Generalized Adaptive Possibilistic C-Means Clustering Algorithm

Spyridoula Xenaki
Institute for Astronomy, Astrophysics, Space Applications and Remote Sensing, National Observatory of Athens
Penteli, Athens, Greece
ixenaki@noa.gr

Konstantinos Koutroumbas
Institute for Astronomy, Astrophysics, Space Applications and Remote Sensing, National Observatory of Athens
Penteli, Athens, Greece
koutroum@noa.gr

Athanasios Rontogiannis
Institute for Astronomy, Astrophysics, Space Applications and Remote Sensing, National Observatory of Athens
Penteli, Athens, Greece
tronto@noa.gr

ABSTRACT

In this paper a generalized possibilistic c-means clustering algorithm, called Generalized Adaptive Possibilistic C-Means (GAPCM), is presented. The algorithm extents the abilities of its ancestor, Adaptive Possibilistic C-Means (APCM), allowing the study of cases where the data form compact and hyper-ellipsoidally shaped clusters, whose points may lie around certain subspaces in the feature space. In addition, these clusters may be located very close, or even intersect each other. More specifically, a proper definition and an adaptivity concept of the parameters that GAPCM involves, during its execution, renders the algorithm able to unravel on its own the actual hyper-ellipsoidal shape of the clusters formed by the data. The performance of the algorithm is assessed through its comparison with other related algorithms on the basis of both simulated and real data sets.

CCS CONCEPTS

• **Information systems → Clustering**; • **Theory of computation → Unsupervised learning and clustering**; • **Computing methodologies → Cluster analysis**;

KEYWORDS

possibilistic clustering, hyper-ellipsoidally shaped clusters, adaptivity

ACM Reference Format:
Spyridoula Xenaki, Konstantinos Koutroumbas, and Athanasios Rontogiannis. 2018. Generalized Adaptive Possibilistic C-Means Clustering Algorithm. In *SETN '18: 10th Hellenic Conference on Artificial Intelligence, July 9–15, 2018, Rio Patras, Greece.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3200947.3201012

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).

SETN '18, July 9–15, 2018, Rio Patras, Greece

ACM ISBN 978-1-4503-6433-1/18/07…$15.00
https://doi.org/10.1145/3200947.3201012

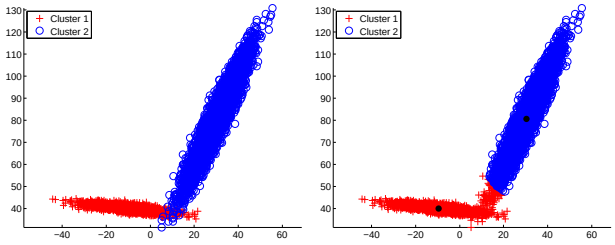
1 INTRODUCTION

Clustering is a significant branch of the general area of machine learning, where given a set of objects, the aim is to group more "similar" objects to the same group (cluster) and less "similar" objects to different groups, based on a suitable similarity measure. To achieve this, each object is represented by a set of, say l, features, which form its associated *feature vector*, while the set of all these feature vectors is called *data set*. In the present work we deal with parametric clustering, where each cluster is represented by a 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.

According to the way the data vectors are associated with the clusters, various types of clustering arise, the most significant of which are: (a) *hard clustering*, where each data vector belongs exclusively to a single cluster, (b) *fuzzy clustering*, where each data vector is shared among all available clusters and (c) *possibilistic clustering*, where each data vector is associated with a certain cluster independently of how it is associated with the remaining ones.

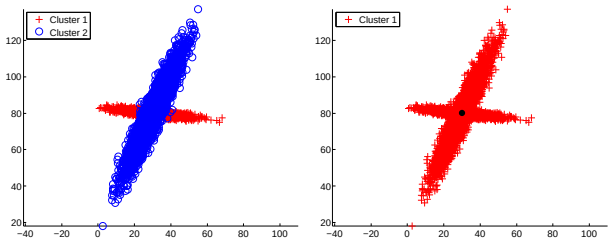
A vast amount of clustering algorithms that have been developed during the last decades, belongs to the so called *cost function optimization based algorithms* category. Celebrated algorithms of this category are the k-means (hard clustering), e.g. [5], the fuzzy c-means (FCM - fuzzy clustering), e.g. [2], [3] and the possibilistic c-means (PCM - possibilistic clustering), e.g. [6], [7]. In the present paper we focus on PCM clustering algorithms. These 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, via the minimization of suitably defined cost functions.

In PCMs the cluster representatives are updated, based on the *degrees of compatibility* of the data vectors with the clusters. In contrast to FCM and k-means, in PCM algorithms, the degrees of compatibility of a certain data vector with the various clusters are mutually independent. A direct consequence of this fact is that even if the number of clusters is overestimated, in principle, all representatives will be driven to dense regions (which is not the case for FCM or k-means), making thus feasible the uncovering of the actual clusters. However, in this case, the scenario where two or more cluster representatives are led to the same dense in data region, may arise [14], [1]. In addition, PCM deals well with noisy



(a) Data set (b) PCM clustering result

Figure 1: An example data set where PCM algorithm is not able to correctly classify the data points that lie towards the "tails" of the hyper-ellipsoids. Cluster representatives are denoted by black dots.



(a) Data set (b) PCM clustering result

Figure 2: An example data set where PCM algorithm is not able to distinguish hyper-ellipsoidally shaped clusters whose centers coincide. Cluster representatives are denoted by black dots.

data points and outliers, compared to k-means [5] and FCM [2], [3]. However, it involves a set of additional parameters, one for each cluster, usually denoted by γ, whose accurate estimation is of crucial importance. Once γ's have been estimated, they are kept fixed during the execution of the PCM algorithm. It is therefore clear that poor estimates for them 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, two new possibilistic algorithms have been proposed that overcome the above issues of PCMs [12], [13]. Specifically, in [12], the Adaptive Possibilistic C-Means (APCM) algorithm is presented, whose 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 physical clusters that exist, provided that APCM starts with a reasonable overestimate of their number. This is carried out through a cluster elimination procedure, which eliminates clusters gradually as the algorirthm evolves, making thus possible the reduction of the initially estimated number of clusters. In [13], the proposed Sparse Possibilistic C-Means (SPCM) algorithm exploits sparsity in the clustering framework. Sparsity is related to the fact that a data point may be compatible with one or only a few (or even none) clusters. SPCM exhibits increased immunity to noisy data points and outliers, by excluding them from contributing to the estimation of the cluster representatives. Thus, SPCM concludes to more accurate estimates for the cluster representatives, especially in noisy environments. Note that the fusion of the sparsity of SPCM and the adaptivity of γ's of APCM, gave rise to the Sparse Adaptive Possibilistic C-Means (SAPCM) [13] that inherits the nice features from both its ancestors.

All hitherto mentioned algorithms assume the case where the data points are spread isotropically around certain central points, that is, they form compact and hyper-spherically shaped clusters. However, in the case of compact hyperellipsoidally shaped clusters, although the above algorithms are able to move the cluster representatives towards the center of dense in data regions, they may fail to capture the "shape" of the clusters, by classifying incorrectly the points at the "tails" of the hyper-ellipsoids (see Fig. 1). If, in addition, clusters of the above shape have coincident centers but different "orientations" in space (see Fig. 2), the above algorithms are very likely to fail in identifying them.

In the sequel, we broaden the focus to the general case where the points may not be spread isotropically around certain points, that is, they may form compact and hyper-ellipsoidally shaped clusters. This possibility has been studied in [10], where Minimum Volume Ellipsoids (MVE) clustering method allocates data points into clusters in a way that minimizes the geometric mean of the volumes of each cluster covering ellipsoids. Also, the issue of hyper-ellipsoidally shaped clusters is treated by the Gaussian Mixture Model (GMM), utilizing the expectation-minimization (EM) algorithm [11].

In this paper, we generalize the classical PCM algorithm ([7]) to handle the general case of hyper-ellipsoidally shaped clusters. In addition, motivated by [12], we extend the parameter adaptivity concept during the execution of the proposed algorithm, in order the algorithm to be able to unravel on its own the actual hyper-ellipsoidal shape of the clusters.

The paper is organized as follows. In Section 2, a brief overview of the PCM ([7]) and APCM [12]) is given, while in Section 3, the proposed Generalized Adaptive Possibilistic C-Means (GAPCM) algorithm is derived. Section 4 contains experimental results conducted on both artificially generated and real data sets (blind source separation application) that validate the algorithm. Finally, Section 5 concludes the paper.

2 BRIEF REVIEW OF PCM AND APCM

Let $X = \{\mathbf{x}_i \in \mathfrak{R}^l, i = 1, \ldots, N\}$ be a set of N l-dimensional data vectors to be clustered and $\Theta = \{\boldsymbol{\theta}_j \in \mathfrak{R}^l, j = 1, \ldots, m\}$ be a set of m l-dimensional vectors that will be used as *representatives* of the clusters formed by the points in X. Moreover, let $U = [u_{ij}], i = 1, \ldots, N, j = 1, \ldots, m$ be an $N \times m$ matrix whose (i, j) element stands for the so called *degree of compatibility* of $\mathbf{x}_i$ with the jth cluster, denoted by C_j and represented by the vector $\boldsymbol{\theta}_j$. Finally, let $\mathbf{u}_i^T = [u_{i1}, \ldots, u_{im}]$ be the ith row of U that contains the degrees of compatibility of $\mathbf{x}_i$ for all the m clusters. In what follows, Euclidean norms are denoted by $\|\cdot\|$.

According to [7], the u_{ij}'s should satisfy the conditions,

$$\text{(C1)}\ \ u_{ij} \in [0,1],\ \ i = 1,\ldots,N,\ \ j = 1,\ldots,m,$$

$$\text{(C2)}\ \ \max_{j=1,\ldots,m} u_{ij} > 0,\ \ i = 1,\ldots,N,\ \ \text{and}$$

$$\text{(C3)}\ \ 0 < \sum_{i=1}^{N} u_{ij} \leq N,\ \ j = 1,\ldots,m.$$

2.1 Possibilistic C-Means - PCM

The updating equations for u_{ij}'s and $\boldsymbol{\theta}_j$'s in PCM result from the minimization of the following cost function[1]

$$J_{PCM}(\Theta, U) = \sum_{j=1}^{m}\left[\sum_{i=1}^{N} u_{ij}\|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2 + \gamma_j \sum_{i=1}^{N}(u_{ij}\ln u_{ij} - u_{ij})\right], \tag{1}$$

where the parameter γ_j is associated with the cluster $C_j, j = 1, ..., m$. Minimizing with respect to u_{ij} and $\boldsymbol{\theta}_j$, $i = 1,\ldots,N$, $j = 1,\ldots,m$ we end up with

$$u_{ij} = \exp\left(-\frac{\|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2}{\gamma_j}\right) \quad (2) \qquad \boldsymbol{\theta}_j = \frac{\sum_{i=1}^{N} u_{ij}\mathbf{x}_i}{\sum_{i=1}^{N} u_{ij}} \quad (3)$$

The parameters $\gamma_j, j = 1,\ldots,m$ are a priori estimated and kept fixed during the execution of PCM. A common strategy for their estimation is to run the FCM algorithm first and set

$$\gamma_j = \frac{\sum_{i=1}^{N} u_{ij}^{FCM}\|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2}{\sum_{i=1}^{N} u_{ij}^{FCM}}, \quad j = 1,\ldots,m, \tag{4}$$

where $\boldsymbol{\theta}_j$'s and u_{ij}^{FCM}'s are the final FCM estimates for cluster representatives and u_{ij} coefficients, respectively. Note that γ_j can be considered as a measure of the variance of cluster C_j around its representative. Alternatively, γ_j measures the "range of influence" of cluster C_j around its representative. The basic steps of PCM are given below.

Algorithm 1 $[\Theta, U]$ = PCM(X, m)

Require: X, m

1: $t = 0$
▷ *Initialization of $\boldsymbol{\theta}_j$'s part*
2: **Initialize** $\Theta(t)$, $U^{FCM}(t)$ **via FCM**
▷ *Determination of γ_j's part*
3: $\gamma_j = \frac{\sum_{i=1}^{N} u_{ij}^{FCM}(t)\|\mathbf{x}_i - \boldsymbol{\theta}_j(t)\|^2}{\sum_{i=1}^{N} u_{ij}^{FCM}(t)}, \quad j = 1,\ldots,m$
4: **repeat**
▷ *Update U part*
5: $u_{ij}(t) = \exp\left(-\frac{\|\mathbf{x}_i - \boldsymbol{\theta}_j(t)\|^2}{\gamma_j}\right), i = 1,\ldots,N, j = 1,\ldots,m$
6: $t = t + 1$
▷ *Update Θ part*
7: $\boldsymbol{\theta}_j(t) = \frac{\sum_{i=1}^{N} u_{ij}(t-1)\mathbf{x}_i}{\sum_{i=1}^{N} u_{ij}(t-1)}, j = 1,\ldots,m$
8: **until** the difference in $\boldsymbol{\theta}_j$'s between two successive iterations becomes sufficiently small
9: **return** $\Theta = \{\boldsymbol{\theta}_1(t), \boldsymbol{\theta}_2(t), \ldots, \boldsymbol{\theta}_m(t)\}$, $U = [u_{ij}(t-1)]$

[1] An alternative PCM algorithm, resulting from the optimization of a different cost function is given in [6].

2.2 Adaptive Possibilistic C-Means - APCM

The APCM algorithm stems from the optimization of the cost function of the original PCM (eq. (1)), by setting

$$\gamma_j = \frac{\hat{\eta}}{\alpha}\eta_j, \tag{5}$$

where parameter η_j is a measure of the *mean absolute deviation* of the current form of cluster C_j, $\hat{\eta}$ is a constant defined as the minimum among all *initial* η_j's, $\hat{\eta} = \min_j \eta_j$ and α is a user-defined positive parameter that gives the algorithm the agility to deal well with closely located clusters. Specifically, larger values for parameter α are suitable for closer located. The parameter α usually takes values around 1. In principle, APCM is robust to α (see also [12]).

APCM is initialized with an overestimate of the number of natural clusters m, denoted by m_{ini}. To this end, the initialization of $\boldsymbol{\theta}_j$'s is carried out using the final cluster representatives obtained from the FCM algorithm, when the latter is executed with m_{ini} clusters. After the initialization of $\boldsymbol{\theta}_j$'s, η_j's are initialized as follows:

$$\eta_j = \frac{\sum_{i=1}^{N} u_{ij}^{FCM}\|\mathbf{x}_i - \boldsymbol{\theta}_j\|}{\sum_{i=1}^{N} u_{ij}^{FCM}}, \quad j = 1,\ldots,m_{ini}, \tag{6}$$

where $\boldsymbol{\theta}_j$'s and u_{ij}^{FCM}'s in eq. (6) are the final parameter estimates obtained by FCM.

Minimizing eq. (1) with respect to u_{ij} and $\boldsymbol{\theta}_j$, we end up with the same equations as in PCM, that is eqs. (2), (3). However, the difference between APCM and PCM in the treatment of γ_j's is that in APCM γ_j's have different definition from PCM (see eq. (5)), while, in addition, they are no longer constant but they are adapted at each iteration of the algorithm (through η_j's, see line 12 of Alg. 2). Moreover, APCM provides also a mechanism for pruning clusters as it evolves, which is related to the adaptation of γ_j's that takes into account only the most compatible to cluster C_j data points[2]. The APCM algorithm is given below.

Algorithm 2 $[\Theta, U]$ = APCM(X, m_{ini}, α)

Require: X, m_{ini}, α

1: $t = 0$
▷ *Initialization of $\boldsymbol{\theta}_j$'s part*
2: **Initialize** $\Theta(t)$, $U^{FCM}(t)$ **via FCM**
▷ *Determination of γ_j's part*
3: $\eta_j(t) = \frac{\sum_{i=1}^{N} u_{ij}^{FCM}(t)\|\mathbf{x}_i - \boldsymbol{\theta}_j(t)\|}{\sum_{i=1}^{N} u_{ij}^{FCM}(t)}, \quad j = 1,\ldots,m_{ini}$
4: $\hat{\eta} = \min_{j=1,\ldots,m_{ini}} \eta_j(t)$
5: $m(t) = m_{ini}$
6: **repeat**
7: $\gamma_j(t) = \hat{\eta}\eta_j(t)/\alpha, \quad j = 1,\ldots,m(t)$
▷ *Update U part*
8: $u_{ij}(t) = \exp\left(-\frac{\|\mathbf{x}_i - \boldsymbol{\theta}_j(t)\|^2}{\gamma_j(t)}\right), i = 1,\ldots,N, j = 1,\ldots,m(t)$
▷ *Update Θ part*
9: $\boldsymbol{\theta}_j(t+1) = \frac{\sum_{i=1}^{N} u_{ij}(t)\mathbf{x}_i}{\sum_{i=1}^{N} u_{ij}(t)}, j = 1,\ldots,m(t)$

[2] A vector $\mathbf{x}_i$ is considered to be most compatible with cluster C_j, if $u_{ij} = \max_{r=1,\ldots,m} u_{ir}$.

10: **Remove** C_j, $j = 1, \ldots, m(t)$ if there is no $\mathbf{x}_i$ that is most compatible with it, **decrease** $m(t)$ to $m(t+1)$ and **renumber** Θ and the columns of U, accordingly

11: $\boldsymbol{\mu}_j(t) = \frac{1}{n_j(t)} \sum_{\mathbf{x}_i : u_{ij}(t) = \max\limits_{r=1,\ldots,m(t+1)} u_{ir}(t)} \mathbf{x}_i, \quad j = 1, \ldots, m(t+1)$

12: $\eta_j(t+1) = \frac{1}{n_j(t)} \sum_{\mathbf{x}_i : u_{ij}(t) = \max\limits_{r=1,\ldots,m(t+1)} u_{ir}(t)} \|\mathbf{x}_i - \boldsymbol{\mu}_j(t)\|, \quad j = 1, \ldots, m(t+1)$

13: $t = t + 1$

14: **until** the difference in $\boldsymbol{\theta}_j$'s between two successive iterations becomes sufficiently small

15: **return** $\Theta = \{\boldsymbol{\theta}_1(t), \boldsymbol{\theta}_2(t), \ldots, \boldsymbol{\theta}_m(t)\}$, $U = [u_{ij}(t-1)]$

3 THE PROPOSED METHOD - GENERALIZED ADAPTIVE POSSIBILISTIC C-MEANS ALGORITHM (GAPCM)

In this section, we describe in detail the proposed GAPCM clustering algorithm, which copes with the general case of hyper-ellipsoidal clusters in the feature space. To understand how the generalization is carried out let us write eq. (1) in the following slightly different form

$$J(\Theta, U) = \sum_{j=1}^{m} \left[\sum_{i=1}^{N} u_{ij} (\mathbf{x}_i - \boldsymbol{\theta}_j)^T (\gamma_j^{-1} I)(\mathbf{x}_i - \boldsymbol{\theta}_j) + \right. \\ \left. + \sum_{i=1}^{N} (u_{ij} \ln u_{ij} - u_{ij}) \right]. \quad (7)$$

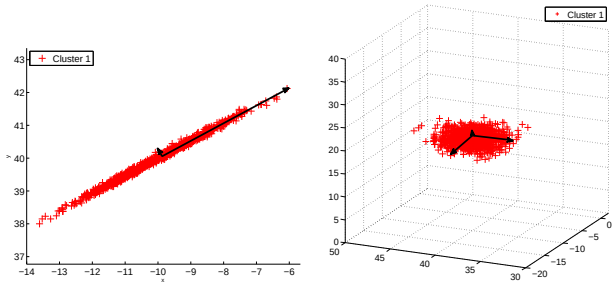
Replacing the quantity $\gamma_j^{-1} I$ with $\alpha \Sigma_j^{-1}$, where Σ_j is the covariance matrix associated with cluster C_j and α plays the same role as in APCM algorithm, the cost function associated with GAPCM is

$$J_{GAPCM}(\Theta, U) = \sum_{j=1}^{m} \left[\sum_{i=1}^{N} u_{ij} (\mathbf{x}_i - \boldsymbol{\theta}_j)^T \alpha \Sigma_j^{-1} (\mathbf{x}_i - \boldsymbol{\theta}_j) + \right. \\ \left. + \sum_{i=1}^{N} (u_{ij} \ln u_{ij} - u_{ij}) \right]. \quad (8)$$

In contrast to PCM and APCM, the introduction of covariance information Σ_j makes the algorithm capable to deal with clusters that are not spread isotropically towards all directions around their representatives. This renders GAPCM capable of dealing with subspace clustering (in cases where the spread around some principal directions is almost zero, see Fig. 3). In addition, through parameter α, GAPCM is able to control the range of influence of the hyper-ellipsoidal clusters around their representatives, towards the principal directions, as defined by Σ_j. In particular, in more demanding data sets, which contain very closely located natural clusters, larger values for the parameter α should be chosen, compared to cases of less closely located clusters, in order to discourage the movement of a representative from one dense region to another.

3.1 Parameter initialization in GAPCM

In the initialization phase of GAPCM, we generate m_{ini} initial representatives, based on the m_{ini} "most distant" points in the data set, denoted by $\boldsymbol{\theta}_j$, $j = 1, \ldots, m_{ini}$, as they are defined by the *MaxMin* procedure in [8]. In the sequel we calculate the *grade of memberships* of all $\mathbf{x}_i$'s with each $\boldsymbol{\theta}_j$, $j = 1, \ldots, m_{ini}$ as FCM algorithm indicates, in order to initialize u_{ij}'s, $i = 1, \ldots, N$, $j = 1, \ldots, m_{ini}$.



(a) 1-dim cluster in 2-dim space (b) 2-dim cluster in 3-dim space

Figure 3: Cluster cases where the spread around some principal directions are almost zero.

After the initialization of the cluster representatives $\boldsymbol{\theta}_j$'s and u_{ij}'s, the covariance matrices Σ_j's of all clusters C_j's are initialized as follows:

$$\Sigma_j = \frac{\sum_{i=1}^{N} u_{ij} (\mathbf{x}_i - \boldsymbol{\theta}_j)(\mathbf{x}_i - \boldsymbol{\theta}_j)^T}{\sum_{i=1}^{N} u_{ij}}, \quad j = 1, \ldots, m_{ini}. \quad (9)$$

3.2 Parameter adaptation in GAPCM

In GAPCM algorithm, all parameters are adapted during its execution. More specifically, this refers to, (a) the degrees of compatibility u_{ij}'s and the cluster representatives $\boldsymbol{\theta}_j$'s, (b) the Σ_j's and (c) the number of clusters m, with (b) and (c) being achieved through two interrelated processes.

As far as the updating of u_{ij}'s is concerned, minimizing $J_{GAPCM}(\Theta, U)$ (eq. (8)) with respect to u_{ij}, $i = 1, \ldots, N$, $j = 1, \ldots, m$, we end up with the following expression

$$u_{ij} = \exp\left(-(\mathbf{x}_i - \boldsymbol{\theta}_j)^T \alpha \Sigma_j^{-1} (\mathbf{x}_i - \boldsymbol{\theta}_j) \right). \quad (10)$$

In addition, the updating of $\boldsymbol{\theta}_j$'s is done as in the original PCM scheme according to eq. (3).

Motivated by APCM, Σ_j's in GAPCM are adapted at each iteration of the algorithm. More specifically, we propose to compute Σ_j of a cluster C_j, $j = 1, \ldots, m$ as the covariance matrix of the data vectors that are *most compatible* to cluster C_j, i.e.,

$$\Sigma_j = \frac{\sum_{\mathbf{x}_i : u_{ij} = \max\limits_{r=1,\ldots,m} u_{ir}} (\mathbf{x}_i - \boldsymbol{\mu}_j)(\mathbf{x}_i - \boldsymbol{\mu}_j)^T}{n_j}, \quad (11)$$

where n_j denotes the number of the data points $\mathbf{x}_i$ that are most compatible with cluster C_j and $\boldsymbol{\mu}_j$ the mean vector of these data points[3].

Note that, as eq. (10) indicates, matrix Σ_j is required to be invertible,. To this end, at each iteration t of the algorithm we examine whether the rank of matrix Σ_j, $j =, \ldots, m$ is less than the

[3]Note that in the adaptation mechanism of Σ_j, $\boldsymbol{\mu}_j$ is used instead of $\boldsymbol{\theta}_j$. This is because only most compatible to C_j points are taken into account, whose mean vector is $\boldsymbol{\mu}_j$ and not $\boldsymbol{\theta}_j$.

dimensionality l. If this is the case, the associated cluster C_j is eliminated and U and Θ are updated accordingly. As a result, the number of current clusters is reduced. This is the cluster elimination mechanism that adjusts the number of clusters m, as the algorithm evolves. In the sequel, the GAPCM algorithm is stated explicitly.

Algorithm 3 $[\Theta, U]$ = GAPCM(X, m_{ini}, α)

Require: X, m_{ini}, α

$t = 0$

▷ *Initialization part*

Find the m_{ini} "most distant" points $\boldsymbol{\theta}_j(t)$'s, $j = 1, \ldots, m_{ini}$ in X as in [8].

Calculate $u_{ij}(t)$, $i = 1, ..., N$, $j = 1, ..., m_{ini}$ via the related equation of FCM.

▷ *Determination of Σ_j's part*

$\Sigma_j = \frac{\sum_{i=1}^{N} u_{ij}(\mathbf{x}_i - \boldsymbol{\theta}_j)(\mathbf{x}_i - \boldsymbol{\theta}_j)^T}{\sum_{i=1}^{N} u_{ij}}$, $j = 1, \ldots, m_{ini}$,

$m(t) = m_{ini}$

repeat

▷ *Update U part*

$u_{ij}(t) = \exp\left(-(\mathbf{x}_i - \boldsymbol{\theta}_j(t))^T \alpha \Sigma_j(t)^{-1} (\mathbf{x}_i - \boldsymbol{\theta}_j(t))\right), i = 1, \ldots, N, j = 1, \ldots, m(t)$

▷ *Update Θ part*

$\boldsymbol{\theta}_j(t+1) = \frac{\sum_{i=1}^{N} u_{ij}(t)\mathbf{x}_i}{\sum_{i=1}^{N} u_{ij}(t)}, j = 1, \ldots, m(t)$

Remove C_j if rank $(\Sigma_j) < l$, **decrease** $m(t)$ to $m(t+1)$ and **renumber** Θ and the columns of U, accordingly

$\boldsymbol{\mu}_j(t) = \frac{1}{n_j(t)} \sum_{\mathbf{x}_i : u_{ij}(t) = \max_{r=1,\ldots,m(t+1)} u_{ir}(t)} \mathbf{x}_i$, $j = 1, ..., m(t+1)$

$\Sigma_j(t+1) = \frac{\sum_{\mathbf{x}_i : u_{ij}(t) = \max_{r=1,\ldots,m(t+1)} u_{ir}(t)} (\mathbf{x}_i - \boldsymbol{\mu}_j(t))(\mathbf{x}_i - \boldsymbol{\mu}_j(t))^T}{n_j(t)}$, $j = 1, ..., m(t+1)$

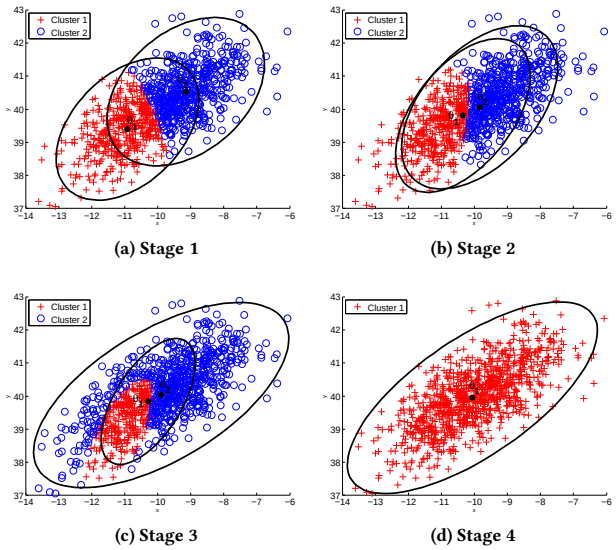
$t = t + 1$

until the difference in $\boldsymbol{\theta}_j$'s between two successive iterations becomes sufficiently small

return $\Theta = \{\boldsymbol{\theta}_1(t), \boldsymbol{\theta}_2(t), \ldots, \boldsymbol{\theta}_m(t)\}$, $U = [u_{ij}(t-1)]$

Focusing on the definition of Σ_j's in the proposed updating mechanism from eg. (11), recall that only the data vectors that are most compatible to cluster C_j are taking into account. This particularity is responsible for succeeding cluster elimination. In order to gain some insight into the way GAPCM eliminates clusters, let us focus on a *single* physical cluster. To further ease the discussion, let us assume that all data vectors form a single physical cluster and two representatives $\boldsymbol{\theta}_1$, $\boldsymbol{\theta}_2$ have been moved iteratively by the algorithm to its center (see Fig. 4). Consider also the possibilistic nature of GAPCM, that is, all representatives are moved towards the centers of their closest dense in data regions (physical clusters), independently from each other. Loosely speaking, between coincident cluster representatives that represent the same physical cluster, one of them, say $\boldsymbol{\theta}_2$ will have *stronger* influence around it than the other ($\boldsymbol{\theta}_1$)[4]. In the example of Fig. 4, both $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ are moved towards the center of the physical cluster. However, the influence of cluster C_2 is stronger than that of C_1, thus all the data points of

[4] As is defined by the eigenvalues of the Σ_j's.



(a) Stage 1 (b) Stage 2

(c) Stage 3 (d) Stage 4

Figure 4: Cluster elimination stages.

the physical cluster will gradually become most compatible with C_2. Thus, at a stage where cluster C_1 will have no (or less than l) compatible points within it, it will be eliminated, leaving only one representative representing the specific physical cluster ($\boldsymbol{\theta}_2$) (see Fig. 4d)[5].

4 EXPERIMENTAL RESULTS

In this section we assess the performance of GAPCM by comparing it with other relative algorithms, on the basis of both simulated and real cases. More specifically, in the first part of this section, we compare the clustering performance of GAPCM with that of the k-means [5], the PCM [7], the APCM [12] and the GMM [11] algorithm, in two synthetic experimental data sets and we illustrate the obtained results. In order to compare a clustering with the true data label information, we use the Success Rate (SR), which measures the percentage of the points that have been correctly labeled by each algorithm. Finally, the number of iterations (Iter) and the time (in seconds) required for the convergence of each algorithm, are provided[6]. In the second part of this section, we describe in detail a specific audio blind source separation application and we assess the performance of the proposed method in two experiments of real audio data sets.

4.1 Synthetic Data Sets

Experiment 1: Let us 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. 5a). 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

[5] Note that this scenario should not be confused with cases like the one of Fig. 2 where two distinct physical clusters with coincident cluster centers exist.

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

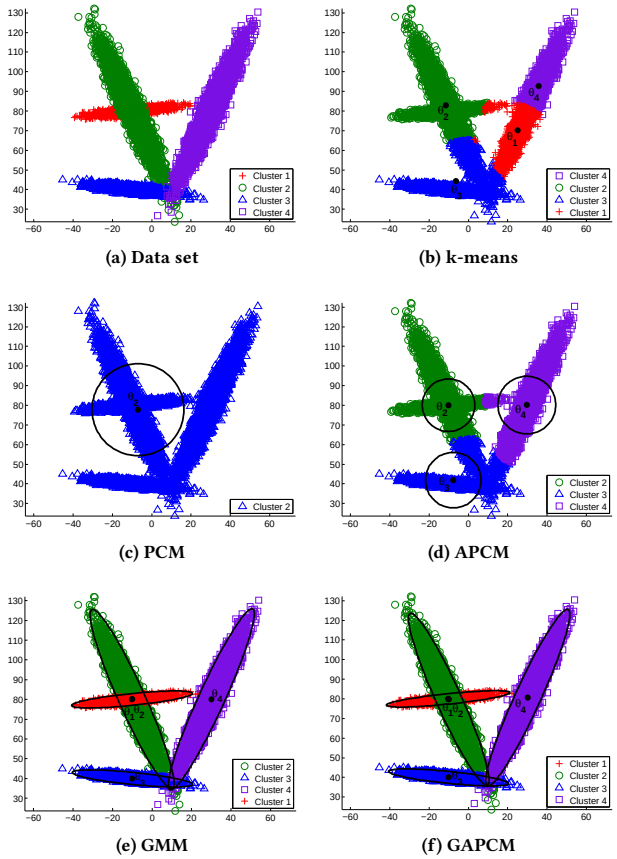


Figure 5: (a) Data set of experiment 1. Clustering results for (b) k-means, $m = 4$, (c) PCM, $m_{ini} = 10$, (d) APCM, $m_{ini} = 10$ and $\alpha = 1$, (e) GMM, $m = 4$ and (f) GAPCM, $m_{ini} = 10$ and $\alpha = 1$.

matrices are set to $\Sigma_1 = \begin{bmatrix} 100 & 10 \\ 10 & 2 \end{bmatrix}$, $\Sigma_2 = \begin{bmatrix} 50 & -100 \\ -100 & 225 \end{bmatrix}$, $\Sigma_3 = \begin{bmatrix} 100 & -10 \\ -10 & 2 \end{bmatrix}$, 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 the other two distributions. Note that the centers of C_1 and C_2 coincide and the "tails" of the ellipsoids of C_2, C_3 and C_4 are partially overlapped. Table. 1 shows the clustering results of all algorithms where m_{ini} and m_{final} denote the initial and the final number of clusters, respectively. Fig. 5 illustrates the best clustering results obtained by each algorithm, with its parameters chosen as stated in the figure caption.

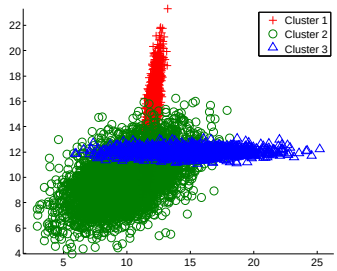
As it can be deduced from Fig. 5 and Table. 1, even when the k-means is initialized with the actual number of clusters ($m = 4$), it fails to distinguish cluster C_1 from C_2, due to their coincident centers, thus splitting physical cluster C_4 to two clusters. The classical PCM fails to detect any clustering structure. 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. Finally, GMM and GAPCM produce very accurate results identifying correctly all physical clusters. However, it is highlighted the fact that, in contrast to GMM where the actual number of clusters is a pre-requisite, no such requirement is imposed by GAPCM.

Table 1: Performance of all algorithms for the experiment 1 data set.

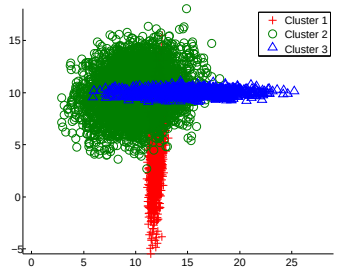
	m_{ini}	m_{final}	$SR(\%)$	$Iter$	$Time$ (secs)
k-means	10	10	37.18	43	0.30
k-means	4	4	62.86	21	0.14
PCM	10	1	35.71	22	1.31
PCM	15	1	35.71	23	2.03
APCM ($\alpha = 1$)	10	3	80.35	36	0.33
APCM ($\alpha = 1$)	15	3	80.57	82	0.66
GMM	10	10	56.44	166	1495(!)
GMM	**4**	**4**	**96.28**	22	77.10
GAPCM ($\alpha = 1$)	**10**	**4**	**96.16**	41	124
GAPCM ($\alpha = 1$)	**15**	**4**	**96.17**	46	152

Experiment 2: Let us now consider a three-dimensional data set consisting of $N = 8000$ points, which form three physical clusters C_1, C_2 and C_3 (Fig. 6a, 6b, 6c). Each cluster is modelled by a normal distribution. The means of the distributions are $\mathbf{c}_1 = [12, 15, 5]^T$, $\mathbf{c}_2 = [10, 10, 10]^T$ and $\mathbf{c}_3 = [15, 12, 10]^T$ respectively, while their covariance matrices are set to $\Sigma_1 = \begin{bmatrix} 0.1 & 0.5 & 0.4 \\ 0.5 & 5 & 0.5 \\ 0.4 & 0.5 & 10 \end{bmatrix}$, $\Sigma_2 = \begin{bmatrix} 5 & 2 & 0.8 \\ 2 & 3 & 0.2 \\ 0.8 & 0.2 & 4 \end{bmatrix}$ and $\Sigma_3 = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.1 \end{bmatrix}$, respectively. A number of 2000 points are generated by the first distribution, 5000 points are generated by the second one and 1000 points by the third one. Note that the points of the cluster C_1 lie around a plane, the points of the cluster C_2 form a 3-dimensional ellipsoid and, finally, the points of the cluster C_3 lie around a line in the 3-dimensional space. Table. 2 shows the clustering results of all algorithms and Fig. 6 illustrates the best clustering results obtained by each algorithm, with its parameters chosen as stated in the figure caption. Specifically, the first column in Fig. 6 illustrates the 1st and 2nd dimensions of the data set, the second column illustrates the 1st and 3rd dimensions and the third column illustrates the 2nd and 3rd dimensions of the data set, in order to get a better visualization of the clusters and of the clustering results of the algorithms in the 3-dimensional space.

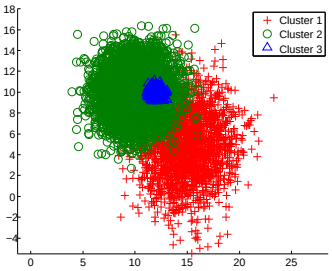
As it can be deduced from Fig. 6 and Table. 2, when k-means is initialized with the actual number of clusters ($m = 3$), it manages to identify correctly the cluster centers, however, it fails to classify correctly the data points to the three detected clusters. The PCM exhibits degraded performance, for all choices of m_{ini}. Similar to PCM is the behaviour observed for the APCM algorithm, which also fails to distinguish any clustering structure. On the contrary, GAPCM and GMM (the latter only given the actual number of clusters) exhibit superior performance compared to other algorithms, detecting correctly all the underlying clusters of various shapes.



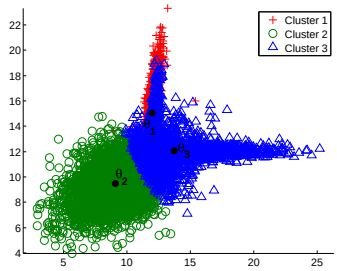
(a) Data set (1st and 2nd dimensions)



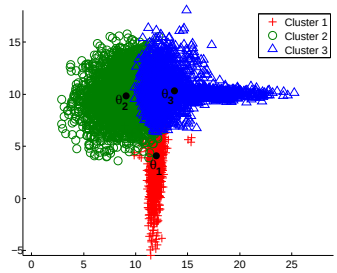
(b) Data set (1st and 3rd dimensions)



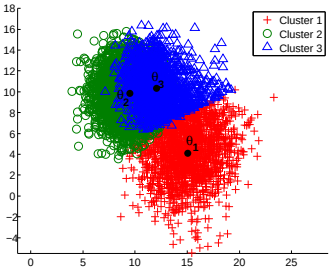
(c) Data set (2nd and 3rd dimensions)



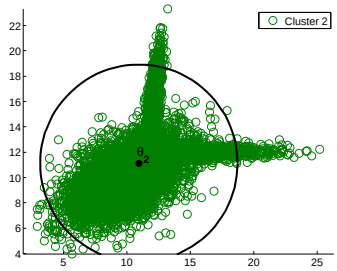
(d) k-means (1st and 2nd dimensions)



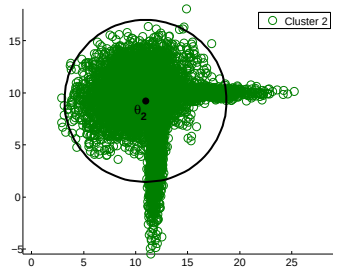
(e) k-means (1st and 3rd dimensions)



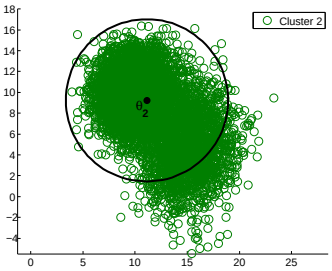
(f) k-means (2nd and 3rd dimensions)



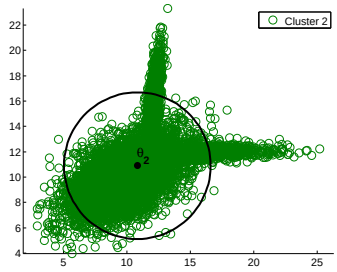
(g) PCM (1st and 2nd dimensions)



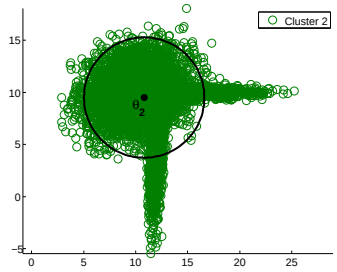
(h) PCM (1st and 3rd dimensions)



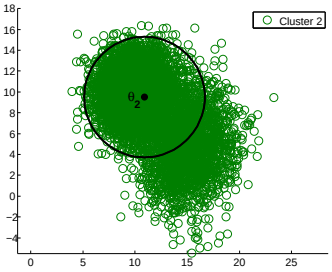
(i) PCM (2nd and 3rd dimensions)



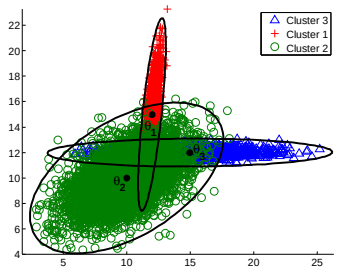
(j) APCM (1st and 2nd dimensions)



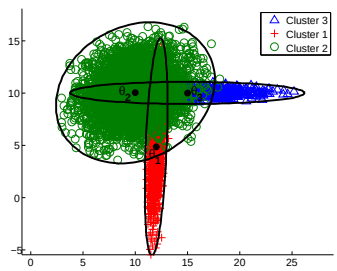
(k) APCM (1st and 3rd dimensions)



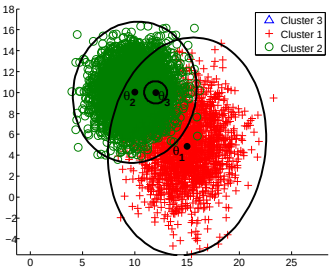
(l) APCM (2nd and 3rd dimensions)



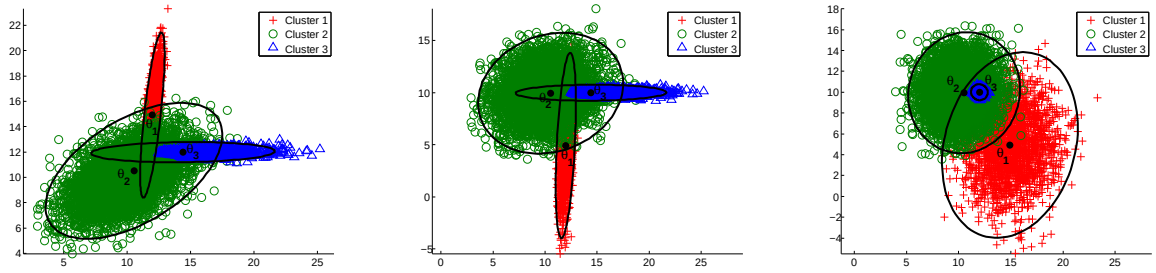
(m) GMM (1st and 2nd dimensions)



(n) GMM (1st and 3rd dimensions)



(o) GMM (2nd and 3rd dimensions)

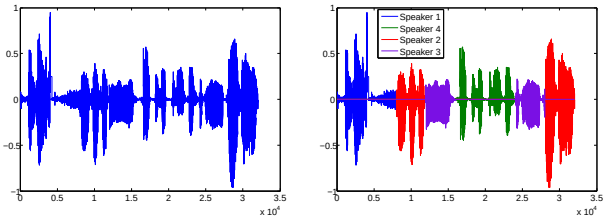


(p) GAPCM (1st and 2nd dimensions) (q) GAPCM (1st and 3rd dimensions) (r) GAPCM (2nd and 3rd dimensions)

Figure 6: (a-c) Data set of experiment 2. Clustering results for (d-f) k-means, $m = 3$, (g-i) PCM, $m_{ini} = 5$, (j-l) APCM, $m_{ini} = 5$ and $\alpha = 1$, (m-o) GMM, $m = 3$ and (p-r) GAPCM, $m_{ini} = 10$ and $\alpha = 1$.

Table 2: Performance of all algorithms for the experiment 2 data set.

	m_{ini}	m_{final}	$SR(\%)$	$Iter$	$Time$ (secs)
k-means	5	5	50.20	35	0.14
k-means	3	3	78.39	18	0.20
PCM	10	1	62.50	11	0.41
PCM	5	1	62.50	8	0.17
APCM ($\alpha = 1$)	10	1	62.50	16	0.09
APCM ($\alpha = 1$)	5	1	62.50	13	0.06
GMM	10	10	43.58	512	2726(!)
GMM	**3**	**3**	**95.96**	19	30.92
GAPCM ($\alpha = 1$)	**10**	**3**	**93.06**	51	67.7
GAPCM ($\alpha = 1$)	**5**	**3**	**93.05**	41	50.2



(a) Audio signal (b) Labelled audio signal

Figure 7: (a) Audio signal of experiment 3 and (b) its association with the speakers.

4.2 Application to Blind Source Separation Problem

A common problem that is met in several applications is the separation of the original signals coming from differnet sources, out of mixtures of them. The formation for the audio source separation case is as follows: we have in the same room m audio signal sources and q microphones that record mixtures of the signals emitted from these sources. The aim is to recover the original signals based on their mixtures recorded through the microphones. In this section we show the potential usefulness of the GAPCM algorithm in this application. Formally speaking (ignoring time delays and reverberations), let $s_j(n)$ be the value of the j-th source at time n, $j = 1, \ldots, m$, $x_k(n)$ the value recorded from the k-th microphone at time n, $k = 1, \ldots, q$, a_{jk} the coefficient that measures the contribution of the j-th source to the k-th audio mixture and $e_k(n)$ is the (additive) noise associated with the k-th microphone, which is assumed to be zero mean. These quantities are combined via the following equation[7]

$$x_k(n) = \sum_{j=1}^{m} a_{jk} s_j(n) + e_k(n), \quad k = 1, \ldots, q, \quad n = 1, \ldots, N \qquad (12)$$

(where N is the number of mixed samples recorded at each microphone) or, in matrix notation,

$$X = SA + E \qquad (13)$$

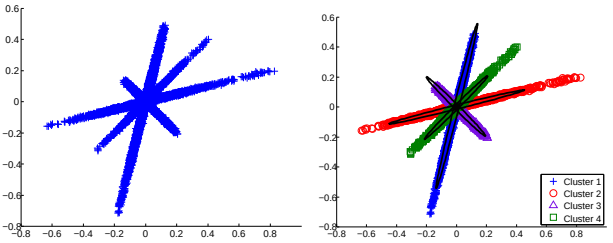
where X is a $N \times q$ matrix, whose k-th column contains the N mixed samples recorded by microphone k, S and E are $N \times m$ and $N \times q$ matrices, respectively, defined accordingly and, finally, A is the $m \times q$ *mixing matrix*. Note that, the rows of X are aggregated along m linear manifolds in the q-dimensional space, where the row vectors of A specify their orientation.

In the sequel we consider the blind source separation problem where both A and S are unknown and we show how GAPCM can be used to assist the identification of S. First, we apply the GAPCM algorithm for the rows of X, where each row corresponds to a single point in the q dimensional feature space $x_1 - x_2 - \ldots - x_q$ and we identify the lines they form, through their corresponding direction vectors and their intecepts $\boldsymbol{w}_j$'s and w_{0j}'s, respectively. We set $\boldsymbol{a}_j$ equal to the eigenvector that corresponds to the minimum eigenvalue of Σ_j, $j = 1, \ldots, m$. Then, to determine the elements of the matrix S we work as follows. For the i-th row vector of $X = [\boldsymbol{X}_1^T | \ldots | \boldsymbol{X}_N^T]^T$, we determine, among the m hyper-line clusters in the $x_1 - \ldots - x_q$ space, the closest to it, say i_r, and we set $\hat{S}_{i,i_r} = \boldsymbol{X}_i \boldsymbol{a}_{i_r}$[8] and $\hat{S}_{is} = 0$, for $s \neq i_r$.

Experiment 3: Consider the case of four audio sources ($m = 4$) and two microphones ($q = 2$). Fig. 7a illustrates the mixtured audio signal of the four sources and Fig. 7b highlights the contribution of each speaker. Fig. 8a depicts the data corresponding to the four audio sources in the 2-dimensional space, which form four lines (one for each source). Fig. 8b shows the GAPCM clustering result, detecting correctly all lines. Finally, Fig. 11 illustrates the results

[7] For more details, see [9], from where the following formalism has been borrowed.

[8] Keep in mind that $\boldsymbol{X}_i$ is a row vector.

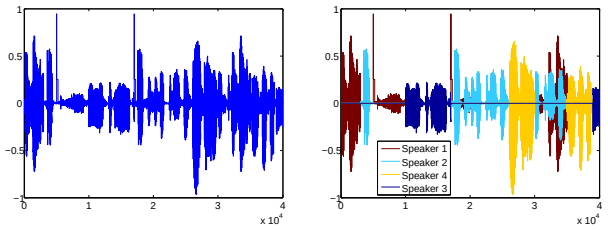


(a) Audio signal in feature space (b) Clustering result of GAPCM

Figure 8: (a) The audio signal of experiment 3, depicted in the feature space and (b) the clustering result of GAPCM ($m_{ini} = 5$, $\alpha = 1$).

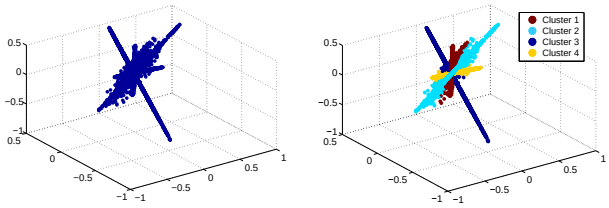
of GAPCM, where all audio sources are correctly identified. Note that the correlation coefficient ($R_j, j = 1, ..., 4$) and the mixing error ratio ($\text{MER}_j, j = 1, ..., 4$, see [4]) of each estimated source with the original one are given in the caption of Fig. 11.

Experiment 4: Consider now the case of four audio sources ($m = 4$) and three microphones ($q = 3$). Fig. 9a illustrates the mixtured audio signal of the four sources and Fig. 9b highlights the contribution of each speaker. As it can be deduced, in this particular experiment, there are several time stamps where two audio sources are activated simultaneously (speaker 1 and 2). Recalling arguments from experiment 3, we expect that the time stamps where only one speaker is activated correspond to entries of X that lie around a line (1-dimensional cluster). On the other hand, the time stamps where two speakers are activated simultaneously, correspond to entries of X that lie around a plane (2-dimensional cluster) defined by the two 1-dimensional clusters of the associated speakers (see Fig. 10a). As a result, five ellipsoidally shaped clusters are formed (four 1-dimensional clusters and one 2-dimensional cluster) by the data set in the 3-dimensional feature space. Applying GAPCM for this data set, we identify correctly all hyper-ellipsoidal clusters. This result indicates an additional potentiality of GAPCM, that is, its ability not to simply unravel clusters lying around subspaces of the feature space, but also clusters whose associated subspaces lie entirely to subspaces of other clusters. Returning to the frame of the application considered in this experiment, taking into consideration that each data point, initially classified in the 2-dimensional cluster, has to be classified to its closest of the 1-dimensional clusters, we work as follows. First, we consider only the clusters, whose associated subspaces have dimensionality 1 (cases where only one speaker is activated), by identifying all the clusters $C_j, j = 1, ..., m$, whose Σ_j has all but one eigenvalues close to zero. Then, we assign the rest data points that do not belong to these clusters, to their closest among these identified 1-dimensional clusters. The final clustering result of GAPCM for the experiment 4 is depicted in Fig. 10b. Finally, Fig. 12 illustrates the source separation results utilizing GAPCM, which identifies all audio sources with high accuracy, managing to satisfactorily distinguish the two speakers that were talking simultaneously on several time stamps.



(a) Audio signal (b) Labelled audio signal

Figure 9: (a) Audio signal of experiment 4 and (b) its association with the speakers.



(a) Audio signal in feature space (b) Clustering result of GAPCM

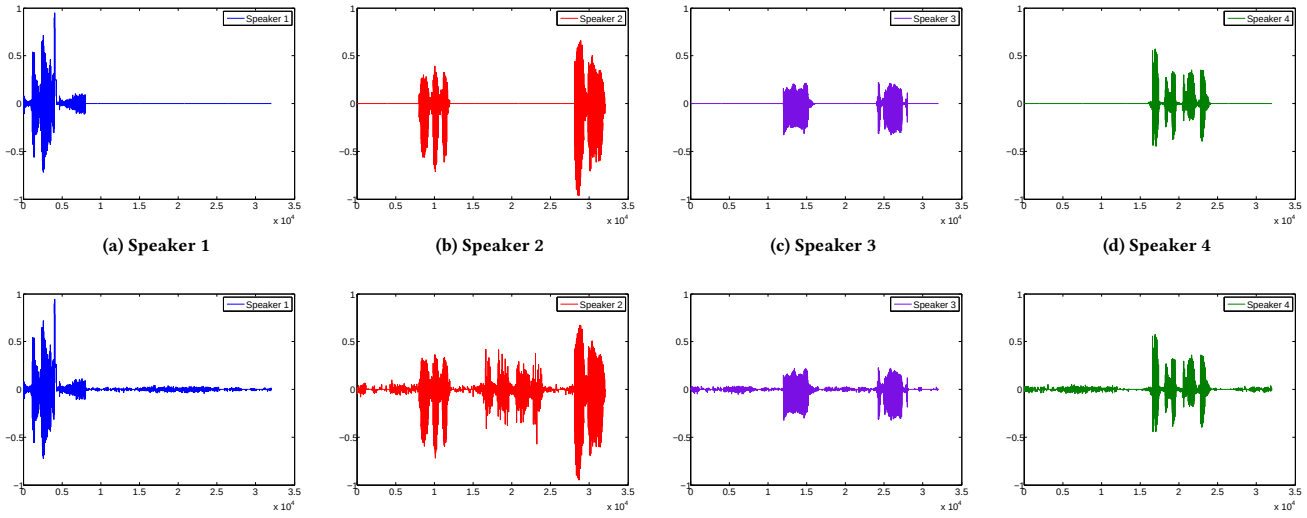
Figure 10: (a) The audio signal of experiment 4, depicted in the feature space and (b) the clustering result of GAPCM ($m_{ini} = 10$, $\alpha = 1$).

5 CONCLUSION

In this paper a generalized possibilistic c-means clustering algorithm, called Generalized Adaptive Possibilistic C-Means (GAPCM), has been derived. The main feature of GAPCM is the adaptation of its parameters during the execution of the algorithm, in order GAPCM to successfully identify the centers and the actual "shapes" of hyper-ellipsoidal clusters in the feature space. Compared to its ancestor APCM, GAPCM is also able to unravel clusters whose points lie around certain subspaces of the feature space. In addition, a cluster elimination procedure that GAPCM incorporates, renders it able to detect the true number of physical clusters, m, provided that it is initialized with an overestimate of it. The latter releases GAPCM from the noose of knowing exactly in advance the actual number of m. The experimental results on both simulated and real data (blind source separation application) show that GAPCM exhibits superior performance compared to several other algorithms.

REFERENCES

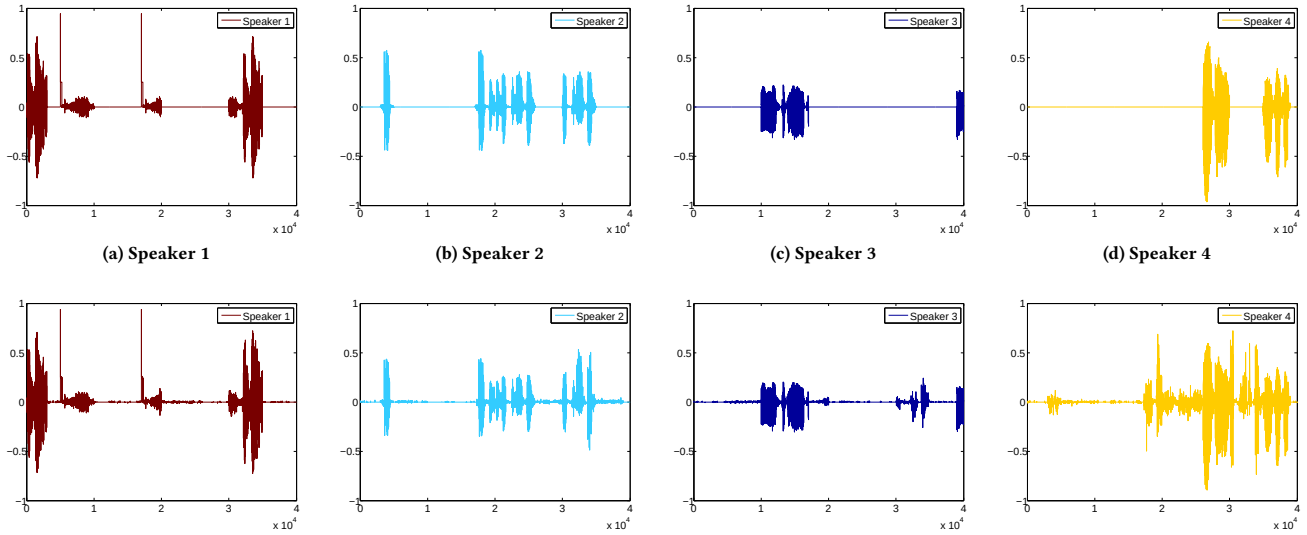
[1] M. Barni, V. Cappellini, and A. Mecocci. 1996. Comments on A Possibilistic Approach to Clustering. *IEEE Transactions on Fuzzy Systems* 4 (1996), 393–396.
[2] J. C. Bezdek. 1980. A Convergence Theorem for the Fuzzy Isodata Clustering Algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2 (1980), 1–8.
[3] J. C. Bezdek. 1981. *Pattern Recognition with Fuzzy Objective Function Algorithms.* Plenum.
[4] P. Bofill E. Vincent, S. Araki. 2009. The 2008 Signal Separation Evaluation Campaign: A community-based approach to large-scale evaluation. *8th Int. Conf. on Independent Component Analysis and Signal Separation (ICA)* (2009), 734741.
[5] J. A. Hartigan and M. A. Wong. 1979. Algorithm AS 136: A k-Means Clustering Algorithm. *Journal of the Royal Statistical Society* 28 (1979), 100–108.



(a) Speaker 1 **(b) Speaker 2** **(c) Speaker 3** **(d) Speaker 4**

(e) Estimated Speaker 1 ($R_1 = 0.9973$, $MER_1 = 24.3dB$) **(f) Estimated Speaker 2** ($R_2 = 0.9842$, $MER_2 = 13.3dB$) **(g) Estimated Speaker 3** ($R_3 = 0.9821$, $MER_3 = 12.3dB$) **(h) Estimated Speaker 4** ($R_4 = 0.9953$, $MER_4 = 20.4dB$)

Figure 11: (a-d) The audio signals of each speaker individually for experiment 3 and (e-h) their respective estimations by GAPCM.



(a) Speaker 1 **(b) Speaker 2** **(c) Speaker 3** **(d) Speaker 4**

(e) Estimated Speaker 1 ($R_1 = 0.9832$, $MER_1 = 6.2dB$) **(f) Estimated Speaker 2** ($R_2 = 0.9219$, $MER_2 = 7.8dB$) **(g) Estimated Speaker 3** ($R_3 = 0.9729$, $MER_3 = 12.3dB$) **(h) Estimated Speaker 4** ($R_4 = 0.9899$, $MER_4 = 14.9dB$)

Figure 12: (a-d) The audio signals of each speaker individually for experiment 4 and (e-h) their respective estimations by GAPCM.

[6] R. Krishnapuram and J. M. Keller. 1993. A Possibilistic Approach to Clustering. *IEEE Transactions on Fuzzy Systems* 1 (1993), 98–110.
[7] R. Krishnapuram and J. M. Keller. 1996. The Possibilistic C-Means Algorithm: Insights and Recommendations. *IEEE Transactions on Fuzzy Systems* 4 (1996), 385–393.
[8] B. Mirkin. 2005. *Clustering for Data Mining: A Data Recovery Approach.* CRC Press.
[9] M.D. Plumbley, T. Blumensath, L. Daudet, and M.E. Davies R. Gribonval. 2010. Sparse representation in Audio and Music: from Coding to Source separation. *Proc. IEEE* 98, 6 (2010), 995–1005.
[10] R. Shioda and L. Tunçel. 2007. Clustering via minimum volume ellipsoids. *Springer Science and Business Media* 37 (2007), 247–295.
[11] S. Theodoridis and K. Koutroumbas. 2009. *Pattern Recognition* (4th ed.). Academic Press.
[12] S. D. Xenaki, K. D. Koutroumbas, and A. A. Rontogiannis. 2016. A Novel Adaptive Possibilistic Clustering Algorithm. *IEEE Transactions on Fuzzy Systems* 24, 4 (2016), 791–810.
[13] S. D. Xenaki, K. D. Koutroumbas, and A. A. Rontogiannis. 2016. Sparsity-aware Possibilistic Clustering Algorithms. *IEEE Transactions on Fuzzy Systems* 24, 6 (2016), 1611–1626.
[14] J-S. Zhang and Y-W. Leung. 2004. Improved Possibilistic C-Means Clustering Algorithms. *IEEE Transactions on Fuzzy Systems* 12, 2 (2004), 209–217.