NATIONAL AND KAPODISTRIAN UNIVERSITY OF ATHENS

SCHOOL OF SCIENCES
DEPARTMENT OF INFORMATICS AND TELECOMMUNICATIONS

PROGRAM OF POSTGRADUATE STUDIES

PhD THESIS

Advances in Possibilistic Clustering with Application to Hyperspectral Image Processing

Spyridoula D. Xenaki

ATHENS

MAY 2017

ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

Προηγμένες Τεχνικές Ομαδοποίησης Δεδομένων με Βάση τα Ενδεχόμενα με Εφαρμογή στην Επεξεργασία Υπερφασματικών Εικόνων

Σπυριδούλα Δ. Ξενάκη

ΑΘΗΝΑ

ΜΑΪΟΣ 2017

PhD THESIS

Advances in Possibilistic Clustering with Application to Hyperspectral Image Processing

Spyridoula D. Xenaki

SUPERVISOR: Sergios Theodoridis, Professor UoA

THREE-MEMBER ADVISORY COMMITTEE:

Sergios Theodoridis, Professor UoA

Konstantinos Koutroumbas, Senior Researcher IAASARS/NOA

Athanasios Rontogiannis, Research Director IAASARS/NOA

SEVEN-MEMBER EXAMINATION COMMITTEE

Sergios Theodoridis,
Professor UoA

Konstantinos Koutroumbas,
Senior Researcher IAASARS/NOA

Athanasios Rontogiannis,
Research Director IAASARS/NOA

Konstantinos Berberidis,
Professor Uni. Patras

Dimitrios Maroulis,
Professor UoA

Dimitrios Gunopulos,
Professor UoA

Elias Manolakos,
Professor UoA

Examination Date: May 10, 2017

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

Προηγμένες Τεχνικές Ομαδοποίησης Δεδομένων με Βάση τα Ενδεχόμενα με Εφαρμογή στην Επεξεργασία Υπερφασματικών Εικόνων

Σπυριδούλα Δ. Ξενάκη

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ: Σέργιος Θεοδωρίδης, Καθηγητής ΕΚΠΑ

ΤΡΙΜΕΛΗΣ ΕΠΙΤΡΟΠΗ ΠΑΡΑΚΟΛΟΥΘΗΣΗΣ:

Σέργιος Θεοδωρίδης, Καθηγητής ΕΚΠΑ

Κωνσταντίνος Κουτρούμπας, Κύριος Ερευνητής ΙΑΑΔΕΤ/ΕΑΑ

Αθανάσιος Ροντογιάννης, Διευθυντής Ερευνών ΙΑΑΔΕΤ/ΕΑΑ

ΕΠΤΑΜΕΛΗΣ ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ

Σέργιος Θεοδωρίδης,
Καθηγητής ΕΚΠΑ

Κωνσταντίνος Κουτρούμπας,
Κύριος Ερευνητής ΙΑΑΔΕΤ/ΕΑΑ

Αθανάσιος Ροντογιάννης,
Διευθυντής Ερευνών ΙΑΑΔΕΤ/ΕΑΑ

Κωνσταντίνος Μπερμπερίδης,
Καθηγητής Παν. Πατρών

Δημήτριος Μαρούλης,
Καθηγητής ΕΚΠΑ

Δημήτριος Γουνόπουλος,
Καθηγητής ΕΚΠΑ

Ηλίας Μανωλάκος,
Καθηγητής ΕΚΠΑ

Ημερομηνία Εξέτασης: 10 Μαίου 2017

ABSTRACT

Clustering is a well established data analysis methodology that lie in the framework of pattern recognition and it has been extensively used in various fields of applications during the last decades. Given a set of *objects*, the aim of clustering is the identification of groups (*clusters*) formed by "similar" objects. A major effort in the clustering bibliography has been devoted to the identification of compact and hyperellipsoidally shaped clusters and one of the most well-known categories of clustering algorithms that are commonly used in the literature for this purpose is that of the *cost function optimization* based algorithms. The present thesis focuses on the Possibilistic C-Means (PCM) algorithms that are one of the most well-known representatives of the aforementioned category. Specifically, exposing first their shortcomings, they are extended next, in order to overcome them. These extensions rely on the adoption of the parameter adaptivity and the sparsity concepts. In the sequel, the main contributions of the present thesis are briefly exposed.

First, a novel approach in the context of possibilistic clustering algorithms, named *Adaptive Possibilistic C-Means* (APCM) has been developed. APCM addresses several of the weaknesses of original PCM, by allowing the *adaptation* of some parameters that are characteristic to all PCM algorithms, during its execution. This is in contrast to classical PCM algorithms where these parameters, once they are set, they remain fixed. This characteristic of APCM gives rise to two new features that are not met in classical PCM algorithms. The first one is that APCM is capable, in principle, to reveal the true number of physical clusters formed by the data, provided that it starts with a reasonable overestimate of it, thus overcoming a long-standing issue in the clustering literature. This is carried out by removing the clusters that gradually become obsolete (i.e., the clusters whose characteristic parameter diminishes towards zero as the algorithm evolves). The other feature resulting from the adaptation of the characteristic parameters of APCM is the increase of its flexibility in following the variations in the formation of the clusters during the algorithm execution. This makes APCM able to uncover the underlying clustering structure, even in demanding cases, where the physical clusters are closely located to each other and/or have significant differences in their variances. APCM is compared against several related state-of-the-art algorithms through extensive simulations on both synthetic and real data and the provided results show that APCM exhibits superior performance in almost all the considered data sets. Moreover, theoretical results that are indicative of the convergence behavior of the algorithm are also provided.

Next, the focus is turned on extending PCM by introducing the concept of *sparsity*. The rationale behind this extension is that, in practice, a data point is most compatible with at most one, a few or even none cluster (outlier). Thus, taking into account the data points that are most compatible with a given cluster and excluding those that are not compatible with it, leads to more accurate estimations of the clusters' parameters. The resulting algorithm, called *Sparse Possibilistic C-Means* (SPCM) can deal well with closely located clusters that may also be of significantly different densities, while at the same time it exhibits immunity to noise and outliers. Finally, a non-trivial convergence proof for the SPCM algorithm is conducted. The main source of difficulty in the provided convergence anal-

ysis, compared to those given for previous possibilistic algorithms, relies on the fact that one of its updating parameter equations is not given in closed form but is computed via a two-branch expression, which defines a non-continuous mapping. In the present dissertation, it is shown that SPCM will converge to one of the local minima of its associated cost function. As a side effect, it is also shown that similar convergence results can be derived for the PCM algorithm, viewed as a special case of SPCM, which are stronger than those established in previous works.

In the sequel, the main features of the proposed APCM and SPCM algorithms are combined giving rise to the *Sparse Adaptive Possibilistic C-Means* (SAPCM) algorithm, which, inheriting all the advantages of its ancestors, has the ability to (a) cope well with demanding data sets with closely located physical clusters with possibly different densities and/or variances, (b) determine the number of physical clusters and (c) improve even more the estimates of the clusters' parameters, compared to APCM and SPCM. Extensive experimentation verified the overall advantages of SAPCM compared to other related algorithms. Moreover, two variants of SAPCM, which use the above original SAPCM algorithm as a building block, have been devised. The first one is an iterative bottom-up version, called *Sequential SAPCM* (SeqSAPCM), which, at each iteration, determines a single new cluster by employing SAPCM. Thus, it unravels sequentially the underlying clustering structure. The basic advantage of SeqSAPCM is that it does not require knowledge of the number of physical clusters (not even a crude overestimate, as is the case with APCM, SPCM and SAPCM). The second variant of SAPCM is called *Layered SAPCM* (L-SAPCM) and works in layers. Specifically, the SAPCM algorithm is initially applied in the whole data set and then it is recursively applied individually on each resulting cluster, in order to reveal possible clustering structure within it, working in a tree structure basis. L-SAPCM terminates when none of the clusters resulting so far has further clustering structure within it. As is verified by the experimental results, L-SAPCM can provide accurate clustering even in cases where the data form closely located clusters at various "resolutions", i.e. the variances of the clusters may differ orders of magnitude from each other.

Also, a considerable contribution of this thesis is the development of an online version of the APCM algorithm, called *Online APCM* (O-APCM), which processes data points one by one and memorizes their impact to suitably defined accumulating variables. O-APCM embodies three new procedures for (a) generating, (b) merging or (c) deleting clusters dynamically and it is a good candidate for clustering of big data sets, whose size and dimensionality are prohibitive for batch algorithms. Finally, it is highlighted that O-APCM may be utilized for applications in both stationary, as well as dynamically varying environments, where the physical clusters may change their location in data space over time. Experimental results show that O-APCM offers high quality clustering results at a very low computational cost.

The potential of the proposed methods is also demonstrated via experimentation on the basis of three case studies, concerning real hyperspectral images (HSIs). The images have been collected from different hyperspectral sensors and depict various land cover cases. The proposed algorithms gave, in general, superior performance compared to other related algorithms.

Finally, a sparsity-aware feature selection technique suitable for HSIs has been developed in the frame of the current thesis. The proposed method is based on the optimization of a sparsity promoting cost-function, in order to identify the bands with the most significant ability in discriminating the various homogeneous regions in the HSI under study. Experimental results on real HSI data have shown remarkable quality of the clustering considering only the selected bands that result from the above technique.

SUBJECT AREA: Pattern recognition

KEYWORDS: possibilistic clustering, parameter adaptation, sparsity, cluster elimination, convergence analysis, online clustering, feature selection, hyperspectral image processing

ΠΕΡΙΛΗΨΗ

Η ομαδοποίηση δεδομένων είναι μια εδραιωμένη μεθοδολογία ανάλυσης δεδομένων στο πλαίσιο της αναγνώρισης προτύπων και έχει χρησιμοποιηθεί εκτενώς σε διάφορα πεδία εφαρμογών κατά τη διάρκεια των τελευταίων δεκαετιών. Δεδομένου ενός συνόλου αντικειμένων, σκοπός της ομαδοποίησης είναι η ταυτοποίηση των ομάδων που αποτελούνται από "όμοια" αντικείμενα. Ένα σημαντικό τμήμα της βιβλιογραφίας στην ομαδοποίηση δεδομένων έχει αφιερωθεί στην αναγνώριση συμπαγών και υπερελλειψοειδούς σχήματος ομάδων και μία από τις πιο γνωστές κατηγορίες αλγορίθμων ομαδοποίησης που χρησιμοποιούνται συνήθως στην βιβλιογραφία για τον σκοπό αυτό, είναι οι αλγόριθμοι που βασίζονται στη βελτιστοποίηση κατάλληλων συναρτήσεων κόστους. Η παρούσα διατριβή εστιάζει στους αλγόριθμους ομαδοποίησης δεδομένων με βάση τα ενδεχόμενα (Possibilistic C-Means, PCM), που είναι από τους πιο γνωστούς της προαναφερθείσας κατηγορίας. Συγκεκριμένα, αναδεικνύοντας πρώτα τις αδυναμίες τους, προτείνονται στη συνέχεια μέθοδοι επέκτασής τους για την αντιμετώπιση των αδυναμιών αυτών. Οι μέθοδοι αυτές βασίζονται κυρίως στην υιοθέτηση των εννοιών της προσαρμοστικότητας παραμέτρων (parameter adaptivity) και της αραιότητας (sparsity). Στη συνέχεια, παρουσιάζονται εν συντομία, τα κύρια σημεία συνεισφοράς της παρούσας διατριβής.

Αρχικά, αναπτύχθηκε μια νέα προσέγγιση στο πλαίσιο των αλγορίθμων PCM, που ονομάζεται *Adaptive Possibilistic C-Means* (APCM). Ο APCM αντιμετωπίζει πολλές από τις αδυναμίες των PCM αλγορίθμων, επιτρέποντας την προσαρμογή ορισμένων παραμέτρων που είναι χαρακτηριστικές για όλους τους αλγορίθμους PCM, κατά την διάρκεια εκτέλεσής του. Αυτό έρχεται σε αντίθεση με τους κλασικούς PCM αλγορίθμους, όπου αυτές οι παράμετροι, από τη στιγμή που ορίζονται αρχικά, δεν αλλάζουν. Η προσαρμογή αυτή των παραμέτρων, δίνει στον APCM δύο νέα χαρακτηριστικά που δεν συναντώνται στους κλασικούς αλγορίθμους PCM. Το πρώτο είναι ότι ο APCM είναι σε θέση, κατ' αρχήν, να προσδιορίσει τον πραγματικό αριθμό των φυσικών ομάδων που σχηματίζονται από τα δεδομένα, υπό την προϋπόθεση ότι ξεκινά με μια λογική υπερεκτίμηση του, αντιμετωπίζοντας έτσι ένα μακροχρόνιο πρόβλημα στη βιβλιογραφία της ομαδοποίησης δεδομένων. Αυτό πραγματοποιείται με την αφαίρεση των ομάδων που σταδιακά καθίστανται παρωχημένες (δηλαδή, ομάδες των οποίων η χαρακτηριστική παράμετρος ελαττώνεται προς το μηδέν, καθώς ο αλγόριθμος εξελίσσεται). Ένα άλλο χαρακτηριστικό της προσαρμογής των χαρακτηριστικών παραμέτρων στον APCM είναι η αύξηση της ευελιξίας του στο να ακολουθεί τις μεταβολές στο σχηματισμό των ομάδων κατά την εκτέλεση του αλγορίθμου. Το γεγονός αυτό καθιστά τον APCM ικανό να αποκαλύψει την υποκείμενη δομή ομαδοποίησης, ακόμα και σε δύσκολες περιπτώσεις, όπου οι φυσικές ομάδες βρίσκονται κοντά η μια στην άλλη και/ή έχουν σημαντικές διαφορές στις διακυμάνσεις τους. Ο APCM συγκρίνεται με αρκετούς σχετικούς αλγορίθμους μέσω εκτεταμένων προσομοιώσεων τόσο σε συνθετικά όσο και σε πραγματικά δεδομένα και τα αποτελέσματα δείχνουν ότι παρουσιάζει καλύτερη απόδοση σε όλες σχεδόν τις περιπτώσεις. Επιπλέον, παρέχονται θεωρητικά αποτελέσματα, τα οποία είναι ενδεικτικά της συμπεριφοράς σύγκλισης του αλγορίθμου.

Στη συνέχεια, εστιάζουμε στην επέκταση του PCM εισάγοντας την έννοια της αραιότητας. Το σκεπτικό πίσω από αυτήν την επέκταση είναι ότι, στην πράξη, ένα σημείο-δεδομένο είναι συμβατό με το πολύ μία, λίγες ή ακόμη και καμία ομάδα (ακραίο σημείο). Έτσι, χρησιμοποιώντας τα σημεία δεδομένων που είναι συμβατά με μια συγκεκριμένη ομάδα και

εξαιρώντας εκείνα που δεν είναι συμβατά με αυτή, οδηγούμαστε σε πιο ακριβείς εκτιμήσεις των παραμέτρων της ομάδας. Ο αλγόριθμος που προκύπτει ονομάζεται *Sparse Possibilistic C-Means* (SPCM) και μπορεί να αντιμετωπίσει καλά περιπτώσεις όπου έχουμε κοντινές ομάδες, οι οποίες μπορεί επίσης να έχουν σημαντική διαφορά στην πυκνότητά τους, ενώ ταυτόχρονα παρουσιάζει ευρωστία σε θόρυβο και ακραία σημεία. Τέλος, δίνεται μια μη τετριμμένη απόδειξη σύγκλισης του αλγορίθμου SPCM. Η κύρια πηγή δυσκολίας στην ανάλυση σύγκλισης, σε σύγκριση με εκείνες που δίνονται για προηγούμενους αλγορίθμους PCM, έγκειται στο γεγονός ότι μια από τις εξισώσεις ενημέρωσης των παραμέτρων του δεν δίνεται σε κλειστή μορφή, αλλά υπολογίζεται μέσω μιας μαθηματικής έκφρασης δύο κλάδων, η οποία ορίζει μια μη-συνεχή απεικόνιση. Στην παρούσα διατριβή, αποδεικνύεται ότι ο SPCM συγκλίνει σε ένα από τα τοπικά ελάχιστα της συνάρτησης κόστους του. Επίσης, αποδεικνύεται ότι παρόμοια αποτελέσματα σύγκλισης μπορεί να προκύψουν και για τον κλασικό αλγόριθμο PCM, αν ο τελευταίος ιδωθεί ως ειδική περίπτωση του SPCM, τα οποία είναι ισχυρότερα από αυτά που παρατίθενται σε προηγούμενες μελέτες.

Στη συνέχεια, τα κυριότερα χαρακτηριστικά των προτεινόμενων αλγορίθμων APCM και SPCM συνδυάζονται, οδηγώντας στον *Sparse Adaptive Possibilistic C-Means* (SAPCM) αλγόριθμο, ο οποίος κληρονομεί όλα τα πλεονεκτήματα των προγόνων του και έχει τη δυνατότητα (α) να αντιμετωπίσει καλά απαιτητικά σύνολα δεδομένων, όπου τα σημεία-δεδομένα σχηματίζουν κοντινές φυσικές ομάδες, με πιθανώς διαφορετικές πυκνότητες ή/και διακυμάνσεις, (β) να προσδιορίσει τον αριθμό των φυσικών ομάδων και (γ) να βελτιώσει ακόμη περισσότερο τις εκτιμήσεις των παραμέτρων των ομάδων, σε σύγκριση με τους APCM και SPCM. Εκτεταμένα πειράματα επαληθεύουν τα συνολικά πλεονεκτήματα του SAPCM σε σύγκριση με άλλους συναφείς αλγορίθμους. Επιπλέον, αναπτύχθηκαν δύο παραλλαγές του SAPCM, οι οποίες χρησιμοποιούν τον SAPCM ως δομικό στοιχείο. Η πρώτη είναι μια επαναληπτική από κάτω-προς-τα-πάνω (bottom-up) εκδοχή, που ονομάζεται *Sequential SAPCM* (SeqSAPCM), όπου σε κάθε επανάληψη καθορίζεται μια νέα ομάδα με τη χρήση του SAPCM. Έτσι, αναδεικνύονται ακολουθιακά μια-μια οι ομάδες που σχηματίζουν τα σημεία-δεδομένα. Το βασικό πλεονέκτημα του SeqSAPCM είναι ότι δεν απαιτεί τη γνώση του αριθμού των φυσικών ομάδων (ούτε καν μια υπερεκτίμηση αυτού, όπως συμβαίνει με τους APCM, SPCM και SAPCM). Η δεύτερη παραλλαγή του SAPCM ονομάζεται *Layered SAPCM* (L-SAPCM) και εργάζεται σε επίπεδα. Συγκεκριμένα, ο αλγόριθμος SAPCM εφαρμόζεται αρχικά σε όλο το σύνολο δεδομένων και στη συνέχεια, εφαρμόζεται χωριστά σε κάθε προκύπτουσα ομάδα, προκειμένου να αποκαλυφθεί πιθανή δομή ομαδοποίησης μέσα σε αυτή. Ο L-SAPCM τερματίζει όταν καμία από τις ομάδες που προέκυψαν μέχρι στιγμής δεν έχει περαιτέρω δομή ομαδοποίησης μέσα της. Όπως επαληθεύεται από τα πειραματικά αποτελέσματα, ο L-SAPCM μπορεί να παρέχει ακριβή ομαδοποίηση, ακόμη και σε περιπτώσεις όπου τα σημεία δεδομένων δημιουργούν πολύ κοντινές ομάδες σε διαφορετικές ”αναλύσεις” (resolutions), δηλαδή ομάδες των οποίων οι διακυμάνσεις διαφέρουν τάξεις μεγέθους μεταξύ τους.

Επίσης, μια σημαντική συμβολή αυτής της διατριβής είναι η ανάπτυξη μιας online έκδοσης του αλγορίθμου APCM, που ονομάζεται *Online APCM* (O-APCM), ο οποίος επεξεργάζεται τα σημεία δεδομένων ένα προς ένα και συσσωρεύει την πληροφορία που περιέχουν σε κατάλληλα ορισμένες μεταβλητές. Επιπλέον, ο O-APCM ενσωματώνει τρεις νέες διαδικασίες για δυναμική (α) δημιουργία, (b) συγχώνευση ή (γ) διαγραφή ομάδων και είναι ένας καλός υποψήφιος αλγόριθμος για ομαδοποίηση μεγάλων συνόλων δεδομένων, των οποί-

ων το μέγεθος και η διάσταση είναι απαγορευτικά για τους αλγόριθμους επεξεργασίας κατά δέσμες. Τέλος, τονίζεται ότι ο O-APCM μπορεί να χρησιμοποιηθεί για εφαρμογές τόσο σε στατικά, όσο και δυναμικά περιβάλλοντα, όπου οι φυσικές ομάδες ενδέχεται να αλλάζουν θέση στο χώρο των δεδομένων με την πάροδο του χρόνου. Πειραματικά αποτελέσματα δείχνουν ότι ο O-APCM προσφέρει υψηλής ποιότητας αποτελέσματα έναντι πολύ χαμηλού υπολογιστικού κόστους.

Η δυναμική των προτεινόμενων μεθόδων αναδεικνύεται επίσης μέσω πειραμάτων με βάση τρεις περιπτώσεις μελέτης, που αφορούν σε πραγματικές υπερφασματικές εικόνες (HSI). Οι εικόνες συλλέχθηκαν από διαφορετικούς υπερφασματικούς αισθητήρες και απεικονίζουν ποικίλες περιπτώσεις κάλυψης γης. Οι προτεινόμενοι αλγόριθμοι είχαν, σε γενικές γραμμές, ανώτερη απόδοση σε σύγκριση με άλλους συναφείς αλγορίθμους.

Τέλος, στα πλαίσια της παρούσας διατριβής αναπτύχθηκε και μια νέα τεχνική επιλογής χαρακτηριστικών βασισμένη στην ιδέα της αραιότητας, κατάλληλη για HSIs. Η προτεινόμενη μέθοδος βασίζεται στη βελτιστοποίηση μιας συνάρτησης κόστους προώθησης αραιής αναπαράστασης, προκειμένου να εντοπιστούν οι μπάντες με την πιο σημαντική ικανότητα στη διάκριση των διαφορετικών ομοιογενών περιοχών της υπό μελέτη HSI. Πειραματικά αποτελέσματα σε πραγματικά δεδομένα HSI έδειξαν αξιοσημείωτη ποιότητα στην ομαδοποίηση που προκύπτει, λαμβάνοντας υπόψη μόνο τις επιλεγμένες μπάντες που προκύπτουν από την ανωτέρω τεχνική.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Αναγνώριση προτύπων

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: ομαδοποίηση με βάση τα ενδεχόμενα, προσαρμογή παραμέτρων, αραιότητα, εξάλειψη ομάδας, ανάλυση σύγκλισης, online ομαδοποίηση, επιλογή χαρακτηριστικών, επεξεργασία υπερφασματικής εικόνας

Dedicated to my inspiring dad, Dimitris,
and the whole supportive family

ACKNOWLEDGEMENTS

I would greatly like to thank all those who contributed in any way during the demanding process of completing this dissertation.

Foremost, I would like to express my sincere gratitude to my supervisors Dr. Konstantinos Koutroumbas and Dr. Athanasios Rontogiannis for their continuous support and guidance throughout the research and writing of my thesis. Their immense knowledge, patience and motivation have inspired me not only in the scientific area, but also on a personal level.

I am also sincerely grateful to Prof. Sergios Theodoridis for his influencing lectures and for providing me the opportunity to pursue a doctoral degree.

My work has also been benefited from my collaboration with Dr. Olga Sykioti and my colleagues Dr. Konstantinos Themelis, Mr. Paris Giambouras and Dr. Eleutheria Mylona, to whom I would also like to give special thanks.

Finally, I would like to thank my family and friends for their support throughout this long road.

LIST OF PUBLICATIONS

Refereed Journal Papers

1. 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.
2. S. D. Xenaki, K. D. Koutroumbas and A. A. Rontogiannis,“Sparsity-aware Possibilistic Clustering Algorithms”, *IEEE Transactions on Fuzzy Systems*, vol. 24, no. 6, pp. 1611-1626, 2016.
3. K. D. Koutroumbas, S. D. Xenaki and A. A. Rontogiannis, “On the Convergence of the Sparse Possibilistic C-Means Algorithm”, *IEEE Transactions on Fuzzy Systems*, 2017, to appear. DOI: 10.1109/TFUZZ.2017.2659739

Refereed Conference Papers

1. S. D. Xenaki, K. D. Koutroumbas and A. A. Rontogiannis, “Adaptive Possibilistic Clustering”, *IEEE International Symposium on Signal Processing and Information Technology*, pp. 422-427, Dec 2013.
2. S. D. Xenaki and K. D. Koutroumbas and A. A. Rontogiannis, “Sparse Adaptive Possibilistic Clustering”, *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 3072-3076, Florence 2014.
3. S. D. Xenaki, K. D. Koutroumbas and A. A. Rontogiannis, “Sequential Sparse Adaptive Possibilistic Clustering”, *Artificial Intelligence: Methods and Applications (SETN)*, pp. 29-42, 2014.
4. S. D. Xenaki, K. D. Koutroumbas and A. A. Rontogiannis and O. A. Sykioti, “A Layered Sparse Adaptive Possibilistic Approach for Hyperspectral Image Clustering”, *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pp. 2890-2893, 2014.
5. S. D. Xenaki, K. D. Koutroumbas and A. A. Rontogiannis and O. A. Sykioti, “A New Sparsity-Aware Feature Selection Method for Hyperspectral Image Clustering”, *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pp. 445-448, 2015.
6. K.D. Koutroumbas, S.D. Xenaki, A.A. Rontogiannis, “Detecting Hyperplane Clusters with Adaptive Possibilistic Clustering”, *In proceedings of the 10th Hellenic Conference on Artificial Intelligence (SETN)*, Thessaloniki, May 2016.
7. 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)*, pp. 2020-2024, 2016.

CONTENTS

PREFACE **33**

1 INTRODUCTION **35**

1.1 An Overview of Clustering 35

1.1.1 Related work 39

1.2 Clustering of Hyperspectral Data 41

1.2.1 Hyperspectral Image Representation 42

1.2.2 HSI Processing 43

1.2.3 Hyperspectral Image Clustering 43

1.3 Thesis Contribution 45

1.4 Outline of the Thesis 47

2 CLUSTERING ALGORITHMS BASED ON A COST OPTIMIZATION FUNCTION **49**

2.1 Introduction 49

2.2 The k-means Algorithm 49

2.3 The Fuzzy C-Means Algorithm 52

2.4 The Possibilistic C-Means Algorithm 56

2.4.1 Possibilistic C-Means Issues and Potential Solutions 59

3 ADAPTIVE POSSIBILISTIC C-MEANS ALGORITHM **61**

3.1 Introduction 61

3.2 Initialization in APCM 62

3.3 Parameter adaptation in APCM 62

3.4 Rationale of the algorithm 65

3.5 Selection of parameter α 68

3.6 Convergence results for APCM 71

3.7 Experimental results 76

3.7.1 Clustering behavior of APCM 76

3.7.2 Comparison of APCM with state-of-the-art clustering algorithms . . 79

3.8 Conclusions 84

4 SPARSE POSSIBILISTIC C-MEANS ALGORITHM 87

4.1 Introduction 87
4.2 Enforcing Sparsity - The Sparse PCM (SPCM) 87
4.3 Initialization in SPCM 90
4.4 Updating of θ_j's and u_{ij}'s in SPCM 90
4.5 Selection of the parameter λ 93
4.6 The SPCM algorithm 94
4.7 Collation of SPCM with PCM 96
4.8 On the convergence of the SPCM 98
4.8.1 Fulfilling Assumption 1 114
4.8.2 On the convergence of the PCM algorithm 115
4.9 Conclusion 116

5 THE SPARSE ADAPTIVE POSSIBILISTIC C-MEANS ALGORITHM AND ITS VARIANTS 117

5.1 Introduction 117
5.2 The Sparse Adaptive PCM (SAPCM) 117
5.3 Experimental results 123
5.4 The Sequential SAPCM (SeqSAPCM) 128
5.4.1 The SeqSAPCM algorithm 129
5.4.2 Experimental Results 131
5.5 The Layered SAPCM (L-SAPCM) 134
5.5.1 The L-SAPCM algorithm 136
5.5.2 Experimental Results 139
5.6 Conclusion 140

6 ONLINE ADAPTIVE POSSIBILISTIC C-MEANS ALGORITHM 143

6.1 Introduction 143
6.2 The Online APCM (O-APCM) 144
6.2.1 Parameter initialization 144
6.2.2 Parameter adaptation - Cluster generation 145

6.2.3 Cluster merging procedure 146
6.3 Experimental results 148
6.3.1 Experiments in a stationary environment 149
6.3.2 Experiments in a non-stationary environment 151
6.4 Conclusion 159

7 CLUSTERING ALGORITHMS APPLIED TO HYPERSPECTRAL IMAGES: A COMPARATIVE STUDY 161
7.1 Introduction 161
7.2 Case Study 1: South Polar Cap 161
7.3 Case Study 2: Salinas Valley 168
7.4 Case Study 3: Washington DC Mall 171

8 A SPARSITY-AWARE FEATURE SELECTION METHOD FOR HYPERSPECTRAL IMAGES 177
8.1 Introduction 177
8.2 The Feature Selection Method 177
8.3 An extension for large HSI data sets 179
8.4 Experimental Results 180
8.5 Conclusion 182

9 CONCLUSION AND FUTURE DIRECTIONS 183
9.1 Summary 183
9.2 Future Directions 185
9.2.1 Further Extensions on the Proposed Algorithms 185
9.2.2 Subspace Possibilistic Clustering 186

ABBREVIATIONS 187

NOTATION 189

APPENDICES 189

A PROOF OF APCM PROPOSITION 191

B PROOFS OF SPCM PROPOSITIONS **193**

C SPCM CONVERGENCE PROPOSITIONS AND PROOFS **195**

BIBLIOGRAPHY **196**

LIST OF FIGURES

1.1 Data sets of different shaped clusters. 37

1.2 Different clustering results of the same 2-dimensional data set, based on different similarity measures. 38

1.3 A HSI cube and the spectral signatures of three pixels of different materials. 41

1.4 Example of hyperspectral data representation in a 3-dimensional space (only three wavelengths are used). 42

2.1 The degree of compatibility u_{ij} with respect to d_{ij}/γ_j, where $d_{ij} = \|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2$, for J_{PCM_1} (for several values of q, solid lines) and J_{PCM_2} (dashed line). . . . 57

3.1 An example of a two dimensional data set consisting of two physical clusters that have big difference in their variances and are located very close to each other. (a) The data set, (b) the initial stage of PCM, (c) the 3rd iteration of PCM, (d) the initial stage of APCM, (e) the 3rd iteration of APCM and (f) the final stage of APCM. The circles are centered at $\boldsymbol{\theta}_j$'s and have radius $\sqrt{\gamma_j}$'s. 66

3.2 The degree of compatibility u_{ij} with respect to distance d_{ij} ($\eta_2 > \eta_1 = \hat{\eta}/\alpha$). 67

3.3 A two dimensional data set consisting of a single physical cluster. APCM is intialized with two representatives and the cluster elimination procedure is illustrated at several iteration steps. 68

3.4 Plot of the APCM cost function with respect to θ_j for a two-class 1-dim data set. (a) The data set. Data points are denoted by stars on the x-axis and representatives by black dots. Results for (b) $m_{ini} = 3$, $\alpha = 0.05$, (c) $m_{ini} = 3$, $\alpha = 1$, (d) $m_{ini} = 3$, $\alpha = 2$, (e) $m_{ini} = 10$, $\alpha = 0.05$, (f) $m_{ini} = 10$, $\alpha = 1$ and (g) $m_{ini} = 10$, $\alpha = 2$. 71

3.5 PCM and APCM snapshots at their initialization step, 1st iteration and 13th iteration for PCM and 10th (final) iteration for APCM (experiment 1). 76

3.6 The clustering results of APCM for experiment 2, when it is initialized with $m_{ini} = 5$, for several values of parameter α. 78

3.8 (a) Data set of experiment 3. Clustering results for (b) k-means, $m_{ini} = 3$, (c) FCM, $m_{ini} = 3$, (d) FCM & XB, (e) PCM, $m_{ini} = 15$, (f) APCM, $m_{ini} = 15$ and $\alpha = 1$, (g) UPC, $m_{ini} = 8$ and $q = 3$, (h) PFCM, $m_{ini} = 15$, $K = 1$, $\alpha = 1$, $\beta = 3$, $q = 2.5$ and $n = 2$, (i) UPFC, $m_{ini} = 15$, $\alpha = 1$, $\beta = 1.5$, $q = 3$ and $n = 2$, (j) GRPCM and (k) AMPCM. 81

4.1 (a) The data set of Example 1 and (b) the clustering result of Example 1 for PCM with $m = 5$ ($m_{final} = 2$). Open (closed) circles stand for the initial (final) location of the representatives ($\boldsymbol{\theta}_j$'s) and crosses represent the true centers of the clusters ($\mathbf{c}_j$'s). The circles centered at each $\boldsymbol{\theta}_j$ and having radius $\sqrt{\gamma_j}$ are also drawn. $\mathbf{x}_s$ is a specific typical point of C_1 that will also be considered in Figs. 4.2 and 4.5 and u_{s2}^{ini} is its corresponding degree of compatibility with $\boldsymbol{\theta}_2^{ini}$. 88

4.2 (a) The data set of Example 2 and (b) the clustering result of Example 2 for PCM with $m = 5$ ($m_{final} = 2$). Note that the contribution of the typical point $\mathbf{x}_s$ to the computation of $\boldsymbol{\theta}_2^{ini}$ is now much smaller compared to its counterpart in Fig. 4.1b. See also the caption of Fig. 4.1. 89

4.3 In all plots the dashed parts of the graphs correspond to the interval $(0, u_{min})$, which is not accessible by the algorithm (see eq. (4.3)). (a) The shape of function $f(u_{ij})$, when $f(\hat{u}_{ij}) < 0$ and the right-most condition of eq. (4.3) is satisfied and (b) the corresponding shape of the cost function $J(u_{ij})$. (c) The shape of function $f(u_{ij})$, when $f(\hat{u}_{ij}) > 0$ and (d) the corresponding shape of $J(u_{ij})$. (e) The shape of function $f(u_{ij})$, when $f(\hat{u}_{ij}) < 0$ and the right-most condition of eq. (4.3) is not satisfied and (f) the corresponding shape of $J(u_{ij})$. 92

4.4 A representative $\boldsymbol{\theta}_j$ is denoted with a black dot. The circles delimit the regions around $\boldsymbol{\theta}_j$ that contain points with $u_{ij} > 0$, for (a) $K > 1$, (b) $K = 1$ and (c) $K < 1$. 93

4.5 The clustering results of SPCM for the data set of (a) Example 1 with $m_{ini} = 5$ and (b) Example 2 with $m_{ini} = 5$. In both cases the contribution of the typical point $\mathbf{x}_s$ to the determination of $\boldsymbol{\theta}_2^{ini}$ becomes zero. See also the caption of Fig. 4.1. 95

4.6 The degree of compatibility u_{ij} as a function of the d_{ij}/γ_j for PCM [10] (red curve), SPCM for several values of $p < 1$ (blue curves) and SPCM for $p = 1$ (pink curve). 96

4.7 (a) PCM and SPCM snapshots at their initialization step, (b), (c) their first iteration, (d), (f) 5th and 8th iteration for PCM and (e) 5th (final) iteration for SPCM. 97

4.8 An active set of $k = 3$ points in cases when (a) $\Theta_I \subset I$ and (b) $\Theta_I \not\subset I$ . . . 102

4.9 Graphical presentation of the continuity of the mapping $\{d_{ij} \rightarrow u_{ij}\}$. Small variations in d_{ij} cause small variations in u_{ij}. 109

4.10 (a) An active set of $k = 3$ points where $(I \cap (\cap_{i:\ u_i=0} ext(\mathcal{C}_i))) \not\equiv I$ and (b) an active set of $k = 4$ points where $(I \cap (\cap_{i:\ u_i=0} ext(\mathcal{C}_i))) \equiv I$ 112

4.11 The upper bound $B(p)$ of K with respect to parameter p for different values of μ_{max}, so that each initial cluster has at least one data point with $u > 0$. . . 115

5.1 Graphical presentation of $f^r(u)$ and $f^s(u)$ for constant d, λ and p, with $\gamma_r > \gamma_s$. The largest of the two solutions of $f^r(u) = 0$ and $f^s(u) = 0$, u_{ir} and u_{is}, are also shown, respectively. It is observed that $u_{ir} \geq u_{is}$. 118

5.2 (a) The data set of Example 1, (b) The data set of Example 2. $\mathbf{x}_s$ is a specific typical point that will also be considered in Figs. 5.3 and 5.4. 122

5.3 The clustering results of Example 1 for (a) APCM, $m_{ini} = 5$ and $\alpha = 1.6$ (b) SAPCM, $m_{ini} = 5$ and $\alpha = 2$. See also the caption of Fig. 5.2. Note that the degree of compatibility of $\mathbf{x}_s$ (defined in Fig. 5.2) with $\boldsymbol{\theta}_2^{ini}$, u_{s2}^{ini}, is positive in APCM and zero in SAPCM. 122

5.4 The clustering results of Example 2 for (a) APCM, $m_{ini} = 5$ and $\alpha = 1.5$ (b) SAPCM, $m_{ini} = 5$ and $\alpha = 1$. See also the caption of Fig. 5.2. In this case, u_{s2}^{ini} is significantly smaller than in Fig. 5.3a. 123

5.5 (a) The data set of Experiment 1. Clustering results for (b) k-means, $m_{ini} = 3$, (c) FCM, $m_{ini} = 3$, (d) PCM, $m_{ini} = 5$, (e) UPC, $m_{ini} = 5$, $q = 1.5$, (f) UPFC, $m_{ini} = 10$, $\alpha = 5$, $\beta = 1$, $q = 2.2$, $n = 3$, (g) PFCM, $m_{ini} = 5$, $K = 1$, $\alpha = 1$, $\beta = 5$, $q = 1.5$, $n = 1.5$, (h) SPCM-L_1, $\lambda = 15$, $q = 2$ (i) APCM, $m_{ini} = 5$, $\alpha = 0.3$, (j) SPCM, $m_{ini} = 5$, and (k) SAPCM, $m_{ini} = 10$ and $\alpha = 0.15$. 125

5.6 (a) The data set of Experiment 2. Clustering results for (b) k-means, $m_{ini} = 3$, (c) FCM, $m_{ini} = 3$, (d) PCM, $m_{ini} = 10$, (e) UPC, $m_{ini} = 5$, $q = 1.5$, (f) UPFC, $m_{ini} = 10$, $\alpha = 5$, $\beta = 1$, $q = 2.5$, $n = 2$, (g) PFCM, $m_{ini} = 5$, $K = 1$, $\alpha = 1$, $\beta = 1$, $q = 1.5$, $n = 1.5$, (h) SPCM-L_1, $\lambda = 17$, $q = 2$ (i) APCM, $m_{ini} = 5$, $\alpha = 0.4$, (j) SPCM, $m_{ini} = 5$, and (k) SAPCM, $m_{ini} = 10$ and $\alpha = 0.18$. 127

5.7 Clustering results of Experiment 1 for (a) k-means, $m_{ini} = 3$, (b) k-means, $m_{ini} = 5$, (c) FCM, $m_{ini} = 3$, (d) FCM, $m_{ini} = 5$, (e) PCM, $m_{ini} = 10$, (f) APCM, $m_{ini} = 5$, $\alpha = 2$, (g) SPCM, $m_{ini} = 3$, (h) SAPCM, $m_{ini} = 5$, $\alpha = 1.5$ and (i) SeqSAPCM, $\alpha = 1.3$. 132

5.8 The data set in Experiment 2. Colors indicate the true label information. . . 134

5.9 (a) Initial data, (b) Data after removing “noisy” points 136

5.10 L-SAPCM flaw example 137

5.11 (a) The data set of the experiment (the area containing the three low variance clusters is magnified in a separate window). Clustering results for (b) k-means, $m_{ini} = 5$, (c) FCM, $m_{ini} = 5$, (d) PCM, $m_{ini} = 5$, (e) APCM, $m_{ini} = 5$, $\alpha = 1$, (f) SPCM, $m_{ini} = 20$, (g) SAPCM, $m_{ini} = 5$, $\alpha = 1$, (h) SeqSAPCM, $\alpha = 1$ and (i) L-SAPCM. 140

6.1 (a) The data set of Experiment 1 (stationary environment). Clustering results for (b) APCM, $m_{ini} = 10$, $\alpha = 0.7$, (c) SAPCM, $m_{ini} = 10$, $\alpha = 1$, (d)-(f) O-APCM at 10000, 20000 and 31200 (final) data points, respectively, $\alpha = 0.7$, $\xi = 1$, (g)-(i) O-kmeans at 10000, 20000 and 31200 (final) data points, respectively, $m = 3$, $z = 0.01$. 150

6.2 Data set of Experiment 2 (non stationary environment). 152

6.3 Clustering results of Experiment 2 (non-stationary environment) at 1000, 3000, 6000, 8000, 12000, 16000, 18000 and 20000 data points, respectively, for (a), (d), (g), (j), (m), (p), (s), (v) O-kmeans, $m = 4$, $z = 0.01$, (b), (e), (h), (k), (n), (q), (t), (w) O-kmeans, $m = 5$, $z = 0.01$ and (c), (f), (i), (l), (o), (r), (u), (x) O-APCM, $\alpha = 0.8$, $\xi = 0.99$ (the older points of each cluster are gray-colored). 154

6.4 (a) Differences between frames 24 and 25 (matrix I_{24} depicted as an image), (b) smoothed differences between frames 24 and 25 (image $I_{24}^{Smoothed}$) and (c) pixels with high (above pre-defined threshold) differences between frames 24 and 25. 155

6.5 (a), (c), (e), (g), (i), (k), (m), (o), (q), (s), (u), (w) Data set of Experiment 3 (non-stationary environment, video) at frame 17, 19, 24, 32, 36, 53, 56, 59, 70, 72, 76 and 81, respectively, and (b), (d), (f), (h), (j), (l), (n), (p), (r), (t), (v), (x) the respective clustering results of O-APCM, $\alpha = 0.05$, $\xi = 0.05$. . . . 158

7.1 (a) 1st PC, (b) 2nd PC, (c) 3rd PC, (d) reference map of the South Polar Cap HSI cube and the corresponding clustering results of (e) k-means ($m = 3$), (f) k-means ($m = 10$), (g) FCM ($m = 3$), (h) FCM ($m = 10$), (i) PCM ($m = 10$), (j) UPC ($m = 3, q = 2$), (k) UPFC ($m = 3$), (l) PFCM ($m = 3$), (m) H2NMF ($m = 3$), (n) KNNCLUST ($K = 1200$), (o) APCM ($m = 3$), (p) SPCM ($m = 10$), (q) SAPCM ($m = 3$), (r) L-SAPCM (1st layer, $m = 3$), (s) L-SAPCM (2nd layer, $m = 3$), (t) O-kmeans ($m = 3$) and (u) O-APCM ($m = 10$). 166

7.2 (a) 1st PC, (b) 4th PC, (c) ground truth of the Salinas Valley HSI cube and the corresponding clustering results of (d) k-means ($m = 7$), (e) FCM ($m = 7$), (f) PCM ($m = 15$), (g) UPC ($m = 15$), (h) UPFC ($m = 15$), (i) PFCM ($m = 15$), (j) H2NMF ($m = 7$), (k) KNNCLUST ($K = 1000$), (l) APCM ($m = 15$), (m) SPCM ($m = 15$), (n) SAPCM ($m = 15$), (o) L-SAPCM (1st layer, $m = 15$), (p) L-SAPCM (2nd layer/final, $m = 15$), (q) O-kmeans ($m = 7$) and (r) O-APCM ($m = 15$). 170

7.3 (a) 1st PC, (b) k-means ($m = 5$), (c) FCM ($m = 5$), (g) FCM ($m = 5$), (d) PCM ($m = 5$), (e) UPC ($m = 5$), (f) UPC ($m = 10$), (g) UPFC ($m = 5$), (h) UPFC ($m = 10$), (i) PFCM ($m = 10$), (j) H2NMF ($m = 5$), (k) KNNCLUST ($K = 1500$), (l) APCM ($m = 10$), (m) SPCM ($m = 10$), (n) SAPCM ($m = 10$), (o) L-SAPCM (1st layer/final, $m = 10$), (p) O-kmeans ($m = 5$), (q) O-APCM ($m = 5$) and (r) O-APCM ($m = 10$). 175

8.1 Graphical illustration of the adopted problem formulation for a toy 3×3 HSI ($N = 9$). 178

8.2 Graphical illustration of the procedure that forms the HSI pixel feature vectors using the spectral bands that correspond to non-zero components of **w**. 179

8.3 An example that an HSI cube is split into $K = 9$ smaller equallised HSIs. . . 180

8.4 The 5 mean cluster signatures of Experiment 1 and the 24 selected bands (marked with red lines). 181

8.5 Results of Experiment 1. (a) The reference classified image. The clustering results obtained from APCM ($m_{ini} = 10$, $\alpha = 1$), when (b) all bands are used (5 clusters), (c) only the 24 selected bands are used (5 clusters). Labels in (a) correspond to classes (Cs) and in (b),(c) to clusters (Cl). 181

8.6 Results of Experiment 2. (a) The reference classified image. The clustering results obtained from APCM ($m_{ini} = 20$, $\alpha = 2$), for (b) all bands (9 clusters), (c) for the 33 selected bands (10 clusters). 182

LIST OF TABLES

3.1 The degrees of compatibility of the data points of experiment 1 for PCM and APCM algorithms, after: (a) initialization (common to both algorithms), (b) first iteration and (c) 13th iteration for PCM and 10th (final) iteration for APCM. 77

3.2 Range of values of the parameter α, in which APCM concludes correctly to $m_{final} = 3$ clusters, for specific values of m_{ini} for experiment 2. 78

3.3 Performance of clustering algorithms for the experiment 3 data set. 82

3.4 Performance of clustering algorithms for the Iris data set. 83

3.5 Performance of clustering algorithms for the New Thyroid data set. 84

4.1 Performance of PCM and SPCM for the data sets of Examples 1 and 2. . . 95

4.2 The degrees of compatibility of the data points of Experiment 1 for PCM and SPCM algorithms, after: (a) first iteration (for both algorithms), (b) 5th iteration for PCM and 5th (final) iteration for SPCM and (c) 8th iteration for PCM. 98

5.1 Performance of APCM and SAPCM for the data sets of Examples 1 and 2. 121

5.2 Performance of clustering algorithms for the data set of Experiment 1. . . . 124

5.3 Performance of clustering algorithms for the data set of Experiment 2. . . . 126

5.4 Performance of clustering algorithms for the Iris data set. 128

5.5 The results of the Experiment 1 synthetic data set 133

5.6 The results of the Experiment 2 synthetic data set 135

5.7 The results of the real Iris data set - Experiment 3 135

5.8 The results of the experiment 141

6.1 Performance of clustering algorithms for the synthetic data set of Experiment 1 (stationary environment). 149

6.2 Data set of Experiment 2 (non stationary environment). 151

6.3 Performance of clustering algorithms for the synthetic data set of Experiment 2 (non-stationary environment). 152

7.1 Clustering results for the South Polar Cap HSI cube 167

7.2 Clustering results for the Salinas Valley HSI cube 171

7.3 Clustering results for the Washington DC Mall HSI cube 176

PREFACE

This research has been conducted at the facilities of the Institute for Astronomy, Astrophysics, Space Applications and Remote Sensing (IAASARS) of the National Observatory of Athens (NOA).

It has been financed in part 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 and in part by the PHySIS project (http://www. physis-project.eu/), contract no. 640174, within the H2020 Framework Program of the European Commission.

1. INTRODUCTION

Pattern recognition is a branch of machine learning whose main task is the search of *patterns* in sets of *objects* with the aim of assigning them into a number of categories or *classes* based on a set of preselected *features*. The most well-known branch of pattern recognition is the *statistical pattern recognition*, which is the framework where the present thesis lie[1]. Here, each object is represented by a set of, say l, measurements-features[2] (this verifies the term "statistical"), which forms its associated *feature vector*, also called *data vector*. Any decision concerning a certain object is based exclusively on its associated feature vector. In most cases, the design of a pattern recognition system is based on a certain set of objects, whose associated feature vectors constitute the so-called *data set*, while the l-dimensional space where all these vectors "live" is called *feature space*.

Traditionally, the two major problems that lie in the frame of pattern recognition are those of *classification* (supervised pattern recognition) and *clustering* (unsupervised pattern recognition). In the first one, a set of known classes is given and the aim is to assign a certain object[3] to one of these classes (according to an optimality rule/criterion). The rule according to which classification is carried out is called *classifier*. In most cases, the design of a classifier is based on a given data set where the class to which each data vector belongs is known. This process is usually called *classifier training* and the data vectors used in this process are called *training vectors*, while the corresponding set is called *training set*. Once the classifier has been properly designed, it can be used to classify objects that have not been used for its training. Also, it should be pointed out that the term "supervised" is justified by the fact that the classes where the training data vectors belong are known.

The second problem in pattern recognition, i.e. clustering (or unsupervised pattern recognition), has a different formulation from that of classification. Specifically, here the only available information is a set of objects without, however, any class knowledge about them (this justifies the term "unsupervised"). The goal is to unravel their underlying similarities, in order to group "similar" objects to the same group (*cluster*) and "dissimilar" objects to different groups, based on a suitable similarity measure. Clustering tasks may arise in many application fields, such as remote sensing, image segmentation, speech and text recognition, etc. Next, we focus only on the clustering task for which a brief overview is given.

1.1 An Overview of Clustering

At the heart of the clustering task is the *clustering algorithm*, that is, the algorithm that groups the data vectors into *clusters*, producing thus a *clustering*, which is simply the set of

[1] Another well-known pattern recognition framework is the structural (or syntactic) pattern recognition ([1], [2]).

[2] Features may be discrete or real-valued. In this work we consider exclusively the real-valued case.

[3] It is assumed that the l features used for the representation of the objects at hand have already been selected and fixed, during the previously applied *feature selection/generation stage*.

these clusters. However, before focusing on the clustering algorithms, some prerequisites are necessary.

According to the way the data vectors are associated with the clusters, we have various types of clustering, the most significant of which are: (a) *hard clustering*, where each data vector belongs exclusively to a single cluster and all pairs of clusters are disjoint, (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.

Before touching the clustering algorithms issue, the following two additional issues need to be firstly resolved. The first one is the *cluster representation* issue. A cluster may be represented either (a) by all of its data vectors, or (b) by a representative. In the second case, the representative is chosen so that to be indicative of the location and the "shape" of the cluster it represents. For example, compact and hyperellipsoidally shaped clusters (see Fig. 1.1a) are usually represented by a single vector that is located at the center of the cluster. Other choices, such as hyperplanes (Figs. 1.1b, 1.1c) or second degree curves (Fig. 1.1d) may also appear, in applications related to image processing. In this work, we consider the case where a single point is used to represent a cluster (that is, we focus on compact and hyperellipsoidally shaped clusters).

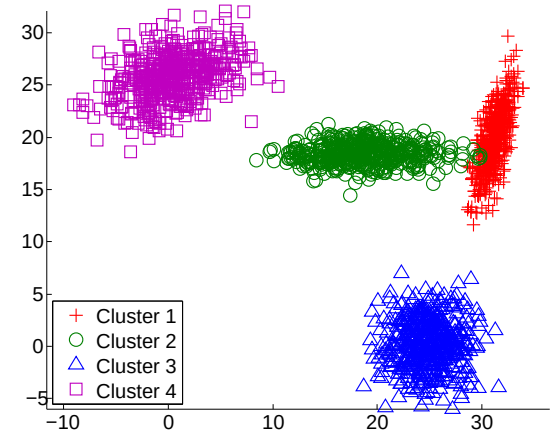
The second issue to be resolved is that of the *proximity measure*. Proximity may be either a *similarity* (the higher its value the closer the compared entities) or a *dissimilarity* measure (the lower its value the closer the compared entities). Depending on the way a cluster is represented, we have proximity measures between (a) two clusters, (b) two data vectors and (c) a data vector and a cluster. Obviously, different choices of similarity measures leads to different clustering results, as shown in Fig. 1.2.

Having resolved the above issues one can proceed now to the heart of the clustering task, the *clustering algorithm*. A vast amount of clustering algorithms has been developed during the past eight decades. Taking into account their underlying rationale, they can be divided into the following main categories[4].

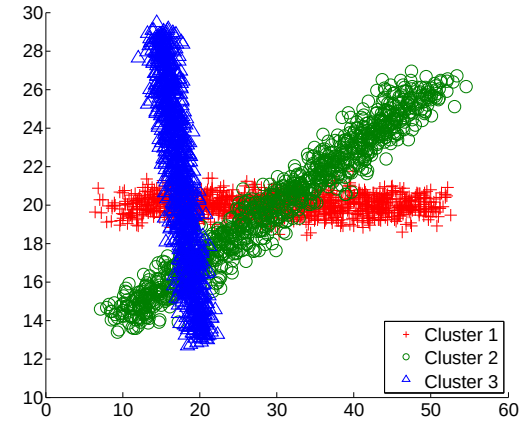
- *Hierarchical algorithms*. These algorithms produce a sequence of *nested clusterings*, where the clustering of each step emanates from that of the previous step either by merging the two most similar clusters into one (*agglomerative* algorithms, bottom-up nesting) or by dividing the cluster with the highest variance into two (*divisive* algorithms, top-down nesting). Typical examples of agglomerative algorithms are the *single link* and *complete link* algorithms ([3]), the *Ward*'s algorithm [4], which are differentiated from each other in the definition of the proximity between two clusters. For example, in single link the proximity between two clusters is defined based on the two "most similar" data vectors of them, while in complete link the respective proximity is based on the two "most dissimilar" data vectors of the clusters.

  As far as the divisive algorithms are concerned, they are discriminated to *polythetic*,

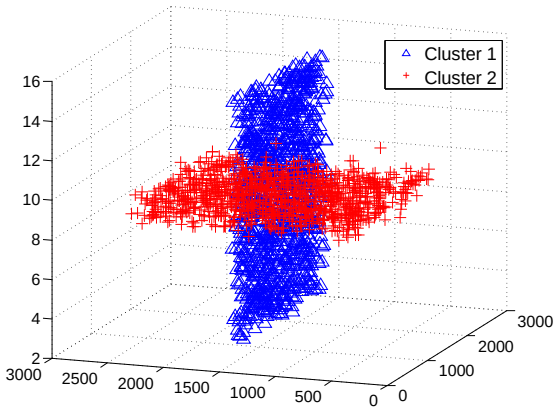
[4]Note that the categorization is not strict, since several (usually more sophisticated) algorithms may exhibit characteristics of more than one category.



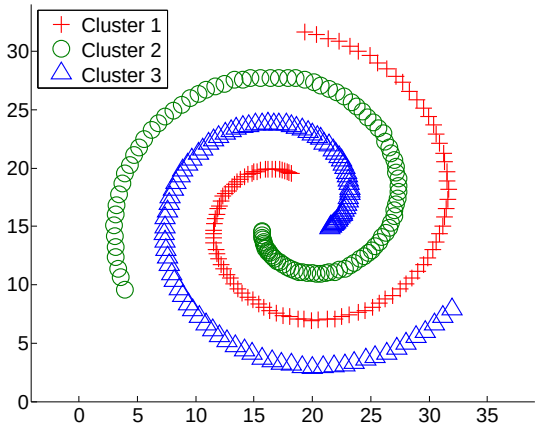
(a) Compact and hyperellipsoidally shaped clusters



(b) Compact and 1-dimensional hyperplaned shaped clusters



(c) Compact and 2-dimensional hyperplaned shaped clusters



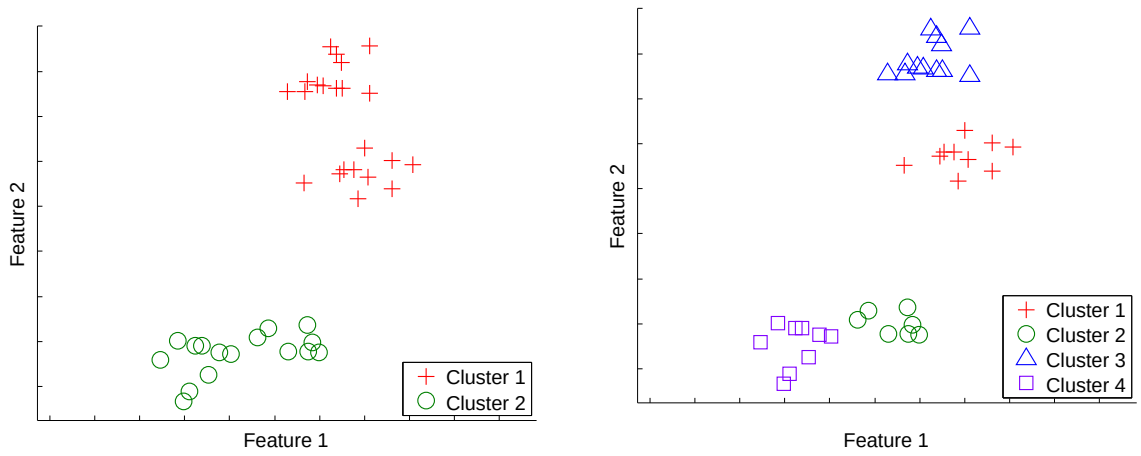
(d) Spiral shaped clusters

Figure 1.1: Data sets of different shaped clusters.

where all features contribute to the selection of the cluster that will be split (and how it will be split) and *monothetic*, where single features are used, in order to determine which cluster will be split. It is noted that due to the very high computational requirements for detecting the most appropriate cluster to split (since it is needed to consider at each clustering, each cluster separately and for each cluster all possible splits), almost all of these algorithms apply heuristic techniques at the cost of providing suboptimal solutions.

In addition, there are algorithms, e.g. *Chameleon* [5] that model data dynamically. Specifically, *Chameleon* combines elements from both agglomerative and divisive concepts, as it merges and divides clusters dynamically.

- *Cost function optimization based algorithms*. These algorithms iteratively optimize suitably defined cost functions, in order to produce “sensible” clusterings. They terminate when a local optimum of the associated cost function is reached. Celebrated



(a) Two clusters **(b) Four clusters**

Figure 1.2: Different clustering results of the same 2-dimensional data set, based on different similarity measures.

algorithms of this type are the *k-means* (hard clustering), e.g. [6], the *fuzzy c-means* (FCM - fuzzy clustering), e.g. [7], [8] and the *possibilistic c-means* (PCM - possibilistic clustering), e.g. [9], [10]. Since all algorithms developed in the present thesis fall into this category, this category is analysed in more detail in the next chapter.

- *Density-based algorithms*. The density-based clustering algorithms use a local cluster criterion, according to which clusters are defined as regions in the data space where the objects are densely located, and clusters are separated from each other by low-density regions. DBSCAN [11], a well-known representative of this category, uses density-based notions to define clusters. More specifically, it identifies via suitably chosen criteria each data point as either a “core” point or a “border” point of a cluster. DBSCAN visits points in a random manner and if a point is a core point, it tries to expand and form a cluster around it. Other algorithms in this category are the DBCLASD [12], DENCLUE [13], while an additional one is given in [14].

- *Subspace-based algorithms*. In certain applications (e.g. in biology), the data vectors are aggregated along subspaces of a (usually high dimensional) feature space. Subspace clustering aims at finding a multi-subspace representation that best fits a collection of data points taken from a high dimensional space. In other words, subspace clustering could be thought as an extension of feature selection that attempts to find clusters in different subspaces of the same dataset. Thus, the problem here is not only to identify the clusters themselves, as is the case with the algorithms of the previous categories, but also the subspaces where these clusters live. In this category, there are two prevailing algorithms. First, CLIQUE [15], which was one of the first subspace-based algorithms, creates a histogram for each dimension and selects those bins with densities[5] above a given threshold. Candidate subspaces

[5]The measure of density is based on the number of data points, whose projection in the relative dimension

in two dimensions can then be formed using only those dimensions which contain dense units, dramatically reducing thus the search space. The algorithm proceeds until there are no more dense units found. Adjacent dense units are then combined to form clusters. Secondly, the recently proposed *Sparse Subspace Clustering* (SSC) algorithm [16] (a) finds a sparse representation of each data point in the dictionary of the other data points, (b) builds a similarity graph using the sparse coefficients, and (c) obtains the segmentation of the data into clusters using spectral clustering. The algorithm, under appropriate conditions, is expected to recover the desired sparse representations of data points.

- *Combination-based algorithms*. Clustering combination relies on the idea that evidence gathered by a number of clusterings can be combined to produce a final (hopefully) more accurate result. Clustering combination methods receive as input an ensemble of clusterings that may have been generated either by applying different clustering algorithms or by applying the same clustering algorithm with different values of parameters or initializations on the data set under study. Furthermore, combinations of different data representations (feature spaces) and clustering algorithms can also provide a multitude of significantly different data clusterings. A simple framework for extracting a consistent clustering, given the various partitions in a clustering ensemble, is proposed in [17].

It is noted that the selection of the appropriate algorithmic scheme for clustering is adjusted according to the application under study, while the interpretation of the results is carried out by experts in the specific application domain.

1.1.1 Related work

Speaking more precisely, in this work we consider only the case of compact and hyperellipsoidally shaped clusters, represented by a single vector. Moreover, we focus on the possibilistic clustering alternative. In the sequel, a brief summary of the relative bibliography is given.

A great amount of work reported in the clustering literature has been devoted to the identification of compact and hyperellipsoidally shaped clusters as those shown in Fig. 1.1a. As stated before, each such cluster is represented by a vector called *cluster representative* or simply *representative*, which lies in the same l-dimensional space with the data and (ideally) is located at the *center* of the cluster.

The most well-known algorithms that deal with this problem, belong to the family of cost optimization clustering algorithms and are the *k-means* (hard clustering), e.g. [6], the *fuzzy c-means* (FCM - fuzzy clustering), e.g. [7], [8] and the *possibilistic c-means* (PCM - possibilistic clustering), e.g. [9], [10], [18], [19], [20], [21]. The main goal of all these algorithms is to move iteratively the representatives towards the centers of the regions that

lies in the bin.

are *dense in data points* (dense regions), that is, to regions where significant aggregations of data points (clusters) exist. Under this perspective, we say that each such vector *represents* a cluster, while, as mentioned previously, their movement towards the centers of the clusters is carried out via the minimization of appropriately defined cost functions.

Let us consider first the k-means and FCM, which share some significant features. First of all, they both require prior knowledge of the exact number of clusters m underlying in the data set (which, of course, is rarely known in practice). In addition, in both schemes the updating equations of the representatives are interrelated. As a result, these algorithms *impose* a specific clustering structure on the data set (rather than uncovering the underlying one), in the sense that they will return m clusters *irrespective* of the actual number of physical clusters existing in the data set. Specifically, if m is less than the actual number of clusters, at least some representatives will fail to move to dense regions, while in the opposite case, some naturally formed clusters will split into more than one pieces[6]. A common method for estimating m is via the use of suitable validity indices (e.g., [22], [20]). Finally, as shown in [18], [19], k-means and FCM are vulnerable to noisy data and outliers.

As far as the PCM algorithms are concerned, 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 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, 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 [23], [24], which, however, can be faced after the termination of the algorithm by seeking for (almost) coincident representatives. In addition, PCM deals well with noisy data points and outliers, compared to k-means and FCM. However, it involves additional parameters, usually denoted by γ. Each of these parameters is associated with a single cluster, while their accurate estimation is of crucial importance. Since, once they have been estimated they are kept fixed during the execution of the PCM algorithm, it is clear that poor initial estimates are likely to lead to poor clustering performance, especially in more demanding data sets (e.g. where clusters with significantly different variances are encountered in the data set).

Many variants of PCM have been proposed to deal with the weaknesses associated with the parameters γ. More specifically, [25] tries to avoid coincident clusters by introducing mutual repulsion of the clusters, so that they are forced away from each other. The same problem is tackled in [23], [26] and [19] by combining possibilistic and fuzzy arguments. Also, in [27] a strategy is proposed that intoduces a “gray zone” around each representative, which contains the points around the cluster boundary. The latter overcomes the coincident clusters problem, is robust to outliers and uses less ad hoc defined parameters than PCM. Another algorithm that involves very few parameters and is robust to noise and outliers is described in [20]. In [28] ideas from [20] and [19] are combined for dealing

[6] Of course, if the value of m corresponds to the actual number of physical clusters, the algorithms have the ability to recover the physical clusters; that is, in this case “imposition” coincides with “uncovering”.

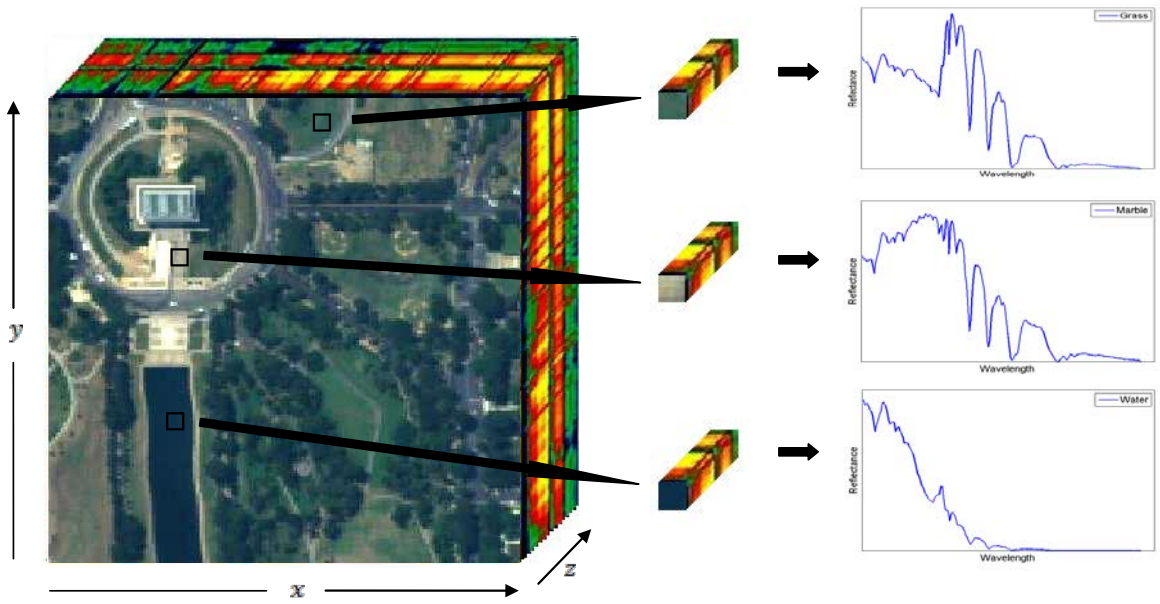


Figure 1.3: A HSI cube and the spectral signatures of three pixels of different materials.

additionaly with the coincident clusters issue. The same issues are also addressed in [21] using, however, a different approach than [28].

The original versions of PCM algorithms are not equipped with a cluster elimination mechanism, that is, if they are initialized with an overestimated number of clusters, they cannot eliminate any of them as they evolve. Inspired by [29], PCM-type algorithms that perform cluster elimination during their execution are described in [30] and [31]. In these algorithms the parameters γ are considered equal for all clusters and are kept fixed as they evolve. Consequently, their ability to deal with closely located clusters with significantly different variances is drastically decreased. In addition, their computational complexity is dramatically increased [30].

1.2 Clustering of Hyperspectral Data

Hyperspectral image (HSI) analysis has attracted considerable attention in the signal processing and pattern recognition literature during the last two decades and is widely recognized as a valuable tool in diverse applications, such as remote sensing, biomedical imaging and astronomy to name but a few. In this work, we consider HSI in the frame of remote sensing, which refers to information acquisition for an object, phenomenon or geographical area without any physical contact with it. Therefore, in this framework, hyperspectral imagery deals with the process of extracting information about an object or geographical area using hyperspectral sensors.

HSIs are acquired by specific hyperspectral sensors, which sample certain regions (bands) of the electromagnetic spectrum. The majority of them sample the electromagnetic spec-

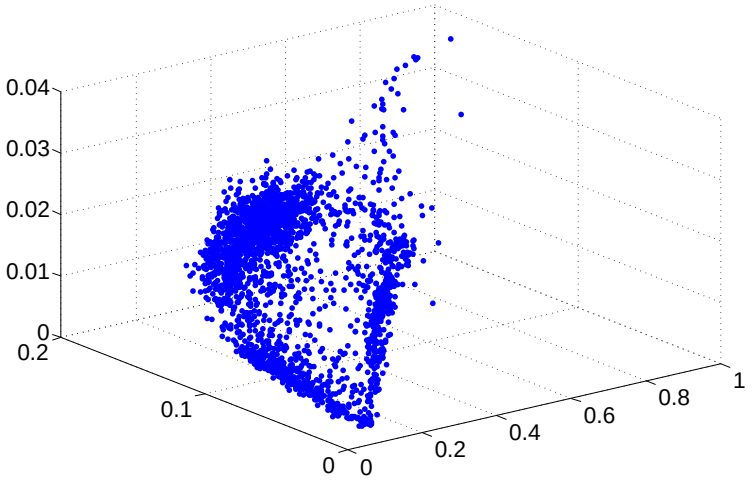


Figure 1.4: Example of hyperspectral data representation in a 3-dimensional space (only three wavelengths are used).

trum in hundreds of contiguous spectral bands for a specific scene [32]; from the visible region (0.4 to 0.7 μm) through the SWIR (Short-Wave InfraRed) (up to 2.5 μm) bands. The spectral sampling results to a set of different (yet related) images of the scene under study, which, stacked together, form the so-called *HSI cube* (see Fig. 1.3) (in the sequel by the term HSI, we mean the HSI cube). As a consequence, the structuring elements (*pixels*) of an HSI are represented by vectors containing the measurements of reflectance in all spectral bands.

Examples of hyperspectral sensors include the Hyperion, a VNIR/SWIR (Visible Near InfraRed, VNIR) hyperspectral sensor with 220 spectral bands aboard the NASA EO-1 satellite, the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) of NASA with 224 contiguous spectral channels, the HYperspectral Digital Imagery Collection Experiment sensor (HYDICE) that collects data in 210 spectral bands and OMEGA (Observatoire pour la Minéralogie, l'Eau, les Glaces et l'Activité) spectrometer that is onboard ESA's Mars Express satellite and measures the solar radiation reflected by the surface of planet Mars.

1.2.1 Hyperspectral Image Representation

As has already been stated, hyperspectral data may be viewed as a sequence of two-dimensional images produced by sampling at specific (usually contiguous) wavelengths, thus they are frequently called hyperspectral image *cubes*. An image cube contains spatial information on the x and y axes and spectral information on the z axis, as shown in Fig. 1.3. A hyperspectral data cube may be also viewed as a matrix of individual pixels conveying spectral information. Thus, isolating a single pixel, its spectral information can be plotted as shown in Fig. 1.3 and it is called *spectral signature* of the pixel.

An alternative representation of hyperspectral data could be obtained by considering the

following. Each HSI pixel can be represented by a vector of length l, whose entries are the measurements of radiance at the corresponding wavelengths and l is the total number of spectral bands. Associating an axis of the $\Re^l$ space with each wavelength, each measurement of a pixel is a coordinate in the corresponding axis. From this perspective, each HSI pixel corresponds to a data point in the l-dimensional space. To get an idea of how the pixels of a HSI are depicted in the l-dimensional space, we consider the (unrealistic) case of Fig. 1.4 ,where we assume that the number of available spectral bands is $l = 3$.

1.2.2 HSI Processing

Hyperspectral data provide very detailed and accurate spectral information of the observed object or scene not only because of the large number of the involved spectral bands, but also due to the continuous nature of measurements. Taking into consideration that radiance reflection and absorption are differentiated among types of materials and spectral channels, HSI processing and analysis provides the means of determining and identifying different materials in a remote scene, based on spectral information. This can be achieved, for example, by matching the scene reflectance spectra to a library of known spectra. However, designing and implementing a spectral imaging system requires many practical issues to be addressed. These issues include the specification of the spatial and spectral resolution of the sensor, the identification of the atmospheric effects such as absorption and scattering, the spectral variability of surface materials in the scene and other environmental effects such as viewing angle, secondary illumination and shadowing [33].

Many different types of hyperspectral imaging applications exist that have been exploiting hyperspectral imagery capabilities. These may be separated into three major categories: *anomaly detection*, *target recognition* and *background characterization*. Anomaly detection is the process of identifying and locating uncommon features in an image, whereas target detection is distinguished from anomaly detection by the availability of some a priori information about the target. Finally, background characterization emphasizes mainly on a background scene analysis and identification and it refers to the domains of land, ocean and atmosphere.

In the next subsection we focus on the processing of HSIs using clustering methods.

1.2.3 Hyperspectral Image Clustering

Hyperspectral image classification is a challenging task that is gaining increasing interest in the remote sensing literature lately [34]. One of the main issues that frequently arise in the processing of HSIs is the partial or total lack of ground truth information. A rational way to deal with this issue is to resort to unsupervised classification (clustering). Under the clustering perspective the data vectors are the l-dimensional spetral signatures of the image pixels. The goal now is to identify spectrally homogeneous regions in HSIs. In particular, the objective of HSI clustering is to assign (a) to the same cluster pixels that are more similar to each other and (b) to different clusters pixels that are less similar to each

other. HSI clustering may find application at mineral exploration, precision argiculture, disaster monitoring, etc. [32], [33].

As mentioned previously, modern hyperspectral sensors are able to acquire measurements in numerous, almost contiguous spectral channels [32]. Although there are several available measurements for each HSI pixel, which, in principle, may be a desirable fact, this may also become a source of problems. This is due to (a) the high spectral correlation (mainly) between adjacent channels, as well as to the noise contained in them ([35]), which is likely to lead to the formation of overlapping clusters and (b) the high dimensionality and (usually) very large size of HSI data, which increase dramatically both computation and memory requirements. Therefore, applying clustering to HSIs becomes a very challenging task.

In recent years, a major effort in the remote sensing literature has been devoted to the design and application of clustering techniques in HSIs, in order to identify spectrally homogeneous regions, which correspond to certain types of land cover. The aim here is to partition a given image into groups such that pixels in the same group are similar to each other and correspond to a specific type of land cover. Several algorithms have been devised for remote sensing image clustering, however, most of them require knowledge of the true number of clusters m underlying in the data set, which is rarely known in practice. Thus, a major issue in HSI clustering is the accurate estimation of m.

In [36], the Gauss Mixture Vector Quantization (GMVQ) algorithm [38] is considered using a correlation-based distance between a data vector and a cluster. The method is assessed to pick up most of the variability in the dataset. Several runs are performed and the optimum number of clusters is chosen to be the one associated with the minimum value for the adopted objective function. In [38], a method for classifying HSIs based on Fuzzy C-Means (FCM) and Markov Random Fields (MRF) is presented. Firstly, the FCM algorithm is used to generate an ensemble of partitions and then, the MRF method is employed to fuse the obtained partitions by taking into account the spatial and inter-partition contextual information. Besides the need for prior knowledge of the number of clusters, a second limitation of this method is that it does not work well at the borders of the classes. In [39], a hyperspectral image clustering method is proposed, based on the Artificial Bee Colony (ABC) algorithm, which is a bionics random optimization algorithm that simulates the self-organizing behavior and intelligence of bee swarms. An extension of this work is reported in [40], where it is combined with the MRF classification discriminant function (ABC-MRF method) and maintains the advantages of the ABC algorithm in finding the cluster center. In addition, it effectively utilized the spatial information contained in the pixels to further improve the accuracy of the cluster. However, a major disadvantage of the last two algorithms is that they consume a considerable amount of computing time to find an optimal solution. Bandyopadhyay *et al.* [41], utilize a multiobjective optimization algorithm (NSGA-II [42]) to determine the appropriate cluster centers and the corresponding partition matrix, where a number of fuzzy cluster validity indexes are simultaneously optimized. A HSI clustering procedure, which is based upon the Fully Constrained Least Squares (FCLS) spectral unmixing method, is described in [43]. This method consists of three steps, namely endmember extraction, unmixing and hardening clustering via the

winner-takes-all approach. Experimental results on a known real HSI data set substantiate the validity of the method. In [44], the FCM algorithm is applied using similarity measures such as the spectral angle and correlation, which are less sensitive to brightness variations. Tran *et al.* [45] presented a density-based clustering algorithm, called KNNCLUST, which combines nonparametric k-nearest-neighbour (kNN) and kernel (kNN-kernel) density estimation. The kNN-kernel density estimation technique makes it possible to model clusters of different densities in high-dimensional data sets. A major advantage of the algorithm is that it is expected to identify automatically the number of the clusters. However, KNNCLUST has very high computational complexity, mainly due to the calculation of the kNN distance matrix. Cariou and Chehdi [46] proposed KSEM that is an extension of the KNNCLUST method inspired from the Stochastic Expectation-Maximization (SEM) algorithm [47]. KSEM is based on iteratively sampling label states via pseudo-posterior label distributions estimated at the local level of the objects and attempts to address the problem of the unknown number of clusters in HSIs. Moreover, in [48], a density-based approach that tries to automatically estimate the number of clusters in hyperspectral imagery, is proposed. Furthermore, several hierarchical clustering approaches have been proposed to deal with HSIs ([49], [50], [51]). Recently, a sparse subspace clustering approach has been applied in HSI data that simultaneously explores the nonlinear structure and the inherent spectral-spatial attributes of HSIs [52].

1.3 Thesis Contribution

The scientific contributions of the present thesis are mainly related, but not restricted to the area of possibilistic clustering and its application in the identification of homogeneous regions in HSIs. The first contribution of the thesis is the development of a novel Possibilistic C-Means (PCM) clustering scheme, called *Adaptive Possibilistic C-Means* (APCM) algorithm, [53], [54]. The main feature of the proposed algorithm is that its parameters γ, after their initialization, are properly adapted during its execution. Provided that the algorithm starts with a reasonable overestimate of the number of physical clusters formed by the data, APCM is capable to unravel them, overcoming a long-standing issue in the clustering literature. Due to the fully adaptive nature of the proposed algorithm, a cluster elimination procedure is enabled, which makes possible the reduction of the initially estimated number of clusters. In addition, the adjustment of the parameters increases the flexibility of the algorithm in following the variations in the formation of the clusters that occur from iteration to iteration. Theoretical results that are indicative of the convergence behavior of APCM are also provided. Finally, extensive simulation results on both synthetic and real data highlight the effectiveness of the proposed algorithm.

The second contribution of the thesis concerns the exploitation of sparsity in the clustering framework. To this end, two novel sparsity-promoting possibilistic clustering algorithms are proposed [55], [56]. The main idea here is that a data point may be compatible with one or only a few (or even none) clusters. The first algorithm, termed *Sparse Possibilistic C-Means* (SPCM), leverages the sparsity of the degree of compatibility vectors and exhibits increased immunity to data points that may be considered as noise or outliers,

by not allowing them to contribute to the estimation of the cluster representatives. Thus, SPCM concludes to more accurate estimates for the cluster representatives, especially in noisy environments. Additionally, sparsity exploitation makes SPCM able to deal well with closely located clusters that may also be of different densities. A rigorous convergence analysis of SPCM is also provided [57] and it is shown that SPCM convergences to one of the local minima of its associated cost function. The main source of difficulty in this analysis, compared to those given for previous possibilistic algorithms, is due to the lack of continuity resulting from the updating rule of certain parameters involved in the algorithm that must be suitably handled.

The second algorithm, called *Sparse Adaptive Possibilistic C-Means* (SAPCM), is an extension of the first, where now the involved parameters are dynamically adapted. SAPCM can deal well with even more challenging situations, where, in addition to the above, clusters may be of significantly different variances and/or densities. More specifically, it provides improved estimates of the cluster representatives and has the additional ability to estimate the actual number of clusters, given an overestimate of it.

Extending SAPCM, two variants of it that use the SAPCM as a structuring element, have been devised. The first one is an iterative bottom-up version, called *Sequential Sparse Adaptive Possibilistic C-Means* (SeqSAPCM) and presented in [58]. Here, at each iteration, SeqSAPCM determines a single new cluster by employing the SAPCM. That way it finally unravels sequentially the underlying clustering structure. The second version of SAPCM, which is accustomed to HSI processing, is *Layered Sparse Adaptive Possibilistic C-Means* (L-SAPCM) and is presented in [59]. L-SAPCM works in layers where at each layer, after suitable pre-processing, the SAPCM algorithm is applied on a tree structure basis. More specifically, initially, SAPCM is applied on the whole HSI data set producing some subsets (sub-clusters) that constitute the first processed layer. Then, for each identified sub-cluster of the first layer, SAPCM is recursively applied to reveal possible sub-clusters within it. The procedure terminates when at each sub-cluster no more than one cluster can be identified.

Another contribution of this thesis is the development of an efficient online clustering algorithm, called *Online Adaptive Possibilistic C-Means* (O-APCM), recently presented in [61]. The algorithm is an online version of the APCM [54], in which data vectors (pixels) are being processed one by one and their impact is memorized to suitably defined parameters; thus O-APCM is released from the noose of storing the whole data (hyperspectral data cubes in HSI processing) and using it at each iteration, as is the case with the batch schemes. Due to its online nature, O-APCM is much more computationally efficient with lower memory requirements without (rather surprisingly) any degradation of the quality of the resulting clustering, compared to its batch ancestor, in which the whole data cube is considered at each iteration of the algorithm. From this perspective, O-APCM is suitable for HSI clustering. The basic advantage that O-APCM inherits from APCM is the ability to adapt the involved parameters during its execution, in order to track variations during the clustering formation. In addition, it embodies new procedures for creating new clusters or merging existing ones, as data points are clustered sequentially. Experimental results show that O-APCM offers high discrimination ability at a very low computational cost and

it compares favourably with other online related algorithms, as well as APCM. In addition, it should be noted that the usage of O-APCM is twofold. It is not only recommended in cases where the environment in which the data live is stationary, where it alleviates a computationally demanding processing of a huge amount of (static) data, but also it is appropriate for data clustering under non-stationary environment conditions; that is, in real time application cases where data may received in a streaming fashion and their statistics vary with time. Such data are real-time financial stock market data, video surveillance data, social media data, etc [60].

The performance of the above proposed clustering algorithms has been assessed in terms of three HSI case studies. Specifically, the first case study is related to the clustering of data acquired by the 256-bands OMEGA hyperspectral sensor, depicting the South Polar Cap of Mars, which is characterized by spatial resolution of $3km$ and spectral resolution 0.93 to 2.98μm. The size of the HSI is 871$\times$128 pixels and the 256 bands are reduced to 186 after the removal of the noisy ones. The pixels of this image stem from three classes: "CO_2" ice, "Water" ice and "Dust". The second case study is captured by the 224-band AVIRIS hyperspectral sensor over Salinas Valley, California, which is characterized by high spatial resolution of 3.7m and spectral resolution 0.2 to 2.4μm. The area covered comprises 512 lines by 217 samples (512 rows $\times$ 217 columns), while 20 water absorption bands (108-112, 154-167, 224) are discarded. This image includes cultures of vegetables, bare soils, and vineyard fields with its corresponding reference map containing 16 classes. The data processed in the last case study came from the 210-band HYDICE hyperspectral sensor over the Washington DC Mall area. The sensor response is in the 0.4 to 2.4 μm region of the visible and infrared spectrum. Bands in the 0.9 and 1.4 μm region, where the atmosphere is opaque, have been omitted from the data set, leaving 191 bands. The data set contains 150 lines by 150 samples (150 rows $\times$ 150 columns) and it has a spatial resolution of approximately 1m.

Finally, considerable research effort was also invested into the development of a sparsity-aware feature selection method for hyperspectral data clustering, which was presented in [62]. The proposed method selects bands that exhibit significant discrimination ability, based on the optimization of a sparsity promoting cost function. Clustering algorithms that use only the selected spectral bands export results of the same quality compared to the case where all spectral bands are used, while, in some cases, they are able to unravel patterns that are not identified using the whole bands information.

1.4 Outline of the Thesis

The remaining chapters of the thesis are organized as follows.

In Chapter 2, an overview of the most well-known clustering algorithms based on cost function optimization is given. In particular, the K-means, the Fuzzy C-Means (FCM) and the Possibilistic C-Means (PCM) algorithms are described and discussed in detail. Also, previous attempts for dealing with their shortcomings are briefly presented.

In Chapter 3, the novel Adaptive Possibilistic C-Means (APCM) clustering algorithm is presented. Indicative theoretical convergence results of APCM are provided. Finally, the performance of APCM is tested against several state-of-the-art algorithms.

In Chapter 4, the Sparse Possibilistic C-Means (SPCM) clustering algorithm is introduced, accompanied with its convergence proof. Also, extensive experiments are performed with synthetic and real data.

In Chapter 5, the Sparse Adaptive Possibilistic C-Means (SAPCM) clustering algorithm is presented and its properties are analysed. Moreover, two new clustering variations, namely, Sequential Sparse Adaptive Possibilistic C-Means (SeqSAPCM) and Layered Sparse Adaptive Possibilistic C-Means (L-SAPCM), that incorporate SAPCM as a building block, are described.

In Chapter 6, the Online Adaptive Possibilistic C-Means (O-APCM) clustering algorithm that is an online implementation of APCM, is presented in detail and its usage in both static and dynamic environments is discussed. Additionally, experimental results verifying that O-APCM offers high discrimination ability at a very low computational cost, are also provided.

Chapter 7 provides experimental results of the proposed methods discussed so far when they apply on real hyperspectral images. Moreover, their results are compared against those obtained by several state-of-the-art related algorithms.

Chapter 8 departs from the thematic area of clustering algorithms and presents a novel sparsity-aware feature selection method for hyperspectral data clustering, whose performance is tested on real data sets.

Finally, Chapter 9 gives a summary and the conclusions of the research work presented in the context of the thesis and highlights some possible future research directions.

2. CLUSTERING ALGORITHMS BASED ON A COST OPTIMIZATION FUNCTION

2.1 Introduction

As it has already been announced, in the present thesis we focus on possibilistic clustering algorithms, which lie in the general framework of the cost function optimization clustering techniques. For reasons of thoroughness, we devote the present chapter to the description of the main features of the algorithms of this category via the presentation of well-known clustering algorithms that follow from three different interpretations of the concept of cluster. In the algorithms of this category, linear varieties (e.g. points, lines, planes, etc.) or quadrics are usually employed for the representation of the clusters, with the actual type of representative to be adopted, being strongly dependent on the types of natural clusters formed by the points of the data set X under study. For example, if compact and hyperellipsoidally shaped clusters are expected, the representatives may be points lying in the space where the points of X lie (see Fig. 1.1a), while, if linear clusters are expected, the representatives may be hyperplanes in the space where the data live. Besides the representatives, some additional parameters are also involved in these algorithms that may be linked with other quantities that describe the clusters formation, such as their spread.

In this chapter, we focus on cost function optimization algorithms that stem from three major philosophies: the hard, the fuzzy and the possibilistic philosophy. In the hard clustering approach each vector belongs exclusively to a single cluster. The fuzzy approach may be viewed as a generalization of the hard clustering approach, in the sense that each vector is *shared* among more than one clusters, up to a certain degree (*grade of membership*). However, the grades of membership of a certain vector with the various clusters are interrelated. In contrast, in possibilistic clustering the so-called *degree of compatibility* of a vector with a certain cluster depends exclusively on the vector and the cluster; that is, it is independent of the degrees of compatibility of this vector with the remaining clusters.

Since in the framework of this thesis we consider compact and hyperellipsoidally shaped clusters, we focus in the sequel on cost function optimization algorithms where the clusters are represented by points in the data vectors space. In the following, we present the most famous members of hard, fuzzy and possibilistic cost function optimization algorithms with point representatives (k-means, FCM and PCMs, respectively).

2.2 The k-means Algorithm

The most celebrated hard clustering algorithm is the k-means algorithm [6]. Here, a point representative is used to represent each cluster and the squared Euclidean distance is adopted to measure the dissimilarity between data vectors and cluster representatives. Initializing the representatives from (usually) random positions, the aim here is to move

gradually each representative to the center of regions where the data points form aggregations (i.e., physical clusters), provided that the actual number of clusters is known a-priori.

Speaking in mathematical terms, let $X = \{\mathbf{x}_i \in \Re^l, i = 1, \ldots, N\}$ be a set of N l-dimensional data vectors to be clustered and $\Theta = \{\boldsymbol{\theta}_j \in \Re^l, j = 1, \ldots, m\}$ be a set of m l-dimensional vectors that will be used as *representatives* of the clusters formed by the points in X (it is $m \leq N$). 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 *membership coefficient* of $\mathbf{x}_i$ with the jth cluster, denoted by C_j and represented by the vector $\boldsymbol{\theta}_j$. Specifically, $u_{ij} = 1(0)$, if $\mathbf{x}_i$ belongs (does not belong) to C_j. Finally, let $\mathbf{u}_i^T = [u_{i1}, \ldots, u_{im}]$ be the ith row of U that contains the membership coefficients of $\mathbf{x}_i$ for all the clusters. In what follows, we consider only Euclidean norms, denoted by $\|\cdot\|$.

As already mentioned, in the k-means algorithm the membership coefficients u_{ij} are either 0 or 1. In addition, if for a data vector $\mathbf{x}_i$ it is $u_{ij} = 1$, then it is $u_{ik} = 0, k = 1, \ldots, m, k \neq j$. The above statements can be expressed as follows:

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

and

$$\text{(C2)} \quad \sum_{j=1}^{m} u_{ij} = 1, \quad i = 1, \ldots, N.$$

The aim of moving the vectors $\boldsymbol{\theta}_j$'s towards the centers of the physical clusters is achieved via the minimization of the cost function

$$J_{k-means}(\Theta, U) = \sum_{i=1}^{N} \sum_{j=1}^{m} u_{ij} \|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2. \tag{2.1}$$

Due to the fact that u_{ij}'s are binary-valued, the minimization of the above cost function with respect to u_{ij}'s cannot be achieved via standard mathematical analysis tools. Moreover, minimizing eq. (2.1) with respect to $\boldsymbol{\theta}_j$'s leads to an equation that is related to u_{ij}'s; thus no closed-form solutions can be derived. Therefore, an alternating optimization scheme is employed, where $J_{k-means}$ is optimized with respect to u_{ij}'s for fixed $\boldsymbol{\theta}_j$'s and then it is optimized with respect to $\boldsymbol{\theta}_j$'s for fixed u_{ij}'s. To this end, we proceed as follows.

Assuming that the representatives $\boldsymbol{\theta}_j$'s are fixed, it is straightforward to see that $J_{k-means}(\Theta, U)$ is minimized, if each $\mathbf{x}_i$ is assigned to its closest cluster; that is to the cluster whose representative is closest to $\mathbf{x}_i$, since for each $\mathbf{x}_i$ only one u_{ij} is 1 and all the others are equal to 0. Thus,

[1] Here, it is implicitly assumed that we are absolutely confident that $\mathbf{x}_i$ belongs to a single cluster and that it does not belong to any of the remaining clusters. However, if no such absolute confidence occurs, we could allow u_{ij} to take values in the internal $[0, 1]$, which may be interpreted as the probability that $\mathbf{x}_i$ belongs to the jth cluster. Schemes related with the latter consideration are called *Probabilistic* and are discussed, e.g. in [63].

$$u_{ij} = \begin{cases} 1, & \text{if } \|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2 = \min\limits_{k=1,\ldots,m} \|\mathbf{x}_i - \boldsymbol{\theta}_k\|^2 \\ 0, & \text{otherwise} \end{cases}, \quad i = 1, \ldots, N. \tag{2.2}$$

Assume now that the membership coefficients u_{ij}'s are fixed. Taking the derivative of $J_{k-means}$ with respect to $\boldsymbol{\theta}_j$ and setting it equal to zero, we obtain that $\boldsymbol{\theta}_j$ is the mean vector of cluster C_j, i.e.,

$$\boldsymbol{\theta}_j = \frac{\sum\limits_{i=1}^{N} u_{ij}\mathbf{x}_i}{\sum\limits_{i=1}^{N} u_{ij}}, \quad j = 1, \ldots, m. \tag{2.3}$$

Eqs. (2.2) and (2.3) give rise to the k-means algorithm, which is explicitly given in "Algorithm 1" box below.

Algorithm 1 [Θ, U] = k-means(X, m)

Input: X, m

1: $t = 0$

▷ *Initialization of $\boldsymbol{\theta}_j$'s part*

2: Choose randomly the initial estimates for $\boldsymbol{\theta}_j$, $\boldsymbol{\theta}_j(t), j = 1, \ldots, m$

3: **repeat**

▷ *Update U part*

4: $u_{ij}(t) = \begin{cases} 1, & \text{if } \|\mathbf{x}_i - \boldsymbol{\theta}_j(t)\|^2 = \min\limits_{k=1,\ldots,m} \|\mathbf{x}_i - \boldsymbol{\theta}_k(t)\|^2 \\ 0, & \text{otherwise} \end{cases}, \quad i = 1, \ldots, N, j = 1, \ldots, m$

5: $t = t + 1$

▷ *Update Θ part*

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

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

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

It has been proved [64] that the algorithm converges to a minimum of the cost function; that is, it recovers as compact clusters as possible, provided that their number is known [18]. However, k-means cannot guarantee convergence to the global minimum of its cost function. In other words, different initializations may lead k-means to produce different clusterings, due to the possible convergence to different local minima of $J_{k-means}(\Theta, U)$.

Remark 1: The major advantage of k-means algorithm is its very low computational cost, which makes it attractive for a wide range of applications, especially in big data applications. More specifically, its complexity is $O(Nm \cdot iter)$, where $iter$ is the (usually small) number of required iterations until convergence.

Remark 2: As already stated, k-means requires prior knowledge of the exact number of physical clusters m underlying the data set, which is rarely known in practice (this has been implicitly assumed in the previous discussion). Poor estimates of the true m make k-means incompetent for revealing the underlying clustering structure in X. In addition, given an estimate of the actual number of clusters, m, k-means will return m clusters even if more or less than m clusters may actually underlie in X. From this perspective, we say that k-means *imposes* a clustering structure on X. Finally, k-means is sensitive to outliers and noise, due to the fact that it assigns every data point to exactly one of the clusters. Thus, outliers and noisy points, being points of X, influence the cluster representatives $\boldsymbol{\theta}_j$'s, by taking part in their updating and, as a consequence, affect the final clustering result. Some techniques for dealing with the problem of the estimation of the number of clusters are discussed in [65] and [66].

Remark 3: It should be mentioned that in some cases, e.g. when the physical clusters are located close to each other and have big differences in their variances, k-means may fail to unravel the underlying clustering structure, even if it is initialized with the true number of clusters. That is, it may split the high-variance cluster into more than one clusters, in its attempt to minimize $J_{k-means}$.

2.3 The Fuzzy C-Means Algorithm

In contrast to the k-means algorithm discussed above, where $u_{ij} \in \{0, 1\}$, in the Fuzzy C-Means algorithm (FCM) [7], [8], u_{ij}'s are allowed to take any value in the interval $[0, 1]$ [2]. In this framework, u_{ij} is called *grade of membership* of data point $\mathbf{x}_i$ with cluster C_j.

The FCM algorithm is derived by minimizing the following cost function

$$J_{FCM}(\Theta, U) = \sum_{i=1}^{N} \sum_{j=1}^{m} u_{ij}^{q} \|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2, \tag{2.4}$$

with respect to $\boldsymbol{\theta}_j$ and u_{ij}, subject to the constraints

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

$$\text{(C2)} \;\; \sum_{j=1}^{m} u_{ij} = 1, \;\; i = 1, \ldots, N \text{ (sum-to-one constraint)}$$

[2]Note, however, that u_{ij}'s have not a probability interpretation here, since here it is assumed that a data point may shared to more than one clusters. That is in contrast to the probabilistic framework, where a vector belongs exclusively to a certain cluster, but we are not absolutely sure to which one.

and

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

where q is a user-defined parameter called *fuzzifier* that takes values greater than 1 (usually in the range [2,3]).

In words, (C2) means that the grades of membership of a certain data vector with all clusters are interrelated and more specifically, they sum to one. This constraint is the one that justifies the basic concept in fuzzy clustering; that is, each $\mathbf{x}_i$ is *shared* among all clusters. Also, (C3) means that for a given cluster, there exists at least one data vector that is not totally incompatible with it or, loosely speaking, no cluster is allowed to be "empty".

Minimization of $J_{FCM}(\Theta, U)$ with respect to u_{ij}, subject to the constraint (C2), leads to the following Lagrangian function:

$$\mathcal{L}(\Theta, U) = \sum_{i=1}^{N}\sum_{j=1}^{m} u_{ij}^{q}\|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2 - \sum_{i=1}^{N}\lambda_i\left(\sum_{j=1}^{m} u_{ij} - 1\right). \tag{2.5}$$

Taking the partial derivative of $\mathcal{L}(\Theta, U)$ with respect to u_{ij} and equating to zero, we obtain

$$u_{ij} = \left(\frac{\lambda_i}{q\|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2}\right)^{\frac{1}{q-1}}, i = 1, \ldots, N, j = 1, \ldots, m. \tag{2.6}$$

Combining eq. (2.6) with constraint (C2) and solving for λ_i, we get

$$\lambda_i = \frac{q}{\left(\sum\limits_{j=1}^{m}\left(\frac{1}{\|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2}\right)^{\frac{1}{q-1}}\right)^{q-1}}, i = 1, \ldots, N. \tag{2.7}$$

Finally, substituting λ_i from eq. (2.7) to eq. (2.6), we conclude to the following equation for u_{ij}'s:

$$u_{ij} = \frac{1}{\sum\limits_{k=1}^{m}\left(\frac{\|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2}{\|\mathbf{x}_i - \boldsymbol{\theta}_k\|^2}\right)^{\frac{1}{q-1}}}, i = 1, \ldots, N, j = 1, \ldots, m \ \ [3]. \tag{2.8}$$

Similarly, setting the partial derivative of $\mathcal{L}(\Theta, U)$ with respect to $\boldsymbol{\theta}_j$ equal to zero, we obtain

$$\boldsymbol{\theta}_j = \frac{\sum\limits_{i=1}^{N} u_{ij}^{q}\mathbf{x}_i}{\sum\limits_{i=1}^{N} u_{ij}^{q}}, j = 1, \ldots, m. \tag{2.9}$$

[3] Note that u_{ij} satisfies (C1) and (C3).

Since the updating equations of the grade of memberships and the cluster representatives (eqs. (2.8), (2.9)) are interrelated, they cannot give a closed-form solution of $J_{FCM(\Theta,U)}$. Therefore, an iterative algorithmic scheme[4] is used, in order to obtain the estimates for U and Θ, giving rise to the FCM algorithm, which is explicitly given in "Algorithm 2" box below.

Algorithm 2 [Θ, U] = FCM(X, m, q)

Input: X, m, q

1: $t = 0$

▷ *Initialization of $\boldsymbol{\theta}_j$'s part*

2: Choose randomly the initial estimates for $\boldsymbol{\theta}_j$, $\boldsymbol{\theta}_j(t), j = 1, \ldots, m$

3: **repeat**

▷ *Update U part*

4: $u_{ij}(t) = \frac{1}{\sum_{k=1}^{m} \left(\frac{\|\mathbf{x}_i - \boldsymbol{\theta}_j(t)\|^2}{\|\mathbf{x}_i(t) - \boldsymbol{\theta}_k(t)\|^2} \right)^{\frac{1}{q-1}}}, i = 1, \ldots, N, j = 1, \ldots, m$

5: $t = t + 1$

▷ *Update Θ part*

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

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

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

Note that FCM does not always converge to the global minimum of its cost function (eq. (2.4)), that is, different initializations may lead to different clustering results. Specifically, in [7] the theorem of Zangwill [68] is used to prove that FCM terminates at a local minimum, or at worst, always contains a subsequence which convergences to a local minimum of its cost function.

Remark 1: FCM requires a priori knowledge of the exact number of clusters underlying in the dataset. Specifically, given an estimate m of the number of possible clusters underlying in X, the algorithm splits the data set to m distinct clusters, irrespectively of the number of the actual clusters that underlie the data set. This is an effect of the sum-to-one constraint. Thus, FCM *imposes* a clustering structure on the data set under study, meaning that it will end up with as many clusters as the user provides (as is the case with k-means also). A method for estimating the actual number of clusters could be via the use of suitable validity indices, as it is proposed in [20] and [22].

[4] Such a scheme is alteratively called "alternating optimization" [67].

Remark 2: An additional characteristic of FCM is its vulnerability to noisy data and outliers, which is also featured on the sum-to-one constraint. For instance, assume a structure of two physical clusters. In this case, outliers that lie far away from both clusters, will have around 0.5 membership to both clusters, although this is not rationale. A method for facing this problem is discussed in [69].

Remark 3: It can be shown that the larger the q, the more the grade of memberships u_{ij}'s are spread in the whole interval of [0,1]. On the other hand, if $q \rightarrow 1$ then FCM approximates k-means, under the concept that u_{ij}'s are either close to 0 or to 1. Usually, the parameter q is set to 2.

2.4 The Possibilistic C-Means Algorithm

An obvious way to overcome the shortcomings of the fuzzy clustering approach is simply to remove the sum-to-one constraint imposed on the rows of U. This gives rise to a new family of clustering algorithms, namely the possibilistic clustering algorithms. Obviously, now the summation $\sum_{j=1}^{m} u_{ij}$ is not necessarily equal to 1 for each $\mathbf{x}_i$, while u_{ij} is now called *degree of compatibility* of the data vector $\mathbf{x}_i$ with the cluster representative $\boldsymbol{\theta}_j$. The most well-known algorithms in this direction are the Possibilistic C-Means (PCM) algorithms, where, according to [9], [10], the u_{ij}'s should satisfy the conditions,

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

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

and

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

In words, (C2) means that no vector $\mathbf{x}_i$ is allowed to be totally incompatible with all clusters, whereas (C3) means that for a given cluster, there is at least one data point that is not totally incompatible with it. Loosely speaking, each data point should "belong" to at least one cluster (C2), whereas no cluster is allowed to be "empty" (C3). The aim of a possibilistic algorithm is to move $\boldsymbol{\theta}_j$'s towards the centers of dense regions. However, minimization of eq. (2.4) without imposition of the sum-to-one constraint leads to the trivial zero solution for u_{ij}'s. In order to alleviate this problem, an additional term should be added in eq. (2.4) that will be a function only of u_{ij}'s and it will be maximized as u_{ij} decreases. Two such candidate terms are proposed in [9] and [10] giving rise to the following two cost functions

$$J_{PCM_1}(\Theta, U) = \sum_{j=1}^{m} J_j \equiv \sum_{j=1}^{m} \overbrace{\left[\sum_{i=1}^{N} u_{ij}^q \|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2 + \gamma_j \sum_{i=1}^{N} (1 - u_{ij})^q \right]}^{J_j}, \tag{2.10}$$

where q is a parameter that "resembles" to the fuzzifier in FCM and

$$J_{PCM_2}(\Theta, U) = \sum_{j=1}^{m} J_j \equiv \sum_{j=1}^{m} \overbrace{\left[\sum_{i=1}^{N} u_{ij} \|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2 + \gamma_j \sum_{i=1}^{N} (u_{ij} \ln u_{ij} - u_{ij}) \right]}^{J_j}, \tag{2.11}$$

where parameter γ_j's are *fixed* user-defined positive parameters. More specifically, the parameter γ_j is related to the "size" of cluster C_j and could be described as a measure of the variance of the jth cluster around its representative. More specifically, γ_j affects

the degree of compatibility u_{ij} of a data vector $\mathbf{x}_i$ with the jth cluster (see Fig. 2.1). Thus, implicitly, γ_j determines the degree of influence of a specific data point on the estimation of the representative of the jth cluster.

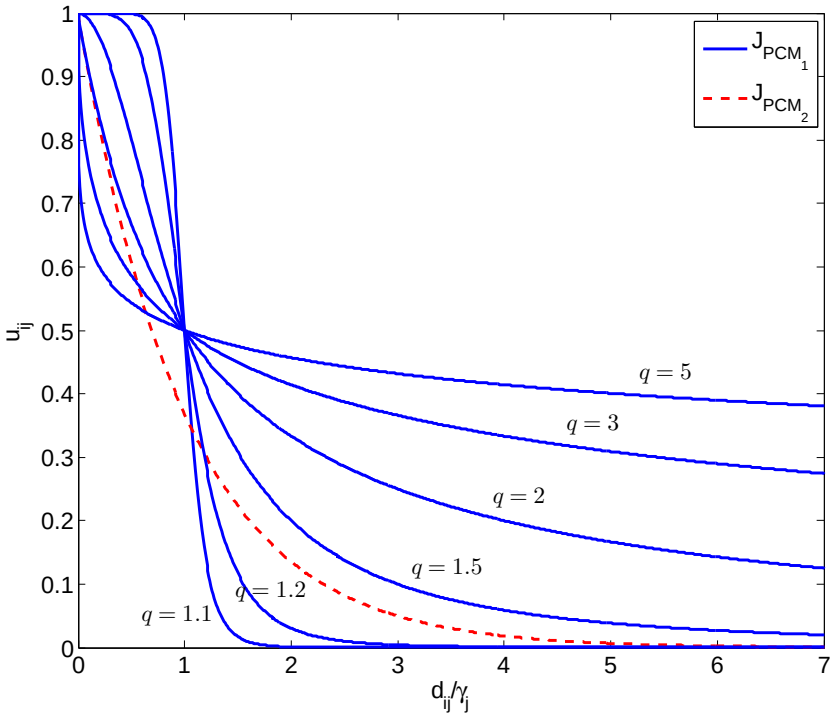


Figure 2.1: The degree of compatibility u_{ij} with respect to d_{ij}/γ_j, where $d_{ij} = \|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2$, for J_{PCM_1} (for several values of q, solid lines) and J_{PCM_2} (dashed line).

Proceeding with the minimization of $J_{PCM_1}(\Theta, U)$ and $J_{PCM_2}(\Theta, U)$ with respect to u_{ij} and $\boldsymbol{\theta}_j$, we end up with the following two sets of equations,

PCM$_1$		PCM$_2$	
$u_{ij} = \dfrac{1}{1 + \left(\frac{\|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2}{\gamma_j}\right)^{\frac{1}{q-1}}}$	(2.12)	$u_{ij} = \exp\left(-\dfrac{\|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2}{\gamma_j}\right)$	(2.13)
$\boldsymbol{\theta}_j = \dfrac{\sum_{i=1}^{N} u_{ij}^q \mathbf{x}_i}{\sum_{i=1}^{N} u_{ij}^q}$	(2.14)	$\boldsymbol{\theta}_j = \dfrac{\sum_{i=1}^{N} u_{ij} \mathbf{x}_i}{\sum_{i=1}^{N} u_{ij}}$	(2.15)

Note from eqs. (2.12), (2.13) that u_{ij} decreases as the distance between $\mathbf{x}_i$ and $\boldsymbol{\theta}_j$ increases for both PCM$_1$ and PCM$_2$. However, in PCM$_2$ this happens exponentially fast, which is more appropriate for the recovery of physical clusters that lie close to each other

(see also Fig. 2.1). Also, from eqs. (2.14), (2.15), it follows that each representative is computed based on *all* data vectors, weighted by their corresponding u_{ij}'s. However, the farther a data vector lies from the current location of a specific $\boldsymbol{\theta}_j$ the less it contributes to the determination of its new location, as eqs. (2.12), (2.13) indicate.

The two algorithms are given in a unified format in "Algorithm 3" box.

Algorithm 3 [Θ, U] = PCM(X, m, q (only for PCM$_1$))

Input: X, m, q (only for PCM$_1$))

1: $t = 0$

▷ *Initialization of $\boldsymbol{\theta}_j$'s part*

2: [$\Theta(t)$, $U^{FCM}(t)$] = FCM(X, m, 2) [5]

▷ *Determination of γ_j's part*

3: $\gamma_j = B\frac{\sum_{i=1}^{N} u_{ij}^{FCM}(t)\|\mathbf{x}_i-\boldsymbol{\theta}_j(t)\|^2}{\sum_{i=1}^{N} u_{ij}^{FCM}(t)}, \quad j = 1, \ldots, m$

4: **repeat**

▷ *Update U part*

5: $u_{ij}(t) = \frac{1}{1+\left(\frac{\|\mathbf{x}_i-\boldsymbol{\theta}_j(t)\|^2}{\gamma_j}\right)^{\frac{1}{q-1}}}, i = 1, \ldots, N, j = 1, \ldots, m$ *for PCM$_1$*

$u_{ij}(t) = \exp\left(-\frac{\|\mathbf{x}_i-\boldsymbol{\theta}_j(t)\|^2}{\gamma_j}\right), i = 1, \ldots, N, j = 1, \ldots, m$ *for PCM$_2$*

6: $t = t + 1$

▷ *Update Θ part*

7: $\boldsymbol{\theta}_j(t) = \frac{\sum_{i=1}^{N} u_{ij}^q(t-1)\mathbf{x}_i}{\sum_{i=1}^{N} u_{ij}^q(t-1)}, j = 1, \ldots, m$ *for PCM$_1$*

$\boldsymbol{\theta}_j(t) = \frac{\sum_{i=1}^{N} u_{ij}(t-1)\mathbf{x}_i}{\sum_{i=1}^{N} u_{ij}(t-1)}, j = 1, \ldots, m$ *for PCM$_2$*

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

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

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

$$\gamma_j = B\frac{\sum_{i=1}^{N} u_{ij}^{FCM}\|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2}{\sum_{i=1}^{N} u_{ij}^{FCM}}, \quad j = 1, \ldots, m, \tag{2.16}$$

[5] See Algorithm 2 of Chapter 2. We set the fuzzifier $q = 2$.

where $\boldsymbol{\theta}_j$'s and u_{ij}^{FCM}'s are the final FCM estimates for cluster representatives and u_{ij} coefficients, respectively[6]. Parameter B is user-defined and is usually set equal to 1 [7]. From eq. (2.16), it is obvious that γ_j is a measure of the variance of cluster C_j around its representative.

It is worth noting that, due to the functional independence between u_{ij}'s, $j = 1, \ldots, m$, for a specific $\mathbf{x}_i$ (eqs. (2.12), (2.13)), the optimization problem solved by PCM can be decomposed into m sub-problems, each one optimizing a specific function J_j (see eqs. (2.10), (2.11)). Considering the representative $\boldsymbol{\theta}_j$ associated with a given J_j, we have from eqs. (2.12), (2.13) that points that lie closer to the cluster representative will have larger degrees of compatibility with C_j. On the other hand, eqs. (2.14), (2.15) imply that the new position of $\boldsymbol{\theta}_j$ is mainly specified by the data points that are most compatible with C_j. It is not difficult to see that such a coupled iteration is expected to lead the representative $\boldsymbol{\theta}_j$ towards the center of the dense in data region that lies closer to their position, for appropriate choices of γ_j's (see also Propositions 2 and 3, in chapter 3).

2.4.1 Possibilistic C-Means Issues and Potential Solutions

Having described the main characteristics of the algorithm and the rationale behind them, let us focus now on some issues that a user faces with PCM. The first one concerns the m parameters γ_j's. An improper choice of γ_j's may lead PCM to failure in identifying properly the physical clusters underlying in the data set, e.g., a sparse cluster that is located very close to a denser cluster (see also experiment 1, in section 3.7), or it may even lead the algorithm to recover the whole data set as a single cluster [24]. Referring to eq. (2.16), the u_{ij}'s produced by the FCM (u_{ij}^{FCM}'s), are not always accurate (e.g. in the presence of noise, [10]). In addition, the choice of the parameter B is clearly data-dependent and there is no general clue on how to select it. In order to deal with this problem, [20] proposes the replacement of all γ_j's by a single quantity that is controlled by only two parameters: (a) the number of clusters and (b) a parameter that plays a "fuzzifier" role.

An additional source of inconveniences concerning γ_j's is the fact that, once they have been set, they remain fixed during the execution of PCM. This reduces the ability of the algorithm to track the variations in the clusters formation during its evolution. A way out of this problem is to allow γ_j's to vary during the execution of the algorithm. A hint on this issue has been given in [9], but, to the best of our knowledge, no further work has been done towards this direction, until the presentation of algorithms like APCM, SAPCM, which are the main contribution of the present thesis.

The second issue, which is related with the first one, is that of *coincident clusters*. As stated before, with a proper choice of γ_j's, PCM drives, in principle, the cluster representatives towards the centers of the dense in data regions that are closer to their positions.

[6] The version of eq. (2.16) proposed in [9] for the cost function J_{PCM1} (eq. (2.10)), raises u_{ij}^{FCM}'s to the qth power. However, in J_{PCM_2} (eq. (2.11)) no parameter q is involved.

[7] An alternative choice for γ_j's, given in [9] is $\gamma_j = \frac{\sum_{u_{ij}>k} \|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2}{\sum_{u_{ij}>k} 1}$, with k being an appropriate threshold.

Therefore, if two or more representatives are initialized close to the same dense region, they will move towards its center, i.e., all of them will represent the same cluster. Alternatively, one could say that the clusters represented by these representatives are coincident[8]. This situation arises due to the absence of dependence between the coefficients u_{ij}, $j = 1, \ldots, m$, associated with a specific $\mathbf{x}_i$ (see eqs. (2.12), (2.13)), which, as an indirect consequence, allows the representatives to move independently from each other (see eqs. (2.14), (2.15)). Note that such an issue does not arise in FCM due to the sum-to-one constraint imposed on the u_{ij}'s associated with each $\mathbf{x}_i$. Several ways to deal with this problem have been proposed in the literature. More specifically, in [19], a variation of PCM is proposed, named Possibilistic Fuzzy c-means (PFCM), which combines concepts from PCM and FCM. Relative approaches are discussed in [23], [28] and [70], while other approaches are proposed in [21], [25].

A common feature in all the previously mentioned works, is that condition (C3), which basically requires all clusters to be non-empty, is respected. Thus, in all the algorithms, the true number of clusters m is implicitly required, in order to give them the ability to recover all clusters, without, hopefully, returning coincident clusters. Thus, the requirement of the knowledge of the number of clusters is still here in disguise. A conceptually simple solution to address this requirement, while respecting condition (C3), comes from the PCM itself. Specifically, one could run the original PCM with an overestimated number of cluster representatives which will be initialized appropriately (at least one representative should lie at each dense in data region). Then, after a proper selection of γ_j's, PCM will (hopefully) recover the physical clusters, that is, it will move at least one representative to the center of each dense region. Then, an additional step is required in order to identify coincident clusters and remove duplicates. This idea has been partially discussed in [10], without, however, proposing explicitly to run the algorithm with an overdetermined number of clusters. However, in this case a reliable method for identifying duplicate clusters should be invented.

In the rest of this thesis, we focus exclusively on the PCM_2 algorithm. Thus, from now on, in order to ease the notation, we will write PCM implying PCM_2.

[8] This point of view justifies the term "coincident clusters".

3. ADAPTIVE POSSIBILISTIC C-MEANS ALGORITHM

3.1 Introduction

Having exposed the weaknesses of the classical PCM algorithm [10] and trying to face them, in this section we introduce an extended version of it by modifying the way the parameters γ are defined and treated. This gives rise to a new algorithm called *Adaptive Possibilistic c-means* (*APCM*) [54][1]. In APCM the parameters γ, after their initialization, are properly adapted as the algorithm evolves. In particular, the parameter γ of each specific cluster is updated based only on those data vectors that are "*most compatible*" with this cluster.

The adaptation of γ's renders the algorithm more flexible in uncovering the underlying clustering structure, compared to other related possibilistic algorithms, where γ's are kept fixed during their execution. This happens especially when dealing with demanding data sets such as those whose data vectors form closely located to each other clusters or clusters with significant differences in their variances. In addition, as a direct consequence of this adaptation mechanism, the algorithm has the ability to estimate the (unknown in most cases in practice) true number of *physical* (or *natural*) clusters. More specifically, if the number of the representatives with which APCM starts is a crude overestimate of the number of natural clusters, the algorithm gradually reduces their number, as it progresses, and, finally, it places a single representative to the center of each dense region. In this sense, it provides not only the number of natural clusters, which is a long-standing issue in the clustering literature, but also the clusters themselves. Analytical results are presented that verify the *cluster elimination* capability of the proposed algorithm and provide strong indications of its convergence behavior. Extensive simulation results on both synthetic and real data, corroborate our theoretical analysis and show that APCM offers in general superior clustering performance compared to relative state-of-the-art clustering schemes.

In the next sections, the various stages of the APCM algorithm are described in detail. Specifically, the way its parameters are initialized is firstly described (Section 3.2). Next, in Section 3.3 the updating of its parameters (u_{ij}'s, $\boldsymbol{\theta}_j$'s, γ_j's, m) is commented, while in Section 3.4 the rationale of the algorithm is exposed, where it is explained in detail, how the initial estimate of the number of natural clusters can be reduced to the true one, as a result of the adaptation of γ_j's. Section 3.5 describes the method for selecting the unique user-defined parameter of the algorithm (α). In Section 3.6, some theoretical results that are indicative of the convergence behaviour of APCM are given. Finally, extensive experimental results are presented in Section 3.7 and conclusions are provided in Section 3.8.

The proposed APCM algorithm stems from the optimization of the cost function of the

[1] A preliminary version of APCM has been presented in [53].

original PCM (eq. (2.11)), by setting

$$\gamma_j = \frac{\hat{\eta}}{\alpha}\eta_j, \tag{3.1}$$

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. The rationale of the adopted expression for γ_j's as given in eq. (3.1) will be analysed and further discussed in Section 3.4.

3.2 Initialization in APCM

As mentioned previously, first, we make an overestimation, denoted by m_{ini}, of the true number of natural clusters m, formed by the data points; that is, we begin with m_{ini} $\boldsymbol{\theta}_j$'s and their corresponding η_j's. Regarding $\boldsymbol{\theta}_j$'s and η_j's, their initialization drastically affects the final clustering result in APCM. Recalling that APCM is a possibilistic-type algorithm and these algorithms move the cluster representatives towards "dense in data points" regions (physical clusters), care should be taken so that at least one representative lies "close" to each physical cluster with its associated η_j being initialized suitably. Thus, a good starting point for them is of crucial importance. 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 run with m_{ini} clusters. Taking into account that FCM is very likely to drive the representatives to dense in data regions (since $m_{ini} > m$), the probability that at least one of the initial $\boldsymbol{\theta}_j$'s is placed in each dense region (cluster) of the data set, increases with m_{ini}.

After the initialization of $\boldsymbol{\theta}_j$'s, η_j's are initialized as follows:

$$\eta_j = \frac{\sum_{i=1}^{N} u_{ij}^{FCM} \|\mathbf{x}_i - \boldsymbol{\theta}_j\|}{\sum_{i=1}^{N} u_{ij}^{FCM}}, \quad j = 1, \ldots, m_{ini}, \tag{3.2}$$

where $\boldsymbol{\theta}_j$'s and u_{ij}^{FCM}'s in eq. (3.2) are the final parameter estimates obtained by FCM[2].

It is worth noting that the above initialization of η_j's involves *Euclidean instead of squared Euclidean distances*, as is the case for γ_j's in the classical PCM. As it will be shown next, this convention will also be kept in the update expressions of η_j's, given below, while its rationale is explained in Section 3.4.

3.3 Parameter adaptation in APCM

In the proposed APCM algorithm, all parameters are adapted during its execution. More specifically, this refers to, (a) the degrees of compatibility u_{ij}'s and the cluster represen-

[2]An alternative initialization for $\boldsymbol{\theta}_j$'s and η_j's is proposed in [53].

tatives $\boldsymbol{\theta}_j$'s, (b) the number of clusters and (c) the η_j's, with (b) and (c) being achieved through two interrelated processes.

As far as the updating of u_{ij}'s is concerned, by setting $\gamma_j = \frac{\hat{\eta}}{\alpha}\eta_j$ in eq. (2.11) and minimizing $J_{PCM}(\Theta, U)$ with respect to u_{ij}, we end up with the following expression

$$u_{ij}(t) = \exp\left(-\frac{\|\mathbf{x}_i - \boldsymbol{\theta}_j(t)\|^2}{\gamma_j(t)}\right) = \exp\left(-\frac{\alpha}{\hat{\eta}}\frac{\|\mathbf{x}_i - \boldsymbol{\theta}_j(t)\|^2}{\eta_j(t)}\right), \tag{3.3}$$

where the iteration dependence of η_j's has now been inserted. In addition, the updating of $\boldsymbol{\theta}_j$'s is done as in the original PCM scheme according to eq. (2.15). Concerning the adjustment of the number of clusters $m(t)$ at the tth iteration, we proceed as follows. Let $label$ be a N-dimensional vector, whose ith element is the index of the cluster which is *most compatible* with $\mathbf{x}_i$, that is the index j for which $u_{ij}(t) = \max_{r=1,\ldots,m(t)} u_{ir}(t)$. At each iteration of the algorithm, the adjustment (reduction) of the number of clusters $m(t)$ is achieved by examining, for each cluster C_j, if its index j appears at least once in the vector $label$ (i.e. if there exists at least one vector $\mathbf{x}_i$ that is most compatible with C_j). If this is the case, C_j is preserved. Otherwise, C_j is eliminated and, thus, U and Θ are updated accordingly. As a result, the current number of clusters $m(t)$ is reduced (see *Possible cluster elimination* part in Algorithm 4).

Finally, concerning γ_j's and in contrast to the classical PCM where they are kept fixed, in APCM they are given by eq. (3.1) and are *adapted* at each iteration of the algorithm through the adaptation of the corresponding η_j's. More specifically, we propose to compute the parameter η_j of a cluster C_j at each iteration, as the *mean absolute deviation* of the most compatible to cluster C_j data vectors, i.e.,

$$\eta_j(t+1) = \frac{1}{n_j(t)} \sum\nolimits_{\mathbf{x}_i : u_{ij}(t) = \max_{r=1,\ldots,m(t+1)} u_{ir}(t)} \|\mathbf{x}_i - \boldsymbol{\mu}_j(t)\|, \tag{3.4}$$

where $n_j(t)$ denotes the number of the data points $\mathbf{x}_i$ that are most compatible with C_j at iteration t and $\boldsymbol{\mu}_j(t)$ the mean vector of these data points (see also *Adaptation of η_j's* part in Algorithm 4). The APCM algorithm can be stated as follows.

Algorithm 4 [Θ, H, m, U] = APCM(X, m_{ini}, α)

Input: X, m_{ini}, α

1: $t = 0$

▷ *Initialization of $\boldsymbol{\theta}_j$'s part*

2: [$\Theta(t)$, $U^{FCM}(t)$] = FCM(X, m_{ini}, 2) [3]

▷ *Initialization of η_j's part*

---

[3] See Algorithm 2 of Chapter 2. We set the fuzzifier $q = 2$.

3: **Set:** $\eta_j(t) = \frac{\sum_{i=1}^{n} u_{ij}^{FCM} \|\mathbf{x}_i - \boldsymbol{\theta}_j(t)\|}{\sum_{i=1}^{n} u_{ij}^{FCM}}$, $j = 1, ..., m_{ini}$

4: **Set:** $\hat{\eta} = \min_{j=1,...,m_{ini}} \eta_j(t)$

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

6: **repeat**

▷ *Update U part*

7: $u_{ij}(t) = \exp\left(-\frac{\alpha}{\hat{\eta}} \frac{||\mathbf{x}_i - \boldsymbol{\theta}_j(t)||^2}{\eta_j(t)}\right)$, $i = 1, ..., N$, $j = 1, ..., m(t)$

▷ *Update Θ part*

8: $\boldsymbol{\theta}_j(t+1) = \sum_{i=1}^{N} u_{ij}(t)\mathbf{x}_i \Big/ \sum_{i=1}^{N} u_{ij}(t)$, $j = 1, ..., m(t)$

▷ *Possible cluster elimination part*

9: **for** $i \leftarrow 1$ **to** N **do**

10: **Determine:** $u_{ir}(t) = \max_{j=1,...,m(t)} u_{ij}(t)$

11: **Set:** $label(i) = r$

12: **end for**

13: **Compute:** $n_j(t)$, $j = 1, ..., m(t)$

14: $p = 0$ *//number of removed clusters at iteration t*

15: **for** $j \leftarrow 1$ **to** m **do**

16: **if** $n_j(t) = 0$ **or** 1 **then**

17: **Remove:** C_j (and **renumber** accordingly Θ and the columns of U)

18: $p = p + 1$

19: **end if**

20: **end for**

21: $m(t+1) = m(t) - p$

▷ *Adaptation of η_j's*

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

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

24: $t = t + 1$

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

26: **return** $\Theta = \{\boldsymbol{\theta}_1(t), \boldsymbol{\theta}_2(t), \ldots, \boldsymbol{\theta}_{m(t)}(t)\}$, $H = [\eta_1(t), \ldots, \eta_{m(t)}(t)]$, $m(t)$, $U = [u_{ij}(t-1)]$

Note that, the definition of γ_j's in the proposed updating mechanism from eqs. (3.1), (3.4), differs from others used in the classical PCM, as well as in many of its variants, in two distinctive points. First, η_j's in APCM are updated taking into account *only* the data vectors that are most compatible to cluster C_j and not all the data points weighted by their corresponding u_{ij} coefficients. This particularity is an essential condition for succeeding cluster elimination, as by this way the value of a parameter η_j may be pushed to zero, thus eliminating the corresponding cluster C_j, whereas in the case where all data points were taken into account, η_j would remain always positive. Note that in the case where a cluster C_j has no most compatible points with it, it is eliminated (see *Possible cluster elimination part*). If a cluster C_j has only one most compatible point to it, say $\mathbf{x}_i$, C_j is also eliminated. This happens because otherwise its parameter η_j becomes zero (through eq. (3.4)) and, as a consequence, at the next iteration all u_{ij}'s, $i = 1, ..., N$ will become zero (see line 7 of Algorithm 4)[4]. Thus, the updating of its parameter $\boldsymbol{\theta}_j$ will become impossible (see line 8, Algorithm 4).

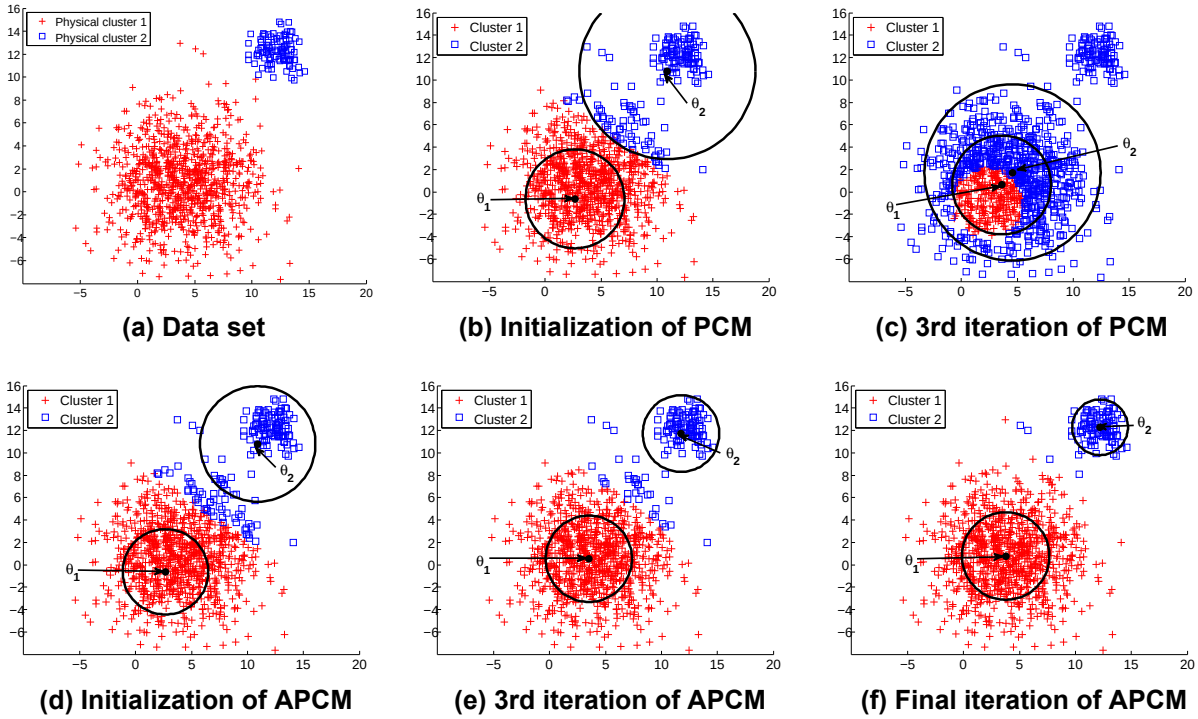
Second, the distances involved in eq. (3.4) are between a data vector and the mean vector $\boldsymbol{\mu}_j(t)$ of the most compatible points of the cluster; *not* from $\boldsymbol{\theta}_j(t)$, as in previous works (e.g. [9], [23]). This allows more accurate estimates of η_j's, since $\boldsymbol{\mu}_j(t)$ is expected to be closer to the next location of $\boldsymbol{\theta}_j$, $\boldsymbol{\theta}_j(t+1)$, than $\boldsymbol{\theta}_j(t)$. This is crucial mainly during the first few iterations of the algorithm where the position of $\boldsymbol{\theta}_j$ may vary significantly from iteration to iteration. It is also noted that, in the (rare) case where there are two or more clusters, that are equally compatible with a specific $\mathbf{x}_i$, the latter will contribute to the determination of the parameter η of *only* one of them, which is chosen arbitrarily. This modification prevents a scenario of having equal η_j's in such exceptional cases (e.g. in data sets consisting of symmetrically arranged data points) which assists the successful cluster elimination procedure, in situations where this must be carried out. Finally, it is worth pointing out that the definition of eq. (3.4), implicitly interrelates the various γ_j's and this interrelation passes to the u_{ij}'s concerning a given $\mathbf{x}_i$ through eq. (3.3).

3.4 Rationale of the algorithm

As mentioned in the previous section, the modifications made in the original PCM leading to APCM aim at a) making the algorithm capable to handling stringent clustering situations and b) allowing for cluster elimination. In the following we describe in more detail the hidden mechanisms of APCM that render these two goals feasible.

First, we consider the case where we have two physical clusters of very different variances that are located very close to each other (Fig. 3.1a). This is a difficult clustering problem,

[4]Note that $\boldsymbol{\theta}_j(t) \neq \mathbf{x}_i$ by its definition.



(a) Data set (b) Initialization of PCM (c) 3rd iteration of PCM

(d) Initialization of APCM (e) 3rd iteration of APCM (f) Final iteration of APCM

Figure 3.1: An example of a two dimensional data set consisting of two physical clusters that have big difference in their variances and are located very close to each other. (a) The data set, (b) the initial stage of PCM, (c) the 3rd iteration of PCM, (d) the initial stage of APCM, (e) the 3rd iteration of APCM and (f) the final stage of APCM. The circles are centered at θ_j's and have radius $\sqrt{\gamma_j}$'s.

in which most state-of-the-art clustering techniques fail. We assume that after initialization with FCM, PCM has two representatives in the areas of the physical clusters, as shown in Fig. 3.1b, with $\boldsymbol{\theta}_1$ lying in the high variance physical cluster and $\boldsymbol{\theta}_2$ in the low variance physical cluster. Then, from eq. (2.16) and due to the proximity of the two physical clusters, it turns out that γ_2 will be much larger than the actual variance of physical cluster 2. This is so because, besides the points of physical cluster 2, the numerous, yet more distant, points of physical cluster 1, will contribute to the computation of γ_2 from eq. (2.16). This means that the representative of the small variance cluster ($\boldsymbol{\theta}_2$) is affected by the data points of its nearby cluster (C_1), according to eqs. (2.13), (2.15). As a result, PCM is likely to end up with both representatives converging erroneously in the center of the large variance physical cluster 1 (Fig. 3.1c).

This issue of PCM is alleviated in APCM, by taking care for each representative to stay in the region of the physical cluster where it was first placed. To this end, APCM reduces (compared to PCM) the range of influence arround each $\boldsymbol{\theta}_j$ that has γ_j larger than the variance of the smallest physical cluster formed in the data set. In this way, the probability of the movement of a representative, which is initialized in the region of a specific physical cluster with a given variance, towards the center of a nearby physical cluster with a larger variance, is reduced. In particular, the larger (smaller) the γ_j than the variance of the smallest physical cluster is, the more it is reduced (increased). On the other hand, a γ_j

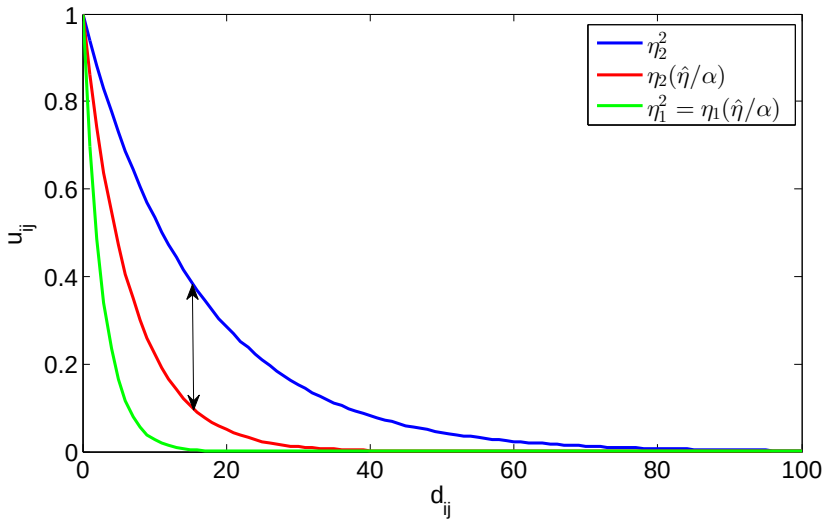


Figure 3.2: The degree of compatibility u_{ij} with respect to distance d_{ij} ($\eta_2 > \eta_1 = \hat{\eta}/\alpha$).

that is equal to the variance of the smallest physical cluster is not affected at all. Focusing on a given iteration (dropping the index t), this is achieved in APCM by defining γ_j as in eq. (3.1). This definition results from the γ_j's as defined in the original PCM via the following transformations.

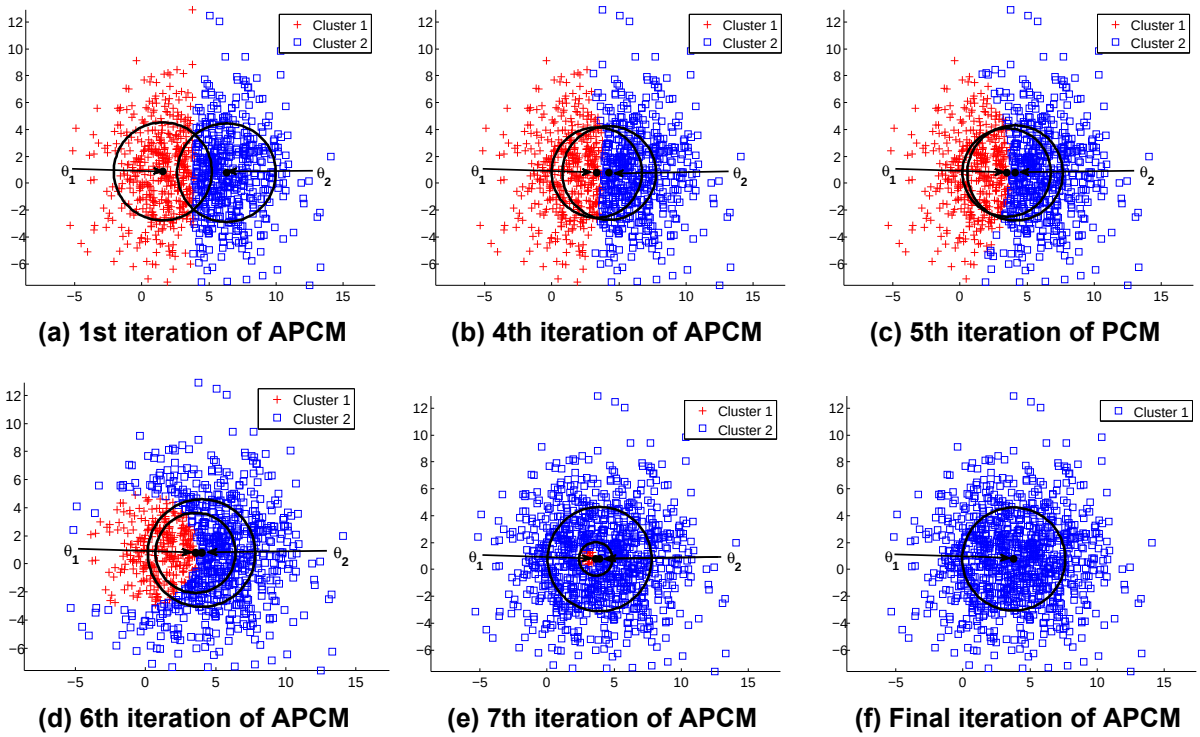
$$\gamma_j^{PCM} = \frac{\sum_{i=1}^{N} u_{ij}^{FCM} \|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2}{\sum_{i=1}^{N} u_{ij}^{FCM}} \overset{①}{\rightsquigarrow}$$

$$\gamma_j' = \frac{\sum_{\mathbf{x}_i : u_{ij} = \max_{r=1,\ldots,m} u_{ir}} \|\mathbf{x}_i - \boldsymbol{\mu}_j\|^2}{n_j} \overset{②}{\rightsquigarrow}$$

$$\eta_j^2 = \left(\frac{\sum_{\mathbf{x}_i : u_{ij} = \max_{r=1,\ldots,m} u_{ir}} \|\mathbf{x}_i - \boldsymbol{\mu}_j\|}{n_j} \right)^2 \overset{③}{\rightsquigarrow} \eta_j \frac{\hat{\eta}}{\alpha}. \tag{3.5}$$

Under transformation ①, γ_j^{PCM} is transformed to γ_j', where (a) only the $\mathbf{x}_i$'s that are most compatible with $\boldsymbol{\theta}_j$ are taken into account and (b) $\boldsymbol{\theta}_j$ is replaced by $\boldsymbol{\mu}_j$. The adoption of the above hard computation of γ_j''s is necessary for the cluster elimination procedure, as it will be further explained in the sequel. Transformation ② leads γ_j' to η_j^2, which carries the same "quality of information" with its predecessor and moreover, η_j^2 is upper bounded by γ_j', $j = 1, \ldots, m$ (see Proposition 1 in Appendix A). This intermediate step on the one hand reduces the influence of clusters arround their representatives while, on the other hand, is a prerequisite for transformation ③. Assuming that α is chosen so that the quantity $\hat{\eta}/\alpha$ equals to the mean alsolute deviation of the smallest physical cluster formed in the data set, then for each $\eta_j \geq \hat{\eta}/\alpha$ ($\eta_j \leq \hat{\eta}/\alpha$), we have that $\eta_j^2 \geq \eta_j\,(\hat{\eta}/\alpha)$ ($\eta_j^2 \leq \eta_j\,(\hat{\eta}/\alpha)$). That is, by substituting η_j^2 with $\eta_j(\hat{\eta}/\alpha)$, the greater (smaller) the η_j of a cluster C_j than $\hat{\eta}/\alpha$ is, the more the range of influence arround its $\boldsymbol{\theta}_j$ is reduced (enhanced) (see Fig. 3.2 and Figs. 3.1d, 3.1e). This justifies our choice for the γ_j's given in eq. (3.1).

In the sequel, we will focus on the cluster elimination property of the APCM algorithm. To



(a) 1st iteration of APCM (b) 4th iteration of APCM (c) 5th iteration of PCM

(d) 6th iteration of APCM (e) 7th iteration of APCM (f) Final iteration of APCM

Figure 3.3: A two dimensional data set consisting of a single physical cluster. APCM is intialized with two representatives and the cluster elimination procedure is illustrated at several iteration steps.

this end, consider the case where a single physical cluster is formed by the data points where $k(>1)$ representatives $\boldsymbol{\theta}_j$'s, $j=1,\ldots,k$, are initialized within it (see Fig. 3.3a for $k=2$). As eq. (2.15) suggests, each representative will move towards the center of the dense region (see also Propositions 2 and 3 in Section 3.6 for a more rigorous justification). As $\boldsymbol{\theta}_j$'s move towards the center of the region, they are getting closer to each other. At a specific iteration t_0 ($t_0=6$ in Fig. 3.3d) where, say $\gamma_r(t_0)=\max_{j=1,\ldots,k}\gamma_j(t_0)$, the hypersphere centered at $\boldsymbol{\theta}_r(t_0)$ and having radius $\sqrt{\gamma_r(t_0)}$ will enclose all the hyperspheres associated with the other representatives. From this point on, the region of influence (γ_j) of all the clusters except C_r shrinks to 0 as is shown in Fig. 3.3, due to their definition (see eqs. (3.1), (3.4)) (a theoretical justification for the two representatives case is given in Proposition 4 in Section 3.6).

3.5 Selection of parameter α

As it was mentioned previously, α is a user-defined parameter that has to be fine-tuned, so that $\hat{\eta}/\alpha$ becomes equal to the mean absolute deviation of the smallest physical cluster. As it is expected, larger values of m_{ini} lead to smaller initial η_j's and thus to a smaller $\hat{\eta}$. As a consequence, there exists a trade-off between m_{ini} and parameter α: large (small) values of m_{ini} require small (large) values of α, so that the ratio $\hat{\eta}/\alpha$ approximates the

mean absolute deviation of the smallest physical cluster. Note that although the latter quantity is fixed for a given data set, it is unknown in practice.

In the sequel, we discuss how different choices of α affect the behavior of APCM, focusing on the limiting cases $\alpha \to 0$ and $\alpha \to +\infty$. Specifically, we consider a single representative $\boldsymbol{\theta}_j$ and we concentrate on its corresponding "subcost" function[5]

$$J_j(\boldsymbol{\theta}_j) = \sum_{i=1}^{N} u_{ij}\|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2 + \eta_j \frac{\hat{\eta}}{\alpha} \sum_{i=1}^{N}(u_{ij} \ln u_{ij} - u_{ij}),$$

where we assume for the time being that η_j is constant, while u_{ij} is given as $u_{ij} = \exp\left(-\frac{\alpha}{\hat{\eta}}\frac{\|\mathbf{x}_i-\boldsymbol{\theta}_j\|^2}{\eta_j}\right)$ (see eq. (3.3)). Utilizing the last equation and after some algebra, $J_j(\boldsymbol{\theta}_j)$ can be written as

$$J_j(\boldsymbol{\theta}_j) = -\eta_j \frac{\hat{\eta}}{\alpha} \sum_{i=1}^{N} \exp\left(-\frac{\alpha}{\hat{\eta}}\frac{\|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2}{\eta_j}\right). \tag{3.6}$$

Taking the gradient of J_j with respect to $\boldsymbol{\theta}_j$, we have:

$$\frac{\partial J_j(\boldsymbol{\theta}_j)}{\partial \boldsymbol{\theta}_j} = 2\sum_{i=1}^{N} \exp\left(-\frac{\alpha}{\hat{\eta}}\frac{\|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2}{\eta_j}\right)(\mathbf{x}_i - \boldsymbol{\theta}_j). \tag{3.7}$$

For $\alpha \to 0$, we have that $\exp\left(-\frac{\alpha}{\hat{\eta}}\frac{\|\mathbf{x}_i-\boldsymbol{\theta}_j\|^2}{\eta_j}\right) \to 1$. Thus, $\frac{\partial J_j(\boldsymbol{\theta}_j)}{\partial \boldsymbol{\theta}_j}$ tends to $2\sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\theta}_j)$ and equating the latter to zero, we end up with $\boldsymbol{\theta}_j = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i$. Thus, in this case there exists a single minimum; the mean of the data set.

For $\alpha \to +\infty$, it is clear from eq. (3.6) that, identically, $J_j(\boldsymbol{\theta}_j) = 0$. Thus, all possible choices for $\boldsymbol{\theta}_j$ are (trivially) local minima of $J_j(\boldsymbol{\theta}_j)$. As α gradually increases from 0, the number of minima of $J_j(\boldsymbol{\theta}_j)$ increases and it is expected that, for a specific range of α values, the minima of $J_j(\boldsymbol{\theta}_j)$ will correspond to the centers of the physical clusters. Of course, this cease to hold as we move outside this range towards $+\infty$.

The above are illustrated via a simple clustering example. Specifically, we consider an one-dimensional data set consisting of two Gaussian clusters with 50 points each, shown on the x-axis in Fig. 3.4a. The centers of the clusters are at locations 28 and 67 and their variances are 100 and 121, respectively. We consider two cases: in the first, the number of initial representatives is $m_{ini} = 3$ while in the second, $m_{ini} = 10$. We run first the FCM algorithm for each case and we obtain the resulting u_{ij}^{FCM}'s and $\boldsymbol{\theta}_j$'s, from which the initial γ_j's are computed using eqs. (3.2) and (3.1). Note that for $m_{ini} = 3$ and $m_{ini} = 10$, the corresponding $\hat{\eta}$ values are 7.0094 and 2.3213.

In order to investigate further the relation between α and m_{ini}, we focus on J_j that corresponds to the minimum initial γ_j and we drop time dependence. Thus, in this case, γ_j is fixed to $\hat{\eta}^2/\alpha$. The "subcost" function $J_j(\theta_j) = \sum_{i=1}^{N} u_{ij}|x_i - \theta_j|^2 + \frac{\hat{\eta}^2}{\alpha}\sum_{i=1}^{N}(u_{ij}\ln u_{ij} - u_{ij})$ is plotted with respect to θ_j, for various values of α. We consider first $m_{ini} = 3$, i.e., m_{ini} is very close to the number of actual clusters ($m = 2$). Thus, in this case, the FCM algorithm

[5]We write $J_j(\boldsymbol{\theta}_j)$ to explicitly denote the dependence of J_j on $\boldsymbol{\theta}_j$.

is more likely to give good initial estimates for η_j's (through eq. (3.2)), i.e. the minimum initial $\eta_j(\equiv \hat{\eta})$ approximates the mean absolute deviation of the smallest physical cluster. We consider the following indicative cases:

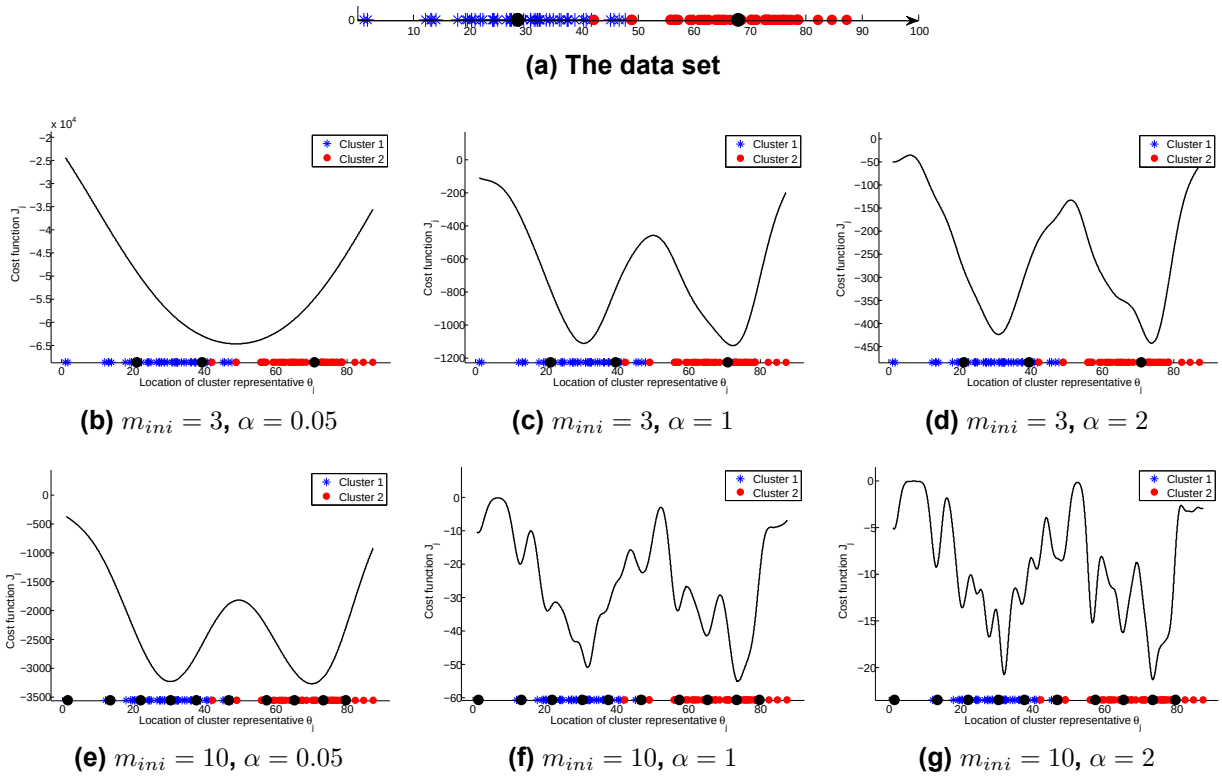
- $\alpha = 0.05$: In this case the ratio $\hat{\eta}/\alpha$ becomes much larger than the mean absolute deviation of the smallest physical cluster, leading all data points to have significant u_{ij}'s for all representatives (through eq. (3.3)). This justifies the plot of Fig. 3.4b, where J_j exhibits just a single valley centered at the mean of the data set. Clearly, the minimization of J_j will lead θ_j to this position, which means that in this case the algorithm will fail to detect any of the two true clusters.
- $\alpha = 1$ or 2: In this case the ratio $\hat{\eta}/\alpha$ approximates the mean absolute deviation of the smallest physical cluster and as we can see in Figs. 3.4c, 3.4d, two well formed valleys are centered at the means of the two natural clusters (despite the presence of a bit disturbance in the $\alpha = 2$ case). Thus, minimization of J_j will lead θ_j to the center of a true cluster.

In conclusion, when m_{ini} is close to the actual m and provided that at least one representative is placed at each dense region, the minimum η_j ($\hat{\eta}$) that is obtained from FCM (eq. (3.2)) is *a good estimate* of the mean absolute deviation of the smallest physical cluster, thus values of α around 1 allow the algorithm to work properly.

However, in the case where $m_{ini} = 10$ (that is $m_{ini} \gg m$) the situation changes. There, all initial η_j's and thus $\hat{\eta}$ are much smaller than the mean absolute deviation of the smallest physical cluster. We consider the following indicative cases:

- $\alpha = 0.05$: In this case the ratio $\hat{\eta}/\alpha$ approximates the mean absolute deviation of the smallest physical cluster. Thus, two well formed valleys are centered at the means of the two natural clusters (see in Fig. 3.4e) and the APCM will lead a θ_j to the center of a true cluster.
- $\alpha = 1$ or 2: In this case J_j exhibits many local minima (see Figs. 3.4f, 3.4g), as the ratio $\hat{\eta}/\alpha$ is significantly smaller than the mean absolute deviation of the smallest physical cluster, leading all data points to have negligible u_{ij}'s values, even with θ_j's that are placed very close to them (through eq. (3.3)). As a consequence, J_j exhibits several local minima that do not correspond to any of the two true clusters and APCM is most likely to end up with clusters that do not correspond to the underlying data set structure.

This example indicates that in cases where m_{ini} is chosen not to be very larger than the actual number of clusters m, appropriate values for the parameter α are around 1. On the other hand, when m_{ini} is chosen much larger than m, parameter α should be taken much less than 1. However, in more demanding data sets, which contain very closely located natural clusters and for a fixed value of m_{ini}, 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. Experiments showed



(a) The data set

(b) $m_{ini} = 3$, $\alpha = 0.05$ **(c)** $m_{ini} = 3$, $\alpha = 1$ **(d)** $m_{ini} = 3$, $\alpha = 2$

(e) $m_{ini} = 10$, $\alpha = 0.05$ **(f)** $m_{ini} = 10$, $\alpha = 1$ **(g)** $m_{ini} = 10$, $\alpha = 2$

Figure 3.4: Plot of the APCM cost function with respect to θ_j for a two-class 1-dim data set. (a) The data set. Data points are denoted by stars on the x-axis and representatives by black dots. Results for (b) $m_{ini} = 3$, $\alpha = 0.05$, (c) $m_{ini} = 3$, $\alpha = 1$, (d) $m_{ini} = 3$, $\alpha = 2$, (e) $m_{ini} = 10$, $\alpha = 0.05$, (f) $m_{ini} = 10$, $\alpha = 1$ and (g) $m_{ini} = 10$, $\alpha = 2$.

that values of α around 1 and up to 3 are appropriate for almost any data set, provided that m_{ini} is not extremely larger than m (about 3-4 times larger).

3.6 Convergence results for APCM

In this section we prove some propositions that are indicative of the basic properties of APCM, namely the convergence of the representatives to the center of dense regions and cluster elimination. Note that some convergence results on the possibilistic algorithms are given in [72]. However, these are not applicable to APCM, due to the adaptation mechanism employed for the parameters η_j's. We begin with the following proposition.

Proposition 1. *Let $\boldsymbol{\theta}_1$, $\boldsymbol{\theta}_2$ be two cluster representatives with $\eta_2 < \eta_1$. The geometrical locus of the points $\mathbf{x} \in \Re^\ell$ having $u_2(\mathbf{x}) > u_1(\mathbf{x})$, where $u_j(\mathbf{x}) = \exp\left(-\frac{\alpha d_j(\mathbf{x})}{\eta_j \hat{\eta}}\right)$ and $d_j(\mathbf{x}) = \|\mathbf{x} - \boldsymbol{\theta}_j\|^2$, $j = 1, 2$, is the set of points that lie in the interior of the hypersphere C:*

$$\|\mathbf{x} - \frac{k\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1}{k-1}\|^2 = \frac{k}{(k-1)^2}\|\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1\|^2 \equiv r^2, \tag{3.8}$$

centered at $\frac{k\boldsymbol{\theta}_2-\boldsymbol{\theta}_1}{k-1}$ *and having radius* $r=\frac{\sqrt{k}}{k-1}\|\boldsymbol{\theta}_2-\boldsymbol{\theta}_1\|$, *where* $k=\eta_1/\eta_2(>1)$.

Proof. It is $u_1(\mathbf{x})<u_2(\mathbf{x}) \Leftrightarrow \frac{d_1(\mathbf{x})}{\eta_1}>\frac{d_2(\mathbf{x})}{\eta_2} \Leftrightarrow d_1(\mathbf{x})>kd_2(\mathbf{x}) \Leftrightarrow \|\mathbf{x}-\boldsymbol{\theta}_1\|^2>k\|\mathbf{x}-\boldsymbol{\theta}_2\|^2 \Leftrightarrow \|\mathbf{x}\|^2-2\mathbf{x}^T\boldsymbol{\theta}_1+\|\boldsymbol{\theta}_1\|^2>k\|\mathbf{x}\|^2-2k\mathbf{x}^T\boldsymbol{\theta}_2+k\|\boldsymbol{\theta}_2\|^2 \Leftrightarrow (k-1)\|\mathbf{x}\|^2-2(k\boldsymbol{\theta}_2-\boldsymbol{\theta}_1)^T\mathbf{x}+k\|\boldsymbol{\theta}_2\|^2-\|\boldsymbol{\theta}_1\|^2<0 \Leftrightarrow \|\mathbf{x}\|^2-2\left(\frac{k\boldsymbol{\theta}_2-\boldsymbol{\theta}_1}{k-1}\right)^T\mathbf{x}+\frac{k\|\boldsymbol{\theta}_2\|^2-\|\boldsymbol{\theta}_1\|^2}{k-1}<0 \Leftrightarrow \|\mathbf{x}\|^2-2\left(\frac{k\boldsymbol{\theta}_2-\boldsymbol{\theta}_1}{k-1}\right)^T\mathbf{x}+\|\frac{k\boldsymbol{\theta}_2-\boldsymbol{\theta}_1}{k-1}\|^2-\|\frac{k\boldsymbol{\theta}_2-\boldsymbol{\theta}_1}{k-1}\|^2+\frac{k\|\boldsymbol{\theta}_2\|^2-\|\boldsymbol{\theta}_1\|^2}{k-1}<0 \Leftrightarrow \|\mathbf{x}-\frac{k\boldsymbol{\theta}_2-\boldsymbol{\theta}_1}{k-1}\|^2<\frac{\|k\boldsymbol{\theta}_2-\boldsymbol{\theta}_1\|^2}{(k-1)^2}-\frac{k\|\boldsymbol{\theta}_2\|^2-\|\boldsymbol{\theta}_1\|^2}{k-1}$ or $\|\mathbf{x}-\frac{k\boldsymbol{\theta}_2-\boldsymbol{\theta}_1}{k-1}\|^2<\frac{k}{(k-1)^2}\|\boldsymbol{\theta}_2-\boldsymbol{\theta}_1\|^2$. □

Note that the radius r of C can be written in terms of η_1, η_2 as

$$r=\frac{\sqrt{\eta_1\eta_2}}{|\eta_1-\eta_2|}\|\boldsymbol{\theta}_2-\boldsymbol{\theta}_1\|^2. \tag{3.9}$$

We consider next the continuous case where the data vectors are modelled by a random vector **x** that follows a continuous pdf distribution $p(\mathbf{x})$. In this case, the updating equations for the APCM algorithm (with a slight modification in notation, in order to denote explicitly the dependence of $u_j(\mathbf{x})$ from the continuous random variable **x**) are given below.

$$\boldsymbol{\theta}_j^{t+1}=\frac{\int_{\Re^\ell}u_j^t(\mathbf{x})\mathbf{x}p(\mathbf{x})d\mathbf{x}}{\int_{\Re^\ell}u_j^t(\mathbf{x})p(\mathbf{x})d\mathbf{x}}, \tag{3.10}$$

$$\text{where}\quad u_j^t(\mathbf{x})=\exp\left(-\frac{\|\mathbf{x}-\boldsymbol{\theta}_j^t\|^2}{\gamma_j^t}\right) \tag{3.11}$$

$$\gamma_j^t=\frac{\hat{\eta}}{\alpha}\frac{\int_{T_j^t}\|\mathbf{x}-\boldsymbol{\mu}_j^t\|p(\mathbf{x})d\mathbf{x}}{\int_{T_j^t}p(\mathbf{x})d\mathbf{x}} \tag{3.12}$$

$$\text{and}\quad \boldsymbol{\mu}_j^t=\frac{\int_{T_j^t}\mathbf{x}p(\mathbf{x})d\mathbf{x}}{\int_{T_j^t}p(\mathbf{x})d\mathbf{x}}, \tag{3.13}$$

with $T_j^t=\{\mathbf{x}: u_j^t(\mathbf{x})=\max_{q=1,\ldots,m}u_q^t(\mathbf{x})\}$, $j=1,\ldots,m$.

The above equations define the iterative scheme $\boldsymbol{\theta}_j^{t+1}=f(\boldsymbol{\theta}_j^t)$, where

$$f(\boldsymbol{\theta}_j^t)=\frac{\int_{\Re^\ell}\exp\left(-\frac{\|\mathbf{x}-\boldsymbol{\theta}_j^t\|^2}{\gamma_j^t}\right)\mathbf{x}p(\mathbf{x})d\mathbf{x}}{\int_{\Re^\ell}\exp\left(-\frac{\|\mathbf{x}-\boldsymbol{\theta}_j^t\|^2}{\gamma_j^t}\right)p(\mathbf{x})d\mathbf{x}}. \tag{3.14}$$

In the sequel we give some indicative theoretical results concerning aspects of the behavior of APCM, namely (a) the convergence of the cluster representatives to the centers of the dense in data regions and (b) the cluster elimination mechanism. First, we state two assumptions that will be used as premises in the propositions to follow.

Assumption 1: (a) $p(\mathbf{x})$ decreases isotropically along all directions around its center $\mathbf{c}$ [6].

(b) Without loss of generality, we consider the case $\mathbf{c} = \mathbf{0}$.

Note that this assumption indicates the existence of a *single* dense in data region.

Assumption 2: $p(\mathbf{x})$ is a zero mean normal distribution $\mathcal{N}(\mathbf{0}, \sigma^2 I)$.

(Clearly, Assumption 2 is more restrictive than assumption 1.)

Proposition 2. *Under assumption 1, the center* $\mathbf{c} = \mathbf{0}$ *of* $p(\mathbf{x})$ *is a fixed point for the iterative scheme defined by eq.* (3.14)*.*

Proof. Assuming that $\boldsymbol{\theta}_j^t = \mathbf{0}$, we will show that $\boldsymbol{\theta}_j^{t+1} = \mathbf{0}$ also. Dropping the index j from $\boldsymbol{\theta}_j$, γ_j from eq. (3.14) we have

$$\boldsymbol{\theta}^{t+1} = \frac{\int_0^\infty \left[\int_{\|\mathbf{x}\|^2=r^2} \exp\left(-\frac{\|\mathbf{x}\|^2}{\gamma^t}\right) \mathbf{x} p(\mathbf{x}) dA_r\right] dr}{\int_0^\infty \left[\int_{\|\mathbf{x}\|^2=r^2} \exp\left(-\frac{\|\mathbf{x}\|^2}{\gamma^t}\right) p(\mathbf{x}) dA_r\right] dr}, \tag{3.15}$$

where $\int_{\|\mathbf{x}\|^2=r^2}(\cdot)dA_r$ is the integral over the hypersphere $\|\mathbf{x}\|^2 = r^2$.

Continuing from eq. (3.15) we have

$$\boldsymbol{\theta}^{t+1} = \frac{\int_0^\infty \exp\left(-\frac{r^2}{\gamma^t}\right) \left[\int_{\|\mathbf{x}\|^2=r^2} \mathbf{x} p(\mathbf{x}) dA_r\right] dr}{\int_0^\infty \exp\left(-\frac{r^2}{\gamma^t}\right) \left[\int_{\|\mathbf{x}\|^2=r^2} p(\mathbf{x}) dA_r\right] dr}. \tag{3.16}$$

But, due to the isotropic property of $p(\mathbf{x})$ along all directions around $\mathbf{0}$, all points on the hypersphere $\|\mathbf{x}\|^2 = r^2$ are evenly distributed (and have the same magnitude). Thus, it is:

$$\int_{\|\mathbf{x}\|^2=r^2} \mathbf{x} p(\mathbf{x}) dA_r = \mathbf{0}. \tag{3.17}$$

Noting also that $\exp\left(-\frac{r^2}{\gamma^t}\right) > 0$ and $\int_{\|\mathbf{x}\|^2=r^2} p(\mathbf{x}) dA_r$ is the area of the hypersphere $\|\mathbf{x}\|^2 = r^2$, the denominator in eq. (3.16) is positive. Thus, eqs. (3.16) and (3.17) finally give $\boldsymbol{\theta}^{t+1} = \mathbf{0}$. In other words, $\mathbf{0}$ is indeed a fixed point of the iterative scheme defined by eq. (3.14). □

Proposition 3. *Adopt the assumption 2 and consider the mapping* $f : \Re^\ell \to \Re^\ell$ *defined by eq.* (3.14)*. Then, the fixed point* $\mathbf{0}$ *of the scheme* $\boldsymbol{\theta}^{t+1} = f(\boldsymbol{\theta}^t)$ *is stable.*

Proof. Focusing on the s-th component $f_s(\boldsymbol{\theta})$ of the above mapping and utilizing the assumption 2 of $p(\mathbf{x})$ as well as the fact that $\exp\left(-\frac{\|\mathbf{x}-\boldsymbol{\theta}\|^2}{\gamma}\right) = \prod_{q=1}^{\ell} \exp\left(-\frac{(x_q-\theta_q)^2}{\gamma}\right)$, it is easy

[6] Such pdf's are e.g. the independent identically distributed (i.i.d) multivariate normal and Laplace distributions.

to verify that:

$$f_s(\boldsymbol{\theta}) = \frac{\int_{\Re} x_s \exp\left(-\frac{(x_s-\theta_s)^2}{\gamma}\right) p(x_s)dx_s}{\int_{\Re} \exp\left(-\frac{(x_s-\theta_s)^2}{\gamma}\right) p(x_s)dx_s} \equiv f_s(\theta_s). \tag{3.18}$$

Thus, $f_s(\boldsymbol{\theta})$ depends only on θ_s.

In order to prove the stability of $\boldsymbol{\theta} = \mathbf{0}$, we will compute the Jacobian matrix on $\boldsymbol{\theta} = \mathbf{0}$ and we will show that $|J(\boldsymbol{\theta})| < 1$.

Since, $\frac{\partial f_s(\boldsymbol{\theta})}{\partial \theta_q} = 0$, for $q \neq s$ the Jacobian is diagonal. Computing its diagonal elements at $\boldsymbol{\theta} = \mathbf{0}$, we have after some algebra

$$\left.\frac{\partial f_s(\boldsymbol{\theta})}{\partial \theta_s}\right|_{\boldsymbol{\theta}=\mathbf{0}} = \frac{2}{\gamma}\frac{\int_{\Re} x_s^2 \exp\left(-\frac{x_s^2}{\gamma}\right) p(x_s)dx_s}{\int_{\Re} \exp\left(-\frac{x_s^2}{\gamma}\right) p(x_s)dx_s} - \frac{2}{\gamma}\frac{\left(\int_{\Re} x_s \exp\left(-\frac{x_s^2}{\gamma}\right) p(x_s)dx_s\right)^2}{\left(\int_{\Re} \exp\left(-\frac{x_s^2}{\gamma}\right) p(x_s)dx_s\right)^2}. \tag{3.19}$$

In addition, due to the fact that $p(x_s)$ is $\mathcal{N}(0, \sigma^2)$, it is easy to verify that

$$\exp\left(-\frac{x_s^2}{\gamma}\right) p(x_s) = \frac{\sigma'}{\sigma}\hat{p}(x_s), \tag{3.20}$$

where $\hat{p}(x_s)$=$\mathcal{N}(0, \sigma'^2)$, with

$$\sigma'^2 = \frac{1}{2\left(\frac{1}{\gamma} + \frac{1}{2\sigma^2}\right)}. \tag{3.21}$$

Substituting eq. (3.20) to eq. (3.19) and taking into account that (a) the numerator of the second fraction is the mean of $\hat{p}(x_s)$, (b) the numerator of the first fraction is the variance of $\hat{p}(x_s)$ and (c) the denominators are both equal to 1, we end up with

$$\left.\frac{\partial f_s(\boldsymbol{\theta})}{\partial \theta_s}\right|_{\boldsymbol{\theta}=\mathbf{0}} = \frac{2\sigma'^2}{\gamma}. \tag{3.22}$$

Substituting eq. (3.21) to eq. (3.22), it is: $\left.\frac{\partial f_s(\boldsymbol{\theta})}{\partial \theta_s}\right|_{\boldsymbol{\theta}=\mathbf{0}} = \frac{2\sigma^2}{2\sigma^2+\gamma}$, which is always less than 1, due to the positivity of σ^2 and γ.

Thus, $\boldsymbol{\theta} = \mathbf{0}$ is a stable fixed point of the iterative scheme $\boldsymbol{\theta}^{t+1} = f(\boldsymbol{\theta}^t)$. □

It is noted that propositions 2 and 3 are valid for both constant and time varying positive γ_j's.

In the general case where the data form more than one dense regions[7], the above propositions are still valid, assuming that the influence on a representative that belongs to a given dense region from data points from other dense regions is negligible. This can be ensured by choosing γ_j's properly. However, an alternative way to achieve this is to equate to zero

[7]That is, when $p(\mathbf{x})$ has more than one peaks.

all u_{ij}'s that are below a certain threshold T. In this way, assuming that the data set under study comprises well-formed and well-separated clusters, each data point $\mathbf{x}_i$ is very likely to have only one $u_{ij} > 0$. In such a case the above convergence analysis remains valid.

Remark: The use of a threshold T for pushing all u_{ij}'s that have values less than T to zero ensures, in general, that a data point $\mathbf{x}_i$ will be compatible with a few at the most clusters. This implies that the vector $\mathbf{u}_i = [u_{i1}, ..., uim]$ has only a few non-zero elements or, in other words, $\mathbf{u}_i$ is *sparse* on non-zero values. The idea of sparsity will be explained in a more sophisticated way in the next chapter.

In the next proposition, we focus on the cluster elimination property of APCM for the case of two representatives that lie in the same physical cluster.

Proposition 4. *Adopt assumption 1 and consider two cluster representatives $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$. Assuming that $\eta_1(t) \neq \eta_2(t)$ and $\eta_j(t) < +\infty$, $j = 1, 2$, $\forall t$, one of the clusters represented by $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ will be eliminated*[8]*.*

Proof. Utilizing Propositions 2 and 3, we have that $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ converge towards $\mathbf{c}$. Thus, the distance between them decreases towards zero, i.e.

$$\|\boldsymbol{\theta}_1(t) - \boldsymbol{\theta}_2(t)\| \to 0. \tag{3.23}$$

Taking into account eq. (3.9), the radius of the hypersphere C_t that delimits $T_1(t)$ and $T_2(t)$ at iteration t can be written as

$$r_t = \frac{\sqrt{\eta_1(t)\eta_2(t)}}{|\eta_1(t) - \eta_2(t)|}\|\boldsymbol{\theta}_2(t) - \boldsymbol{\theta}_1(t)\|. \tag{3.24}$$

From hypothesis it follows that $\frac{\sqrt{\eta_1(t)\eta_2(t)}}{|\eta_1(t)-\eta_2(t)|}$ is finite, i.e.,

$$\exists M > 0 : \left|\frac{\sqrt{\eta_1(t)\eta_2(t)}}{\eta_1(t) - \eta_2(t)}\right| < M \quad \forall t. \tag{3.25}$$

Combining eqs. (3.23), (3.24) and (3.25) we have that $r_t \to 0$. Thus $T_j(t)$ for one of the two representatives will eventually becomes empty, which will lead the corresponding $\eta_j(t)$ to zero value (see eq. (3.4)) and thus to the elimination of cluster C_j (from the execution of steps 14-21 of APCM, Algorithm 4). □

Remark: Note that if $\eta_1(t) = \eta_2(t)$, it is $\frac{\sqrt{\eta_1(t)\eta_2(t)}}{\eta_1(t)-\eta_2(t)} = +\infty$. Thus, in this case r_t does not tend to zero. This implies that none of the representatives will be eliminated, since both $T_1(t)$, $T_2(t)$ remain non-empty. This scenario becomes extremely improbable, due to the

[8]Note that, in practice, the hypothesis for $\eta_1(t)$ and $\eta_2(t)$ is almost always met, due to their definition.

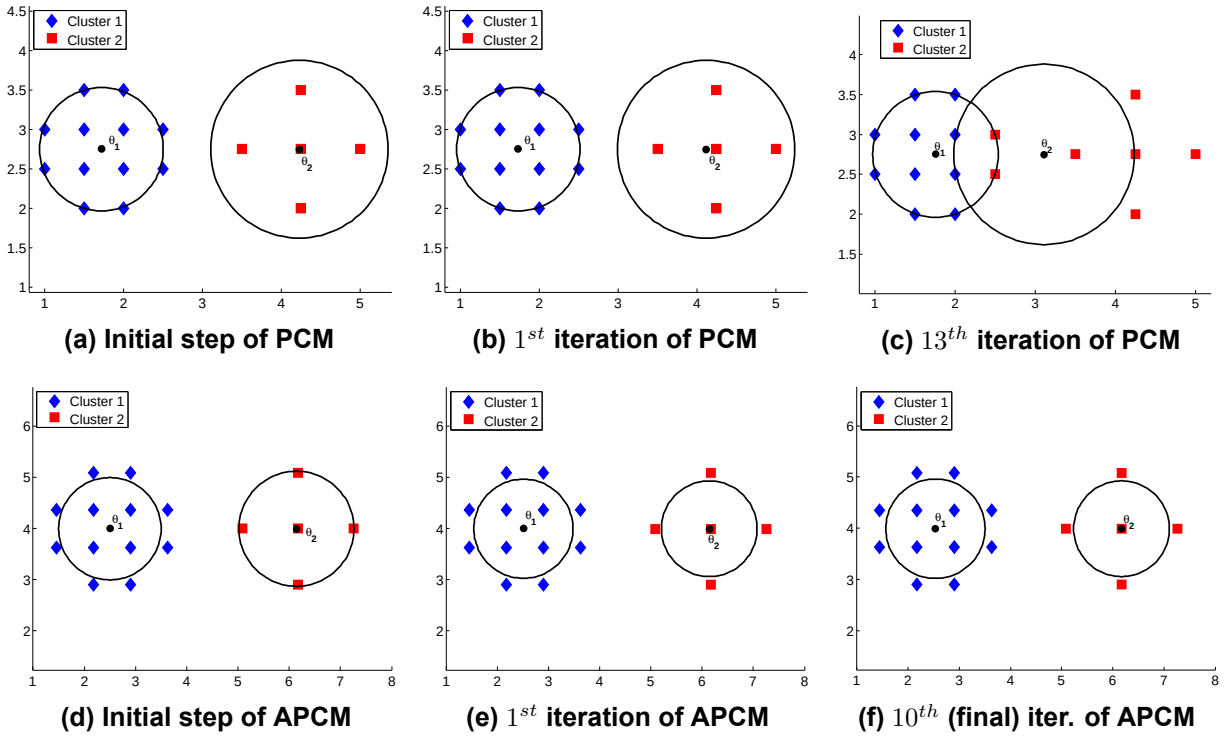


Figure 3.5: PCM and APCM snapshots at their initialization step, 1st iteration and 13th iteration for PCM and 10th (final) iteration for APCM (experiment 1).

updating of η_j via eq. (3.4) (hard computation). On the contrary, this scenario becomes more probable when a soft computation of η_j's is adopted (e.g. $\eta_j = \frac{\sum_{i=1}^{N} u_{ij}\|\mathbf{x}_i - \boldsymbol{\theta}_j\|}{\sum_{i=1}^{N} u_{ij}}$).

3.7 Experimental results

In this section, we assess the performance of the APCM method in several experimental settings and illustrate the obtained results. More specifically, we consider two series of experiments. In the first one, we use two-dimensional simulated data sets in order to exhibit more clearly certain aspects of the behavior of the APCM itself. In the second one, we use both simulated and real-world data sets (Iris [73], New Thyroid [73] to evaluate the performance of APCM in comparison with several other related algorithms.

3.7.1 Clustering behavior of APCM

Experiment 1: Let us consider a two-dimensional data set consisting of $N = 17$ points, which form two natural clusters C_1 and C_2 with 12 and 5 data points, respectively (see Fig. 3.5). The means of the clusters are $\mathbf{c}_1 = [1.75, 2.75]$ and $\mathbf{c}_2 = [4.25, 2.75]$. In this experiment, we consider only the PCM (with $m = 2$) and the APCM (with $m_{ini} = 2$, $\alpha = 1$)

Table 3.1: The degrees of compatibility of the data points of experiment 1 for PCM and APCM algorithms, after: (a) initialization (common to both algorithms), (b) first iteration and (c) 13th iteration for PCM and 10th (final) iteration for APCM.

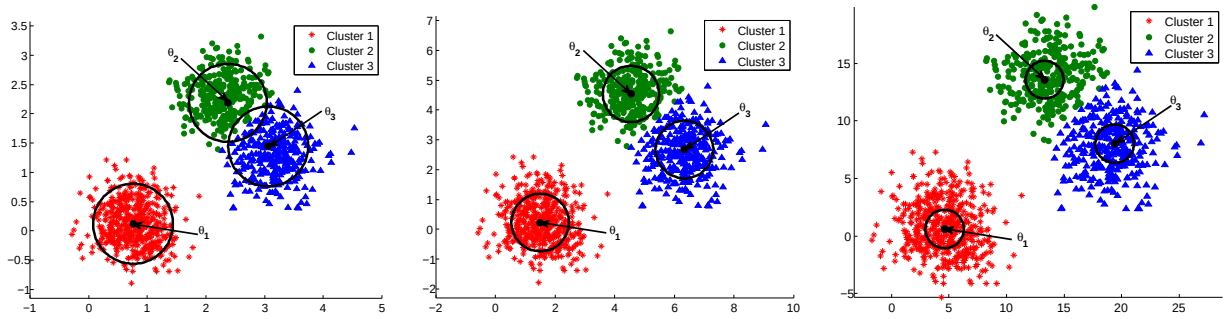
	Initialization		1^{st} iteration				13^{th} iteration		10^{th} iteration	
	PCM/APCM		PCM		APCM		PCM		APCM	
$\mathbf{x}_i$	C_1	C_2	C_1	C_2	C_1	C_2	C_1	C_2	C_1	C_2
$(1.5, 3.5)$	0.9292	0.0708	0.3701	0.0018	0.2757	1.6e-06	0.3604	0.0831	0.2449	3.0e-09
$(2.0, 3.5)$	0.8963	0.1037	0.3526	0.0127	0.2590	9.6e-05	0.3632	0.2428	0.2447	1.3e-06
$(1.0, 3.0)$	0.9475	0.0525	0.3884	2.6e-04	0.2936	2.4e-08	0.3575	0.0284	0.2451	7.2e-12
$(1.5, 3.0)$	0.9854	0.0146	0.8348	0.0027	0.7913	3.4e-06	0.8178	0.1232	0.7550	1.0e-08
$(2.0, 3.0)$	0.9728	0.0272	0.7954	0.0188	0.7432	2.2e-04	0.8192	0.3602	0.7544	4.3e-06
$(2.5, 3.0)$	0.8201	0.1799	0.3360	0.0897	0.2433	0.0060	0.3661	0.7098	0.2445	5.4e-04
$(1.0, 2.5)$	0.9475	0.0525	0.3884	2.6e-04	0.2936	2.4e-08	0.3575	0.0284	0.2451	7.2e-12
$(1.5, 2.5)$	0.9854	0.0146	0.8348	0.0027	0.7913	3.4e-06	0.8128	0.1232	0.7550	1.0e-08
$(2.0, 2.5)$	0.9728	0.0272	0.7954	0.0188	0.7432	2.2e-04	0.8192	0.3602	0.7544	4.3e-06
$(2.5, 2.5)$	0.8201	0.1799	0.3360	0.0897	0.2433	0.0060	0.3661	0.7098	0.2445	5.4e-04
$(1.5, 2.0)$	0.9292	0.0708	0.3701	0.0018	0.2757	1.6e-06	0.3604	0.0831	0.2449	3.0e-09
$(2.0, 2.0)$	0.8963	0.1037	0.3526	0.0127	0.2590	9.6e-05	0.3632	0.2428	0.2447	1.3e-06
$(4.25, 3.5)$	0.0748	0.9252	1.2e-05	0.6415	4.2e-07	0.3903	1.6e-05	0.2302	2.2e-07	0.2563
$(3.5, 2.75)$	0.1441	0.8559	0.0058	0.6566	0.0013	0.4101	0.0071	0.8869	0.0010	0.2600
$(4.25, 2.75)$	6.0e-05	0.9999	3.0e-05	0.9997	1.3e-06	0.9994	4.0e-05	0.3587	7.7e-07	1.0000
$(5.0, 2.75)$	0.0522	0.9478	2.6e-08	0.6267	1.4e-10	0.3715	3.6e-08	0.0597	4.7e-11	0.2527
$(4.25, 2.0)$	0.0748	0.9252	1.2e-05	0.6415	4.2e-07	0.3903	1.6e-05	0.2302	2.2e-07	0.2563

algorithms. Figs. 3.5a and 3.5d show the initial positions of the cluster representatives that are taken from FCM and the circles with radius equal to $\sqrt{\gamma_j}$'s resulting from eq. (2.16) (for $B = 1$) for PCM and from eq. (3.2) for APCM. Similarly, Figs. 3.5b and 3.5e show the new locations of $\boldsymbol{\theta}_j$'s after the first iteration of the algorithms and Figs. 3.5c, 3.5f show the locations of $\boldsymbol{\theta}_j$'s at a later iteration. Table 3.1 shows the degrees of compatibility u_{ij}'s of all data points $\mathbf{x}_i$ with the cluster representatives $\boldsymbol{\theta}_j$'s at the three specific iterations depicted in Fig. 3.5 (initial, 1^{st} for both algorithms, 13^{th} for PCM and 10^{th} (final) for APCM).

As it can be deduced from Table 3.1 and Fig. 3.5, the degrees of compatibility of the data points of C_1 with the cluster representative $\boldsymbol{\theta}_2$ increase as PCM evolves, leading gradually $\boldsymbol{\theta}_2$ towards the region of the cluster C_1 and thus, ending up with two coincident clusters, although $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ are initialized properly through the FCM algorithm (see Fig. 3.5a). This is not the case though with the APCM algorithm, as both the cluster representatives remain in the centers of the actual clusters. Obviously, this differentation on the behavior of the two algorithms is due to the different definition of the parameters γ_j's, which affect the degrees of compatibility of the data points with each cluster (see eqs. (2.16), (3.4) and (2.13)). This experiment indicates that, in principle, APCM can handle successfully cases where relatively closely located clusters with different densities are involved.

In the next experiment, we investigate on the relation between m_{ini} and parameter α.

Experiment 2: Let us consider now a two-dimensional data set consisting of $N = 1100$ points, which form three natural clusters C_1, C_2 and C_3 (see Fig. 3.6). Each such cluster is modelled by a normal distribution. The (randomly generated) means of the distributions are $\mathbf{c}_1 = [1.35, 0.23]^T$, $\mathbf{c}_2 = [4.03, 4.09]^T$ and $\mathbf{c}_3 = [5.64, 2.28]^T$, respectively, while their (common) covariance matrix is set equal to $0.4 \cdot I_2$, where I_2 is the 2×2 identity matrix. A number of 500 points is generated by the first distribution and 300 points are generated by



(a) $\alpha = 0.5$ **(b)** $\alpha = 1.0$ **(c)** $\alpha = 3.0$

Figure 3.6: The clustering results of APCM for experiment 2, when it is initialized with $m_{ini} = 5$, for several values of parameter α.

Table 3.2: Range of values of the parameter α, in which APCM concludes correctly to $m_{final} = 3$ clusters, for specific values of m_{ini} for experiment 2.

m_{ini}	α_{min}	α_{max}
3	0.35	5.00
5	0.33	3.08
10	0.28	1.38
20	0.23	0.90
50	0.17	0.36
100	0.15	0.29

each one of the other two distributions. Note that clusters C_2 and C_3 lie very close to each other and, therefore, their discrimination is considered as a difficult task for a clustering algorithm. Table 3.2 shows the ranges of values of the parameter α, for which APCM manages to identify correctly the naturally formed $m = 3$ clusters, for various values of m_{ini}. Fig. 3.6 shows the clustering results of the APCM algorithm, when it is initialized with $m_{ini} = 5$, in cases where (a) $\alpha = 0.5$, (b) $\alpha = 1.0$ and (c) $\alpha = 3.0$, respectively. Note from Table 3.2, that these values of parameter α belong to the range where APCM identifies correctly the actual clusters, when $m_{ini} = 5$. Also, in Fig. 3.6, it is shown how γ_j's are affected when varying the parameter α, after APCM is initialized with $m_{ini} = 5$.

Running APCM on the previous data set, for various values of m_{ini} and α, we end up with Fig. 3.7, where regions in the $\alpha - m_{ini}$ plot are drawn with different colors, each one corresponding to a different number of final clusters, m_{final}. The light-blue colored region corresponds to the case where $m_{final} = 3$, i.e., when APCM identifies correctly the underlying clusters. From the shape of this region, we can verify the "rule of thumb" stated already in Section 3.5, that is, α is inversely related to m_{ini}. Moreover, from Fig. 3.7, we deduce that by fixing α to a value arround 1 and taking m_{ini} $3 - 4$ times greater than the actual number of clusters, APCM will identify correctly the underlying physical clusters. Interestingly, the situation depicted in Fig. 3.7 has also been observed for several other data sets. Thus, the above rule of thumb seems to hold more generally.

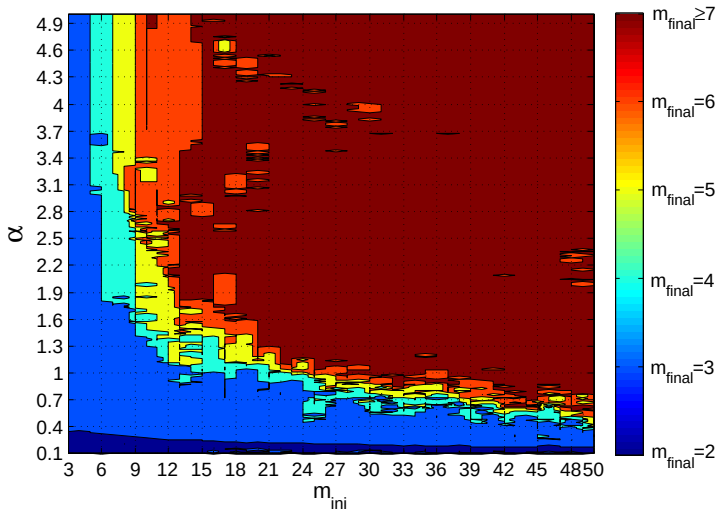


Figure 3.7: Graphical representation of the number of final clusters, m_{final}, returned by APCM for experiment 2, for various combinations of α and m_{ini} [9].

3.7.2 Comparison of APCM with state-of-the-art clustering algorithms

In the sequel, we compare the clustering performance of APCM with that of the k-means, the FCM, the FCM with the XB validity index [22], the PCM, the UPC [20], the PFCM [19], the UPFC [28], the GRPCM [31] and the AMPCM [30] algorithms, which all result from cost optimization schemes. For a fair comparison, the representatives $\boldsymbol{\theta}_j$'s of all algorithms, except for GRPCM and AMPCM, are initialized based on the FCM scheme and the parameters of each algorithm are first fine-tuned. In order to compare a clustering with the true data label information, we use (a) the Rand Measure (RM) (e.g. [18]), which measures the degree of agreement between the obtained clustering and the true data classification and can handle clusterings whose number of clusters may differ from the number of true data labels, and (b) the Success Rate (SR), which measures the percentage of the points that have been correctly labeled by each algorithm. Moreover, the mean of the Euclidean distances (MD) between the true mean of each physical cluster $\mathbf{c}_j$ and its closest cluster representative ($\boldsymbol{\theta}_j$) obtained by each algorithm, is given[10]. In cases where a clustering algorithm ends up with a higher number of clusters than the actual one ($m_{final} > m$), only the m cluster representatives that are closest to the true m centers of the physical clusters, are taken into account in the determination of MD. On the other hand, in cases where $m_{final} < m$, the MD measure refers to the distances of all cluster representatives from their nearest actual center; thus some actual centers are ignored[11]. It is noted that lower MD values indicate more accurate determination of the cluster center locations. Finally,

[9] Note that for each value of m_{ini} the same initial representatives (produced by FCM) have been used, for all values of α. Results may differ slightly for different initializations of APCM.

[10] This is also called "quantization distortion" in centroid-based methods, provided that the number of $\mathbf{c}_j$'s and $\boldsymbol{\theta}_j$'s are the same.

[11] In such cases, increased MD values are expected, indicating the fact that some actual clusters have not been identified.

the number of iterations and the time (in seconds) required for the convergence of each algorithm, are provided[12]. Note that in all reported results for the UPC, the PFCM and the UPFC algorithms, clusters that coincide are considered as a single one. Moreover, for the FCM with XB validity index case, only the clustering obtained for the m_{ini} that minimizes the XB index is given and discussed. All algorithms have been executed using MATLAB R2013a on an Intel i7-4790 machine with 16 GB RAM and 3.60 GHz.

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

Experiment 3: Consider a two-dimensional data set consisting of $N = 2100$ points, where three natural clusters C_1, C_2 and C_3 are formed. Each such cluster is modelled by a normal distribution. The means of the distributions are $\mathbf{c}_1 = [6.53, 1.39]^T$, $\mathbf{c}_2 = [20.32, 20.39]^T$ and $\mathbf{c}_3 = [28.09, 11.38]^T$, respectively, while their covariance matrices are set to $10 \cdot I_2$, $20 \cdot I_2$ and I_2, respectively. A number of 1000 points are generated by each one of the first two distributions and 100 points are generated by the last one. Moreover, 200 data points are added randomly as noise in the region where data live (see Fig. 3.8a).

Table 3.3 shows the clustering results of all algorithms, where m_{ini} and m_{final} denote the initial and the final number of the obtained clusters, respectively. Fig. 3.8b and Fig. 3.8c show the clustering result obtained using the k-means and FCM algorithms, respectively, for $m_{ini} = 3$. Figs. 3.8d, 3.8e, 3.8f, 3.8g, 3.8h, 3.8i, 3.8j and 3.8k depict the performance of FCM & XB, PCM, APCM, UPC, PFCM, UPFC, GRPCM and AMPCM respectively, with their parameters chosen as stated in the figure caption. In addition, the circles, centered at each $\boldsymbol{\theta}_j$ and having radius $\sqrt{\gamma_j}$ (as they have been computed after the convergence of the algorithms), are also drawn.

As it can be deduced from Fig. 3.8 and Table. 3.3, even when the k-means and the FCM are initialized with the (unknown in practice) true number of clusters ($m = 3$), they fail to unravel the underlying clustering structure, most probably due to the noise encountered in the data set and the big difference in the variances between nearby clusters. The FCM & XB validity index and the classical PCM also fail to detect the cluster with the smallest variance. On the other hand, the proposed APCM algorithm produces very accurate results for various initial values of m_{ini}, estimating with high accuracy the center of the actual clusters (see the corresponding MD measure in Table 3.3). The UPC algorithm has been exhaustively fine tuned so that the parameters γ_j's, which remain fixed during its execution and are the same for all clusters, get small enough values, in order to identify the cluster with the smallest variance (C_3). However, under these circumstances, a representative that is initially placed at the region where only noisy points exist (due to bad initialization from FCM), is trapped there and cannot be moved towards a dense region (due to the small value of its γ_j). Thus, UPC concludes to 4 clusters when $q = 3$, but if we set $q = 2$, UPC will conclude to 2 clusters, identifying C_1 and C_2 and missing C_3. The PFCM and UPFC algorithms constantly produce 3 clusters, at the cost of a computationally demanding fine tuning of the (several) parameters they involve (not included in the last column

[12] In the FCM & XB validity index only the total time required for the execution of FCM 19-times (for $m_{ini} = 2, \ldots, 20$) is given.

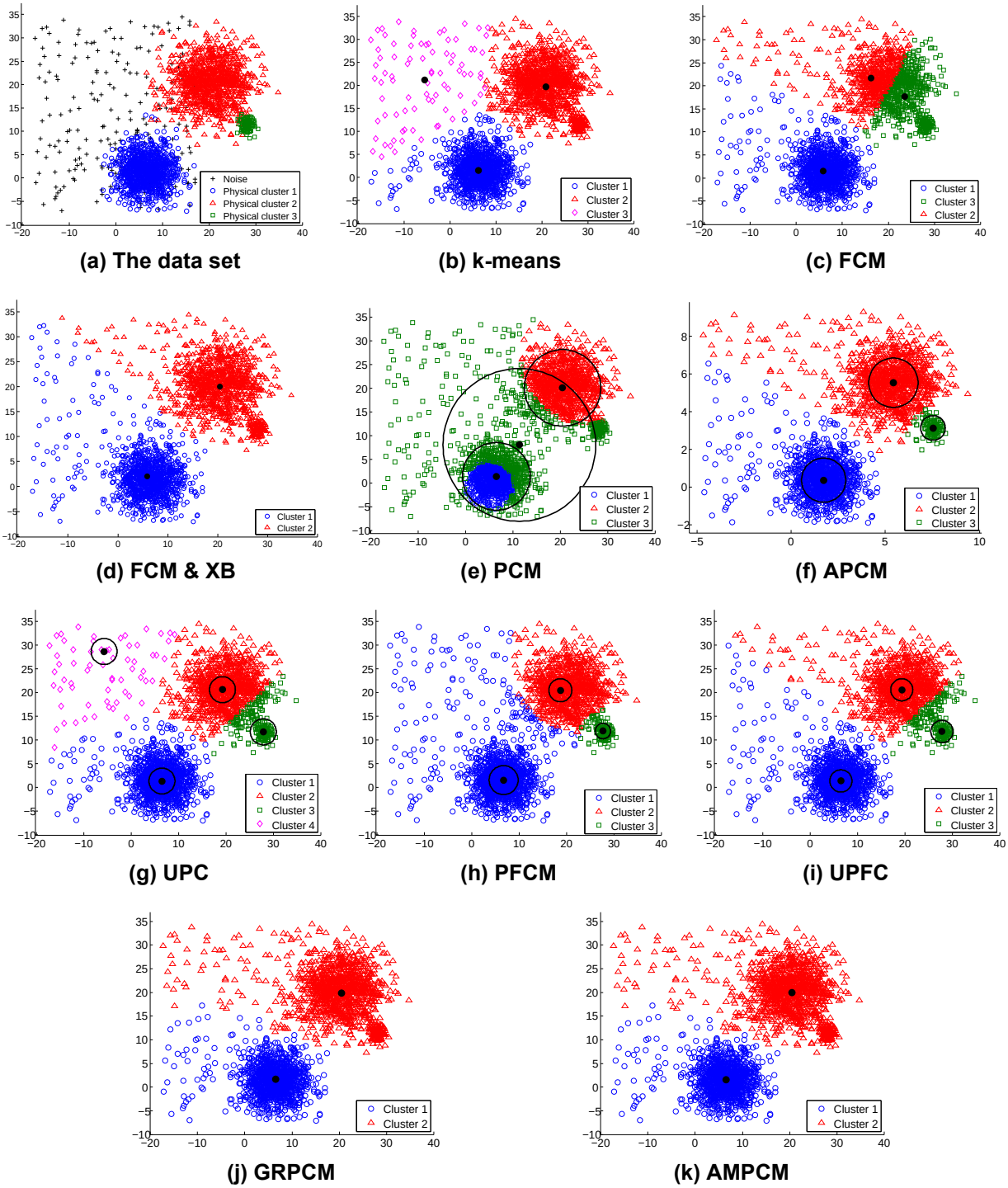


Figure 3.8: (a) Data set of experiment 3. Clustering results for (b) k-means, $m_{ini} = 3$, (c) FCM, $m_{ini} = 3$, (d) FCM & XB, (e) PCM, $m_{ini} = 15$, (f) APCM, $m_{ini} = 15$ and $\alpha = 1$, (g) UPC, $m_{ini} = 8$ and $q = 3$, (h) PFCM, $m_{ini} = 15$, $K = 1$, $\alpha = 1$, $\beta = 3$, $q = 2.5$ and $n = 2$, (i) UPFC, $m_{ini} = 15$, $\alpha = 1$, $\beta = 1.5$, $q = 3$ and $n = 2$, (j) GRPCM and (k) AMPCM.

Table 3.3: Performance of clustering algorithms for the experiment 3 data set.

	m_{ini}	m_{final}	RM	SR	MD	$Iter$	$Time$
k-means	3	3	91.02	86.74	6.8509	45	0.13
k-means	8	8	73.83	42.22	2.5267	60	0.33
k-means	10	10	71.41	34.52	2.3544	48	0.41
k-means	15	15	68.35	27.00	0.8074	31	0.46
FCM	3	3	82.05	65.39	4.2089	66	0.04
FCM	8	8	71.88	36.91	2.5468	100	0.34
FCM	10	10	69.67	28.74	2.3466	100	0.48
FCM	15	15	67.18	21.96	0.8593	100	0.50
FCM & XB	-	2	87.62	86.74	0.6346	-	3.60
PCM	3	2	87.62	86.78	0.4778	10	0.14
PCM	8	3	75.14	67.87	0.2138	26	0.59
PCM	10	3	75.64	68.35	0.1918	23	0.74
PCM	15	3	78.64	70.04	0.1877	41	1.06
APCM ($\alpha = 1$)	3	2	87.73	86.83	0.0655	12	0.07
APCM ($\alpha = 1.5$)	8	3	**90.83**	**90.04**	0.2268	38	0.40
APCM ($\alpha = 1$)	10	3	90.80	90.00	0.2131	28	0.52
APCM ($\alpha = 1$)	15	3	**90.83**	**90.04**	0.2157	35	0.55
UPC ($q = 2$)	3	2	87.69	86.78	0.1331	20	0.07
UPC ($q = 3$)	8	4	90.04	85.96	0.5517	76	0.54
UPC ($q = 3$)	10	4	89.92	85.78	0.5829	89	0.57
UPC ($q = 3$)	15	4	89.79	85.61	0.6618	111	0.80
PFCM ($K = 1$, $a = 1$, $b = 1$, $q = 2$, $n = 2$)	3	2	87.62	86.78	1.2927	25	0.07
PFCM ($K = 1$, $a = 1$, $b = 1$, $q = 4$, $n = 2$)	8	3	83.11	84.65	0.5595	55	0.47
PFCM ($K = 1$, $a = 1$, $b = 2$, $q = 3$, $n = 2$)	10	3	84.30	85.78	0.7517	119	0.74
PFCM ($K = 1$, $a = 1$, $b = 3$, $q = 2.5$, $n = 2$)	15	3	86.74	87.70	0.8414	201	1.83
UPFC ($a = 1$, $b = 1$, $q = 4$, $n = 2$)	3	2	87.76	86.83	0.4588	20	0.08
UPFC ($a = 1$, $b = 3$, $q = 3$, $n = 2$)	8	3	87.39	85.43	0.7260	85	0.49
UPFC ($a = 1$, $b = 3$, $q = 3$, $n = 2$)	10	3	87.40	85.43	0.7364	101	0.68
UPFC ($a = 1$, $b = 1.5$, $q = 3$, $n = 2$)	15	3	87.64	85.91	0.5555	94	0.83
GRPCM	-	2	87.54	86.74	0.3611	90	148.03
AMPCM	-	2	87.54	86.74	0.3189	87	151.64

of Table 3.3). However, even when their parameters are fine tuned, the final estimates of $\boldsymbol{\theta}_j$'s are not closely located to the true cluster centers (see MD measure in Table 3.3). The GRPCM and AMPCM algorithms conclude to two clusters, failing to unravel the underlying clustering structure. It is worth noting that these two algorithms require too much time to converge, mainly due to the way they perform cluster elimination. Finally, as it is deduced from Table 3.3, the APCM algorithm achieves the best RM and SR results, estimating more accurately the true centers of the clusters (minimum MD), while, in addition, it requires the fewest iterations for convergence. It is worth noting that the operation time of APCM is less than that of PCM, even when APCM requires more iterations than PCM to converge. This is so because the APCM iterations become "lighter" as the algorithm evolves, since several clusters are eliminated.

The last two experiments are conducted on the basis of real world data sets.

Experiment 4: Let us consider the Iris data set ([73]) consisting of $N = 150$, 4-dimensional

Table 3.4: Performance of clustering algorithms for the Iris data set.

	m_{ini}	m_{final}	RM	SR	MD	$Iter$	$Time$
k-means	3	3	87.97	89.33	0.1271	3	0.30
k-means	10	10	76.64	40.00	0.7785	4	0.13
FCM	3	3	87.97	89.33	0.1287	19	0.02
FCM	10	10	76.16	36.00	0.7793	35	0.02
FCM & XB	-	2	76.37	66.67	0.3986	-	0.16
PCM	3	2	77.19	66.67	0.3563	19	0.11
PCM	10	2	77.63	66.67	0.3488	28	0.11
APCM ($\alpha = 3$)	3	3	**91.24**	**92.67**	0.1406	26	0.06
APCM ($\alpha = 1$)	10	3	84.15	84.67	0.4030	67	0.09
UPC ($q = 4$)	3	3	**91.24**	**92.67**	0.1438	26	0.03
UPC ($q = 2.4$)	10	3	81.96	81.33	0.5569	150	0.11
PFCM ($K = 1$, $a = 1$, $b = 10$, $q = 7$, $n = 2$)	3	3	**90.55**	**92.00**	0.1833	17	0.03
PFCM ($K = 1$, $a = 1$, $b = 1.5$, $q = 2$, $n = 2$)	10	3	84.64	85.33	0.5411	92	0.05
UPFC ($a = 1$, $b = 5$, $q = 4$, $n = 2$)	3	3	**91.24**	**92.67**	0.1642	32	0.03
UPFC ($a = 1$, $b = 1.5$, $q = 2.5$, $n = 2$)	10	3	81.96	81.33	0.5566	180	0.16
GRPCM	-	2	77.63	66.67	0.3675	26	0.47
AMPCM	-	2	77.63	66.67	0.3643	28	0.47

data points that form three classes, each one having 50 points. In this 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 3.4, k-means and FCM work well, only if they are initialized with the true number of clusters ($m_{ini} = 3$). The FCM & XB and the classical PCM fail to end up with $m_{final} = 3$ clusters, independently of the initial number of clusters. The same result holds for the GRPCM and the AMPCM algorithms. On the contrary, the APCM, the UPC, the PFCM and the UPFC algorithms, after appropriate fine tuning of their parameters, produce very accurate results in terms of RM, SR and MD. However, the APCM algorithm estimates more accurately the centers of the true clusters (in most cases), compared to the other algorithms. It is noted again that the main drawback of the PFCM and the UPFC algorithms is the requirement for fine tuning of several parameters, which increases excessively the computational load required for detecting the appropriate combination of parameters that achieves the best clustering performance.

Experiment 5: Let us consider now the so-called New Thyroid three-class data set ([73]) consisting of $N = 215$, 5-dimensional data points. The experimental results for all algorithms are shown in Table 3.5. It can be seen that both k-means and FCM provide satisfactory results, only if they are initialized with the true number of clusters ($m_{ini} = 3$), and the XB validity index is correctly minimized for $m_{ini} = 3$, thus FCM & XB concludes to the same results as FCM for $m_{ini} = 3$, however at the cost of increased computational time. The classical PCM exhibits degraded performance, for all choices of m_{ini}. Similar to PCM behavior is observed for the GRPCM and the AMPCM algorithms, which fail to distinguish any clustering structure. On the contrary, the APCM and UPC algorithms detect the actual number of clusters independently of m_{ini} after appropriate fine tuning of their parameters. However, again the APCM algorithm constantly produces higher RM and SR values. Finally, the PFCM and UPFC exhibit (a) inferior performance compared to APCM and UPC and (b) superior performance with respect to k-means and FCM provided that

Table 3.5: Performance of clustering algorithms for the New Thyroid data set.

	m_{ini}	m_{final}	RM	SR	MD	$Iter$	$Time$
k-means	3	3	79.65	87.44	0.8949	3	0.16
k-means	5	5	70.78	63.72	0.8548	12	0.14
k-means	15	15	55.01	25.12	0.7159	16	0.17
FCM	3	3	83.29	89.77	0.4385	53	0.02
FCM	5	5	60.32	46.98	1.0785	55	0.02
FCM	15	15	52.83	21.86	0.8816	91	0.11
FCM & XB	-	3	83.29	89.77	0.4385	-	0.44
PCM	3	1	53.05	69.77	0.1177	7	0.06
PCM	5	1	53.05	69.77	0.0559	7	0.06
PCM	15	1	53.05	69.77	0.0577	8	0.16
APCM ($\alpha = 8$)	3	3	**94.58**	**96.74**	0.7231	30	0.08
APCM ($\alpha = 3$)	5	3	87.59	92.56	1.0026	21	0.06
APCM ($\alpha = 1.2$)	15	3	73.73	83.72	2.7123	54	0.16
UPC ($q = 3$)	3	3	83.85	90.23	0.6982	41	0.03
UPC ($q = 2$)	5	3	77.94	86.51	1.0739	16	0.02
UPC ($q = 1$)	15	3	67.21	79.53	2.7617	34	0.05
PFCM ($K = 1$, $a = 1$, $b = 5$, $q = 8$, $n = 2$)	3	1	53.05	69.77	0.0507	15	0.03
PFCM ($K = 1$, $a = 1$, $b = 5$, $q = 8$, $n = 2$)	5	2	64.95	77.21	1.3855	41	0.05
PFCM ($K = 1$, $a = 1$, $b = 8$, $q = 2$, $n = 2$)	15	3	66.64	79.07	1.8381	28	0.09
UPFC ($a = 1$, $b = 5$, $q = 8$, $n = 2$)	3	2	68.21	79.53	0.4108	21	0.05
UPFC ($a = 1$, $b = 3$, $q = 6$, $n = 2$)	5	3	78.76	86.98	0.9682	27	0.05
UPFC ($a = 1$, $b = 0.1$, $q = 1.5$, $n = 2$)	15	3	72.85	83.26	1.5909	34	0.09
GRPCM	-	1	53.05	69.77	0.2732	63	2.04
AMPCM	-	1	53.05	69.77	0.2667	64	1.98

the latter are not initialized with the correct number of clusters.

3.8 Conclusions

In this chapter, commencing from the classic possibilistic c-means (PCM) algorithm proposed in [10], a new possibilistic clustering algorithm, called Adaptive Possibilistic c-means (APCM), has been presented. APCM addresses several of the weaknesses of PCM and exhibits several new features. The most important one is that its parameters γ are adapted as the algorithm evolves, in contrast to all other related possibilistic algorithms, where parameters γ, once they are set, they remain fixed. This makes APCM more flexible in tracking the variations in the cluster formation as it evolves. Additional significant features are related with the computation of the parameters γ. Specifically, in contrast to previous possibilistic algorithms, each γ_j is expressed in terms of the mean absolute deviation of the vectors that are most compatible with the jth cluster (C_j), from their mean. The use of the Euclidean distance, instead of the squared Euclidean one, makes the algorithm capable to distinguish closely located to each other clusters. Moreover, the use of the mean $\boldsymbol{\mu}_j$ of the most compatible points to a certain cluster C_j instead of the previous location of the corresponding representative $\boldsymbol{\theta}_j$ in the computation of γ_j's leads to better estimates for the latter. A significant side-effect of the adaptation of γ_j's is

that APCM is (in principle) capable to reveal the true number, m, of physical clusters via a cluster elimination procedure, provided that it is initialized with an overestimate of it, m_{ini}. The latter releases APCM from the noose of knowing exactly in advance the true number of "physical" clusters at the cost of performing a light fine tuning for specifying the value of the parameter α involved in the definition of γ_j's. It is worth noting that as experiments show, m_{ini} and α should vary inversely to each other, in order for the algorithm to work properly, a fact that makes their choice not entirely arbitrary. In addition, they show that if α is fixed to a value around 1 and m_{ini} is around 3-4 times greater than m, then, in several cases, the algorithm works properly. Finally, the experimental results provided show that APCM exhibits superior performance compared to several other related algorithms, in almost all the considered data sets.

4. SPARSE POSSIBILISTIC C-MEANS ALGORITHM

4.1 Introduction

In this chapter we extent the classical PCM algorithm, proposed in [10], in a different way from that presented in the previous section. In contrast to the APCM algorithm, where the parameters γ were adapted during its execution, here they are considered fixed. The present extension relies on the fact that, in practice, each data vector is compatible with only a few or even *none* clusters. Based on this, a suitable sparsity constraint is imposed on the vector containing the degrees of compatibility of each data vector with all the clusters, giving rise to the *Sparse PCM* (SPCM) algorithm [56]. SPCM exhibits increased immunity to data points that may be considered as noise or outliers by excluding them, in principle, from contributing to the estimation of the cluster representatives. As a consequence, SPCM concludes to more accurate estimates for the cluster representatives than PCM, especially in noisy enviroments.

Moreover, in difficult cases, where the physical clusters underlying in the data set under study are very closely located to each other, SPCM allows only the data points that are very close to the current location of the representatives to contribute to the estimation of the next location of the latter. As a result, SPCM is, in principle, capable of identifying very closely located clusters of possibly various densities. However, the requirement of the estimation of the specific parameters γ involved in all PCMs still remains.

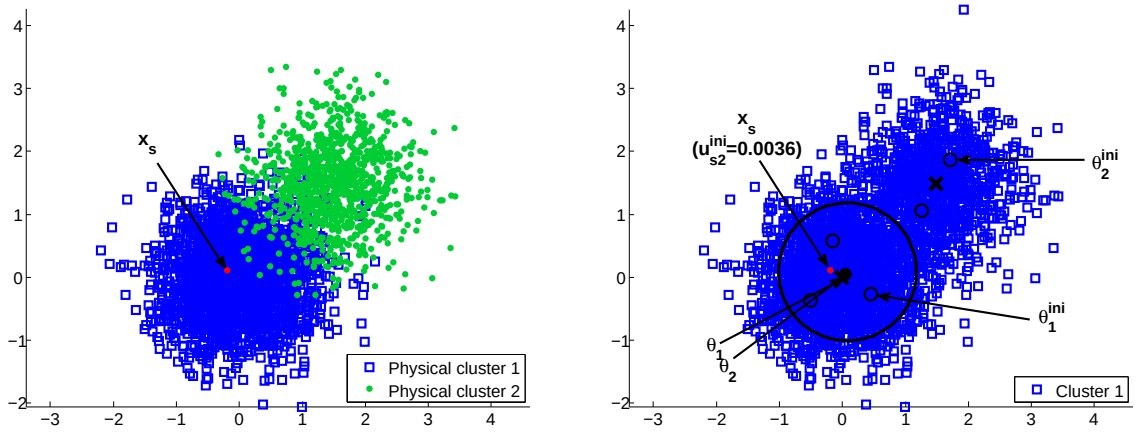
It is worth noting that the proposed method is not the first one that introduces the sparsity idea in clustering. Sparsity in the, so-called, outlier domain has been proposed previously in [74], [69]. Also in [75], [76], two variants of possibilistic clustering that impose sparsity constraints, adopting the l_1 norm, are proposed. In [76] the clusters are recovered in a sequential manner, in contrast to [75], where clusters are recovered simultaneously.

In the sequel we proceed as follows. In Section 4.2 we introduce the concept of sparsity. In Sections 4.3-4.6 we present in detail the SPCM algorithm and in Section 4.7 we focus on an experiment for collating SPCM with PCM. Finally, Section 4.8 contains the convergence analysis of the algorithm, where it is proved that the algorithm converges to a local minimum of its cost function. Also, the same result is established for PCM as a special case of the convergence proof of SPCM. Conclusions are presented in Section 4.9. Note that further extended experimental results of SPCM are presented in the next chapter.

4.2 Enforcing Sparsity - The Sparse PCM (SPCM)

A notable feature of the PCM algorithm is that *all* data vectors contribute to the updating of the representatives (see eq. (2.15)) since, from eq. (2.13), we have that all u_{ij}'s are positive. When the physical clusters are well separated from each other, the updating of a specific $\boldsymbol{\theta}_j$ will only slightly be affected by distant from it data points. However, in the case

where the physical clusters are closely located to each other and have different densities, the affection on $\boldsymbol{\theta}_j$ from data points that belong to other physical clusters will be increased. Moreover, the affection will be higher for the representative of the less dense cluster. That may drive this representative towards the center of the denser cluster, failing thus to identify the less dense cluster. Note that even if this does not happen, the corresponding final estimates of $\boldsymbol{\theta}_j$'s will represent less accurately the physical cluster centers. The previous arguments are illustrated qualitatively via the following two examples[1].



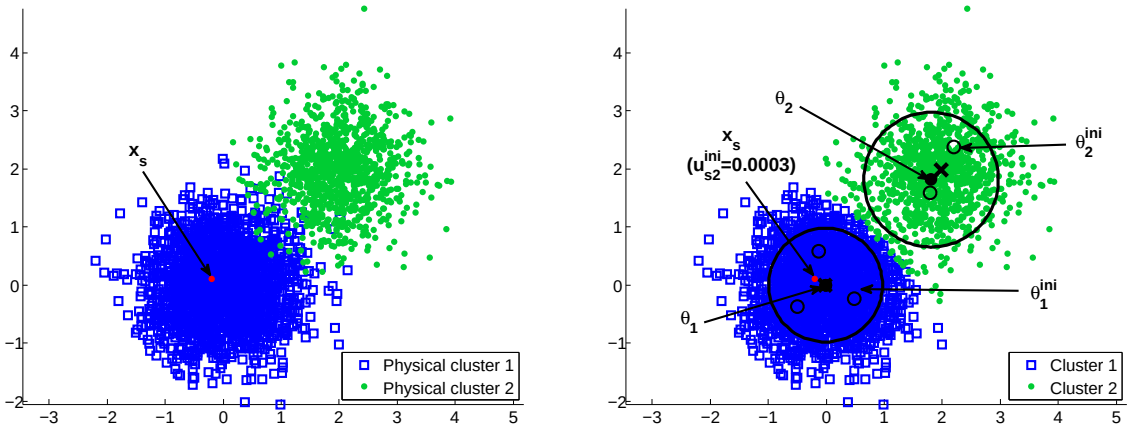
(a) Data set of Example 1 **(b) PCM**

Figure 4.1: (a) The data set of Example 1 and (b) the clustering result of Example 1 for PCM with $m = 5$ ($m_{final} = 2$). Open (closed) circles stand for the initial (final) location of the representatives ($\boldsymbol{\theta}_j$'s) and crosses represent the true centers of the clusters ($\mathbf{c}_j$'s). The circles centered at each $\boldsymbol{\theta}_j$ and having radius $\sqrt{\gamma_j}$ are also drawn. $\mathbf{x}_s$ is a specific typical point of C_1 that will also be considered in Figs. 4.2 and 4.5 and u_{s2}^{ini} is its corresponding degree of compatibility with $\boldsymbol{\theta}_2^{ini}$.

Example 1: Consider a two-dimensional data set X consisting of $N = 3000$ points, where two physical clusters C_1 and C_2 are formed. The clusters are modelled by normal distributions with means $\mathbf{c}_1 = [0, 0]^T$ and $\mathbf{c}_2 = [1.5, 1.5]^T$, respectively, while their covariance matrices are both equal to $0.4 \cdot I_2$, where I_2 is the 2×2 identity matrix. A number of 2000 points of X is generated by the first distribution and 1000 points are generated by the second one. Note that the clusters share the same covariance matrix, they are located very close to each other and they have different densities, as shown in Fig. 4.1a. The clustering result of the PCM, initialized with $m = 5$ clusters, is shown in Fig. 4.1b. Apparently, PCM fails to uncover the less dense cluster. To see qualitatively why this happens let us focus on $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ in Fig. 4.1b. As it can be seen, $\boldsymbol{\theta}_2$ was finally attracted towards C_1, although it was initially placed in C_2. This occurred because in the process of determining the next location of $\boldsymbol{\theta}_2$, the many small contributions from the data points of C_1 gradually prevailed over the larger but less contributions from the data points of C_2 (see eqs. (2.13), (2.15)).

Example 2: Consider now the same two-dimensional data set of Example 1, where now the two normal distributions are more distant from each other with means $\mathbf{c}_1 = [0, 0]^T$ and $\mathbf{c}_2 = [2, 2]^T$, respectively (see Fig. 4.2a). As is shown in Fig. 4.2b, PCM now succeeds in

[1] A more quantitative illustration is given in Experiment 1.



(a) Data set of Example 2 **(b) PCM**

Figure 4.2: (a) The data set of Example 2 and (b) the clustering result of Example 2 for PCM with $m = 5$ ($m_{final} = 2$). Note that the contribution of the typical point $\mathbf{x}_s$ to the computation of θ_2^{ini} is now much smaller compared to its counterpart in Fig. 4.1b. See also the caption of Fig. 4.1.

identifying both clusters. It seems that, in determining the next location of $\boldsymbol{\theta}_2$, the many small contributions from the data points of C_1 were much smaller than their counterparts in Example 1 and they did not succeed to prevail over the larger but less contributions from the data points of C_2. However, the final estimates of the true centers (means of the Gaussians) are not very accurate, as shown qualitatively in Fig. 4.2b and established quantitatively later in Table 4.1.

One way to face situations, such as those encountered in Examples 1 and 2, is to suppress the contribution in the updating of representatives from data points that are distant from them. Focusing on a specific representative $\boldsymbol{\theta}_j$, this can be achieved by imposing $u_{ij} = 0$ for data points $\mathbf{x}_i$ whose distance from $\boldsymbol{\theta}_j$ is sufficiently large. Recalling that $\mathbf{u}_i^T = [u_{i1}, \ldots, u_{im}]$, $i = 1, \ldots, N$, this is tantamount to imposing sparsity on $\mathbf{u}_i$, i.e., forcing the corresponding data point $\mathbf{x}_i$ to contribute only to its (currently) closest representatives. To incorporate sparsity in PCM, we propose the following extension of the cost function J_{PCM} given in eq. (2.11),

$$J_{SPCM}(\Theta, U) = \sum_{j=1}^{m} \left[\sum_{i=1}^{N} u_{ij} \|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2 + \gamma_j \sum_{i=1}^{N} (u_{ij} \ln u_{ij} - u_{ij}) \right] + \lambda \sum_{i=1}^{N} \|\mathbf{u}_i\|_p^p, \; u_{ij} > 0, \quad (4.1)$$

where $\|\mathbf{u}_i\|_p$ is the ℓ_p-norm of vector $\mathbf{u}_i$ with $p \in (0, 1)$, defined as $\|\mathbf{u}_i\|_p^p = \sum_{j=1}^{m} u_{ij}^p$ [2]. The last term in eq. (4.1) is expected to induce sparsity on each one of the vectors $\mathbf{u}_i$, while λ (≥ 0) is a regularization parameter that controls the degree of the imposed sparsity. The selection of the value of λ, which remains constant during the execution of the algorithm, is discussed in subsection 4.5. It is clear that by setting $\lambda = 0$, we end up with the cost

[2] The condition $u_{ij} > 0$ is a prerequisite, in order for $\ln u_{ij}$ to be well-defined. However, in the sequel, when referring to $u_{ij} \ln u_{ij}$ for $u_{ij} = 0$, we mean $\lim_{u_{ij} \to 0^+} u_{ij} \ln u_{ij} (= 0)$.

function which is associated with the classical PCM (eq. (2.11)). The algorithm resulting by the minimization of $J_{SPCM}(\Theta, U)$ is called sparse possibilistic c-means (SPCM) clustering algorithm.

We describe next in detail the various stages of the SPCM algorithm. Specifically, we first describe the way its parameters are initialized. Next, the updating of u_{ij}'s and $\boldsymbol{\theta}_j$'s is considered. Note that the updating of $\boldsymbol{\theta}_j$'s is the same as in classical PCM, while, the updating of u_{ij}'s is quite different. Although the latter is more complicated than in the classical PCM, proposed in [10], at the same time, it is far more simpler[3] than the updating in other problems where sparsity is induced through the ℓ_p-norm with $0 < p < 1$.

4.3 Initialization in SPCM

First, we make an overestimation, denoted by m_{ini}, of the true number of clusters m, underlying in the data set. Regarding $\boldsymbol{\theta}_j$'s, their initialization drastically affects the final clustering result in PCM. Thus, a good starting point for them is of crucial importance. Ideally, we would like to have at least one representative in the region of each physical cluster. 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. Taking into account that FCM is likely to drive the representatives to "dense in data" regions (since $m_{ini} > m$), we have a good probability of placing at least one of the initial $\boldsymbol{\theta}_j$'s in each dense region (cluster) of the data set.

After the initialization of $\boldsymbol{\theta}_j$'s, we initialize γ_j's utilizing eq. (2.16) for $B = 1$.

4.4 Updating of $\boldsymbol{\theta}_j$'s and u_{ij}'s in SPCM

Minimization of $J_{SPCM}(\Theta, U)$ with respect to $\boldsymbol{\theta}_j$ leads to the same updating equation as in the original PCM scheme (eq. (2.15)), since the last term added to the cost function does not depend on $\boldsymbol{\theta}_j$'s. It is only the updating of u_{ij}'s that will be modified, in the light of the last term of $J_{SPCM}(\Theta, U)$. Taking the derivative of $J_{SPCM}(\Theta, U)$ with respect to u_{ij}, we obtain

$$\frac{\partial J_{SPCM}(\Theta, U)}{\partial u_{ij}} \equiv f(u_{ij}) = d_{ij} + \gamma_j \ln u_{ij} + \lambda p u_{ij}^{p-1}, \tag{4.2}$$

where $d_{ij} = \|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2$. Obviously, $\frac{\partial J_{SPCM}(\Theta, U)}{\partial u_{ij}} = 0$ is equivalent to $f(u_{ij}) = 0$, the solution of which will give the requested u_{ij}. Clearly, this equation cannot be solved analytically. However, it can be efficiently solved arithmetically based on the following propositions.

Proposition 1. *$f(u_{ij})$ does not become zero for $u_{ij} \in (-\infty, 0) \cup (1, +\infty)$.*

[3]Note that, as it will become evident in the sequel, the simplicity of the updating of u_{ij}'s stems from the fact that the problem is decomposed with respect to u_{ij}'s (due to the nature of PCM).

Proof. It is clear that if $u_{ij} \in (1, +\infty)$, all terms in eq. (4.2) are strictly positive and, as a consequence, $f(u_{ij})$ is positive. Moreover, $u_{ij} \in (-\infty, 0)$ is meaningless, since in this case $\ln u_{ij}$ is not defined. □

Proposition 2. *The stationary points of* $f(u_{ij})$ *are* $\hat{u}_{ij} = \left[\frac{\lambda}{\gamma_j} p(1-p)\right]^{\frac{1}{1-p}}$ *and* $\tilde{u}_{ij} = +\infty$ [4].

Proposition 3. *The unique minimum of* $f(u_{ij})$ *appears at* $\hat{u}_{ij} = \left[\frac{\lambda}{\gamma_j} p(1-p)\right]^{\frac{1}{1-p}}$.

Proposition 4. *If* $f(\hat{u}_{ij}) < 0$ *then* $f(u_{ij}) = 0$ *has exactly two solutions* $u_{ij}^{\{1\}}$, $u_{ij}^{\{2\}} \in (0, 1)$ *with* $u_{ij}^{\{1\}} < u_{ij}^{\{2\}}$.

Proposition 5. *If* $f(u_{ij}) = 0$ *has two solutions* $u_{ij}^{\{1\}}$, $u_{ij}^{\{2\}}$ *(with* $u_{ij}^{\{1\}} < u_{ij}^{\{2\}}$*),* $J_{SPCM}(\Theta, U)$ *exhibits a local minimum at the largest of them (*$u_{ij}^{\{2\}}$*).*

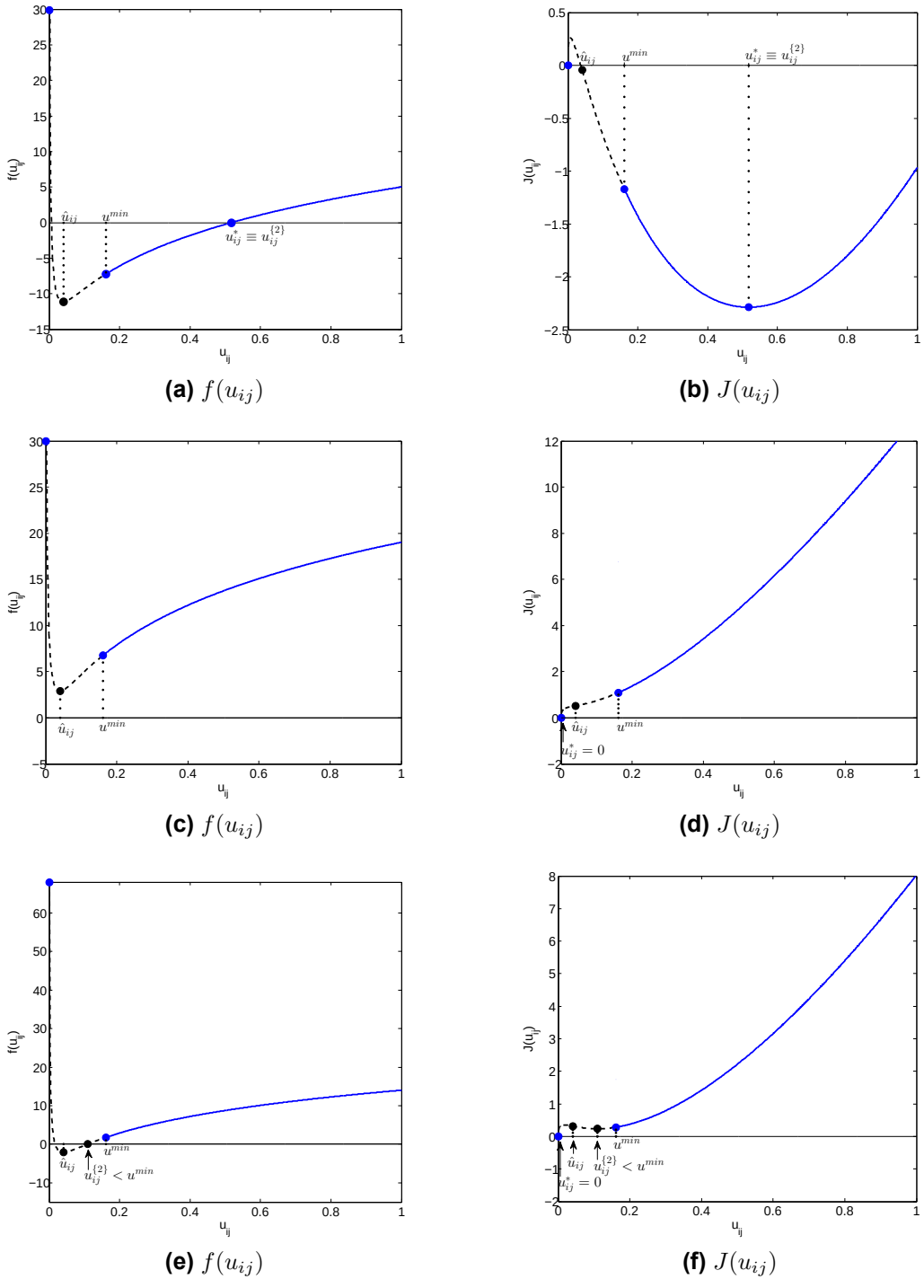
Proposition 6. $J_{SPCM}(\Theta, U)$ *exhibits its global minimum (with respect to* u_{ij}*) at* u_{ij}^**, where:*

$$u_{ij}^* = \begin{cases} u_{ij}^{\{2\}}, & \text{if } f(\hat{u}_{ij}) < 0 \text{ and } u_{ij}^{\{2\}} > \left(\frac{\lambda(1-p)}{\gamma_j}\right)^{\frac{1}{1-p}} \ (\equiv u_{min}) \\ 0, & \text{otherwise} \end{cases} \tag{4.3}$$

Based on the above propositions, to determine u_{ij}^*, we solve $f(u_{ij}) = 0$ as follows. First, we determine $\hat{u}_{ij}$ and check whether $f(\hat{u}_{ij}) > 0$. If this is the case, then $f(u_{ij})$ has no roots in $[0, 1]$. Note that, in this case, it is $f(u_{ij}) > 0$ for all $u_{ij} \in (0, 1]$, since $f(\hat{u}_{ij}) > 0$ (see Fig. 4.3c). Thus, J_{SPCM} is increasing with respect to u_{ij} in $(0, 1]$ (see Fig. 4.3d). Consequently, in this case we set $u_{ij}^* = 0$, *imposing sparsity*. In the rare case, where $f(\hat{u}_{ij}) = 0$, we set $u_{ij}^* = 0$, as $\hat{u}_{ij}$ is the unique root of $f(u_{ij}) = 0$ and $f(u_{ij}) > 0$ for $u_{ij} \in (0, \hat{u}_{ij}) \cup (\hat{u}_{ij}, 1]$. If $f(\hat{u}_{ij}) < 0$, then $f(u_{ij}) = 0$ has exactly two solutions that both lie in $[0, 1]$ (see Figs. 4.3a, 4.3e). In order to determine the largest of the solutions ($u_{ij}^{\{2\}}$), we apply the bisection method (see e.g. [77]) in the range $(\hat{u}_{ij}, 1]$, as $u_{ij}^{\{2\}}$ is greater than $\hat{u}_{ij}$. The bisection method is known to converge very rapidly to the optimum u_{ij}, that is, in our case, to the largest of the two solutions of $f(u_{ij}) = 0$ [5]. If the obtained solution $u_{ij}^{\{2\}}$ satisfies the rightmost condition in the first branch of eq. (4.3), then we set $u_{ij}^* = u_{ij}^{\{2\}}$ (see Fig. 4.3b). Otherwise, u_{ij}^* is set to 0 (see Fig. 4.3f).

[4] The proofs of Propositions 2 to 6 are given in Appendix B.

[5] Alternatively, any other method of this kind can also be used, e.g. [78].



(a) $f(u_{ij})$ **(b)** $J(u_{ij})$

(c) $f(u_{ij})$ **(d)** $J(u_{ij})$

(e) $f(u_{ij})$ **(f)** $J(u_{ij})$

Figure 4.3: In all plots the dashed parts of the graphs correspond to the interval $(0, u_{min})$, which is not accessible by the algorithm (see eq. (4.3)). (a) The shape of function $f(u_{ij})$, when $f(\hat{u}_{ij}) < 0$ and the right-most condition of eq. (4.3) is satisfied and (b) the corresponding shape of the cost function $J(u_{ij})$. (c) The shape of function $f(u_{ij})$, when $f(\hat{u}_{ij}) > 0$ and (d) the corresponding shape of $J(u_{ij})$. (e) The shape of function $f(u_{ij})$, when $f(\hat{u}_{ij}) < 0$ and the right-most condition of eq. (4.3) is not satisfied and (f) the corresponding shape of $J(u_{ij})$.

4.5 Selection of the parameter λ

As it follows from the previous analysis, considering a specific data point $\mathbf{x}_i$ and a cluster C_j, a necessary condition in order for the equation $f(u_{ij}) = 0$ to have a solution is $f(\hat{u}_{ij}) < 0$, which, taking into account eq. (4.2) and solving with respect to λ gives $\lambda < \frac{\gamma_j}{p(1-p)} \exp\left(-1 - \frac{d_{ij}(1-p)}{\gamma_j}\right)$. Consequently, selecting

$$\lambda \geq \frac{\gamma_j}{p(1-p)} \exp\left(-1 - \frac{d_{ij}(1-p)}{\gamma_j}\right), \tag{4.4}$$

the degree of compatibility u_{ij} of a data point $\mathbf{x}_i$ with a cluster C_j is set to 0, promoting sparsity. Aiming at retaining the smallest sized cluster, say C_q (i.e., the cluster with $\gamma_q = \min_{j=1,\ldots,m} \gamma_j$) until the termination of the algorithm (provided of course that at least one representative has been initially placed in it), a reasonable choice for λ would be the one for which u_{ij} becomes 0 for points $\mathbf{x}_i$ that lie at distance d_{iq} greater than γ_q from the representative $\boldsymbol{\theta}_q$. In this way, $\boldsymbol{\theta}_q$ will be less likely to be "attracted" by nearby larger clusters, aiding it to remain in the region of the physical cluster where it was first placed. This is so because the cluster representative will be affected only by the data points that are very close to it (i.e., points with $d_{iq} < \gamma_q = \min_{j=1,\ldots,m} \gamma_j$).

To this end, applying inequality (4.4) for d_{ij} and γ_j equal to $\gamma_q = \min_{j=1,\ldots,m} \gamma_j$, we end up with $\lambda \geq \frac{\gamma_q}{p(1-p)\mathrm{e}^{2-p}}$, where e is the base of natural logarithm. In practice, we select λ as

$$\lambda = K \frac{\min_{j=1,\ldots,m} \gamma_j}{p(1-p)\mathrm{e}^{2-p}}, \tag{4.5}$$

where K is set to values around 1, i.e., actually we allow non-zero u_{ij}'s for points that lie at distance around γ_q from $\boldsymbol{\theta}_q$. In most of the experiments of SPCM, we take $K = 0.9$ (see Fig. 4.4).

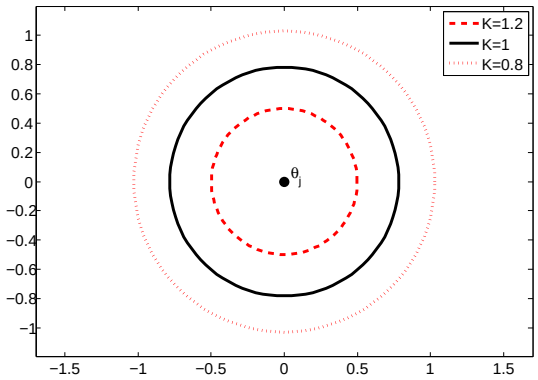


Figure 4.4: A representative θ_j is denoted with a black dot. The circles delimit the regions around θ_j that contain points with $u_{ij} > 0$, for (a) $K > 1$, (b) $K = 1$ and (c) $K < 1$.

4.6 The SPCM algorithm

From the analysis provided in the previous sections, the SPCM algorithm can be summarized as follows.

Algorithm 5 [Θ, U] = SPCM(X, m_{ini})

Input: X, m_{ini}

1: $t = 0$

2: $m = m_{ini}$

▷ *Initialization of $\boldsymbol{\theta}_j$'s part*

3: [$\Theta(t)$, $U^{FCM}(t)$] = FCM(X, m_{ini}, 2) [6]

▷ *Initialization of γ_j's part*

4: **Set:** $\gamma_j = \frac{\sum_{i=1}^{n} u_{ij}^{FCM} \|\mathbf{x}_i - \boldsymbol{\theta}_j(t)\|^2}{\sum_{i=1}^{n} u_{ij}^{FCM}}$, $j = 1, ..., m$

5: **Set:** $\lambda = K \frac{\min_{j=1,...,m} \gamma_j}{p(1-p)\mathrm{e}^{2-p}}$

6: **repeat**

▷ *Update U part*

7: **Update:** $U(t)$ (as described in section 4.4)

▷ *Update Θ part*

8: $\boldsymbol{\theta}_j(t+1) = \sum_{i=1}^{N} u_{ij}(t)\mathbf{x}_i \Big/ \sum_{i=1}^{N} u_{ij}(t)$, $j = 1, ..., m$

9: $t = t + 1$

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

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

In the sequel, we discuss how the exploitation of sparsity affects the clustering result in Examples 1 and 2, by comparing PCM and SPCM through the use of some quantitative indices. Specifically, in order to compare a clustering outcome with the true data label information, we use (a) the *Rand Measure* (RM), (b) the *Success Rate* (SR) and (c) the mean of the Euclidean distances (MD), which are described in section 3.7.2.

Example 1 (cont.): Table 4.1 shows the clustering results of PCM and SPCM, where m_{ini} and m_{final} denote the initial and the final number of distinct clusters. Figs. 4.1b and 4.5a depict the performances of PCM and SPCM, respectively.

[6] See Algorithm 2 of Chapter 2. We set the fuzzifier $q = 2$.

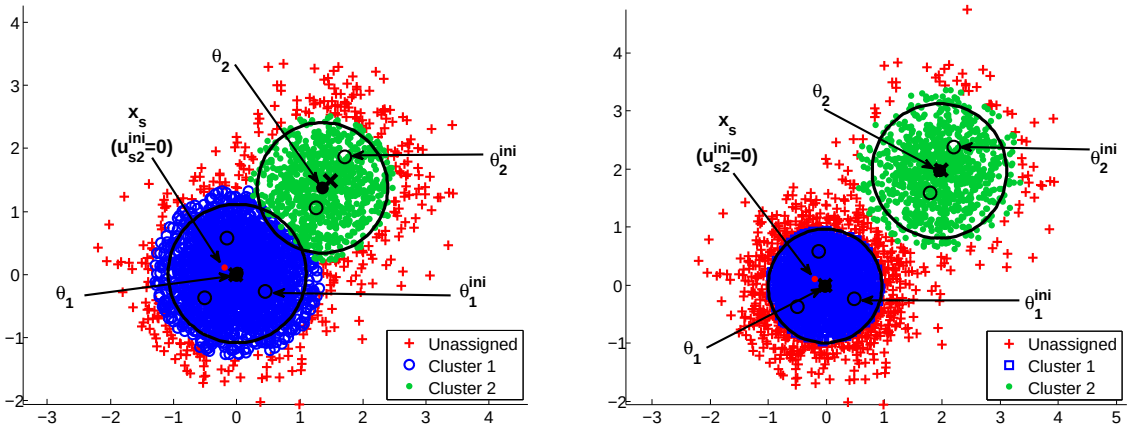


Figure 4.5: The clustering results of SPCM for the data set of (a) Example 1 with $m_{ini} = 5$ and (b) Example 2 with $m_{ini} = 5$. In both cases the contribution of the typical point $\mathbf{x}_s$ to the determination of $\boldsymbol{\theta}_2^{ini}$ becomes zero. See also the caption of Fig. 4.1.

As we have already seen, PCM fails to uncover the underlying clustering structure (as is clearly depicted quantitatively in Table 4.1), whereas SPCM distinguishes the two physical clusters, since it annihilates the contributions of most of the points of C_1 (C_2) in the determination of the next location of $\boldsymbol{\theta}_2$ ($\boldsymbol{\theta}_1$) through the imposition of sparsity. Note also that the fact that C_1 is denser than C_2 did not affect the computation of $\boldsymbol{\theta}_2$, since u_{i2} becomes 0 for most of the points of C_1. This is also verified through the achieved RM, SR and MD values (see Fig. 4.5a and Table 4.1).

Table 4.1: Performance of PCM and SPCM for the data sets of Examples 1 and 2.

	Data Set	m_{ini}	m_{final}	RM	SR	MD
PCM	Example 1	5	1	55.54	66.67	1.0271
SPCM	Example 1	5	2	91.22	95.40	0.0822
PCM	Example 2	5	2	95.35	97.60	0.1042
SPCM	Example 2	5	2	96.21	98.07	0.0194

Example 2 (cont.): Table 4.1 shows the clustering results of PCM and SPCM and Fig. 4.5b depicts the performance of SPCM. As we have seen in this case, PCM is able to uncover the underlying clustering structure. However, SPCM manages to detect more accurately the true centers of the clusters, as the MD index clearly indicates.

Remark 1: Note that for $p = 1$ the last term in eq. (4.2) becomes constant and u_{ij} can be expressed in closed form as $u_{ij} = \exp\left(-\frac{d_{ij}+\lambda}{\gamma_j}\right)$, i.e., it is a scaled version of eq. (2.13) of the classical PCM (see pink curve in Fig. 4.6).

Remark 2: In Fig. 4.6, the degree of compatibility u_{ij} versus d_{ij}/γ_j, resulting from SPCM, is plotted for several values of $p \in (0, 1)$. It can be seen that in each curve corresponding to $p < 1$, there is a critical point where a discontinuity is observed; that is u_{ij} "jumps" from a positive value to zero. The existence of such a point indicates that "hard" sparsity is

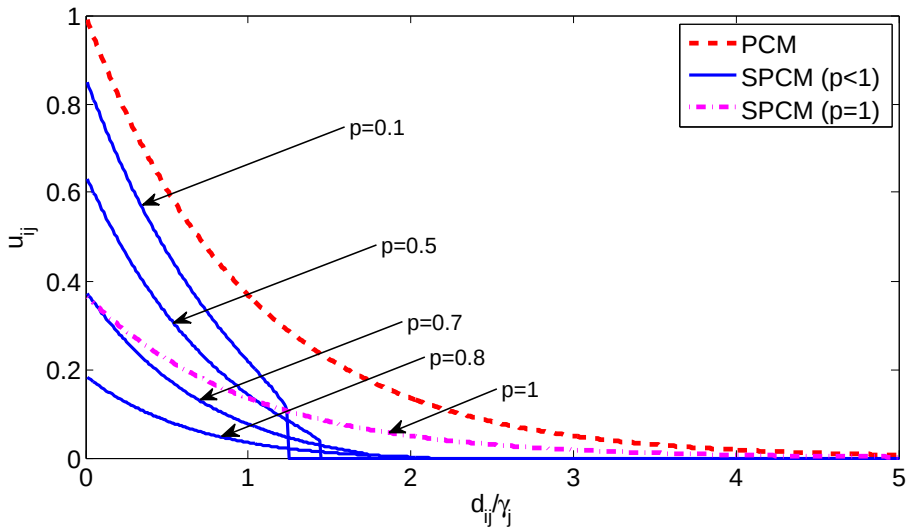
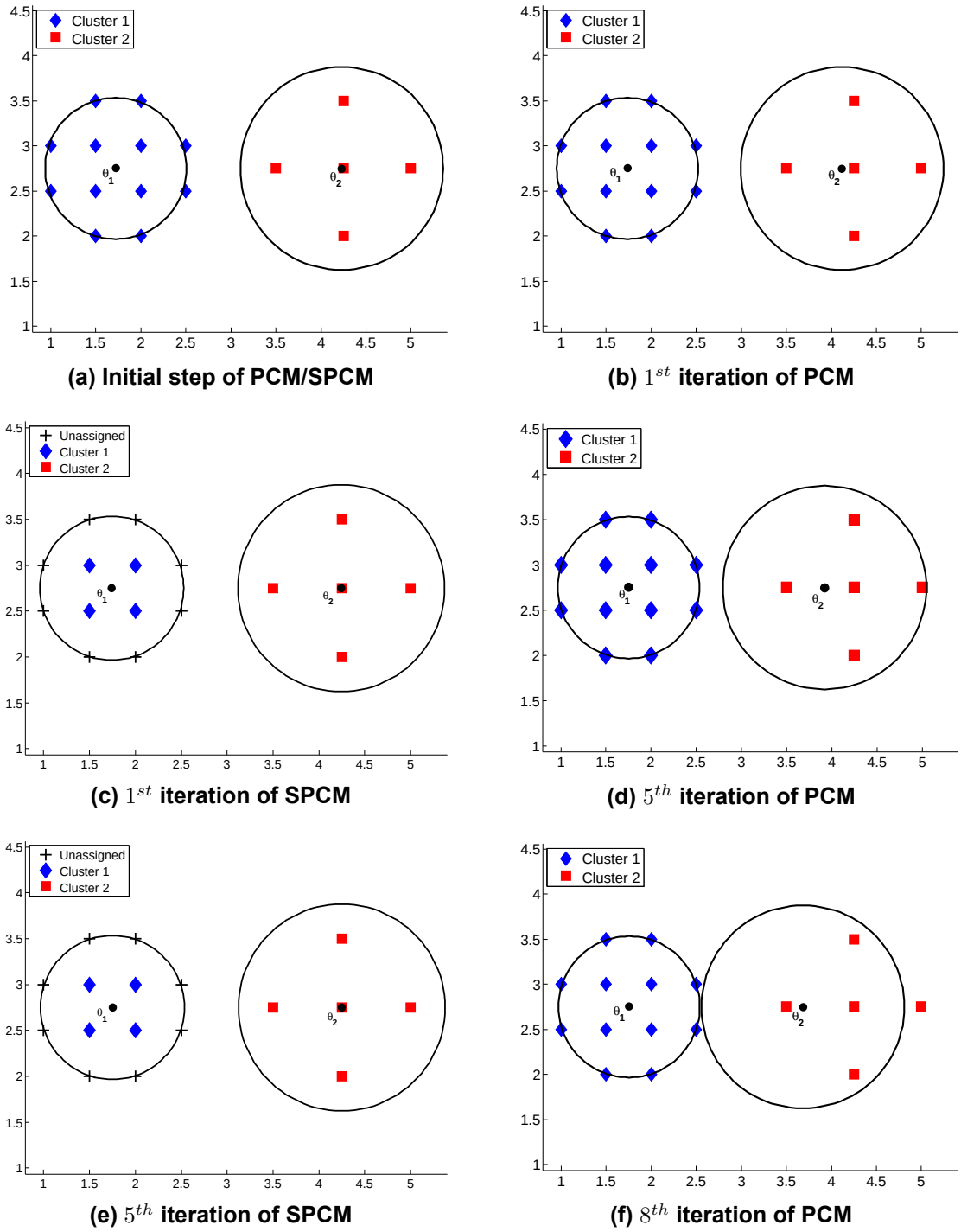


Figure 4.6: The degree of compatibility u_{ij} as a function of the d_{ij}/γ_j for PCM [10] (red curve), SPCM for several values of $p < 1$ (blue curves) and SPCM for $p = 1$ (pink curve).

imposed on $\mathbf{u}_i$'s, being the result of the inclusion of the third term in the cost function of J_{SPCM}, which, in turn, leads to the numerical computation of u_{ij}'s. "Hard" sparsity means that we do not have to define a small threshold below which u_{ij} is set to zero, but sparsity is forced automatically. Note also that as p increases towards 1, the "jump" becomes smaller and is moved to the right in the graph. The "jump" ceases to exist in the curves of PCM and SPCM with $p = 1$, i.e. no hard sparsity is imposed in these cases. Finally, from this diagram, it can also be noted that u_{ij}'s take generally lower values in SPCM, compared to PCM, which, in addition to the induced sparsity, contributes to the ability of SPCM in distinguishing closely located clusters.

4.7 Collation of SPCM with PCM

This experiment illustrates the rationale of SPCM, which has been approached in Example 1 more qualitatively. Let us consider a two-dimensional data set consisting of $N = 17$ points, which form two clusters C_1 and C_2 with 12 and 5 data points, respectively (see Fig. 4.7). The means of the clusters are $\mathbf{c}_1 = [1.75, 2.75]$ and $\mathbf{c}_2 = [4.25, 2.75]$. In this experiment, we consider only the PCM and the SPCM algorithms, both with $m = 2$. Fig. 4.7a shows the initial positions of the cluster representatives that are taken from FCM and the circles with radius equal to $\sqrt{\gamma_j}$'s resulting from eq. (2.16) (for $B = 1$) for both PCM and SPCM. Similarly, Figs. 4.7c and 4.7b show the new locations of $\boldsymbol{\theta}_j$'s after the first iteration of the algorithms and Figs. 4.7e, 4.7d show the locations of $\boldsymbol{\theta}_j$'s after the 5th and 5th (final) iterations for PCM and SPCM, respectively. Finally, Fig. 4.7f shows the locations of $\boldsymbol{\theta}_j$'s after the 8th iteration for PCM. Table 4.2 shows the degrees of compatibility u_{ij}'s of all data points $\mathbf{x}_i$'s with the cluster representatives $\boldsymbol{\theta}_j$'s at the three iterations considered in Fig. 4.7 for both PCM and SPCM.



(a) Initial step of PCM/SPCM

(b) 1^{st} iteration of PCM

(c) 1^{st} iteration of SPCM

(d) 5^{th} iteration of PCM

(e) 5^{th} iteration of SPCM

(f) 8^{th} iteration of PCM

Figure 4.7: (a) PCM and SPCM snapshots at their initialization step, (b), (c) their first iteration, (d), (f) 5th and 8th iteration for PCM and (e) 5th (final) iteration for SPCM.

Table 4.2: The degrees of compatibility of the data points of Experiment 1 for PCM and SPCM algorithms, after: (a) first iteration (for both algorithms), (b) 5th iteration for PCM and 5th (final) iteration for SPCM and (c) 8th iteration for PCM.

	1^{st} iteration				5^{th} iteration		5^{th} (final) iteration		8^{th} iteration	
	PCM		SPCM		PCM		SPCM		PCM	
$\mathbf{x}_i$	C_1	C_2	C_1	C_2	C_1	C_2	C_1	C_2	C_1	C_2
$(1.5, 3.5)$	0.3701	0.0018	0	0	0.3616	0.0064	0	0	0.3606	0.0118
$(2.0, 3.5)$	0.3526	0.0127	0	0	0.3619	0.0352	0	0	0.3630	0.0570
$(1.0, 3.0)$	0.3884	2.5e-04	0	0	0.3613	0.0012	0	0	0.3583	0.0024
$(1.5, 3.0)$	0.8348	0.0027	0.4625	0	0.8157	0.0095	0.4478	0	0.8134	0.0174
$(2.0, 3.0)$	0.7954	0.0188	0.4316	0	0.8164	0.0523	0.4476	0	0.8186	0.0846
$(2.5, 3.0)$	0.3360	0.0897	0	0	0.3623	0.1949	0	0	0.3653	0.2766
$(1.0, 2.5)$	0.3884	2.5e-04	0	0	0.3613	0.0012	0	0	0.3583	0.0024
$(1.5, 2.5)$	0.8348	0.0027	0.4625	0	0.8157	0.0095	0.4478	0	0.8134	0.0174
$(2.0, 2.5)$	0.7954	0.0188	0.4316	0	0.8164	0.0523	0.4476	0	0.8186	0.0846
$(2.5, 2.5)$	0.3360	0.0897	0	0	0.3623	0.1949	0	0	0.3653	0.2766
$(1.5, 2.0)$	0.3701	0.0018	0	0	0.3616	0.0064	0	0	0.3606	0.0118
$(2.0, 2.0)$	0.3526	0.0127	0	0	0.3619	0.0352	0	0	0.3630	0.0570
$(4.25, 3.5)$	1.2e-05	0.6415	0	0.4850	1.5e-05	0.5883	0	0.4852	1.6e-05	0.5276
$(3.5, 2.75)$	0.0058	0.6566	0	0.4983	0.0069	0.8712	0	0.4854	0.0070	0.9512
$(4.25, 2.75)$	3.0e-05	0.9997	0	0.8046	3.9e-05	0.9168	0	0.8049	4.0e-05	0.8222
$(5.0, 2.75)$	2.5e-08	0.6267	0	0.4720	3.5e-08	0.3972	0	0.4849	3.6e-08	0.2926
$(4.25, 2.0)$	1.2e-05	0.6415	0	0.4850	1.5e-05	0.5883	0	0.4852	1.6e-05	0.5276

As it can be deduced from Table 4.2 and Fig. 4.7, the degrees of compatibility of the data points of C_1 with the cluster representative $\boldsymbol{\theta}_2$ increase as PCM evolves, leading gradually $\boldsymbol{\theta}_2$ towards the region of the cluster C_1 and thus, ending up with two coincident clusters, although $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ are initialized properly through the FCM algorithm (see Fig. 4.7a). However, this is not the case in SPCM algorithm, as both the cluster representatives remain in the centers of the actual clusters. It is of great interest to mention that in SPCM $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ conclude closest to the actual centers compared to their initial state through the FCM algorithm (see Fig. 4.7e). Obviously, the superior performance of SPCM is due to the sparsity imposed on $\mathbf{u}_i$'s leading several u_{ij}'s to 0 for points $\mathbf{x}_i$ that lie "away" from $\boldsymbol{\theta}_j$ (see Table 4.2), thus preventing these points from contributing to the estimation of $\boldsymbol{\theta}_j$. This experiment indicates the ability of SPCM to handle successfully cases where relatively closely located clusters with different densities are involved.

More experiments of SPCM on both synthetic and real data sets are presented in the "Experimental Results" section of the next chapter.

4.8 On the convergence of the SPCM

In the sequel, a proof of the convergence of the SPCM is provided. Note that, in principle, the proof holds for any choice of (fixed) γ_j's, not only for the one given in eq. (2.16).

A vital observation is that, as long as u_{ij} is given by the first branch of eq. (4.3), it is bounded as follows

$$u^{min} \leq u_{ij} \leq u^{max}, \tag{4.6}$$

where u^{max} is obtained by solving the equation $f(u_{ij}) = 0$, for $d_{ij} = 0$; that is the equation

$\gamma_j \ln u_{ij} + \lambda p u_{ij}^{p-1} = 0$. Note that both u^{min} and u^{max} depend exclusively on λ, γ_j and p.

Before we proceed, we will give an alternative expression for eq. (4.3), which will be extensively exploited in the convergence proof below. More specifically, we will express the condition of the first branch of (4.3) in terms of $\boldsymbol{\theta}_j$. To this end, we consider the case where $u_{ij}^{\{2\}} = u^{min}$. This implies that $f(u_{ij}^{\{2\}}) = 0$ or $f(u^{min}) = 0$. Substituting u^{min} by its equal given in eq. (4.3) and after some straightforward algebraic manipulations, it follows that $f(u_{ij}^{min}) = 0$ is equivalent to

$$||\mathbf{x}_i - \boldsymbol{\theta}_j||^2 = \overbrace{\frac{\gamma_j}{1-p}\left(-\ln\frac{\lambda(1-p)}{\gamma_j} - p\right)}^{R_j^2}. \tag{4.7}$$

The above is the equation of a hypersphere, denoted by $\mathcal{C}_{ij}$, centered at $\mathbf{x}_i$ and having radius R_j (note that R_j depends exclusively on the parameters γ_j, p, λ and not on the data points $\mathbf{x}_i$ or on $\boldsymbol{\theta}_j$'s and u_{ij}'s). Clearly, its interior $int(\mathcal{C}_{ij})$ (which in the subsequent analysis is assumed to contain $\mathcal{C}_{ij}$ itself) contains all the positions of $\boldsymbol{\theta}_j$ which give $u_{ij} > 0$, while all the points in its exterior $ext(\mathcal{C}_{ij})$ correspond to positions of $\boldsymbol{\theta}_j$ that give $u_{ij} = 0$. In order to ensure that $\mathcal{C}_{ij}$ is properly defined, we should ensure that R_j is positive. This holds true if K is chosen so that $K < pe^{2(1-p)}$ (see Proposition C1 in Appendix C). In the light of the above result, eq. (4.3) can be rewritten as follows

$$u_{ij}^* = \begin{cases} u_{ij}^{\{2\}}, & \text{if } ||\mathbf{x}_i - \boldsymbol{\theta}_j||^2 \leq R_j^2 \\ 0, & \text{otherwise} \end{cases} \tag{4.8}$$

Note that the expressions for u_{ij}^* given by eqs. (4.3) and (4.8) are equivalent and will be used interchangeably in the subsequent analysis.

Before we proceed, we note that the cost function associated with SPCM (eq. (4.1)) can be recasted as

$$J_{SPCM}(U,\Theta) \;=\; \sum_{j=1}^{m} J_j(\mathbf{u}_j, \boldsymbol{\theta}_j) \;\equiv\; \sum_{j=1}^{m}\left[\sum_{i=1}^{N} \overbrace{u_{ij}\|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2 + \gamma_j \sum_{i=1}^{N}(u_{ij}\ln u_{ij} - u_{ij}) + \lambda u_{ij}^p}^{h(u_{ij},\boldsymbol{\theta}_j)}\right], \tag{4.9}$$

where $\mathbf{u}_j = [u_{1j}, \ldots, u_{Nj}]^T$. Since (a) u_{ij}'s, $j = 1, \ldots, m$, are not interrelated to each other, for a specific $\mathbf{x}_i$, (b) u_{ij}'s, $i = 1, \ldots, N$ are related exclusively with $\boldsymbol{\theta}_j$ and vice versa and (c) $\boldsymbol{\theta}_j$'s are not interrelated to each other, minimization of $J_{SPCM}(U,\Theta)$ can be considered as the minimization of m independent cost functions J_j's, $j = 1, \ldots, m$. Thus, in the sequel, we focus on the minimization of a specific $J_j(\mathbf{u}_j, \boldsymbol{\theta}_j)$ and, for the ease of notation, we drop the index j, i.e., when we write $J(\mathbf{u}, \boldsymbol{\theta})$, we refer to a $J_j(\mathbf{u}_j, \boldsymbol{\theta}_j)$. In addition, we write $\mathbf{u} = [u_1, \ldots, u_N]^T$ and $\boldsymbol{\theta}$ for the vectors $\mathbf{u}_j$, $\boldsymbol{\theta}_j$ that correspond to the j-th cluster.

The proof is given under the very mild assumption that for each cluster at least one equa-

tion $f(u_i) = 0$, $i = 1, \ldots, N$ has two solutions at each iteration of SPCM (*Assumption 1*). This is a rational assumption, since if this does not hold at a certain iteration, the algorithm cannot identify new locations for $\boldsymbol{\theta}$ at the next iteration. In subsection 4.8.1, it is shown how this assumption can always be fulfilled.

Some definitions are now in order. Let $\mathcal{M}$ be the set containing all the $N \times 1$ vectors $\mathbf{u}$ whose elements lie in the union $\{0\} \cup [u^{min},\ u^{\max}]$, i.e. $\mathcal{M} = (\{0\} \cup [u^{min}, u^{max}])^N$. Also, let $\mathcal{R}^l$ be the space where the vector $\boldsymbol{\theta}$ lives. The SPCM algorithm produces a sequence $(\mathbf{u}^{(t)}, \boldsymbol{\theta}^{(t)})|_{t=0}^{\infty}$, which will be examined in terms of its convergence properties.

Let

$$G : \mathcal{M} \rightarrow \mathcal{R}^l, \text{ with } G(\mathbf{u}) = \boldsymbol{\theta},$$

where G is calculated via the following equation

$$\boldsymbol{\theta} = \frac{\sum_{i=1}^{N} u_i \mathbf{x}_i}{\sum_{i=1}^{N} u_i} \tag{4.10}$$

and

$$F : \mathcal{R}^l \rightarrow \mathcal{M}, \text{ with } F(\boldsymbol{\theta}) = \mathbf{u},$$

where F is calculated via eq. (4.8). Then, the SPCM operator $T : \mathcal{M} \times \mathcal{R}^l \rightarrow \mathcal{M} \times \mathcal{R}^l$ is defined as

$$T = T_2 \circ T_1, \tag{4.11}$$

where

$$T_1 : \mathcal{M} \times \mathcal{R}^l \rightarrow \mathcal{M},\ \ T_1(\mathbf{u}, \boldsymbol{\theta}) = F(\boldsymbol{\theta}) \tag{4.12}$$

and

$$T_2 : \mathcal{M} \rightarrow \mathcal{M} \times \mathcal{R}^l,\ \ T_2(\mathbf{u}) = (\mathbf{u}, G(\mathbf{u})). \tag{4.13}$$

For operator T we have that

$$T(\mathbf{u}, \boldsymbol{\theta}) = (T_2 \circ T_1)(\mathbf{u}, \boldsymbol{\theta}) = T_2(T_1(\mathbf{u}, \boldsymbol{\theta})) = T_2(F(\boldsymbol{\theta})) =$$

$$(F(\boldsymbol{\theta}), G(F(\boldsymbol{\theta}))) = (F(\boldsymbol{\theta}), (G \circ F)(\boldsymbol{\theta})).$$

Thus, the iteration of SPCM can be expressed in terms of T as

$$(\mathbf{u}^{(t)}, \boldsymbol{\theta}^{(t)}) = T(\mathbf{u}^{(t-1)}, \boldsymbol{\theta}^{(t-1)}) = (F(\boldsymbol{\theta}^{(t-1)}), (G \circ F)(\boldsymbol{\theta}^{(t-1)})).$$

The above decomposition of T to T_1 and T_2 will facilitate the subsequent convergence analysis, since certain properties for T can be proved relying on T_1 and T_2 (and, ultimately, on F and G).

Remark 3: Note that F (and as a consequence T_1) are, in general, not continuous (actually they are piecewise continuous).

In the sequel some required definitions are given. Let $Z : X \rightarrow X$ ($X \subset \mathcal{R}^r$) be a point-to-point map that gives rise to an iterative algorithm $z(t) = Z(z(t-1))$, which generates

a sequence $z(t)|_{t=0}^{\infty}$, for a given $z(0)$. A *fixed point* z^* of Z is a point for which $Z(z^*) = z^*$. Also, we say that Z is *strictly monotonic* with respect to a (continuous) function g if $g(Z(z)) < g(z)$, whenever z is not a fixed point of Z. Having said the above, we can now state the following theorem that will be proved useful in the sequel:

Theorem 1 [79] [7] *:* Let $Z : X \to X$ ($X \in \mathcal{R}^r$) be a point-to-point map that gives rise to an iterative algorithm $z(t) = Z(z(t-1))$, which generates a sequence $z(t)|_{t=0}^{\infty}$, for a given $z(0)$. Supposing that:

(i) Z is strictly monotonic with respect to a continuous function $g : X \to \mathcal{R}$,

(ii) Z is continuous on X,

(iii) the set of all points $z(t)|_{t=0}^{\infty}$ is bounded and

(iv) the number of fixed points having any given value of g is finite

then

the algorithm corresponding to Z will converge to a fixed point of Z regardless where it is initialized in X [8].

In the SPCM case, Z is the mapping T (SPCM operator) defined by eq. (4.11) and g is the cost function J. Due to the fact that SPCM has been resulted from the minimization of J, it turns out that its fixed points $(\mathbf{u}^*, \boldsymbol{\theta}^*)$ satisfy $\nabla J|_{(\mathbf{u},\boldsymbol{\theta})} = \mathbf{0}$.

Although the general strategy to prove convergence for an algorithm is to show that it fulfills the requirements of the convergence theorem, this cannot be adopted in this straightforward manner in this framework. The reason is that Theorem 1 requires continuity of T, which is not guaranteed in the SPCM case due to T_2 (F) (see eq. (4.8)), which is not continuous in its domain (which is the convex hull of X, $CH(X)$)[9]. However, it is continuous on certain subsets of $CH(X)$. This fact will allow the use of Theorem 1 for certain small regions where continuity is preserved.

Some additional definitions are now in order. Without loss of generality, let $I = (\cap_{i=1}^{k} int(\mathcal{C}_i))$ [10]; that is I is the (nonempty) intersection of the interiors of the hyperspheres of radius R (eq. (4.7)) that correspond to $\mathbf{x}_i$'s, $i = 1, \ldots, k$ (see Fig. 4.8)[11,12]. Note that for $\boldsymbol{\theta} \in I$ the above k points will have $u_i > 0$. The set of all data points that have $u_i > 0$ with the previous $\boldsymbol{\theta}$ form the so-called *active set*, while the points themselves are called *active points*. In

[7] This is a direct combination of Theorem 3.1 and Corollary 3.2 in [79].

[8] Actually, this theorem has been stated for the more general case where Z is a one-to-many mapping [79]. The present form of the theorem is for the special case where Z is a one-to-one mapping, which is the case for SPCM.

[9] Due to its updating (eq. (2.15)), $\boldsymbol{\theta}$ will always lie in $CH(X)$, provided that its initial position lies in at least one hypersphere of radius R centered at a data point.

[10] The notation of I should not be confused with the identity matrix.

[11] Clearly, by reordering the data points we can take all the possible corresponding I intersections.

[12] The accurate notation here would be instead of I, to use $I_{\{\mathbf{x}_{i_1}, \mathbf{x}_{i_2}, \ldots, \mathbf{x}_{i_k}\}}$, in order to explicitly denote the dependence on the associated data points (i_j's should be different to each other and $i_1, ..., i_k \in \{1, ..., N\}$). However, we choose not to do the latter for keeping the notation as light as possible.

addition, an active set X_q is called *valid* if its corresponding intersection of hyperspheres I_q is nonempty. Finally, the points with $u_i = 0$ are called *inactive*.

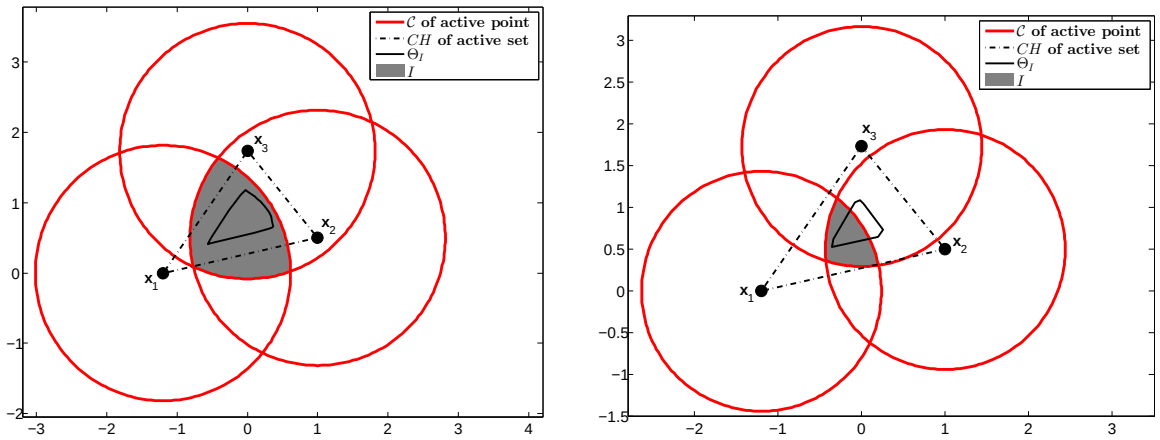


Figure 4.8: An active set of $k = 3$ points in cases when (a) $\Theta_I \subset I$ and (b) $\Theta_I \not\subset I$

Let also

$$U_I = \{\mathbf{u} = [u_1, \ldots, u_k] : \mathbf{u} = F(\boldsymbol{\theta}), \text{ for } \boldsymbol{\theta} \in I\} \tag{4.14}$$

be the set containing all vectors $\mathbf{u}$, whose components u_i span the range of all possible values of the degrees of compatibility of $\boldsymbol{\theta}$ with the k active $\mathbf{x}_i$'s. Clearly, u_i's are computed via the first branch of eq. (4.8) and F is continuous in this specific case (as it will be explicitly shown later). Also, let

$$\Theta_I = \{\boldsymbol{\theta} : \boldsymbol{\theta} = G(\mathbf{u}), \text{ for } \mathbf{u} \in U_I\} \tag{4.15}$$

(see Fig. 4.8 for the possible scenarios for Θ_I). Three observations are now in order:

- First, due to the fact that u_i's are independent from each other, U_I can also be expressed as

$$U_I = \Pi_{i=1}^{k}[u^{min}, u_i^{max}], \tag{4.16}$$

  where Π denotes the Cartesian product and u_i^{max} is the maximum possible value u_i can take, provided that $\boldsymbol{\theta} \in I$ (clearly $u_i^{max} \leq u^{max}$).

- If at a certain iteration t of SPCM, $\boldsymbol{\theta}(t) \in I$, Θ_I contains all possible positions of $\boldsymbol{\theta}(t+1)$.

- Θ_I always lies in the convex hull of the associated active set.

In the sequel, we proceed by showing the following facts, that are preliminary for the establishment of the final convergence result. Specifically, we will show that

- (A) $J(\mathbf{u}, \boldsymbol{\theta})$ decreases at each iteration of the SPCM operator T

- (B) T is continuous on every region $U_I \times I$ that corresponds to a valid active set.
- (C) The sequence produced by the algorithm is bounded
- (D) The fixed points corresponding to a certain valid active set (if they exist) are strict local minima of J and they are finite.

• Proof of item (A):

To achieve this goal, we prove first the following two lemmas

Lemma 1: Let $\phi : \mathcal{M} \to \mathcal{R}$, $\phi(\mathbf{u}) = J(\mathbf{u}, \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is fixed. Then $\mathbf{u}^*$ is the global minimum solution of ϕ if and only if $\mathbf{u}^* = F(\boldsymbol{\theta})$, where F is defined as in eq. (4.3).

Proof: We proceed by showing that

(a) the unique point $\mathbf{u}^*$ that satisfies the KKT conditions for the minimization problem

$$\begin{array}{ll} \min \phi(\mathbf{u}) & \\ \text{subject to} & u_i \geq 0, \qquad i = 1, \ldots, N \\ \text{and} & 1 - u_i \geq 0, \; i = 1, \ldots, N \end{array} \tag{4.17}$$

is the one determined by eq. (4.3) and

(b) this point is a minimizer of J, which implies (due to the uniqueness) that it is the global minimizer.

Let $\mathbf{u}^* = [u_i^*]$ be a point that satisfies the KKT conditions for (4.17). Then we have

$$(i)\; u_i^* \geq 0, \quad (ii)\; 1 - u_i^* \geq 0 \tag{4.18}$$

$$(i)\; \exists\; \kappa_i \geq 0 :\; \kappa_i u_i^* = 0, \quad (ii)\; \exists\; \tau_i \geq 0 :\; \tau_i (1 - u_i^*) = 0 \tag{4.19}$$

and

$$\frac{\partial \mathcal{L}(\mathbf{u})}{\partial u_i}|_{\mathbf{u}=\mathbf{u}^*} = 0, \tag{4.20}$$

where $\mathcal{L}(\mathbf{u})$ is the Lagrangian function defined as

$$\mathcal{L}(\mathbf{u}) = \phi(\mathbf{u}) - \sum_{i=1}^{N} \kappa_i u_i - \sum_{i=1}^{N} \tau_i (1 - u_i). \tag{4.21}$$

Recalling eq. (4.1), $\phi(\mathbf{u})$ can be written as

$$\phi(\mathbf{u}) = \sum_{i=1}^{N} \overbrace{[u_i ||\mathbf{x}_i - \boldsymbol{\theta}||^2 + \gamma(u_i \ln u_i - u_i) + \lambda u_i^p]}^{h(u_i;\boldsymbol{\theta})}, \tag{4.22}$$

where $h(u_i; \boldsymbol{\theta})$ is a function of u_i for a fixed value of $\boldsymbol{\theta}$. Noting that all u_i's are computed

independently from each other, for fixed $\boldsymbol{\theta}$, it is easy to verify that, for a specific u_i it is

$$\frac{\partial \phi(\mathbf{u})}{\partial u_i} = \frac{\partial h(u_i;\boldsymbol{\theta})}{\partial u_i} = ||\mathbf{x}_i - \boldsymbol{\theta}||^2 + \gamma \ln u_i + \lambda p u_i^{p-1} \equiv f(u_i).$$

As a consequence, eq. (4.20) gives

$$||\mathbf{x}_i - \boldsymbol{\theta}||^2 + \gamma \ln u_i^* + \lambda p u_i^{*p-1} - \kappa_i + \tau_i = 0. \tag{4.23}$$

We will prove next that $\kappa_i = 0$ and $\tau_i = 0$, for $i = 1, \ldots, N$; that is, the constraints on u_i's are inactive, i.e., the optimum of $\phi(\mathbf{u})$ lies always in the region defined by the constraints. Assume, on the contrary, that there exists $\kappa_s > 0$. From eq. (4.19-(i)) it follows that $u_s^* = 0$ and from eq. (4.19-(ii)) that $\tau_s = 0$. Taking into account that $\lim_{u_s^* \to 0^+} (\gamma \ln u_s^* + \lambda p {u_s^*}^{p-1}) = +\infty$ [13] and applying eq. (4.23) for u_s^* we have

$$||\mathbf{x}_s - \boldsymbol{\theta}||^2 + \infty = \kappa_s \text{ or } \kappa_s = +\infty \tag{4.24}$$

which contradicts the fact that κ_s is finite.

Assume next that there exists $\tau_s > 0$. From eq. (4.19-(ii)) it follows that $u_s^* = 1$ and from eq. (4.19-(i)), it is $\kappa_s = 0$. Applying eq. (4.23) for u_s^* and substituting the above we have

$$||\mathbf{x}_s - \boldsymbol{\theta}||^2 + \gamma \ln 1 + \lambda p 1^{p-1} + \tau_s = 0 \text{ or } \tau_s = -||\mathbf{x}_s - \boldsymbol{\theta}||^2 - \lambda p < 0, \tag{4.25}$$

which contradicts the fact that $\tau_s > 0$. Thus $\tau_s = 0$.

Since $\kappa_i = \tau_i = 0$, for all i, eq. (4.23) becomes

$$||\mathbf{x}_i - \boldsymbol{\theta}||^2 + \gamma \ln u_i^* + \lambda p {u_i^*}^{p-1} \equiv f(u_i^*) = 0, \quad i = 1, \ldots, N. \tag{4.26}$$

Note that the SPCM algorithm relies on eq. (4.26) in order to derive the updating formula of eq. (4.3) (thus step (a) has been shown). We proceed now to show that the point corresponding to eq. (4.3) (derived through eq. (4.26)) minimizes J. We consider the following two cases:

• u_i^* is given by the first branch of eq. ((4.3)). This implies that $f(u_i) = 0$ has two solutions $u_i^{\{1\}}$ and $u_i^{\{2\}}$ ($u_i^{\{1\}} < u_i^{\{2\}}$) and $u_i^{\{2\}} > \left(\frac{\lambda(1-p)}{\gamma_j}\right)^{\frac{1}{1-p}} (= u^{min})$ (figures 1a, 1d). Taking into account the definition of $h(u_i;\boldsymbol{\theta})$ in eq. (4.22) and Proposition 5, it follows that the maximum of the two solutions $u_i^{\{1\}}$, $u_i^{\{2\}}$ ($u_i^{\{1\}} < u_i^{\{2\}}$) is the one that minimizes $h(u_i;\boldsymbol{\theta})$ and, as a consequence, $\phi(\mathbf{u})$ also (which equals to $J(\mathbf{u},\boldsymbol{\theta})$) with $\boldsymbol{\theta}$ fixed.

• u_i^* is given by the second branch of eq. (4.3). In this case we have that either (i) $f(u_i)$ is strictly positive, which implies that $J(\mathbf{u},\boldsymbol{\theta})$ is strictly increasing with respect to u_i (case shown in figures 1b, 1e) or (ii) $h(u_i^{\{2\}},\boldsymbol{\theta}) \geq h(0,\boldsymbol{\theta}) = 0$ (case shown in figures 1c, 1f). In both (i) and (ii) cases, $J(\mathbf{u},\boldsymbol{\theta})$ is minimized with respect to u_i only for $u_i = 0$ (the second

[13] Utilization of the L' Hospital rule gives that $\lim_{x\to 0^+} x^{1-p} \ln x = 0$ ($p < 1$). Then $\lim_{x\to 0^+}(\ln x + \beta \frac{1}{x^{1-p}}) = \lim_{x\to 0^+} \frac{x^{1-p}\ln x + \beta}{x^{1-p}} = +\infty$, for $\beta > 0$. Setting $x = u_s^*$, $\beta = \frac{\lambda p}{\gamma}$, the claim follows.

branch of eq. (4.3)).

From the above, it follows that $\mathbf{u}^*$ is the global minimum solution of ϕ if and only if $\mathbf{u}^*$ is given by eq. (4.3). Q.E.D.

Lemma 2: Let $\psi : \mathcal{R}^l \to \mathcal{R}$, with $\psi(\boldsymbol{\theta}) = J(\mathbf{u}, \boldsymbol{\theta})$, where $\mathbf{u} \in U_I$ is fixed. Then, $\boldsymbol{\theta}^*$ ($\in \Theta_I$) is the unique global minimum of ψ if and only if $\boldsymbol{\theta}^* = G(\mathbf{u})$, where G is calculated as in eq. (4.10).

Proof: In contrast to the situation in Lemma 1, the minimization of $\psi(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ is an unconstrained optimization problem. The stationary points of $\psi(\boldsymbol{\theta})$ are obtained as the solutions of the equations

$$\frac{\partial \psi}{\partial \boldsymbol{\theta}} \;=\; \frac{\partial}{\partial \boldsymbol{\theta}} \left[\sum_{i=1}^{N} \Big(u_i ||\mathbf{x}_i - \boldsymbol{\theta}||^2 + \gamma (u_i \ln u_i - u_i) + \lambda u_i^p \Big) \right] \;=\; 2 \sum_{i=1}^{N} u_i (\boldsymbol{\theta} - \mathbf{x}_i) \;=\; \mathbf{0}, \quad (4.27)$$

which, after some manipulations, give

$$\boldsymbol{\theta}^* = \frac{\sum_{i=1}^{N} u_i \mathbf{x}_i}{\sum_{i=1}^{N} u_i}. \quad (4.28)$$

Also, it is

$$H_\psi \equiv \frac{\partial^2 \psi}{\partial \boldsymbol{\theta}^2} = \overbrace{2 \sum_{i=1}^{N} u_i}^{b} \, I^l, \quad (4.29)$$

where I^l is the $l \times l$ identity matrix. Under *Assumption 1*, stating that at least one u_i is computed by the first branch of eq. (4.3), it is $b > 0$. Therefore, ψ is a convex function over $\mathcal{R}^l$, with a unique stationary point, given by eq. (4.28), which is the unique global minimum of $\psi(\boldsymbol{\theta})$. Q.E.D.

Combining now the previous two lemmas, we are in a position to prove the following lemma.

Lemma 3: Consider a valid active set, whose corresponding hyperspheres intersection is denoted by I. Let

$$S = \{(\mathbf{u}, \boldsymbol{\theta}) = ([u_1, \ldots, u_k], \boldsymbol{\theta}) \in U_I \times I : \nabla J|_{(\mathbf{u}, \boldsymbol{\theta})} = \mathbf{0}$$
$$\text{with } u_i \text{ being the largest of the two solutions of } f_{\boldsymbol{\theta}}(u_i) = 0, i = 1, \ldots, k\}\ ^{14}. \quad (4.30)$$

Then J is continuous over $U_I \times I$ and

$$J(T(\mathbf{u}, \boldsymbol{\theta})) < J(\mathbf{u}, \boldsymbol{\theta}), \text{ if } (\mathbf{u}, \boldsymbol{\theta}) \notin S.$$

[14] In the sequel, we insert $\boldsymbol{\theta}$ as subscript in the notation of f in order to show explicitly the dependence of u_i from $\boldsymbol{\theta}$.

Proof: Since $\{y \to ||y||^2\}$, $\{y \to \ln y\}$, $\{y \to y^p\}$ are continuous and J is a sum of products of such functions, it follows that J is continuous on $U_I \times I$. Let $(\mathbf{u}, \boldsymbol{\theta}) \notin S$. Recalling that

$$T(\mathbf{u}, \boldsymbol{\theta}) = (F(\boldsymbol{\theta}), (G \circ F)(\boldsymbol{\theta})) = (F(\boldsymbol{\theta}), G(F(\boldsymbol{\theta}))),$$

we have

$$J(T(\mathbf{u}, \boldsymbol{\theta})) = J(F(\boldsymbol{\theta}), G(F(\boldsymbol{\theta}))). \tag{4.31}$$

Applying Lemma 1 for fixed $\boldsymbol{\theta}$, we have that $F(\boldsymbol{\theta})$ is the unique global minimizer of J. Thus,

$$J(F(\boldsymbol{\theta}), \boldsymbol{\theta}) < J(\mathbf{u}, \boldsymbol{\theta}). \tag{4.32}$$

Applying Lemma 2 for fixed $F(\boldsymbol{\theta})$, we have that $G(F(\boldsymbol{\theta}))$ is the unique global minimizer of J. Thus, it is

$$J(F(\boldsymbol{\theta}), G(F(\boldsymbol{\theta}))) < J(F(\boldsymbol{\theta}), \boldsymbol{\theta}). \tag{4.33}$$

From eqs. (4.31), (4.32) and (4.33), it follows that

$$J(T(\mathbf{u}, \boldsymbol{\theta})) < J(\mathbf{u}, \boldsymbol{\theta}), \quad \text{for } (\mathbf{u}, \boldsymbol{\theta}) \notin S.$$

Q.E.D.

Remark 4: It is noted that although the above proof has been focused on the k (active) points, its generalization that takes also into account the rest data points is straightforward since $u_i = 0$, for $i = k+1, \ldots, N$ and the corresponding terms $h(u_i, \boldsymbol{\theta})$ that contribute to J are 0.

Remark 5: Taking into account that SPCM has been resulted from the minimization of J ($\nabla J|_{(\mathbf{u},\boldsymbol{\theta})} = \mathbf{0}$) on a $U_I \times I$ corresponding to an active set, it follows that S contains all the fixed points of T, which (as will be shown later) are local minima of the cost function J (of course, J may have additional local minima than those belong to S which are not accessible by the algorithm).

Now we proceed by showing that T decreases J, in the whole domain $(\{0\} \cup [u^{min}, u^{max}])^N \times CH(X)$.

Lemma 4: The strict monotonically decreasing property of T with respect to J remains valid in the domain $(\{0\} \cup [u^{min}, u^{max}])^N \times CH(X)$ excluding the fixed points of T of each valid active set.

Proof: Let $(\bar{\mathbf{u}}, \bar{\boldsymbol{\theta}})$ be the outcome of SPCM at a specific iteration, $\hat{\mathbf{u}} = F(\bar{\boldsymbol{\theta}})$ be the $\mathbf{u}$ for the next iteration and $\hat{\boldsymbol{\theta}} = G(\hat{\mathbf{u}})$ be the subsequent $\boldsymbol{\theta}$. Recall that the ordering of the updating is

$$\bar{\mathbf{u}} \to \bar{\boldsymbol{\theta}} \to \hat{\mathbf{u}} \to \hat{\boldsymbol{\theta}}. \tag{4.34}$$

We define

$$\bar{\Gamma} = \{i \ : \ \bar{u}_i \text{ is computed via the second branch of eq. } (4.3)\}$$

and

$$\hat{\Gamma} = \{i \ : \ \hat{u}_i \text{ is computed via the second branch of eq. } (4.3)\}.$$

Recalling that $h(u_i;\boldsymbol{\theta}) = u_i||\mathbf{x}_i - \boldsymbol{\theta}||^2 + \gamma(u_i \ln u_i - u_i) + \lambda u_i^p$, we can write

$$J(\bar{\mathbf{u}},\bar{\boldsymbol{\theta}}) = \overbrace{\sum_{i\in\bar{\Gamma}\cap\hat{\Gamma}} h(\bar{u}_i;\bar{\boldsymbol{\theta}})}^{\bar{A}_1} + \overbrace{\sum_{i\in\tilde{}\,\bar{\Gamma}\cap\hat{\Gamma}} h(\bar{u}_i;\bar{\boldsymbol{\theta}})}^{\bar{A}_2} + \overbrace{\sum_{i\in\tilde{}\,\hat{\Gamma}} h(\bar{u}_i;\bar{\boldsymbol{\theta}})}^{\bar{A}_3} \tag{4.35}$$

and

$$J(\hat{\mathbf{u}},\bar{\boldsymbol{\theta}}) = \overbrace{\sum_{i\in\bar{\Gamma}\cap\hat{\Gamma}} h(\hat{u}_i;\bar{\boldsymbol{\theta}})}^{\hat{A}_1} + \overbrace{\sum_{i\in\tilde{}\,\bar{\Gamma}\cap\hat{\Gamma}} h(\hat{u}_i;\bar{\boldsymbol{\theta}})}^{\hat{A}_2} + \overbrace{\sum_{i\in\tilde{}\,\hat{\Gamma}} h(\hat{u}_i;\bar{\boldsymbol{\theta}})}^{\hat{A}_3}, \tag{4.36}$$

where $\tilde{}\,\Gamma$ denotes the complement of Γ.

Focusing on $\bar{A}_1$ and $\hat{A}_1$, we have that $h(\bar{u}_i;\bar{\boldsymbol{\theta}}) = h(\hat{u}_i;\bar{\boldsymbol{\theta}}) = 0$, since $i \in \bar{\Gamma}\cap\hat{\Gamma}$. Thus

$$\hat{A}_1 = \bar{A}_1 = 0. \tag{4.37}$$

Considering $\bar{A}_2$ and $\hat{A}_2$, since $i \in \hat{\Gamma}$, we have $\hat{u}_i = 0$. Thus, taking into account the order of updating (eq. (4.34)) and Lemma 1, we have $(0 =)\ h(\hat{u}_i;\bar{\boldsymbol{\theta}}) < h(\bar{u}_i;\bar{\boldsymbol{\theta}})$. Thus, it follows that

$$\hat{A}_2 < \bar{A}_2. \tag{4.38}$$

Finally, focusing on $\bar{A}_3$ and $\hat{A}_3$, since $i \in \tilde{}\,\hat{\Gamma}$, the argumentation of Lemma 1 implies that the global minimum of $h(u_i;\bar{\boldsymbol{\theta}})$ is met at $\hat{u}_i = u_i^{\{2\}}$. Thus, taking also into account the order of updating in eq. (4.34), it is $h(\hat{u}_i;\bar{\boldsymbol{\theta}}) < h(\bar{u}_i;\bar{\boldsymbol{\theta}})$. Therefore, it is

$$\hat{A}_3 < \bar{A}_3. \tag{4.39}$$

Combining eqs. (4.37), (4.38) and (4.39) it follows that

$$J(\hat{\mathbf{u}},\bar{\boldsymbol{\theta}}) < J(\bar{\mathbf{u}},\bar{\boldsymbol{\theta}}). \tag{4.40}$$

Also, lemma 2 gives

$$J(\hat{\mathbf{u}},\hat{\boldsymbol{\theta}}) < J(\hat{\mathbf{u}},\bar{\boldsymbol{\theta}}) \ \ [15]. \tag{4.41}$$

Combining eqs. (4.40), (4.41), we have that

$$J(\hat{\mathbf{u}},\hat{\boldsymbol{\theta}}) < J(\bar{\mathbf{u}},\bar{\boldsymbol{\theta}}).$$

Q.E.D.

[15] Considering a valid active set with corresponding hypersphere intersection $\bar{I}$ and $\Theta_{\bar{I}}$ defined as in eqs. (4.14), (4.15), it is noted that although $\bar{\boldsymbol{\theta}} \in \bar{I}$, this does not necessarily hold for $\hat{\boldsymbol{\theta}}$, as Fig. 4.8b indicates, since $\hat{\boldsymbol{\theta}} \in \Theta_{\bar{I}}$, with $\Theta_{\bar{I}} \not\subset \bar{I}$.

• **Proof of item (B):**

In the sequel, we give two useful Propositions concerning the continuity of the F and G mappings. In both Propositions, without loss of generality, we consider a valid active set, having $\mathbf{x}_i, i = 1, \ldots, k$ as active points, whose corresponding hypersphere intersection is denoted by I and U_I, Θ_I are defined via eqs. (4.14), (4.15).

Proposition 7: The mapping G is continuous on $U_I \times \{0\}^{N-k}$.

Proof: To prove that G is continuous in the N variables u_i, note that G is a vector field with the resolution by (l) scalar fields, written as

$$G = (G_1, \ldots, G_l) : U_I \times \{0\}^{N-k} \to \mathcal{R}^l,$$

where $G_q : U_I \times \{0\}^{N-k} \to \mathcal{R}$ is defined as:

$$G_q(\mathbf{u}) = \frac{\sum_{i=1}^{N} u_i \mathbf{x}_i}{\sum_{i=1}^{N} u_i} \equiv \theta_q, \quad q = 1, \ldots, l. \tag{4.42}$$

Since $\{u_i \to u_i \mathbf{x}_i\}$ is a continuous function and the sum of continuous functions is also continuous, G_q is also continuous as the quotient of two continuous functions. Under the assumption that $\sum_{i=1}^{N} u_i > 0$, the denominator in eq. (4.42) never vanishes. Thus, G_q is well-defined in all cases and it is also continuous. Therefore, G is continuous in its entire domain. Q.E.D.

Proposition 8: The mapping F is continuous over I.

Proof: It suffices to show that F is continuous on the l variables θ_q. F is a vector field with the resolution by (N) scalar fields, i.e.,

$$F = (F_1, \ldots, F_N) : I \to U_I,$$

where F_q is given by eq. (4.8).

The mapping $\{\boldsymbol{\theta} \to ||\mathbf{x}_i - \boldsymbol{\theta}||^2 (\equiv d_i)\}$ is continuous. Let us focus on the u_i's, $i = 1, \ldots, k$, for which $int(\mathcal{C}_i)$ contributes to the formation of I; that is, on u_i's given by the first branch of (4.8). The mapping $\{d_i \to u_i\}$ is continuous. To see this, note that (since γ is constant), the graph of $f(u_i)$ (which is continuous), viewed as a function of d_i, is simply shifted upwards or downwards as d_i varies (see fig. 4.9). Focusing on the rightmost point, $u_i^{\{2\}}$, where the graph intersects the horizontal axis, it is clear that small variations of d_i cause small variations to $u_i^{\{2\}}$, which implies the continuity of $\{d_i \to u_i\}$ in this case.

Let us focus next on the u_i's, $i = k+1, \ldots, N$, for which $int(\mathcal{C}_i)$ do not contribute to the formation of I; in this case u_i is given by the second branch of (4.8) and the claim follows trivially. Q.E.D.

As a direct consequence of Propositions 7 and 8, we have the following lemma.

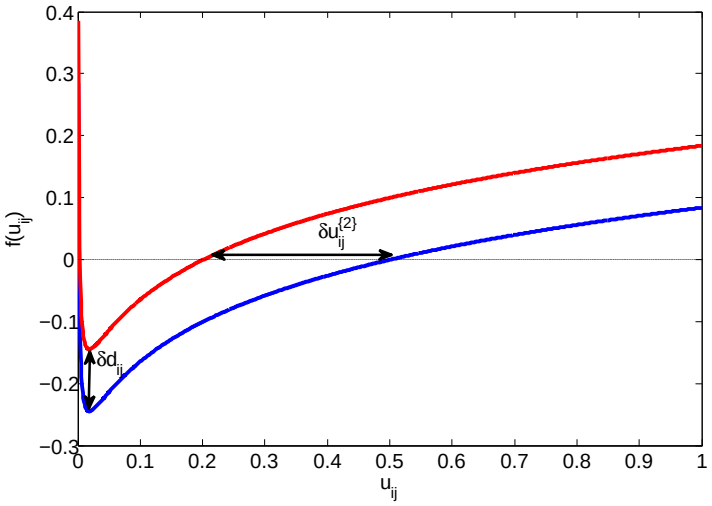


Figure 4.9: Graphical presentation of the continuity of the mapping $\{d_{ij} \rightarrow u_{ij}\}$. Small variations in d_{ij} cause small variations in u_{ij}.

Lemma 5: T is continuous on $U_I \times I$.

Proof: Recall that $T = T_2 \circ T_1$ and T_2 and T_1 are defined in terms of G and F, respectively (eqs. (4.12), (4.13)). G is continuous on U_I, as a consequence of Proposition 7, while F is continuous on I from Proposition 8. Thus, T is continuous on $U_I \times I$ as composition of two continuous functions. Q.E.D.

• **Proof of item (C):**

We proceed now to prove that the sequence $(\mathbf{u}^{(t)}, \boldsymbol{\theta}^{(t)})|_{t=0}^{\infty}$ produced by the SPCM falls in a bounded set.

Lemma 6: Let $(F(\boldsymbol{\theta}^{(0)}), \boldsymbol{\theta}^{(0)})$ be the starting point of the iteration with the SPCM operator T, with $\boldsymbol{\theta}^{(0)} \in CH(X)$ and $\mathbf{u}^{(0)} = F(\boldsymbol{\theta}^{(0)})$. Then

$$(\mathbf{u}^{(t)}, \boldsymbol{\theta}^{(t)}) \equiv T^t(\mathbf{u}^{(0)}, \boldsymbol{\theta}^{(0)}) \in [0,\ 1]^N \times CH(X).$$

Proof: For a given $\boldsymbol{\theta}^{(0)} \in CH(X)$, $\mathbf{u}^{(0)} = F(\boldsymbol{\theta}^{(0)}) \in [0,\ 1]^N$, since $u_i^{(0)} \in [0,\ 1]$ (see eq. (4.3)). Also, $\boldsymbol{\theta}^{(1)} = G(\mathbf{u}^{(0)})$ is computed by eq. (4.10), which can be recasted as

$$\boldsymbol{\theta}^{(1)} = \sum_{i=1}^{N} \frac{u_i^{(0)}}{\sum_{i=1}^{N} u_i^{(0)}} \mathbf{x}_i.$$

Since $u_i^{(0)} \in [0, 1]$, it easily follows that $0 \leq \frac{u_i^{(0)}}{\sum_{i=1}^{N} u_i^{(0)}} \leq 1$ and $\sum_{i=1}^{N} \frac{u_i^{(0)}}{\sum_{i=1}^{N} u_i^{(0)}} = 1$. Thus $\boldsymbol{\theta}^{(1)} \in CH(X)$. Continuing recursively we have $\mathbf{u}^{(1)} = F(\boldsymbol{\theta}^{(1)}) \in [0,\ 1]^N$ by eq. (4.3) and $\boldsymbol{\theta}^{(2)} = G(\mathbf{u}^{(1)}) \in CH(X)$, using the same argumentation as above. Thus, inductively, we

conclude that

$$(\mathbf{u}^{(t)}, \boldsymbol{\theta}^{(t)}) \equiv T^t(\mathbf{u}^{(0)}, \boldsymbol{\theta}^{(0)}) \in [0,\ 1]^N \times CH(X).$$

Q.E.D.

Remark 6: Note that it is possible to have $\boldsymbol{\theta}^{(0)}$ outside $CH(X)$, yet in a position where at least one u_i is positive. Computing $\mathbf{u}^{(0)} = F(\boldsymbol{\theta}^{(0)})$ by eq. (4.3), the latter will lie in $\mathcal{M}$ and, as a consequence, $\boldsymbol{\theta}^{(1)} = G(\mathbf{u}^{(0)})$ will lie in $CH(X)$ as it follows by the argumentation given in the proof of Lemma 5.

• **Proof of item (D):**

In the sequel, we will prove that the elements of the set S (eq. 4.30), for a given valid active set with hyperspheres intersection I (if they exist) are strict local minima of the cost function J and thus the cardinality of S is finite.

The elements of S are the solutions $\mathbf{z}^* = (\mathbf{u}^*, \boldsymbol{\theta}^*) \equiv (u_1^*, \ldots, u_k^*, \theta_1^*, \ldots, \theta_l^*)$ [16] of $\nabla J|_{(\mathbf{u},\boldsymbol{\theta})} = \mathbf{0}$ with u_i^* being the largest of the two solutions of $f_{\boldsymbol{\theta}}(u_i) = 0$, $i = 1, \ldots, k$. They should satisfy the following equations

$$2\sum_{i=1}^{k} u_i^*(\theta_q^* - x_{iq}) = 0, \quad q = 1, \ldots, l \tag{4.43}$$

and

$$||\mathbf{x}_i - \boldsymbol{\theta}^*||^2 + \gamma \ln u_i^* + \lambda p {u_i^*}^{p-1} = 0, \quad i = 1, \ldots, k. \tag{4.44}$$

Then, we have the following lemma.

Lemma 7: The points $\mathbf{z}^*$ that satisfy eqs. (4.43) and (4.44) (if they exist) are strict local minima of J in the domain $U_I \times I$. Moreover, their number is finite.

Proof: In order to prove that $\mathbf{z}^*$ are local minima we need to prove that the Hessian matrix of J computed at $\mathbf{z}^*$, $H_{\mathbf{z}^*}$, is positive definite over a small region around $\mathbf{z}^*$. It is

$$H_{\mathbf{z}^*} = \begin{bmatrix} g_1^* & 0 & & 0 & 2(\theta_1^* - x_{11}) & 2(\theta_2^* - x_{12}) & & 2(\theta_l^* - x_{1l}) \\ 0 & g_2^* & & 0 & 2(\theta_1^* - x_{21}) & 2(\theta_2^* - x_{22}) & & 2(\theta_l^* - x_{2l}) \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & & g_k^* & 2(\theta_1^* - x_{k1}) & 2(\theta_2^* - x_{k2}) & & 2(\theta_l^* - x_{kl}) \\ 2(\theta_1^* - x_{11}) & 2(\theta_1^* - x_{21}) & & 2(\theta_1^* - x_{k1}) & 2\sum_{i=1}^k u_i^* & 0 & & 0 \\ 2(\theta_2^* - x_{12}) & 2(\theta_2^* - x_{22}) & & 2(\theta_2^* - x_{k2}) & 0 & 2\sum_{i=1}^k u_i^* & & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 2(\theta_l^* - x_{1l}) & 2(\theta_l^* - x_{2l}) & \ldots & 2(\theta_l^* - x_{kl}) & 0 & 0 & \ldots & 2\sum_{i=1}^k u_i^* \end{bmatrix} \tag{4.45}$$

where

$$g_i^* = \gamma {u_i^*}^{-1} - \lambda p(1-p) {u_i^*}^{p-2}, \ i = 1, \ldots, k. \tag{4.46}$$

[16] Without loss of generality, we assume that the $\mathbf{x}_i$'s, $i = 1, \ldots, k$ are the active points of the valid active set under study.

Let $\mathbf{z}' = (\mathbf{u}', \boldsymbol{\theta}') \equiv (u'_1, \ldots, u'_k, \theta'_1, \ldots, \theta'_\ell)$ be a point in $U_I \times I$ that is close to $\mathbf{z}^*$. More specifically, let $u'_1, \ldots, u'_k$ be close to $u^*_1, \ldots, u^*_k$, respectively, so that

$$||\boldsymbol{\theta}^* - \frac{\sum_{i=1}^k u'_i \mathbf{x}_i}{\sum_{i=1}^k u'_i}|| < \varepsilon. \tag{4.47}$$

After some straightforward algebraic operations it follows that

$$\mathbf{z}'^T H_{\mathbf{z}^*} \mathbf{z}' = 2||\boldsymbol{\theta}'||^2 \sum_{i=1}^k u^*_i + 4 \sum_{i=1}^k u'_i \boldsymbol{\theta}'^T (\boldsymbol{\theta}^* - \mathbf{x}_i) + \sum_{i=1}^k u'^2_i g^*_i. \tag{4.48}$$

It is easy to verify that $\sum_{i=1}^k u'_i \boldsymbol{\theta}'^T (\boldsymbol{\theta}^* - \mathbf{x}_i) = \sum_{i=1}^k u'_i \boldsymbol{\theta}'^T (\boldsymbol{\theta}^* - \frac{\sum_{i=1}^k u'_i \mathbf{x}_i}{\sum_{i=1}^k u'_i}) \geq -\sum_{i=1}^k u'_i ||\boldsymbol{\theta}'|| \varepsilon$.

Utilizing the fact that $u_i > u^{min} \equiv (\frac{\lambda(1-p)}{\gamma})^{1/(1-p)}$, $i = 1, \ldots, k$, for the second appearance of u^*_i in the right hand side of (4.46), it turns out that $g^*_i \geq \frac{(1-p)\gamma}{u^*_i}$.

Combining the last two inequalities with eq. (4.48), it follows that

$$\mathbf{z}'^T H_{\mathbf{z}^*} \mathbf{z}' \quad \geq \quad 2 \sum_{i=1}^k u^*_i ||\boldsymbol{\theta}'||^2 \; - \; 4 \sum_{i=1}^k u'_i ||\boldsymbol{\theta}'|| \varepsilon \; + \; (1 \; - \; p) \gamma \sum_{i=1}^k \frac{u'^2_i}{u^*_i} \quad \equiv \quad \phi(||\boldsymbol{\theta}'||). \tag{4.49}$$

Since $\sum_{i=1}^k u^*_i > 0$, the second degree polynomial $\phi(||\boldsymbol{\theta}'||)$ becomes positive if and only if its discriminant

$$\Delta = 8[2\varepsilon^2 (\sum_{i=1}^k u'_i)^2 - (1-p)\gamma \sum_{i=1}^k u^*_i \sum_{i=1}^k \frac{u'^2_i}{u^*_i}] \tag{4.50}$$

is negative. But, from Proposition C2 in Appendix C, it is

$$(\sum_{i=1}^k u'_i)^2 \leq \sum_{i=1}^k u^*_i \sum_{i=1}^k \frac{u'^2_i}{u^*_i}.$$

Also, choosing $\varepsilon < \frac{1}{2}\sqrt{\frac{(1-p)\gamma}{2}}$, we have that Δ is negative. As a consequence and due to the continuity of J in $U_I \times I$, ε defines a region $Y_{\mathbf{z}^*}$ around $\mathbf{z}^*$, for which $\mathbf{z}'^T H_{\mathbf{z}^*} \mathbf{z}' > 0$ for $\mathbf{z}' \in Y_{\mathbf{z}^*}$. Thus $\mathbf{z}^*$ is a strict local minimum.

In addition, since the domain $U_I \times I$ is bounded, it easily follows that the number of strict local minima is finite. Q.E.D.

Remark 7: It can be shown that in the specific case where (a) $\frac{\gamma}{\bar{\gamma}} < \frac{1}{p} e^{(1-p)^2/2}$ and (b) K in eq. (4.5) is chosen in the range $[\frac{\gamma}{\bar{\gamma}} p e^{2-\frac{(1+p)^2}{2}}, \; p e^{2(1-p)}]$, then the set S_q (eq. (4.30)) that corresponds to each valid active set X_q has one element at the most. The proof of this fact follows the line of proof of lemma 7, with the difference that ε in eqs. (4.47), (4.49) and (4.50) is replaced by R (since the maximum possible distance between two points in the (nonempty) intersection of hyperspheres of distance R, is equal to R). Then, the conditions (a) and (b) above follow from the requirement to have $2R^2 < (1-p)\gamma$, in order

to have negative discriminant Δ. Taking into account eq. (4.7) and utilizing eq. (4.5) in the previous requirement it follows that $K > \frac{\gamma}{\tilde{\gamma}} p e^{2-\frac{(1+p)^2}{2}}$. Also, since $K < p e^{2(1-p)}$ (Proposition C1), condition (a) results from the requirement to have $\frac{\gamma}{\tilde{\gamma}} p e^{2-\frac{(1+p)^2}{2}} < p e^{2(1-p)}$.

In the sequel we denote by $Y_{\mathbf{z}^*}$ a region around a point $\mathbf{z}^*$ in the set S_q corresponding to a valid active set X_q, where J is convex. $Y_{\mathbf{z}^*}$ will be called as a *valley* around $\mathbf{z}^*$ (such a region always exists, as shown in Proposition C3).

Having completed the proof of the prerequisites (A)-(D) and before we proceed any further, some remarks are in order.

Remark 8: Although J is well defined in $[0,1]^N \times \mathcal{R}^l$, there are several regions in the landscape of $J(\mathbf{u}, \boldsymbol{\theta})$ that are not accessible by the algorithm. For example, some positions $(\mathbf{u}, \boldsymbol{\theta})$ where $u_i < u^{min}$ and those where $\boldsymbol{\theta}$ is expressed through eq. (2.15) with coefficients u_i less that u^{min}, are not accessible by the algorithm.

Remark 9: It is highlighted again the fact that a certain set of active points X_q, with corresponding (nonempty) union of hyperspheres I_q and U_{I_q}, Θ_{I_q} as defined by eqs. (4.14) and (4.15), respectively, may have no local minima of J in $U_{I_q} \times I_q$ that are accessible by T. Equivalently, this means that the solution set S_q (see Lemma 3) corresponding to X_q is empty.

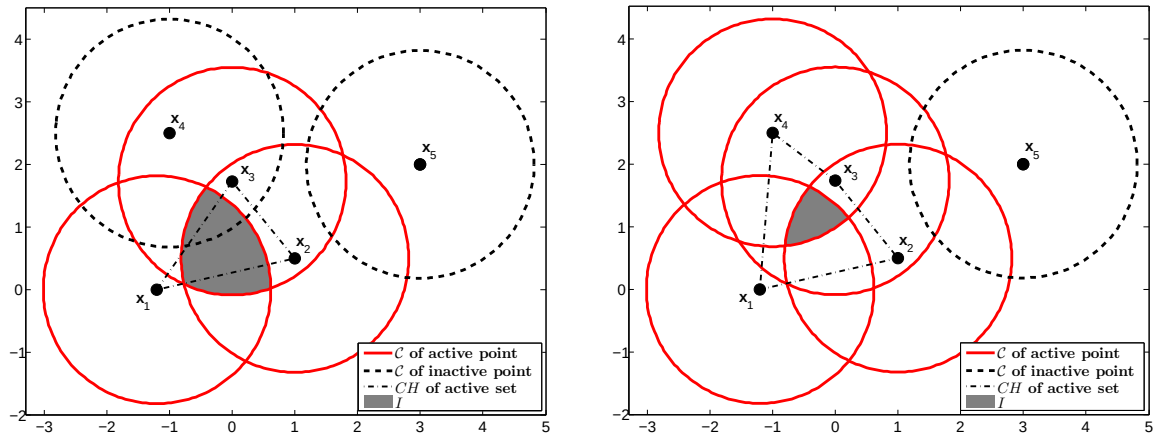


Figure 4.10: (a) An active set of $k = 3$ points where $(I \cap (\cap_{i:\ u_i=0} ext(\mathcal{C}_i))) \not\equiv I$ and (b) an active set of $k = 4$ points where $(I \cap (\cap_{i:\ u_i=0} ext(\mathcal{C}_i))) \equiv I$

We prove next the following lemma.

Lemma 8: There exists at least one valid active set X_q (with $I_q \neq \emptyset$) for which there exists at least one local minimum $(\mathbf{u}^*_{q_r}, \boldsymbol{\theta}^*_{q_r})$, with $\boldsymbol{\theta}^*_{q_r} \in I_q \cap (\cap_{i:\ u_i=0} ext(\mathcal{C}_i))$ [17].

Proof: Suppose on the contrary that for all possible active sets X_q, there is no local mini-

[17] Note that $\boldsymbol{\theta}^*_{q_r} \in \Theta_{I_q}$ due to the definition of the latter set from eq. (4.15).

mum $(\mathbf{u}^*_{q_r}, \boldsymbol{\theta}^*_{q_r})$ with $\boldsymbol{\theta}^*_{q_r} \in I_q \cap (\cap^N_{i:\ u_i=0} ext(\mathcal{C}_i))$ (see fig. 4.10). Equivalently, this means that the solution sets S_q for all valid active sets are empty. Then from lemma 3 we have that if at a certain iteration t_1, $\boldsymbol{\theta}(t_1)$ belongs to the intersection I_q of a certain active set X_q, the algorithm may move $\boldsymbol{\theta}(t)$ ($t > t_1$) to other positions in I_q that always strictly decrease the value of J. Since J is bounded below (due to the fact that $\mathbf{u} \in [0,\ 1]^N$ and $\boldsymbol{\theta} \in CH(X)$) it follows that $\boldsymbol{\theta}$ will leave I_q at a certain iteration. In addition, lemma 4 secures the decrease of the value of J as we move from one hypersphere intersection to another (or, equivalently, from one active set to another). Thus, the algorithm will always move $(\mathbf{u}(t), \boldsymbol{\theta}(t))$ from one position to another in the domain $[0,\ 1]^N \times CH(X)$, without converging to any one of them, while, at the same time the value of J decreases from iteration to iteration.

Assuming that at a specific iteration t', $\boldsymbol{\theta}(t')$ belongs to a certain I_q, then, due to the continuity of J in I_q, there exists a region $V(t')$ around $(\mathbf{u}(t'), \boldsymbol{\theta}(t'))$, for which $J(\mathbf{u}, \boldsymbol{\theta}) > J(\mathbf{u}(t'+1), \boldsymbol{\theta}(t'+1))$, for $(\mathbf{u}, \boldsymbol{\theta}) \in V(t')$.

From the previous argumentation, it follows that, since the domain where $(\mathbf{u}(t), \boldsymbol{\theta}(t))$ moves is bounded, the regions $V(t)$ (defined as above) will cover the regions of the whole domain that are accessible by T. Thus there exists an iteration t'' at which the algorithm will visit a point in the region $V(t')$, where t' corresponds to a position the algorithm visited before ($t' < t''$). Then, due to the strict decrease of J as SPCM evolves we have that $J(\mathbf{u}(t''), \boldsymbol{\theta}(t'')) < J(\mathbf{u}(t'+1), \boldsymbol{\theta}(t'+1)) < J(\mathbf{u}(t'), \boldsymbol{\theta}(t'))$. However, since $(\mathbf{u}(t''), \boldsymbol{\theta}(t'')) \in V(t')$, it follows that $J(\mathbf{u}(t''), \boldsymbol{\theta}(t'')) > J(\mathbf{u}(t'+1), \boldsymbol{\theta}(t'+1))$, which leads to a contradiction. Therefore, there exists at least one active set X_q for which there exists at least one local minimum $(\mathbf{u}^*_{q_r}, \boldsymbol{\theta}^*_{q_r})$, with $\boldsymbol{\theta}^*_{q_r} \in I_q \cap (\cap^N_{i:\ u_i=0} ext(\mathcal{C}_i))$. Q.E.D.

Now we are in the position to state the general theorem concerning the convergence of SPCM.

Theorem 2: Suppose that a data set $X = \{\mathbf{x}_i \in \mathcal{R}^l, i = 1, \ldots, N\}$ is given. Let $J(\mathbf{u}, \boldsymbol{\theta})$ be defined as in eq. (4.9) for $m = 1$, where $(\mathbf{u}, \boldsymbol{\theta}) \in \mathcal{M} \times CH(X)$. If $T : \mathcal{M} \times CH(X) \to \mathcal{M} \times CH(X)$ is the operator corresponding to SPCM algorithm, then for any $(\mathbf{u}(0), \boldsymbol{\theta}(0)) \in \mathcal{M} \times CH(X)$ the SPCM converges to one of the points of the set S_q that corresponds to a valid active set X_q, $\mathbf{z}^*_{q_r} = (\mathbf{u}^*_{q_r}, \boldsymbol{\theta}^*_{q_r})$, provided that $\boldsymbol{\theta}^*_{q_r} \in I_q \cap (\cap_{i:\ u_i=0} ext(\mathcal{C}_i))$.

Proof: Following a reasoning similar to that of lemma 8 we have that the regions of the whole space that are accessible by T will eventually be covered by regions $V(t')$ defined as in the proof of lemma 8. Then the algorithm

(i) either will visit a valley $Y_{\mathbf{z}^*_{q_r}}$ in $U_{I_q} \times I_q$ around a (strict) local minimum $(\mathbf{u}^*_{q_r}, \boldsymbol{\theta}^*_{q_r})$ of a certain active set X_q and, as a consequence of theorem 1 (due to (a) the local convexity of J in $Y_{\mathbf{z}^*_{q_r}}$, (b) the monotonic decrease of J with T, (c) the continuity of T in the corresponding $U_I \times I$ and (d) the uniqueness of the minimum in this valley) it will converge to it,

(ii) or it will never visit the valley of such a local minimum. This means that the algorithm starts from a $(\mathbf{u}(0), \boldsymbol{\theta}(0))$, whose $J(\mathbf{u}(0), \boldsymbol{\theta}(0))$ is less than the values of J at all local minima. However, this case can be rejected following exactly the same reasoning with that in the proof of lemma 8.

Therefore, the algorithm will converge to a local minimum $\mathbf{z}^*_{q_r}$ that corresponds to one of the possible active sets X_q (with $I_q \neq \emptyset$) provided that $\boldsymbol{\theta}^*_{q_r} \in I_q \cap (\cap_{i:\ u_i=0} ext(\mathcal{C}_i))$. Q.E.D.

4.8.1 Fulfilling Assumption 1

Next, we show how the *Assumption 1* requiring that at each iteration of SPCM at least one equation $f(u_i) = 0$, $i = 1, \ldots, N$ for each cluster C_j, $j = 1, \ldots, m$ has two solutions, can always be kept valid. In other words, we show that each cluster has at least one data point $\mathbf{x}_i$, $i = 1, \ldots, N$ with $u_i > 0$ at each iteration. To this end, we will prove that (a) the *Assumption 1* is fulfilled at the initial step of SPCM (*base case*) and (b) this inductively holds also for each subsequent iteration of the algorithm (*induction step*).

(a) *Base case*: Taking into account that the initialization of SPCM is defined by the FCM algorithm and in particular eq. (2.16), it is obvious that initially each cluster C_j with representative $\boldsymbol{\theta}_j$ has at least one data point with $\|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2 \leq \gamma_j$. Focusing on a certain cluster C_j, let $\mathbf{x}_q$ be the closest to $\boldsymbol{\theta}_j$ data point, where $\boldsymbol{\theta}_j$ denotes the initial (FCM) estimate of the representative of C_j. Then, in general, $\|\mathbf{x}_q - \boldsymbol{\theta}_j\|^2 << \gamma_j$. According to Proposition C4 (see Appendix C), this data point has $u_{qj} > 0$, if $K \leq \frac{\gamma_j}{\bar{\gamma}} pe^{(2-\mu_j)(1-p)}$, where here $\mu_j = \frac{\|\mathbf{x}_q - \boldsymbol{\theta}_j\|^2}{\gamma_j} (<< 1)$. In order to fulfill the *Assumption 1* for each cluster, K should be chosen such that $K \leq \min\limits_{j=1,\ldots,m} \left[\frac{\gamma_j}{\bar{\gamma}} pe^{(2-\mu_j)(1-p)}\right]$. Also, it is $\min\limits_{j=1,\ldots,m} \left[\frac{\gamma_j}{\bar{\gamma}} pe^{(2-\mu_j)(1-p)}\right] \geq \frac{\bar{\gamma}}{\bar{\gamma}} pe^{(2-\mu_{max})(1-p)} \equiv pe^{(2-\mu_{max})(1-p)}$, where we recall that $\bar{\gamma} = \min\limits_{j=1,\ldots,m} \gamma_j$. Thus, if K is chosen so that $K \leq pe^{(2-\mu_{max})(1-p)} \equiv B(p)$, where $\mu_{max} = \max\limits_{j=1,\ldots,m} \mu_j (<< 1)$, the *Assumption 1* is satisfied. Note also that $B(p) \leq pe^{2(1-p)}$, thus the condition of Proposition C1 is valid.

In Fig. 4.11, the upper bound $B(p)$ of K is illustrated with respect to parameter p for different values of μ_{max}, so that each initial cluster has at least one data point with $u > 0$. Note that $K = 0.9$ is an appropriate value for $p = 0.5$ that ensures that the *Assumption 1* is fulfilled at the initial step of SPCM.

(b) *Induction step*: Let us focus on a specific cluster C [18]. Assume that at iteration t, its represenative is $\boldsymbol{\theta}(t)$ and it has a certain set of active points X^t [19] with its corresponding nonempty intersection of hyperspheres, denoted by I^t. Obviously, it is $CH(X^t) \subseteq (\cup_{i:u_i>0} int(\mathcal{C}_i))$. Taking into consideration that all possible positions of $\boldsymbol{\theta}(t+1)$ lie inside $CH(X^t)$, we have that $\boldsymbol{\theta}(t+1)$ will lie inside $\cup_{i:u_i>0} int(\mathcal{C}_i)$. As a consequence, there exists at least one data point of X^t that will remain active at the next iteration of the algorithm.

As a result, each cluster will have at least one data point $\mathbf{x}_i$, $i = 1, \ldots, N$ with $u_i > 0$ at each iteration of SPCM.

[18] For notational convenience, we drop the cluster index j for the rest of this subsection.

[19] We drop the index q, in order to lighten the notation. Index t shows the time dependence of the active set corresponding to C, as it evolves in time.

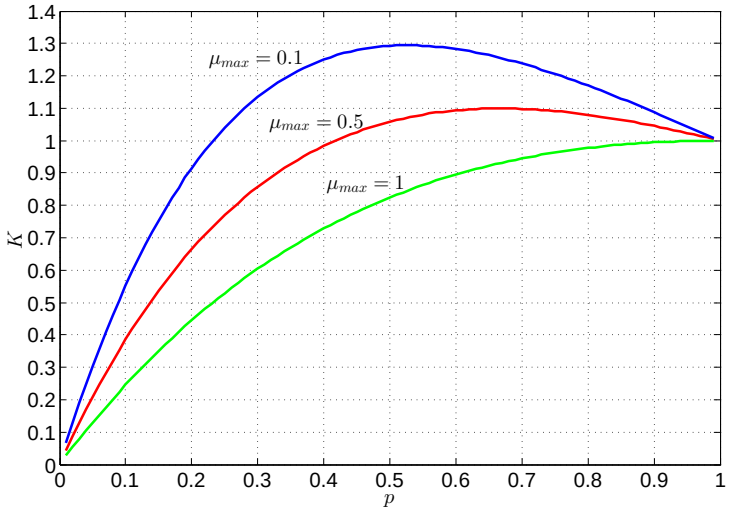


Figure 4.11: The upper bound $B(p)$ of K with respect to parameter p for different values of μ_{max}, so that each initial cluster has at least one data point with $u > 0$.

4.8.2 On the convergence of the PCM algorithm

In [72] it is proved that the sequence $T^t(U^{(0)}, \boldsymbol{\Theta}^{(0)})$ produced by PCM [10] terminates to (i) either a local minimum or a saddle point of J, or (ii) every convergent subsequence of the above sequence terminates to a local minimum or a saddle point of J. This result follows as a direct application of the Zangwill's convergence theorem ([68]). However, viewing PCM as a special case of SPCM, we can utilize the convergence results of the latter to establish stronger results for PCM, compared to those given in [72].

Let us be more specific. We focus again to a single $\boldsymbol{\theta}$ and its corresponding $\mathbf{u} = [u_1, \ldots, u_N]^T$ vector. Note that J_{PCM} results directly from J_{SPCM}, for $\lambda = 0$. In this case, the radius R (eq. (4.7)) becomes infinite for any (finite) value of p. This means that the convex hull of X, $CH(X)$, lies entirely in the intersection of the hyperspheres centered at the data points of X. As a consequence, $u_i > 0$, for $i = 1, \ldots, N$. This implies that the whole X is the active set. Also, note that for $\lambda = 0$, $f(u_i) = 0$ gives a single positive solution, i.e. $u_i = \exp(-\frac{||\mathbf{x}_i - \boldsymbol{\theta}||^2}{\gamma})$.

Let us define the solution set S for PCM as

$$S_{PCM} = \{(\mathbf{u}, \boldsymbol{\theta}) \in [0,\ 1]^N \times CH(X) :\ \nabla J|_{(\mathbf{u},\boldsymbol{\theta})} = \mathbf{0}\}.$$

The requirements for (i) the decreasing of J_{PCM}, (ii) the continuity of T_{PCM} (the operator that corresponds to PCM, defined in a fashion similar to T) and (iii) the boundness of the sequence produced by PCM can be viewed as special cases of Lemmas 3, 5 and 6, respectively, where $U_I \times I$ is replaced by $[0,\ 1]^N \times CH(X)$ [20]. Then Theorem C1

[20]The only slight difference compared to SPCM concerns the establishment of requirement (i). Specifically, in the proof of Lemma 1 in (eq. (4.24)), it turns out that for PCM, it is $\kappa_s = -\infty$, which still contradicts the fact that κ_s is finite. Also, in (4.25) in the same proof it results that $\tau_s \leq 0$, which gives also a contradiction.

(see Appendix C) guarantees that there exist fixed points for T_{PCM} and lemma 7 proves that these are strict local minima of J_{PCM} [21]. Finally, in correspondance with SPCM, the following theorem can be established for PCM.

Theorem 3: Suppose that a data set $X = \{\mathbf{x}_i \in \mathcal{R}^l,\ i = 1, \ldots, N\}$ is given. Let $J_{PCM}(\mathbf{u}, \boldsymbol{\theta})$ be defined by eq. (2.11) for $m = 1$, where $(\mathbf{u}, \boldsymbol{\theta}) \in [0,\ 1]^N \times CH(X)$. If $T_{PCM} : [0,\ 1]^N \times CH(X) \rightarrow [0,\ 1]^N \times CH(X)$ is the operator corresponding to the PCM algorithm, then for any $(\mathbf{u}^{(0)}, \boldsymbol{\theta}^{(0)}) \in [0,\ 1]^N \times CH(X)$, the PCM algorithm converges to a fixed point of T (which is a local minimum of J_{PCM}).

4.9 Conclusion

In this chapter a novel possibilistic c-means algorithm is proposed, namely SPCM, which imposes a sparsity constraint on the degrees of compatibility of each data vector with the clusters. The algorithm is initialized through FCM with the latter executed for an overestimated number of the actual number of clusters. SPCM, which results by extending the cost function of the original PCM with a sparsity promoting term, unravels the underying clustering structure much more accurately than PCM. This is achieved via the improvement on the estimation of the cluster representatives by excluding points that are distant from them in contributing to their estimation. Thus, it is able to identify closely located clusters with possibly different densities. In addition, SPCM exhibits immunity to noise and outliers. In extensive experiments, it is shown that SPCM has a steadily superior performance, compared to its ancestor PCM. Finally, in this chapter, a convergence proof for the SPCM algorithm is conducted. The main source of difficulty in the provided SPCM convergence analysis, compared to those given for previous possibilistic algorithms, relies on the updating of the degrees of compatibility, which are not given in closed form and are computed via a two-branch expression. Here, it is shown that the iterative sequence generated by SPCM coverges to a local minimum (fixed point) of its accosiated cost function J_{SPCM}. Finally, the above analysis for SPCM has been applied to the case of PCM ([10]) and gave much stronger convergence results compared to those provided in [72]. Experiments that compare SPCM with relevant state-of-the-art algorithms are provided in Chapter 5 and show that SPCM exhibits in most cases a very satisfactory clustering performance.

[21] The only thing that is differentiated in the PCM case is that $g_i^* = \frac{\gamma}{u_i^*}$. As a consequence, ε is chosen as $\varepsilon < \frac{1}{2}\sqrt{\frac{\gamma}{2}}$.

5. THE SPARSE ADAPTIVE POSSIBILISTIC C-MEANS ALGORITHM AND ITS VARIANTS

5.1 Introduction

Despite the fact that SPCM, described in the previous chapter, can handle successfully cases of closely located and different in density clusters, it still suffers from the problem of its ancestor PCM as far as the estimation of γ_j's is concerned. Specifically, the estimation of γ_j's is based on the outcomes of the FCM, which can be significantly affected by the possible presence of noise or outliers in the data, as well as by the possible differences in the variance of the clusters. Moreover, once the γ_j's have been estimated, they remain fixed during the whole course of the algorithm. Thus, poor initial estimates of γ_j's may lead SPCM to degraded performance. Furthermore, as is the case with all PCMs, SPCM may end up with coincident clusters (duplicates of the same cluster). This happens when more than one representatives are led to the center of the same physical cluster.

One way to deal with these issues is to adopt the key feature of APCM (discussed in Chapter 3) that allows γ_j's to adapt as the algorithm evolves, in the SPCM context. This will equip the algorithm with the ability to track the changes occurring in the formation of clusters. In addition, as shown in Chapter 3 of the present thesis, by adopting the updating mechanism of APCM for the adaptation of γ_j's, the true number of clusters could be determined. In the sequel, we extend SPCM in order to incorporate the adaptation of γ_j's by embedding the relevant mechanism of APCM. The resulting algorithm is called Sparse Adaptive PCM (SAPCM).

In the next section (5.2) the SAPCM algorithm is described, while in section 5.3 experimental results are provided for assessing its performance. Finally, in the last two sections, two variants of SAPCM are discussed, namely the *Sequential SAPCM* and the *Layered SAPCM*.

5.2 The Sparse Adaptive PCM (SAPCM)

The proposed SAPCM algorithm stems from the optimization of the cost function in eq. (4.1), that is

$$J_{SPCM}(\Theta, U) = \sum_{j=1}^{m} \left[\sum_{i=1}^{N} u_{ij} \|\mathbf{x}_i - \boldsymbol{\theta}_j\|^2 + \gamma_j \sum_{i=1}^{N} (u_{ij} \ln u_{ij} - u_{ij}) \right] + \lambda \sum_{i=1}^{N} \|\mathbf{u}_i\|_p^p, \; u_{ij} > 0, \quad (5.1)$$

where now γ_j varies as in APCM. Recall that γ_j is defined as

$$\gamma_j = \frac{\hat{\eta}}{\alpha} \eta_j \quad (5.2)$$

with η_j being a measure of the mean absolute deviation of C_j as it has been formed in the current iteration, α is a user-defined positive parameter (see section 3.5) and $\hat{\eta}$ is a constant defined as the minimum among all initial η_j's, i.e., $\hat{\eta} = \min_{j=1,\ldots,m_{ini}} \eta_j$, where m_{ini} is the initial number of clusters.

The parameters that need to be initialized in SAPCM are the representatives $\boldsymbol{\theta}_j$'s and the parameters γ_j's. This is carried out as for APCM where the FCM algorithm is executed first for m_{ini} clusters and the estimated by FCM $\boldsymbol{\theta}_j$'s are used as initial estimates in the SAPCM algorithm. Then the η_j's are initialized as

$$\eta_j = \frac{\sum_{i=1}^{N} u_{ij}^{FCM} \|\mathbf{x}_i - \boldsymbol{\theta}_j\|}{\sum_{i=1}^{N} u_{ij}^{FCM}}, \quad j = 1, \ldots, m_{ini}, \tag{5.3}$$

where $\boldsymbol{\theta}_j$'s and u_{ij}^{FCM}'s in eq. (5.3) are the final parameter estimates obtained by FCM[1]. Combining eqs. (5.2) and (5.3), the initialization of γ_j's is completely defined.

The parameters that are adapted in SAPCM are (a) the number of clusters, m, (b) the parameters $\boldsymbol{\theta}_j$'s, (c) the parameters u_{ij}'s and (d) the parameters γ_j's. As far as m is concerned, this is adapted in the same way as in APCM (see section 3.3), while $\boldsymbol{\theta}_j$'s and u_{ij}'s are updated as in SPCM (see section 4.4). Finally, γ_j's are adjusted as in APCM via the adaptation of η_j's (in eq. (5.2)) through

$$\eta_j(t+1) = \frac{1}{n_j(t)} \sum_{\mathbf{x}_i : u_{ij}(t) = \max_{r=1,\ldots,m(t+1)} u_{ir}(t)} \|\mathbf{x}_i - \boldsymbol{\mu}_j(t)\|. \tag{5.4}$$

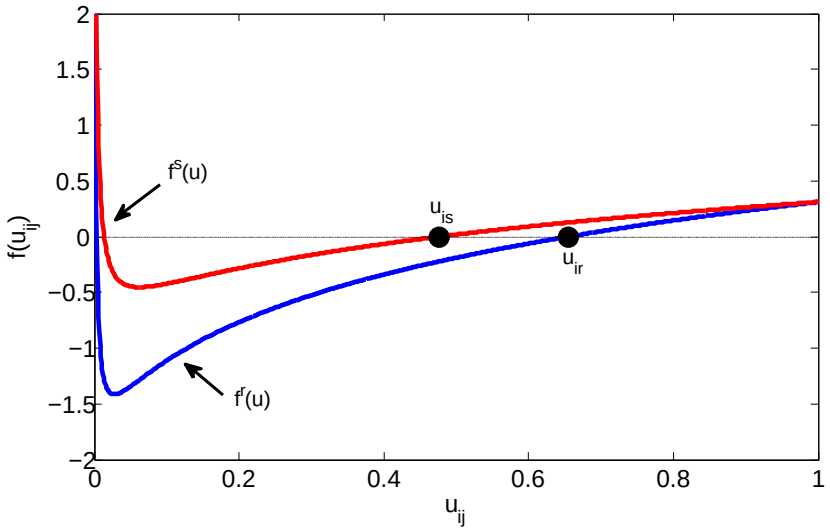


Figure 5.1: Graphical presentation of $f^r(u)$ and $f^s(u)$ for constant d, λ and p, with $\gamma_r > \gamma_s$. The largest of the two solutions of $f^r(u) = 0$ and $f^s(u) = 0$, u_{ir} and u_{is}, are also shown, respectively. It is observed that $u_{ir} \geq u_{is}$.

[1]An alternative initialization for γ_j's is proposed in [53].

A slight difference in the cluster elimination mechanism of SAPCM from that of APCM is that the N-dimensional vector $label$, the ith element of which is the index of the class that is most compatible with $\mathbf{x}_i$ (see section 3.3), may have *zero* entries, in contrast to the APCM. These zero entries correspond to data points that have no compatibility with any of the clusters (possibly outliers). This implies that care must be taken in choosing the regularizing parameter λ in eq. (5.1), since if the imposed sparsity is too strict it may happen that none of the data points is compatible with any cluster.

Let us focus for a while on the immunity of the SAPCM algorithm to its initialization with an overestimated number of clusters. Taking into account (a) that all representatives are driven to dense in data regions, due to the possibilistic nature of SAPCM, (b) that the probability to select as representative at least one point in each dense region is increased, since the overestimated number of representatives are initially selected via FCM algorithm and (c) the mechanism for reducing the number of clusters, then, in principle, the number of the representatives which move to the same dense region will be reduced to a single one. In order to get some further insight on this issue, assume that two cluster representatives $\boldsymbol{\theta}_r$, $\boldsymbol{\theta}_s$ almost coincide, which, for a given $\mathbf{x}_i$ implies that $d_{ir} \simeq d_{is} \equiv d$, but let say that $\gamma_r > \gamma_s$. Consider also the functions $f^r(u) = d + \gamma_r \ln u + \lambda p u^{p-1}$ and $f^s(u) = d + \gamma_s \ln u + \lambda p u^{p-1}$ for $u \in (0, 1]$ (see section 4.4). It is easy to see that $f^r(u) \leq f^s(u)$, for each $u \in (0, 1]$. Assume now that both have positive solutions. It is easy to verify that $u_{ir} \geq u_{is}$, where u_{ir} and u_{is} are the largest of the two solutions of $f^r(u) = 0$ and $f^s(u) = 0$, respectively (see Fig. 5.1). In the case where $u_{ir} = 0$ then, trivially follows that $u_{is} = 0$. Finally, if $u_{is} = 0$ then $u_{ir} \geq u_{is}$. Thus, the influence of the cluster with the smaller γ (γ_s) will be vanished by the influence of the one with the greater γ (γ_r), in the sense that $u_{ir} > u_{is}$, for all data points $\mathbf{x}_i \in X$. As a consequence the index s will not appear in the *label* vector and, thus, C_s will be eliminated.

The proposed SAPCM algorithm is summarized below (the choice of λ is justified later).

Algorithm 6 [Θ, U, H, $label$, m] = SAPCM(X, m_{ini}, α)

Input: X, m_{ini}, α

1: $t = 0$

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

▷ *Initialization of $\boldsymbol{\theta}_j$'s part*

3: [$\Theta(t)$, $U^{FCM}(t)$] = FCM(X, m_{ini}, 2) [2]

▷ *Initialization of η_j's part*

4: **Set:** $\eta_j(t) = \frac{\sum_{i=1}^{n} u_{ij}^{FCM} \|\mathbf{x}_i - \boldsymbol{\theta}_j(t)\|}{\sum_{i=1}^{n} u_{ij}^{FCM}}$, $j = 1, ..., m(t)$

5: **Set:** $\hat{\eta} = \min_{j=1,...,m(t)} \eta_j(t)$

---

[2] See Algorithm 2 of Chapter 2. We set the fuzzifier $q = 2$.

6: **Set:** $\gamma_j(t) = \hat{\eta}\eta_j(t)/\alpha$, $j = 1, ..., m(t)$

7: **Set:** $\lambda(t) = K\frac{\min_{j=1,...,m(t)}\gamma_j(t)}{p(1-p)\mathrm{e}^{2-p}}$, $K = 0.1$

8: **repeat**

▷ *Update U part*

9: **Update:** $U(t)$ as in SPCM (see section 4.4)

▷ *Update Θ part*

10: $\boldsymbol{\theta}_j(t+1) = \sum_{i=1}^{N} u_{ij}(t)\mathbf{x}_i \Big/ \sum_{i=1}^{N} u_{ij}(t)$, $j = 1, ..., m(t)$

▷ *Possible cluster elimination part*

11: **for** $i \leftarrow 1$ **to** N **do**

12: **Determine:** $u_{ir}(t) = \max_{j=1,...,m(t)} u_{ij}(t)$

13: **if** $u_{ir}(t) \neq 0$ **then**

14: **Set:** $label(i) = r$

15: **else**

16: **Set:** $label(i) = 0$

17: **end if**

18: **end for**

19: $p = 0$ *//number of removed clusters at iteration t*

20: **for** $j \leftarrow 1$ **to** $m(t)$ **do**

21: **if** $j \notin label$ **then**

22: **Remove:** C_j (and **renumber** accordingly $\Theta(t+1)$, the columns of $U(t)$ and γ_j's)

23: $p = p + 1$

24: **end if**

25: **end for**

26: $m(t+1) = m(t) - p$

▷ *Adaptation of η_j's part*

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

28: **Set:** $\gamma_j(t+1) = \hat{\eta}\eta_j(t+1)/\alpha$, $j = 1, ..., m(t+1)$

29: **Set:** $\lambda(t+1) = K \frac{\min_{j=1,\ldots,m(t+1)} \gamma_j(t+1)}{p(1-p)\mathrm{e}^{2-p}}$, $K = 0.1$

30: $t = t + 1$

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

32: **return** $\Theta = \{\boldsymbol{\theta}_1(t), \boldsymbol{\theta}_2(t), \ldots, \boldsymbol{\theta}_{m(t)}(t)\}$, $U = [u_{ij}(t-1)]$, $H = [\eta_1(t), \ldots, \eta_{m(t)}(t)]$, $label$, $m = m(t)$

In the sequel, we give some very demanding experimental set ups which exhibit the enhanced abilities of SAPCM compared to APCM[3].

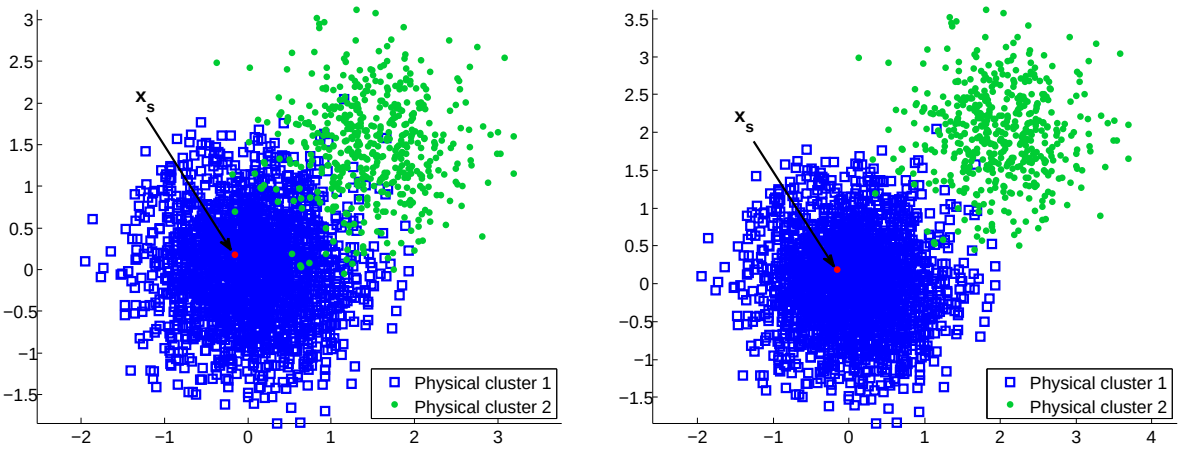
Example 1: Consider the set up of Example 1 of Chapter 4, where now C_1 and C_2 consist of 2000 and 500 points, respectively. Note that the clusters have the same variances yet even more different densities compared to the data set of Example 1 of Chapter 4, while at the same time they are located very close to each other, as shown in Fig. 5.2a. Table 5.1 shows the clustering results of APCM and SAPCM and Figs. 5.3a and 5.3b depict the performance of APCM and SAPCM, respectively, with their parameter α being chosen as stated in the figure caption (after fine-tuning). As it can be deduced from Fig. 5.3 and Table 5.1, APCM fails to uncover the underlying clustering structure, whereas SAPCM distinguishes the two physical clusters and achieves very satisfactory results in terms of RM, SR and MD. To see why this happens, let us focus on $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ in Figs. 5.3a and 5.3b. Clearly, APCM fails to recover C_2 since, in determining the next location of $\boldsymbol{\theta}_2$ the many small contributions from the points of C_1 gradually prevail over the larger but less contributions from the points of C_2. Note that this happens despite the fact that APCM adjusts dynamically the γ_j's and it is due to the combination of (a) the strict positivity of all u_{ij}'s, (b) the very different cluster densities and (c) the closeness of the clusters. However, this is not the case for SAPCM, since the latter annihilates the contributions of the points of C_1 in the determination of the next location of $\boldsymbol{\theta}_2$, via the imposition of sparsity.

Table 5.1: Performance of APCM and SAPCM for the data sets of Examples 1 and 2.

	Data Set	α	m_{ini}	m_{final}	RM	SR	MD
APCM	Ex. 1	1.5	5	1	67.99	80.00	1.0363
SAPCM	Ex. 1	2	5	2	90.07	94.76	0.0673
APCM	Ex. 2	1.5	5	2	97.86	98.92	0.0183
SAPCM	Ex. 2	1	5	2	97.78	98.88	0.0142

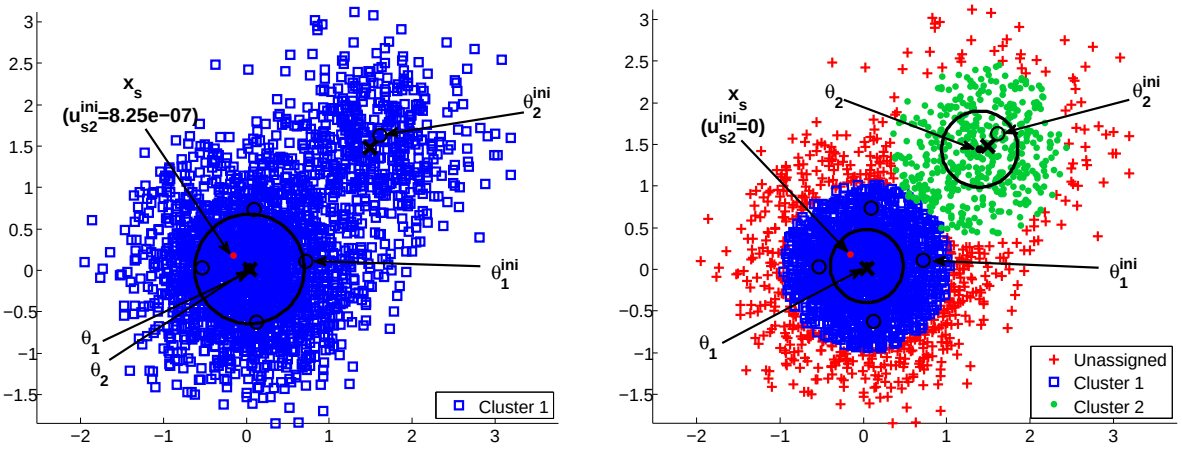
Example 2: Consider the same two-dimensional data set but now the means of the two normal distributions are $\mathbf{c}_1 = [0, 0]^T$ and $\mathbf{c}_2 = [2, 2]^T$, respectively, as shown in Fig. 5.2b. Table 5.1 shows the clustering results of APCM and SAPCM and Figs. 5.4a and 5.4b depict the performance of APCM and SAPCM, respectively. As it can be deduced, APCM is now able to uncover the underlying clustering structure. However, SAPCM manages to detect even more accurately the true centers of the clusters (as MD index indicates).

[3]Note that SPCM is not examined here, due to the fact that only APCM and SAPCM are able to determine



(a) Data set of Example 1 **(b) Data set of Example 2**

Figure 5.2: (a) The data set of Example 1, (b) The data set of Example 2. $\mathbf{x}_s$ is a specific typical point that will also be considered in Figs. 5.3 and 5.4.

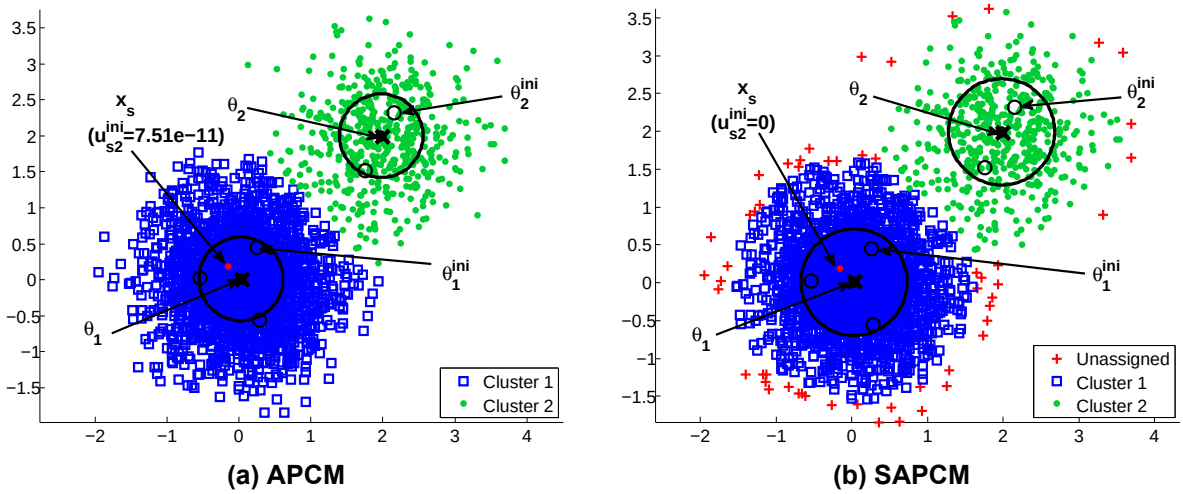


(a) APCM **(b) SAPCM**

Figure 5.3: The clustering results of Example 1 for (a) APCM, $m_{ini} = 5$ and $\alpha = 1.6$ (b) SAPCM, $m_{ini} = 5$ and $\alpha = 2$. See also the caption of Fig. 5.2. Note that the degree of compatibility of $\mathbf{x}_s$ (defined in Fig. 5.2) with θ_2^{ini}, u_{s2}^{ini}, is positive in APCM and zero in SAPCM.

Remark 1: In SAPCM the parameter λ is chosen as in SPCM, as eq. (4.5) indicates. Note that in SAPCM, the parameters γ_j's are updated during the execution of the algorithm, thus the parameter λ should also be updated after the adaptation of γ_j's (see line 29 in Algorithm 6). Moreover, in SAPCM the parameter K should take much smaller values than in SPCM, due to the definition of γ_j's. This has to do with the fact that in SAPCM the adaptation of the parameters γ_j's leads to more accurate estimates for the variances of the clusters (see the radius of the circles ($\sqrt{\gamma_j}$) in Figs. 4.5a, 4.5b for SPCM and the corresponding ones for SAPCM in Figs. 5.3b, 5.4b). Taking into account that (a) the choice of the value of

the true number of clusters.



(a) APCM **(b) SAPCM**

Figure 5.4: The clustering results of Example 2 for (a) APCM, $m_{ini} = 5$ and $\alpha = 1.5$ (b) SAPCM, $m_{ini} = 5$ and $\alpha = 1$. See also the caption of Fig. 5.2. In this case, u_{s2}^{ini} is significantly smaller than in Fig. 5.3a.

λ via eq. (4.5) imposes sparsity for all the points at distance greater than $\min_{j=1,...,m} \gamma_j$ from a given representative and (b) γ_j's in SAPCM are of much smaller sizes with respect to their corresponding ones in SPCM, values of K close to 1 would lead to such a large degree of sparsity (as indicated by $f(u_{ij})$ in eq. (4.2)), where the cluster representatives could hardly move (see line 10 in Algorithm 6). Extensive experimentation indicated that values around 0.1 are the most appropriate. Therefore, in all SAPCM experiments we set $K = 0.1$.

5.3 Experimental results

In this section, we assess the performance of the SAPCM algorithm in several experimental settings and illustrate the results. More specifically, we use two-dimensional simulated data sets as well as a real-world data set (Iris [73]) to evaluate its performance in comparison with several other related algorithms. We compare the clustering performance of SAPCM with that of the k-means, the FCM, the PCM [10], the UPC [20], the UPFC [28], the PFCM [19], the SPCM-L_1 [76], the APCM (chapter 3) and the SPCM (chapter 4) algorithms, which all result from cost optimization schemes. For a fair comparison, the representatives $\boldsymbol{\theta}_j$'s of all algorithms (except for SPCM-L_1) are initialized based on the FCM scheme and the parameters of each algorithm are first fine tuned. Moreover, in PCM, UPC, UPFC, PFCM and SPCM, duplicate clusters are removed after their termination. In order to compare a clustering with the true data label information, we utilize again the RM, SR and the MD indices defined in section 3.7.2. In particular, in Experiments 1 and 2 the SR of each physical cluster ($SR_{c_j}, j = 1, ..., m$) is presented, which measures the percentage of the points of each physical cluster that have been correctly labeled by each algorithm. Finally, the number of iterations and the total time required for the convergence

of each algorithm, is provided.

Experiment 1: Consider a two-dimensional data set consisting of $N = 5300$ points, where three clusters C_1, C_2 and C_3 are formed. Each cluster is modelled by a normal distribution. The means of the distributions are $\mathbf{c}_1 = [0.27, 7.99]^T$, $\mathbf{c}_2 = [6.28, 1.49]^T$ and $\mathbf{c}_3 = [7.81, 3.76]^T$, respectively, while their covariance matrices are set to $3 \cdot I_2$, $0.5 \cdot I_2$ and $0.01 \cdot I_2$, respectively. A number of 200 points are generated by the first distribution, 100 points are generated by the second one and 5000 points are generated by the third one. Note that C_2 and C_3 clusters are very close to each other and they have a big difference in their variances (see Fig. 5.5a). Also, note the difference in the density among the three clusters.

Table 5.2: Performance of clustering algorithms for the data set of Experiment 1.

	m_{ini}	m_{final}	SR$_{c_1}$	SR$_{c_2}$	SR$_{c_3}$	MD	Iter	Time
k-means	3	3	51	0	100	3.4066	2	0.265
k-means	5	5	51	94	51.48	0.5369	20	0.202
FCM	3	3	51	0	100	3.3432	110	0.140
FCM	5	5	50.50	93	51.62	0.5537	86	0.218
PCM	5	2	100	0	100	0.9242	15	0.514
PCM	10	2	100	0	100	0.9254	18	1.185
UPC ($q = 1.5$)	5	4	50	95	100	0.4589	65	0.390
UPC ($q = 1.2$)	10	4	50	95	100	0.4480	89	0.910
UPFC ($a = 5$, $b = 1$, $q = 2$, $n = 1.5$)	5	4	50.50	96	100	0.4170	41	0.390
UPFC ($a = 5$, $b = 1$, $q = 2.2$, $n = 3$)	10	3	100	94	100	0.3601	190	2.940
PFCM ($K = 1$, $a = 1$, $b = 5$, $q = 1.5$, $n = 1.5$)	5	4	51.50	100	100	0.4573	38	0.380
PFCM ($K = 1$, $a = 2$, $b = 1$, $q = 2$, $n = 1.2$)	10	5	44	97	100	0.4011	60	0.880
SPCM-L$_1$ ($\lambda = 15$, $q = 2$)	-	2	76	0	100	1.1831	6	0.031
APCM ($\alpha = 0.3$)	5	4	53	100	100	0.4469	73	0.390
APCM ($\alpha = 0.3$)	10	4	52.50	100	100	0.4748	90	0.889
SPCM ($K = 0.9$)	5	2	100	0	100	0.9256	15	3.276
SPCM ($K = 0.9$)	10	2	100	0	100	0.9263	19	7.769
SAPCM ($\alpha = 0.18$)	5	3	100	100	100	0.3222	91	13.40
SAPCM ($\alpha = 0.15$)	10	3	100	100	100	0.3020	100	18.94

Table 5.2 shows the results of all algorithms for Experiment 1. Fig. 5.5b and Fig. 5.5c show the clustering obtained using the k-means and FCM algorithms, respectively, both for $m_{ini} = 3$. Figs. 5.5d, 5.5e, 5.5f, 5.5g, 5.5h, 5.5i and 5.5j, depict the performance of PCM, UPC, UPFC, PFCM, SPCM-L_1, APCM and SPCM, respectively, with their parameters chosen (after fine-tuning) as stated in the caption. In addition, the circles, centered at each $\boldsymbol{\theta}_j$ and having radius $\sqrt{\gamma_j}$ (as they have been computed after the convergence of the algorithms), are also drawn.

As it can be deduced from Fig. 5.5 and Table. 5.2, even when the k-means and the FCM are initialized with the (unknown in practice) true number of clusters ($m = 3$), they fail to unravel the underlying clustering structure mainly due to the big difference in the variances and densities between clusters. The classical PCM also fails to detect the physical cluster 2, because it is located very close to the densest physical cluster. The UPC algorithm has been fine tuned so that the parameters γ_j's, which remain fixed during its execution

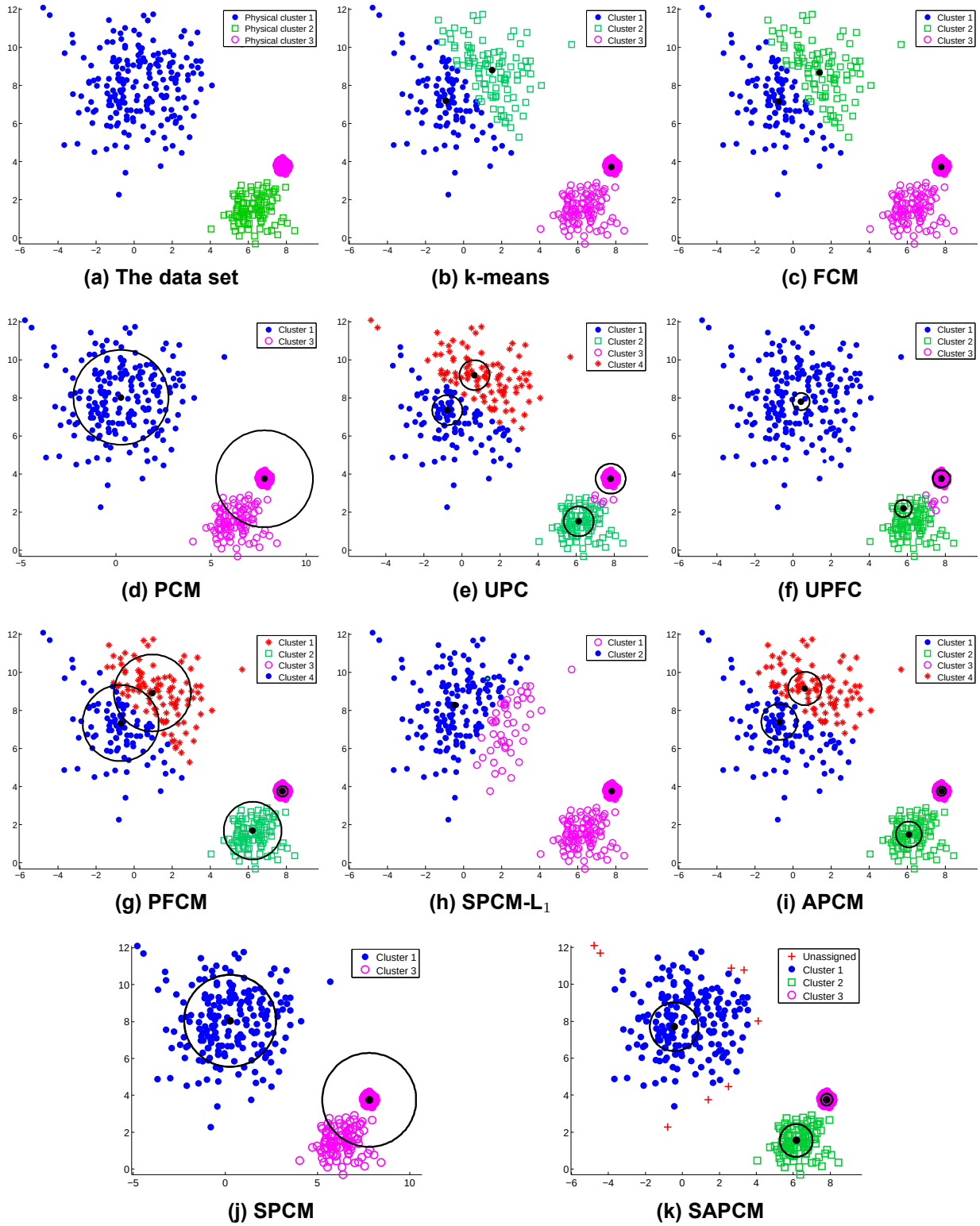


Figure 5.5: (a) The data set of Experiment 1. Clustering results for (b) k-means, $m_{ini} = 3$**, (c) FCM,** $m_{ini} = 3$**, (d) PCM,** $m_{ini} = 5$**, (e) UPC,** $m_{ini} = 5$, $q = 1.5$**, (f) UPFC,** $m_{ini} = 10$, $\alpha = 5$, $\beta = 1$, $q = 2.2$, $n = 3$**, (g) PFCM,** $m_{ini} = 5$, $K = 1$, $\alpha = 1$, $\beta = 5$, $q = 1.5$, $n = 1.5$**, (h) SPCM-**L_1**,** $\lambda = 15$, $q = 2$ **(i) APCM,** $m_{ini} = 5$, $\alpha = 0.3$**, (j) SPCM,** $m_{ini} = 5$**, and (k) SAPCM,** $m_{ini} = 10$ **and** $\alpha = 0.15$**.**

and are the same for all clusters, get small enough values, in order to identify cluster C_2. However, it splits the high variance/low density cluster C_1 in two clusters. The same seems to hold for the PFCM algorithm, after fine tuning of its several parameters. The UPFC algorithm produces 3 clusters, at the cost of a computationally demanding fine tuning of the (several) parameters it involves. The APCM algorithm also splits the big variance cluster in two subclusters, failing to detect the underlying clustering structure. On the other hand, SPCM identifies two clusters with high accuracy with respect to the center of the actual clusters, but misses the third one. Finally, as it is deduced from Table 5.2, the SAPCM algorithm manages to identify all clusters, achieving the best SR and MD results and estimating very accurately the true centers of the clusters, since it exhibits the minimum MD among all algorithms.

Experiment 2: Consider the dataset of Experiment 1, where 50 data points are now added randomly as noise in the region where data live (see Fig. 5.6a). It can be seen that APCM and SAPCM algorithms are the only algorithms that distinguish all clusters. In addition, SAPCM keeps MD at low values, whereas all other algorithms conclude to higher MD values compared to the results of Experiment 1 (see Tables 5.2 and 5.3). Finally, as shown in Fig. 5.6, SAPCM is the only algorithm that identifies the noisy points of the data set and ignores them in the updating of the location of the cluster representatives.

Table 5.3: Performance of clustering algorithms for the data set of Experiment 2.

	m_{ini}	m_{final}	SR_{c_1}	SR_{c_2}	SR_{c_3}	MD	Iter	Time
k-means	3	3	54.50	0	100	3.8296	8	0.156
k-means	5	5	99.50	94	50.96	0.0843	35	0.203
FCM	3	3	56	0	100	3.4345	75	0.110
FCM	5	5	99.50	92	38.92	0.3334	129	0.375
PCM	5	1	0	0	100	3.7899	9	0.421
PCM	10	2	99	0	97.60	0.9254	29	1.943
UPC ($q = 1.5$)	5	4	50	95	100	0.4424	80	0.328
UPC ($q = 1.3$)	10	4	50	95	100	0.4517	113	1.186
UPFC ($a = 1$, $b = 1$, $q = 2.5$, $n = 2$)	5	2	100	0	100	1.1388	60	0.421
UPFC ($a = 5$, $b = 1$, $q = 2.5$, $n = 2$)	10	2	100	0	100	1.1346	151	2.044
PFCM ($K = 1$, $a = 1$, $b = 1$, $q = 1.5$, $n = 1.5$)	5	2	100	0	100	0.9519	45	0.343
PFCM ($K = 1$, $a = 1$, $b = 1$, $q = 1.2$, $n = 1.5$)	10	2	98.50	0	100	0.9575	61	1.358
SPCM-L_1 ($\lambda = 17$, $q = 2$)	-	3	58.50	0	100	4.1291	9	0.016
APCM ($\alpha = 0.3$)	5	3	100	100	100	0.3150	83	0.374
APCM ($\alpha = 0.3$)	10	4	97	100	100	0.3518	93	0.655
SPCM ($K = 0.9$)	5	2	100	0	100	0.9117	19	4.695
SPCM ($K = 0.9$)	10	2	100	0	100	0.9118	13	5.991
SAPCM ($\alpha = 0.24$)	5	3	100	100	100	0.3808	202	27.82
SAPCM ($\alpha = 0.19$)	10	3	100	100	100	0.3193	122	21.20

Experiment 3: Let us consider the Iris data set ([73]) consisting of $N = 150$, 4-dimensional data points that form three classes, each one having 50 points. In this 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 5.4, k-means and FCM work well, only if they are initialized with the true number of clusters ($m_{ini} = 3$). The classical PCM and SPCM fail to end up with $m_{final} = 3$ clusters, independently of the choice of the initial number of clusters. On the contrary,

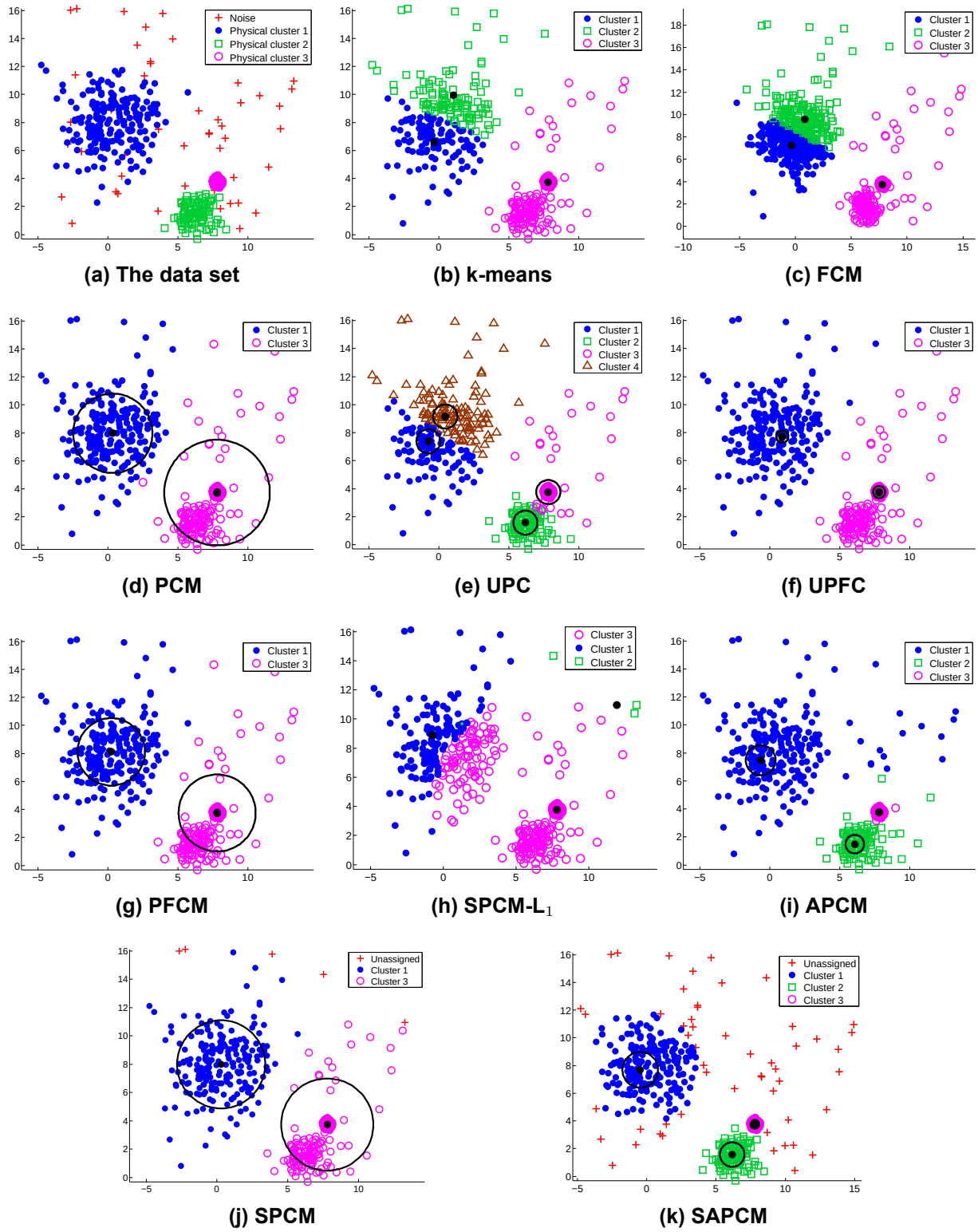


Figure 5.6: (a) The data set of Experiment 2. Clustering results for (b) k-means, $m_{ini} = 3$**, (c) FCM,** $m_{ini} = 3$**, (d) PCM,** $m_{ini} = 10$**, (e) UPC,** $m_{ini} = 5$, $q = 1.5$**, (f) UPFC,** $m_{ini} = 10$, $\alpha = 5$, $\beta = 1$, $q = 2.5$, $n = 2$**, (g) PFCM,** $m_{ini} = 5$, $K = 1$, $\alpha = 1$, $\beta = 1$, $q = 1.5$, $n = 1.5$**, (h) SPCM-**L_1, $\lambda = 17$, $q = 2$ **(i) APCM,** $m_{ini} = 5$, $\alpha = 0.4$**, (j) SPCM,** $m_{ini} = 5$**, and (k) SAPCM,** $m_{ini} = 10$ **and** $\alpha = 0.18$**.**

Table 5.4: Performance of clustering algorithms for the Iris data set.

	m_{ini}	m_{final}	RM	SR	MD	Iter	Time
k-means	3	3	87.97	89.33	0.1271	3	0.30
k-means	10	10	76.64	40.00	0.7785	4	0.13
FCM	3	3	87.97	89.33	0.1287	19	0.02
FCM	10	10	76.16	36.00	0.7793	35	0.02
PCM	3	2	77.19	66.67	0.5428	19	0.11
PCM	10	2	77.63	66.67	0.5286	28	0.11
UPC ($q = 4$)	3	3	91.24	92.67	0.1438	26	0.03
UPC ($q = 2.4$)	10	3	81.96	81.33	0.5569	150	0.11
UPFC ($a = 1$, $b = 5$, $q = 4$, $n = 2$)	3	3	91.24	92.67	0.1642	32	0.03
UPFC ($a = 1$, $b = 1.5$, $q = 2.5$, $n = 2$)	10	3	81.96	81.33	0.5566	180	0.16
PFCM ($K = 1$, $a = 1$, $b = 10$, $q = 7$, $n = 2$)	3	3	90.55	92.00	0.1833	17	0.03
PFCM ($K = 1$, $a = 1$, $b = 1.5$, $q = 2$, $n = 2$)	10	3	84.64	85.33	0.5411	92	0.05
SPCM-L_1 ($\lambda = 4.5$, $q = 2$)	-	3	66.65	58.67	0.69.04	13	0.02
APCM ($\alpha = 3$)	3	3	91.24	92.67	0.1405	26	0.03
APCM ($\alpha = 1$)	10	3	84.15	84.67	0.4030	61	0.06
SPCM ($K = 1.2$)	3	3	83.22	83.33	0.3631	27	0.14
SPCM ($K = 0.95$)	10	3	79.38	76.00	0.2151	35	0.36
SAPCM ($\alpha = 2.2$)	3	3	91.24	92.67	0.1419	33	0.16
SAPCM ($\alpha = 0.8$)	10	3	84.15	84.67	0.4224	60	0.34

the UPC, the PFCM, the UPFC, the APCM and the SAPCM algorithms, after appropriate fine tuning of their parameters, produce very accurate results in terms of the RM, SR and MD metrics. However, the APCM and SAPCM algorithms estimate more accurately the centers of the true clusters compared to the other algorithms. It is noted again that the main drawback of PFCM and UPFC is the requirement for fine tuning of several parameters, which increases excessively the computational load required for detecting the appropriate combination of the values of the parameters that achieves the best clustering performance. Finally, the SPCM-L_1 algorithm concludes also to three clusters, however with degraded clustering performance.

5.4 The Sequential SAPCM (SeqSAPCM)

In this section, an iterative bottom-up version of SAPCM, termed Sequential SAPCM (SeqSAPCM), which involves in its heart the SAPCM, is presented[4]. Unlike SAPCM, SeqSAPCM does not initially require any overestimation of the number of clusters. On the contrary, it first identifies two of the clusters that underlie in the data set and then, at each iteration, SAPCM is applied increasing the number of clusters by one at a time, until the true number of clusters is (hopefully) reached. From this point of view, if SAPCM 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.

[4]A preliminary version of SeqSAPCM has been presented in [58].

5.4.1 The SeqSAPCM algorithm

Note that in the framework of SeqSAPCM, the SAPCM algorithm cannot initialize by itself the parameters $\boldsymbol{\theta}$'s and η's [5] and therefore, neither $\hat{\eta} = \min_j \eta_j$. It rather takes as input the initial estimates of these parameters. To denote this explicitly we write[6]

$$[\Theta, U, H, label, m] = SAPCM(X, m_{ini}, \alpha, \Theta^{ini}, H^{ini}, \hat{\eta}), \tag{5.5}$$

which implies that SAPCM (see Algorithm 6) is used *excluding the initialization phase* (lines 3, 4 and 5).

The SeqSAPCM algorithm works as follows. Initially, the two most distant points of X, say $\mathbf{x}_s$ and $\mathbf{x}_t$, 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 $\boldsymbol{\theta}_2^{ini}$ [7]. Thus, at this time it is $m = 2$ and $\Theta^{ini} = \{\boldsymbol{\theta}_1^{ini}, \boldsymbol{\theta}_2^{ini}\}$. The initial estimates of the parameters η_1 and η_2 (η_1^{ini}, η_2^{ini}) that correspond to the first two clusters are computed as the *maximum* of the following two quantities:

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

  $$d_{\mathsf{max}} = \max_{i=1,\ldots,N} \|\mathbf{x}_i - \mathbf{x}_i^{nei}\|,$$

  where $\mathbf{x}_i^{nei} \in X$ is the nearest neighbour of $\mathbf{x}_i$.

- d_{slope}^j, which is determined as follows: The Euclidean distances of $\boldsymbol{\theta}_j^{ini}$ from its q nearest neighbours in X [8], d_s^j, $s = 1, \ldots, q$, are computed and plotted in increasing order. The neighbouring 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 Euclidean distance between $\boldsymbol{\theta}_j^{ini}$ and its rth neighbour).

Thus $\eta_j^{ini} = \mathsf{max}\{d_{\mathsf{max}}, d_{slope}^j\}$ [9], $H^{ini} = \{\eta_1^{ini}, \eta_2^{ini}\}$ and $\hat{\eta} = \min_j \eta_j^{ini}$. Then, we run the SAPCM algorithm (5.5) 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. The next representative is initialized by executing only a single step

[5] Recall that the latter are needed for the estimation of the parameters γ of SAPCM.

[6] Note that H^{ini} contains the initial parameters η that are needed for the estimation of the parameters γ of SAPCM.

[7] In order to avoid high computational burden, this step is carried out approximately using the method described in [80].

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

[9] See below for the rationale about the choice of η_j^{ini}.

of the *Max-Min* algorithm [81], as follows. For each $\mathbf{x}_i \in X$ 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 $\mathbf{x}_r$ is the initial estimate of the next representative ($\boldsymbol{\theta}_3$), that is $\boldsymbol{\theta}_3^{ini} \equiv \mathbf{x}_r$. In mathematical terms, $\mathbf{x}_r$ is the point that corresponds to the distance $\max_{i=1,\ldots,N}(\min_{j=1,\ldots,m} d(\mathbf{x}_i, \boldsymbol{\theta}_j))$. Note that this procedure for initializing the representatives aims at increasing the probability to select a point from each one of the physical clusters underlying the data set. In the sequel, η_3^{ini} is computed following the same procedure as for the previous ones. Then, 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 SeqSAPCM algorithm can be stated as follows:

Algorithm 7 [Θ, U] = SeqSAPCM(X, α)

Input: X, α

1: $t = 0$

2: $m(t) = 2$

3: **Initialize:** $\Theta^{ini} = \{\boldsymbol{\theta}_1^{ini}, \boldsymbol{\theta}_2^{ini}\}$, that is, the two most distant points in X.

4: **Initialize:** $H^{ini} = \{\eta_1^{ini}, \eta_2^{ini}\}$ as described in the text.

5: **Set:** $\hat{\eta} = \min_{j=1,2} \eta_j^{ini}$

6: $[\Theta(t), U(t), H(t), label(t), m'(t)] = SAPCM(X, m(t), \alpha, \Theta^{ini}, H^{ini}, \hat{\eta})$

7: **if** $m'(t) < m(t)$ (*single cluster case*) **then** go to line 14

8: **end if**

9: **repeat**

10: $t = t + 1$

11: **Set:** $\Theta^{ini} = \Theta(t-1) \cup \{\boldsymbol{\theta}_{new}^{ini}\}$, where $\boldsymbol{\theta}_{new}^{ini}$ is the initial estimate of the next cluster, that is, the point that corresponds to the distance $\max_{i=1,\ldots,N}(\min_{j=1,\ldots,m(t-1)} d(\mathbf{x}_i, \boldsymbol{\theta}_j))$.

12: **Set:** $H^{ini} = H(t-1) \cup \{\eta_{new}^{ini}\}$, where η_{new}^{ini} is computed as described in the text.

13: $m(t) = m(t-1) + 1$

14: $[\Theta(t), U(t), H(t), label(t), m'(t)] = SAPCM(X, m(t), \alpha, \Theta^{ini}, H^{ini}, \hat{\eta})$

15: **until** $m'(t) < m(t)$ (no new cluster is detected)

16: **return** Θ, U

Although 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 cluster points[10], $d_{\max}$ 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.

Some remarks on the proposed SeqSAPCM are now in order.

Remark 1: 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.

Remark 2: The SeqSAPCM algorithm actually requires fine tuning only for the parameter α. On the other hand, SAPCM requires additional fine tuning for the initial number of clusters.

Remark 3: 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.

5.4.2 Experimental Results

In this subsection, we assess the performance of SeqSAPCM 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, the SPCM and the SAPCM algorithms. In order to evaluate a clustering outcome, we use the Rand Measure (RM), the Generalized Rand Measure (GRM), which is a generalization of RM that is described in [82] and takes into account the *degrees of compatibility* of all data points to clusters, and the classical Success Rate (SR). 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 time (in seconds) required for the convergence of each algorithm is provided.

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

[10] Usually, they are "peripheral" points of the clusters.

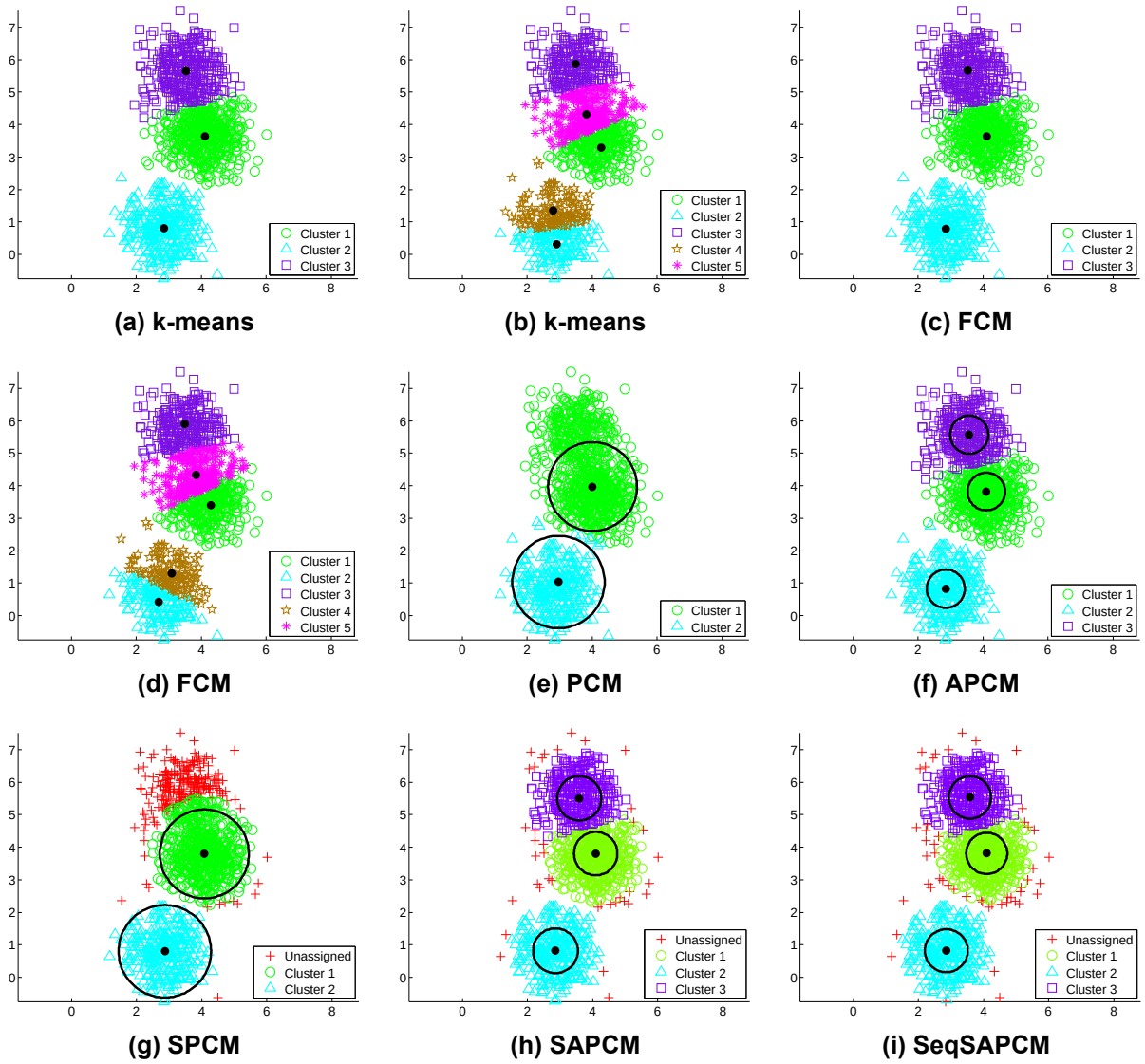


Figure 5.7: Clustering results of Experiment 1 for (a) k-means, $m_{ini} = 3$, (b) k-means, $m_{ini} = 5$, (c) FCM, $m_{ini} = 3$, (d) FCM, $m_{ini} = 5$, (e) PCM, $m_{ini} = 10$, (f) APCM, $m_{ini} = 5$, $\alpha = 2$, (g) SPCM, $m_{ini} = 3$, (h) SAPCM, $m_{ini} = 5$, $\alpha = 1.5$ and (i) SeqSAPCM, $\alpha = 1.3$.

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. 5.7a, 5.7b show the clustering outcome obtained using the k-means algorithm with $m_{ini} = 3$ and $m_{ini} = 5$, respectively. Similarly, in Figs. 5.7c, 5.7d we present the corresponding results for FCM. Fig. 5.7e depicts the performance of PCM for $m_{ini} = 10$, while, in addition, it shows the

Table 5.5: The results of the Experiment 1 synthetic data set

	m_{ini}	m_{final}	RM	GRM	SR	$Time$
k-means	3	**3**	**93.62**	-	**95.36**	0.14
k-means	5	5	81.14	-	61.82	0.17
k-means	10	10	72.84	-	31.00	0.14
k-means	15	15	70.27	-	25.91	0.25
FCM	3	**3**	**93.62**	**79.10**	**95.36**	0.02
FCM	5	5	81.38	67.81	63.09	0.03
FCM	10	10	72.97	60.78	31.82	0.06
FCM	15	15	70.14	57.96	21.64	0.22
PCM	3	2	74.19	72.27	71.73	0.11
PCM	5	2	73.35	71.94	70.73	0.16
PCM	10	2	74.33	73.03	72.00	0.45
PCM	15	2	74.19	72.80	71.73	0.72
APCM ($\alpha = 2$)	3	**3**	**93.74**	**92.69**	**95.45**	0.15
APCM ($\alpha = 2$)	5	**3**	**93.74**	**92.84**	**95.45**	0.15
APCM ($\alpha = 1.3$)	10	3	93.40	91.85	95.18	0.50
APCM ($\alpha = 1.15$)	15	3	93.51	91.70	95.27	0.70
SPCM	3	2	74.50	74.32	72.18	1.48
SPCM	5	2	74.04	73.62	71.55	4.15
SPCM	10	2	74.50	74.23	72.18	6.35
SPCM	15	2	74.33	74.14	72.00	10.30
SAPCM ($\alpha = 1.5$)	3	**3**	**93.74**	**92.39**	**95.45**	0.84
SAPCM ($\alpha = 1.5$)	5	**3**	**93.74**	**92.56**	**95.45**	1.09
SAPCM ($\alpha = 1.2$)	10	3	93.51	92.41	95.27	1.97
SAPCM ($\alpha = 1$)	15	3	93.51	92.13	95.27	3.88
SeqSAPCM ($\alpha = 1.3$)	-	**3**	**93.74**	**92.55**	**95.45**	4.37

circled regions, centered at each $\boldsymbol{\theta}_j$ and having radius equal to $\sqrt{\gamma_j}$, in which C_j has increased influence. Fig. 5.7f shows the results of APCM with $m_{ini} = 5$ and $\alpha = 2$, Fig. 5.7g shows the results of SPCM with $m_{ini} = 3$ and Fig. 5.7h shows the results of SAPCM with $m_{ini} = 5$ and $\alpha = 1.5$. Finally, Fig. 5.7i shows the results of SeqSAPCM with $\alpha = 1.3$. Moreover, Table 5.5 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. 5.7 and Table 5.5, 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 underlying 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 γ_j's in order to cope with this situation. APCM and SAPCM algorithms steadily result in $m_{final} = 3$ clusters independently of the initialization of the number of clusters m_{ini}, under appropriate values of parameter α. This is not the case for SPCM, which fails to unravel the less dense cluster C_3 that is located next to the denser one (C_1). Finally, SeqSAPCM produces very accurate results after cross validating just a single parameter (α).

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 [83]), as shown in Fig. 5.8.

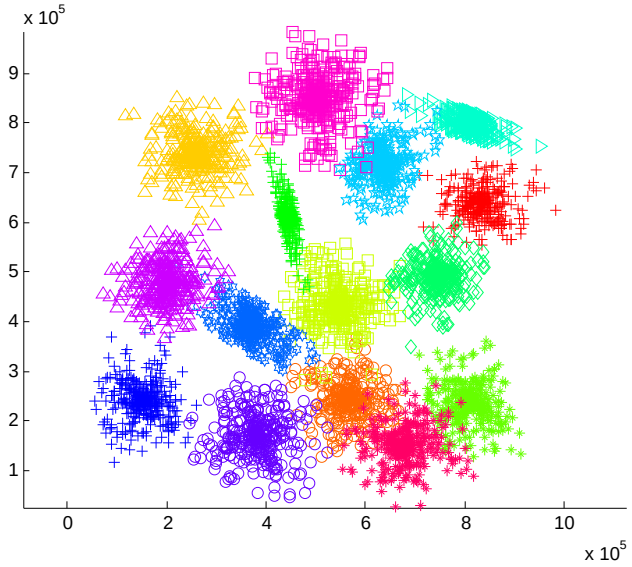


Figure 5.8: The data set in Experiment 2. Colors indicate the true label information.

All clusters are modelled by normal distributions with different covariance matrices. As it is shown in Table 5.6, k-means and FCM work well when they are 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 produce very accurate results for various initial values of m_{ini}. Although the SPCM algorithm achieves better results than PCM, however, it fails to end up with $m_{final} = 15$ clusters. Finally, SeqSAPCM is able to capture the underlying clustering structure very accurately.

Experiment 3: Let us consider the real *Iris* data set ([73]) considered in experiment 4 of chapter 3. As it is shown in Table 5.7, k-means and FCM work well 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 properly selected overestimated value for the initial number of clusters, produce very accurate results. Finally, SeqSAPCM reveals also the actual number of clusters and is very accurate, offering though a slightly degraded performance compared to SAPCM and APCM (note that, due to the small size of the *Iris* data set the difference in SR corresponds to about 12-13 wrongly assigned data vectors in SeqSAPCM than in SAPCM and APCM).

5.5 The Layered SAPCM (L-SAPCM)

In this section, another variant of SAPCM called *Layered SAPCM (L-SAPCM)* is presented. L-SAPCM performs a layered processing, where at each layer it uses as structural

Table 5.6: The results of the Experiment 2 synthetic data set

	m_{ini}	m_{final}	RM	GRM	SR	$Time$
k-means	15	**15**	**99.23**	-	**97.00**	0.33
k-means	20	20	98.42	-	88.22	1.79
k-means	25	25	97.67	-	78.58	7.11
FCM	15	**15**	**99.23**	**80.09**	**97.00**	0.22
FCM	20	20	98.35	75.47	87.24	1.84
FCM	25	25	97.52	71.85	76.58	7.52
PCM	15	3	69.46	65.68	21.00	7.01
PCM	20	4	77.05	64.24	27.32	13.05
PCM	25	6	77.21	61.41	33.90	23.77
APCM ($\alpha = 1$)	15	15	99.28	97.81	97.20	0.7
APCM ($\alpha = 1$)	20	15	99.28	98.06	97.20	2.94
APCM ($\alpha = 1$)	25	**15**	**99.28**	**98.23**	**97.20**	8.21
SPCM	15	10	94.17	90.13	67.06	47.57
SPCM	20	13	96.22	71.09	79.34	313.9
SPCM	25	11	93.24	61.39	64.00	411.9
SAPCM ($\alpha = 1.5$)	15	**15**	**99.28**	**98.64**	**97.20**	9.07
SAPCM ($\alpha = 1$)	20	15	99.27	98.35	97.18	14.69
SAPCM ($\alpha = 1$)	25	15	99.27	98.51	97.16	20.87
SeqSAPCM ($\alpha = 1.2$)	-	**15**	**99.27**	**98.69**	**97.18**	95.90

Table 5.7: The results of the real Iris data set - Experiment 3

	m_{ini}	m_{final}	RM	GRM	SR	$Time$
k-means	3	**3**	**87.97**	-	**89.33**	0.1
k-means	5	5	78.41	-	54.67	0.1
k-means	10	10	77.32	-	42.00	0.1
FCM	3	**3**	**87.97**	**79.33**	**89.33**	0.01
FCM	5	5	78.85	68.32	55.33	0.01
FCM	10	10	77.02	63.76	38.00	0.02
PCM	3	2	77.19	77.10	66.67	0.05
PCM	5	2	77.19	76.88	66.67	0.06
PCM	10	2	77.63	77.55	66.67	0.07
APCM ($\alpha = 3$)	3	**3**	**91.24**	**90.01**	**92.67**	0.04
APCM ($\alpha = 1.5$)	5	3	83.68	82.58	84.00	0.05
APCM ($\alpha = 1$)	10	3	84.15	82.35	84.67	0.07
SPCM	3	2	77.63	77.22	66.67	0.31
SPCM	5	2	77.19	77.01	66.67	0.53
SPCM	10	2	77.63	77.59	66.67	1.26
SAPCM ($\alpha = 2.5$)	3	**3**	**91.24**	**90.54**	**92.67**	0.19
SAPCM ($\alpha = 1.2$)	5	3	83.68	82.64	84.00	0.35
SAPCM ($\alpha = 0.9$)	10	3	83.68	82.64	84.00	0.44
SeqSAPCM ($\alpha = 1.4$)	-	**3**	**83.68**	**82.64**	**84.00**	1.23

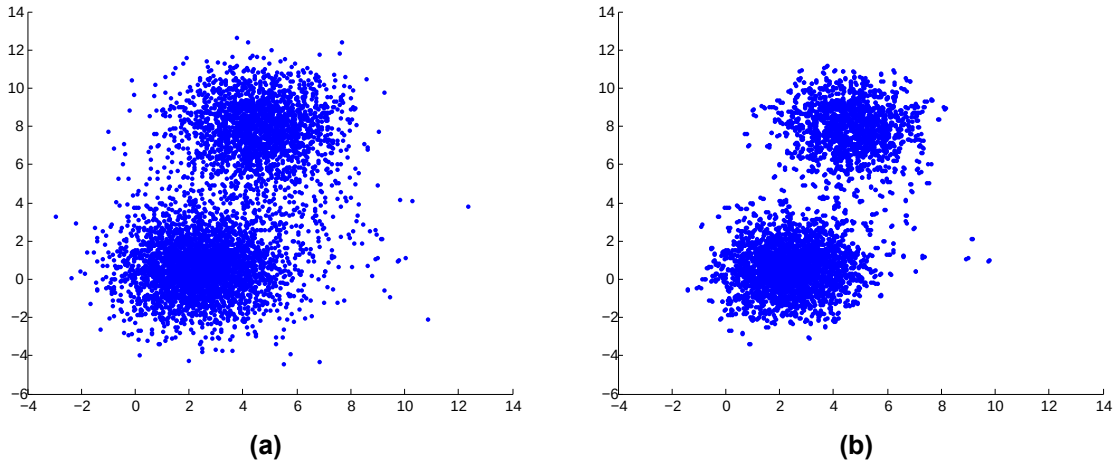


Figure 5.9: (a) Initial data, (b) Data after removing "noisy" points

element the SAPCM algorithm[11]. The two key features of SAPCM that L-SAPCM inherits, are the following: a) certain critical parameters are dynamically adapted, and b) sparsity is induced in the sense that each data point is forced to belong to only *a few* (or even *none*) of the clusters. These features make the algorithm capable in revealing the underlying clustering structure. Moreover, L-SAPCM as a layered algorithm has the ability to detect clusters that lie in *different resolutions* in the data space. Thus, L-SAPCM is an ideal choice, when the data set under study is composed of clusters with difference in their variance of several orders of magnitude. Such data sets are the HSIs in which the application of clustering becomes much more challenging, due to a) their high dimensionality and b) the tendency of HSI pixels to form not clearly distinguishable clusters. Actually, L-SAPCM has been initially designed to cope with HSI data, where the entities to be clustered are the pixels of the HSI under study. To this end, in the sequel, we present L-SAPCM using HSI terminology. Note that L-SAPCM contains some addtional processing steps (such as the data-purifying step and the PCA step) that facilitate the clustering procedure of the HSI pixels. Clearly, L-SAPCM can be adopted in any other application framework to perform clustering with those additional processing steps being deactivated depending on the will of the application expert.

5.5.1 The L-SAPCM algorithm

In HSIs, the number of image pixels, N, as well as the number of spectral bands, l, are usually very large. This increases dramatically both processing complexity and memory requirements. Taking into account, however, that contiguous HSI bands are usually highly correlated [84], computational complexity can be reduced by removing the redundancy introduced by the spectral information. To this end, we apply principal component analysis (PCA) as a first pre-processing step. The PCA transforms the original data so that almost

[11] A preliminary version of Layered SAPCM has been presented in [59]

Algorithm 8 [$X\_cleared$] = data_purifying(X)

Input: X

1: **Determine:** $d_{\min}(i) = \min_{\mathbf{x}_s \in X - \{\mathbf{x}_i\}} \|\mathbf{x}_i - \mathbf{x}_s\|^2$, $i = 1, ..., N$

2: **Compute:** $\mu = \frac{1}{N} \sum_{i=1}^{N} d_{min}(i)$

3: **Set:** $X\_cleared = \{\mathbf{x}_i \in X : d_{min}(i) < \mu, i = 1, ..., N\}$

4: **return** $X\_cleared$

all information is contained in the first few principal components, which are used from now on. As a result, the dimension l is dramatically reduced.

Another serious problem, frequently met in HSIs, is that the pixels are grouped to not very well distinguished "clouds". Thus, direct application of density-based clustering algorithms (such as SAPCM), could lead to poor clustering results. To face this problem, a pre-processing step, which removes the pixels that are not "too close" to the physical cluster centers, unravels the "cores" of the clusters, which are expected to be better distinguished. This can be achieved by first determining the mean of the distances of all pixels from their nearest neighbour and then removing those pixels whose distance from their nearest neighbour is larger than the mean (see Algorithm 8). As shown in Fig. 5.9, this pre-processing step allows clusters to be better distinguished, assisting density-based algorithms in unravelling the underlying clustering structure.

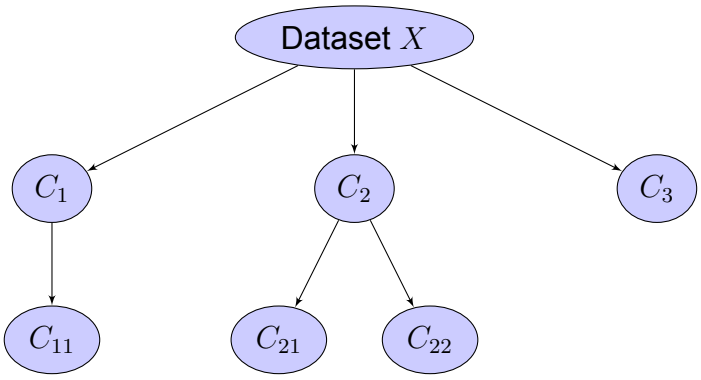


Figure 5.10: L-SAPCM flaw example

We describe now the proposed L-SAPCM algorithm, which is suitable for HSI clustering (see Algorithm 9). The algorithm first performs PCA on the data set and then executes the SAPCM algorithm in a layered form. Before each execution of SAPCM, $data\_purifying$ (Algorithm 8) is applied, as described above. Initially, SAPCM is applied on the whole data set producing some subsets that constitute the first layer clusters (leaf-layer clusters). Then, L-SAPCM uses the SAPCM algorithm (Algorithm 6) in each layer, in order to provide the clustering structure of the current layer, and then for each one of the leaf-level clusters, the SAPCM algorithm is applied again in a recursive manner. Note that L-SAPCM performs the noisy points removal pre-processing step for each leaf-layer clus-

ter before specifying the next leaf-level clusters through the SAPCM algorithm (see also Fig. 5.10). The algorithm terminates when the SAPCM algorithm returns one cluster for each leaf-layer cluster (which means that the currently processed leaf-layer cluster does not possess a clustering structure). Such a cluster is considered as a cluster of the final clustering structure provided by L-SAPCM[12].

The whole procedure of L-SAPCM algorithm is given in Algorithm 9.

Algorithm 9 [$clusters$] = L-SAPCM(X)

Input: X

1: [X]=PCA(X) and keep the l first components of PCA

2: $pending\_sets = \{X\}$

3: $clusters = \{\}$

4: **repeat**

5: **Take** an element C of $pending\_sets$

6: [C]=data_purifying(C)

7: **Give:** m_{ini}, α

8: [$\Theta, U, H, label, m$]=SAPCM(C, m_{ini}, α)

9: $C_j = \{\mathbf{x}_i \in X : label(i) = j, i = 1, ..., N\}$, $j = 1, ..., m$, where m is the final number of clusters that SAPCM returns when applied on C [13]

10: **if** $m > 1$ **then**

11: $pending\_sets = (pending\_sets - \{C\}) \cup \{C_1, ..., C_m\}$

12: **else if** $m = 1$ **then**

13: $pending\_sets = pending\_sets - \{C\}$

14: $clusters = clusters \cup \{C\}$

15: **end if**

16: **until** $pending\_sets = \emptyset$

---

[12] L-SAPCM can also be viewed as an *in-depth* processing algorithm.

[13] A data point $\mathbf{x}_i$ with $label(i) = 0$ (unassigned point) is assigned to its closest among the m formed clusters ($C_1, ..., C_m$).

17: **Assign** each $\mathbf{x}_i \in X$ that has been removed from the $data\_purifying$ scheme to its closest among $clusters$

18: **return** $clusters$

5.5.2 Experimental Results

In this subsection, we demonstrate the effectiveness of L-SAPCM to unravel clusters of various resolutions (clusters whose variances may differ considerably) via an experiment on synthetic data[14]. In order to evaluate the clustering performance of all algorithms, we use the Rand Measure (RM), the Generalized Rand Measure (GRM) and the classical Success Rate (SR). Finally, the time (in seconds) required for the convergence of each algorithm is provided.

Experiment: Let us consider a synthetic two-dimensional data set consisting of $N = 2900$ points and five clusters C_1, C_2, C_3, C_4, C_5. Each cluster is modelled by a normal distribution. The means of the distributions are $\boldsymbol{c}_1 = [20,\ 70]^T$, $\boldsymbol{c}_2 = [30,\ 80]^T$, $\boldsymbol{c}_3 = [40.3,\ 64]^T$, $\boldsymbol{c}_3 = [41,\ 65]^T$ and $\boldsymbol{c}_3 = [40.7,\ 65.6]^T$ respectively. Their covariance matrices are $10 \cdot I_2$, $10 \cdot I_2$, $0.01 \cdot I_2$, $0.02 \cdot I_2$ and $0.05 \cdot I_2$, respectively, where I_2 is the 2×2 identity matrix. A number of 2000 points is generated by the first distribution, 500 points by the second one, 100 points by the third one, 100 points by the fourth one and 200 points are generated by the fifth distribution. Note that the variance of clusters C_1 and C_2 is much larger than the variances of clusters C_3, C_4 and C_5 (see Fig. 5.11a). Figs. 5.11b, 5.11c show the clustering outcome obtained using the k-means and the FCM algorithm, respectively, both with $m_{ini} = 5$. Fig. 5.11d depicts the performance of PCM for $m_{ini} = 5$, while, in addition, it shows the circled regions, centered at each $\boldsymbol{\theta}_j$ and having radius equal to $\sqrt{\gamma_j}$, in which C_j has increased influence. Fig. 5.11e shows the results of APCM with $m_{ini} = 5$ and $\alpha = 1$, Fig. 5.11f shows the results of SPCM with $m_{ini} = 20$, Fig. 5.11g shows the results of SAPCM with $m_{ini} = 5$ and $\alpha = 1$ and Fig. 5.11h depicts the results of SeqSAPCM with $\alpha = 1$. Finally, Fig. 5.11i shows the results of L-SAPCM with $m_{ini} = 5$ and $\alpha = 1$ for each layer. Moreover, Table 5.8 shows the RM, GRM, SR and Time 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. 5.11 and Table 5.8, even when k-means and FCM are initialized with the true number of clusters ($m = 5$), they recognize clusters C_3, C_4 and C_5 as one, thus they fail to unravel the underlying clustering structure. The classical PCM also fails to distinguish clusters C_3, C_4 and C_5 from each other and, in addition, it misses cluster C_2, due to its proximity with cluster C_1, which is also much denser than it. As it can be seen, APCM, SPCM, SAPCM and SeqSAPCM are unable to distinguish C_3, C_4 and C_5 as three different clusters and thus they constantly produce a clustering result of only three clusters in total (see Figs. 5.11e-5.11h). On the other hand, L-SAPCM in the first layer identifies a three clustering structure (Fig. 5.11g), while in the second layer identifies

[14] Experimental results for L-SAPCM on HSI images are presented in chapter 7.

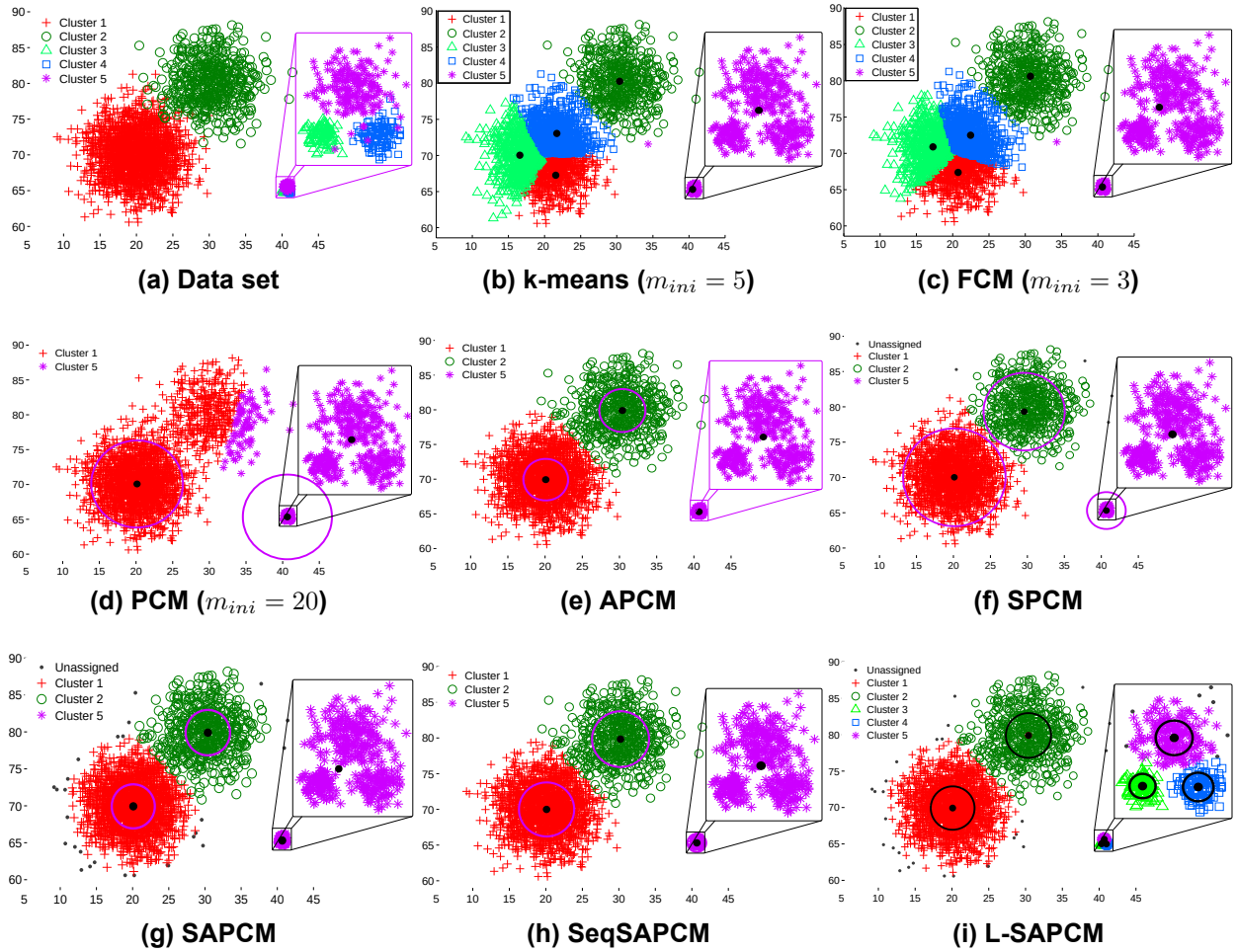


Figure 5.11: (a) The data set of the experiment (the area containing the three low variance clusters is magnified in a separate window). Clustering results for (b) k-means, $m_{ini} = 5$, (c) FCM, $m_{ini} = 5$, (d) PCM, $m_{ini} = 5$, (e) APCM, $m_{ini} = 5$, $\alpha = 1$, (f) SPCM, $m_{ini} = 20$, (g) SAPCM, $m_{ini} = 5$, $\alpha = 1$, (h) SeqSAPCM, $\alpha = 1$ and (i) L-SAPCM.

three different clusters within cluster C_5 of Fig. 5.11g (see Fig. 5.11i). Thus, L-SAPCM is the only algorithm that succeeds in unravelling accurately the clustering structure of this challenging experiment.

5.6 Conclusion

In this chapter a novel possibilistic c-means algorithm is proposed, termed SAPCM, which extends SPCM by embedding in it the adaptation mechanism of γ_j's as well as the cluster elimination mechanism from APCM. The algorithm is initialized through FCM with the latter executed for an overestimated number of the actual number of clusters. The SAPCM algorithm is immune to noise and outliers, as its predecessor SPCM. In addition, SAPCM has the ability (a) to cope well with closely located clusters with possibly different densi-

Table 5.8: The results of the experiment

	m_{ini}	m_{final}	RM	GRM	SR	$Time$
k-means	5	5	71.10	-	55.35	2.87
k-means	20	20	52.92	-	19.41	1.16
FCM	5	5	66.36	67.33	47.45	0.20
FCM	20	20	52.68	59.35	17.66	1.0
PCM	5	2	77.36	77.62	75.86	0.53
PCM	20	2	75.02	75.02	75.86	2.26
APCM ($\alpha = 1$)	5	3	97.22	96.97	92.17	0.41
APCM ($\alpha = 0.15$)	20	3	96.70	95.27	91.86	2.20
SPCM	5	3	94.86	92.43	90.72	4.56
SPCM	20	3	97.69	94.82	92.45	59.8
SAPCM ($\alpha = 1$)	5	3	97.28	97.01	92.21	2.45
SAPCM ($\alpha = 0.2$)	20	3	97.16	96.43	92.14	6.96
SeqSAPCM ($\alpha = 1$)	-	3	97.16	96.56	92.14	3.40
L-SAPCM ($\alpha = 1$)	5	**5**	**98.39**	**98.12**	**98.72**	2.51

ties and/or variances, (b) to determine the number of natural clusters and (c) to improve even more the estimates of the cluster representatives compared to SPCM and APCM. In extensive experiments, it is shown that SAPCM has a steadily superior performance, compared to other related algorithms, irrespective of the initial estimate of the number of clusters.

Moreover, two variants of SAPCM have been devised. The first one is an iterative bottom-up version, called SeqSAPCM, which, at each iteration, determines a single new cluster by employing SAPCM, and thus, unravels sequentially the underlying clustering structure in a bottom up fashion (from two clusters to the actual number of clusters existing in the data set). SeqSAPCM does not require knowledge of the number of clusters (not even a crude overestimate, as APCM, SPCM and SAPCM require). SeqSAPCM 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 two (initial estimate of the number of representatives and the parameter α) against only one in SeqSAPCM (α). The second variant of SAPCM, which is called L-SAPCM, works in layers. Specifically, it applies first SAPCM on the whole data set and the resulting clusters are considered as leafs forming the first layer of clusters. Then SAPCM is applied on the clusters corresponding to the leafs, giving rise to the second layer of leafs and so on. The algorithm terminates when no cluster in the last layers produces new clusters. The main advantage of the algorithm (as the experiment also verifies) is that, in principle, it can provide accurate clustering results even in cases where the data form clusters at various “resolutions”, i.e. the variances of the clusters may differ orders of magnitude from each other. This property makes L-SAPCM a good candidate for HSI processing, as also verified by the experimental results presented in chapter 7.

6. ONLINE ADAPTIVE POSSIBILISTIC C-MEANS ALGORITHM

6.1 Introduction

In this chapter, we present a novel online extension of the batch APCM clustering algorithm (chapter 3), called *Online Adaptive Possibilistic C-Means* (O-APCM). As is usually the case in all online algorithms, the data vectors in O-APCM are being processed one by one and their impact is memorized into suitably defined parameters; thus O-APCM is released from the noose of storing the whole data set and using it at each iteration, as is the case with the batch schemes. O-APCM achieves high quality clustering results much more computationally efficiently compared to batch methods. From this perspective, O-APCM is a serious candidate for processing big data sets (e.g. HSIs cubes).

The basic feature that O-APCM inherits from its ancestor is the adaptation of the involved parameters, which makes the algorithm flexible in tracking variations during the clustering formation. Furthermore, in contrast to the batch APCM, which starts with a crude overestimate of the number of clusters and gradually reduces it, O-APCM, due to its online nature, starts with zero clusters and gradually, as more data are processed sequentially, it alters it by (a) generating new clusters, (b) merging and (c) deleting existing clusters. These operations are carried out via associated novel mechanisms embedded in the algorithm.

Note that the usage of O-APCM is twofold. First, it can be used for clustering efficiently big static data, by dramatically reducing the required computational burden. Second, it is also appropriate for data clustering under non-stationary conditions; that is, in real time applications where the centers of the actual clusters may change their location over time within the data space. A prime example is the processing of a video stream with static background, aiming at tracking moving objects and monitoring their orbit.

It is worth noting that online clustering algorithms have already been presented in the bibliography, e.g. [85], [86], [87], albeit they have not been applied for HSI processing. However, all of them require knowledge of the exact number of physical clusters beforehand, which is kept fixed during their execution. A basic drawback, arising from this, is the fact that these online clustering algorithms are not able to alter (reduce or increase) the number of clusters that underlie in the data set over time. Moreover, they employ a random initialization of the cluster representatives, which, obviously, affects the final clustering result.

Closing this introduction, it is worth noting that the quality of the clustering results of O-APCM in stationary environments compares well with that of the previously introduced batch algorithms, while it outperforms its competitors under time-varying conditions.

6.2 The Online APCM (O-APCM)

In this section, following a unified formulation for both static and dynamic conditions, we describe in detail the proposed online APCM (O-APCM) clustering algorithm, which considers a single data vector per iteration and updates its parameters accordingly. To this end, the cost function (2.11) in the online context is modified as follows,

$$J(\Theta(t), U_t) = \sum_{i=1}^{t} \xi^{t-i} \sum_{j=1}^{m(t-1)} \Big[u_{ij} \|\mathbf{x}_i - \boldsymbol{\theta}_j(t)\|^2 + \gamma_j(t-1)\,(u_{ij} \ln u_{ij} - u_{ij}) \Big], \tag{6.1}$$

where t denotes the current iteration of the algorithm, $\mathbf{x}_t$ is the current data vector to be processed, $m(t-1)$ is the number of clusters formed up to the t-th iteration and ξ is an exponentially weighted factor, $0 \ll \xi \leq 1$. Moreover, $\mathbf{u}_t = [u_{t1}, ..., u_{tm(t-1)}]$ contains the degrees of compatibility of the current data vector $\mathbf{x}_t$ with the current number of clusters $m(t-1)$.

Let us focus on the parameter ξ appeared here for the first time. When $\xi = 1$, all data points have equally weighted contribution in the updating of the algorithm parameters. Thus, in this case, the algorithm is expected to behave well in static environments, in the sense that, in principle, it is able to capture the statistics of the clusters formed by the data vectors. On the other hand, when $\xi = 1$ the algorithm in not able to track the possible changes of the cluster statistics in a dynamically varying environment. However, if for the latter case a value less but very close to 1 is selected for ξ, the more recent data vectors will gain greater importance than the older ones in the parameters updating process. This will make the algorithm capable to track cluster variations in a dynamically changing environment[1].

6.2.1 Parameter initialization

Although O-APCM, as a descendent of APCM, inherits the main features of the latter, at the same time it is differentiated from it, due to its online nature. Specifically, the parameter initialization of O-APCM should take into account that no clusters are initially available due to the lack of data points at this stage. In addition, a cluster generation step should be adopted as clusters are formed dynamically during the execution of the algorithm. Also, a cluster merging step is required for merging two clusters growing in parallel that are part of a larger physical cluster. In the sequel, we describe in detail the above issues.

As it is easily understood, the initialization of the parameters $\boldsymbol{\theta}_j$ and η_j in O-APCM cannot be implemented as in batch APCM, due to the fact that the whole data set is not available a-priori. A natural way to proceed would be to create the first cluster containing only the data point $\mathbf{x}_1$ at the first iteration of the algorithm. However, it is obvious that it is not

[1]Note that the number of the most recent data points that are weighted more heavily and are actually taken into account (memory size) is approximately equal to $\lim_{N\to\infty} \frac{\xi^N - 1}{\xi - 1} = \frac{1}{1-\xi}$.

feasible to extract information about the "size" of this cluster from just a single data point, in order to initialize its parameter η_1. In order to address this issue, in the initialization phase of O-APCM, we run the batch APCM (see Algorithm 4) on a small sample of the, say K, first pixels (e.g. $K = 100$)[2], with an overestimated number of clusters, m_{ini}. After its convergence, APCM ends up with m ($\leq m_{ini}$) clusters and the obtained estimates for $\boldsymbol{\theta}_j$'s and η_j's, are used as the current state of O-APCM. Also, we set $\hat{\eta} = \min_j \eta_j$ (as in APCM) and then we run O-APCM for the remaining data points of the data set.

6.2.2 Parameter adaptation - Cluster generation

In O-APCM, this part refers to (a) the computation of the degree of compatibility u_{tj}, $j = 1, \ldots, m(t-1)$ of the current data point $\mathbf{x}_t$ with all clusters and the adaptation of all cluster representatives $\boldsymbol{\theta}_j$'s, $j = 1, \ldots, m(t-1)$, taking into consideration only the corresponding u_{tj}'s, and (b) the adaptation of the parameters η_r and $\boldsymbol{\mu}_r$ of the cluster C_r, with $u_{tr} = \max_{j=1,\ldots,m(t-1)} u_{tj}$, i.e. of the most compatible to $\mathbf{x}_t$ cluster.

Minimization of eq. (6.1) with respect to u_{tj} and $\boldsymbol{\theta}_j$, $j = 1, \ldots, m(t-1)$ results to

$$u_{tj} = \exp\left(-\frac{\|\mathbf{x}_t - \boldsymbol{\theta}_j(t-1)\|^2}{\gamma_j(t-1)}\right) \quad [3], \tag{6.2}$$

where (as in APCM)

$$\gamma_j(t-1) = \frac{\hat{\eta}}{\alpha}\eta_j(t-1) \tag{6.3}$$

and to the following time recursive equation for $\boldsymbol{\theta}_j(t)$,

$$\boldsymbol{\theta}_j(t) = \left(1 - \frac{u_{tj}}{U_j(t)}\right)\boldsymbol{\theta}_j(t-1) + \frac{u_{tj}}{U_j(t)}\mathbf{x}_t, \tag{6.4}$$

where the quantity $U_j(t)$ is defined recursively as follows

$$U_j(t) = \xi U_j(t-1) + u_{tj},\ j = 1, \ldots, m(t-1), \tag{6.5}$$

respectively. Note that *all* cluster representatives 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 η_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 far enough from the data points processed up to now so that to create a new cluster; only the parameters $\boldsymbol{\mu}_r$ and η_r of the most compatible to $\mathbf{x}_t$ cluster, C_r, are updated. Setting $I_{tj} = 0$, $j \neq r$ and $I_{tr} = 1$, we can easily get from (3.4) the

[2] Note that data points are processed in a random order.

[3] Note that direct optimization of eq. (6.1) with respect to u_{tj} involves in the right-hand side of eq. (6.2), $\boldsymbol{\theta}_j(t)$ instead of $\boldsymbol{\theta}_j(t-1)$. However, due to the interdependence of the updating equations eq. (6.2), eq. (6.4) for u_{tj} and $\boldsymbol{\theta}_j(t)$, respectively, we proceed with an alternating optimization concept and we use $\boldsymbol{\theta}_j(t-1)$ in eq. (6.2).

following time-recursive formulas:

$$\boldsymbol{\mu}_j(t) = \left(1 - \frac{I_{tj}}{S_j(t)}\right)\boldsymbol{\mu}_j(t-1) + \frac{I_{tj}}{S_j(t)}\mathbf{x}_t, \tag{6.6}$$

$$\eta_j(t) = \left(1 - \frac{I_{tj}}{S_j(t)}\right)\eta_j(t-1) + \frac{I_{tj}}{S_j(t)}\|\mathbf{x}_t - \boldsymbol{\mu}_j(t)\|, \tag{6.7}$$

where

$$S_j(t) = \xi S_j(t-1) + I_{tj}. \tag{6.8}$$

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 η is set to the minimum η among all current clusters, $\min_{j=1,\ldots,m(t-1)}\eta_j(t-1)$. Finally, the number of clusters is increased by one.

Before we close this subsection, let us comment on $U_j(t)$ and $S_j(t)$ defined above. Intuitively speaking, for $\xi = 1$, $U_j(t)$ accumulates the degrees of compatibility of the data points processed so far with cluster C_j, while $S_j(t)$ counts the number of points that are most compatible with C_j, among the points processed so far. On the other hand, for $\xi < 1$, the computation of $U_j(t)$, $S_j(t)$ is dominated by the most recent data points.

6.2.3 Cluster merging procedure

In online clustering schemes, the clustering result is usually dependent on the order in which the data are processed in the sense that several clusters might be created, which nevertheless represent parts of the same physical cluster. Consequently, 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-APCM considers all pairs of clusters C_s and C_k and checks whether they exhibit overlapping via the following rule: if $\sqrt{\gamma_s}+\sqrt{\gamma_k} > \beta \cdot d(\boldsymbol{\theta}_s, \boldsymbol{\theta}_k)$, where $d(\boldsymbol{\theta}_s, \boldsymbol{\theta}_k)$ is the Euclidean distance between the two cluster representatives $\boldsymbol{\theta}_s$, $\boldsymbol{\theta}_k$, (β controls the acceptable degree of overlapping and it is set equal to 1.1 in our case), the two clusters are merged into one cluster and the parameters of the newly formed cluster are defined as follows:

$$\boldsymbol{\theta}_{new} = \frac{U_s\boldsymbol{\theta}_s + U_k\boldsymbol{\theta}_k}{U_s + U_k}, \tag{6.9}$$

$$U_{new} = U_s + U_k, \tag{6.10}$$

$$\boldsymbol{\mu}_{new} = \frac{S_s\boldsymbol{\mu}_s + S_k\boldsymbol{\mu}_k}{S_s + S_k}, \tag{6.11}$$

$$\eta_{new} = \frac{S_s\eta_s + S_k\eta_k}{S_s + S_k}, \tag{6.12}$$

$$S_{new} = S_s + S_k. \tag{6.13}$$

In the case where the above criterion is not satisfied for any pair of clusters, no merging takes place. Note that the values for the parameters $\boldsymbol{\theta}_{new}$, $\boldsymbol{\mu}_{new}$, η_{new} of the newly formed cluster are computed as the weighted sum of the constituting clusters C_s and C_k.

Having analyzed the basic features of O-APCM separately, we give in Algorithm 10 below the whole O-APCM algorithm.

Algorithm 10 [Θ, Γ] = O-APCM(X, m_{ini}, α, ξ)

Input: X, m_{ini}, α, ξ

$K = 100$, $thres = 1e-05$ **and** $T = 100$

$t = K$

▷ *Initialization (use the first K points)*

$[\Theta(t), H(t), m(t), U(t)]$=APCM$(X(:, 1:K), m_{ini}, \alpha)$ [4]

Determine: $u_{ir} = \max_{j=1,...,m(t)} u_{ij}$ **and set: Label**$_i = r$, $i = 1, ..., K$

$\hat{\eta} = \min_{j=1,...,m(t)} \eta_j(t)$

Set: $\boldsymbol{\mu}_j(t) = \boldsymbol{\theta}_j(t)$, $j = 1, \ldots, m(t)$

Set: $U_j(t) = \sum_{i=1}^{K} u_{ij}$ **and** $S_j(t) = \sum_{\substack{i=1,\ldots,K: \\ \textbf{Label}_i = j}} 1$, $j = 1, \ldots, m(t)$

while there are still unprocessed data vectors **do**

  Set: $\gamma_j(t) = \frac{\hat{\eta}}{\alpha}\eta_j(t)$, $j = 1, \ldots, m(t)$

  $t = t + 1$

  ▷ *Take the next unprocessed data vector*

  $\mathbf{x}_t = X(:, t)$

  ▷ *Compute degrees of compatibility*

  $u_{tj} = \exp(-\frac{\|\mathbf{x}_t - \boldsymbol{\theta}_j(t-1)\|^2}{\gamma_j(t-1)})$, $j = 1, \ldots, m(t-1)$

  ▷ *Update cluster representatives*

  $U_j(t) = \xi U_j(t-1) + u_{tj}$, $j = 1, \ldots, m(t-1)$

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

  Determine: $u_{tr} = \max_{j=1,...,m(t-1)} u_{tj}$

[4] See Algorithm 4, section 3.3.

16: **if** $u_{tr} < thres$ **then**

▷ *Generate a new cluster*

17: $m(t) = m(t-1) + 1$

18: $\boldsymbol{\theta}_{m(t)}(t) = \mathbf{x}_t$

19: $\boldsymbol{\mu}_{m(t)}(t) = \mathbf{x}_t$

20: $\eta_{m(t)}(t) = \mathsf{min}_{j=1,\ldots,m(t-1)}\, \eta_j(t-1)$

21: $U_{m(t)}(t) = 1$ **and** $S_{m(t)}(t) = 1$

22: **Set:** $I_{tj} = 0$, $j = 1, \ldots, m(t)$

23: **Set:** $\textbf{Label}_t = m(t)$

24: **else**

▷ *Update parameters of cluster* C_r

25: $m(t) = m(t-1)$

26: **Set:** $I_{tj} = 0, \forall j \neq r$ **and** $I_{tr} = 1$

27: **Set:** $\textbf{Label}_t = r$

28: **end if**

29: $S_j(t) = \xi S_j(t-1) + I_{tj}$, $j = 1, \ldots, m(t-1)$

30: $\boldsymbol{\mu}_j(t) = \left(1 - \frac{I_{tj}}{S_j(t)}\right) \boldsymbol{\mu}_j(t-1) + \frac{I_{tj}}{S_j(t)} \mathbf{x}_t$, $j = 1, \ldots, m(t-1)$

31: $\eta_j(t) = \left(1 - \frac{I_{tj}}{S_j(t)}\right) \eta_j(t-1) + \frac{I_{tj}}{S_j(t)} \|\mathbf{x}_t - \boldsymbol{\mu}_j(t)\|$, $j = 1, \ldots, m(t-1)$

▷ *Every* T *iterations perform the* ***cluster merging procedure***

32: **end while**

33: **return** $\textbf{Label} = [\textbf{Label}_1, ..., \textbf{Label}_t]$

6.3 Experimental results

In this section, we assess the performance of O-APCM in both synthetic and real experiments under both static and dynamic conditions. Specifically, in a stationary environment, we compare the clustering performance of O-APCM with that of the batch APCM, SAPCM

and the online k-means (O-kmeans)[5] [86]. In a non-stationary environment, we compare the clustering performance of O-APCM with that of O-kmeans, which both are expected to be able to track the dynamical changes occurring in the physical clusters as they evolve. In order to measure the quality of clustering obtained by the algorithms, we use the Rand Measure (RM), the classical Success Rate (SR) and the runtime (Time) required for convergence (in seconds).

6.3.1 Experiments in a stationary environment

Experiment 1: Let us consider a synthetic two-dimensional data set consisting of $N = 31200$ points, where three clusters C_1, C_2, C_3 are formed (Fig. 6.1(a)). Each cluster is modelled by a normal distribution. The means of the distributions are $\boldsymbol{c}_1 = [7, 7]^T$, $\boldsymbol{c}_2 = [0, 0]^T$ and $\boldsymbol{c}_3 = [11, 0]^T$, respectively, while the (common) covariance matrix of C_1 and C_2 is set to $4 \cdot I_2$, and the covariance matrix of C_3 is set to I_2, where I_2 is the 2×2 identity matrix. A number of 1000 points are generated from the first distribution, 30000 points are generated from the second one and 200 points are generated from the third one. Note that clusters C_1 and C_2 differ significantly in their density (since both share the same covariance matrix but C_1 has significantly less points than C_2) and due to their close proximity, a clustering algorithm could hardly distinguish them. In addition, cluster C_3 consists very few data points, thus it is hardly identified, too.

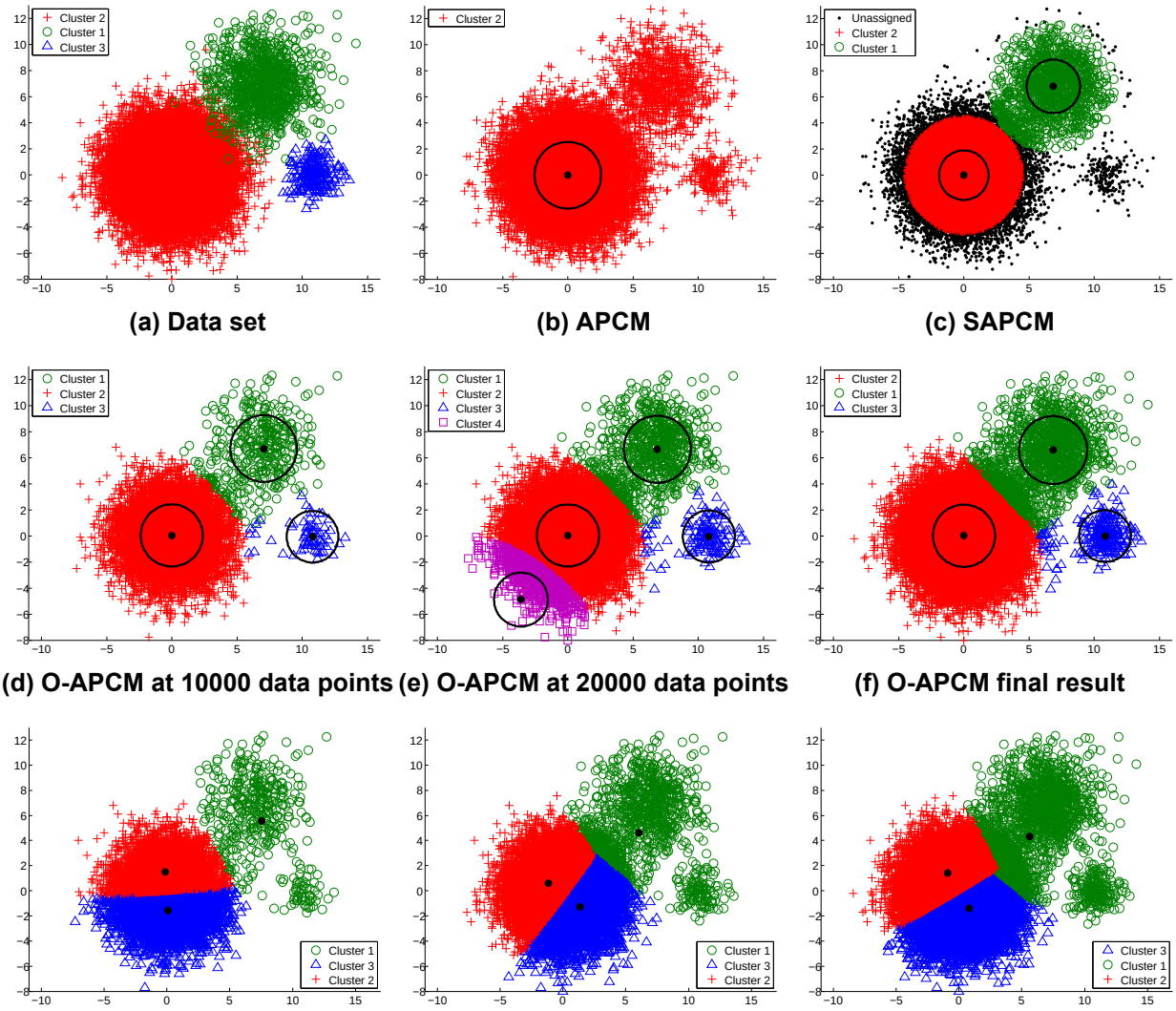
Fig. 6.1(b) shows the clustering outcome obtained using the APCM algorithm with $\alpha = 0.7$ and $m_{ini} = 10$. Similarly, in Fig. 6.1(c) we present the corresponding results for SAPCM with $\alpha = 1$ and $m_{ini} = 10$. Figs. 6.1(d)-(f) depict the performance of O-APCM ($\alpha = 0.7$, $\xi = 1$) after the processing of the first 10000, 20000 and 31200 (final stage) data points, respectively. The corresponding results of O-kmeans ($m = 3$, $z = 0.01$) at the same stages are shown in Figs. 6.1(g)-(i). Moreover, Table 6.1 shows RM, SR and Time (in seconds) for the previously mentioned algorithms, where m_{ini} and m_{final} denote the initial and the final number of clusters, respectively.

Table 6.1: Performance of clustering algorithms for the synthetic data set of Experiment 1 (stationary environment).

	m_{ini}	m_{final}	RM	SR	Time
APCM ($\alpha = 0.7$)	10	1	92.56	96.15	11.90
SAPCM ($\alpha = 1$)	10	2	97.85	98.29	52.78
O-APCM ($\alpha = 0.7$, $\xi = 1$)	5	**3**	97.49	**98.69**	**0.39**
O-kmeans ($z = 0.01$)	3	3	50.33	50.02	0.47

As it is deduced from Fig. 6.1 and Table 6.1, APCM fails to unravel the underlying clustering structure, due to the great difference on the density of the clusters and their close proximity. Although SAPCM manages to distinguish cluster C_1 from C_2, it fails to uncover cluster C_3.

[5] O-kmeans is basically a set of m gradient descent-like schemes, one for each cluster representative, sharing the same learning rate parameter z. Upon the arrival of a new data vector, only the closest to it representative is updated.



(a) Data set **(b) APCM** **(c) SAPCM**

(d) O-APCM at 10000 data points **(e) O-APCM at 20000 data points** **(f) O-APCM final result**

(g) O-kmeans ($m = 3$) at 10000 data points **(h) O-kmeans ($m = 3$) at 20000 data points** **(i) O-kmeans ($m = 3$) final result**

Figure 6.1: (a) The data set of Experiment 1 (stationary environment). Clustering results for (b) APCM, $m_{ini} = 10$, $\alpha = 0.7$, (c) SAPCM, $m_{ini} = 10$, $\alpha = 1$, (d)-(f) O-APCM at 10000, 20000 and 31200 (final) data points, respectively, $\alpha = 0.7$, $\xi = 1$, (g)-(i) O-kmeans at 10000, 20000 and 31200 (final) data points, respectively, $m = 3$, $z = 0.01$.

As it is seen from Figs.6.1(g)-(i), O-kmeans is initialized with the true number of clusters ($m = 3$), however, its clustering performance is not as satisfactory as expected, due to the peculiar data set structure. On the other hand, O-APCM is able to detect all clusters, producing very satisfactory results with high computational efficiency, requiring less time even than O-kmeans.

6.3.2 Experiments in a non-stationary environment

Experiment 2: Let us consider a synthetic two-dimensional data set consisting of $N = 20800$ points, where five clusters C_1, C_2, C_3, C_4, C_5 are dynamically formed (Fig. 6.2). Each cluster is modelled by a normal distribution. The means of the distributions are dynamically changed over time, starting from $\boldsymbol{c}_1 = [100, 50]^T$, $\boldsymbol{c}_2 = [20, 90]^T$, $\boldsymbol{c}_3 = [20, 0]^T$, $\boldsymbol{c}_4 = [30, 20]^T$ and $\boldsymbol{c}_5 = [100, 50]^T$ and ending at $\boldsymbol{c}_1 = [50, 80]^T$, $\boldsymbol{c}_2 = [0, 40]^T$, $\boldsymbol{c}_3 = [0, 40]^T$, $\boldsymbol{c}_4 = [80, 10]^T$ and $\boldsymbol{c}_5 = [30, 40]^T$, respectively, moving on straight lines, as shown in Fig. 6.2 (see also Table 6.2). Their (common) covariance matrix is set to $10 \cdot I_2$. A number of 3200 points are generated from each of the first and the third distributions, 4000 points are generated from the fifth one and 5200 points are generated from each of the second and the fourth ones. Note that clusters C_1 and C_5 start generating data points around the same data point, however, as time passes by, their centers move towards different positions, thus C_1 and C_5 get split. The opposite holds for clusters C_2 and C_3, which start from different locations and eventually converge to the same position, thus they are merged.

Table 6.2: Data set of Experiment 2 (non stationary environment).

Cluster	Start point ($\mathbf{c}_j$)	End point ($\mathbf{c}'_j$)	Number of total steps	Number of points per step	Total number of points
C_1	$[100, 50]^T$	$[50, 80]^T$	400	8	3200
C_2	$[20, 90]^T$	$[0, 40]^T$	400	13	5200
C_3	$[20, 0]^T$	$[0, 40]^T$	400	8	3200
C_4	$[30, 20]^T$	$[80, 10]^T$	400	13	5200
C_5	$[100, 50]^T$	$[30, 40]^T$	400	10	4000

Subfigures of the first column of Fig. 6.3 show the clustering outcome obtained by O-kmeans with $m = 4$ and $z = 0.01$ at several time instants. The corresponding clustering results at the same time instants for O-kmeans with $m = 5$ and $z = 0.01$ are shown in the subfigures of the second column of Fig. 6.3. Finally, the last column of Fig. 6.3 contains the subfigures that depict the corresponding clustering result of O-APCM with $\alpha = 0.8$ and $\xi = 0.99$ for the same time instants. Moreover, Table 6.3 shows RM, SR and Time (in seconds) for both algorithms, where m_{ini} and m_{final} denote the initial and the final number of clusters, respectively.

As it is deduced from Fig. 6.3 and Table 6.3, O-kmeans with $m = 4$ initially behaves well as long as clusters C_1 and C_5 are still united and the number of underlying clusters is 4 (Figs.6.3(a),(d)). However, when clusters C_1, C_5 are split into two different clusters, O-kmeans fails to follow each one of them (Figs.6.3(j),(m),(p),(s)); thus, O-kmeans concludes to a four-clusters clustering result, completely failing to detect the center of clusters C_1 and C_5 (Fig.6.3(v)). Additionally, O-kmeans (with $m = 4$) is not able to merge C_2, C_3, when the latter are eventually merged (Fig. 6.3(v)). On the other hand, O-kmeans with $m = 5$ in the early stages of clustering with the number of clusters being 4, erroneously separates cluster C_1 into two clusters (Figs.6.3(b),(e)). However, in the sequel where the underlying clusters are 5, O-kmeans ($m = 5$) detects very accurately the clustering structure of the data set (Figs.6.3(h),(k),(n),(q),(t)). Again, however, O-kmeans (with $m = 5$) fails to merge C_2, C_3, when the latter become almost coincident (Fig. 6.3(w)). Finally, O-APCM produces

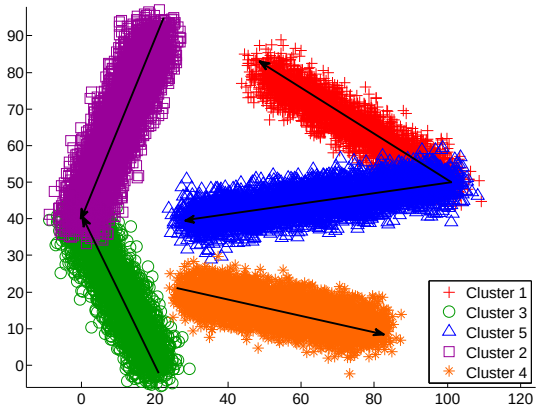


Figure 6.2: Data set of Experiment 2 (non stationary environment).

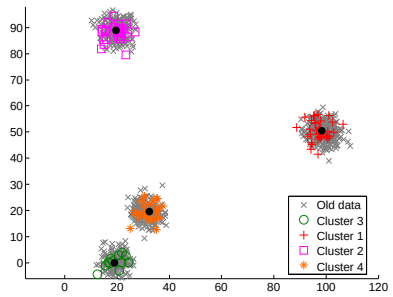
very accurate results at each time instant, as the centers of the clusters are dynamically changed, and it has the ability to delete or create clusters on demand, thus tracking with high accuracy and computational efficiency the clustering structure of the data set over time (3rd column of Fig.6.3).

Table 6.3: Performance of clustering algorithms for the synthetic data set of Experiment 2 (non-stationary environment).

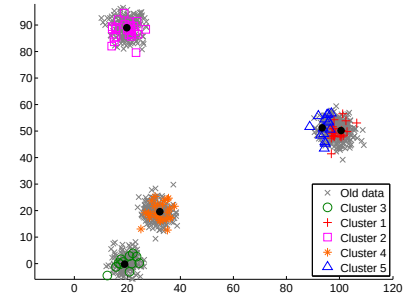
	m_{ini}	m_{final}	RM	SR	Time
O-kmeans ($z = 0.01$)	4	4	93.20	84.96	0.30
O-kmeans ($z = 0.01$)	5	5	96.64	94.93	0.31
O-APCM ($\alpha = 0.8$, $\xi = 0.99$)	5	4	**97.55**	**96.27**	**0.28**

In the next experiment, we test O-APCM on a real application, where the data set under study is a series of successive images in the scale of time (video). The aim here is the identification of moving objects and the tracking of their orbit in a video data sequence produced by a static camera (fixed background). As it is expected in such a video, the number of objects may change, as the objects enter into or leave the vision range of the camera used. Consequently, O-kmeans is not examined in this experiment, due to the fact that O-kmeans needs a-priori knowledge of the number of clusters, m, which is not a realistic requirement for video data sets. Besides, in general, m varies over time.

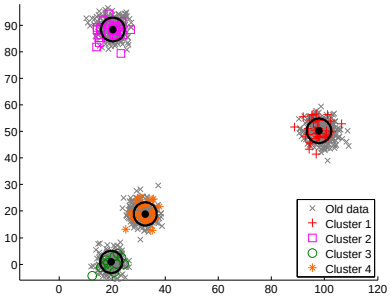
Experiment 3: Let us consider the real *Basketball* RGB video data set [88]. The video has a total duration of 4.64 seconds and it consists of $K = 111$ time frames, each one of duration of 0.0418 seconds. The spatial resolution of each image frame is 1280$\times$720, thus each frame has $N_k = 921600$ pixels, $k = 1, \ldots, 111$. The total number of data points (pixels) of the whole data set (video) is $N = 102297600$. Note that for each pixel we have three values of intensity (RGB images). In this video, four moving objects are identified in total. Specifically, a walking man in the image background (C_1), a man initially holding a basketball that subsequently throws away (C_2), the basketball by the time instant it moves away from the man (C_3) and a tent that is moving because of the wind on the left side of the vision range of the camera (C_4). Note that the aforementioned basketball disappears at some point from the visible field of the camera and then re-enters straightaway.



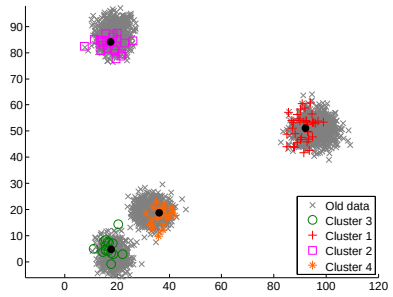
(a) O-kmeans ($m = 4$) at 1000 data points



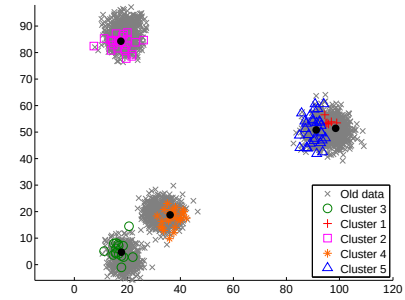
(b) O-kmeans ($m = 5$) at 1000 data points



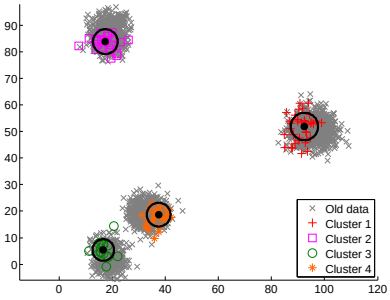
(c) O-APCM at 1000 data points



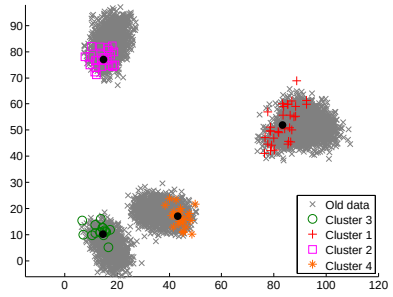
(d) O-kmeans ($m = 4$) at 3000 data points



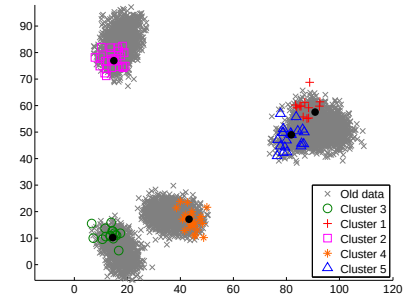
(e) O-kmeans ($m = 5$) at 3000 data points



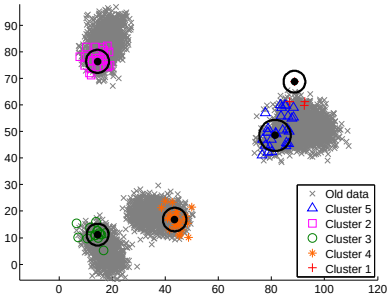
(f) O-APCM at 3000 data points



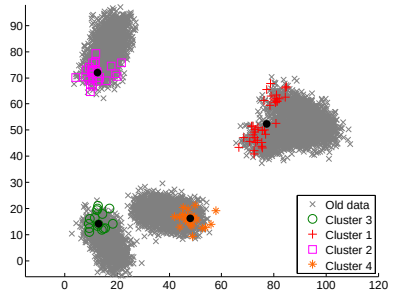
(g) O-kmeans ($m = 4$) at 6000 data points



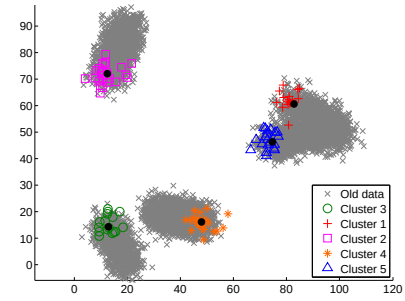
(h) O-kmeans ($m = 5$) at 6000 data points



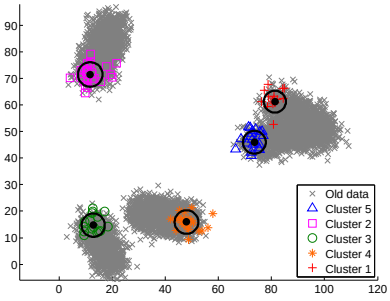
(i) O-APCM at 6000 data points



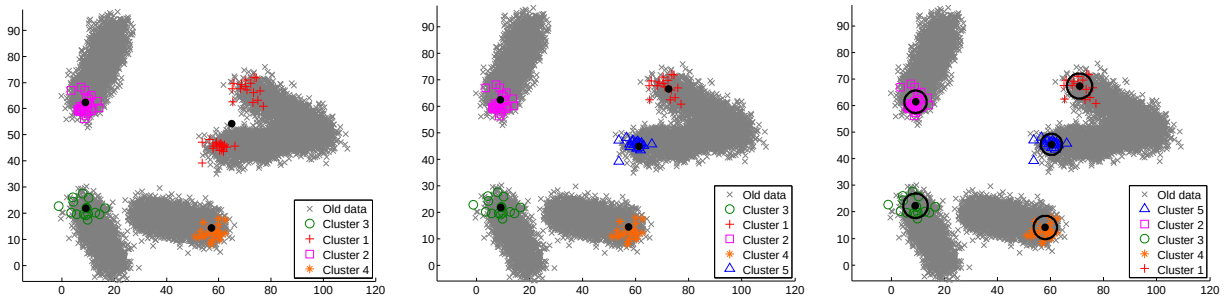
(j) O-kmeans ($m = 4$) at 8000 data points



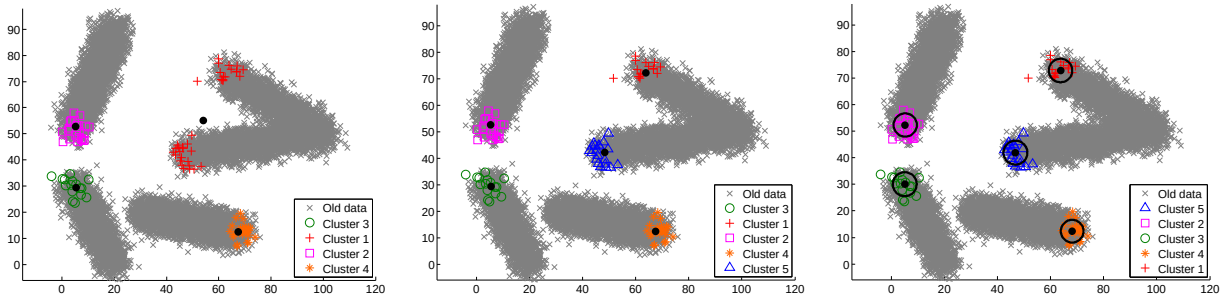
(k) O-kmeans ($m = 5$) at 8000 data points



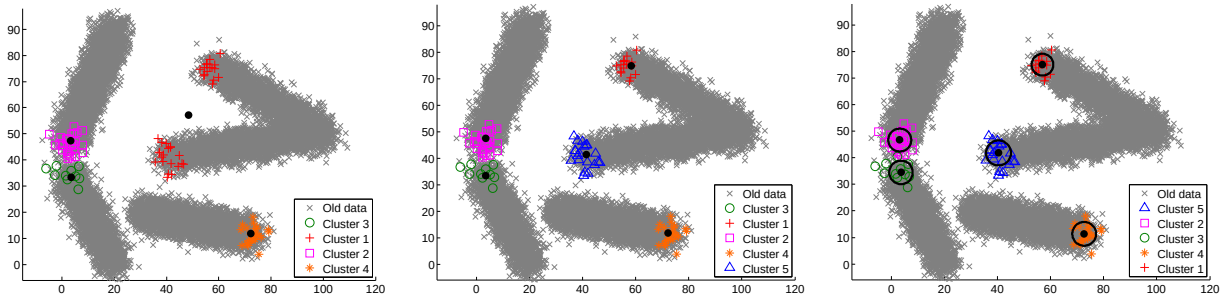
(l) O-APCM at 8000 data points



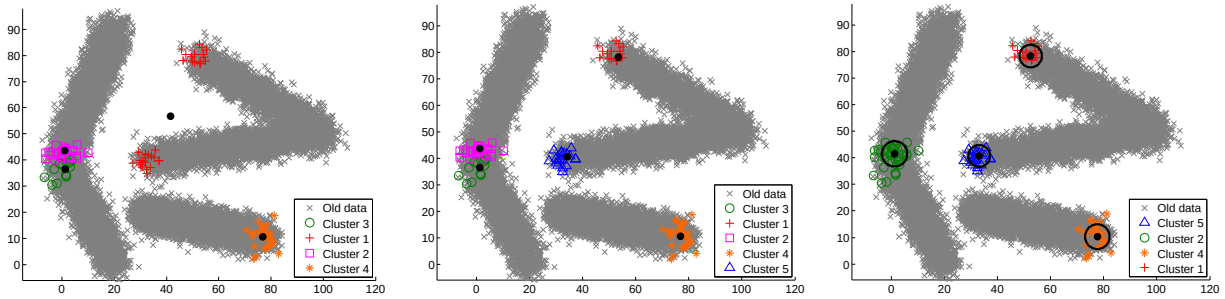
(m) O-kmeans ($m = 4$) at 12000 data points **(n) O-kmeans ($m = 5$) at 12000 data points** **(o) O-APCM at 12000 data points**



(p) O-kmeans ($m = 4$) at 16000 data points **(q) O-kmeans ($m = 5$) at 16000 data points** **(r) O-APCM at 16000 data points**



(s) O-kmeans ($m = 4$) at 18000 data points **(t) O-kmeans ($m = 5$) at 18000 data points** **(u) O-APCM at 18000 data points**



(v) O-kmeans ($m = 4$) at 20000 data points **(w) O-kmeans ($m = 5$) at 20000 data points** **(x) O-APCM at 20000 data points**

Figure 6.3: Clustering results of Experiment 2 (non-stationary environment) at 1000, 3000, 6000, 8000, 12000, 16000, 18000 and 20000 data points, respectively, for (a), (d), (g), (j), (m), (p), (s), (v) O-kmeans, $m = 4$, $z = 0.01$, (b), (e), (h), (k), (n), (q), (t), (w) O-kmeans, $m = 5$, $z = 0.01$ and (c), (f), (i), (l), (o), (r), (u), (x) O-APCM, $\alpha = 0.8$, $\xi = 0.99$ (the older points of each cluster are gray-colored).

(a) (b) (c)

Figure 6.4: (a) Differences between frames 24 and 25 (matrix I_{24} depicted as an image), (b) smoothed differences between frames 24 and 25 (image $I_{24}^{Smoothed}$) and (c) pixels with high (above pre-defined threshold) differences between frames 24 and 25.

The whole process for motion tracking is divided into two stages. At the first stage, the aim is to identify the pixels of the moving objects, examining the differences between each pair of successive frames. At the second stage, these pixels are processed by O-APCM, in order to identify each moving object as a cluster in the spatial domain and to track its orbit. Concerning the first stage, we aim to identify all the pixels that compose the moving objects for each frame. To this end, we first compute the pixel-by-pixel Euclidean distance matrix I_k, $k = 1, \ldots, K-1$ between each pair of successive frames (say frames k, $k+1$) taking into account the RGB values of the pixels (Fig. 6.4(a)). In the sequel, we produce the smoothed image of matrix I_k, $I_k^{Smoothed}$, by applying a 10$\times$10 mean filter, in order to reduce noise impact and smooth I_k (Fig. 6.4(b)). Finally, from image $I_k^{Smoothed}$, we keep only the pixels whose $I_k^{Smoothed}$ value is larger than a pre-defined threshold, th (Fig. 6.4(c)). The coordinates (x, y) of all these pixels constitute the data set that will be processed by O-APCM at the second stage. Closing, we note that the above threshold, th, is determined by the point where a significant "knee" is met, at the histogram of the positive values of the vectorized matrix $I_k^{Smoothed}$. In practice, this needs to take place only for one of the matrices $I_k^{Smoothed}$ (e.g. the first one, $k = 1$).

Concerning the second stage, the aim is to run O-APCM for the two dimensional (two coordinates for each pixel) data set resulting from the first stage. The subfigures of the first column of Fig. 6.5 show several frames of the data set under study, while the subfigures of its second column depict the corresponding clustering outcome obtained by O-APCM. Note that in this experiment, all pixels of the same frame, produced by the first processing stage (object detection), are equally weighted, while, in addition, these are points that will be more influential in the algorithm parameters adjustment, compared to pixels of previous frames. The latter requirement is achieved by setting $\xi = 0.05$ [6]. Finally, we set parameter α equal to 0.05.

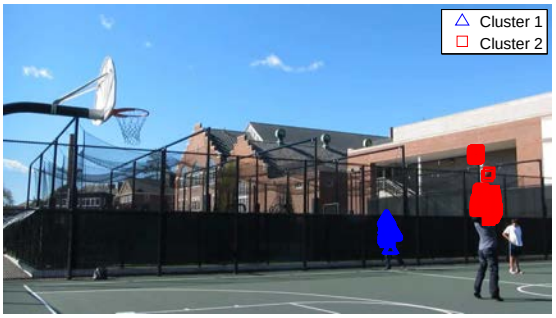
As it is deduced from Figs. 6.5(b),(d), during the first frames O-APCM identifies correctly the two clusters C_1 (the man at the background of the image) and C_2 (the man that holds the basketball). Around frames 19-20, the man labeled as cluster C_2 throws away the

[6]Note that $\lim_{N\to\infty} \frac{\xi^N - 1}{\xi - 1} = \frac{1}{1-\xi} \simeq 1$, for $\xi = 0.05$. That is, O-APCM processes the data, aiming at uniformly weighting pixels of the same frame and weighting them more heavily, compared to pixels of previous frames.

(a) Data set at frame 17

(b) O-APCM result at frame 17

(c) Data set at frame 19



(d) O-APCM result at frame 19

(e) Data set at frame 24

(f) O-APCM result at frame 24

(g) Data set at frame 32

(h) O-APCM result at frame 32

(i) Data set at frame 36

(j) O-APCM result at frame 36

(k) Data set at frame 53

(l) O-APCM result at frame 53

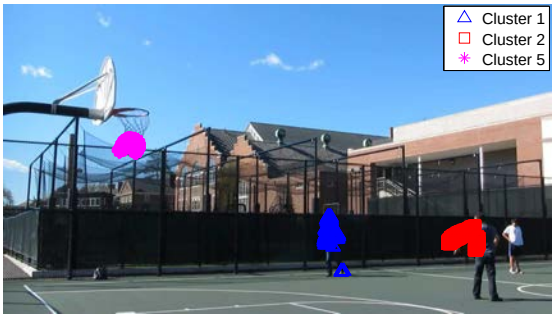
(m) Data set at frame 56

(n) O-APCM result at frame 56

(o) Data set at frame 59

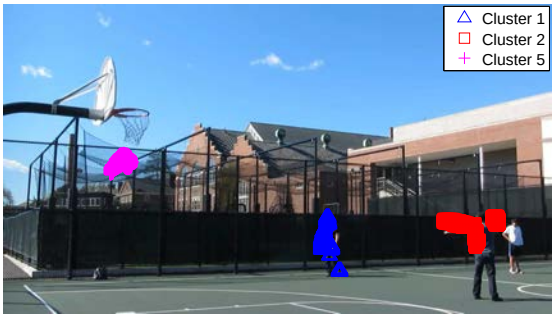
(p) O-APCM result at frame 59

(q) Data set at frame 70



(r) O-APCM result at frame 70

(s) Data set at frame 72



(t) O-APCM result at frame 72

(u) Data set at frame 76

(v) O-APCM result at frame 76

(w) Data set at frame 81

(x) O-APCM result at frame 81

Figure 6.5: (a), (c), (e), (g), (i), (k), (m), (o), (q), (s), (u), (w) Data set of Experiment 3 (non-stationary environment, video) at frame 17, 19, 24, 32, 36, 53, 56, 59, 70, 72, 76 and 81, respectively, and (b), (d), (f), (h), (j), (l), (n), (p), (r), (t), (v), (x) the respective clustering results of O-APCM, $\alpha = 0.05$, $\xi = 0.05$.

basketball (Fig. 6.5(d)), thus a new cluster C_3 (the basketball) is created by O-APCM (Fig. 6.5(f)). Around frames 32-33, the basketball (C_3) disappears from the visible range of the camera (Fig. 6.5(h)) and it re-enters around frame 51 (Figs. 6.5(j),(l)). As it was expected, O-APCM considers the returned basketball as a new cluster labeled as C_5 (see Figs. 6.5(l) and the following). Note also, that from around frame 36, O-APCM manages to identify with high accuracy the moving tent (cluster C_4) at the bottom left side of the image and to track its full orbit until around frames 56-57 (see Figs. 6.5(j),(l),(n)). However, around frames 57-58, the tent stops moving; thus, O-APCM deletes correctly cluster C_4 (Fig. 6.5(p)), recognizing that it is not a moving object any more. Concluding, note from all subfigures of the second column of Fig. 6.5 that O-APCM identifies all moving objects and manages with high accuracy to track their orbits.

6.4 Conclusion

In this chapter, a novel online adaptive possibilistic c-means clustering algorithm, called O-APCM, which is an online implementation of the recently proposed APCM algorithm (chapter 3), is presented. O-APCM incorporates the relative parameter adaptation mechanism of APCM, which makes it capable to deal well with closely located and hardly distinguished clusters. Moreover, O-APCM embodies three new procedures for generating new clusters, merging or deleting dynamically existing ones, when necessary, as data points are being processed one by one. In addition, O-APCM does not need a prior knowledge of the underlying number of physical clusters. Due to its online nature, O-APCM requires very low memory and computational time, compared to a batch mode implementation, where the whole data set is considered at each iteration of the algorithm. This makes O-APCM applicable for clustering of big data sets, whose size and dimensionality are prohibitive for batch processing. Moreover, apart static conditions, O-APCM is recommended for cases where the dynamics of the data set change with time, i.e. the centers of the physical clusters change their location in the data space over time. Specifically, O-APCM has the ability to weight more heavily the most recent data, compared to older data, in the estimation of its parameters. Experimental results show that O-APCM offers high discrimination ability at a very low computational cost for data sets in stationary conditions and, additionally, it is able to track with high accuracy the physical clusters at a non-stationary environment. Finally, the application of O-APCM to a real video data set, in order to identify and track moving objects, highlights its great potential in monitoring the evolution of dynamically varying phenomena.

7. CLUSTERING ALGORITHMS APPLIED TO HYPERSPECTRAL IMAGES: A COMPARATIVE STUDY

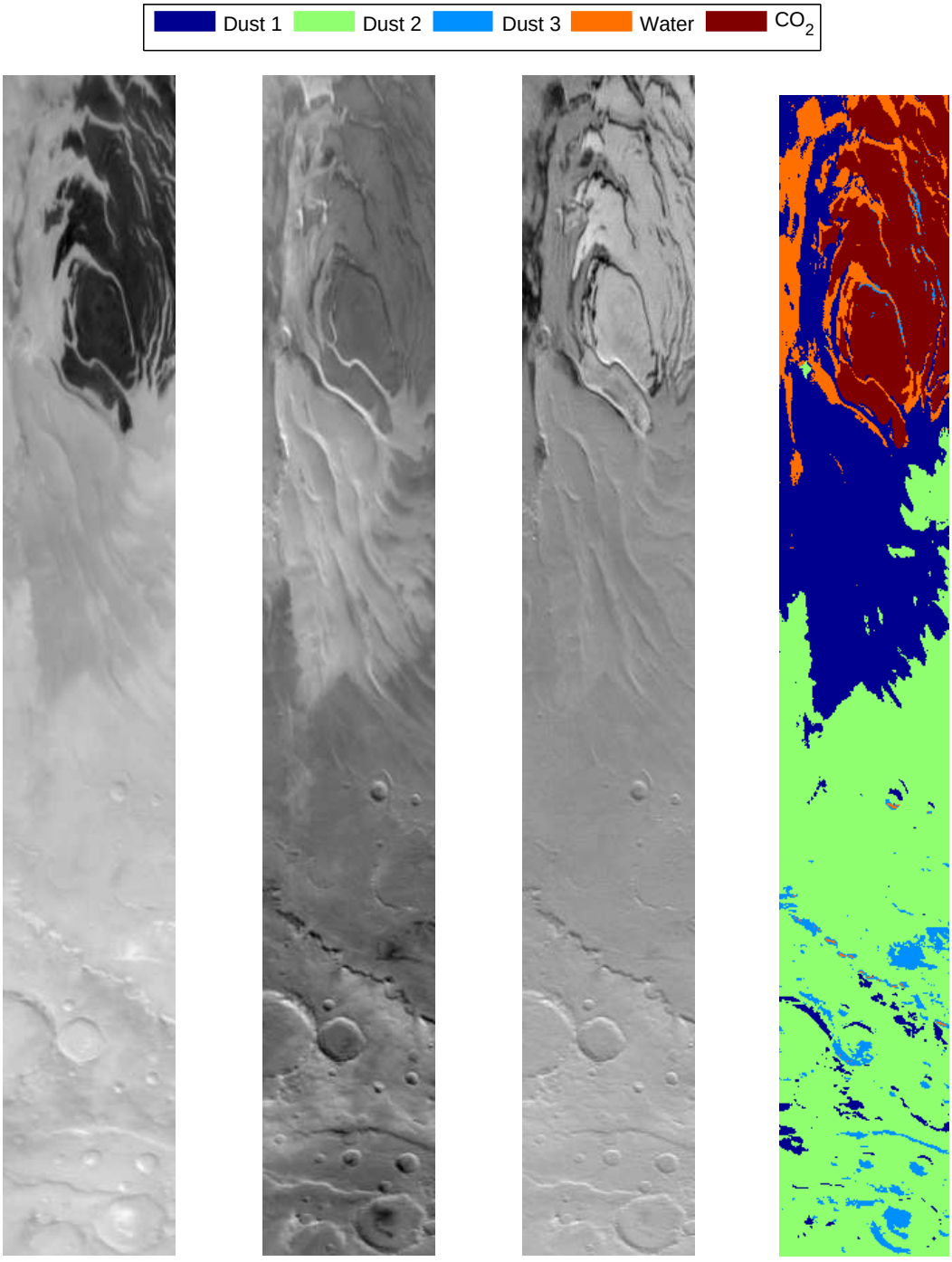
7.1 Introduction

In the present chapter, we assess the performance of the clustering algorithms proposed in this thesis on real HSI data sets. Specifically, we evaluate the performance of APCM, SPCM, SAPCM, L-SAPCM and O-APCM in comparison with several other related algorithms, including k-means [6], FCM [7], [8], PCM [10], UPC [20], UPFC [28], PFCM [19], H2NMF [51], KNNCLUST [45] and O-kmeans [86]. The clustering algorithms are applied to three HSIs, collected from different hyperspectral sensors, depicting: (a) a scene of Mars' South Polar Cap (OMEGA), (b) a scene of Salinas Valley, California (AVIRIS) [89] and (c) a scene of Washington DC Mall (HYDICE) [90]. These HSIs have been selected on purpose since they depict three totally different environments: planetary, agricultural and urban, respectively, in order to evaluate the performance of the algorithms in diverse cases.

For fair comparison, the initial representatives $\boldsymbol{\theta}_j$'s of all algorithms are initialized based on the FCM scheme and the parameters of each algorithm are first fine-tuned. Moreover, we remove duplicate clusters after the termination of algorithms in which coincident clusters may arise (PCM, UPC, UPFC, PFCM, SPCM). In order to compare a clustering with the true data label information (if existed), we utilize again the RM and SR indices defined in subsection 3.7.2. Finally, the number of iterations (Iter) and the total time (Time (in seconds)) required for the convergence of each algorithm, are provided. All algorithms have been executed using MATLAB R2013a on Intel i7-4790 machine (16 GB RAM, 3.60 GHz).

7.2 Case Study 1: South Polar Cap

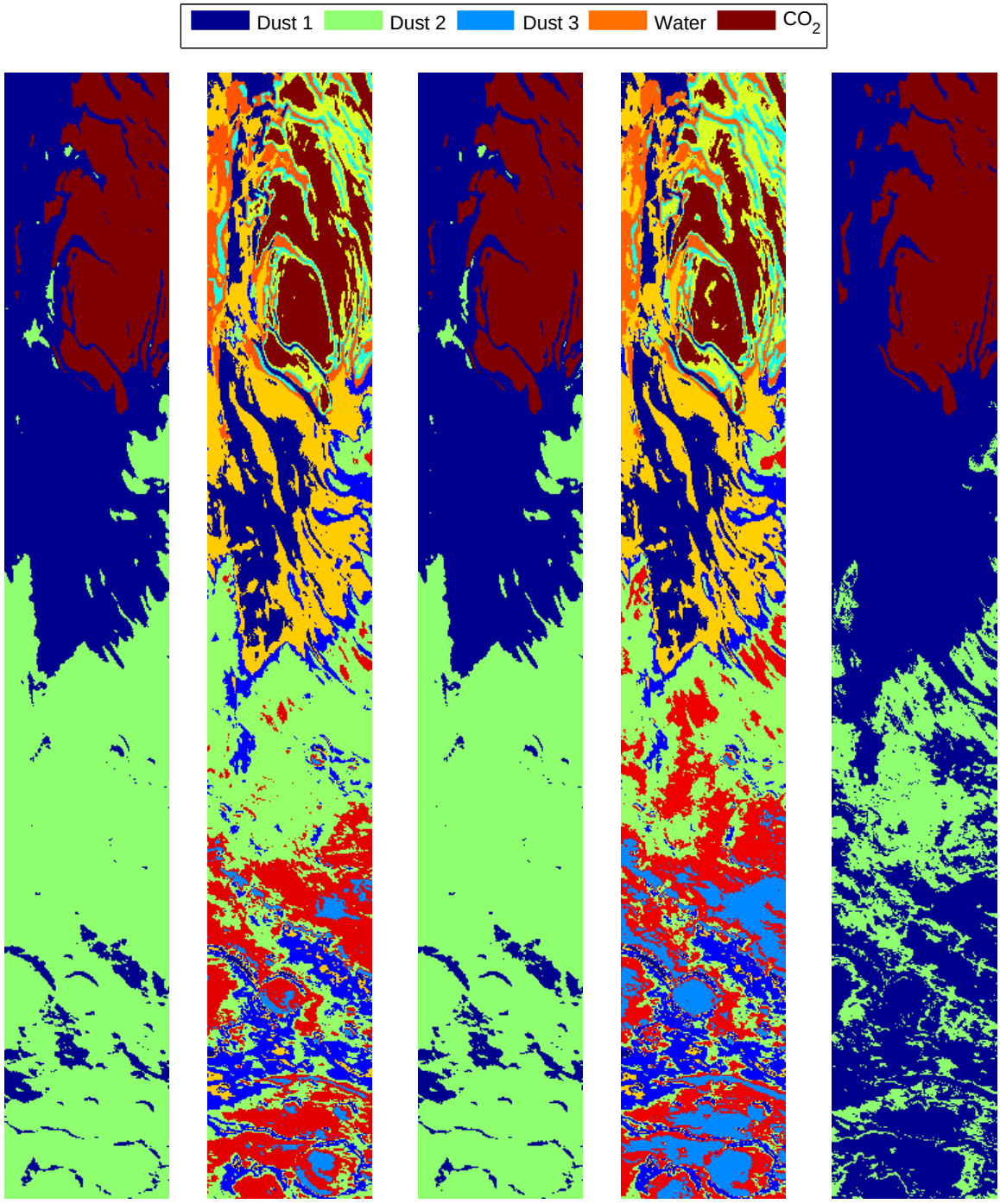
This data set consists of a hyperspectral data cube depicting the South Polar Cap of Mars in the local summer (Jan. 2004), acquired by the OMEGA sensor, which generates 128 spectral bands across the range from 0.93 to 2.73μm with a spectral resolution of 14nm and 128 spectral bands from 2.55 to 5.11μm with a spectral resolution of 21nm. The total number of the 256 bands is reduced to 186 in the region from 0.93 to 2.98μm, after the removal of the noisy bands, with a spatial resolution of approximately 3km. The size of the HSI is 871$\times$128, which results to a total size of $N = 111488$ samples-pixels. Note that there is no available ground truth information for this data set. To this end, in Figs. 7.1(a)-(c) the 1st, 2nd and 3rd principal components (PCs) of the data set are depicted, where the main features of the data set are visible. Moreover, in Fig. 7.1(d) a "virtual" reference map is composed, based on the first three PCs and [91], which is also verified by [92],



(a) 1st PC **(b) 2nd PC** **(c) 3rd PC** **(d) "Virtual" reference map**

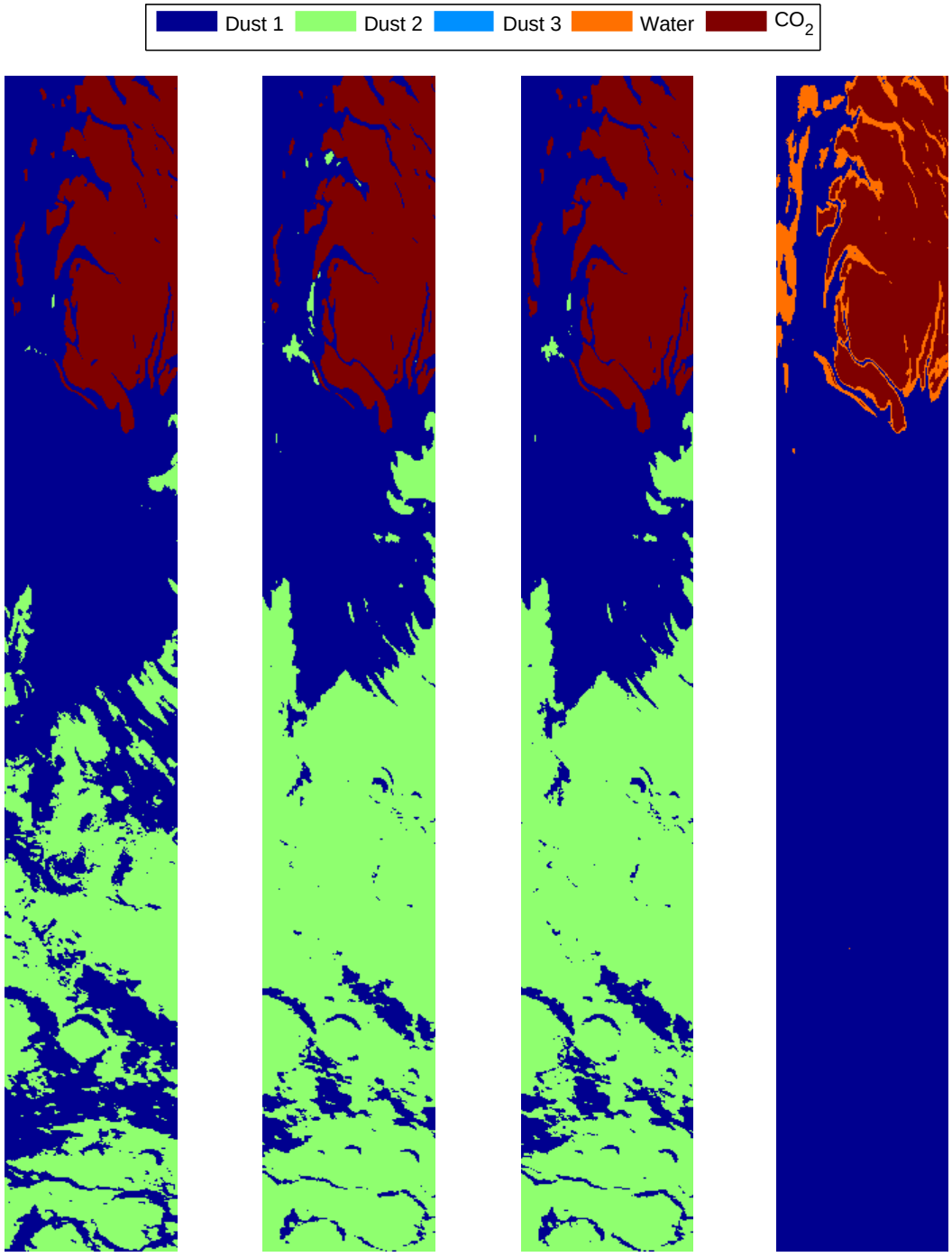
using a method proposed in [93][1]. Specifically, the pixels stem from three classes: "CO_2" ice (brown class in Fig. 7.1(d)), "Water" ice (orange class in Fig. 7.1(d)) and "Dust". The

[1]This has been generated only for assessing the performances of the clustering algorithms qualitatively by no means it is a real reference map. This is the reason that RM and SR measures are not computed in this case.



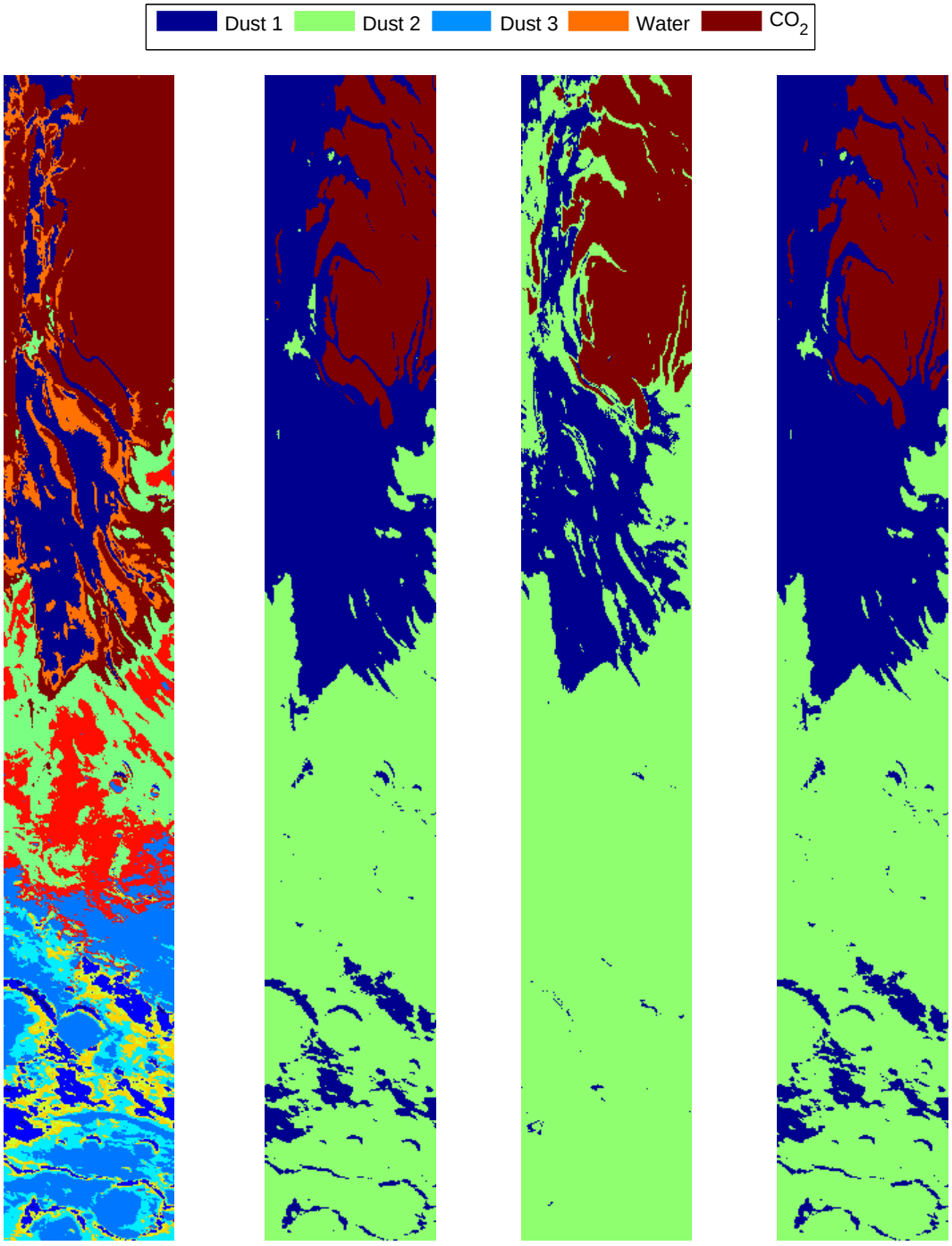
(e) k-means ($m = 3$) **(f) k-means** ($m = 10$) **(g) FCM** ($m = 3$) **(h) FCM** ($m = 10$) **(i) PCM**

latter class is composed of three closely located clusters in the feature space, as it can be deduced from 2nd PC (Fig. 7.1(b)) and it is depicted in Fig. 7.1(d) (dark blue, light blue, green). Figs. 7.1(e)-(u) depict the best mappings obtained by the algorithms running on the full HSI hypercube $871 \times 128 \times 186$, with respect to the "reference map" in Fig 7.1(d). Relative quantitative results are shown in Table 7.1.



(j) UPC (k) UPFC (l) PFCM (m) H2NMF

Concerning k-means and FCM, it can be deduced from Figs. 7.1(e),(g) that even when they are initialized with the true number of classes ($m_{ini} = 3$), they are actually unable to distinguish both (a) the three "Dust" sub-clusters, as well as (b) the "Water" class. Specifically, apart from the "CO_2" class that is correctly identified, they return two additional clusters labeled "Dust 1" and "Dust 2", where "Dust 2" is composed of the two "Dust" areas depicted by green and light blue in Fig. 7.1(d), while "Dust 1" includes the third "Dust" area (dark



(n) KNNCLUST (o) APCM (p) SPCM (q) SAPCM

blue in Fig. 7.1(d)) with the "Water" class. Figs. 7.1(f) and (h) depict the clustering results of k-means and FCM, respectively, when they are initialized with a larger than the actual one number of clusters ($m_{ini} = 10$). Clearly, the algorithms in this case fragment all physical classes. The results of PCM algorithm (Fig. 7.1(i)) are different from those of k-means and FCM. However, PCM also fails to uncover the "Water" class, concluding to a similar result with k-means and FCM when the latter are initialized with $m_{ini} = 3$. The same holds

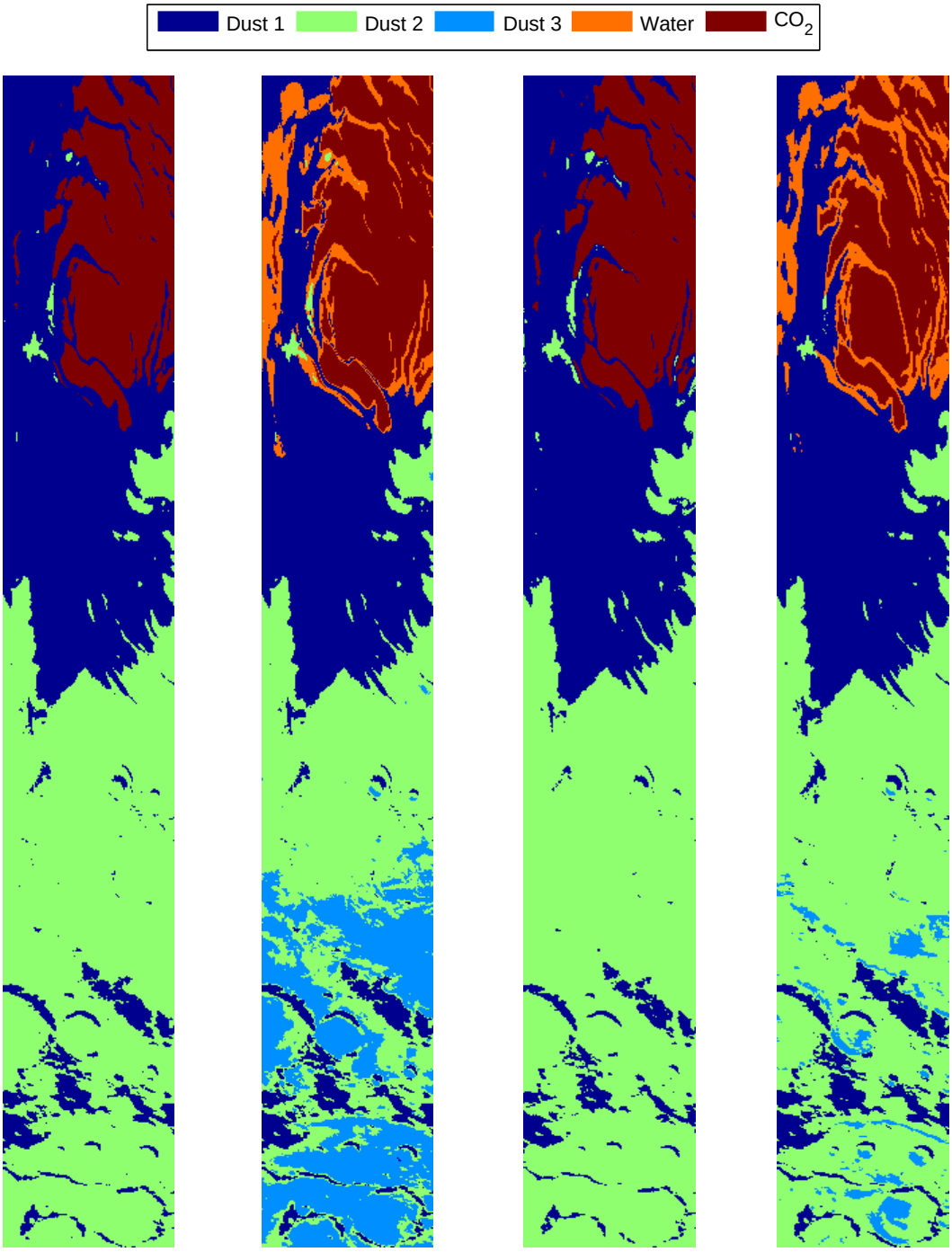
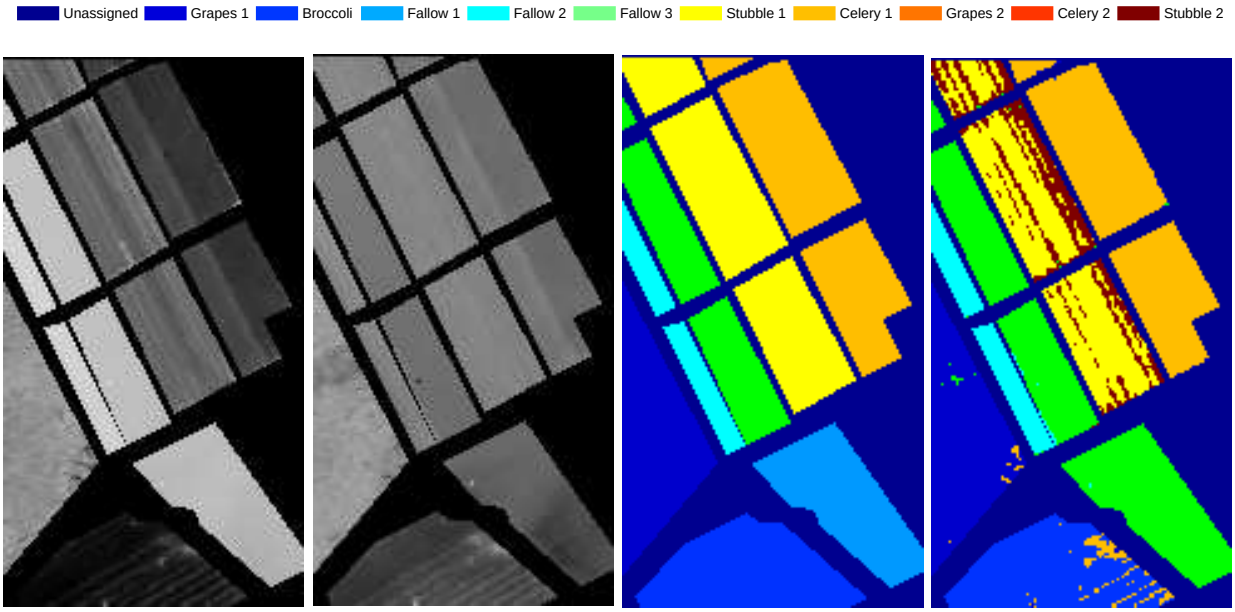


Figure 7.1: (a) 1st PC, (b) 2nd PC, (c) 3rd PC, (d) reference map of the South Polar Cap HSI cube and the corresponding clustering results of (e) k-means ($m = 3$), (f) k-means ($m = 10$), (g) FCM ($m = 3$), (h) FCM ($m = 10$), (i) PCM ($m = 10$), (j) UPC ($m = 3, q = 2$), (k) UPFC ($m = 3$), (l) PFCM ($m = 3$), (m) H2NMF ($m = 3$), (n) KNNCLUST ($K = 1200$), (o) APCM ($m = 3$), (p) SPCM ($m = 10$), (q) SAPCM ($m = 3$), (r) L-SAPCM (1st layer, $m = 3$), (s) L-SAPCM (2nd layer, $m = 3$), (t) O-kmeans ($m = 3$) and (u) O-APCM ($m = 10$).

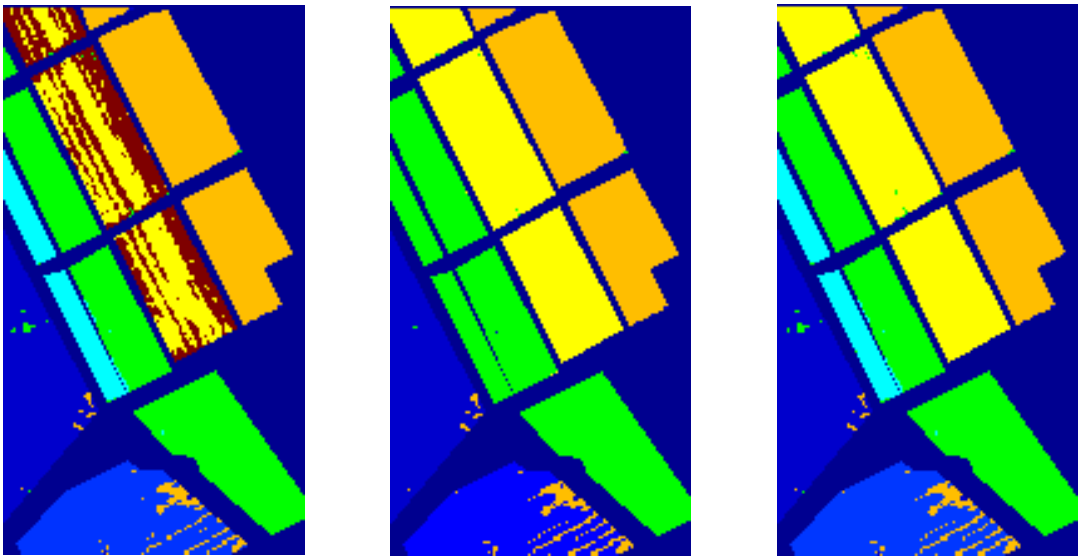
Table 7.1: Clustering results for the South Polar Cap HSI cube

	m_{ini}	m_{final}	Iter	Time
k-means	3	3	8	14.30
k-means	10	10	79	257.65
FCM	3	3	21	12.54
FCM	10	10	144	287.25
PCM	3	2	20	28.22
PCM	10	3	24	348.71
UPC ($q = 2$)	3	3	26	31.43
UPC ($q = 2$)	10	3	28	293.27
UPFC ($q = 2, n = 2, \alpha = 1, b = 1$)	3	3	30	43.72
UPFC ($q = 2, n = 2, \alpha = 1, b = 1$)	10	3	39	348.06
PFCM ($K = 1, q = 2, n = 2, \alpha = 1, b = 1$)	3	3	24	37.27
PFCM ($K = 1, q = 2, n = 2, \alpha = 1, b = 1$)	10	3	52	395.27
H2NMF	3	3	-	6.35
H2NMF	10	10	-	14.31
KNNCLUST ($K = 1200$)	-	10	8	3205.7
APCM ($\alpha = 2$)	3	3	16	46.23
APCM ($\alpha = 1$)	10	3	24	350.61
SPCM	3	2	24	102.86
SPCM	10	3	69	1071.2
SAPCM ($\alpha = 1$)	3	3	16	82.13
SAPCM ($\alpha = 1$)	10	3	30	473.47
L-SAPCM (1st layer: $\alpha = 1$, 2nd layer: $\alpha = 5$)	3	5	248	386.56
O-kmeans ($z = 0.05$)	3	3	-	2.04
O-kmeans ($z = 0.05$)	10	10	-	2.30
O-APCM ($\alpha = 0.8$, $\xi = 1$)	3	5	-	3.07
O-APCM ($\alpha = 0.5$, $\xi = 1$)	10	5	-	3.05

for the UPC, UPFC and PFCM algorithms, which are able to detect three clusters, splitting the “Dust” class into two sub-clusters and merging one of them with “Water” class into a single cluster, thus failing to distinguish the latter class (see Figs. 7.1(j), (k), (l), respectively). The H2NMF hierarchical algorithm succeeds in identifying correctly the three main classes in remarkably short time (Fig. 7.1(m)). However, it was unable to further distinguish the sub-clusters incorporated in the “Dust” class. On the other hand, KNNCLUST algorithm fails to detect the underlying clustering structure, and besides, it takes too long time to converge (Fig. 7.1(n)). In Figs. 7.1(o) and (q), the APCM and SAPCM algorithms are depicted, respectively, which also perform similar to k-means and FCM, when the latter are executed with $m_{ini} = 3$. Specifically, they split the “Dust” class into two clusters “Dust 1” and “Dust 2”, however they fail to distinguish the “Water” class (which is incorporated with “Dust 1” cluster). The SPCM algorithm behaves similarly, but the “Water” class is incorporated with “Dust 2” cluster (see Fig. 7.1(p)). L-SAPCM, performing a clustering of two layers, manages to distinguish the “Water” class, while at the same time, splits the “Dust” class into the three sub-clusters discerned in the second PC component (see the clustering result of the 1st layer of L-SAPCM in Fig. 7.1(r) and the final result (2nd layer) in Fig. 7.1(s)). Finally, concerning the online clustering algorithms O-kmeans and O-APCM, the former behaves as the k-means (Fig. 7.1(t)), however in significantly less time, while



(a) 1st PC **(b) 4th PC** **(c) Ground truth** **(d) k-means**

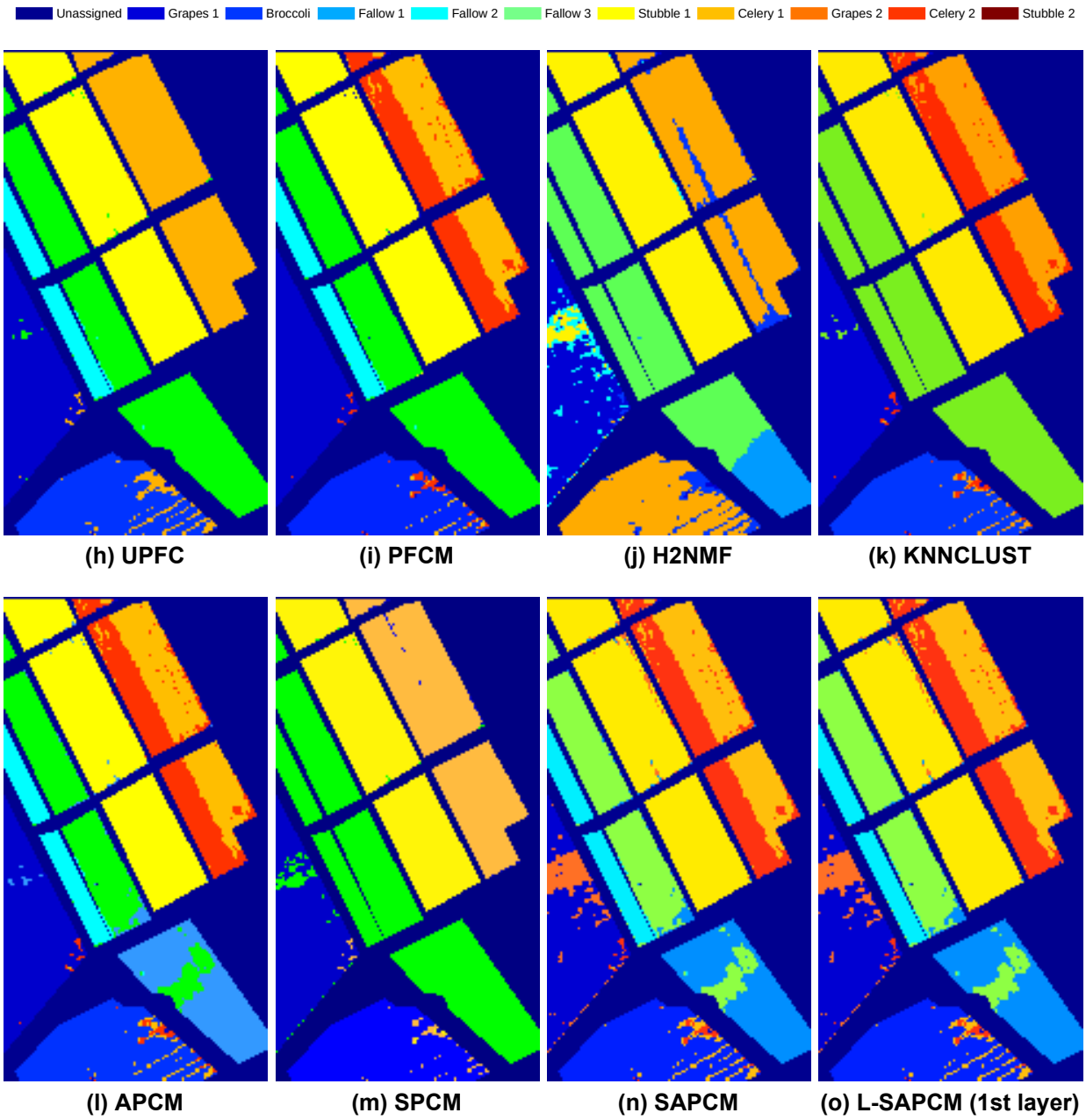


(e) FCM **(f) PCM** **(g) UPC**

the latter concludes to a five-cluster clustering result, identifying correctly all classes, as well as the sub-classes of the “Dust” class at a very short time (Fig. 7.1(u)).

7.3 Case Study 2: Salinas Valley

This data set depicts a subscene of size 220×120 of the flightline acquired by the AVIRIS sensor over Salinas Valley, California [89]. The AVIRIS sensor generates 224 spectral bands across the range from 0.2 to 2.4 μm with a spectral resolution of approximately $10nm$ and a spatial resolution of $3.7m$. The number of bands is reduced to 204 by removing



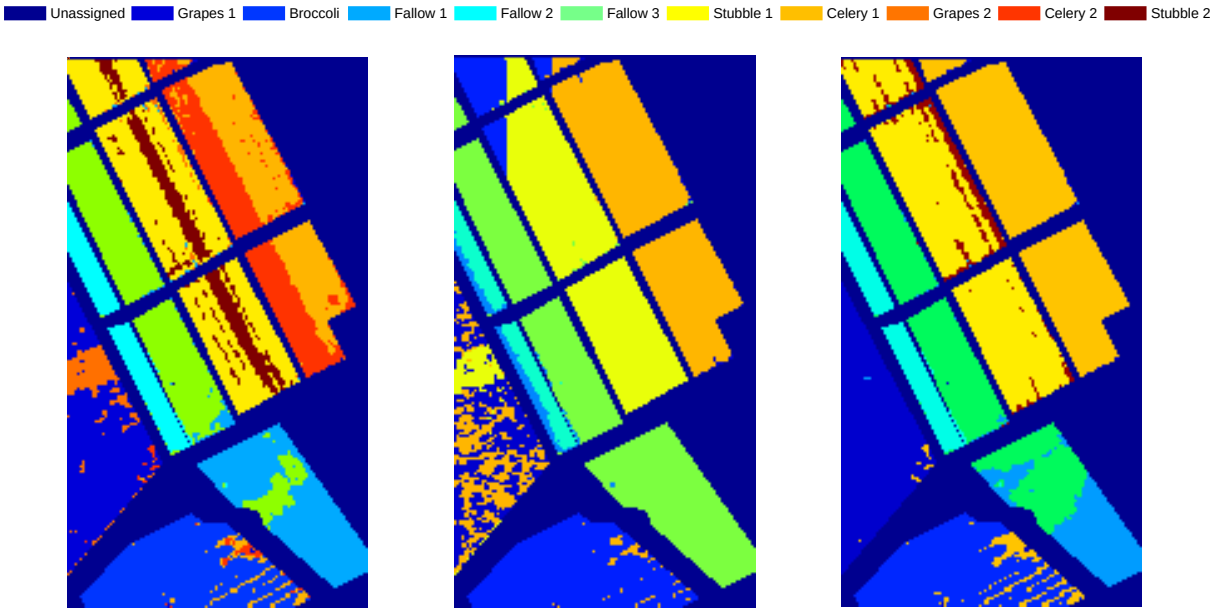
(h) UPFC (i) PFCM (j) H2NMF (k) KNNCLUST

(l) APCM (m) SPCM (n) SAPCM (o) L-SAPCM (1st layer)

20 water absorption bands. 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. 7.2(c). Note that there is no available ground truth information for the dark blue pixels in Fig. 7.2(c)[2]. It is also noted that Figs. 7.2(d)-(r) depict the best mapping obtained by each algorithm taking into account not only the “dry” performance indices, but also their physical interpretation.

As it can be deduced from Fig. 7.2 and Table 7.2, when k-means and FCM are initialized

[2]Note that the last three colors in the color bar on the top of Fig. 7.2 are used in the cases where (one or more of) the classes “Grapes”, “Celery” and “Stubble” are split by the considered algorithm.



(p) L-SAPCM (2nd layer/final) **(q) O-kmeans** **(r) O-APCM**

Figure 7.2: (a) 1st PC, (b) 4th PC, (c) ground truth of the Salinas Valley HSI cube and the corresponding clustering results of (d) k-means ($m = 7$), (e) FCM ($m = 7$), (f) PCM ($m = 15$), (g) UPC ($m = 15$), (h) UPFC ($m = 15$), (i) PFCM ($m = 15$), (j) H2NMF ($m = 7$), (k) KNNCLUST ($K = 1000$), (l) APCM ($m = 15$), (m) SPCM ($m = 15$), (n) SAPCM ($m = 15$), (o) L-SAPCM (1st layer, $m = 15$), (p) L-SAPCM (2nd layer/final, $m = 15$), (q) O-kmeans ($m = 7$) and (r) O-APCM ($m = 15$).

with $m_{ini} = 7$, they actually split the "Stubble" class into two clusters and merge the "Fallow 1" and "Fallow 3" classes (Figs. 7.2(d),(e)). The PCM and SPCM algorithms fail to uncover more than 5 discrete clusters, merging the three different types of the "Fallow" class (Figs. 7.2(f),(m)). The UPC and UPFC algorithms are able to detect up to 6 clusters, merging the "Fallow 1" and "Fallow 3" classes (Figs. 7.2(g),(h),(m)). PFCM, after exhaustive fine tuning of its parameters, manages additionally to distinguish two types of "Celery" (Fig. 7.2(i)), compared to UPC, UPFC and SPCM, although this information is not reflected to the ground-truth labeling; however, it is justified taking into account the 1st or 4th PC (Figs. 7.2(a)-(b)). The same result is achieved by KNNCLUST algorithm (Fig. 7.2(k)). The H2NMF algorithm, when it is initialized with $m_{ini} = 7$, manages to distinguish a part of "Fallow 1" class from "Fallow 3", however, it misses "Fallow 2" class. In addition, it merges "Celery" class with "Broccoli" class, thus failing to detect the latter. APCM, SAPCM, L-SAPCM and O-APCM are the only algorithms that manage to distinguish the "Fallow 1" from the "Fallow 3" class, while at the same time they do not merge any other of the existing classes (Figs. 7.2(l),(n),(p),(r)). Finally, the O-kmeans algorithm concludes to a decreased performance merging "Fallow 1" with "Fallow 3" class and mingling "Grapes" class with "Celery" (Fig. 7.2(q)).

Let us focus for a while on the "Celery" class. This class is formed by two similar yet distinguished from each other "subclasses", although this is not reflected to the ground-

Table 7.2: Clustering results for the Salinas Valley HSI cube

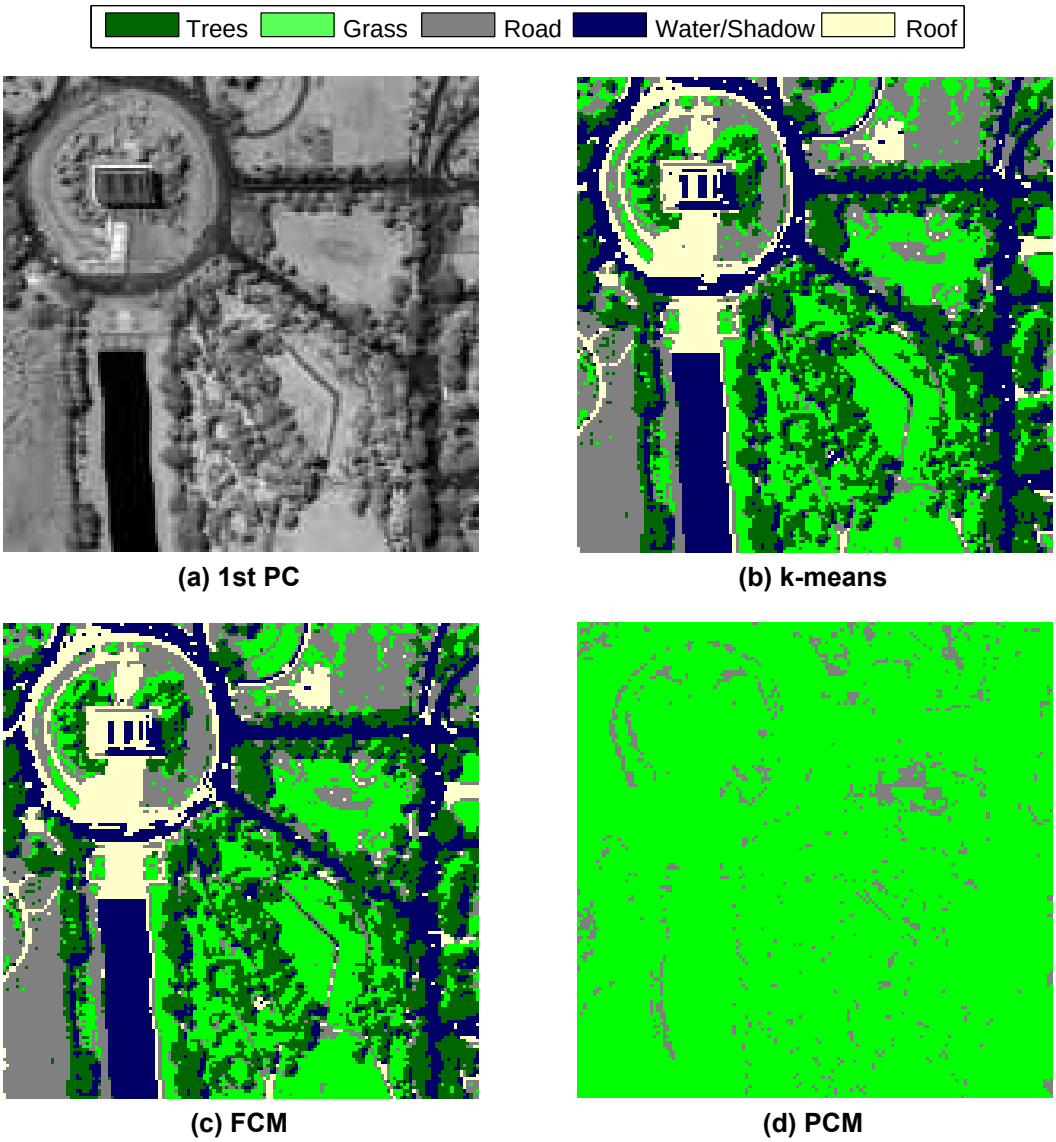
	m_{ini}	m_{final}	RM	SR	Iter	Time
k-means	7	7	93.75	79.89	25	18
k-means	15	15	89.90	59.18	28	60
FCM	7	7	93.18	75.31	99	23
FCM	15	15	89.75	57.89	137	67
PCM	7	4	88.09	69.37	28	52
PCM	15	5	92.75	80.84	29	100
UPC ($q = 4$)	7	3	80.97	57.43	38	27
UPC ($q = 4$)	15	6	95.61	86.21	48	85
UPFC ($q = 4, n = 2, \alpha = 1, b = 5$)	7	3	80.98	57.43	38	31
UPFC ($q = 5, n = 2, \alpha = 1, b = 3$)	15	6	95.67	86.31	45	93
PFCM ($K = 1, q = 2, n = 2, \alpha = 1, b = 7$)	7	3	80.98	57.44	349	148
PFCM ($K = 1, q = 3, n = 2, \alpha = 1, b = 2$)	15	7	94.17	76.86	162	186
H2NMF	7	7	89.03	72.23	-	2.03
H2NMF	15	15	92.14	70.34	-	2.97
KNNCLUST ($K = 1000$)	-	6	93.63	82.10	3	164.31
APCM ($\alpha = 4$)	7	6	95.45	85.92	82	50
APCM ($\alpha = 3$)	15	8	95.91	85.85	191	160
SPCM	7	5	92.73	81.19	35	52
SPCM	15	5	93.33	81.79	47	151
SAPCM ($\alpha = 2$)	7	6	95.85	86.51	71	84
SAPCM ($\alpha = 1.8$)	15	9	95.25	83.40	223	369
L-SAPCM (1st layer: $\alpha = 1.8$, 2nd layer: $\alpha = 1.6$)	15	10	93.37	78.02	758	561
O-kmeans ($z = 0.05$)	7	7	90.34	76.06	-	1.11
O-kmeans ($z = 0.05$)	15	15	90.68	65.10	-	1.25
O-APCM ($\alpha = 0.2$, $\xi = 1$)	7	8	94.26	85.47	-	1.22
O-APCM ($\alpha = 0.12$, $\xi = 1$)	15	8	95.38	89.33	-	0.97

truth labeling (note however that this can be deduced from the 1st PC, as well as the 4th PC component in Fig. 7.2(a),(b)). These subclasses are likely to form two spectrally closely located clusters. It is important to note that only PFCM, KNNCLUST, APCM, SAPCM and L-SAPCM succeed in identifying each one of them. The fact that this is not reflected in the ground-truth labeling causes a misleading decrease in the SR performance of these five algorithms. Similar comments hold for the “Grapes” class, after the inspection of the 4th PC component in Fig. 7.2(b). However, in this case, only the SAPCM and L-SAPCM algorithms succeed in unravelling this situation.

7.4 Case Study 3: Washington DC Mall

This dataset is a part of the HSI flightline acquired by HYDICE over Washington DC Mall [90] that includes the Lincoln Memorial. The HYDICE sensor generates 210 spectral bands across the range from 0.4 to 2.5μm with a spectral resolution of 10nm and a spatial resolution of approximately 1m. The number of bands is reduced to 191 after removing noisy bands. The size of the HSI is 150$\times$150 that results to a total size of $N = 22500$

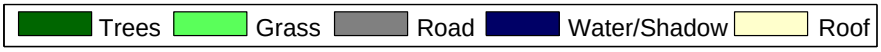
pixels. Note that there is no available ground truth labeling, however, exploting a reference map given in [94] and the 1st PC (Fig. 7.3(a)), we conclude that the pixels stem from five classes: “Trees”, “Grass”, “Road”, “Water/Shadow” and “Roof”. Figs. 7.3(b)-(r) depict the mapping obtained by the algorithms after fine tuning of their parameters, shown in Table. 7.3, where the above classes are depicted by dark green, light green, gray, blue and beige, respectively.

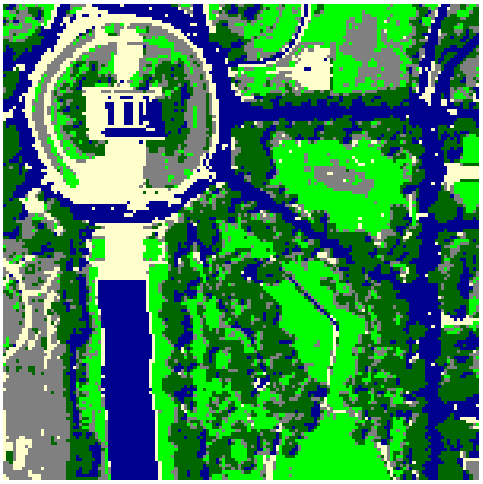


(a) 1st PC **(b) k-means**

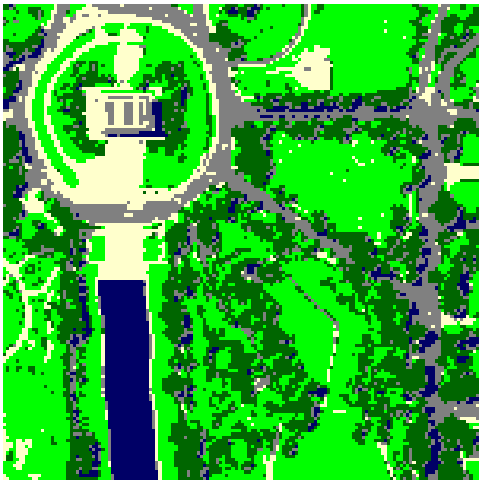
(c) FCM **(d) PCM**

As it can be deduced from Figs. 7.3(b),(c) and Table 7.3, even when k-means and FCM are initialized with the actual number of clusters ($m_{ini} = 5$), they split the “Grass” class into two clusters and merge the “Water/Shadow” with the “Road” class (Figs. 7.3(b),(c)). Similar results are achieved by UPC and UPFC, when they are initialized with $m_{ini} = 5$ (Figs. 7.3(e)(g)). The PCM and SPCM algorithms fail to unravel the underlying clustering structure, both concluding a degraded two-cluster result (Figs. 7.3(d)(m)). The PFCM algorithm is able to detect 4 clusters, missing the “Roof” class (Fig. 7.3(i)). The H2NMF

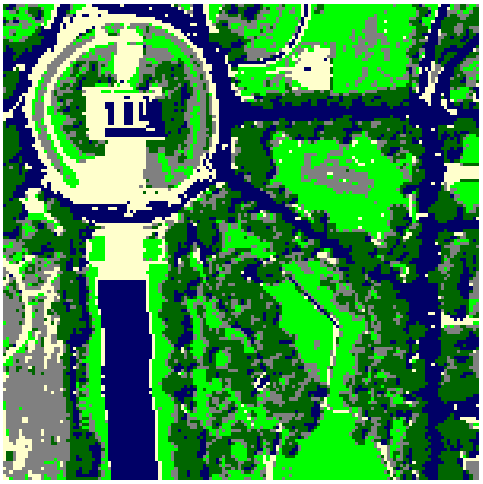




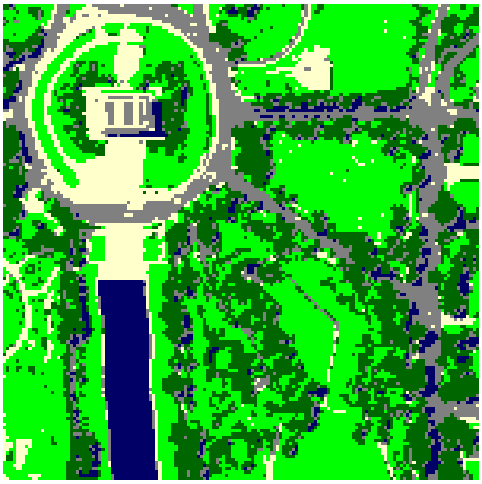
(e) UPC ($m_{ini} = 5$)



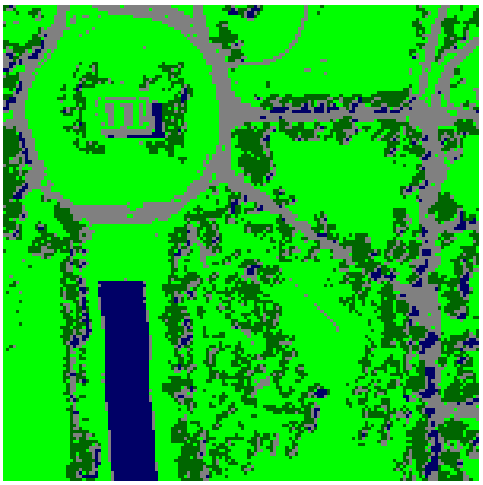
(f) UPC ($m_{ini} = 10$)



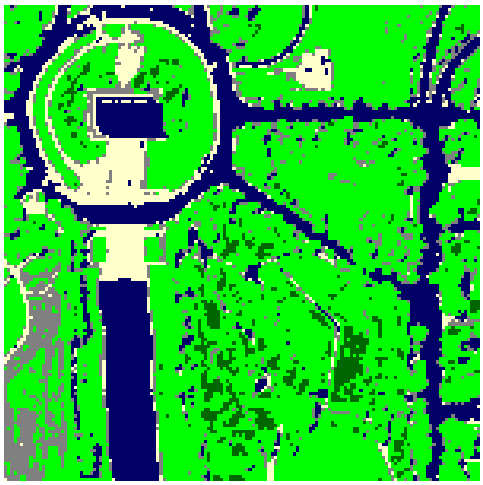
(g) UPFC ($m_{ini} = 5$)



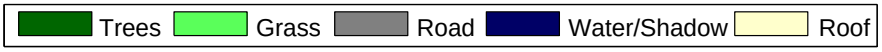
(h) UPFC ($m_{ini} = 10$)

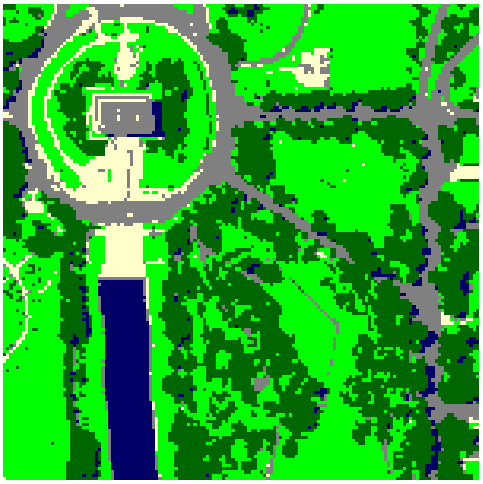


(i) PFCM

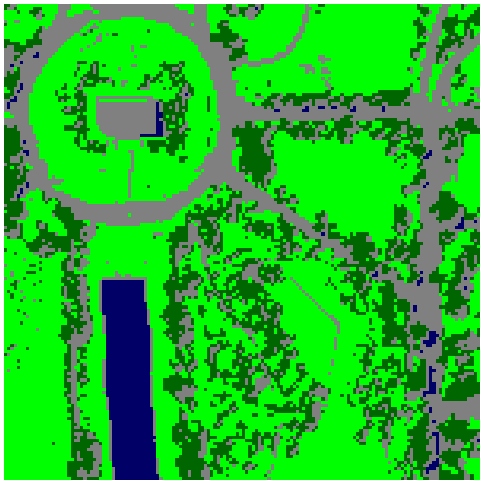


(j) H2NMF

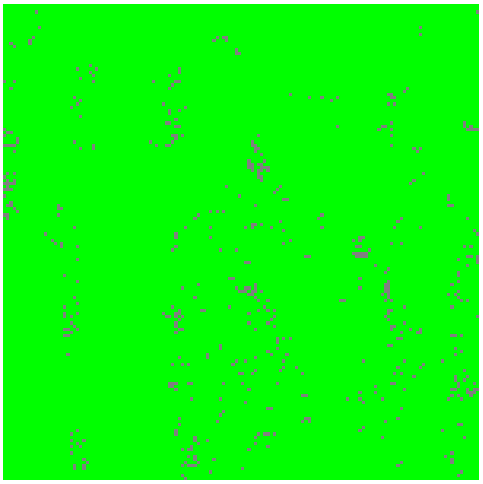




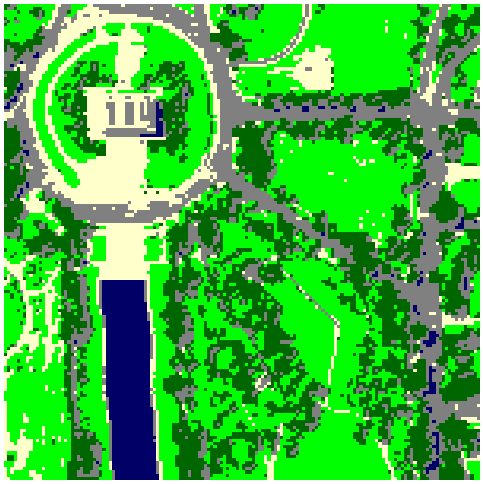
(k) KNNCLUST



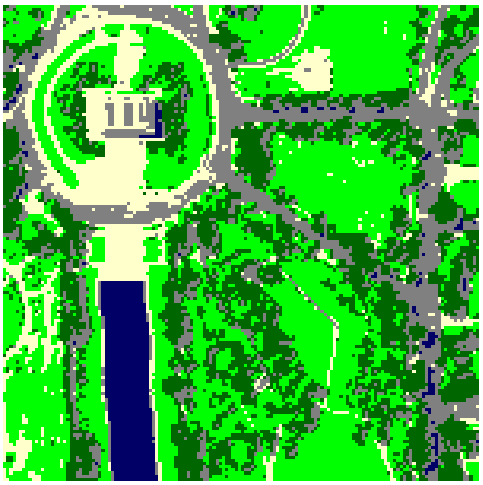
(l) APCM



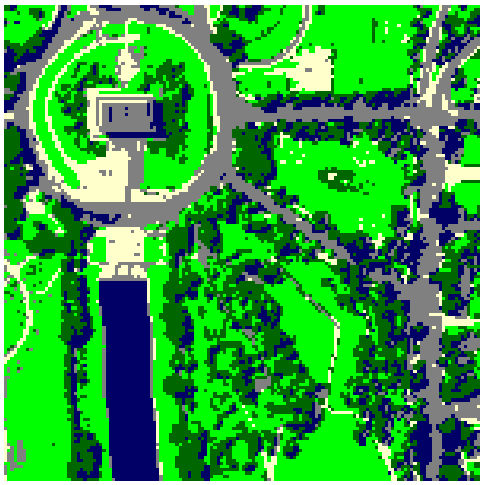
(m) SPCM



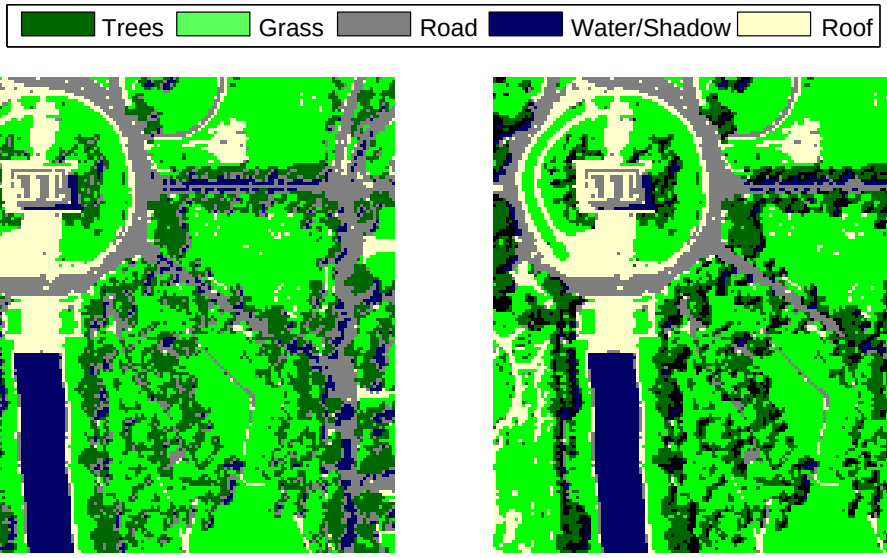
(n) SAPCM



(o) L-SAPCM (1st layer/final)



(p) O-kmeans



(q) O-APCM ($m_{ini} = 5$) **(r) O-APCM ($m_{ini} = 10$)**

Figure 7.3: (a) 1st PC, (b) k-means ($m = 5$), (c) FCM ($m = 5$), (g) FCM ($m = 5$), (d) PCM ($m = 5$), (e) UPC ($m = 5$), (f) UPC ($m = 10$), (g) UPFC ($m = 5$), (h) UPFC ($m = 10$), (i) PFCM ($m = 10$), (j) H2NMF ($m = 5$), (k) KNNCLUST ($K = 1500$), (l) APCM ($m = 10$), (m) SPCM ($m = 10$), (n) SAPCM ($m = 10$), (o) L-SAPCM (1st layer/final, $m = 10$), (p) O-kmeans ($m = 5$), (q) O-APCM ($m = 5$) and (r) O-APCM ($m = 10$).

algorithm merges the "Water/Shadow" with the "Road" class and splits the "Grass" class into three clusters (Fig. 7.3(j)). The APCM algorithm manages to distinguish correctly four classes, without splitting to more than one piece any of them, but fails to uncover the "Roof" class that is incorporated in "Grass" class Fig. 7.3(l). The UPC ($m_{ini} = 10$), UPFC ($m_{ini} = 10$), KNNCLUST, SAPCM, L-SAPCM, O-kmeans and O-APCM succeed in identifying accurately all the underlying classes, without splitting or merging any of them (Figs. 7.3(k),(n)-(r)). Moreover, the last two online algorithms (O-kmeans, O-APCM) run at significantly short time. Finally, it is remarkable that O-APCM manages to identify correctly the underlying clustering structure independently of the initial number of clusters (m_{ini}). Specifically, note that in case O-APCM is initialized with $m_{ini} = 10$, concludes to six clusters, distinguishing correctly all classes and moreover it identifies a sixth cluster corresponding to "Shadow"; distinguishing the latter from the "Water" class (see black-coloured pixels in Fig. 7.3(r)).

Concluding this chapter, we notice that the proposed algorithms exhibit, in general, superior or, at least, comparable performance with respect to several well-known relative algorithms, in identifying with accuracy the underlying clustering structure in all three case studies. Moreover, we highlight the fact that L-SAPCM and O-APCM are able to detect in high detail and precision sub-clusters of which a physical class may be consisted. Finally, it is important to note that none of the proposed algorithms, independently of the given initial number of clusters, splits a physical cluster into two or more clusters. This may be a consequence of the fact that due to their possibilistic nature, the proposed algorithms do not impose a clustering structure but rather (ideally) identify "dense in data" regions

Table 7.3: Clustering results for the Washington DC Mall HSI cube

	m_{ini}	m_{final}	Iter	Time
k-means	5	5	12	37.50
k-means	10	10	34	432.79
FCM	5	5	133	32.31
FCM	10	10	716	392.50
PCM	5	2	60	47.91
PCM	10	2	75	381.57
UPC ($q = 2$)	5	5	184	86.65
UPC ($q = 2$)	10	5	204	508.85
UPFC ($q = 5, n = 2, \alpha = 1, b = 1$)	5	5	324	199.19
UPFC ($q = 4, n = 2, \alpha = 1, b = 1$)	10	5	227	570.64
PFCM ($K = 1, q = 4, n = 2, \alpha = 1, b = 5$)	5	3	238	110.97
PFCM ($K = 1, q = 3, n = 2, \alpha = 1, b = 1$)	10	4	332	632.20
H2NMF	5	5	-	2.37
H2NMF	10	10	-	4.76
KNNCLUST ($K = 1500$)	-	5	9	481.73
APCM ($\alpha = 3$)	5	4	260	158.98
APCM ($\alpha = 2$)	10	4	288	445.67
SPCM	5	2	69	109.41
SPCM	10	2	91	534.58
SAPCM ($\alpha = 3$)	5	4	324	374.04
SAPCM ($\alpha = 2$)	10	5	268	790.59
L-SAPCM (1st layer/final: $\alpha = 2$)	10	5	268	790.59
O-kmeans ($z = 0.01$)	5	5	-	1.67
O-kmeans ($z = 0.01$)	10	10	-	1.76
O-APCM ($\alpha = 1.3$, $\xi = 1$)	5	5	-	1.04
O-APCM ($\alpha = 1.1$, $\xi = 1$)	10	6	-	1.27

in the feature space as clusters. This is also observed for almost all possibilistic-based algorithms used in the present study.

8. A SPARSITY-AWARE FEATURE SELECTION METHOD FOR HYPERSPECTRAL IMAGES

8.1 Introduction

A general issue that needs to be addressed in several classification or clustering tasks is that of feature selection; that is, selecting from a set of available features, used for the representation of the involved entities, the most "informative" ones. In this section, we focus on the HSIs case and we propose a new feature selection method for HSIs. In this context, the entities to be processed are the pixels, which are represented by their spectral signatures (vectors containing the spectral values in the observed bands). Although, there are several available measurements for each HSI pixel (due to the large number of the considered bands) which, in principle, may be a desirable fact, this may also become a source of problems. This is due to the high correlation (mainly) between adjacent bands, as well as to the noise contained in them. This, in turn, may affect the performance of any adopted clustering methodology for identifying homogeneous regions in a given HSI.

The proposed spectral band selection method selects only a small subset of the available bands that are suitable for cluster identification. The selected bands exhibit significant discrimination ability among different kinds of HSI pixels and their choice has also a physical interpretation. The proposed procedure is based on the optimization of a suitably defined *sparsity promoting* cost function. This allows classification or clustering algorithms to provide results of the same quality compared to cases where all spectral bands are used, while, in some cases, it allows the unravelling of some less-obvious patterns. Experimental results on real data have shown remarkable quality of the clustering produced by considering only the selected by the proposed method bands, compared to the case where all bands are used.

8.2 The Feature Selection Method

Let $X = \{\mathbf{x}_i \in \Re^L, i = 1, ..., N\}$ be the set of $N = n_1 \cdot n_2$, L-dimensional data vectors (spectral signatures), each one corresponding to a pixel of the $n_1 \times n_2$ HSI under study (L is the number of the involved bands).

First, we construct the $N \times N$ symmetric distance matrix, $\mathbf{D}_{\text{tot}}$, containing all the squared Euclidean distances[1] between any pair of points, *based on all bands* and then, we determine the L $N \times N$ distance matrices $\mathbf{D}_r$, $r = 1, \ldots, L$, taking each time into account *only the corresponding band* r. That is, in $\mathbf{D}_r$, the distances among the data points will be equal to the squared difference of their rth spectral band values. In the sequel, we vectorize $\mathbf{D}_{\text{tot}}$ columnwise, thus producing a N^2-dimensional column vector, termed $\mathbf{y} = vec(\mathbf{D}_{\text{tot}})$. The

[1] Note that $\|\mathbf{x}_i - \mathbf{x}_j\|^2$ denotes the squared Euclidean distance between vectors $\mathbf{x}_i$, $\mathbf{x}_j$, while $|x_i^{(r)} - x_j^{(r)}|^2$ denotes the squared absolute difference between the scalars $x_i^{(r)}$, $x_j^{(r)}$.

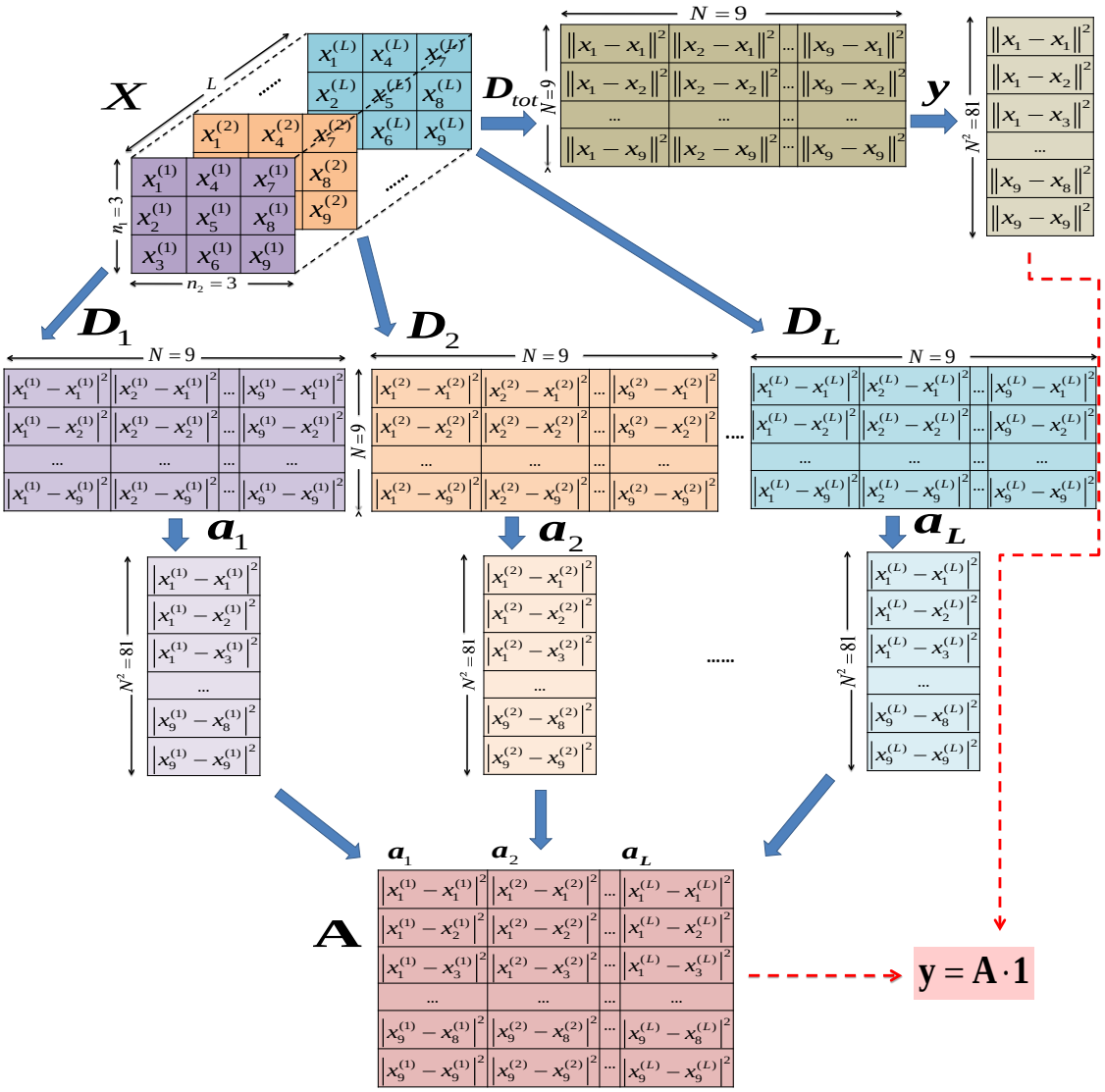


Figure 8.1: Graphical illustration of the adopted problem formulation for a toy 3×3 HSI ($N = 9$).

same is applied to all $\mathbf{D}_r$'s, producing the corresponding N^2-dimensional column vectors, denoted by $\boldsymbol{a}_r$, $r = 1, \ldots, L$. These $\boldsymbol{a}_r$'s are stacked together to form a $N^2 \times L$ matrix **A**. Apparently, $\mathbf{y} = \mathbf{A1}$, where **1** is the all ones column vector. The whole problem formulation is illustrated in Fig. 8.1. In HSIs, it is expected that several spectral bands (and their corresponding $\boldsymbol{a}_r$'s) will not contribute significantly to **y**, since the values of almost all HSI pixels in each one of these bands will be very close to each other. Motivated by this observation, we propose the formulation

$$\mathbf{y} = \mathbf{Aw} + \mathbf{v} = \sum_{r=1}^{L} w_r \boldsymbol{a}_r + \mathbf{v}, \tag{8.1}$$

where $\mathbf{w} = [w_1, \ldots, w_L]^T$ is a $L \times 1$ *sparse* weight vector and the reconstruction error term **v** accumulates the contributions of the "less-significant" spectral bands. Note that the non-

zero components of **w** weight the most significant (for clustering) columns (bands) of **A**, while its zero components annihilate the less significant ones. In order to estimate the sparse vector **w**, we adopt the celebrated lasso minimization criterion [95]

$$\min_{\mathbf{w}}\{||\mathbf{y}-\mathbf{A}\mathbf{w}||_2^2+\lambda||\mathbf{w}||_1\}, \tag{8.2}$$

where λ is a regularization parameter and $||.||_1$, $||.||_2$ denote the l_1 and l_2 distances, respectively. The above formulation imposes sparsity on **w**, via the inclusion of its l_1 norm in the cost function to be minimized. Then, the spectral bands that correspond to the non-zero components of **w** are retained and used to form the HSI pixel feature vector that will be used subsequently in the classification or clustering task (see Fig. 8.2).

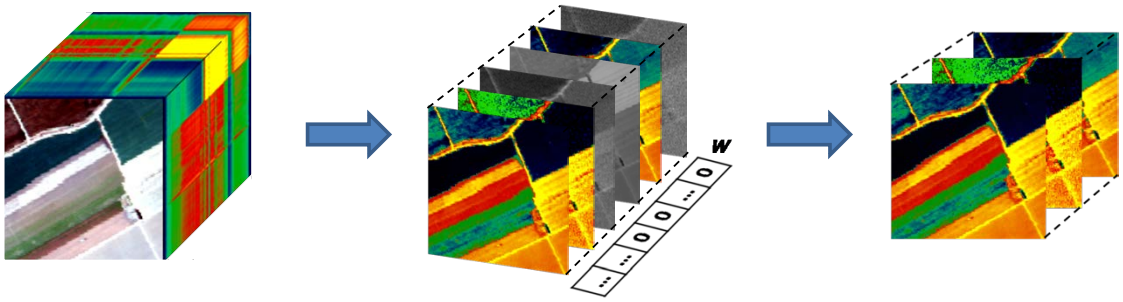


Figure 8.2: Graphical illustration of the procedure that forms the HSI pixel feature vectors using the spectral bands that correspond to non-zero components of w.

In order to solve the above minimization problem, we adopt the Bi-ICE algorithm, proposed in [96], which does not require any parameter fine tuning and works well even for highly correlated data (as is expected to be the case for the columns of **A**). Note that Bi-ICE does not, in general, force some w_r's to be exactly zero, but rather imposes to them very small values. Thus, an appropriate threshold t should be defined so that, after the convergence of Bi-ICE, the w_r's that are less than t are set to zero.

8.3 An extension for large HSI data sets

Although the proposed formulation for feature selection is valid in the HSI clustering framework, it exhibits a significant problem when it is used even for data sets of moderate size. More specifically, the dimensionality of the vectors **y** and $\boldsymbol{a}_r$'s is N^2, which is very large, even for moderate values of N. Thus, solving the lasso may become computationally prohibitive, especially in terms of memory requirements.

In the sequel, we show how the proposed method can be extended for medium-size or large-size HSIs. Specifically, first, we split the original HSI (X) into K smaller equal-sized HSIs with $X_1,\ldots,X_K$ being their corresponding sets of data vectors, each one of size, say m, and $X=\cup_{k=1}^{K}X_k$ (see Fig. 8.3). As a result, the associated **y**'s and $\boldsymbol{a}_r$'s will be of size m^2 in each subimage, with m chosen so that the resulting problem can be handled using the available computational resources. Then, for each such set X_k, $k=1,...,K$,

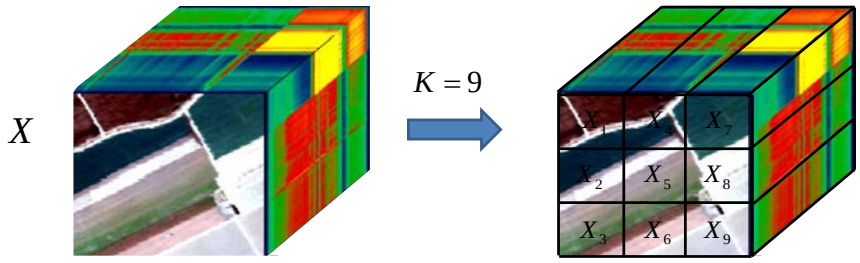


Figure 8.3: An example that an HSI cube is split into $K = 9$ smaller equallised HSIs.

we solve the above lasso minimization problem and let F_k denote the set of selected features for the corresponding set X_k. Note that the rth feature is included in F_k, if $w_r > t$, where t is the user-defined threshold mentioned before. In the sequel, we compute the frequency of occurrence, p_r, of the rth feature in all the F_k's, $k = 1, ..., K$. The final set of features F, concerning the whole data set X, will comprise those features for which $p_r > b$, $r = 1, \ldots, L$, where $b \in [0, 1]$ is a user-defined constant and indicates the minimum required frequency of occurrence of a feature in all sets $F_1, \ldots, F_K$. Clearly, the larger the b, the less features are likely to be included in F.

8.4 Experimental Results

The proposed method is assessed through tests performed on an 150×150 HSI dataset from a rural area in Salinas California. This dataset is a part of a scene of the flightline acquired by the AVIRIS sensor over Salinas Valley, California with 204 bands (after removal of 20 noisy bands). Salinas classification used as reference contains eight vegetation classes: Cs-1 and Cs-2 (two types of "broccoli"), Cs-3 ("grapes"), Cs-4 ("corn") and Cs-5 - Cs-8 (four types of "lettuce"), (see Fig. 8.6(a)). The aim here is to apply the proposed methodology, in order to get a reduced set of features and then to run a clustering algorithm (APCM, see chapter 3) on both the complete and reduced feature vector sets, in order to assess the impact of the proposed methodology.

In **Experiment 1**, the proposed band selection method was applied on a 40×40 sub-image of the HSI under study that includes the four "lettuce" classes Cs-5 - Cs-8. The method selected 24 out of the 204 bands (for $t = 0.1$) as the most significant ones for clustering (Fig. 8.4). Then, the APCM clustering method [54] (with m_{ini}=10, α=1) was applied for (i) all the 204 bands and (ii) the 24 selected bands. In both cases, five clusters are distinguished, as shown in Fig. 8.5(b) and Fig. 8.5(c), respectively, giving both SR=99.25%. Three clusters Cl-5, Cl-7, Cl-8 correspond to the three reference classes Cs-5, Cs-7, Cs-8, while the reference class Cs-6 is divided into clusters Cl-6a and Cl-6b. This can be explained by the different spectral pattern of the latter within bands 25-40, which, however, present the same diagnostic absorption features in position and depth at higher wavelengths, indicating thus the same species over the same soil with different abundances in the pixel.

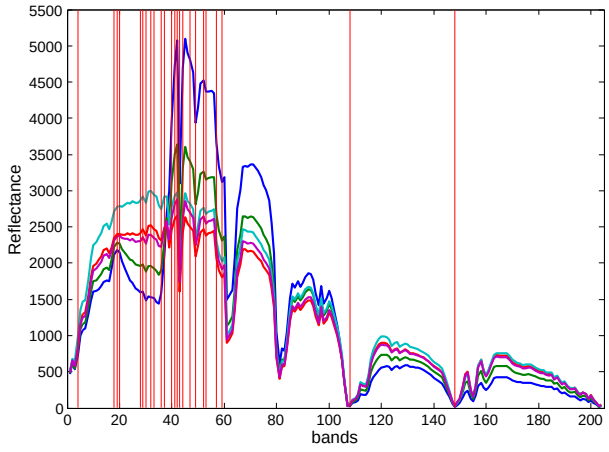
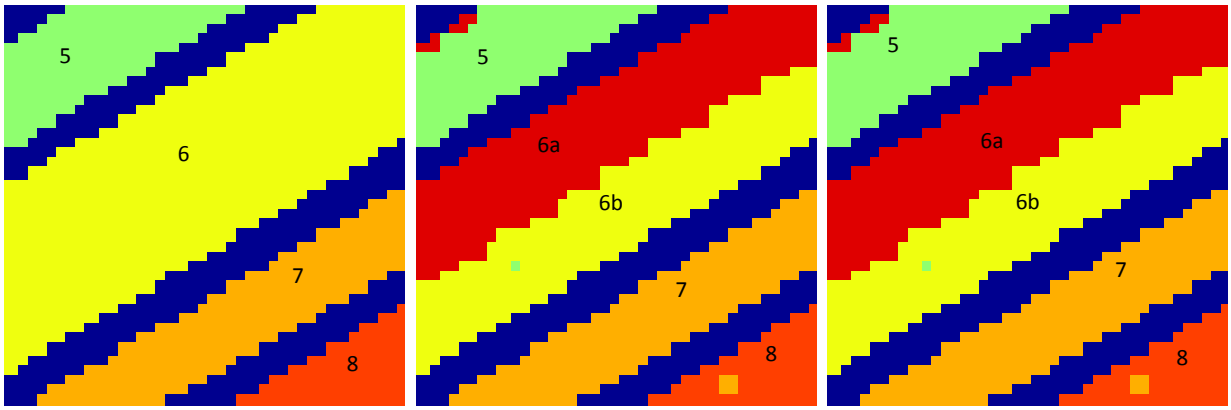


Figure 8.4: The 5 mean cluster signatures of Experiment 1 and the 24 selected bands (marked with red lines).

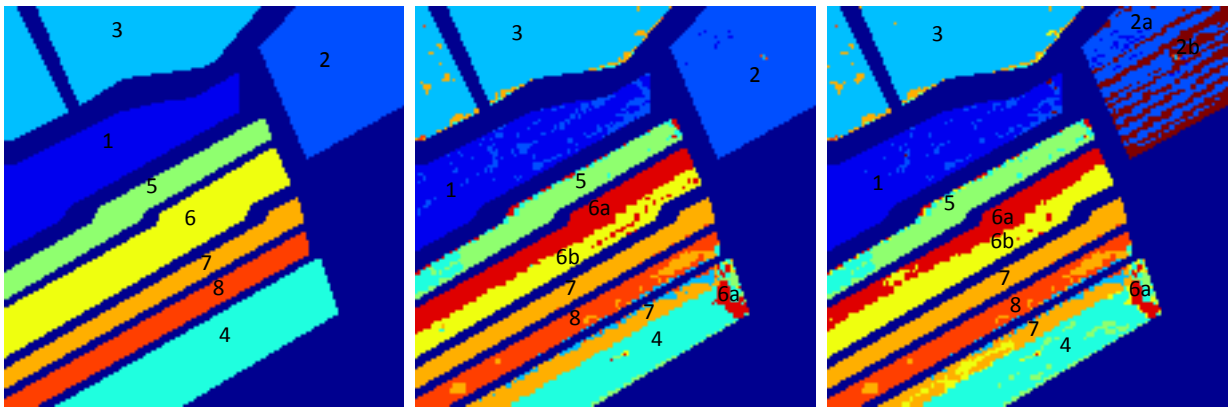


(a) Reference image **(b) All bands clustering (99.25%)** **(c) 24 selected bands clustering (99.25%)**

Figure 8.5: Results of Experiment 1. (a) The reference classified image. The clustering results obtained from APCM ($m_{ini} = 10$, $\alpha = 1$), when (b) all bands are used (5 clusters), (c) only the 24 selected bands are used (5 clusters). Labels in (a) correspond to classes (Cs) and in (b),(c) to clusters (Cl).

Fig. 8.4 shows the mean spectral signatures of the 5 clusters identified in Experiment 1, as well as the 24 bands selected by our method. It is interesting to note that most of them are located in the VNIR area (bands 20-60) over characteristic vegetation features. Significant bands are located either in the maximum absorption depth of typical vegetation features, such as the red absorption and the infra-red water absorptions, or in abrupt reflectance changes such as the red edge pattern (680-750nm). It must be noted here, that the algorithm does not necessarily select all the bands where significant differentiation between signatures occurs (for example, bands 68 to 72).

In **Experiment 2**, the extended version of the proposed band selection method was applied to the whole 150$\times$150 HSI image (as aforementioned, with eight classes) and 33



(a) Reference image **(b) All bands clustering (92.01%)** **(c) 33 selected bands clustering (91.82%)**

Figure 8.6: Results of Experiment 2. (a) The reference classified image. The clustering results obtained from APCM ($m_{ini} = 20$, $\alpha = 2$), for (b) all bands (9 clusters), (c) for the 33 selected bands (10 clusters).

bands were selected (for $t = 0.2$ and $b = 0.5$). Then, clustering via the APCM algorithm (m_{ini}=20, α=2) was performed using (i) all spectral bands (Fig. 8.6(b), SR=92.01%) and (ii) the 33 selected bands (Fig. 8.6(c), SR=91.82%). As shown in Fig. 8.6, in both cases the results provided additional information compared to the reference classes. In particular, both distinguished Cl-6a and Cl-6b for Cs-6, while Cl-7 corresponds to the whole Cs-7 and a compact rectangular region of Cs-4. Also, we have additional spectral presence of Cl-6a at the right edge of the Cs-4 area. Furthermore, the clustering using the 33 selected bands provided additional information on Cs-2, as shown in Fig. 8.6(c) in the form of a new cluster Cl-2b (named “soil/plant”) appearing in linear stripes within the Cs-2 area (upper right part of the image). This seems to correspond to mixed vegetation/soil pixels with higher spectral participation of soil in the overall spectral signature of the pixel.

8.5 Conclusion

In this chapter, a novel sparsity-aware feature selection method for HSI clustering is presented, which is based on the optimization of a sparsity promoting cost function. The proposed method is likely to select less correlated bands, which, in principle, will help subsequent classification or clustering tasks in identifying patterns that are not identified using the whole bands information. Finally, the proposed method was tested in the clustering framework and gave results of the same quality with those produced in the case where the full pixels’ spectral signatures are used.

9. CONCLUSION AND FUTURE DIRECTIONS

In the present thesis, we have addressed the task of clustering within the possibilistic framework, introducing three novel features: parameter adaptivity, sparsity and online processing. Although the resulting algorithms are of generic nature, in this thesis they are mainly applied for hyperspectral image (HSI) processing, aiming at the identification of homogeneous regions in HSIs. In this concluding chapter, we review the main contributions of the thesis and we discuss some directions for future research.

9.1 Summary

The first contribution of the thesis is a novel clustering scheme, called Adaptive Possibilistic C-Means (APCM) (chapter 3), whose main feature is that its involved parameters γ's, after their initialization, are properly adapted at each iteration of the algorithm. This renders APCM more flexible in tracking the variations in the cluster formation occurring as the algorithm evolves. Specifically, in contrast to most state-of-the-art possibilistic clustering algorithms, APCM is able to uncover the underlying clustering structure, even in cases where the physical clusters are closely located to each other and/or have significant differences in their variances. In addition to the above, as direct consequence of the adaptation mechanism embedded in APCM, a long-standing issue in the clustering literature is overcome. In particular, APCM is equipped with a cluster elimination mechanism, which makes it capable to reveal the actual number of physical clusters formed by the data set under study, as well as the clusters themselves, provided that the algorithm starts with an overestimate of it. In addition, theoretical results that are indicative of the convergence behavior of APCM, are also provided. Finally, the provided experimental results show that APCM exhibits superior clustering performance compared to several related algorithms and in particular the conventional PCM.

As another direction in expanding the possibilistic clustering framework, we considered the issue of sparsity. The sparsity hypothesis is supported by the fact that, in practice, a data point may be compatible with only one or a few (or even none) clusters. To this end, our research efforts have focused on inducing sparsity on the vectors containing the degrees of compatibility of each data point with the clusters, giving rise to the so called Sparse Possibilistic C-Means (SPCM) algorithm (chapter 4). SPCM exhibits increased immunity to the presence of data points that may be considered as noise or outliers, by not allowing them to contribute to the computation of the cluster representatives. Thus, the exploitation of sparsity makes SPCM capable in dealing well (in principle) with closely located clusters that may also be of different densities. This, in turn, implies that SPCM ends up with more accurate estimates for the cluster representatives, compared to other related possibilistic algorithms, especially in noisy environments. Finally, a rigorous convergence analysis of SPCM has been also conducted, which shows that the iterative sequence generated by SPCM converges to a local minimum of its associated cost function. The proof is not trivial with the main source of difficulty, compared to those given for previous possibilistic algo-

rithms, being the lack of continuity of the updating formula of the degrees of compatibility, which is actually a two-branch expression. This theoretical analysis applies also trivially to the conventional PCM algorithm [10], leading to much stronger convergence results, compared to those given in the literature.

The next step was an effort towards the combination of the main features of the APCM and SPCM algorithms into a single scheme. The resulting algorithm, called Sparse Adaptive Possibilistic C-Means (SAPCM) (chapter 5), inherits all the advantages of APCM and SPCM, being able to deal well with even more challenging situations, where clusters may be of significantly different variances and/or densities, and exhibiting robustness to outliers. More specifically, SAPCM has the ability (a) to cope well with closely located clusters with possibly different densities and/or variances, (b) to determine the number of natural clusters and (c) to improve even more the estimates of the cluster representatives compared to SPCM and APCM. The overall advantages of SAPCM, compared to other related algorithms, are verified via extensive experimentation.

Continuing with our tour in the thesis, we met next two variants of SAPCM (chapter 5). The first one is called Sequential Sparse Adaptive Possibilistic C-Means (SeqSAPCM) algorithm and is an iterative bottom-up version of SAPCM. Specifically, SeqSAPCM, which adopts SAPCM as its structuring element, begins with two clusters and proceeds by examining at each iteration whether a new cluster can be formed. Thus, it unravels sequentially the underlying clustering structure. The basic advantage of SeqSAPCM is that it does not require an initial (over)estimation of the number of clusters, like APCM, SPCM and SAPCM. The second variant of SAPCM, called Layered Sparse Adaptive Possibilistic C-Means (L-SAPCM) algorithm, also utilizes SAPCM as its structuring element (yet in a different way than that of SeqSAPCM) and works in layers. Specifically, the SAPCM algorithm is initially applied on the whole data set and then it is recursively applied separately on each resulting cluster, in order to reveal possible clustering structure within it, working as in a tree structure basis. The L-SAPCM algorithm terminates when none of the so far produced clusters contains further clustering structure within it. L-SAPCM, due to its philosophy, can distinguish very closely located clusters that lie at different “resolutions” in the feature space (i.e. in cases where the variances of the clusters may differ orders of magnitude from each other). This feature has been confirmed experimentally on HSI data, where clusters with variances differing orders of magnitude from each other may appear.

The most well-known clustering schemes, as well as the so far discussed ones, operate in a batch mode, that is, they need to consider the whole data set at each iteration before updating their parameters. Thus, they turn out to be extremely demanding in terms of computational complexity when processing large sized and/or high dimensional data sets (e.g., HSIs). The above issue was the motivation to explore the possibility of transforming the proposed batch clustering schemes to online, where data points are processed one-by-one. To this end, we developed an efficient online version of the APCM algorithm, called Online Adaptive Possibilistic C-Means (O-APCM) (chapter 6), which embodies three new procedures for (a) generating, (b) merging and (c) deleting clusters dynamically, as new data points become available. Instead of storing and using the whole data set, O-APCM processes data vectors one by one and memorizes their impact to suitably defined pa-

rameters. As a result, it is much more computationally efficient and has very low memory requirements with no degradation on the resulting clustering quality, compared to its batch counterpart. This makes O-APCM a good candidate for clustering of big data sets, whose size and dimensionality are prohibitive for batch algorithms. It is worth highlighting that O-APCM can be utilized for applications in both stationary, as well as dynamically varying environments, where the centers of the physical clusters may change their location in data space over time. Experimental results show that O-APCM offers high discrimination capability at a very low computational cost, in both static and dynamically changing environments.

After the presentation of the new clustering algorithms, a comparative study of all of them with several other related algorithms is performed, on the basis of HSIs processing (chapter 7). Specifically, three HSIs depicting areas with totally different land cover are used. The obtained results are commented and interpreted on the basis of the nature of the application at hand and it is shown that most of the proposed clustering algorithms exhibit superior performance with respect to other related algorithms.

Departing from the thematic area of clustering and entering to the area of feature selection that is closely related to it, a sparsity-aware feature selection method suitable for hyperspectral data has been developed (chapter 8). The proposed method is based on the optimization of a sparsity promoting cost function, in order to identify the spectral bands that are more informative and thus possess the higher discrimination capability. Experimental results on real data have shown remarkable quality of clustering, resulting from a certain algorithm, considering only the selected features/bands.

9.2 Future Directions

The present thesis has opened up several possible directions for future work, as it is described in the sections to follow.

9.2.1 Further Extensions on the Proposed Algorithms

In all clustering possibilistic algorithms in the bibliography, as well as those proposed in the present thesis, each cluster is associated with a parameter γ that is a measure of its variance around its representative. This parameter actually defines a hypersphere around the representative of the cluster and, as a consequence, the presence of hyperspherically-shaped clusters is implied. Therefore, in the case where e.g. hyperellipsoidally-shaped clusters are considered, a single parameter like γ cannot capture the variance around the representatives in the various directions. One way to address this issue would be to model each cluster with a hyperellipse centered on its cluster representative, at the cost of inserting more parameters per cluster which, in turn, need to be estimated. It is expected that this approach will work efficiently only for low dimensional data sets.

In chapter 6 we presented an online implementation of the APCM algorithm, named O-

APCM, that requires very low memory and computation time compared to a batch implementation. Another research possibility would be the online implementation of the SAPCM algorithm (O-SAPCM), aiming at exploiting its sparsity-promoting characteristic in a sequential framework, in order to reduce significantly the required computational burden. In addition, an online implementation of the proposed layered sparse adaptive possibilistic c-means clustering algorithm, L-SAPCM, is a further interesting extension, that may lead to a layered online sparse adaptive possibilistic c-means clustering algorithm, L-OSAPCM. In particular, L-OSAPCM algorithm may perform online clustering at each layer employing, e.g. O-SAPCM as its structuring element, thus inheriting all the advantages of a layered clustering scheme and, additionally, reducing significantly the computational time and complexity.

9.2.2 Subspace Possibilistic Clustering

In certain applications, the data vectors are aggregated along subspaces of the (usually high dimensional) feature space. Subspace clustering aims at finding a multi-subspace representation that best fits a collection of data points taken from a high dimensional space. The problem here is not only to identify the clusters themselves but also the subspaces where these clusters live. In this spirit, an extension of possibilistic clustering that is worth to be investigated, is under the framework of subspace clustering. Specifically, the development of subspace possibilistic clustering algorithms that aim at the identification of the clusters along with the subspaces where they live, enriched with the features of adaptivity and sparsity introduced in this thesis, is a very challenging future research topic.

ABBREVIATIONS

Abbreviation	Meaning
ABC	Artificial Bee Colony
AMPCM	Automatic Merging Possibilistic Clustering Method
APCM	Adaptive Possibilistic C-Means
AVIRIS	Airborne Visible/Infrared Imaging Spectrometer
Bi-ICE	Bayesian Inference Iterative Conditional Expectations
FCLS	Fully Constrained Least Squares
FCM	Fuzzy C-Means
GMVQ	Gauss Mixture Vector Quantization
GRM	Generalized Rand Measure
HSI	HyperSpectral Image
HYDICE	HYperspectral Digital Imagery Collection Experiment
i.i.d	Independent and identically distributed
L-SAPCM	Layered Sparse Adaptive Possibilistic C-Means
MD	Mean Distance
MRF	Markov Random Fields
O-APCM	Online Adaptive Possibilistic C-Means
OMEGA	Observatoire pour la Minéralogie, l'Eau, les Glaces et l'Activité
PCM	Possibilistic C-Means
PFCM	Possibilistic Fuzzy C-Means
SAPCM	Sparse Adaptive Possibilistic C-Means
SEM	Stochastic Expectation-Maximization
SeqSAPCM	Sequential Sparse Adaptive Possibilistic C-Means
SPCM	Sparse Adaptive Possibilistic C-Means
SR	Success Rate
SSC	Sparse Subspace Clustering
SWIR	Short-Wave InfraRed
UPC	Unsupervised Possibilistic Clustering
UPFC	Unsupervised Possibilistic Fuzzy Clustering
VNIR	Visible Near InfraRed

NOTATION

Symbol	Meaning
x	Scalar
$\|x\|$	Absolute value of a scalar
$\mathbf{x}$	Vector
$\mathbf{0}$	Zero vector
$\mathbf{1}$	All ones vector
X	Matrix or set
X^T	Transpose of matrix X
I_l	$l \times l$ identity matrix
$\|X\|$	Determinant of matrix X
$CH(X)$	Convex hull of set X
$\|\mathbf{x}\|_p$	ℓ_p norm of vector $\mathbf{x}$
$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	Normal distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$
$\Re$	Field of real numbers
$\Re^l$	l-dimensional space of real numbers
$\approx$	Approximately equal

APPENDIX A. PROOF OF APCM PROPOSITION

Proposition 1. *Let* $\gamma'_j = \frac{\sum_{\mathbf{x}_i: u_{ij}=\max_{r=1,\ldots,m} u_{ir}} \|\mathbf{x}_i - \boldsymbol{\mu}_j\|^2}{n_j}$ *and* $\eta_j^2 = \left(\frac{\sum_{\mathbf{x}_i: u_{ij}=\max_{r=1,\ldots,m} u_{ir}} \|\mathbf{x}_i - \boldsymbol{\mu}_j\|}{n_j}\right)^2$ *(see eq.* (3.5)*). Then* $\eta_j^2 \leq \gamma'_j$.

Proof. Let $q_{ij} = \|\mathbf{x}_i - \boldsymbol{\mu}_j\|$ and $\mathbf{q}_j = [q_{i1}, \ldots, q_{in_j}]^T$. Then $\gamma'_j = \frac{1}{n_j}\|\mathbf{q}_j\|_2^2$ (squared l_2-norm) and $\eta_j^2 = \frac{1}{n_j^2}\|\mathbf{q}_j\|_1^2$ (squared l_1-norm). From the relation between the l_1 and l_2 norms (see e.g. [71]), it is: $\|\mathbf{q}_j\|_1 \leq n_j^{1/2}\|\mathbf{q}_j\|_2 \leq n_j^{1/2}\|\mathbf{q}_j\|_1$ or $\|\mathbf{q}_j\|_1^2 \leq n_j\|\mathbf{q}_j\|_2^2 \leq n_j\|\mathbf{q}_j\|_1^2$ or $\frac{1}{n_j^2}\|\mathbf{q}_j\|_1^2 \leq \frac{1}{n_j}\|\mathbf{q}_j\|_2^2 \leq n_j\frac{1}{n_j^2}\|\mathbf{q}_j\|_1^2$ or $\eta_j^2 \leq \gamma'_j \leq n_j\eta_j^2$. Note that for finite n_j values, η_j^2 and γ'_j are of the same magnitude. □

APPENDIX B. PROOFS OF SPCM PROPOSITIONS

Proof of Proposition 2. Taking the derivative of $f(u_{ij})$ with respect to u_{ij}, we obtain

$$\frac{\partial f(u_{ij})}{\partial u_{ij}} = \gamma_j u_{ij}^{-1}\left[1 - \frac{\lambda}{\gamma_j}p(1-p)u_{ij}^{p-1}\right]. \tag{B.1}$$

Solving $\frac{\partial f(u_{ij})}{\partial u_{ij}} = 0$ with respect to u_{ij} and taking into account that $u_{ij} > 0$ (by definition), after some elementary algebraic manipulations we have the following solutions

$$\hat{u}_{ij} = \left[\frac{\lambda}{\gamma_j}p(1-p)\right]^{\frac{1}{1-p}} \text{ and } \tilde{u}_{ij} = +\infty. \tag{B.2}$$

□

Proof of Proposition 3. It suffices to show that $\frac{\partial f(u_{ij})}{\partial u_{ij}} \leq 0$ for $u_{ij} \in (0, \hat{u}_{ij}]$ and $\frac{\partial f(u_{ij})}{\partial u_{ij}} \geq 0$ for $u_{ij} \in [\hat{u}_{ij}, +\infty)$. Indeed, for $u_{ij} \in (0, \hat{u}_{ij}]$ we have $u_{ij} \leq \hat{u}_{ij}$, which implies that $u_{ij}^{1-p} \leq \frac{\lambda}{\gamma_j}p(1-p)$ (from eq. (B.2)) or $1 \leq \frac{\lambda}{\gamma_j}p(1-p)u_{ij}^{p-1}$. From the latter and taking into account eq. (B.1) again, it follows that $\frac{\partial f(u_{ij})}{\partial u_{ij}} \leq 0$ in $u_{ij} \in (0, \hat{u}_{ij}]$. Similarly, for $u_{ij} \in [\hat{u}_{ij}, +\infty)$ we have $u_{ij} \geq \hat{u}_{ij}$, which, utilizing eq. (B.2), implies that $u_{ij}^{1-p} \geq \frac{\lambda}{\gamma_j}p(1-p)$ or $1 \geq \frac{\lambda}{\gamma_j}p(1-p)u_{ij}^{p-1}$. From the latter and taking into account eq. (B.1), it follows that $\frac{\partial f(u_{ij})}{\partial u_{ij}} \geq 0$ in $u_{ij} \in [\hat{u}_{ij}, +\infty)$. Consequently, $\hat{u}_{ij}$ is the unique minimum of $f(u_{ij})$, since in $[\hat{u}_{ij}, +\infty)$, $f(u_{ij})$ is increasing and, as a consequence, $\tilde{u}_{ij}$ is not a minimum of $f(u_{ij})$. □

Proof of Proposition 4. It is $f(1) = d_{ij} + \gamma_j \ln 1 + \lambda p \cdot 1^{p-1} = d_{ij} + \lambda p > 0$. Moreover, it is $f(0) = \lim_{u_{ij}\to 0^+} f(u_{ij}) = \lim_{u_{ij}\to 0^+}\left(d_{ij} + \gamma_j \ln u_{ij} + \lambda p u_{ij}^{p-1}\right) = d_{ij} + \lim_{u_{ij}\to 0^+}\left[\frac{1}{u_{ij}^{1-p}}\left(\gamma_j u_{ij}^{1-p}\ln u_{ij} + \lambda p\right)\right] = +\infty$, as it follows from the application of the L' Hospital rule, since $\lim_{u_{ij}\to 0^+}\frac{1}{u_{ij}^{1-p}} = +\infty$ and $\lim_{u_{ij}\to 0^+}\left(\gamma_j u_{ij}^{1-p}\ln u_{ij}\right) = 0$.

Taking into account (a) that $f(0) > 0$ and $f(\hat{u}_{ij}) < 0$, (b) the continuity of $f(u_{ij})$ and (c) the Bolzano's theorem, there is at least one $u_{ij}^{\{1\}} \in (0, \hat{u}_{ij}) : f(u_{ij}^{\{1\}}) = 0$. Moreover, based on Proposition 3, $\frac{\partial f(u_{ij})}{\partial u_{ij}} < 0$ for $u_{ij} \in (0, \hat{u}_{ij})$, thus $f(u_{ij})$ is decreasing on $(0, \hat{u}_{ij})$. Therefore, there is exactly one $u_{ij}^{\{1\}} \in (0, \hat{u}_{ij}) : f(u_{ij}^{\{1\}}) = 0$. Similarly, taking into account (a) that $f(\hat{u}_{ij}) < 0$ and $f(1) > 0$, (b) the continuity of $f(u_{ij})$ and (c) the Bolzano's theorem, there is at least one $u_{ij}^{\{2\}} \in (\hat{u}_{ij}, 1) : f(u_{ij}^{\{2\}}) = 0$. Moreover, based on Proposition 3, it is $\frac{\partial f(u_{ij})}{\partial u_{ij}} > 0$ for $u_{ij} \in (\hat{u}_{ij}, 1)$, thus $f(u_{ij})$ is increasing on $(\hat{u}_{ij}, 1)$. Therefore, there is exactly one $u_{ij}^{\{2\}} \in (\hat{u}_{ij}, 1) : f(u_{ij}^{\{2\}}) = 0$. Consequently, there are exactly two $u_{ij}^{\{1\}}, u_{ij}^{\{2\}} \in (0, 1)$ such that $f(u_{ij}) = 0$. □

Proof of Proposition 5. Obviously, if $f(u_{ij}) = 0$ has two solutions, then $f(\hat{u}_{ij}) < 0$. From

Proposition 4, it is $u_{ij}^{\{1\}} < \hat{u}_{ij} < u_{ij}^{\{2\}}$. Since $\frac{\partial f(u_{ij})}{\partial u_{ij}} \leq 0$ for $u_{ij} \in (0, \hat{u}_{ij}]$ and $\frac{\partial f(u_{ij})}{\partial u_{ij}} \geq 0$ for $u_{ij} \in [\hat{u}_{ij}, +\infty)$ (proof of Proposition 3), it turns out that $f(u_{ij})$ is decreasing for $u_{ij} \in (0, \hat{u}_{ij}]$ and increasing for $u_{ij} \in [\hat{u}_{ij}, +\infty)$. In addition, it can be easily verified that $f(0) \geq 0$ and $f(+\infty) \geq 0$. Taking into account these facts, the continuity of f and the fact that $f(u_{ij}^{\{1\}}) = f(u_{ij}^{\{2\}}) = 0$, it follows that $f(u_{ij})$ is positive for $u_{ij} \in (0, u_{ij}^{\{1\}}) \cup (u_{ij}^{\{2\}}, +\infty)$ and negative for $u_{ij} \in (u_{ij}^{\{1\}}, u_{ij}^{\{2\}})$. Thus, $u_{ij}^{\{2\}}$ is a turning point for $J_{SPCM}(\Theta, U)$ before which $J_{SPCM}(\Theta, U)$ decreases with respect to u_{ij} and after which $J_{SPCM}(\Theta, U)$ increases with respect to u_{ij}. Therefore, $u_{ij}^{\{2\}}$ is a local minimum of $J_{SPCM}(\Theta, U)$, whereas, employing similar reasoning, it turns out that $u_{ij}^{\{1\}}$ is a local maximum of $J_{SPCM}(\Theta, U)$. □

Proof of Proposition 6. Let $J_{SPCM}(\boldsymbol{\theta}_j, u_{ij})$ contain the terms of $J_{SPCM}(\Theta, U)$ that involve $\boldsymbol{\theta}_j$, u_{ij}. According to Propositions 3, 4 and 5, it turns out that if $f(\hat{u}_{ij}) < 0$, then the global minimum of $J_{SPCM}(\boldsymbol{\theta}_j, u_{ij})$ with respect to u_{ij} is $u_{ij}^{\{2\}}$, provided that $J_{SPCM}(\boldsymbol{\theta}_j, u_{ij}^{\{2\}}) < J_{SPCM}(\boldsymbol{\theta}_j, 0)$. However, the latter becomes $u_{ij}^{\{2\}} \left[d_{ij} + \gamma_j \ln u_{ij}^{\{2\}} - \gamma_j + \ +\lambda (u_{ij}^{\{2\}})^{p-1} \right] < 0$ and taking into account that $f(u_{ij}^{\{2\}}) \equiv d_{ij} + \gamma_j \ln u_{ij}^{\{2\}} + \lambda p (u_{ij}^{\{2\}})^{p-1} = 0$, it is equivalent to $u_{ij}^{\{2\}} \left[-\lambda p (u_{ij}^{\{2\}})^{p-1} \ -\gamma_j + \lambda (u_{ij}^{\{2\}})^{p-1} \right] < 0$ or $u_{ij}^{\{2\}} > \left(\frac{\lambda(1-p)}{\gamma_j} \right)^{\frac{1}{1-p}}$. Clearly, in the case where $f(\hat{u}_{ij}) < 0$ and $u_{ij}^{\{2\}} < \left(\frac{\lambda(1-p)}{\gamma_j} \right)^{\frac{1}{1-p}}$, it is $u_{ij} = 0$. Finally, in the case where $f(\hat{u}_{ij}) > 0$, it is $f(u_{ij}) > 0$, for $u_{ij} \in (0, +\infty)$. Thus, $J_{SPCM}(\Theta, U)$ increases with respect to u_{ij} in $(0, +\infty)$ and, as a consequence, its minimum is achieved at $u_{ij} = 0$. □

APPENDIX C. SPCM CONVERGENCE PROPOSITIONS AND PROOFS

Proposition C1: If $K < pe^{2(1-p)}$, then $R_j > 0$.

Proof: Substituting λ from eq. (4.5) into the definition of R_j^2 from eq. (4.7) and after some manipulations, we have

$$R_j^2 = \frac{\gamma_j}{1-p}\left(-\ln\frac{\bar{\gamma}}{\gamma_j} - \ln\frac{K}{e^{2-p}} - p\right)$$

or, since $\frac{\bar{\gamma}}{\gamma_j} < 1$

$$R_j^2 \geq \frac{\gamma_j}{1-p}\left(-\ln\frac{K}{e^{2-p}} - p\right).$$

Straightforward operations show that the positivity of the quantity in parenthesis is equivalent to the hypothesis condition $K < pe^{2(1-p)}$. Q.E.D.

Proposition C2: It is $(\sum_{i=1}^k u_i')^2 \leq \sum_{i=1}^k u_i \sum_{i=1}^k \frac{u_i'^2}{u_i}$, for $u_i, u_i' > 0$, $i = 1, \ldots, k$.

Proof: It is

$$(\sum_{i=1}^k u_i')^2 \leq \sum_{i=1}^k u_i \sum_{i=1}^k \frac{u_i'^2}{u_i} \Leftrightarrow$$

$$\sum_{i=1}^k u_i'^2 + 2\sum_{i=1}^k \sum_{j=i+1}^k u_i' u_j' \leq \sum_{i=1}^k u_i'^2 + \sum_{i=1}^k \sum_{j=1}^k \frac{u_i}{u_j} u_j'^2 \Leftrightarrow$$

$$\sum_{i=1}^k \sum_{j=i+1}^k (\frac{u_i}{u_j} u_j'^2 + \frac{u_j}{u_i} u_i'^2 - 2u_i' u_j') \geq 0 \Leftrightarrow$$

$$\sum_{i=1}^k \sum_{j=i+1}^k \frac{(u_i u_j' - u_j u_i')^2}{u_i u_j} \geq 0,$$

which obviously holds. Q.E.D.

Proposition C3: Let $\mathbf{z}^* = (\mathbf{u}^*, \boldsymbol{\theta}^*) \in S_q$ corresponding to a certain active set X_q. Let also $Y_{\mathbf{z}^*} = Y_{\mathbf{u}} \times Y_{\boldsymbol{\theta}}$ be a set of $(\mathbf{u}, \boldsymbol{\theta})$, such that $Y_{\mathbf{u}} = \{\mathbf{u} \in \mathcal{M} : ||\boldsymbol{\theta}^* - \frac{\sum_{i=1}^k u_i \mathbf{x}_i}{\sum_{i=1}^k u_i}|| < \varepsilon\}$ where $\varepsilon < \frac{1}{2}\sqrt{\frac{(1-p)\gamma}{2}}$ and $Y_{\boldsymbol{\theta}} = \{\boldsymbol{\theta} : \boldsymbol{\theta} = \frac{\sum_{i=1}^k u_i \mathbf{x}_i}{\sum_{i=1}^k u_i}, \mathbf{u} \in Y_{\mathbf{u}}\}$. Then (a) $Y_{\mathbf{z}^*}$ is a convex set and (b) J is a convex function over $Y_{\mathbf{z}^*}$.

Proof: (a) Since the domain $Y_{\mathbf{u}}$ of $\mathbf{u}$ is a cartesian product of closed one-dimensional intervals, it is convex. In addition, the set $Y_{\boldsymbol{\theta}}$ is also convex by its definition. Thus $Y_{\mathbf{z}^*}$ is convex.

(b) We prove that for any $\mathbf{z} \in Y$, it is $\mathbf{z}'^T H_{\mathbf{z}} \mathbf{z}' > 0$, $\forall \mathbf{z}' \in Y$. Following a reasoning similar to

that in Lemma 7, we end up with the following inequality (with corresponds to eq. (4.49))

$$\mathbf{z}'^T H_{\mathbf{z}} \mathbf{z}' \quad \geq \quad 2\sum_{i=1}^{k} u_i ||\boldsymbol{\theta}'||^2 \; - \; 4\sum_{i=1}^{k} u_i' ||\boldsymbol{\theta}'|| (2\varepsilon) \; + \; (1 \; - \; p)\gamma \sum_{i=1}^{k} \frac{u_i'^2}{u_i} \quad \equiv \quad \phi(||\boldsymbol{\theta}'||). \quad \text{(C.1)}$$

Note that the factor 2ε in the right hand side of the above inequality, results from the fact that this is the maximum possible difference between two elements in $Y_{\boldsymbol{\theta}}$. The discriminant of $\phi(||\boldsymbol{\theta}'||)$ is

$$\Delta = 8[8\varepsilon^2 (\sum_{i=1}^{k} u_i')^2 - (1-p)\gamma \sum_{i=1}^{k} u_i^* \sum_{i=1}^{k} \frac{u_i'^2}{u_i^*}]. \quad \text{(C.2)}$$

Proposition C2 and the choice of ε guarantee that Δ is negative, which implies that $\mathbf{z}'^T H_{\mathbf{z}} \mathbf{z}' > 0$ and as a consequence J is convex over $Y_{\mathbf{z}^*}$. Q.E.D.

Proposition C4: A data point $\mathbf{x}$ has $u > 0$ with respect to a cluster C with representative $\boldsymbol{\theta}$ and parameter γ or, equivalently, $f(u) = 0$ has solution(s), if $K \leq \frac{\gamma}{\tilde{\gamma}} p e^{(2-\mu)(1-p)}$, where $\mu = \frac{\|\mathbf{x}-\boldsymbol{\theta}\|^2}{\gamma}$.

Proof: According to eq. (4.7), a data point $\mathbf{x}$ has $u > 0$ if and only if $\|\mathbf{x} - \boldsymbol{\theta}\|^2 \leq R^2 \Leftrightarrow \|\mathbf{x} - \boldsymbol{\theta}\|^2 \leq \frac{\gamma}{1-p}\left(-\ln \frac{\lambda(1-p)}{\gamma} - p\right) \Leftrightarrow \mu \leq \frac{1}{1-p}\left(-\ln \frac{\lambda(1-p)}{\gamma} - p\right)$, which, using eq. (4.5), gives $\mu \leq \frac{1}{1-p}\left(-\ln \frac{K\tilde{\gamma}}{pe^{2-p}\gamma} - p\right) \Leftrightarrow \mu(1-p) \leq -\ln \frac{K\tilde{\gamma}}{p\gamma} + 2 - 2p \Leftrightarrow (2-\mu)(1-p) \geq \ln \frac{K\tilde{\gamma}}{p\gamma} \Leftrightarrow e^{(2-\mu)(1-p)} \geq \frac{K}{p}\frac{\tilde{\gamma}}{\gamma} \Leftrightarrow K \leq \frac{\gamma}{\tilde{\gamma}} p e^{(2-\mu)(1-p)}$. Q.E.D.

Theorem C1 (Leray-Schauder-Tychonoff Fixed point theorem, e.g. [71]): If $X \subset \mathcal{R}^r$ is nonempty, convex and compact and if $Z : X \to X$ is a continuous function, there exists $\mathbf{x}^* \in X$, such that $Z(\mathbf{x}^*) = \mathbf{x}^*$ (fixed point).

BIBLIOGRAPHY

[1] R. J. Schalkoff, “Pattern Recognition: Statistical, Structural and Neural Approaches”, *Wiley, New York, NY*, 1992.

[2] H. Bunke, “Structural and Syntactic Pattern Recognition”, *Chen, Pau* & *Wang (Eds.) Handbook of pattern recognition* & *computer vision*, World Scientific, 1993.

[3] P. Sneath and R. Sokal, “Numerical Taxonomy: The Principles and Practice of Numerical Classification”, *London, UK: Freeman*, 1973.

[4] J. Ward, “Hierarchical Grouping to Optimize an Objective Function”, *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 236-244, 1963.

[5] G. Karypis and E. -H. Han and V. Kumar, “Chameleon: Hierarchical Clustering Using Dynamic Modeling”, *Journal Computer*, vol. 32, no. 8, pp. 68-75, 1999.

[6] J. A. Hartigan and M. A. Wong, “Algorithm AS 136: A k-Means Clustering Algorithm”, *Journal of the Royal Statistical Society*, vol. 28, pp. 100-108, 1979.

[7] J. C. Bezdek, “A Convergence Theorem for the Fuzzy Isodata Clustering Algorithms”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 2, pp. 1-8, 1980

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

[9] R. Krishnapuram and J. M. Keller, “A Possibilistic Approach to Clustering”, *IEEE Transactions on Fuzzy Systems*, vol. 1, pp. 98-110, 1993.

[10] R. Krishnapuram and J. M. Keller, “The Possibilistic C-Means Algorithm: Insights and Recommendations”, *IEEE Transactions on Fuzzy Systems*, vol. 4, pp. 385-393, 1996.

[11] M. Ester and H. -P. Kriegel and J. Sander and X. Xu, “A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise”, *in Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining*, pp. 226-231, 1996.

[12] X. Xu, M. Ester and H.-P. Kriegal and J. Sabder, “A Distribution Based Clustering Algorithm for Mining in Large Spatial Databases”, *in Proceedings of 14th International Conference on Data Engineering (ICDE)*, 1998.

[13] Hinneburg and D. Keim, “An Efficient Approach to Clustering Large Multimedia Databases with Noise” , *in Proceedings of 4th International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 58-65, 1998.

[14] K. Fukunaga and L. D. Hostetler, “The Estimation of the Gradient of a Density Function, with Applications in Pattern Recognition”, *IEEE Transactions on Information Theory*, vol. 21, no. 1 pp. 32-40, 1975.

[15] R. Agrawal and J. Gehrke and D. Gunopoulos and P. Raghavan, “Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications”, *in Proceedings ACM SIGMOD International Conference Management of Data*, pp. 94-105, 1998.

[16] E. Elhamifar and R. Vidar, “Sparse Subspace Clustering: Algorithm, Theory, and Application”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 11, pp. 2765-2781, 2013.

[17] A. L. N. Fred and A. K. Jain, “Combining Multiple Clusterings Using Evidence Accumulation”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 6, pp. 835-850, 2005.

[18] S. Theodoridis and K. Koutroumbas, “Pattern Recognition”, 4th edition, *Academic Press*, 2009.

[19] N. R. Pal and K. Pal and J. M. Keller and J. C. Bezdek, “A Possibilistic Fuzzy C-Means Clustering Algorithm”, *IEEE Transactions on Fuzzy Systems*, vol. 13, pp. 517-530, 2005

[20] M. S. Yang and K. L. Wu, “Unsupervised Possibilistic Clustering”, *Pattern Recognition*, vol. 39, pp. 5-21, 2006.

[21] K. Treerattanapitak and C. Jaruskulchai, “Possibilistic Exponential Fuzzy Clustering”, *Journal of Computer Science and Technology*, vol. 28, pp. 311-321, 2013.

[22] X. L. Xie and G. Beni, “A Validity Measure for Fuzzy Clustering”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, pp. 841-847, 1991.

[23] J-S. Zhang and Y-W. Leung, “Improved Possibilistic C-Means Clustering Algorithms”, *IEEE Trans. on Fuzzy Systems*, vol. 12, No. 2, pp. 209-217, 2004.

[24] M. Barni and V. Cappellini and A. Mecocci, “Comments on “A Possibilistic Approach to Clustering””, *IEEE Transactions on Fuzzy Systems*, vol. 4, pp. 393-396, 1996

[25] H. Timm, C. Borgelt, C. Döring, R. Kruse, “An Extension to Possibilistic Fuzzy Cluster Analysis”, *Fuzzy Sets and Systems*, vol. 147, pp. 3-16, 2004, doi: 10.1016/j.fss.2003.11.009

[26] F. Masulli and S. Rovetta, “Soft Transition from Probabilistic to Possibilistic Fuzzy Clustering”, *IEEE Transactions on Fuzzy Systems*, vol. 14, pp. 516-527, 2006.

[27] Z. Xie and S. Wang and F. L. Chung, “An Enhanced Possibilistic C-Means Clustering Algorithm EPCM”, *Soft Computing, Springer*, vol. 12, pp. 593-611, 2008.

[28] X. Wu, B. Wu, J. Sun, H. Fu, “Unsupervised Possibilistic Fuzzy Clustering”, *Journal of Information & Computational Science*, vol. 5, pp. 1075-1080, 2010.

[29] M. S. Yang and K. L. Wu, “A Similarity-Based Robust Clustering Method”, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 26, no. 4, pp. 434-448, 2004.

[30] M. S. Yang and C. Y. Lai, “A Robust Automatic Merging Possibilistic Clustering Method”, *IEEE Transactions on Fuzzy Systems*, vol. 19, pp. 26-41, 2011.

[31] Y. Y. Liao and K. X. Jia and Z. S. He, “Similarity Measure based Robust Possibilistic C-means Clustering Algorithms”, *Journal of Convergence Information Technology*, vol. 6, no. 12, pp. 129-138, 2011.

[32] D. Landgrebe, “Hyperspectral Image Data Analysis as a High Dimensional Signal Processing Problem,” (Invited), *Special Issue of the IEEE Signal Processing Magazine*, vol. 19, no. 1, pp. 17-28, Jan. 2002.

[33] G. A. Shaw and H. K. Burke, “Spectral Imaging for Remote Sensing,” *Journal of Lincoln Laboratory*, vol. 14, no. 1, pp. 3-28, 2003.

[34] C. I. Chang, “Hyperspectral Imaging: Techniques for Spectral Detection and Classification,” *Springer US, Kluwer Academic/Plenum Publishers, New York*, 2003.

[35] C. I. Chang, “Hyperspectral Data Processing: Algorithm Design and Analysis,” *John Wiley & Sons*, Inc., 2013.

[36] M. Parente and A. Zymnis, “Statistical Clustering and Mineral Spectral Unmixing in Aviris Hyperspectral Image of Cuprite, NV,” *CS2009 report*, Dec 2005.

[37] A. Aiyer and K. Pyun and Y. Huang and D. O’Brien and R.M. Gray, “Lloyd Clustering of Gauss Mixture Models for Image Compression and Classification,” *in Image Communication*, vol. 20, pp. 459-485, 2005.

[38] H. Alhichri and N. Ammour and N. Alajlan and Y. Bazi, “Clustering of Hyperspectral Images with an Ensemble Method Based on Fuzzy C-Means and Markov Random Fields,” *Springer Arabian Journal for Science and Engineering*, vol. 39, pp. 3747-3757, 2014.

[39] X. Sun and L. Yang and B. Zhang and L. Gao and L. Zhang, “Hyperspectral Image Clustering Method Based on Artificial Bee Colony Algorithm,” *Sixth International Conference on Advanced Computational Intelligence*, China, Oct 19-21, 2013.

[40] X. Sun and L. Yang and L. Gao and B. Zhang and S. Li and J. Li, “Hyperspectral Image Clustering Method based on Artificial Bee Colony Algorithm and Markov Random Fields,” *SPIE Journal of Applied Remote Sensing*, vol. 9, no. 1 , pp. 095047(1)-095047(19) , 2015.

[41] S. Bandyopadhyay and U. Maulik and A. Mukhopadhyay, “Multiobjective Genetic Clustering for Pixel Classification in Remote Sensing Imagery,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 45, no. 5, pp. 1506-1511, May 2007.

[42] K. Deb, “Multi-objective Optimization Using Evolutionary Algorithms”, *Chichester, U.K.: Wiley*, 2001.

[43] H. Gholizadeh and M. J. V. Zoej and B. Mojaradi, “A Novel Hyperspectral Image Clustering Method Based on Spectral Unmixing,” *IEEE Aerospace Conference*, Mar 3-10, 2012.

[44] H. E. Pour and S. Homayouni, “Clustering of Hyperspectral Image using Fuzzy C Spectral Similarity Measures,” *Spiral Journal of Computations and Materials in Civil Engineering*, vol. 1, no. 1, pp. 49-56, 2016.

[45] T. N. Tran and R. Wehrens and L. M.C. Buydens, “KNN-kernel Density-based Clustering for High-dimensional Multivariate Data,” *Elsevier Journal of Computational Statistics & Data Analysis*, vol. 51, pp. 513-525, 2006.

[46] C. Cariou and K. Chehdi, “Unsupervised Nearest Neighbors Clustering with Application to Hyperspectral Images,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 6, pp. 1105-1116, 2014.

[47] G. Celuex and J. Diebolt, “The SEM Algorithm: A Probabilistic Teacher Algorithm Derived from the EM Algorithm for the Mixture Problem”, *Computational Statistics Quarterly*, vol. 2, pp. 73–82, 1985.

[48] W. F. Basener and A. Castrodad and D. Messinger and J. Mahle and P. Prue, “A Dynamical Systems Algorithm for Clustering in Hyperspectral Imagery,” *presented at SPIE Algorithms & Technologies for Multispectral, Hyperspectral, and Ultraspectral Imaging XIV*, 2008.

[49] S. Lee and M. M. Crawford, “Hierarchical Clustering Approach for Unsupervised Image Classification of Hyperspectral Data,” *Proceedings of IEEE International Geoscience and Remote Sensing Symposium*, Sept 20-24, 2004.

[50] A. R.S. Marcal and L. Castro, “Hierarchical Clustering of Multispectral Images Using Combined Spectral and Spatial Criteria,” *IEEE Geoscience and Remote Sensing Letters*, vol. 2, no. 1, pp. 59-63, 2005.

[51] N. Gillis and D. Kuang and H. Park, “Hierarchical Clustering of Hyperspectral Images Using Rank-Two Nonnegative Matrix Factorization,” *IEEE Transactions on Geoscience and Remote Sensing*, doi: 10.1109/TGRS.2014.2352857, 2015.

[52] H. Zhang and H. Zhai and W. Liao and L. Cao and L. Zhang and A. Pizurica, “Hyperspectral Image Kernel Sparse Subspace Clustering with Spatial Max Pooling Operation,” *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLI-B3, pp. 945-948, 2016.

[53] S. D. Xenaki and K. D. Koutroumbas and A. A. Rontogiannis, “Adaptive Possibilistic Clustering”, *IEEE International Symposium on Signal Processing and Information Technology*, pp. 422-427, Dec 2013.

[54] S. D. Xenaki and 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.

[55] S. D. Xenaki and K. D. Koutroumbas and A. A. Rontogiannis, “Sparse Adaptive Possibilistic Clustering”, *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 3072-3076, Florence 2014.

[56] S. D. Xenaki and K. D. Koutroumbas and A. A. Rontogiannis, “Sparsity-aware Possibilistic Clustering Algorithms”, *IEEE Transactions on Fuzzy Systems*, vol. 24, no. 6, pp. 1611-1626, 2016.

[57] K. D. Koutroumbas and S. D. Xenaki and A. A. Rontogiannis, “On the Convergence of the Sparse Possibilistic C-Means Algorithm”, *IEEE Transactions on Fuzzy Systems*, 2017, to appear. DOI: 10.1109/TFUZZ.2017.2659739

[58] S. D. Xenaki and K. D. Koutroumbas and A. A. Rontogiannis, “Sequential Sparse Adaptive Possibilistic Clustering”, *SETN 2014: Artificial Intelligence: Methods and Applications*, pp. 29-42, 2014.

[59] S. D. Xenaki and K. D. Koutroumbas and A. A. Rontogiannis and O. A. Sykioti, “A Layered Sparse Adaptive Possibilistic Approach for Hyperspectral Image Clustering”, *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pp. 2890-2893, 2014.

[60] K. -C. Wong, “A Short Survey on Data Clustering Algorithms”, *2nd International Conference on Soft Computing and Machine Intelligence (ISCMI)*, pp. 64-68, 2015.

[61] S. D. Xenaki and K. D. Koutroumbas and A. A. Rontogiannis, “Hyperspectral Image Clustering Using a Novel Efficient Online Possibilistic Algorithm”, *24th European Signal Processing Conference (EUSIPCO)*, pp. 2020-2024, 2016.

[62] S. D. Xenaki and K. D. Koutroumbas and A. A. Rontogiannis and O. A. Sykioti, “A New Sparsity-Aware Feature Selection Method for Hyperspectral Image Clustering”, *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pp. 445-448, 2015.

[63] G. J. McLachlan and K. E. Basford, “Mixture Models: Inference and Applications to Clustering”, *New York: Marcel Dekker*, 1988.

[64] S. Z. Selim and M. A. Ismail, “K-means-type Algorithms: a Generalized Convergence Theorem and Characterization of Local Optimality”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, no. 1, pp. 81-87, 1984.

[65] D. Pelleg and A. W. Moore, “X-means: Extending K-means with Efficient Estimation of the Number of Clusters”, *in Proceedings of the 17th International Conference on Machine Learning*, pp. 727-734, 2000.

[66] G. Hamerly and C. Elkan, “Learning the K in K-Means”, *In Neural Information Processing Systems, MIT Press*, 2003.

[67] F. Hoppner, “Fuzzy Cluster Analysis: Methods for Classification, Data Analysis and Image Recognition”, *John Wiley & Sons*, 1999.

[68] W. Zangwill, “Nonlinear Programming: A Unified Approach”, *Englewood Cliffs, NJ: Prentice-Hall*, ch. 4, 1969.

[69] M. S. Yang and K. L. Wu and J. Yu, “Alpha-cut Implemented Fuzzy Clustering Algorithms and Switching Regressions”, *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 38, pp. 588-603, 2008.

[70] N. R. Pal and K. Pal and J. C. Bezdek, “A Mixed C-Means Clustering Model”, *in Proceedings of the IEEE International Conference on Fuzzy Systems*, pp. 11-21, Spain, 1997.

[71] D. P. Bertsekas and J. N. Tsitsiklis, “Parallel and Distributed Computation”, *Prentice-Hall*, 1989.

[72] J. Zhou and L. Kao and N. Yang, “On the convergence of some possibilistic clustering algorithms”, *Fuzzy Optimization and Decision Making*, vol. 12, pp. 415-432, 2013.

[73] UCI Library database, http://archive.ics.uci.edu/ml/datasets.html

[74] P. A. Forero and V. Kekatos and G. B. Giannakis, “Robust Clustering Using Outlier-Sparsity Regularization”, *IEEE Transactions on Signal Processing*, vol. 60, pp. 4163-4177, 2012.

[75] R. Inokuchi and S. Miyamoto, “Sparse Possibilistic Clustering with L1 Regularization”, *IEEE International Conference on Granular Computing*, pp. 442-445, 2013.

[76] Y. Hamasuna and Y. Endo, “On Sparse Possibilistic Clustering with Crispness — Classification Function and Sequential Extraction”, *Soft Computing and Intelligent Systems (SCIS) and 13th International Symposium on Advanced Intelligent Systems (ISIS)*, pp. 1801-1806, 2012.

[77] G. Corliss, “Which Root Does the Bisection Algorithm Find?”, *Siam Review*, vol. 19, pp. 325-327, 1977.

[78] A. S. Householder, “The Numerical Treatment of a Single Nonlinear Equation”, *McGraw-Hill*, New York, 1970.

[79] R.R. Meyer, “Sufficient Conditions for the Convergence of Monotonic Mathematical Programming Algorithms”, *Journal of Computer and System Sciences*, vol. 12, pp. 108-121, 1976.

[80] O. Egecioglu and B. Kalantari, “Approximating the Diameter of a Set of Points in the Euclidean Space”, *Information Processing Letters*, vol. 32, pp. 205-211, 1989.

[81] B. Mirkin, “Clustering for Data Mining: A Data Recovery Approach”, *Chapman Hall*, London, 2005.

[82] E. Hullermeier and M. Rifqi and S. Henzgen and R. Senge, “Comparing Fuzzy Partitions: A Generalization of the Rand Index and Related Measures”, *IEEE Transactions on Fuzzy Systems*, vol. 20, pp. 546-556, 2012.

[83] P. Franti and O. Virmajoki, “Iterative Shrinking Method for Clustering Problems”, *Pattern Recognition*, vol. 39, no. 5, pp.761-765, 2006.

[84] C. Rodarmel and J. Shan, “Principal Component Analysis for Hyperspectral Image Classification”, *Surveying and Land Information Systems*, vol. 62, no. 2, pp.115-122, 2002.

[85] S. Zhong, “Efficient Online Spherical k-means Clustering”, *IEEE International Joint Conference on Neural Networks (IJCNN)*, vol. 5, pp. 3180-3185, 2005.

[86] W. Barbakh and C. Fyfe, “Online Clustering Algorithms”, *International Journal of Neural Systems (IJNS)*, vol. 18, pp. 185-194 , 2008.

[87] A. Choromanska and C. Monteleoni, “Online Clustering with Experts”, *Journal of Machine Learning Research*, pp. 1-18, 2011.

[88] Video data set from static camera, https://www.youtube.com/watch?v=z7NpkD CDYjg

[89] Salinas Valley, California HSI data set, http://www.ehu.eus/ccwintco/index.php?title=Hyperspectral_Remote_Sensing_Scenes

[90] Washington DC Mall HSI data set, https://engineering.purdue.edu/ landgreb/H yperspectral.Ex.html

[91] F. Schmidt, A. Schmidt, E. Treandguier, M. Guiheneuf, S. Moussaoui and N. Dobigeon, “Implementation Strategies for Hyperspectral Unmixing using Bayesian Source Separation”, *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, no. 11, pp. 4003-4013, 2010.

[92] K. E. Themelis, F. Schmidt, O. Sykioti, A. A. Rontogiannis, K. Koutroumbas and I. Daglis, “On the Unmixing of MeX/OMEGA Hyperspectral Data”, *Planetary and Space Science, Elsevier*, vol. 68, issue 1, pp.34-41, Aug. 2012

[93] K. Themelis and A. A. Rontogiannis, “A Soft Constrained MAP Estimator for Supervised Hyperspectral Signal Unmixing”, *In Proceedings of the 16th European Signal Processing Conference (EUSIPCO)*, Lausanne, Aug. 2008.

[94] J. A. Benediktsson, J. A. Palmason and J. R. Sveinsson, “Classification of Hyperspectral Data from Urban Areas based on Extended Morphological Profiles”, *IEEE Transactions Geoscience and Remote Sensing*, vol. 43, pp. 480-491, 2005.

[95] R. Tibshirani, “Regression Shrinkage and Selection via the lasso”, *Journal of the Royal Statistical Society*, vol. 58, no. 1, pp. 267-288, 1996.

[96] K. E. Themelis, A. A. Rontogiannis and K. D. Koutroumbas, “A Novel Hierarchical Bayesian Approach for Sparse Semisupervised Hyperspectral Unmixing”, *IEEE Transactions on Signal Processing*, vol. 60, no. 2, pp. 585-599, 2012.