusters, provided that an appropriate scaling on the data set is performed first. The scaling issue will be discussed in detail in subsection 3.3.

B. Parameter adaptation in APCM

In the proposed APCM algorithm, all parameters are adapted during its execution. More specifically, this refers to, (a) the update of the degrees of compatibility \(u_{ij}`s\) and the cluster representatives \(\theta_j`s\), (b) the adjustment of the number of clusters and (c) the adaptation of \(\eta_j`s\), with (b) and (c) being two interrelated processes.

First, the updating of \(u_{ij}`s\) and \(\theta_j`s\) is done as in the original PCM scheme according to eqs. (3) and (4). Concerning the adjustment of the number of clusters \(m(t)\) at \(t\)th iteration, we

\(^7\)An alternative initialization for \(\theta_j`s\) and \(\eta_j`s\) is proposed in [17].
proceed as follows. Let label be a $N$-dimensional vector, whose $i$th element is the index of the cluster which is most compatible with $x_i$, that is the index $j$ for which $u_{ij} = \max_{r=1,...,m(t)} u_{ir}$.

At each iteration of the algorithm, the adjustment (reduction) of the number of clusters $m(t)$ is achieved by examining, for each cluster $C_j$, if its index $j$ appears in the vector label. If this is the case (i.e. if there exists at least one vector $x_i$ that is most compatible with $C_j$), $C_j$ is preserved. Otherwise, $C_j$ is eliminated and, thus, $U$, $\Theta$ are updated accordingly. As a result, the current number of clusters $m(t)$ is reduced (see Possible cluster elimination part in Algorithm 1).

Finally, in contrast to the classical PCM, the parameters $\eta_j$’s are adapted at each iteration of the APCM algorithm. More specifically, we propose to compute the parameter $\eta_j$ of a cluster $C_j$ at each iteration, as the mean absolute deviation of the most compatible to cluster $C_j$ data vectors, i.e.,

$$
\eta_j(t+1) = \frac{1}{n_j(t)} \sum_{x_i: u_{ij}(t)=\max_{r=1,...,m(t+1)} u_{ir}(t)} \|x_i - \mu_j(t)\| \tag{7}
$$

where $n_j$ denotes the number of the data points $x_i$ that are most compatible with the cluster $C_j$ and $\mu_j$ the mean vector of these data points (see also Adaptation of $\eta_j$’s part in Algorithm 1).

Note that, the proposed updating mechanism for $\eta_j$’s differs from others used in the classical PCM, as well as in many of its variants, in two distinctive points. First, $\eta_j$’s in APCM are updated taking into account only the data vectors that are most compatible to cluster $C_j$ and not all the data points weighted by their corresponding $u_{ij}$ coefficients. Second, the distances involved in eq. (7) are between a data vector and the mean vector $\mu_j(t)$ of the most compatible points of the cluster; not from the representative $\theta_j$, as in previous works (e.g. [9], [15]). This allows more accurate estimates of $\eta_j$’s, since $\mu_j(t)$ is expected to be close to the next location of $\theta_j$. This is crucial mainly during the first few iterations of the algorithm where the position of $\theta_j$ may vary significantly from iteration to iteration. It is also noted that, in the (rare) case where there are two or more clusters, that are equally compatible with a specific $x_i$, then $x_i$ will contribute to the determination of the parameter $\eta$ of only one of them, which is chosen arbitrarily.

Let us now explain how the above adaptation of $\eta_j$’s leads to the gradual reduction of $m_{ini}$.
towards the true number of clusters, $m$. Taking into account that, (a) all representatives are forced to be moved to dense in data regions, due to the possibilistic nature of APCM and (b) the probability to select as initial representative at least one point in each dense region is increased, due to the way the representatives are initially selected, then it is very likely that at least one representative moves towards each dense region. Assuming that two cluster representatives $\theta_r, \theta_s$ almost coincide in a certain dense region at a given iteration, but let’s say that $\eta_r > \eta_s$, then, for a given data point $x_i$ in this region, it is $\|x_i - \theta_r\|^2 \approx \|x_i - \theta_s\|^2$ and thus $\frac{\|x_i - \theta_r\|^2}{\eta_r} < \frac{\|x_i - \theta_s\|^2}{\eta_s}$. Considering the updating equation of $u_{ij}$’s (eq. (3)), it follows that $u_{ir}$ should be greater than $u_{is}$, and $x_i$ will contribute only to the adaptation of $\eta_r$. Thus, the influence of $C_r$ around $\theta_r$ will be increased, in contrast to that of $C_s$ around $\theta_s$, which will be decreased. Under the same reasoning, at the next iteration, it is more likely for $C_s$ to “lose” additional points from $C_r$, i.e., fewer and fewer points will be most compatible with $C_s$ as the algorithm evolves, leading gradually to its final elimination.

Algorithm 1 $[\Theta, U, \text{label}] = APCM(X, m_{ini}, \alpha)$

Input: $X$, $m_{ini}$, $\alpha$

\(\triangleright\) Initialization of $\theta_j$’s part

1: Initialize: $\theta_j$ via FCM (see subsection 3.1)

\(\triangleright\) Initialization of $\eta_j$’s part

2: Set: $\eta_j = \frac{\sum_{i=1}^{n} u_{FCM}^{ij} \|x_i - \theta_j\|}{\sum_{i=1}^{n} u_{FCM}^{ij}}$, $j = 1, \ldots, m_{ini}$

3: Set: $\hat{\eta} = \min_{j=1,\ldots,m_{ini}} \eta_j$

\(\triangleright\) Scaling part

4: $X \leftarrow \frac{\alpha}{\hat{\eta}} X$

5: $\theta_j \leftarrow \frac{\alpha}{\hat{\eta}} \theta_j$, $j = 1, \ldots, m_{ini}$

6: $\eta_j \leftarrow \frac{\alpha}{\hat{\eta}} \eta_j$, $j = 1, \ldots, m_{ini}$

7: $t = 0$

8: $m(t) = m_{ini}$

9: $\theta_j(t) = \theta_j$, $j = 1, \ldots, m(t)$

10: $\eta_j(t) = \eta_j$, $j = 1, \ldots, m(t)$

11: repeat
▷ Update U part
12: \[ u_{ij}(t) = \exp \left( -\frac{||x_i - \theta_j(t)||^2}{\eta_j(t)} \right), \quad i = 1, \ldots, N, \quad j = 1, \ldots, m(t) \]
▷ Update Θ part
13: \[ \theta_j(t + 1) = \frac{\sum_{i=1}^{N} u_{ij}(t) x_i}{\sum_{i=1}^{N} u_{ij}(t)}, \quad j = 1, \ldots, m(t) \]
▷ Possible cluster elimination part
14: \textbf{for } i \leftarrow 1 \textbf{ to } N \textbf{ do}
15: \textbf{Determine: } u_{ir}(t) = \max_{j=1, \ldots, m(t)} u_{ij}(t)
16: \textbf{Set: } \text{label}(i) = r
17: \textbf{end for}
18: \begin{align*}
p &= 0 & //number of removed clusters at iteration t
\end{align*}
19: \textbf{for } j \leftarrow 1 \textbf{ to } m \textbf{ do}
20: \quad \textbf{if } j \notin \text{label} \textbf{ then}
21: \quad \textbf{Remove: } C_j
22: \quad p = p + 1
23: \textbf{end if}
24: \textbf{end for}
25: \[ m(t + 1) = m(t) - p \]
▷ Adaptation of η_j’s part
26: \[ \eta_j(t + 1) = \frac{1}{\eta_j(t)} \sum_{x_i: u_{ij}(t) = \max_{r=1, \ldots, m(t+1)} u_{ir}(t)} ||x_i - \mu_j(t)||, \quad j = 1, \ldots, m(t + 1) \]
27: \[ t = t + 1 \]
28: \textbf{until} the change in θ_j’s between two successive iterations becomes very small
29: \textbf{Map } X, \Theta, \eta_j’s \textbf{ back to the original space}
30: \textbf{return } \Theta, U, \text{label}

C. Scaling the data set

Due to the modifications made in the original PCM, APCM must be applied to a properly scaled version of the data set under study. In this subsection, we will discuss the need for scaling the data set, as well as its implications.
1) The need for scaling: Let us comment first on the influence of the Euclidean distance (instead of the squared Euclidean one) to the definition of $\eta_j$’s. In the following discussion, we focus on a specific iteration of APCM. Thus, the iteration index $t$ is omitted. Then, the updating equation for $\eta_j$’s used by APCM is written as

$$\eta_j = \frac{1}{n_j} \sum_{x_i; u_{ij} = \max_{r=1,\ldots,m} u_{ir}} \|x_i - \mu_j\|$$

and we compare it with the

$$\eta'_j = \frac{1}{n_j} \sum_{x_i; u_{ij} = \max_{r=1,\ldots,m} u_{ir}} \|x_i - \mu_j\|^2$$

Eq. (9) can be obtained by the definition of parameters $\eta_j$’s in PCM (see eq. (5)), for $K = 1$, by setting the maximum $u_{ij}$ for a given $x_i$ equal to 1 and the remaining $u_{ij}$’s of $x_i$ equal to 0. The latter is a situation that practically arises in cases where very well separated clusters are formed by the data points. Although this is not the general case, it is a reasonable approximation that will help us in understanding the difference in using Euclidean instead of squared Euclidean distance. In addition, we will consider the non-adaptive case, i.e., $\eta_j$ and $\eta'_j$ remain constant during the execution of the algorithm.

For the present section only, $u'_{ij}$ denotes the degree of compatibility of $x_i$ with $C_j$, when $\eta'_j$ (eq. (9)) is taken into account and $u_{ij}$ denote the degree of compatibility of $x_i$ with $C_j$, when $\eta_j$ (eq. (8)) is used.

Let us consider a clustering problem where two compact clusters are involved with their corresponding representatives $\theta_1$ and $\theta_2$ located at the centers of the clusters. Let us focus on the cluster $C_1$ and draw the circle $Circle_1'$ centered at $\theta_1$ and having radius equal to $\sqrt{\eta_1'}$ (see Fig. 1). Then, for all points $x'_i$ on this circle, it is $u'_{i1} = \exp(-1) \approx 0.37$. Let us consider next the circle $Circle_1$ centered also at $\theta_1$ and having radius equal to $\sqrt{\eta_1}$. Then, for all points $x_i$ on this circle, it is $u_{i1} = \exp(-1) \approx 0.37$.

The question now is “Which of the $Circle_1'$ and $Circle_1$ lies inside the other?”. If $Circle_1$ lies in $Circle_1'$ (Fig. 1a), this implies that for a specific data point $x_i$, using $\eta_1$ in eq. (3)
Fig. 1: Two clusters and their corresponding \( \eta_j \)'s and \( \eta'_j \)'s values, in cases where (a) \( 1 < \eta_j < \eta'_j \) and (b) \( \eta'_j < \eta_j < 1 \) (see text for explanation).

This gives less degree of compatibility to \( x_i \) with respect to \( \theta_1 \), compared to the case where \( \eta'_1 \) is used. Taking into account the definition of \( u_{ij} \) from eq. (3), we can deduce that \( u_{i1} \) decreases more rapidly than \( u'_{i1} \), as \( x_i \) moves away from \( \theta_1 \) (see Fig. 2a). On the other hand, if \( Circle_1 \) lies outside \( Circle'_1 \) (Fig. 1b), \( u_{i1} \) decreases slower than \( u'_{i1} \), as \( x_i \) moves away from \( \theta_1 \) (see Fig. 2b). This implies that with \( \eta_1 \), points that are not too close to \( \theta_1 \) contribute more to the determination of the new location of \( \theta_1 \) (see eq. (4)), than with \( \eta'_1 \).

Fig. 2: Plots of the degrees of compatibility \( u_{i1} \) and \( u'_{i1} \), with respect to \( ||x_i - \theta_1||^2 \), for the example illustrated in Fig. 1, when (a) \( 1 < \eta_1 < \eta'_1 \) and (b) \( \eta'_1 < \eta_1 < 1 \).
To answer the previous question, we need to compare the radius of the circles, $\sqrt{\eta'_1}$ and $\sqrt{\eta_1}$, or equivalently, their squares. Assuming that the same points contribute to the determination of $\eta_1$ and $\eta'_1$ via eqs. (8) and (9), respectively, we have

$$\frac{\eta'_1}{\eta_1} = \frac{\sum_{x_i:u_{i1}=\max_{r=1,...,m} u_{ir}} \|x_i - \mu_1\|^2}{\sum_{x_i:u_{i1}=\max_{r=1,...,m} u_{ir}} \|x_i - \mu_1\|}$$  \hspace{1cm} (10)

Clearly, the above ratio may be either less or greater than 1. Indeed, if $\|x_i - \mu_1\| < 1$ for all $x_i$, then $\eta'_1 < \eta_1$ ($< 1$). In this case, APCM allows to data points that are not too close to $\theta_1$, to contribute more to the determination of the new location of $\theta_1$ (through eq. (4)), as compared to PCM. The phenomenon becomes more intense as $\eta_1$ takes smaller values ($< 1$). On the other hand, if $\|x_i - \mu_1\| > 1$ for all $x_i$, then $\eta'_1 > \eta_1$ ($> 1$). In this case, APCM allows only to data points that are very close to $\theta_1$, to contribute more to the determination of the new location of $\theta_1$ (through eq. (4)), as compared to PCM. This phenomenon becomes more intense as $\eta_1$ moves to higher values ($> 1$).

Note that in the previous reasoning, we use the original PCM as a “reference point” in order to understand the operation of APCM. Focusing now on APCM exclusively, we consider again the two-cluster clustering problem, considered before. Let us examine now the following two extreme cases where the range of values of the data set is (a) much smaller than 1 (e.g., $x_i \in [s_1, s_2]$, with $|s_2 - s_1| \ll 1$) and (b) much larger than 1 (e.g., $x_i \in [t_1, t_2]$, with $|t_2 - t_1| \gg 1$). Typical values for $|s_2 - s_1|$ and $|t_2 - t_1|$ are at least one order of magnitude less and greater than 1, respectively. In case (a), it will be $\eta_1 \ll 1$. Since all the distances between any two points in this range will be less than 1, the ratios $\frac{d_{ij}}{\eta_j} = \frac{\|x_i - \theta_j\|^2}{\eta_j}$ will be significantly smaller than 1, since the numerator is of the squared order compared to the denominator. Thus, $u_{i1}$’s will take very large values even for points that are away from $\theta_1$. As a consequence, $\theta_1$ is likely to move to the center of the data set (through eq. (4)). For the same reasons, this will be the case for $\theta_2$ also. In case (b), it is expected that $\eta_1 \gg 1$ (for not degenerative cases). In this case, only a very tiny set of points around $\theta_1$ will contribute to the determination of the new location of it. This situation may cause serious difficulties to the movement of $\theta_1$ towards a dense in data region, in case where $\theta_1$ has not reached yet close to the center of such a region. In fact, the local data point density around the current
location of \( \theta_1 \) may be low, so that the points that are very close to it may be unable to affect significantly the movement of \( \theta_1 \) towards the center of a dense region.

From the above discussion, it is clear that such extreme situations are undesirable. A way out of this problem, is to scale the data set, so that APCM will have the ability to determine the next location of each \( \theta_j \) based on a set of closely located data points that is not too small to hamper the movement of \( \theta_j \) towards a dense region and not too large to consider closely located clusters as a single one. Specifically, we propose the following scaling (see also Algorithm [1]):

\[
X_{\text{scaled}} = \frac{\alpha}{\hat{\eta}} X
\]

\[
\theta_{j,\text{scaled}} = \frac{\alpha}{\hat{\eta}} \theta_j
\]

\[
\eta_{j,\text{scaled}} = \frac{\alpha}{\hat{\eta}} \eta_j
\]

where \( \hat{\eta} = \min_{j=1,...,m_\text{ini}} \eta_j \). From eq. (13), it follows that \( \alpha \) equals the minimum \( \eta_{j,\text{scaled}} \).

Note that, as shown in Algorithm [1], this scaling takes place just after the initialization stage of APCM. Letting \( x_{i,\text{scaled}} \) and \( \theta_{j,\text{scaled}} \) denote the scaled versions of the data vector \( x_i \) and the representative \( \theta_j \), respectively, it is

\[
d_{ij,\text{scaled}} \equiv \|x_{i,\text{scaled}} - \theta_{j,\text{scaled}}\|^2 = \left(\frac{\alpha}{\hat{\eta}}\right)^2 \|x_i - \theta_j\|^2 = \left(\frac{\alpha}{\hat{\eta}}\right)^2 d_{ij},
\]

\[
\eta_{j,\text{scaled}} = \frac{1}{n_j} \sum \|x_{i,\text{scaled}} - \mu_{j,\text{scaled}}\| = \frac{\alpha}{\hat{\eta}} \left(\frac{1}{n_j} \sum \|x_i - \mu_j\|\right) = \frac{\alpha}{\hat{\eta}} \eta_j
\]

Thus, after scaling, the degree of compatibility of a data point \( x_{i,\text{scaled}} \) with \( \theta_{j,\text{scaled}} \) will be

\[
u_{ij,\text{scaled}} = \exp \left( -\frac{d_{ij,\text{scaled}}}{\eta_{j,\text{scaled}}} \right) = \exp \left( -\frac{\alpha}{\hat{\eta}} \frac{d_{ij}}{\eta_j} \right) = \frac{\alpha}{\hat{\eta}} \nu_{ij}
\]

Reconsidering now the previous first extreme case, i.e., when the data set is in a very narrow range such that \( \eta_j \ll 1 \), it follows that \( \frac{1}{\hat{\eta}} \gg 1 \). In such a case, setting \( \alpha \geq 1 \), we have that \( \nu_{ij,\text{scaled}} < \nu_{ij} \), which means that the situation where very distant data points contribute significantly to the determination of \( \theta_j \), is discouraged. On the other hand, in the second extreme case, where the data set is in a range such that \( \eta_j \gg \), it follows that \( \frac{1}{\hat{\eta}} \ll 1 \). In
this case, by setting $\alpha \leq 1$, we have that $u_{ij}^{scaled} > u_{ij}$ (pre-scaled), which means that the situation where only a very tiny set of points around $\theta_j$ contribute to the determination of its new location, is discouraged.

From the above discussion, it follows that suitable values for $\alpha$ are of the order of 1. However, in the idealized scenario where the previous discussion has been based, we considered the case where the number of selected representatives was equal to 2, that is, as many as the actual number of clusters, $m$. What happens when the number of initial representatives, $m_{ini}$, is set greater than the actual number of clusters?

In such a case, when $m_{ini}$ is large, the number of data points that are most compatible with each representative becomes small. Thus, the corresponding $\eta_j$’s (eq. (8)) will take small values, which implies reduced influence around the corresponding representatives. That is, for fixed $\alpha$, the ratio $\alpha/\hat{\eta}$ increases as $m_{ini}$ increases and as is deduced from eq. (16), $u_{ij}^{scaled}$ decreases more rapidly than $u_{ij}$. Therefore, very few data points around $\theta_j$ will affect significantly to the determination of its new position. As it is stated before, this may hamper the movement of $\theta_j$ towards the center of a dense region. A way out for this situation is to decrease $\alpha$. On the other hand, small values of $m_{ini}$ (but no less than the actual number of clusters, $m$), lead to greater values for $\eta_j$’s. Combined with smaller values of $\alpha$, $u_{ij}^{scaled}$ increases, i.e., several data points are allowed to contribute to the determination of the new position of a representative, even though they are not too close to it. This may cause a failure to the algorithm in identifying closely located clusters. Thus, in the case of small values for $m_{ini}$, $\alpha$ should be increased.

As it is clear, there exists a trade-off between $m_{ini}$ and $\alpha$: large (small) values of $m_{ini}$ require small (large) values of $\alpha$, in order for the APCM algorithm to work (in principle) properly. In other words, $m_{ini}$ and $\alpha$ vary inversely proportional to each other. The next question is what can be said in more “quantitive terms” for this inverse proportionality relationship. After extensive experimentation (see also section IV), it has been found that for values of $m_{ini}$ that are about three of four times greater than the true number of clusters, $m$, values of $\alpha$ around 1 suffice in several cases for the proper operation of APCM.

Before closing this subsection, it is worth noting that PCM, in contrast to APCM, is not affected by the above type of scaling. This follows by the fact that $\eta_j^{scaled} = (\frac{\alpha}{\hat{\eta}})^2 \eta_j$. 
(as can be easily verified by eqs. (9),(11)), where \( \eta_j' \) refers to the pre-scaled space and \( \hat{\eta}_j' = \min_{j=1,...,m_{ini}} \eta_j' \). Moreover, for a specific point \( x_i \), it is \( d_{ij}^{scaled} = (\frac{\alpha}{\eta_j'})^2 d_{ij} \), where \( d_{ij} \) is the squared Euclidean distance in the pre-scaled space. Thus, \( u_{ij}^{scaled} = \exp(-\frac{d_{ij}^{scaled}}{\eta_j'}) = \exp(-\frac{d_{ij}}{\eta_j'}) = u_{ij} \). This also justifies the use of PCM as a reference point for better understanding the behavior of APCM.

It should be noted that \( \alpha \) plays a role similar to the parameter (fuzzifier) \( q \) involved in \( J_{PCM}' \). However, in contrast to the original PCM scheme, choosing the initial number of clusters to be 3-4 times greater than the true number of clusters, a value of \( \alpha \) around 1 works in several cases. Thus, the choice of \( \alpha \) is not arbitrary, and, as a consequence, its determination does not require exhaustive fine tuning.

2) The effects of scaling: After the previous analysis where the need for scaling has been justified, a question that naturally arises now is “How the different choices of \( \alpha \) affect the behavior of APCM?”, in more quantitative terms. We will try to give an answer to this question, via a very simple clustering example. Specifically, we consider an one-dimensional data set consisting of two Gaussian clusters with 50 points each, shown on the \( x \)-axis in Fig. 3a. The centers of the clusters are at locations 28 and 67 and their variances are 100 and 121, respectively. We consider two cases: in the first the number of initial representatives is \( m_{ini} = 3 \) and in the second \( m_{ini} = 10 \). We run first the FCM algorithm for each such case and we obtain the resulting \( u_{ij}^{FCM} \)'s and \( \theta_j \)'s, from which the (pre-scaled) \( \eta_j \)'s are computed using eq. (6). Note that, in general, we expect larger values of \( \eta_j \)'s, in case of \( m_{ini} = 3 \) compared to the case where \( m_{ini} = 10 \), since the sum-to-one constraint of the FCM will restrict the \( u_{ij}^{FCM} \)'s to smaller values in the second case, compared to the first. Then, for several values of \( \alpha \), we perform scaling on the data set as explained before, taking the scaled versions of \( X, \theta_j \)'s and \( \eta_j \)'s (eqs. (11)-(13)). In addition, the \( u_{ij} \)'s between each data vector \( x_i \) and each representative \( \theta_j \) are computed (based on the scaled data) using eq. (3).

Due to the independence among the \( J_j \)'s in eq. (2), the minimization of \( J \) with respect to \( \theta_j \)'s results from the minimization of each individual \( J_j \) with respect to the corresponding \( \theta_j \). In the sequel, we focus on the \( J_j \) that corresponds to the minimum (scaled) \( \eta_j \), which is equal to \( \alpha \). The “subcost” function \( J_j(\theta_j) = \sum_{i=1}^{N} u_{ij} \| x_i - \theta_j \|^2 + \alpha \sum_{i=1}^{N} (u_{ij} \ln u_{ij} - u_{ij}) \) is
Fig. 3: Plot of the APCM cost function for the case of a two-class one-dimensional data set. Plot shows (a) the data set in the original space. Data points are denoted by stars on the $x$-axis and representatives by black dots. Results for (b) $m_{ini} = 3$, $\alpha = 0.05$, (c) $m_{ini} = 3$, $\alpha = 1$, (d) $m_{ini} = 3$, $\alpha = 2$, (e) $m_{ini} = 10$, $\alpha = 0.05$, (f) $m_{ini} = 10$, $\alpha = 1$ and (g) $m_{ini} = 10$, $\alpha = 2$ in the respective scaled spaces. Note that the range of values in the $x$-axis varies according to the scaling performed each time.

plotted with respect to $\theta_j$, for various values of $\alpha$\(^\text{10}\). We consider first $m_{ini} = 3$, i.e., $m_{ini}$ is very close to the number of actual clusters ($m = 2$). We plot $J_j$ after scaling the data set with $\alpha = 0.05$ (Fig. 3b), $\alpha = 1$ (Fig. 3c) and $\alpha = 2$ (Fig. 3d). In Fig. 3b we have just a single valley centered at the mean of the data set. Minimization of $J_j$ will lead $\theta_j$ to this position, which means that in this case the algorithm will fail to detect any of the two true clusters. This case corresponds to the situation where all data points have significant

\(^{10}\)We write $J_j(\theta_j)$ to denote explicitly the dependence of $J_j$ on $\theta_j$. [December 12, 2014 DRAFT]
degree of compatibility \( u_{ij} \) with all representatives, leading them gradually to the center of the data set. In Fig. 3c, we see two well formed valleys centered at the centers of the two natural clusters. Minimization of \( J_j \) will lead \( \theta_j \) to the center of a true cluster. Thus, for \( \alpha = 1 \), APCM works properly. Observe that in this case APCM will drive \( \theta_j \) to the center of an actual cluster, without requiring a strictly good initialization for it. A similar situation is shown in Fig. 3d for \( \alpha = 2 \), although the form of the valleys is a bit disturbed. In conclusion, when \( m_{ini} \) is close to \( m \) there exists a significant range of values of \( \alpha \) (around 1) that allow the algorithm to work properly.

In the case where \( m_{ini} = 10 \) (that is \( m_{ini} \gg m(= 2) \)) the situation changes. In this case, two well-formed valleys appear for the very small value \( \alpha = 0.05 \) (Fig. 3e), whereas for values of \( \alpha \) around 1, \( J_j(\theta_j) \) exhibits many local minima. As explained before, this is due to the fact that very few data points around \( \theta_j \) are allowed to affect significantly the determination of its new position, in cases where \( m_{ini} \gg m \). Thus, \( \theta_j \) is likely to get stuck to local minima that do not correspond to any of the two true clusters (it is worth noting that similar arguments hold also for all the other \( J_j \) functions). In this case, APCM is most likely to end up with more than 2 clusters, which does not correspond to the underlying structure of the data set.

Interestingly and loosely speaking, one can say that the behavior of the algorithm for \( m_{ini} = 10 \) seems to be a “shifted towards the right” version of the \( m_{ini} = 3 \) case with respect to \( \alpha \).

This example verifies that in cases where \( m_{ini} \) is chosen not very larger than the actual number of clusters \( m \), appropriate values for the parameter \( \alpha \) are around 1. On the other hand, when \( m_{ini} \) is chosen much larger than \( m \), parameter \( \alpha \) should be set equal to a value significantly smaller than 1. However, in more demanding data sets, which contain very closely located natural clusters and for a given value of \( m_{ini} \), larger values for the parameter \( \alpha \) should be chosen, compared to cases of less closely located clusters. Experiment showed that values of \( \alpha \) around 1 and up to 2 suffice for almost any data set, provided that \( m_{ini} \) is not extremely larger than \( m \) (about 3-4 times larger).
IV. EXPERIMENTAL RESULTS

In this section, we assess the performance of the proposed method in several experimental settings and illustrate the results. More specifically, we consider two series of experiments. In the first one, we use two-dimensional simulated data sets in order to exhibit more clearly certain aspects of the behavior of the APCM itself. In the second one, we use both simulated and real-world data sets (Iris [21], New Thyroid [21], and a hyperspectral image data set [22]) to evaluate the performance of APCM in comparison with several other related algorithms.

A. Behavior of the APCM

**Experiment 1**: Let us consider a two-dimensional data set consisting of $N = 17$ points, which form two clusters $C_1$ and $C_2$ with 12 and 5 data points, respectively (see Fig. 4). The means of the clusters are $c_1 = [1.75, 2.75]$ and $c_2 = [4.25, 2.75]$. In this experiment, we consider the PCM and the APCM algorithms, with $m_{ini} = 2$ and $\alpha = 1$ for APCM. Figs. 4a and 4d show the initial positions of the cluster representatives that are taken from the FCM clustering algorithm and the circles with radius equal to $\sqrt{\eta_j}$’s resulting from eq. (5) (for $K = 1$) and eq. (6) for PCM and APCM algorithm, respectively. Similarly, Figs. 4b and 4e show the new locations of $\theta_j$’s after the first iteration of the algorithms and Figs. 4c, 4f show the locations of $\theta_j$’s after the eighth iteration of them. Table I shows the degrees of compatibility $u_{ij}$’s of all data points $x_i$’s with the cluster representatives $\theta_j$’s at the three iterations considered in Fig. 4 (initial, 1st, 8th).

As it can be deduced from Table I and Fig. 4, the degrees of compatibility of the data points of $C_1$ with the cluster representative $\theta_2$ increase as PCM evolves, leading gradually $\theta_2$ towards the region of the cluster $C_1$ and thus, ending up with two coincident clusters, although $\theta_1$ and $\theta_2$ are initialized properly through the FCM algorithm (see Fig. 4a). However, this is not the case in APCM algorithm, as both the cluster representatives remain in the centers of the actual clusters. Obviously, this differentiation on the behavior of the two algorithms is due to the different definition of the parameters $\eta$’s, which affect the degrees of compatibility of the data points with each cluster (see eqs. (5), (7) and (3)). This experiment indicates that, in principle, APCM can handle successfully cases where relatively closely located clusters with different densities are involved.
Fig. 4: PCM and APCM snapshots at their initialization step and their first and eighth iteration (Experiment 1).

**Experiment 2:** Let us consider now a two-dimensional data set consisting of $N = 1100$ points, which form three clusters $C_1$, $C_2$ and $C_3$ (see Fig. 5). Each cluster is modelled by a normal distribution. The (randomly generated) means of the distributions are $c_1 = [1.35, 0.23]^T$, $c_2 = [4.03, 4.09]^T$ and $c_3 = [4.64, 2.28]^T$, respectively, while their (common) covariance matrix is set equal to $0.4 \cdot I_2$, where $I_2$ is the $2 \times 2$ identity matrix. A number of 500 points are generated by the first distribution and 300 points are generated by each one of the other two distributions. Note that $C_2$ and $C_3$ clusters are very close to each other and, therefore, their discrimination is considered as a difficult task for a clustering algorithm. Table II shows the ranges of values of the parameter $\alpha$, for which APCM manages to identify correctly the naturally formed $m = 3$ clusters, for various values of $m_{ini}$, i.e. it does not merge $C_2$ and $C_3$. Fig. 5 shows the clustering results of the APCM algorithm, when it is initialized with $m_{ini} = 5$, in cases where scaling is performed with (a) $\alpha = 0.5$, (b) $\alpha = 1.0$.
TABLE I: The degrees of compatibility of the data points of Experiment 1 for PCM and APCM algorithms, after: (a) initialization (common to both algorithms), (b) first iteration and (c) eighth iteration.

| x_i  | Initialization PCM/APCM | 1st iteration PCM | 1st iteration APCM | 8th iteration PCM | 8th iteration APCM |
|------|-------------------------|------------------|-------------------|-----------------|------------------|
| (1.5, 3, 5) | 0.9292 0.0708 | 0.3701 0.0018 | 0.2757 1.6e-06 | 0.3606 0.0118 | 0.2453 3.1e-09 |
| (2.0, 3, 5) | 0.8963 0.1037 | 0.3526 0.0127 | 0.2590 9.6e-05 | 0.3630 0.0570 | 0.2443 1.3e-06 |
| (1.0, 3, 0) | 0.9475 0.0525 | 0.3884 2.6e-04 | 0.2936 2.4e-08 | 0.3583 0.0024 | 0.2464 7.4e-12 |
| (1.5, 3, 0) | 0.9854 0.0146 | 0.8348 0.0027 | 0.7913 3.4e-06 | 0.8134 0.0174 | 0.7563 1.0e-08 |
| (2.0, 3, 0) | 0.9728 0.0272 | 0.7954 0.0188 | 0.7432 2.2e-04 | 0.8186 0.0846 | 0.7531 4.3e-06 |
| (2.5, 3, 0) | 0.8201 0.1799 | 0.3360 0.0897 | 0.2433 0.0600 | 0.3653 0.2766 | 0.2433 5.4e-04 |
| (1.0, 2, 5) | 0.9475 0.0525 | 0.3884 2.6e-04 | 0.2936 2.4e-08 | 0.3583 0.0024 | 0.2464 7.4e-12 |
| (1.5, 2, 5) | 0.9854 0.0146 | 0.8348 0.0027 | 0.7913 3.4e-06 | 0.8134 0.0174 | 0.7563 1.0e-08 |
| (2.0, 2, 5) | 0.9728 0.0272 | 0.7954 0.0188 | 0.7432 2.2e-04 | 0.8186 0.0846 | 0.7531 4.3e-06 |
| (2.5, 2, 5) | 0.8201 0.1799 | 0.3360 0.0897 | 0.2433 0.0600 | 0.3653 0.2766 | 0.2433 5.4e-04 |
| (1.5, 2, 0) | 0.9292 0.0708 | 0.3701 0.0018 | 0.2757 1.6e-06 | 0.3606 0.0118 | 0.2453 3.1e-09 |
| (2.0, 2, 0) | 0.8963 0.1037 | 0.3526 0.0127 | 0.2590 9.6e-05 | 0.3630 0.0570 | 0.2443 1.3e-06 |

and (c) \( \alpha = 3.0 \), respectively. Note from Table II that these values of parameter \( \alpha \) belong to the range where APCM identifies correctly the actual clusters, when \( m_{ini} = 5 \). Also, in Fig. 5, it is shown how \( \eta_j \)’s are affected by the various values of the scaling parameter \( \alpha \), when APCM is initialized with \( m_{ini} = 5 \).

Fig. 5: The clustering results of APCM for Experiment 2, when it is initialized with \( m_{ini} = 5 \), for various values of parameter \( \alpha \).

Executing APCM on the previous data set, for various values of \( m_{ini} \) and \( \alpha \), we end up with
the figure shown in Fig. 6, where regions in the $\alpha - m_{ini}$ plot are drawn with different colors, each one corresponding to different number of final clusters, $m_{final}$. The light-blue colored region corresponds to the case where $m_{final} = 3$, i.e., when APCM identifies correctly the underlying clusters. From the shape of this region, it can be deduced the “rule of thumb” stated already in section [III-C1] that relates $\alpha$ and $m_{ini}$: “$\alpha$ varies inversely proportional with respect to $m_{ini}$”. From this rule and Fig. 6, it can be seen that fixing $\alpha$ to a value around 1, if $m_{ini}$ is selected to be 3-4 times greater than the actual number of clusters, APCM will identify correctly the underlying clusters. Interestingly, the behavior depicted in Fig. 6 has also been observed for several other data sets. Thus, the above rule of thumb seems to hold more generally.

TABLE II: Range of values of the parameter $\alpha$, in which APCM concludes to $m_{final} = 3$ clusters, for specific values of $m_{ini}$ for Experiment 2.

| $m_{ini}$ | $\alpha_{min}$ | $\alpha_{max}$ |
|-----------|----------------|----------------|
| 3         | 0.35           | 5.00           |
| 5         | 0.34           | 4.79           |
| 10        | 0.30           | 1.29           |
| 20        | 0.28           | 1.00           |
| 50        | 0.23           | 0.25           |
| 100       | 0.20           | 0.24           |

Fig. 6: Graphical representation of the number of final clusters, $m_{final}$, returned by APCM for Experiment 2, for various combinations of $\alpha$ and $m_{ini}$.
B. Comparison of APCM with other algorithms

In the sequel, we compare the clustering performance of APCM with that of the k-means, the FCM, the PCM, the UPC [12], the PFCM [11] and the UPFC [19] algorithms, which all result from cost optimization schemes. For a fair comparison, the representatives $\theta_j$'s of all algorithms are initialized based on the FCM scheme and the parameters of each algorithm are first fine tuned. In order to compare a clustering with the true data label information, we use the Rand Measure (RM) (e.g. [2]), which can handle clusterings whose number of clusters may differ from the number of true data labels, and the Success Rate (SR), which measures the percentage of the points that have been correctly labeled by each algorithm. Moreover, the mean of the Euclidean distances (MD) between the true mean of each physical cluster $c_j$ and its closest cluster representative ($\theta_j$) obtained by each algorithm, is presented. In cases where a clustering algorithm ends up with a higher number of clusters than the actual one ($m_{\text{final}} > m$), only the $m$ cluster representatives that are closest to the true $m$ centers of the physical clusters, are taken into account in the determination of MD. On the other hand, in cases where $m_{\text{final}} < m$, the MD measure refers to the distances of all cluster representatives from their nearest actual center, i.e. some actual centers are ignored. It is noted that lower MD values indicate more accurate determination of the cluster centers locations. Finally, the number of iterations required for the convergence of each algorithm, is provided. Note that in all reported results for the UPC, the PFCM and the UPFC algorithms, clusters that coincide are considered as a single one.

We begin with a demanding simulated data set with classes exhibiting significant differences with respect to their variance.

**Experiment 3:** Consider a two-dimensional data set consisting of $N = 2100$ points, where three clusters $C_1$, $C_2$ and $C_3$ are formed. Each cluster is modelled by a normal distribution. The means of the distributions are $c_1 = [6.53, 1.39]^T$, $c_2 = [20.32, 20.39]^T$ and $c_3 = [28.09, 11.38]^T$, respectively, while their covariance matrices are set to $10 \cdot I_2$, $20 \cdot I_2$ and $1 \cdot I_2$, respectively. A number of 1000 points are generated by each one of the first two distributions and 100 points are generated by the last one. Moreover, 200 data points are added randomly as noise in the region where data live (see Fig. 7a).
TABLE III: Performance of clustering algorithms for the Experiment 3 data set.

| Algorithm      | \( m_{\text{ini}} \) | \( m_{\text{final}} \) | \( RM \)   | \( SR \)   | \( MD \)    | \( I_{\text{rev}} \) |
|----------------|-----------------------|------------------------|------------|------------|------------|---------------------|
| k-means        | 3                     | 3                      | 91.02      | 86.74      | 6.8509     | 45                  |
| k-means        | 8                     | 8                      | 73.83      | 42.22      | 2.5267     | 60                  |
| k-means        | 10                    | 10                     | 71.41      | 34.52      | 2.3544     | 48                  |
| k-means        | 15                    | 15                     | 68.35      | 27.00      | 0.8074     | 31                  |
| FCM            | 3                     | 3                      | 82.05      | 65.39      | 4.2089     | 66                  |
| FCM            | 8                     | 8                      | 71.88      | 36.91      | 2.5468     | 100                 |
| FCM            | 10                    | 10                     | 69.67      | 28.74      | 2.3466     | 100                 |
| FCM            | 15                    | 15                     | 67.18      | 21.96      | 0.8593     | 100                 |
| PCM            | 3                     | 2                      | 87.62      | 86.78      | 0.0655     | 12                  |
| PCM            | 8                     | 3                      | 75.14      | 68.35      | 0.2138     | 26                  |
| PCM            | 10                    | 3                      | 75.64      | 68.35      | 0.1918     | 23                  |
| PCM            | 15                    | 3                      | 78.64      | 70.04      | 0.1877     | 41                  |
| APCM \( (\alpha = 1) \) | 3                    | 2                      | 87.73      | 86.83      | 0.0655     | 12                  |
| APCM \( (\alpha = 1.5) \) | 8                    | 3                      | 90.83      | 90.04      | 0.2278     | 38                  |
| APCM \( (\alpha = 1) \) | 10                   | 3                      | 90.80      | 90.00      | 0.2131     | 28                  |
| APCM \( (\alpha = 1) \) | 15                   | 3                      | 90.83      | 90.04      | 0.2157     | 35                  |
| UPC \( (q = 2) \) | 3                     | 2                      | 87.69      | 86.78      | 0.1331     | 20                  |
| UPC \( (q = 3) \) | 8                     | 4                      | 90.04      | 85.96      | 0.5517     | 76                  |
| UPC \( (q = 3) \) | 10                    | 4                      | 89.92      | 85.78      | 0.5829     | 89                  |
| UPC \( (q = 3) \) | 15                    | 4                      | 89.79      | 85.61      | 0.6618     | 111                 |
| PFCM \( (K = 1, a = 1, b = 1, q = 2, n = 2) \) | 3                    | 2                      | 87.62      | 86.78      | 1.2927     | 25                  |
| PFCM \( (K = 1, a = 1, b = 1, q = 4, n = 2) \) | 8                    | 3                      | 83.11      | 84.65      | 0.5595     | 55                  |
| PFCM \( (K = 1, a = 1, b = 2, q = 3, n = 2) \) | 10                   | 3                      | 84.30      | 85.78      | 0.7517     | 119                 |
| PFCM \( (K = 1, a = 1, b = 3, q = 2.5, n = 2) \) | 15                   | 3                      | 86.74      | 87.70      | 0.8414     | 201                 |
| UPFC \( (a = 1, b = 1, q = 4, n = 2) \) | 3                    | 2                      | 87.76      | 86.83      | 0.4588     | 20                  |
| UPFC \( (a = 1, b = 3, q = 3, n = 2) \) | 8                    | 3                      | 87.39      | 85.43      | 0.7260     | 85                  |
| UPFC \( (a = 1, b = 3, q = 3, n = 2) \) | 10                   | 3                      | 87.40      | 85.43      | 0.7364     | 101                 |
| UPFC \( (a = 1, b = 1.5, q = 3, n = 2) \) | 15                   | 3                      | 87.64      | 85.91      | 0.5555     | 94                  |

Table III shows the clustering results of all algorithms, where \( m_{\text{ini}} \) and \( m_{\text{final}} \) denote the initial and the final number of clusters, respectively. Fig. 7b and Fig. 7c show the clustering result obtained using the k-means and FCM algorithms, respectively, for \( m_{\text{ini}} = 3 \). Figs. 7d, 7e, 7f, 7g, and 7h depict the performance of PCM, APCM, UPC, and UPFC, respectively, with their parameters chosen as stated in the figure caption. In addition, the circles, centered at each \( \theta_j \) and having radius \( \eta_j \) (as they have been computed after the convergence of the algorithms), are also drawn.

As it can be deduced from Fig. 7 and Table III, even when the k-means and the FCM are initialized with the (unknown in practice) true number of clusters \( (m = 3) \), they fail to unravel the underlying clustering structure, due to the noise encountered in the data set and the big difference in the variances between nearby clusters. The classical PCM also fails...
Fig. 7: (a) The data set of Experiment 3. The clustering results of Experiment 3 for (b) k-means, $m_{ini} = 3$, (c) FCM, $m_{ini} = 3$, (d) PCM, $m_{ini} = 15$, (e) APCM, $m_{ini} = 15$ and $\alpha = 1$, (f) UPC, $m_{ini} = 8$ and $q = 3$, (g) PFCM, $m_{ini} = 15$, $K = 1$, $\alpha = 1$, $\beta = 3$, $q = 2.5$ and $n = 2$, and (h) UPFC, $m_{ini} = 15$, $\alpha = 1$, $\beta = 1.5$, $q = 3$ and $n = 2$.

to detect the cluster with the smallest variance. On the other hand, the proposed APCM algorithm produces very accurate results for various initial values of $m_{ini}$, detecting with high accuracy the center of the actual clusters (see MD measure in Table III). The UPC algorithm has been exhaustively fine tuned so that the parameters $\eta_j$’s, which remain fixed during its execution and are the same for all clusters, get small enough values, in order to
identify the cluster with the smallest variance \((C_3)\). However, under these circumstances, a representative that is initially placed at the region where only noisy points exist (due to bad initialization from FCM), is trapped there and cannot be moved towards a dense region (due to the small value of its \(\eta_j\)). Thus, UPC concludes to 4 clusters when \(q = 3\), but if we set \(q = 2\), UPC will conclude to 2 clusters, identifying \(C_1\) and \(C_2\) and missing \(C_3\). The PFCM and UPFC algorithms constantly produce 3 clusters, at the cost of a computationally demanding fine tuning of the (several) parameters they involve. However, even when their parameters are fine tuned, the final estimates of \(\theta_j\)'s are not closely located to the true cluster centers (see MD measure in Table III). Finally, as it is deduced from Table III, the APCM algorithm achieves the best RM and SR results, detecting more accurately the true centers of the clusters (minimum MD), while, in addition, it requires the fewest iterations for convergence.

We move now to the last three experiments which are conducted on the basis of real world data sets.

**Experiment 4:** Let us consider the Iris data set ([21]) consisting of \(N = 150\), 4-dimensional data points that form three classes, each one having 50 points. In Iris data set, two classes are overlapped, thus one can argue whether the true number of clusters \(m\) is 2 or 3. As it is shown in Table IV, k-means and FCM work well, only if they are initialized with the true number of clusters \((m_{\text{ini}} = 3)\). The classical PCM fails to end up with \(m_{\text{final}} = 3\) clusters, independently of the initial number of clusters. On the contrary, the APCM, the UPC, the PFCM and the UPFC algorithms, after appropriate cross validation of their parameters, produce very accurate results in terms of RM, SR and MD. However, the APCM algorithm detects more accurately the centers of the true clusters (in most cases), compared to the other algorithms. It is noted again that the main drawback of the PFCM and the UPFC algorithms is the requirement for fine tuning of several parameters, which increases excessively the computational load required for detecting the appropriate combination of parameters that achieves the best clustering performance.

**Experiment 5:** Let us consider now the so-called New Thyroid three-class data set ([21]) consisting of \(N = 215\), 5-dimensional data points. The experimental results for all algorithms are shown in Table V. It can be seen that both k-means and FCM provide satisfactory results,
### TABLE IV: Performance of clustering algorithms for the Iris data set.

| Algorithm     | m\text{ini} | m\text{final} | RM   | SR   | MD    | Iter |
|---------------|-------------|----------------|------|------|-------|------|
| k-means       | 3           | 3              | 87.97| 89.33| 0.1271| 3    |
| k-means       | 10          | 10             | 76.64| 40.00| 0.7785| 4    |
| FCM           | 3           | 3              | 87.97| 89.33| 0.1287| 19   |
| FCM           | 10          | 10             | 76.16| 36.00| 0.7793| 35   |
| PCM           | 3           | 2              | 77.19| 66.67| 0.3563| 19   |
| PCM           | 10          | 2              | 77.63| 66.67| 0.3488| 28   |
| APCM (α = 3)  | 3           | 3              | 91.24| 92.67| 0.1406| 33   |
| APCM (α = 1)  | 10          | 3              | 84.15| 84.67| 0.4030| 67   |
| UPC (q = 4)   | 3           | 3              | 91.24| 92.67| 0.1438| 26   |
| UPC (q = 2.4) | 10          | 3              | 81.96| 81.33| 0.5569| 150  |
| PFCM (K = 1, a = 1, b = 10, q = 7, n = 2) | 3 | 3 | 90.55 | 92.00 | 0.1833 | 17 |
| PFCM (K = 1, a = 1, b = 1.5, q = 2, n = 2) | 10 | 3 | 84.64 | 85.33 | 0.5411 | 92 |
| UPFC (a = 1, b = 5, q = 4, n = 2) | 3 | 3 | 91.24 | 92.67 | 0.1642 | 32 |
| UPFC (a = 1, b = 1.5, q = 2.5, n = 2) | 10 | 3 | 81.96 | 81.33 | 0.5566 | 180 |

Only if they are initialized with the true number of clusters (m\text{ini} = 3), while the classical PCM exhibits degraded performance, independently of m\text{ini}. On the contrary, the APCM and UPC algorithms detect the actual number of clusters independently of m\text{ini} after appropriate fine tuning of their parameters. However, the APCM algorithm constantly produces higher RM and SR values. Finally, the PFCM and UPFC exhibit (a) inferior performance compared to APCM and UPC and (b) superior performance with respect to k-means and FCM provided that the latter are not initialized with the correct number of clusters.

### Experiment 6: In this experiment a hyperspectral image (HSI) data set is considered, which depicts a subscene of the flightline acquired by the AVIRIS sensor over Salinas Valley, California [22]. The AVIRIS sensor generates 224 bands across the spectral range from 0.2 to 2.4 µm. The number of bands is reduced to 204 by removing 20 water absorption bands. The aim in this experiment is to identify homogeneous regions in the Salinas HSI. Before we proceed, we apply principal component analysis (PCA) and we keep the first three principal components (see Fig. 8a). Thus, the dimensionality of the problem is reduced to three. Also, for reducing the required computational load, we select a spatial region of size 150x150 from the whole image. Thus, a total size of N = 22500 samples-pixels are used, stemming from 8 ground-truth classes: “Corn”, two types of “Broccoli”, four types of “Lettuce” and “Grapes”, denoted by different colors in Fig. 8b. Note that there is no available ground truth information for the dark blue pixels in Fig. 8b. It is also noted that Fig. 8 depicts the best
TABLE V: Performance of clustering algorithms for the New Thyroid data set.

| Algorithm          | m_{ini} | m_{final} | RM   | SR   | MD    | Iter |
|--------------------|---------|-----------|------|------|-------|------|
| k-means            | 3       | 3         | 79.65| 87.44| 0.8949| 3    |
| k-means            | 5       | 5         | 70.78| 63.72| 0.8548| 12   |
| k-means            | 15      | 15        | 55.01| 25.12| 0.7159| 16   |
| FCM                | 3       | 3         | 83.29| 89.77| 0.4385| 53   |
| FCM                | 5       | 5         | 60.32| 46.98| 1.0785| 55   |
| FCM                | 15      | 15        | 52.83| 21.86| 0.8816| 91   |
| PCM                | 3       | 1         | 53.05| 69.77| 0.1177| 7    |
| PCM                | 5       | 1         | 53.05| 69.77| 0.0559| 7    |
| PCM                | 15      | 1         | 53.05| 69.77| 0.0577| 8    |
| APCM (α = 8)       | 3       | 3         | 94.58| 96.74| 0.7231| 41   |
| APCM (α = 8)       | 5       | 3         | 93.82| 96.28| 0.7722| 92   |
| APCM (α = 1.2)     | 15      | 3         | 73.73| 83.72| 2.7123| 54   |
| UPC (q = 3)        | 3       | 3         | 83.85| 90.23| 0.6982| 41   |
| UPC (q = 2)        | 5       | 3         | 77.94| 86.51| 1.0739| 16   |
| UPC (q = 1)        | 15      | 3         | 67.21| 79.53| 2.7617| 34   |
| PFCM (K = 1, a = 1, b = 5, q = 8, n = 2) | 3 | 1 | 53.05 | 69.77 | 0.0507 | 15 |
| PFCM (K = 1, a = 1, b = 5, q = 8, n = 2) | 5 | 2 | 64.95 | 77.21 | 1.3855 | 41 |
| PFCM (K = 1, a = 1, b = 8, q = 2, n = 2) | 15 | 3 | 66.64 | 79.07 | 1.8381 | 28 |
| UPFC (α = 1, b = 5, q = 8, n = 2) | 3 | 2 | 68.21 | 79.53 | 0.4108 | 21 |
| UPFC (α = 1, b = 3, q = 6, n = 2) | 5 | 3 | 78.76 | 86.98 | 0.9682 | 27 |
| UPFC (α = 1, b = 0.1, q = 1.5, n = 2) | 15 | 3 | 72.85 | 83.26 | 1.5909 | 34 |

mapping obtained by each algorithm taking into account not only the “dry” performance indices but also its physical interpretation (see [23]).

As it can be deduced from Fig. 8, when k-means and FCM are initialized with m_{ini} = 8, they actually split the “Grapes” class into two clusters and they merge a part of “Corn” class with “Lettuce 3” and the rest of it with “Lettuce 1”. The PCM algorithm fails to uncover more than 5 discrete clusters, merging firstly “Lettuce 3”, “Grapes” and a part of “Corn” class and secondly the rest part of “Corn” with “Lettuce 1”. Moreover, it merges the two types of “Broccoli” into one. Both UPC and UPFC algorithms are able to detect up to 6 clusters, having the same behavior as PCM (when the latter produces 5 clusters), except that they both discriminate the “Grapes” class. PFCM algorithm, after precise fine tuning of its parameters, manages additionally to distinguish the two types of “Broccoli” classes, in comparison with UPC and UPFC. Finally, APCM is the only algorithm that manages to distinguish the “Lettuce 3” from the “Corn” class, while at the same time it does not merge any other of the existing classes.

Let us focus for a while on the “Lettuce 2” class. This class forms two closely located
Fig. 8: (a) The 1st PC component of Salinas HSI and (b) the corresponding ground truth labeling. The clustering results of Experiment 6 obtained from (c) k-means, \(m_{ini} = 8\), (d) FCM, \(m_{ini} = 8\), (e) PCM, \(m_{ini} = 35\), (f) APCM, \(m_{ini} = 15\) and \(\alpha = 2\), (g) UPC, \(m_{ini} = 35\) and \(q = 3\), (h) PFCM, \(m_{ini} = 8\), \(K = 1\), \(\alpha = 1\), \(\beta = 6\), \(q = 2\) and \(n = 2\), and (i) UPFC, \(m_{ini} = 35\), \(\alpha = 1\), \(\beta = 5\), \(q = 5\) and \(n = 2\).
TABLE VI: Performance of clustering algorithms for the Salinas HSI data set.

| Algorithm       | \(m_{\text{ini}}\) | \(m_{\text{final}}\) | \(RM\) | \(SR\)  | \(MD\)  | Iter |
|-----------------|---------------------|-----------------------|--------|--------|--------|------|
| k-means         | 8                   | 8                     | 90.94  | 70.08  | 1.23e+03 | 13   |
| k-means         | 15                  | 15                    | 91.58  | 62.97  | 0.58e+03 | 22   |
| k-means         | 35                  | 35                    | 88.85  | 33.68  | 0.40e+03 | 66   |
| FCM             | 8                   | 8                     | 91.20  | 71.43  | 1.18e+03 | 100  |
| FCM             | 15                  | 15                    | 91.60  | 61.40  | 0.56e+03 | 100  |
| FCM             | 30                  | 35                    | 88.48  | 30.40  | 0.52e+03 | 100  |
| PCM             | 8                   | 3                     | 77.90  | 40.07  | 1.52e+03 | 93   |
| PCM             | 15                  | 4                     | 82.64  | 55.44  | 0.50e+03 | 98   |
| PCM             | 35                  | 5                     | 86.65  | 64.04  | 0.38e+03 | 122  |
| APCM (\(\alpha = 3\)) | 8       | 7                     | 93.48  | 81.27  | 0.62e+03 | 48   |
| APCM (\(\alpha = 2\)) | 15     | 9                     | 93.48  | 80.13  | 0.67e+03 | 92   |
| APCM (\(\alpha = 1.5\)) | 35     | 9                     | 93.44  | 80.06  | 0.67e+03 | 92   |
| UPC (\(q = 3\))   | 8                   | 5                     | 86.91  | 64.09  | 0.63e+03 | 44   |
| UPC (\(q = 3\))   | 15                  | 6                     | 90.29  | 69.54  | 0.53e+03 | 54   |
| UPC (\(q = 3\))   | 35                  | 6                     | 90.33  | 69.67  | 0.44e+03 | 43   |
| PFCM \((K = 1, a = 1, b = 6, q = 2, n = 2)\) | 8       | 7                     | 93.96  | 82.57  | 0.50e+03 | 135  |
| PFCM \((K = 1, a = 1, b = 1, q = 3, n = 2)\) | 15      | 8                     | 92.75  | 74.65  | 0.64e+03 | 107  |
| PFCM \((K = 1, a = 1, b = 1, q = 4, n = 2)\) | 35      | 10                    | 93.17  | 78.44  | 0.43e+03 | 201  |
| UPFC (\(a = 1, b = 8, q = 4, n = 2)\) | 8       | 6                     | 90.33  | 69.70  | 0.42e+03 | 45   |
| UPFC (\(a = 1, b = 5, q = 5, n = 2)\) | 15      | 6                     | 90.34  | 69.70  | 0.98e+03 | 47   |
| UPFC (\(a = 1, b = 5, q = 5, n = 2)\) | 35      | 6                     | 90.34  | 69.82  | 0.43e+03 | 70   |

Clusters in the feature space, although this information is not reflected to the ground-truth labeling (note however that it can be deduced from the 1st PC component in Fig. 8a). It is important to note that, in contrast to APCM, none of the other algorithms succeeds in identifying each one of them. The fact that this is not reflected in the ground-truth labeling causes a misleading decrease in the SR performance of APCM. This is justified by the fact that, as shown in Table VI, the SR measure is almost the same for \(m_{\text{final}}\) equal to 7 and 9 for APCM, while this is not the case with PFCM, where the SR measure for \(m_{\text{final}} = 8\) is significantly decreased compared to the \(m_{\text{final}} = 7\) case. The latter is an indication that, in contrast to APCM, moving from 7 to 8 clusters, the PFCM meshes the physical clustering structure of the data set.

V. CONCLUSION

In this paper, commencing from the classic possibilistic c-means (PCM) algorithm proposed in [10], a novel possibilistic clustering algorithm, called Adaptive Possibilistic c-means (APCM), has been derived exhibiting several new features. The main one is that its parameters
$\eta$ are adapted as the algorithm evolves, in contrast to all the other possibilistic algorithms, where parameters $\eta_i$, once they are set, they remain fixed during the execution of the algorithm. This gives APCM more flexibility in tracking the variations in the cluster formation as the algorithm evolves. Additional significant features are related with the computation of the parameters $\eta$. Specifically, in contrast to previous possibilistic algorithms, each $\eta_j$ is computed as the mean absolute deviation of the vectors that are most compatible with the $j$th cluster ($C_j$), from their mean. The use of the Euclidean distance, instead of the squared Euclidean one, gives the ability to the algorithm to distinguish closely located to each other clusters. However, this choice necessitates the scaling of the data set to a new range of values, which is specified by a single parameter $\alpha$. Moreover, the use of the mean instead of the previous location of the corresponding representative in the computation of $\eta_j$’s gives better estimates for the latter. A significant side-effect of the adaptation of $\eta_j$’s is that APCM is now (in principle) capable to detect the true number, $m$, of clusters provided that it is initialized with an overestimate of it, $m_{ini}$. The latter releases APCM from the noose of knowing exactly in advance the true number of “physical” clusters. It is worth noting that as experiments shown, $m_{ini}$ varies inversely proportional with $\alpha$, in order the algorithm to work properly, which makes their choice not entirely arbitrary. In addition, they show that if $\alpha$ is fixed to a value around 1 and $m_{ini}$ is around 3-4 times greater than $m$, then, in several cases, the algorithm works properly. The experimental results provided show that APCM exhibits superior performance compared to several other related algorithms, in almost all the considered data sets. Extension of APCM for identifying noisy data points and outliers, based on the concept of “sparsity”, is a subject of on going investigation.

ACKNOWLEDGMENT

This research has been co-financed by the European Union (European Social Fund - ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) - Research Funding Program: ARISTEIA- HSI-MARS-1413.

REFERENCES

[1] M. R. Anderberg, "Cluster analysis for applications", Academic Press, New York, 1973.
[2] S. Theodoridis and K. Koutroumbas, “Pattern Recognition”, 4th edition, Academic Press, 2009.
[3] R. O. Duda and P. E. Hart, “Pattern Classification and Scene Analysis”, John Willey & Sons, New York, 1973.
[4] A. K. Jain and R. C. Dubes, “Algorithms for Clustering Data”, Prentice Hall, New Jersey, 1988.
[5] B. S. Everitt and S. Landau and M. Leese, “Cluster Analysis”, John Willey & Sons, London, 2001.
[6] J. A. Hartigan and M. A. Wong, “Algorithm AS 136: A k-means clustering algorithm”, Journal of the Royal Statistical Society, vol. 28, pp. 100-108, 1979.
[7] J. C. Bezdek, “A convergence theorem for the fuzzy Isodata clustering algorithms”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 2, pp. 1-8, 1980.
[8] J. C. Bezdek, “Pattern Recognition with Fuzzy Objective Function Algorithms”, Plenum, 1981.
[9] R. Krishnapuram and J. M. Keller, “A possibilistic approach to clustering”, IEEE Transactions on Fuzzy Systems, vol. 1, pp. 98-110, 1993.
[10] R. Krishnapuram and J. M. Keller, “The possibilistic c-means algorithm: insights and recommendations”, IEEE Transactions on Fuzzy Systems, vol. 4, pp. 385-393, 1996.
[11] N. R. Pal and K. Pal and J. M. Keller and J. C. Bezdek, “A possibilistic fuzzy c-means clustering algorithm”, IEEE Transactions on Fuzzy Systems, vol. 13, pp. 517-530, 2005.
[12] M. S. Yang and K. L. Wu, “Unsupervised possibilistic clustering”, Pattern Recognition, vol. 39, pp. 5-21, 2006.
[13] K. Treerattanapitak and C. Jaruskulchai, “Possibilistic exponential fuzzy clustering”, Journal of Computer Science and Technology, vol. 28, pp. 311-321, 2013.
[14] M. S. Yang and K. L. Wu and J. Yu, “Alpha-cut implemented fuzzy clustering algorithms and switching regressions”, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, vol. 38, pp. 588-603, 2008.
[15] J-S. Zhang and Y-W. Leung, “Improved possibilistic c-means clustering algorithms”, IEEE Trans. on Fuzzy Systems, vol. 12, No. 2, pp. 209-217, 2004.
[16] M. Barni and V. Cappellini and A. Mecocci, “Comments on “A possibilistic approach to clustering””, IEEE Transactions on Fuzzy Systems, vol. 4, pp. 393-396, 1996.
[17] S. D. Xenaki and K. D. Koutroumbas and A. A. Rontogiannis, “Adaptive possibilistic clustering”, IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), 2013.
[18] N. R. Pal, K. Pal, J. C. Bezdek, “A mixed c-means clustering model”, Proc. of the IEEE Int. Conf. on Fuzzy Systems, pp. 11-21, Spain, 1997.
[19] X. Wu, B. Wu, J. Sun, H. Fu, “Unsupervised possibilistic fuzzy clustering”, Journal of Information & Computational Science, vol. 5, pp. 1075-1080, 2010.
[20] H. Timm, C. Borgelt, C. Döring, R. Kruse, “An extension to possibilistic fuzzy cluster analysis”, Fuzzy Sets and Systems, vol. 147, pp. 3-16, 2004, doi: 10.1016/j.fss.2003.11.009.
[21] UCI Library database, http://archive.ics.uci.edu/ml/datasets.html
[22] http://www.ehu.es/ccwintco/index.php?title=Hyperspectral_Remote_Sensing_Scenes
[23] S. D. Xenaki and K. D. Koutroumbas and A. A. Rontogiannis and O. A. Sykioti, “A layered sparse adaptive possibilistic approach for hyperspectral image clustering”, IEEE International Geoscience and Remote Sensing Symposium (IGARSS), July 2014.