A NOVEL ONLINE GENERALIZED POSSIBILISTIC CLUSTERING ALGORITHM FOR BIG DATA PROCESSING

Spyridoula D. Xenaki[1], *Konstantinos D. Koutroumbas*[1] *and Athanasios A. Rontogiannis*[1]

[1]IAASARS, National Observatory of Athens, GR-152 36, Penteli, Greece

ABSTRACT

In this paper a novel efficient online possibilistic c-means clustering algorithm, called Online Generalized Adaptive Possibilistic C-Means (O-GAPCM), is presented. The algorithm extends the abilities of the Adaptive Possibilistic C-Means (APCM) algorithm, allowing the study of cases where the data form compact and hyper-ellipsoidally shaped clusters in the feature space. In addition, the algorithm performs online processing, that is the data vectors are processed one-by-one and their impact is memorized to suitably defined parameters. It also embodies new procedures for creating new clusters and merging existing ones. Thus, O-GAPCM is able to unravel on its own the number and the actual hyper-ellipsoidal shape of the physical clusters formed by the data. Experimental results verify the effectiveness of O-GAPCM both in terms of accuracy and time efficiency.

Index Terms— possibilistic clustering, online clustering, parameter adaptivity, hyperspectral imaging

I. INTRODUCTION

Clustering aims at grouping objects to groups (clusters), so that more "similar" objects are assigned to the same cluster and less "similar" objects to different clusters, based on a suitable similarity measure. In clustering, each object is represented by a set of, say l, features, which form its associated l-dimensional *feature vector*, while the set of all these feature vectors constitute the *data set*. In the present work, we consider the case where each cluster is represented by a single vector called *cluster representative*, which lies in the same l-dimensional space with the data and (ideally) is located at the center of the cluster.

In this paper we deal with *cost function optimization based clustering algorithms*. Celebrated algorithms of this kind are the k-means (hard clustering), e.g. [1], the fuzzy c-means (FCM - fuzzy clustering), e.g. [2], [3] and the possibilistic c-means (PCM - possibilistic clustering), e.g. [4], [5]. More specifically, we focus on PCM clustering algorithms, which are iterative and at each iteration they move the representatives towards their closest regions that are *dense in data points* (dense regions), that is, to regions where significant aggregations of data points (clusters) exist.

In PCMs the cluster representatives are updated, based on the *degrees of compatibility* of the data vectors with the clusters[1]. In addition, PCM algorithms involve a set of parameters, one for each cluster, usually denoted by γ, each one being a measure of the variance of its associated cluster around its center. The accurate estimation of γ's is of crucial importance. Once γ's have been estimated, they are kept fixed during the execution of PCM. Therefore, poor estimates of γ's are likely to lead to poor clustering performance, especially in more demanding data sets (e.g. where clusters are close to each other and/or clusters with significantly different variances are encountered in the data set).

Recently, a new possibilistic algorithm has been proposed in [6], called Adaptive Possibilistic C-Means (APCM) that addresses the above issues of PCMs. Specifically, its main characteristic is that the parameters γ, after their proper initialization, are adapted during the execution of the algorithm. The contribution of this special feature of APCM is twofold. First, it increases the flexibility of the algorithm in tracking the variations in the formation of the clusters that occur from iteration to iteration. Second, it renders APCM capable to unravel the number of the physical clusters that exist, provided that APCM starts with a reasonable overestimate of it. This is achieved through a cluster elimination procedure, which eliminates clusters gradually as the algorithm evolves, making thus possible the reduction of the initially overestimated number of clusters towards the true one.

APCM, as a PCM algorithm, is based on the assumption that the data points form compact and hyper-spherically shaped clusters. However, in the case of hyper-ellipsoidally shaped clusters, PCM algorithms including APCM may fail to capture the shape of the clusters, by classifying incorrectly the points at the "tails" of the hyper-ellipsoids as shown Fig. 1. If, in addition, clusters of the above shape have almost coincident centers but different "orientations" in space, the above algorithms are very likely to fail in identifying them.

We acknowledge support of this work by the project "PROTEAS II - Advanced Space Applications for Exploring the Universe of Space and Earth" (MIS 5002515) which is implemented under the Action "Reinforcement of the Research and Innovation InfrastructurE", funded by the Operational Programme "Competitiveness, Entrepreneurship and Innovation" (NSRF 2014-2020) and co-financed by Greece and the European Union (European Regional Development Fund).

[1]The degree of compatibility is a measure of "affinity" between a data vector and a cluster.

Apart from the above issue, most clustering algorithms perform batch processing, that is, they process the whole data set at each iteration before updating their parameters. This makes them less eligible for processing large sized and high dimensional data sets, whose number increases rapidly nowadays. One way to deal with this issue is to resort to online processing, where parameter updating takes place after the consideration of each single data vector.

In this paper, we generalize the APCM [6], in order to handle the general case of hyper-ellipsoidally shaped clusters in an online manner. The proposed algorithm, named Online Generalized Adaptive Possibilistic C-Means (O-GAPCM), (a) is able to identify the centers and the actual "shapes" of hyper-ellipsoidal clusters in the feature space, (b) is able to detect the true number of physical clusters, and (c) processes the data vectors one by one memorizing their impact to suitably defined parameters. In addition, O-GAPCM has its own novel mechanisms that allow (i) the creation of new clusters and (ii) the merging of clusters, as the algorithm evolves, where necessary. Thus, O-GAPCM starts with a small number of clusters and creates or merges clusters dynamically, as data vectors are processed sequentially.

II. THE ONLINE GAPCM (O-GAPCM)

In this section, we describe in detail the proposed Online Generalized APCM (O-GAPCM) clustering algorithm. Its associated cost function is,

$$J(\Theta, U) = \sum_{i=1}^{t}\sum_{j=1}^{m}\left[u_{ij}d_{ij} + (u_{ij}\ln u_{ij} - u_{ij})\right] \tag{1}$$

where $d_{ij} = (\mathbf{x}_i - \boldsymbol{\theta}_j)^T\beta\Sigma_j^{-1}(\mathbf{x}_i - \boldsymbol{\theta}_j)$, $\mathbf{x}_i$ is the i-th data vector and $\boldsymbol{\theta}_j$ is the representative of cluster C_j. u_{ij} is the degree of compatiblilty of $\mathbf{x}_i$ with C_j, m is the number of clusters and β is a parameter that controls the size of the variance of the clusters[2]. Σ_j is the covariance matrix associated with cluster C_j and is defined taking into account only the most compatible points with C_j, as follows:

$$\Sigma_j = \frac{\sum_{\mathbf{x}_i : u_{ij} = \max\limits_{r=1,\dots,m} u_{ir}} (\mathbf{x}_i - \boldsymbol{\mu}_j)(\mathbf{x}_i - \boldsymbol{\mu}_j)^T}{n_j}, \tag{2}$$

where n_j is the number of the data points $\mathbf{x}_i$ that are most compatible with cluster C_j and

$$\boldsymbol{\mu}_j = \frac{1}{n_j}\sum\nolimits_{\mathbf{x}_i : u_{ij} = \max\limits_{r=1,\dots,m} u_{ir}} \mathbf{x}_i. \tag{3}$$

Minimizing eq. (1) with respect to u_{ij} and $\boldsymbol{\theta}_j$, we take

$$u_{ij} = \exp\left(-d_{ij}\right), \tag{4}$$

$$\boldsymbol{\theta}_j = \frac{\sum_{i=1}^{N} u_{ij}\mathbf{x}_i}{\sum_{i=1}^{N} u_{ij}}. \tag{5}$$

[2]Parameter β takes values around 1. See also the discussion for the related parameter α of APCM algorithm in [6].

In the sequel, we describe the main stages of the O-GAPCM algorithm, where the above updating parameter equations will be given in a recursive form thus complying online processing concept.

II-A. Parameter initialization

In the initialization phase of O-GAPCM, we run the batch APCM on a small sample of, say K, data points (e.g. $K = 100$), starting with an overestimated number of the physical clusters, m_{ini}. After its convergence, APCM ends up with m ($\leq m_{ini}$) clusters and the resulting $\boldsymbol{\theta}_j$'s and γ_j's[3], are used for the initialization of O-GAPCM. Specifically, Σ_j's of all initial clusters C_j's are initialized as $\Sigma_j = \gamma_j I_l$, where I_l is the $l \times l$ identity matrix. From now on, the sequential processing of the remaining data vectors is enabled.

II-B. Parameter adaptation

This part is executed upon the arrival of the data vector $\mathbf{x}_t$ at the t-th iteration and includes (a) the computation of the degree of compatibility u_{tj}, $j = 1, \dots, m(t-1)$ of $\mathbf{x}_t$ with all created up to now clusters and the adaptation of all cluster representatives $\boldsymbol{\theta}_j$'s, $j = 1, \dots, m(t-1)$, taking into consideration only the corresponding u_{tj}'s, and (b) the adaptation of the parameters Σ_r and $\boldsymbol{\mu}_r$ of the cluster C_r, with $u_{tr} = \max_{j=1,\dots,m(t-1)} u_{tj}$, i.e. of the *most compatible* cluster, C_r, to data point $\mathbf{x}_t$.

Taking into account eq. (4), we have

$$u_{tj} = \exp\left[-(\mathbf{x}_t - \boldsymbol{\theta}_j(t))^T\beta\Sigma_j^{-1}(t)(\mathbf{x}_t - \boldsymbol{\theta}_j(t))\right]. \tag{6}$$

In addition, the time recursive equation for $\boldsymbol{\theta}_j(t)$, resulting from eq. (5) is

$$\boldsymbol{\theta}_j(t) = \left(1 - \frac{u_{tj}}{U_j(t)}\right)\boldsymbol{\theta}_j(t-1) + \frac{u_{tj}}{U_j(t)}\mathbf{x}_t, \tag{7}$$

where $U_j(t) = U_j(t-1) + u_{tj},\ j = 1, \dots, m(t-1)$, is the summation of the degrees of compatibility of the data vectors processed so far with C_j. Note that *all* $\boldsymbol{\theta}_j$'s are updated, during the processing of the current data point $\mathbf{x}_t$. However, this is not the case with the parameters $\boldsymbol{\mu}_j$ and Σ_j of the clusters. Specifically, if u_{tr} is above a certain threshold, $thres$ (e.g. $thres = 1e-05$), which implies that $\mathbf{x}_t$ is not "very far" from the currently available clusters so that to create a new cluster, only the parameters $\boldsymbol{\mu}_r$ and Σ_r of the most compatible to $\mathbf{x}_t$ cluster, C_r, are updated. Defining $I_{tj} = 0$, $j \neq r$ and $I_{tr} = 1$, we can easily get from eqs. (2), (3), the following time-recursive formulas:

$$\Sigma_j(t) = \left(1 - \frac{I_{tj}}{S_j(t)}\right)\Sigma_j(t-1) + \frac{I_{tj}}{S_j(t)}(\mathbf{x}_t - \boldsymbol{\mu}_j(t))(\mathbf{x}_t - \boldsymbol{\mu}_j(t))^T, \tag{8}$$

$$\boldsymbol{\mu}_j(t) = \left(1 - \frac{I_{tj}}{S_j(t)}\right)\boldsymbol{\mu}_j(t-1) + \frac{I_{tj}}{S_j(t)}\mathbf{x}_t, \tag{9}$$

[3]Since APCM returns hyper-spherically shaped clusters, a certain γ_j accounts for the variance of the points of C_j around $\boldsymbol{\theta}_j$.

where $S_j(t) = S_j(t-1) + I_{tj}$, is the number of most compatible to C_j data vectors processed so far.

II-C. Cluster creation

Let us now comment on the mechanism that creates new clusters, as the algorithm evolves. This procedure is activated when u_{tr} is less than $thres$. In this case, a new cluster is created, containing only $\mathbf{x}_t$. Its corresponding parameters $\boldsymbol{\theta}$, $\boldsymbol{\mu}$ are set equal to $\mathbf{x}_t$, its parameters U, S are set to 1, while its parameter Σ is set equal to Σ_q, where $q = \arg\min_{j=1,\dots,m(t)} \prod_{k=1}^{l} \sqrt{r_{kj}}$, where r_{kj} is the k-th eigenvalue of Σ_j/β. That is, Σ is set equal to Σ_q, corresponding to the smallest sized cluster C_q. Finally, the number of clusters is increased by one.

II-D. Cluster merging

In online clustering schemes, the clustering result is dependent on the order in which the data are processed. As a consequence, several clusters might be created, which nevertheless represent parts of the same physical cluster. Therefore, a mechanism that identifies and merges such clusters should be incorporated in an online clustering algorithm. To this end, every T iterations (e.g. $T = 100$), O-GAPCM considers all pairs of clusters and checks whether a cluster C_s lies entirely inside cluster C_k. If this is the case, these two clusters are merged into one cluster and the parameters of the new cluster are defined as follows:

$$\boldsymbol{\theta}_{new} = \frac{U_s}{U_s + U_k}\boldsymbol{\theta}_s + \frac{U_k}{U_s + U_k}\boldsymbol{\theta}_k \tag{10}$$

$$\boldsymbol{\mu}_{new} = \frac{S_s}{S_s + S_k}\boldsymbol{\mu}_s + \frac{S_k}{S_s + S_k}\boldsymbol{\mu}_k \tag{11}$$

$$\Sigma_{new} = \frac{S_s}{S_s + S_k}\Sigma_s + \frac{S_k}{S_s + S_k}\Sigma_k \tag{12}$$

$$U_{new} = U_s + U_k \quad (13) \qquad S_{new} = S_s + S_k \quad (14)$$

In order to check whether a C_s lies inside C_k, we act as follows. First we check if $\boldsymbol{\theta}_s$, lies inside the hyper-ellipsoid of cluster C_k, i.e. $(\boldsymbol{\theta}_s - \boldsymbol{\theta}_k)^T \beta \Sigma_k^{-1} (\boldsymbol{\theta}_s - \boldsymbol{\theta}_k) \leq c$, where $c = (\chi_l^2)_q$ results from the Chi-Square distribution with l-degrees of freedom and defines the boundary of the hyper-ellipsoid centered at $\boldsymbol{\theta}_k$ that contains the most compatible to C_k data points, with confidence q. In our experiments, we chose $q \geq 0.95$. If this is the case, we find the $2l$ extreme data points[4], say $\mathbf{p}_i$, $i = 1, ..., 2l$ of hyper-ellipsoid C_s and we check if all of them lie inside the hyper-ellipsoid of C_k, i.e., $\max_{i=1,\dots,2l} \left[(\mathbf{p}_i - \boldsymbol{\theta}_k)^T \beta \Sigma_k^{-1} (\mathbf{p}_i - \boldsymbol{\theta}_k)\right] \leq c$. If the latter holds, clusters C_s and C_k are merged into one as described above. The O-GAPCM in algorithmic form is given below.

Algorithm 1 [Θ, Σ, $label$[5]] = O-GAPCM(X, m_{ini}, β)

```
Input: $X$, $m_{ini}$, $\beta$
 1: $t = 0$
▷ Initialization
 2: $K = 100$, $thres = 1e-05$ and $T = 100$
 3: $[\Theta(t), \Gamma(t), m(t), label]$=APCM($X(:, 1:K), m_{ini}, \alpha$[6])
 4: Set: $\Sigma_j(t) = \gamma_j(t) I_l$, $j = 1, ..., m(t)$
 5: Set: $U_j(t) = 1$ and $S_j(t) = 1$, $j = 1, \dots, m(t)$
 6: while $t + K <$ DataSize do
 7:     $t = t + 1$
 8:     $\mathbf{x}_t = X(:, t+K)$
 9:     $u_{tj} = \exp[-(\mathbf{x}_t - \boldsymbol{\theta}_j(t-1))^T \beta \Sigma_j^{-1}(t-1)(\mathbf{x}_t - \boldsymbol{\theta}_j(t-1))]$, $j = 1, \dots, m(t-1)$
▷ Update cluster representatives
10:     $U_j(t) = U_j(t-1) + u_{tj}$, $j = 1, \dots, m(t-1)$
11:     $\boldsymbol{\theta}_j(t) = \left(1 - \frac{u_{tj}}{U_j(t)}\right)\boldsymbol{\theta}_j(t-1) + \frac{u_{tj}}{U_j(t)}\mathbf{x}_t$, $\forall j$
12:     Determine: $u_{tr} = \max_{j=1,\dots,m(t-1)} u_{tj}$
13:     if $u_{tr} < thres$ then
▷ Create a new cluster
14:         $m(t) = m(t-1) + 1$
15:         $\boldsymbol{\theta}_{m(t)}(t) = \mathbf{x}_t$
16:         $\boldsymbol{\mu}_{m(t)}(t) = \mathbf{x}_t$
17:         Determine: $q = \arg\min_{j=1,\dots,m(t)} \prod_{k=1}^{l} \sqrt{r_{kj}}$
18:         $\Sigma_{m(t)}(t) = \Sigma_q$
19:         $U_{m(t)}(t) = 1$ and $S_{m(t)}(t) = 1$
20:         $label(t+K) = m(t)$
21:         Set: $I_{tj} = 0$, $j = 1, \dots, m(t)$
22:     else
▷ Update parameters of cluster $C_r$
23:         $m(t) = m(t-1)$
24:         $label(t+K) = r$
25:         Set: $I_{tj} = 0, \forall j \neq r$ and $I_{tr} = 1$
26:     end if
27:     $S_j(t) = S_j(t-1) + I_{tj}$, $j = 1, \dots, m(t)$
28:     $\boldsymbol{\mu}_j(t) = \left(1 - \frac{I_{tj}}{S_j(t)}\right)\boldsymbol{\mu}_j(t-1) + \frac{I_{tj}}{S_j(t)}\mathbf{x}_t$, $\forall j$
29:     $\Sigma_j(t) = \left(1 - \frac{I_{tj}}{S_j(t)}\right)\Sigma_j(t-1) + \frac{I_{tj}}{S_j(t)}(\mathbf{x}_t - \boldsymbol{\mu}_j(t))(\mathbf{x}_t - \boldsymbol{\mu}_j(t))^T$, $\forall j$
▷ Every $T$ iterations perform the clusters merging procedure
30: end while
31: return $\Theta$, $\Sigma = \{\Sigma_1, \dots, \Sigma_m\}$, $label$
```

III. EXPERIMENTAL RESULTS

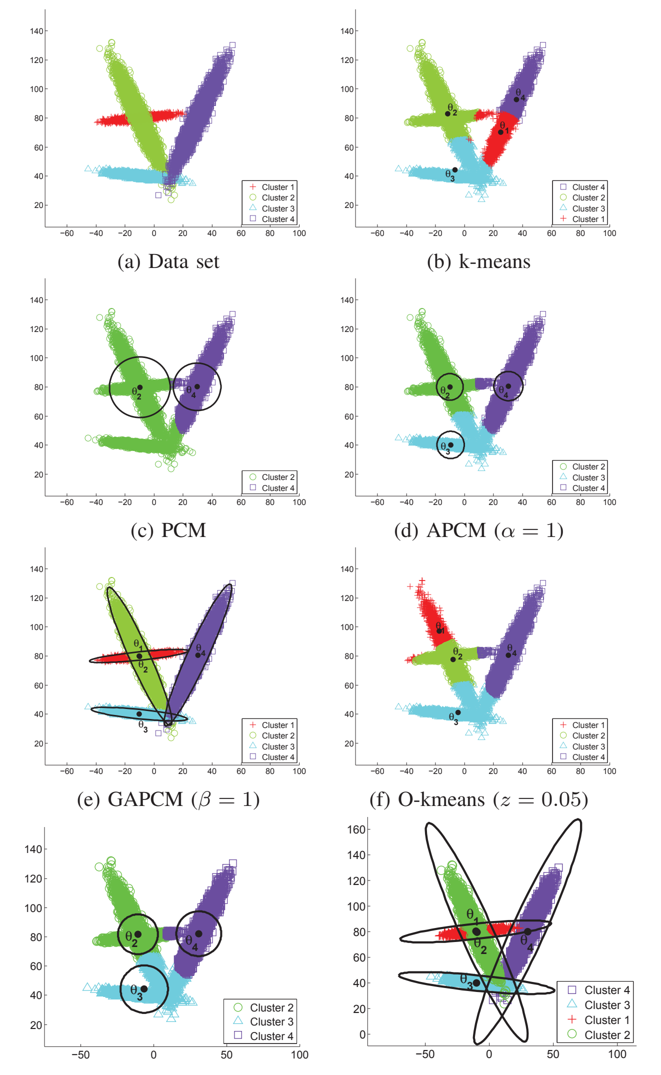
In this section we assess the performance of O-GAPCM by comparing it with (a) the batch clustering algorithms k-

[4]These are the points of the hyper-ellipsoid $(\mathbf{x}-\boldsymbol{\theta}s)^T \beta \Sigma_s^{-1} (\mathbf{x}-\boldsymbol{\theta}s) = c$ lying on its principal directions (2 points per direction).

[5]The vector $label$ contains the indexes of the cluster that data vectors are assigned.

[6]The APCM is executed for a large value of its parameter α, e.g., $\alpha = 10$, which leads to smaller values for γ_j's. Such values are preferred for the initialization phase of O-GAPCM, in order to avoid clusters having "large" γ_j's, which may extend over more than one physical clusters (see also [6]).

means [1], PCM [5], APCM [6] and the batch version of O-GAPCM, called GAPCM algorithm [7], and (b) the online clustering algorithms, online k-means (O-kmeans) [8] and online APCM (O-APCM) [9], on the basis of both simulated and real cases. In order to evaluate the clustering results, we use the Success Rate (SR) criterion, which measures the percentage of the points that have been correctly labeled by each algorithm. Finally, the times (in seconds) required for the convergence of each algorithm, are also provided[7].



(a) Data set (b) k-means (c) PCM (d) APCM ($\alpha = 1$) (e) GAPCM ($\beta = 1$) (f) O-kmeans ($z = 0.05$) (g) O-APCM ($\alpha = 0.25$) (h) O-GAPCM ($\beta = 0.3, q = 0.995$)

Fig. 1. Clustering results on Experiment 1.

Experiment 1: Consider a two-dimensional data set consisting of $N = 14000$ points, which form four physical clusters C_1, C_2, C_3 and C_4 (Fig. 1a). Each cluster is modelled by a normal distribution. The means of the distributions are $\mathbf{c}_1 = \mathbf{c}_2 = [-10, 80]^T$, $\mathbf{c}_3 = [-10, 40]^T$ and $\mathbf{c}_4 = [30, 80]^T$ respectively, while their covariance matrices are $\Sigma_1 = \begin{bmatrix} 100 & 10 \\ 10 & 2 \end{bmatrix}$, $\Sigma_2 = \begin{bmatrix} 50 & -100 \\ -100 & 225 \end{bmatrix}$, $\Sigma_3 = \begin{bmatrix} 100 & -10 \\ -10 & 2 \end{bmatrix}$, and $\Sigma_4 = \begin{bmatrix} 50 & 100 \\ 100 & 225 \end{bmatrix}$, respectively. A number of 2000 points are generated by the first and the third distributions and 5000 points are generated by each of the other two distributions. Note that the centers of C_1 and C_2 coincide and the "tails" of C_2, C_3 and C_4 are partially overlapped. In Table I and Fig. 1, the best clustering results obtained by each algorithm are shown, with its parameters chosen as stated in the figure caption.

[7]All the experiments have been conducted on a workstation of Intel i7-4790 with 16 GB RAM and 3.60 GHz.

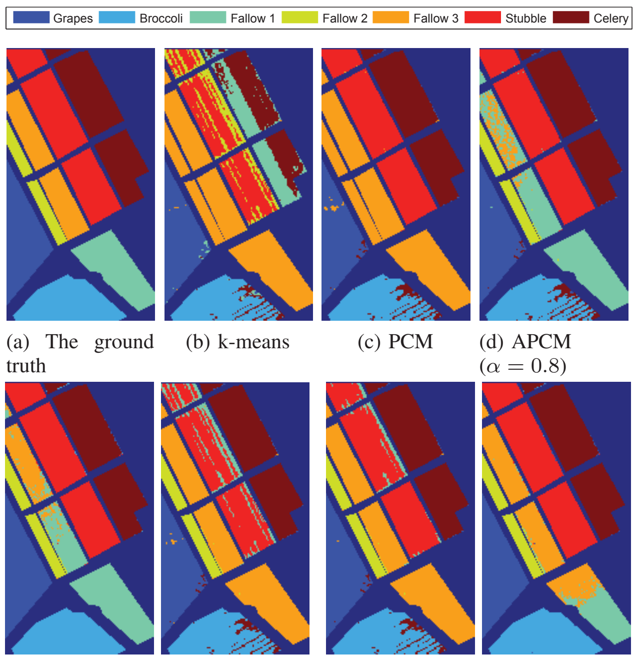
Table I. Clustering algorithms performance on Experiment 1

	m_{ini}	m_{final}	SR (%)	$Time$ (s)
k-means	4	4	62.86	0.32
PCM	10	2	70.06	3.57
APCM	10	3	81.16	1.51
GAPCM	10	**4**	**96.17**	**88.89**
O-kmeans	4	4	69.70	0.23
O-APCM	10	3	77.84	0.76
O-GAPCM	10	**4**	**95.99**	**1.23**

As it can be deduced, even when the k-means is initialized with the actual number of clusters ($m = 4$), it fails to distinguish cluster C_1 from C_2, due to their coincident centers, thus splitting physical cluster C_4 to two clusters. The classical PCM identifies only two actual centers, merging C_1, C_2 and C_3 in one cluster. The APCM algorithm identifies the centers of the three out of the four clusters, failing in distinguishing the clusters with the coincident centers (C_1 and C_2). In addition, it fails to unravel the real "shape" of the identified clusters. On the other hand, batch GAPCM produces very accurate results, identifying correctly all physical clusters, however, it is the slowest algorithm among all. O-kmeans is a little bit faster than k-means, however, it cannot distinguish cluster C_1 and it splits the physical cluster C_2 into two clusters. Moreover, O-APCM is also slightly faster than APCM, but it still cannot identify cluster C_1 and also fails to unravel the real "shape" of the identified clusters. Finally, O-GAPCM is much more computationally efficient than its batch version, GAPCM, without any remarkable degradation on its performance.

Experiment 2: We test now the proposed O-GAPCM on a real hyperspectral image (HSI), which depicts a subscene of size 220×120 of the flightline acquired by the AVIRIS sensor over Salinas Valley, California [10]. The AVIRIS sensor generates 224 spectral bands across the range from 0.2 to 2.4 μm. The number of bands is reduced to 204 by removing 20 water absorption bands. The aim here is to identify spectrally homogeneous regions in the Salinas HSI. A total size of $N = 26400$ samples-pixels are used, stemming from 7 ground-truth classes: "Grapes", "Broccoli", three types of "Fallow", "Stubble" and "Celery", denoted by different colors in Fig. 2a. Note that there is no available ground truth information for the dark blue pixels in Fig. 2a.

In the sequel, we apply the principal component analysis (PCA) and we keep the first 3 significant components; thus, in our experiment $l = 3$. Note also that Fig. 2 depicts the best mapping obtained by each algorithm (results are shown in Table II).



(a) The ground truth (b) k-means (c) PCM (d) APCM ($\alpha = 0.8$)

(e) GAPCM ($\beta = 1.5$) (f) O-kmeans ($z = 0.05$) (g) O-APCM ($\alpha = 0.2$) (h) O-GAPCM ($\beta = 0.2, q = 0.995$)

Fig. 2. Clustering results on Salinas HSI.

Table II. Clustering algorithms performance on Salinas HSI.

	m_{ini}	m_{final}	SR (%)	$Time$ (s)
k-means	7	7	65.69	1.02
PCM	15	5	81.32	6.06
APCM	15	**7**	88.07	5.10
GAPCM	15	**7**	**93.24**	**630.10**
O-kmeans	7	7	81.43	0.28
O-APCM	15	7	85.55	0.96
O-GAPCM	15	**7**	**94.49**	**2.93**

As it is depicted in Fig. 2, even when k-means is initialized with the actual number of clusters ($m_{ini} = 7$), it splits each of the classes "Stubble" and "Celery" into two clusters and merges the "Fallow 1", "Fallow 2" and "Fallow 3" classes. The PCM algorithm fails to uncover more than 5 discrete clusters, merging the three different types of the "Fallow" class. APCM and GAPCM manage to distinguish "Fallow 1" from "Fallow 3" class, identifying all underlying clusters. However, GAPCM achieves significantly higher SR than APCM, which implies that it unravels much better the actual "shape" of the clusters in the feature space. This is also reflected in Figs. 2d, 2e, where the APCM clustering result exhibits more "pixeling" in certain types of land cover, compared to that of GAPCM, where the result is "smoother". On the other hand, GAPCM is more computationally demanding algorithm than APCM. As far as the online algorithms are concerned, both O-kmeans and O-APCM fail in distinguishing "Fallow 1" from "Fallow 3" class. Finally, O-GAPCM manages to detect all clusters, achieving the highest SR amongst all. Furthermore, compared to the GAPCM, O-GAPCM is significantly more computationally efficient.

IV. CONCLUSION

In this paper an online generalized possibilistic c-means clustering algorithm, called Online Generalized Adaptive Possibilistic C-Means (O-GAPCM), is proposed. Based on APCM and extending its abilities, O-GAPCM is also able to unravel hyper-ellipsoidally shaped clusters performing online processing, which makes it very computational efficient. Experimental results show that O-GAPCM exhibits the best compromise between accuracy and time efficiency, compared to other related algorithms. The application of O-GAPCM to non-stationary environments is a subject of current research.

REFERENCES

[1] 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.

[2] J. C. Bezdek, "A convergence theorem for the fuzzy isodata clustering algorithms," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, pp. 1–8, 1980.

[3] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum, 1981.

[4] R. Krishnapuram and J. M. Keller, "A possibilistic approach to clustering," *IEEE Transactions on Fuzzy Systems*, vol. 1, pp. 98–110, 1993.

[5] R. Krishnapuram and J. M. Keller, "The possibilistic c-means algorithm: Insights and recommendations," *IEEE Trans. on Fuzzy Systems*, pp. 385–393, 1996.

[6] S. D. Xenaki, K. D. Koutroumbas, and A. A. Rontogiannis, "A novel adaptive possibilistic clustering algorithm," *IEEE Transactions on Fuzzy Systems*, vol. 24, no. 4, pp. 791–810, 2016.

[7] S. D. Xenaki, K. D. Koutroumbas, and A. A. Rontogiannis, "Generalized adaptive possibilistic c-means clustering algorithm," *to be presented at 10th Hellenic Conference on Artificial Intelligence (SETN)*, Jul 2018.

[8] W. Barbakh and C. Fyfe, "Online clustering algorithms," *International Journal of Neural Systems (IJNS)*, vol. 18, pp. 185–194, 2008.

[9] S. D. Xenaki, K. D. Koutroumbas, and A. A. Rontogiannis, "Hyperspectral image clustering using a novel efficient online possibilistic algorithm," *24th European Signal Processing Conference (EUSIPCO)*, Aug 2016.

[10] ," http://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes.