Sequential Sparse Adaptive Possibilistic Clustering

Spyridoula D. Xenaki, Konstantinos D. Koutroumbas, and Athanasios A. Rontogiannis*

IAASARS, National Observatory of Athens, GR-152 36, Penteli, Greece
{ixenaki,koutroum,tronto}@noa.gr

Abstract. Possibilistic clustering algorithms have attracted considerable attention, during the last two decades. A major issue affecting the performance of these algorithms is that they involve certain parameters that need to be estimated accurately beforehand and remain fixed during their execution. Recently, a possibilistic clustering scheme has been proposed that allows the adaptation of these parameters and imposes sparsity in the sense that it forces the data points to "belong" to only a few (or even none) clusters. The algorithm does not require prior knowledge of the exact number of clusters but, rather, only a crude overestimate of it. However, it requires the estimation of two additional parameters. In this paper, a sequential version of this scheme is proposed, which possesses all the advantages of its ancestor and in addition, it requires the (crude) estimation of just a single parameter. Simulation results are provided that show the effectiveness of the proposed algorithm.

Keywords: possibilistic clustering, parameter adaptivity, sparsity, sequential processing, k-means, fuzzy c-means.

1 Introduction

Clustering is a well-established data analysis method, where the aim is to locate the physical groups involved in the problem at hand (clusters) formed by a number of entities (usually each entity is represented by a set of measurements that constitute the corresponding *feature vector*). Various clustering philosophies have been proposed during the last five decades. Among them are the *hard clustering* philosophy, where each entity belongs exclusively to a single cluster, the *fuzzy clustering* philosophy, where each entity is allowed to be shared among more than one clusters and the *possibilistic clustering* philosophy, where what matters is the "degree of compatibility" of an entity with a given cluster.

Several clustering algorithms that follow one of these philosophies have been previously reported. The most celebrated among them are the k-means (hard case), e.g. [12], the fuzzy c-means, FCM (fuzzy case), e.g. [1], [2], and several

* 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.

A. Likas, K. Blekas, and D. Kalles (Eds.): SETN 2014, LNAI 8445, pp. 29–42, 2014.

possibilistic c-means algorithms, PCM (possibilistic case), e.g. [7], [8], [10], [16], [11]. These algorithms are suitable for recovering compact and hyperellipsoidally shaped clusters and they represent each cluster by a single vector, called *cluster representative*, which lies in the space of the feature vectors. The determination of the cluster representatives is carried out via the minimization of suitable cost functions. Also, all of them require knowledge of the number of clusters underlying in the data set (which, of course, is rarely known in practice). However, the k-means and the FCM differ from PCM algorithms in that the former two *impose* a clustering structure on the data set (that is they split the data set into the given number of clusters, independently of the fact that the data set may contain more or less physical clusters than that number), while the latter, in principle, leads the cluster representatives to regions that are "dense in data points". Thus, in this case, the scenario where two or more cluster representatives are led to the same "dense in data" region in space, may arise.

Focusing on PCM algorithms, they have attracted considerable attention in the recent years. Optimization of different cost functions gives rise to different PCMs (e.g. [7], [8]). A significant issue with these cost functions is that they involve a set of parameters (one for each cluster), usually denoted by η, which need to be accurately estimated before the algorithm starts and they are kept fixed during its execution. Poor estimation of these parameters (often) leads to poor clustering performance (especially in more demanding data sets). Usually, these parameters are estimated by utilizing the results of the FCM that needs to be executed first. However, the resulted estimates are not always accurate. For example, if FCM is not fed with the correct number of clusters, the resulting estimates for η's are expected to be poor.

Recently, in [14] a novel PCM algorithm, termed adaptive PCM (APCM), has been proposed, where the parameters η are adapted during its execution. In addition, APCM has, in principle, the ability to automatically detect the true number of clusters, provided that it is fed with an overestimated value of this number. A further extension of APCM, called sparse APCM (SAPCM), is introduced in [15], where sparsity is imposed on the *degrees of compatibilities* of the data vectors with the clusters, in the sense that each data vector is forced to be compatible with only a *few* (or even *none*) of the clusters. SAPCM inherits the characteristics of APCM and, in addition, it has the ability to locate the clusters more accurately, since points that lie "away" from a given cluster are prevented from contributing to the adjustment of its associate parameters.

Both APCM and SAPCM require (a) a certain parameter, denoted by β, that is used in the initialization of the parameters η, (b) an overestimation of the number of clusters and (c) (only for SAPCM) a certain parameter, λ, that controls sparsity. Although these algorithms exhibit, in principle, some degree of robustness in the choice of the previous parameters, parameter fine tuning is unavoidable. To deal with this issue, a sequential version of SAPCM, termed *SeqSAPCM*, is proposed here, which requires neither an overestimated value of the number of clusters, nor the definition of any parameter like β. The only paramater that needs to be defined is the one that controls the sparsity, i.e. λ.

The paper is organized as follows. In section 2, the SAPCM is described, while in section 3, the proposed SeqSAPCM algorithm is presented in detail. Section 4 contains simulation results that allow the assessment of the performance of the proposed algorithm. Finally, section 5 concludes the paper.

2 The Sparse Adaptive Possibilistic c-Means (SAPCM) Algorithm

Let

$$X = \{\boldsymbol{x}_i \in \Re^\ell, i = 1, ..., N\}$$

be a set of N, l-dimensional data vectors that are to be clustered. Let also

$$\Theta = \{\boldsymbol{\theta}_j \in \Re^\ell, j = 1, ..., m\}$$

be a set of m vectors that will be used for the representation of the clusters formed in X. In what follows, $||\cdot||$ denotes the Euclidean norm. Let $U = [u_{ij}]$ be an $N \times m$ matrix whose (i, j) element stands for the so called *degree of compatibility* of $\boldsymbol{x}_i$ to the jth cluster, denoted by C_j, and represented by the vector $\boldsymbol{\theta}_j$. Let also $\boldsymbol{u}_i^T = [u_{i1}, ..., u_{im}]$ be the vector containing the elements of the ith row of U.

According to [7] the u_{ij}'s in the classical (non-sparse) PCM algorithms should satisfy the conditions, (a) $u_{ij} \in [0, 1]$, (b) $\max_{j=1,...,m} u_{ij} > 0$ and (c) $0 < \sum_{i=1}^N u_{ij} < N$. However, in SAPCM, the last two conditions are removed, since a point may not be compatible with anyone of the clusters (removal of condition (b)). A consequence of the removal of condition (b) is that the case $\sum_{i=1}^N u_{ij} = 0$ for a cluster C_j becomes possible in the extreme scenario where the degrees of compatibility of all points with C_j are zero. Thus, condition (c) is also removed.

SAPCM results from the minimization of the following cost function

$$J(\Theta, U) = \sum_{i=1}^{N}[\sum_{j=1}^{m} u_{ij}\|\boldsymbol{x}_i - \boldsymbol{\theta}_j\|^2 + \sum_{j=1}^{m} \eta_j(u_{ij} \ln u_{ij} - u_{ij}) + \lambda\|\boldsymbol{u}_i\|_p^p] \quad (u_{ij} > 0)\ ^1 \tag{1}$$

where $\|\boldsymbol{u}_i\|_p$ is the p-norm of the vector $\boldsymbol{u}_i$ and $p \in (0,\ 1)$. The first two terms constitute the classical possibilistic cost function proposed and explained in [8], while η_j is a measure of how much the influence of a cluster is spread around its representative. The last term is the sparsity inducing term.

Minimization of $J(\Theta, U)$ with respect to $\boldsymbol{\theta}_j$ leads to the following updating equation

$$\boldsymbol{\theta}_j = \frac{\sum_{i=1}^N u_{ij}\boldsymbol{x}_i}{\sum_{i=1}^N u_{ij}} \tag{2}$$

On the other hand, taking the derivative of $J(\Theta, U)$ with respect to u_{ij}, we obtain

$$\frac{\partial J(\Theta, U)}{\partial u_{ij}} \equiv \eta_j f(u_{ij}) = \eta_j(\frac{d_{ij}}{\eta_j} + \ln u_{ij} + \frac{\lambda}{\eta_j} p u_{ij}^{p-1}), \tag{3}$$

[1] The positivity of u_{ij} is a prerequisite in order for the $\ln u_{ij}$ to be well-defined.

where $d_{ij} = \|\boldsymbol{x}_i - \boldsymbol{\theta}_j\|^2$. Obviously, $\frac{\partial J(\Theta,U)}{\partial u_{ij}} = 0$ is equivalent to $f(u_{ij}) = 0$. For the latter, the following propositions hold.

Proposition 1: $f(u_{ij})$ *may* become zero *only* for $u_{ij} \in (0, 1]$.[2]

This happens since *only* in this case the second term in the right-hand side of eq. (3) is negative (the other two are always positive).

Proposition 2: The unique minimum of $f(u_{ij})$ is $\hat{u}_{ij} = \left[\frac{\lambda}{\eta_j} p(1-p)\right]^{\frac{1}{1-p}} (\in (0, 1])$.

This results from the direct minimization of $f(u_{ij})$ with respect to u_{ij}.

Taking into account the previous two propositions and provided that there exists at least one point $u_{ij}^0 \in (0, 1]$ (e.g., $u_{ij}^0 = \exp(-d_{ij}/\eta_j)$) for which $f(u_{ij}^0) > 0$, it can be deduced that $f(u_{ij}) = 0$ has two solutions, if $f(\hat{u}_{ij}) < 0$ and one solution, if $f(\hat{u}_{ij}) = 0$. In any other case, $f(u_{ij}) = 0$ has no solutions. In any case all solutions (if they exist) lie in $(0,\ 1]$. Also, the following proposition holds.

Proposition 3: If $f(u_{ij}) = 0$ has two solutions, the largest of them is the one that minimzes $J(\Theta, U)$.

This results by proving that $f(u_{ij})$ is positive (negative) on the left (right) of the largest solution.

Based on the above propositions, we proceed to the solution of $f(u_{ij}) = 0$ as follows. First, we check whether $f(\hat{u}_{ij}) > 0$. If this is the case, then $f(u_{ij}) > 0$, for all $u_{ij} > 0$, thus J is increasing with respect to u_{ij}. Therefore, setting $u_{ij} = 0$ (i.e., *imposing sparsity*), J is minimized with respect to u_{ij}. If $f(\hat{u}_{ij}) = 0$, we set $u_{ij} = \hat{u}_{ij}$. If $f(\hat{u}_{ij}) < 0$, $f(u_{ij}) = 0$ has two solutions in $(0, 1]$. In order to determine the largest of the solutions of $f(u_{ij}) = 0$, we apply the bisection method (e.g. [3]) in the range $[\hat{u}_{ij}, 1]$, which is known to converge very rapidly to the optimum u_{ij}, that is, in our case, to the largest of the two solutions of $f(u_{ij}) = 0$.

After the above analysis, the SAPCM algorithm can be stated as follows.

The SAPCM algorithm

- $t = 0$
- **Initialization** of $\boldsymbol{\theta}_j$'s: $\boldsymbol{\theta}_j \equiv \boldsymbol{\theta}_j(0)$, $j = 1, ..., m$, using the **Max-Min** alg. ([9])
- **Initialization** of η_j's: **Set** $\eta_j = \frac{\min_{\boldsymbol{\theta}_s \neq \boldsymbol{\theta}_j} \|\boldsymbol{\theta}_j - \boldsymbol{\theta}_s\|^2/2}{-\log \beta}$, $\beta \in (0, 1)$, $j = 1, ..., m$
- **Repeat:**
  - $t = t + 1$
  - **Update** U: As described in the text
  - **Update** Θ: $\boldsymbol{\theta}_j(t) = \sum_{i=1}^{N} u_{ij}(t-1)\boldsymbol{x}_i \Big/ \sum_{i=1}^{N} u_{ij}(t-1)$, $j = 1, ..., m$
  - *Possible cluster elimination part*:
    - **Determine:** $u_{ir} = \max_{j=1,...,m} u_{ij}$, $i = 1, ..., N$
    - **If** $u_{ir} \neq 0$ **then Set** $label(i) = r$ **else Set** $label(i) = 0$ **end**
    - **Check** for $j = 1, ..., m$**: If** $j \notin label$ **then Remove** C_j **end**

---

[2] Due to space limitations the proof of this and the following propositions are omitted.

- **Adaptation** of η_j's: $\eta_j(t) = \frac{1}{n_j(t)} \sum_{\boldsymbol{x}_i : u_{ij}(t) = \max_{r=1,\ldots,m} u_{ir}(t)} \|\boldsymbol{x}_i - \boldsymbol{\mu}_j(t)\|$, with $\boldsymbol{\mu}_j(t) = \frac{1}{n_j(t)} \sum_{\boldsymbol{x}_i : u_{ij}(t) = \max_{r=1,\ldots,m} u_{ir}(t)} \boldsymbol{x}_i$, $j = 1, ..., m$

– **Until:** the change in $\boldsymbol{\theta}_j$'s between two successive iterations gets very small

Some comments on the algorithm are now in order.

– *Initialization:* In SAPCM, the initialization of $\boldsymbol{\theta}_j$'s for an overestimated number of clusters is carried out using a fast approximate variation of the Max-Min algorithm proposed in [9] (see also [14]). This is done in order to increase the probability of each $\boldsymbol{\theta}_j$ to be placed initially close to a "dense in data" region[3]. Denoting by X_{re} the set of the initial cluster representatives, η_j's are initialized as follows. First, the distance of each $\boldsymbol{\theta}_j \in X_{re}$ from its closest $\boldsymbol{\theta}_s \in X_{re} - \{\boldsymbol{\theta}_j\}$, denoted by $d_{\min}(\boldsymbol{\theta}_j)$, is determined and then η_j is set to $\eta_j = \frac{d_{\min}(\boldsymbol{\theta}_j)/2}{-\log \beta}$, where $\beta \in (0, 1)$ is an appropriately chosen parameter (see *Initialization of η_j's* part in the description of the SAPCM algorithm). As it has been verified experimentally, typical values for β that lead to good initializations are in the range $[0.1, 0.5]$. The experiments showed also that β depends on how densely the natural clusters are located; smaller values of β are more appropriate for sparsely located clusters, while larger values of β are more appropriate for more densely located clusters.
– *Adaptation:* In SAPCM, this part refers to (a) the adjustment of the number of clusters and (b) the adaptation of η_j's, which are two interrelated processes. Here, let *label* be a N-dimensional vector, whose ith component is the index of the cluster that $\boldsymbol{x}_i$ is most *compatible* with, i.e., the cluster C_j for which $u_{ij} = \max_{r=1,\ldots,m} u_{ir}$. Let also n_j denote the number of the data points $\boldsymbol{x}_i$, that are most compatible with the jth cluster and $\boldsymbol{\mu}_j$ be the mean vector of these data points. The adjustment (reduction) of the number of clusters is achieved by examining if the index j of a cluster C_j appears in the vector *label*. If this is the case, C_j is preserved. Otherwise, C_j is eliminated (see *Possible cluster elimination* part in the SAPCM algorithm). Moreover, the parameter η_j of a cluster C_j is estimated as the mean value of the distances of the most compatible to C_j data vectors from their mean vector $\boldsymbol{\mu}_j$ and *not* from the representative $\boldsymbol{\theta}_j$, as in previous works (e.g. [7], [17]) (see *Adaptation of η_j's* part in the SAPCM algorithm). It is also noted that, in the case where there are two or more clusters, that are equally compatible with a specific $\boldsymbol{x}_i$, then $\boldsymbol{x}_i$ will contribute to the determination of the parameter η of *only* one of them, which is chosen arbitrarily.
– *Sparsity:* Taking into account that $d_{ij}/\eta_j \geq 0$ and utilizing proposition 2, for $\ln \hat{u}_{ij} + \frac{\lambda}{\eta_j} p \hat{u}_{ij}^{p-1} > 0$, i.e., $\lambda > \max_{i,j}(\frac{-\ln(\hat{u}_{ij}) \hat{u}_{ij}^{1-p} \eta_j}{p})$, no point is allowed to be compatible with any one of the clusters. On the other hand, for $\lambda \simeq 0$ almost no sparsity is imposed, that is, almost all points will be compatible with all clusters up to a non-zero degree of compatibility. Therefore, λ is required

[3] In contrast, random initialization may lead several representatives to the same physical cluster, leaving other clusters without a representative.

to be chosen carefully between these two extremes. In addition, for $p \to 0$ or $p \to 1$ no sparsity is imposed, since eq. (3) has always a single solution in both cases. A requirement for sparsity to enter into the scene is to have $p \in (0, 1)$, but with p away from both 0 and 1.

3 The Sequential SAPCM (SeqSAPCM)

We proceed now with the description of the sequential SAPCM, which involves in its heart the SAPCM. Note that in the framework of SeqSAPCM, the SAPCM algorithm does not initialize by itself the parameters $\boldsymbol{\theta}$ and η. It rather takes as input the initial estimates of these parameters. To denote this explicitly we write

$$[\Theta, H] = SAPCM(X, \Theta^{ini}, H^{ini}, \lambda) \tag{4}$$

In words, the algorithm takes as input, initial estimates of the cluster representatives (included in Θ^{ini}) and their corresponding parameters η (included in H^{ini}) and returns the updated set of (a) representatives (Θ) and (b) their corresponding parameters η (H). Also, recall that λ is the parameter that controls sparsity.

Unlike SAPCM, in SeqSAPCM the number of clusters increases by one at a time, until the true number of clusters is (hopefully) reached. From this point of view, if SAPCM, as described in Section 2, can be considered as a *top-down* technique in the sense that it starts with an overestimated number of clusters and gradually reduces it, SeqSAPCM can be considered as a *bottom-up* approach in the sense that it starts with two clusters and gradually increases them up to the true number of clusters.

The algorithm works as follows. Initially, the two most distant points of X, say $\boldsymbol{x}_s$ and $\boldsymbol{x}_t$, are determined and serve as initial estimates of the first two cluster representatives, $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$, denoted by $\boldsymbol{\theta}_1^{ini}$ and $\boldsymbol{\theta}_2^{ini}$. Thus, at this time it is $m = 2$ and $\Theta^{ini} = \{\boldsymbol{\theta}_1^{ini}, \boldsymbol{\theta}_2^{ini}\}$. The initial estimation of each one of the parameters η_1 and η_2 (η_1^{ini}, η_2^{ini}) that correspond to the first two clusters, is computed as the *maximum* of the following two quantities:

- $d_{\max}$, which is the maximum among the distances between each data vector $\boldsymbol{x}_i \in X$ and its nearest neighbor $\boldsymbol{x}_i^{nei} \in X$, i.e.,

$$d_{\max} = \max_{i=1,\ldots,N} d(\boldsymbol{x}_i, \boldsymbol{x}_i^{nei})$$

- d_{slope}^j, which is determined as follows: The distances of $\boldsymbol{\theta}_j^{ini}$ from its q nearest neighbors in X [4], d_s^j, $s = 1, \ldots, q$, are computed and plotted in increasing order. The neighboring point of $\boldsymbol{\theta}_j^{ini}$ where the resulting curve exhibits the maximum slope, say the rth one, is identified and d_{slope}^j is set equal to d_r^j (the distance between $\boldsymbol{\theta}_j^{ini}$ and its rth neighbor).

[4] Typically, q is set to a value around 10.

Thus $\eta_j^{ini} = \max\{d_{\max}, d_{slope}^j\}$ and $H^{ini} = \{\eta_1^{ini}, \eta_2^{ini}\}$. Then, we run the SAPCM algorithm (4) and after its convergence, $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ are placed to the centers of dense regions, while η_1 and η_2 take values that characterize the spreads of these regions around $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$, respectively. We have now $\Theta = \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2\}$ and $H = \{\eta_1, \eta_2\}$.

We proceed next by identifying the point in X that will be used as initial estimate of the next representative as follows. For each $\boldsymbol{x}_i \in X$ we compute its distances from the points of Θ and we select the minimum one. Then, among all N minimum distances we select the maximum one and the corresponding point, say $\boldsymbol{x}_r$ is the initial estimate of the next representative ($\boldsymbol{\theta}_3$), that is $\boldsymbol{\theta}_3^{ini} \equiv \boldsymbol{x}_r$. In mathematical terms, $\boldsymbol{x}_r$ is the point that corresponds to the distance $\max_{i=1,\ldots,N}(\min_{j=1,\ldots,m} d(\boldsymbol{x}_i, \boldsymbol{\theta}_j))$. Also, η_3^{ini} is computed as the previous ones. Next, the SAPCM algorithm is employed with $H^{ini} = \{\eta_1, \eta_2, \eta_3^{ini}\}$ and $\Theta^{ini} = \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\theta}_3^{ini}\}$ and executed for three clusters now. After its convergence, all $\boldsymbol{\theta}_j$'s are found to the centers of "dense in data" regions and we have $\Theta = \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\theta}_3\}$ and $H = \{\eta_1, \eta_2, \eta_3\}$. The algorithm terminates when no new cluster is detected between two successive iterations.

The algorithm can be stated as follows:

The SeqSAPCM algorithm

- **Normalize** the data set X to the $[0, 10]^l$ space[5].
- **Set** λ to an appropriate value.
- **Determine** the two most distant points in X, say $\boldsymbol{x}_s$ and $\boldsymbol{x}_t$ and use them as initial estimates of the first two representatives $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ (i.e., $\boldsymbol{\theta}_1^{ini} \equiv \boldsymbol{x}_s$, $\boldsymbol{\theta}_2^{ini} \equiv \boldsymbol{x}_t$)[6].
- **Initialize** η_1 and η_2 (η_1^{ini}, η_2^{ini}) as described in the text.
- $[\Theta, H] = SAPCM(X, \{\boldsymbol{\theta}_1^{ini}, \boldsymbol{\theta}_2^{ini}\}, \{\eta_1^{ini}, \eta_2^{ini}\}, \lambda\}$
- **Repeat**
  - (A) Use as initial estimate of the next cluster the point $\boldsymbol{x}_r \in X$ that corresponds to the distance $\max_{i=1,\ldots,N}(\min_{j=1,\ldots,m} d(\boldsymbol{x}_i, \boldsymbol{\theta}_j))$ and **set** $\boldsymbol{\theta}_{new}^{ini} = \boldsymbol{x}_r$.
  - **Compute** the η_{new}^{ini} as described in the text
  - $[\Theta, H] = SAPCM(X, \Theta \cup \{\boldsymbol{\theta}_{new}^{ini}\}, H \cup \{\eta_{new}^{ini}\}, \lambda\}$
- **Until** no new cluster is detected

Some comments on the proposed SeqSAPCM are in order now.

- The initialization of the representatives is carried out so that to increase the probability to select a point from each one of the underlying clusters. It is

[5] This is a prerequisite that stems from the fact that u_{ij} decreases rapidly as the distance between $\boldsymbol{x}_i$ and $\boldsymbol{\theta}_j$ increases. However, it should be noted that things work also for any value around 10.

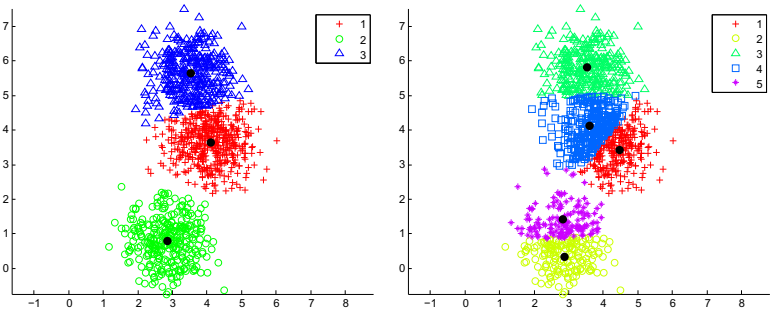
[6] In order to avoid high computational burden, this step is carried out approximately using the method described in [4].

noted that, in contrast to SAPCM where the initialization of the representatives is carried out via the **Max-Min** algorithm, in SeqSAPCM a single step of the **Max-Min** (step (A) in the SeqSAPCM algorithm) is executed each time a new representative is to be initialized.
- The initialization of the parameters η for each new cluster may seem a bit tricky. Its rationale is the following. For data sets whose points form well separated clusters, $d_{\max}$ is, in general, a good estimate for η_{new}^{ini} of each new cluster. In this case, since the initial estimates of the representatives are clusters points[7], $d_{\max}$ is a reasonable value for controlling the influence of a cluster around its representative. Note also, that in this case d_{slope}^{j} is close to $d_{\max}$. On the other hand, when there are points in the data set that lie away from the clusters (e.g. outliers), the algorithm is likely to choose some of them as initial estimates of cluster representatives. However, a small initial value of η for these representatives will make difficult their movement to dense in data regions. In this case η is set initially equal to d_{slope}^{j} which, in this case, turns out to be significantly larger than $d_{\max}$. Experiments show that $d_{\max}$ is a small value for η in this case, while d_{slope}^{j} leads to better cluster estimations.
- It is worth mentioning that previously determined η_j's (and $\boldsymbol{\theta}_j$'s) *may* be adjusted in subsequent iterations, as new clusters are formed.
- The SeqSAPCM algorithm, actually requires fine tuning only for the sparsity promoting parameter λ. On the other hand, SAPCM requires additional fine tuning for the initial number of clusters as well as for the parameter β that is used for the initialization of η's.
- A generalization of the proposed scheme may follow if, instead of adding a single representative at each time, we seek for more than one of them at each iteration. In principle, this may reduce the required computational time.

4 Experimental Results

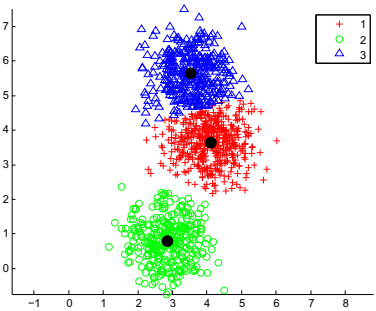
In this section, we assess the performance of the proposed method in several experimental synthetic and real data settings. More specifically, we compare the clustering performance of SeqSAPCM with that of the k-means, the FCM, the PCM, the APCM and the SAPCM algorithms[8]. To this end, we need to evaluate a clustering outcome, that is to compare it with the true data label information. This is carried out via three different measures. The first is the so-called Rand Measure (RM), described in [12], which can cope with clusterings whose number of clusters may differ from the true number of clusters. A generalization of RM is the Generalized Rand Measure (GRM) described in [6], which further takes into account the *degrees of compatibility* of all data points to clusters. Note that in k-means algorithm, the RM does not differ from GRM, since each vector belongs exclusively to a single cluster. Thus, the GRM is not considered in the k-means case. Finally, the classical Success Rate (SR) is employed, which measures the percentage of the points that have been correctly labeled by each algorithm.

[7] Ususally, they are "peripheral" points of the clusters.
[8] In order to make a fair comparison, all algorithms are initialized as SeqSAPCM.

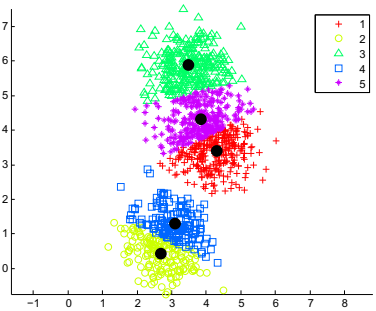


(a) k-means with $m_{ini} = 3$

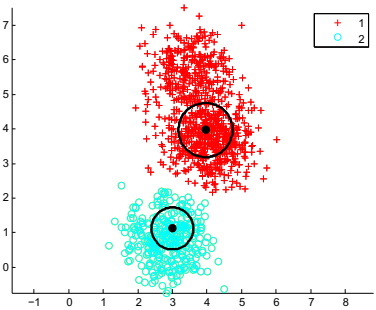
(b) k-means with $m_{ini} = 5$



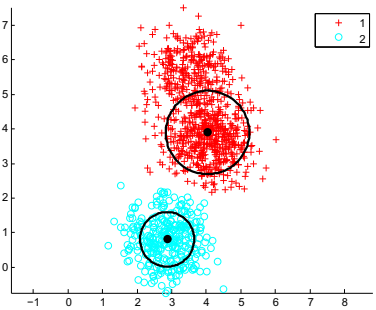
(c) FCM with $m_{ini} = 3$



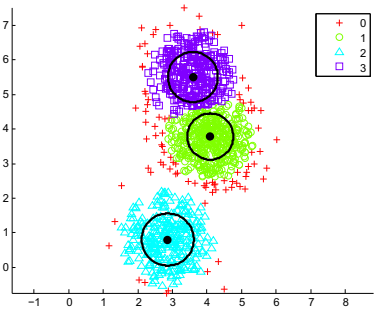
(d) FCM with $m_{ini} = 5$



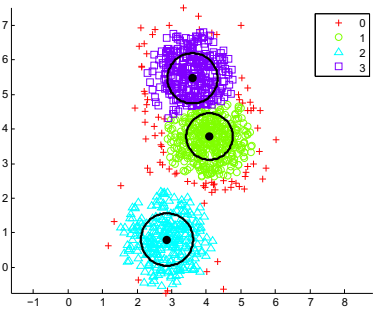
(e) PCM with $m_{ini} = 20$



(f) APCM with $m_{ini} = 5$



(g) SAPCM with $m_{ini} = 5$



(h) SeqSAPCM

Fig. 1. Clustering results for **Experiment 1**. Bolded dots represent the final cluster representatives. Note that in PCM only the truly "different" clusters are taken into account.

Table 1. The results of the **Experiment 1** synthetic data set

	λ	β	m_{ini}	m_{final}	RM(%)	GRM(%)	SR(%)
k-means	-	-	3	3	**93.62**	-	**95.36**
	-	-	5	5	80.82	-	62.00
	-	-	12	12	71.97	-	28.27
FCM	-	-	3	3	**93.62**	**79.10**	**95.36**
	-	-	5	5	81.36	67.81	63.00
	-	-	12	12	71.72	59.51	29.00
PCM	-	-	5	1	35.48	35.48	45.45
	-	-	20	2	74.32	51.67	72.09
APCM	-	0.1	3	2	74.32	73.81	72.09
	-	0.1 to 0.5	5 to 12	2	74.32	73.81	72.09
SAPCM	0.3	0.1	3	2	74.32	73.94	72.09
	0.3	0.1	5	3	**93.39**	**90.27**	**95.18**
	0.3	0.5	5	2	74.32	74.16	72.09
	0.3	0.1	12	2	74.32	74.16	72.09
SeqSAPCM	0.28	-	-	3	**93.51**	**90.03**	**95.27**

Experiment 1: Let us consider a synthetic two-dimensional data set consisting of $N = 1100$ points, where three clusters C_1, C_2, C_3 are formed. Each cluster is modelled by a normal distribution. The means of the distributions are $\boldsymbol{c}_1 = [4.1,\ 3.7]^T$, $\boldsymbol{c}_2 = [2.8,\ 0.8]^T$ and $\boldsymbol{c}_3 = [3.5,\ 5.7]^T$, respectively, while their (common) covariance matrix is set to $0.4 \cdot I_2$, where I_2 is the 2×2 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 clusters C_1 and C_3 differ significantly in their density (since both share the same covariance matrix but C_3 has significantly less points than C_1) and since they are closely located to each other, a clustering algorithm could consider them as a single cluster. Figs. 1 (a), (b) show the clustering outcome obtained using the k-means algorithm with $m_{ini} = 3$ and $m_{ini} = 5$, respectively. Similarly, in Figs. 1 (c), (d) we present the corresponding results for FCM. Fig. 1 (e) depicts the performance of PCM for $m_{ini} = 20$, while, in addition, it shows the circled regions, centered at each $\boldsymbol{\theta}_j$ and having radius equal to η_j, in which C_j has increased influence. Fig. 1 (f) shows the results of APCM with $m_{ini} = 5$ and $\beta = 0.1$ and Fig. 1 (g) shows the results of SAPCM with $m_{ini} = 5$, $\beta = 0.1$ and $\lambda = 0.3$. Finally, Fig. 1 (h) shows the results of SeqSAPCM with $\lambda = 0.28$. Moreover, Table 1 shows RM, GRM, SR for the previously mentioned algorithms, where m_{ini} and m_{final} denote the initial and the final number of clusters, respectively.

As it is deduced from Fig. 1 and Table 1, when k-means and FCM are initialized with the (rarely known in practice) true number of clusters ($m = 3$), their clustering performance is very satisfactory. However, any deviation from this value causes a significant degradation to the obtained clustering quality. On the other hand, the classical PCM fails to unravel the underying clustering structure, due to the fact that two clusters are close enough to each other and

the algorithm does not have the ability to adapt η_j's in order to cope with this situation. In this data set, the APCM algorithm also fails to detect all naturally formed clusters and unites clusters C_1 and C_3 thus leading to a two-cluster clustering result for several values of β. On the other hand, for a large enough value of λ ($\lambda = 0.3$) and a proper overestimation of the initial number of clusters (m_{ini}), SAPCM heavily imposes sparsity so that the remotely located from the mean of the C_3 cluster points are not taken into account to the estimation of the parameters of cluster C_3 ($\boldsymbol{\theta}_3$ and η_3), thus leading to smaller values for η_3. As a consequence, the unification of C_3 with its neighboring (denser) C_1 cluster is prevented. However, as it is deduced from Table 1, the parameters $(\lambda, \beta, m_{ini})$ of SAPCM have to be fine tuned, in order for SAPCM to be successful. This is not the case for SeqSAPCM, which produces very accurate results after cross validating just a single parameter (λ).

Experiment 2: Let us consider a synthetic two-dimensional data set consisting of $N = 5000$ points, where fifteen clusters are formed (data set S_2 in [5]), as shown in Fig. 2. All clusters are modelled by normal distributions with different covariance matrices. As it is deduced from Table 2, k-means fails to unravel the underlying clustering structure, even when it is initialized with the actual number of natural clusters ($m_{ini} = 15$). On the other hand, FCM works well when it is initialized with the true number of clusters ($m_{ini} = 15$), providing very satisfactory results. However, any deviation from this value causes, again, a significant degradation to the obtained clustering quality. The classical PCM fails independently of the initial number of clusters. In this data set, the APCM and the SAPCM algorithms work well after fine-tuning their parameters and properly selecting m_{ini}. Finally, by simply selecting $\lambda = 0.1$, SeqSAPCM is able to capture the underlying clustering structure very accurately.

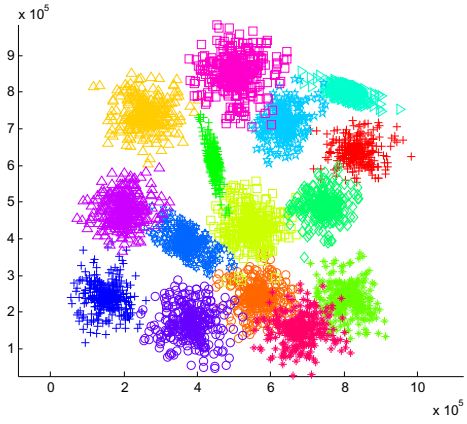


Fig. 2. The data set in **Experiment 2**. Colors indicate the true label information.

Experiment 3: Let us consider the real *Iris* data set ([13]) consisting of $N = 150$, 4-dimensional data points that form three classes, each one having 50

Table 2. The results of the **Experiment 2** synthetic data set

	λ	β	m_{ini}	m_{final}	RM(%)	GRM(%)	SR(%)
k-means	-	-	15	15	97.24	-	81.68
	-	-	20	20	**98.23**	-	**84.84**
	-	-	25	25	97.61	-	78.48
FCM	-	-	15	15	**99.23**	**80.09**	**97.00**
	-	-	20	20	98.40	75.36	87.50
	-	-	25	25	97.46	71.30	75.02
PCM	-	-	15	3	62.05	20.43	20.40
	-	-	25	6	78.67	22.02	39.22
APCM	-	0.1	10	10	93.28	90.98	67.16
	-	0.1	20	14	98.38	95.71	91.18
	-	0.1	25	15	**99.23**	**96.88**	**97.00**
SAPCM	0.1	0.1	10	10	93.28	91.07	67.22
	0.1	0.1	20	14	98.39	95.84	91.18
	0.1	0.1	25	15	**99.24**	**96.94**	**97.04**
SeqSAPCM	0.1	-	-	15	**99.23**	**96.94**	**97.02**

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 3, k-means and FCM provide satisfactory results, only if they are initialized with the true number of clusters ($m_{ini} = 3$). The classical PCM fails to end up with $m_{final} = 3$ clusters independently of the initial number of clusters. On the contrary, the APCM and the SAPCM algorithms, after appropriate cross validation of their parameters and a proper overestimated initial number of clusters, produce very accurate results. Finally, SeqSAPCM detects the actual number of clusters, providing constantly very accurate results.

Experiment 4: Let us now consider the *Wine* real data set ([13]) consisting of $N = 178$, 13-dimensional data points that stem from three classes, the first with 59 points, the second with 71 and the third one with 48 points. The results of the previously mentioned algorithms are summarized in Table 4. Again the same conclusions can be drawn as far as the clustering performance of the algorithms is concerned, with SeqSAPCM providing the best overall clustering quality results.

5 Conclusions

In this paper a novel iterative bottom-up possibilistic clustering algorithm, termed SeqSAPCM, is proposed. At each iteration, SeqSAPCM determines a single new cluster by employing the SAPCM algorithm ([15]). Being a possibilistic scheme, SeqSAPCM does not *impose* a clustering structure on the data set but, rather, *unravels* sequentially the underlying clustering structure. The proposed algorithm does not require knowledge of the number of clusters (not even a crude estimate, as SAPCM and APCM do), but only fine tuning of a single parameter λ that controls sparsity (which is data dependent). SeqSAPCM

Table 3. The results of the *Iris* (**Experiment 3**) real data set

	λ	β	m_{ini}	m_{final}	RM(%)	GRM(%)	SR(%)
k-means	-	-	2	2	77.63	-	66.67
	-	-	3	3	**83.22**	-	**83.33**
	-	-	10	10	72.84	-	36.00
FCM	-	-	2	2	77.63	69.15	66.67
	-	-	3	3	**83.68**	**71.56**	**84.00**
	-	-	10	10	75.97	59.96	37.33
PCM	-	-	2	1	32.89	32.89	33.33
	-	-	3	2	77.63	50.45	66.67
	-	-	10	2	77.63	51.99	66.67
APCM	-	0.2	2	2	77.63	77.63	66.67
	-	0.2	5	3	78.20	77.66	72.00
	-	0.3	5	3	**89.23**	**87.78**	**90.67**
	-	0.3	10	5	83.02	78.06	68.00
SAPCM	0.1	0.1	2	2	77.63	77.63	66.67
	0.1	0.1	5	4	83.57	82.70	78.67
	0.1	0.2	5	3	**88.59**	**88.69**	**90.00**
	0.1	0.2	10	4	83.57	82.72	78.67
SeqSAPCM	0.15	-	-	3	**88.59**	**88.73**	**90.00**

Table 4. The results of the *Wine* (**Experiment 4**) real data set

	λ	β	m_{ini}	m_{final}	RM(%)	GRM(%)	SR(%)
k-means	-	-	3	3	68.55	-	51.69
	-	-	5	5	69.66	-	56.74
	-	-	8	8	69.56	-	40.45
FCM	-	-	3	3	71.05	64.91	68.54
	-	-	5	5	71.68	65.37	54.49
	-	-	8	8	70.15	63.40	35.96
PCM	-	-	3	1	33.80	33.80	39.89
	-	-	15	1	33.80	33.80	39.89
APCM	-	0.2	2	2	68.33	68.42	60.11
	-	0.2	5	2	68.33	68.42	60.11
	-	0.1	5	3	**92.42**	**91.61**	**94.38**
	-	0.1	8	4	89.94	87.72	87.08
SAPCM	0.01	0.1	2	2	67.72	67.91	60.11
	0.01	0.1	5	3	67.70	65.01	60.11
	0.01	0.05	5	3	**93.18**	**92.75**	**94.94**
	0.01	0.05	8	4	89.27	87.77	86.52
SeqSAPCM	0.08	-	-	3	**93.31**	**91.25**	**94.94**

outperforms the classical k-means and FCM when the latter are not fed with the actual number of clusters. In addition, it has almost the same clustering performance with SAPCM, when the latter is equipped with the optimal values for its parameters, which are three (initial estimate of the number of representatives, the parameter β for the initialization of η's and the paramater λ that controls sparsity) against one in SeqSAPCM (λ).

The automatic selection of the sparsity inducing parameter λ and a convergence analysis of the proposed SeqSAPCM are subjects of current research.

References

1. Bezdek, J.C.: A convergence theorem for the fuzzy Isodata clustering algorithms. IEEE Transactions on Pattern Analysis and Machine Intelligence 2(1), 1–8 (1980)
2. Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum (1981)
3. Corliss, G.: Which Root Does the Bisection Algorithm Find? Siam Review 19, 325–327 (1977)
4. Egecioglu, O., Kalantari, B.: Approximating the diameter of a set of points in the Euclidean space. Information Processing Letters 32, 205–211 (1989)
5. Franti, P., Virmajoki, O.: Iterative shrinking method for clustering problems. Pattern Recognition 39(5), 761–765 (2006)
6. Hullermeier, E., Rifqi, M., Henzgen, S., Senge, R.: Comparing Fuzzy Partitions: A Generalization of the Rand Index and Related Measures. IEEE Transactions on Fuzzy Systems 20, 546–556 (2012)
7. Krishnapuram, R., Keller, J.M.: A possibilistic approach to clustering. IEEE Transactions on Fuzzy Systems 1(2), 98–110 (1993)
8. Krishnapuram, R., Keller, J.M.: The possibilistic c-means algorithm: Insights and recommendations. IEEE Transactions on Fuzzy Systems 4(3), 385–393 (1996)
9. Mirkin, B.: Clustering for Data Mining: A Data Recovery Approach. Chapman Hall. London (2005)
10. Pal, N.R., Pal, K., Keller, J.M., Bezdek, J.C.: A Possibilistic Fuzzy c-Means Clustering Algorithm. IEEE Transactions on Fuzzy Systems 13, 517–530 (2005)
11. Treerattanapitak, K., Jaruskulchai, C.: Possibilistic exponential fuzzy clustering. Journal of Computer Science 28, 311–321 (2013)
12. Theodoridis, S., Koutroumbas, K.: Pattern Recognition, 4th edn. Academic Press (2009)
13. UCI Library database, `http://archive.ics.uci.edu/ml/datasets.html`
14. Xenaki, S.D., Koutroumbas, K.D., Rontogiannis, A.A.: Adaptive possibilistic clustering. In: Proceedings of the IEEE International Symposium on Signal Processing and Information Technology, ISSPIT (2013)
15. Xenaki, S.D., Koutroumbas, K.D., Rontogiannis, A.A.: Sparse adaptive possibilistic clustering. In: Proceedings of the IEEE International Conference of Acoustic Speech and Signal Processing, ICASSP (2014)
16. Yang, M.-S., Wu, K.-L.: Unsupervised possibilistic clustering. Pattern Recognition 39, 5–21 (2006)
17. Zhang, J.-S., Leung, Y.-W.: Improved Possibilistic C-Means Clustering Algorithms. IEEE Transactions on Fuzzy Systems 12, 209–217 (2004)