A skewness-based clustering method

Scuola di Dottorato in Scienze Statistiche
Dottorato di Ricerca in Statistica Metodologica
XXX Ciclo

Candidate
Luca Acquafredda
ID number 889680

Thesis Advisor
Prof. Marco Alfò

A thesis submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Statistics
October 2017

A tutti i Dottorandi, la cui presenza mi ha alleggerito questo lungo periodo di lavoro, e in particolare ai "compagni di merende", in ordine alfabetico, e non d'affetto, Francesco, Manuel ed i due "Marchi". Ma soprattutto a un altro Marco, i cui insegnamenti sono sempre andati, e continuano ad andare, ben oltre l'accademico.

Contents

Contents iii

List of figures vii

List of tables ix

Introduction 1

1 Clustering, definition problems and the issue of overlapping clusters 6

1.1 The notion of cluster 6

1.2 Notation 7

1.3 Overlapping clusters 8

1.4 Overlapping regions and a skewness-based proposal 11

2 Prototype-based methods 16

2.1 *Virtual* point prototype-based methods 17

2.1.1 *K-means* 18

2.1.1.1 Variants of the *K-means* method 20

2.1.1.2 Drawbacks of the *K-means* 21

2.1.2 The *Fuzzy C-means* algorithm 23

2.1.2.1 Drawbacks of the *Fuzzy C-means* 28

2.2 *Actual* data point prototype-based methods 29

2.2.1 *Partitioning Around Medoids* (*PAM*) 29

2.2.2 The evolution of *PAM*: *CLARA* and *CLARANS* 32

2.3 Open issues in point prototype-based methods 32

3 Clustering and Finite Mixture Models (with Gaussian kernel) 35

3.1 Mixture Models 35

3.1.1 A brief history 36

3.1.2 Mixture Models: some basic definitions 37

3.2 Mixture Models for clustering 38

3.3 EM algorithm for Mixture Models 42

3.3.1 An early version of the EM algorithm 42

3.3.2 EM algorithm as a solution to incomplete data problems 43

3.4 Gaussian Mixture Models 47

3.4.1 Parametric Mixture Models 47

3.4.2 The case of Gaussian Mixture Models 50

3.4.3 Features and issues with Gaussian Mixture Models 51

3.4.4 EM algorithm for Gaussian Mixture Models 54

3.5 The *R* package *Mclust* 55

4 Clustering, symmetry based methods 61

4.1 Cluster symmetry and *K-means* 61

4.2 A Symmetry based clustering and *MOO* 66

4.3 A line symmetry based approach 71

4.3.1 Drawbacks of previous approaches 72

4.3.2 The line symmetry based distance 73

4.4 Enhancing point symmetry-based distance 79

5 A skewness-based method for clustering **85**

5.1. The case of overlapping clusters 86

5.2 A skewness function and a related cluster validity index 91

5.3 The skewness based objective function 94

5.4 *SBAM* (*Skewness-Based Allocation Method*) 99

5.5 Beyond the gaussianity 101

6 Simulation studies **104**

6.1 Theoretical and practical tools for simulations 104

6.1.1 A measure of the overlapping degree 105

6.1.2 The *R* package *MixSim*. Overlapping Gaussian clusters generator 109

6.1.3 Graphical examples 109

6.2 Measures of performance 114

6.3 Simulation studies scenarios 117

6.4 Simulation study: performance of *SBI* 118

6.5 Simulation study: performance of *SBAM* and *Mclust* 124

7 Analysis of performance on real data **130**

7.1 Application to real data 131

7.1.1 *Iris* Data 131

7.1.2 *Crabs* Data 133

7.1.3 *Wine* Data 134

7.1.4 *Seeds* Data 135

7.1.5 *Ecoli* Data 136

7.2 A proposal for a further development of *SBAM* 137

7.3 Concluding remarks 139

8 Concluding remarks **141**

8.1 Concluding remarks on *SBI* and *SBAM* 141

8.2 Drawbacks and further directions 142

Bibliography **144**

List of Figures

1.1 Two clusters with increasing degree of overlap. (a) Well separated clusters, (b) low degree of overlap, (c) medium degree of overlap, (d) high degree of overlap 9

1.2 Bidimensional convex hull 12

1.3 Bidimensional overlapping region 13

2.1 Overlapping on a tail (*true* partition) 22

2.2 Overlapping on a tail (*K-means* partition) 23

4.1 An example of point symmetry distance 63

4.2 (a) The data set contains a combination of two crossed lines. (b) The clustering result achieved by the *K-means* (c) by the *SBKM* algorithm and (d) by the *SBCL* algorithm 65

4.3 An empirical case where point symmetry distance proposed by Su and Chou (2001) may fail 66

4.4 Example of the point symmetry distance 67

4.5 An example for the distance difference symmetry 73

4.6 Violation of the closure property 74

4.7 An example of line symmetry distance 75

4.8 An example for computing line symmetry distance 77

4.9 Example of the new distance based on point symmetry 80

5.1 Non-elliptical clusters 85

5.2 Real partition and partition estimated by *Mclust* 87

5.3 "Broken" clusters in the partition provided by *Mclust* 88

5.4 Partition found by *K-means* 89

6.1 Clustering scenarios for different values of $\bar{\omega}$ and K. (a)-(b) $\bar{\omega} = 0.005$ with $K = 3$; (c)-(d) $\bar{\omega} = 0.005$ with $K = 5$; (e)-(f) $\bar{\omega} = 0.025$ with $K = 3$; (g)-(h) $\bar{\omega} = 0.025$ with $K = 5$; (i)-(l) $\bar{\omega} = 0.05$ with $K = 3$; (m)-(n) $\bar{\omega} = 0.05$ with $K = 5$ 111

7.1 Bidimensional profiles of data set *Iris* (1935), where X axis and Y axis are, respectively: (a) sepal length, sepal width (b) sepal length, petal length (c) sepal length, petal width (d) sepal width, petal length (e) sepal width, petal width (f) petal length, petal width. 132

List of Tables

3.1 Parameterizations of the covariance matrix Σ_k currently available in *Mclust* 58

4.1 Rankings (in brackets) for algorithms VAMOSA, MOCK, VGAPS and GCUK over 13 datasets, based on the *MS* value obtained 71

4.2 Median values of adjusted Rand index for artificial and real data sets 79

4.3 Rankings for *K-means*, *GAPS*, *GAnPS*, *EM* and *AL* over 21 data sets, based on the *Minkowski Score* (MS). The best MS values are marked in bold 83

6.1 16 scenarios depending on $\bar{\omega}, K, D$. A comparison of *Mclust* and *Kmeans* according to SBI. "Sc" stands for "Scenarios" 121

6.2 16 scenarios depending on $\bar{\omega}, K, D$. Performance of *Mclust* conditionally to SBI_{Mc} and SBI_t. "Sc" stands for "Scenarios" 121

6.3 Performances of *Mclust* and *Sbam* in terms of $ACAE$ and $ARAND$ 126

6.4 Performances of *Mclust* and *Sbam* in terms of $ACAE$ and SBI 127

7.1 Performance of *Mclust* and *Sbam* in real data examples without implementing SBI_{-d} 139

7.2 Performance of *Mclust* and *Sbam* in real data examples implementing SBI_{-d} 139

Introduction

Partitive clustering methods represent one of the earlier and most famous sets of strategy in the field of clustering. The name comes from their main feature: all these methods start from an initial partition and modify it at every step of the process according to a known criterion, until a given convergence rule is satisfied. In other words, as pointed out by Äyrämö and Kärkkäinen (2006), they work essentially as iterative allocation algorithms. In this framework, we do not only focus on "canonical" approaches such as *K-means* and *fuzzy C-means*, but discuss some recent symmetry-based partitive clustering methods, mostly developed in the context of computer science and engineering. As it will be shown, these approaches seem to provide encouraging results, especially in the field of image recognition and some related applications, and for this reason, they represent a starting point for our work.

In this respect, we are particularly interested in the case of overlapping clusters. As we will clarify, this case may represent a critical aspect for most clustering methods we have considered. In particular, we started our analysis by noting that, in a case of high-dimensional data with overlapping clusters, it may be difficult to choose the component-specific distributions, and no graphical device can help us. So, we decided to investigate non parametric approaches to clustering. In this framework, we focused on the case of clusters with elliptical shapes, and in Gaussian mixtures as a special case. Then, we realized that for elliptical shapes the symmetry could be a "natural" choice. So, we searched for such clustering approaches, and we found the symmetry-based methods cited above. But, surprisingly, none of them was intended to focus on elliptical clusters, since their aim is essentially at handling image recognition of different symmetric shapes. So, we decided to discuss this issue, and to test whether a suitable function of symmetry could improve clustering results in the case of elliptical overlapping clusters.

Since we are interested in elliptical shapes, from a clustering point of view, another broad subject that we will discuss is the Gaussian mixture model. This approach, whose starting point is commonly identified in Pearson (1894), has known an increasing interest, especially from the second half of the past century, due essentially to the EM algorithm, see Dempster, Laird and Rubin (1977). In this context, our interest is in the EM-based *Mclust* algorithm from the *R* library $mclust$, see Fraley and Raftery (1999).

Thus, our work addresses both of these topics, partitive clustering methods (with a focus on the symmetry-based approach) and Gaussian model-based clustering.

The main reason of such a choice, that is to address two partially different subjects, derives from the essential features of our proposal: a symmetry-based partitive method which is intended to deal with elliptical clusters (with Gaussian being a special case). In this sense, we provide an evaluation of our clustering performances by proposing a comparison with the Gaussian mixture model implemented in the *Mclust* library. This is surely a challenging task, since this method has home-court advantage in the case of Gaussian clusters. In this framework, as pointed out before, we are mainly interested in the case of overlapping clusters. In this sense, a starting point for our work was the assumption that $Mclust$ (also in its "natural" framework, that is Gaussian mixtures) could have problems in centroid estimation when clusters are highly overlapped. Quite obviously, this drawback could be related to its dependency on the mutivariate Gaussian density. So, we searched for a non parametric skewness-based method, which could be appropriate for elliptical distributions (including Gaussian) in the case of overlapping clusters. This was exactly the framework of the proposed $Sbam$ (*Skewness-Based Allocation Method*) algorithm.

The outline of this work is the following.

In Chapter 1, we briefly present the framework of clustering and introduce some related notions and definitions in a basic way. We discuss the main issue addressed in our work, namely the case of overlapping clusters (with some graphical representations), with a brief view on the possible consequences of such a case on the clustering results. Finally, we introduce the related concept of intersecting areas (which we use to provide a definition of the overlapping regions) and the idea of our proposal: to develop a skewness-based clustering method dealing with elliptical overlapping clusters.

Chapter 2 deals with most used partitive clustering methods, namely the *K-means* in the versions of Forgy (1965) and MacQueen (1967), the *Fuzzy C-means*, see Dunn (1973) and Bezdek (1973), and the *K-medoids* methods, e.g. *PAM* and *CLARA*, see Kaufman and Rousseeuw (1987,1990) respectively and *CLARANS*, see e.g. Ng and Han (2002). In this framework, the concept of point prototype-based clustering has a key role. According to Xiao and Yu (2012) "partitional clustering algorithms suppose that the data set can be represented by a set of prototypes, therefore is also called prototype-based clustering method ... According to different definitions of prototypes, prototype-based clustering methods can be widely categorized into two groups: point-prototype-based clustering algorithms and prototype-based clustering algorithms using non-point prototypes, such as line, hyperplane, and hypersphere, generally called non-point-prototype-based clustering algorithms". Essentially point-prototype-based clustering defines cluster as a set represented by a

point in the space of the observational features (e.g. mean or median or medoid). All of the methods considered in this Chapter are presented as point prototype-based approaches, with the further distinction into two types: *virtual* point prototype clustering and *actual* data point prototype clustering. Roughly speaking, *virtual* point prototype clustering do not belong to the original dataset (such as mean), while *actual* data point do (e.g. medoids). So, according to this formulation, we discuss the *K-means* and the *Fuzzy C-means* as *virtual* point prototype methods, while the *K-medoid* approaches are interpreted as *actual* point prototype methods.

In Chapter 3, we discuss the model based approach to clustering, focusing on Gaussian Finite Mixture Models. To this end, a first subsection is dedicated to Finite Mixture Models, with a brief history and some basic definitions. The second paragraph discusses Finite Mixture Models in a clustering framework; a further section deals with Finite Mixture Models and the related maximum likelihood approach to parameter estimation. In this context, we consider the EM (*Expectation-Maximisation*) algorithm to estimate mixture parameters. Finally, we discuss a specific implementation of the EM algorithm for the Gaussian case, included in the *R* $Mclust$ library, looking to constraints on the component-specific covariance matrices. This last subsection is relevant, because $Mclust$ is one of the reference methods for clustering based on Gaussian mixtures, and it will be the direct competitor of the method proposed in the following Chapter 5.

In Chapter 4, we present some recent contributes to symmetry-based partitive clustering methods. These have been mostly developed in the context of computer science and engineering. All of them are not involved with specifical statistical hypotheses (e.g. assumptions on model structure), but rather aim at identifying symmetric shaped clusters. So, from a statistical point of view, they do not represent parametric approaches. The aim of this section is to provide a short literature review on the skewness-based approaches and to illustrate some drawbacks connected to the use of symmetry in the clustering framework. This section is relevant for at least two reasons: first, the analysis of the drawbacks related to the use of symmetry functions helped us in the formulation of our proposal. Second, the proposed skewness-based method can be considered as an evolution of the algorithm in Su and Chou (2001), discussed at length in the literature review.

In Chapter 5, we present a novel skewness-based clustering method. A source of inspiration for this method could be found in the papers we have discussed in Chapter 4. The results obtained by those approaches suggest that skewness-based techniques may represent a rapidly increasing and promising field of research. In

particular, they have very good performance when compared to other, recently developed, clustering methods. Nevertheless, there are some relevant differences between the proposed method and the skewness-based approaches presented in Chapter 4. First of all, the field of application: all the clustering techniques discussed have been defined and implemented in a non parametric context, while the proposed method is specifically introduced in a model based clustering framework (we are primarily interested in Gaussian mixtures). Only the case of Gaussian densities is explicitly considered, even though in this Chapter we suggest some extensions of the proposed method to elliptical distributions. This fact leads us to a further, non-negligible, difference: despite the generality of the prevoius approaches (they work well with clusters having a different shape), we focus on a particular cluster shape, the elliptical one, which may be associated to the general class of elliptical distributions. This means that our method could not be appropriate when the target is a different kind of clustering. The main features of the proposed method are explained throughout this Chapter, which is organized as follows. In the first section, we give a look at the case of overlapping clusters, and discuss why the proposed method could be appropriate when other competitors fail. Then, we introduce a skewness function, and a related skewness-based index (SBI), adopted as a cluster validation index. In the third section, we define and discuss the objective function and give a sketch of the corresponding algorithm, followed by some remarks on the differences with respect to other skewness-based methods. In the last section, on the basis of Dvoretzky's Theorem (1961), we provide a further support to the search of elliptical clusters, beyond the assumption of an elliptical distribution, and we suggest a further direction of development for the proposed method.

Chapter 6 is devoted to the empirical evaluation of our proposal. In this sense, we provide an analysis of the clustering performances in two different simulation studies: the first one deals with the skewness-based cluster validation index (SBI), while the second involves a comparison between the function $Mclust$ and the proposed $Sbam$ (*Skewness-Based Allocation Method*). We consider Gaussian mixtures in several different clustering scenarios, where the clustering complexity (associated to the overlapping degree) is under control. To this end, the outline of the Chapter follows: in the first section, we introduce the theorical definition of overlapping clusters degree, on the basis of Maitra and Melnykov (2010), and the $MixSim$ function which generates different Gaussian mixtures according to a value of overlapping degree, as developed in Melnykov, Chen and Maitra (2012). In this context, we also provide some bivariate graphical examples to illustrate the potentials of this generator. In the second part, we introduce the definition of absolute error when estimating the cluster centroids (centroid absolute error,CAE), and we discuss the

performance of the skewness-based cluster validation index (SBI) in different simulation scenarios. Finally, we provide a comparison between $Mclust$ and $Sbam$ in terms of clustering performances, in several different clustering scenarios, with some concluding remarks.

In chapter 7, we discuss some applications of the proposed skewness-based algorithm in a real data framework. In this sense, we stress the fact that a single real data set, although highly representative of some interesting features, is still a single one, thus being less informative when compared to the virtually infinite possibilities provided by simulations. We stress as well the fact that the analysis of real data examples can be instructive in a further sense. For instance, taking real data examples may help analyze the behaviour of the proposed clustering method under a potentially misspecified model, that is when we do not know whether clusters come from a Gaussian mixture. This is the main reason of the Chapter.

In the first section, we briefly introduce the real data sets considered, focusing only on their basic features (names, year of reference and relative sources). In further subsections we provide a more detailed description of the same data sets, togheter with the performances achieved by the $Mclust$ function and the proposed $Sbam$ for each dataset, in terms of CAE, SBI and *adjusted Rand Index*. Then, we propose an extended, albeit absolutely tentative, version of $Sbam$ which we test on the same real data examples. Finally, we conclude with some remarks about the performance of the two clustering methods on the real data considered, and some proposals for further developments.

In Chapter 8 we provide some concluding remarks on the proposed method.

Chapter 1

Some clustering problems and the issue of overlapping clusters

In this section, we briefly introduce the framework of clustering and present some related definitions in a basic way. Then, we move to the main issue addressed in our work, namely the case of overlapping clusters (see below for a graphical representation), with a brief discussion on the possible consequences on the clustering results provided by different approaches. Finally, we introduce the related concept of "overlapping regions" and the idea underlying our proposal: to develop a skewness-based clustering method dealing with overlapping clusters.

1.1 The notion of cluster

There is not a unique problem in the framework of clustering, but many different problems. Roughly speaking, clustering techniques aims at associating group to a set of objects. But what is a cluster? A group of objects that are more similar to one another than to members of other clusters? Or a group of objects that are more dissimilar to members of other clusters than to one another? And what do we mean by similar/dissimilar? These seemingly simple questions lay at the heart of the matter: in fact, an objective and univoque answer does rarely exist. It strongly depends on several factors: the specific field we are involved in, the aim of the researcher, and so on. For instance, if our aim is to group some words to form real and meaningful sentences (clusters), the corresponding definition of cluster will be quite different from that of a doctor who wants to group patients on the basis of their blood pressure levels. Even in the same clustering framework, we could well be interested in different kind of clusters: patients with a similar overall pressure (i.e. similar means) or patients with similar changes in blood pressure (i.e. similar variances).

Furthermore, nothing has yet been said about how many clusters we are looking for in the data. Is this inherent to the nature of data, or it depends rather on the specific interest of the researcher? Probably, the best answer to these questions may

be found in the words of Estivill-Castro (2002): "Do not forget that clusters are, in large part, on the eye of the beholder". In short, these are only a few aspects out of those that help to make clustering a quite challenging and uncertain task.

1.2 Notation

To put it formally, let us consider a dataset consisting of n multidimensional points $\mathbf{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n)$ drawn from a set S and let us suppose that our aim is to build K disjoint subsets based on $\mathbf{X}$, say $S_1, S_2, ..., S_K$, so that for $i \neq j = 1, 2, ..., K$ the following conditions hold:
$S_i \cap S_j = \emptyset$
$S_1 \cup S_2 \cup ... \cup S_K = S$
Therefore, the result will be a partition $\{S_1, S_2, ..., S_K\}$ of the original dataset. The subscript is often referred to as label, so that all the data points belonging to cluster S_k, will have the same label k, and K labels will be available in all. In addition, we represent the cardinality of the k-th cluster as $|S_k|$.

So, assuming $|S_k| > 0$ for every k, and forgetting observations' order within a cluster, for fixed n and K ($n \geqslant K$) we may have $\frac{1}{K!}\sum_{i=0}^{K}(-1)^{K-i}\binom{K}{i}i^n$ different partitions (*Stirling number of the second kind*). But how to decide whether a particular solution is a sensible one? There is, indeed, a plethora of different criterions, each corresponding to different aims and/or contexts of the adopted clustering procedure. As pointed out by Hennig et al. (2016), "the reader should be aware that clustering, or grouping of data, can mean different things in different contexts, as well as in different areas of data analysis. There is no unique definition of what a cluster is, or what the "best" clustering ... should be. Hence, the cornerstone of any rigorous cluster analysis is an appropriate and clear definition of what a "good" clustering is in the specific context". It is quite evident, therefore, that there is no silver bullet when a clustering framework is considered.

It is also quite complex to classify the different paradigms and methods used for clustering; as an index of to topic, note that Murtagh and Kurtz (2016) found more than 404.000 contributions to the clustering literature. Here, we focus only on partitive clustering methods and Gaussian mixture models, see Chapter 2 and 3, respectively. For a short list of the main elements of a cluster analysis it is possibile to cite the following structure from Äyrämö and Kärkkäinen (2006):

1. Data presentation.
2. Choice of objects.

3. Choice of variables.
4. What to cluster: data units or variables.
5. Normalization of variables.
6. Choice of (dis)similarity measures.
7. Choice of clustering criterion (objective function).
8. Choice of missing data strategy.
9. Algorithms and computer implementation (e.g. convergence)
10. Number of clusters.
11. Interpretation of results.

For each of these points (and their combinations) we may choose different strategies, and this partially clarifies the extent of the issue. For a detailed discussion of different clustering paradigms see Hennig et al. (2016).

1.3 Overlapping clusters

Our focus here is on partitive clustering methods and, in particular, on model based clustering via Gaussian mixture models (they will be discussed in Chapter 2 and 3, respectively). Within this framework, the main scope is a sensible estimation of clusters centroids, intuitively defined as the barycenters of the corresponding clusters and obtained as the "central position" of all the points, when all of the coordinate directions are considered.

Generally speaking, one of the main issue in this context, is related to the presence of anomalous data which may alter in some way the structure of the clusters (the underlying distributions in the case of Gaussian mixture models). This type of data is commonly known as outliers or contaminating points. Actually, a cluster may be contaminated in many different ways, each one leading to a different kind of outliers. Thus, there are many possibilities for defining what we mean by outlying point. Very basically put, we can distinguish at least the following:
1. *Extreme values*. Data points which show values much larger (or smaller) with respect to the other ones. Clearly, in a multivariate framework, this may occur relatively to one or more dimensions.
2. *Leverage points*. Points that show an abnormal "behaviour" with respect to the others; just to give an example in the bivariate case, an extreme value showing a departure from the "usual" relation between the two coordinates. In the regression framework, for instance, this can crucially affect the intercept and slope estimate.
3. *Bridge points*. Data points which identify a region of intersection between two or more clusters.

To handle these cases, many different robust clustering methods have been developed, see Chapter 2 for a brief list of some recent approaches.

Our work is not involved with such methods and related problems. Nonetheless, for our purpose the case of *bridge points* is a crucial issue. In fact, we are mostly interested in studying the overlapping clusters, which naturally determine the presence of *bridge points*. In this sense, it is possible to represent at least four different scenarios of increasing overlapping degree, which is depicted in the following Figure:

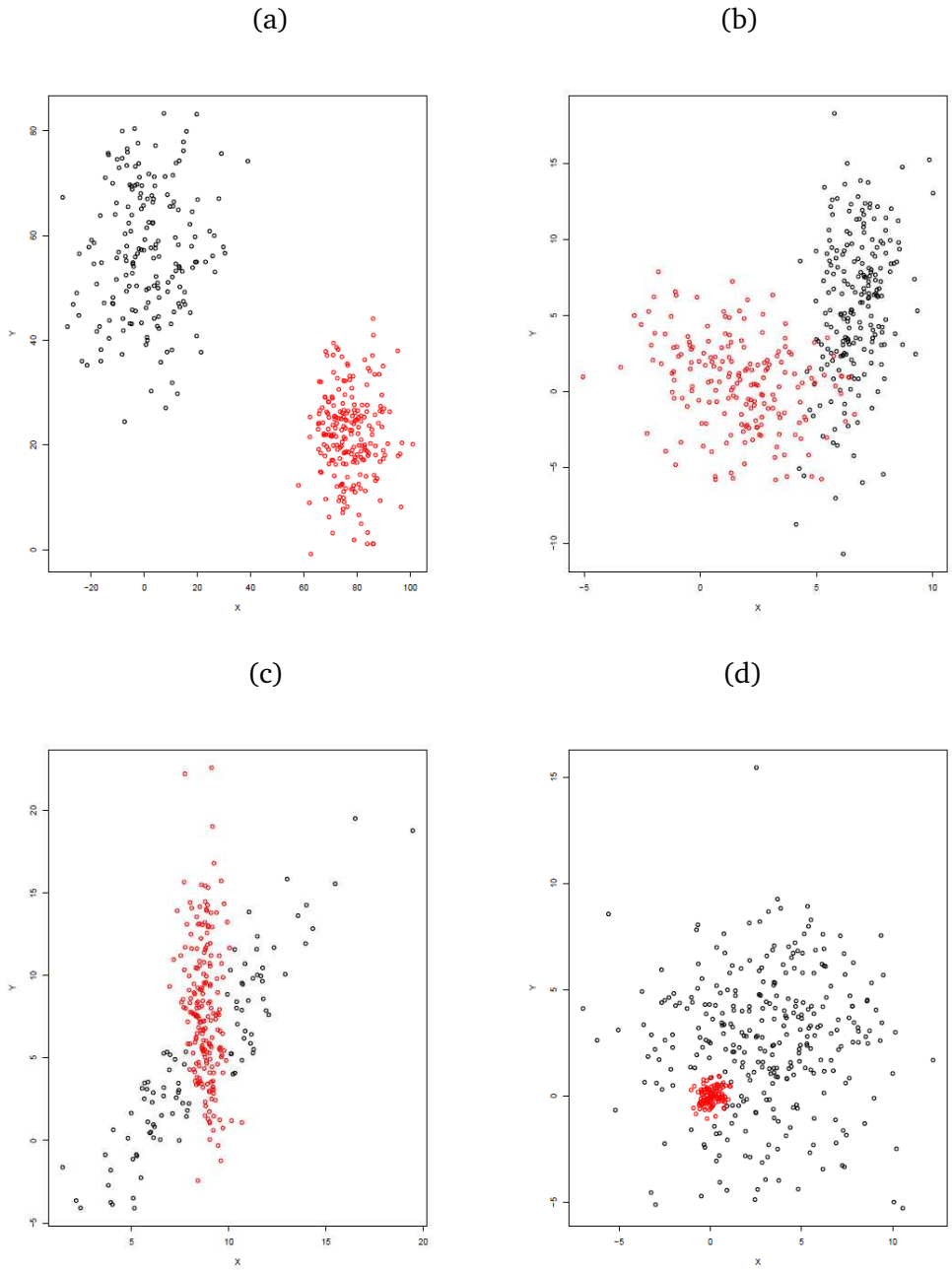


Fig. 1.1: Two clusters with increasing degree of overlap. (a) Well separated clusters, (b) low degree of overlap, (c) medium degree of overlap, (d) high degree of overlap

In the first case, Fig. 1(a), clusters are clearly well separated, and the overlap does not occur, so that a good allocation of points as well as a proper estimates for the centroids are quite naturally obtained.

In the second case, Fig. 1(b), clusters are not well separated, and some overlap does occur in the tail of the "black" cluster. This case is somewhat complex for the estimation of centroids. In fact, if even only few points in the overlapping region were assigned to the red cluster, the estimate of the centroid of the black cluster would be seriously biased. In fact, those few points lying at the extreme tail of the black cluster, are essential to a proper estimate of the corresponding centroid.

In the third case, Fig. 1(c), clusters are even more overlapping. In this case, there is a high risk of misclassification of a relevant number of data points, due to the quite substancial overlapping area. In this sense, it is interesting to stress the difference between case (b) and (c). Let us suppose that, in both cases, all of the points in the overlapping regions are assigned only to the red cluster, and assume that all the other points would be exactly allocated in the corresponding clusters. So, in case (b) we would obtain a severe bias in the centroid estimate of the black cluster but a little rate of misclassification (due to a few *bridge points*), whereas in case (c) we would have the opposite situation: a certain bias (not necessarily negligible, but possibly lower than previous one) in the centroid estimate of the black cluster, but a significant rate of misclassification (due to the overlapping area).

In the last case, Fig. 1(d), we have totally overlapping clusters, a sort of innested clusters (but not in a hierarchical sense). In this case, provided that our clustering method would be able to detect the nested cluster (really not a trivial matter), it is easy to expect a strong bias in the centroid estimates as well as a significant rate of misclassification. This last case, which could seem a somewhat extreme and unrealistic case, can actually be explained in the following way. Let us consider a bidimensional data set with blood pressure and heartbeat measures for two cluster of hypertensive and tachycardic patients. Suppose one of the two clusters (the nested one) is composed by subjects treated with a particular drug against hypertension and tachycardia, while the other cluster is composed by untreated subjects. If the drug had the further feature of reducing the variability in both blood pressure and heart rate, we would be in a scenario quite similar to that shown in Fig. 1(d). In fact, we would have a smaller cluster (due to the reduced variability in both dimensions), and a centroid located down and to the left (due to the lower blood pressure and heartbeat) with respect to the bigger cluster.

It is worth noting that also the other overlapping cases depicted in Figure 1, namely (b) and (c), are liable to analogous explanation. In this sense, in all of the

overlapping cases we have considered, *bridge points* do not really alter the structure of the real clusters involved, but rather reflect a sensible intersection of the clusters. If anything, in these cases *bridge points* alter the structure of the clusters which probably our methods are able to detect. In other words, they must not be necessarily considered as outliers, but they can be seen as "standard" points, determined by an overall overlapping context, depending on different but sensible causes (like those just discussed). From this point of view, with respect to the three type of outliers considered above, only *extreme values* and *leverage points* will be interpreted as outliers, because both alter the real structure we expect to detect in a cluster.

Finally, note that there is a relationship between the overlapping degree and the rate of misclassification depicted in the four sub-figures. In fact, it is even possible to define a sort of clustering complexity degree (related to the lack of separation between clusters) in terms of misclassification probabilities, in a way that will be addressed in Chapter 5, according to Maitra and Melnykov (2010). So, the above considerations on misclassification in case of overlapping clusters will find a natural explanation, which links the complexity of a clustering scenario with its overlapping degree, showing the centrality of this issue.

1.4 Overlapping regions and a skewness-based proposal

Here, we want to point out that, in the above overlapping cases, all the issues related to bias in centroids estimates and misclassification of points can be traced back to the role of the distance that usually involved in partitional clustering methods. In fact, in Fig. 1, (b)-(d), none of the common distance-based methods will be able to assign *bridge points* to the proper cluster; in fact, whatever the type of distance we choose, those points will be allocated to the nearest centroid, which can lead to a wrong assignment (in the case of Fig. 1(b) this is more than a simple possibility). From this point of view, it can be noticed that also Gussian mixture approach to clustering may be interpreted as a distance-based method, since the likelihood value is based on a particular "kernel distance", depending on some parameters we want to estimate. In other words (and roughly speaking) the maximum likelihood approach can be regarded as a distance-based method, which aims at finding centroids minimizing the sum of "kernel distances" of data points to the corresponding centroids. Thus, as we will see in Chapters 5, also the Gussian mixture approach to clustering may suffer from the same limitations in case of overlapping clusters (although

it may outperform other clustering methods). So, once we have chosen a specific metric distance, we can not avoid the potential issue of overlapping clusters and we can only expect that in such cases shapes as well as cluster centroid estimates will not be altered too much.

Note that this is not a trivial matter, as in presence of *bridge points* either partitive clustering methods or Gussian mixture model-based clustering suffer from bias in allocation and centroid estimates (for a brief discussion of these methods see Chapters 2 and 3, respectively). One of the most important clustering approach which can be adopted in such cases is the *Fuzzy C-means*, see Dunn (1973), which will be discribed in Chapter 2.

Here, we would just notice that the solution to the problem of overlapping clusters provided by *Fuzzy C-means* is not the only possible solution. In particular, as we will clarify in Chapter 2, *fuzzy* solutions, even in the best case, are not able to reproduce the *true* underlying partition. To achieve this aim, a clustering method should be able to detect and reproduce the overlapping regions.

Beyond the intuitive meaning of overlapping clusters provided in Fig. 1.1, it is also possible to formalize it. For this purpose, we need first to introduce the concept of intersecting area and the related convex hull of a set. This is, roughly speaking, the smallest convex set containing all the elements of the originary set, see the following figure, which depicts a simple bidimensional example:

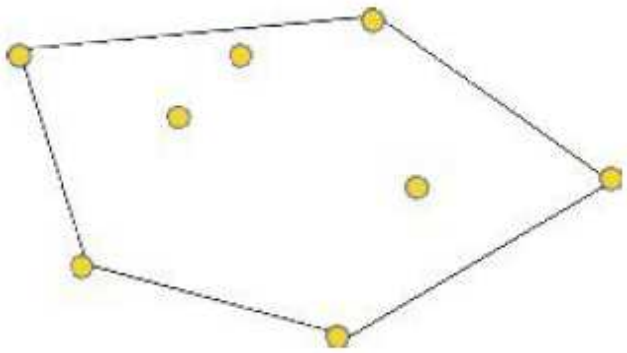


Fig. 1.2: Bidimensional convex hull

As depicted in Fig. 2, the convex hull of the set of yellow points is the set of *all* the points in the pentagon-type area (not only the yellow ones).

Now, let C_{S_k} be the convex hull defined on points belonging to cluster k. Thus, for two clusters, say k and k', the intersecting area $I_{kk'}$ can be defined as the intersection of the convex hulls of the corresponding clusters, such that the same intersection is not a null set, that is

$$I_{kk'} = \left\{ C_{S_k} \cap C_{S_{k'}} \middle| \, C_{S_k} \cap C_{S_{k'}} \neq \emptyset \right\}$$

An intuitive example of this formulation can be found in Fig. 1.1(c), where the graphical intersection between the two clusters is particularly evident. But, clearly the formulation includes all possible cases, a part from those where intersection does not occur, as in Fig. 1.1(a). The same definition has an obvious extension to an arbitrary number of clusters involved in the same intersecting area, i.e. $I_{kk'k''}$ will indicate a three clusters intersection, for distinct values of k, k' and k''.

Now, to introduce the concept of overlapping region, we refer to the aforementioned intersecting area. In fact, the overlapping region is a particular case of intersecting area, when there are points in the region that belong to each of the intersecting clusters, as in the following figure:

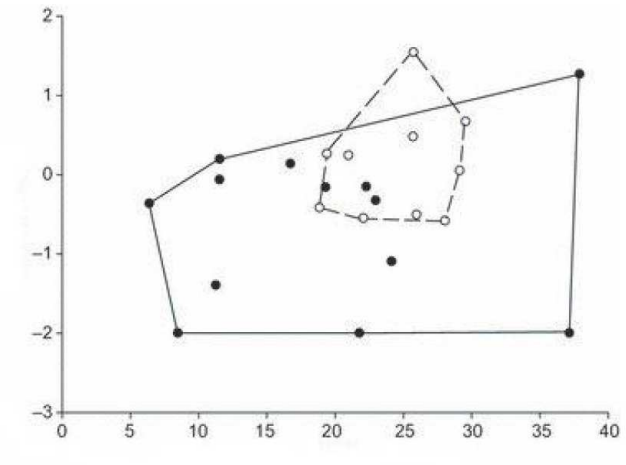


Fig. 1.3. Bidimensional overlapping region

where the dotted lines is used only to distinguish between the two clusters (while theoretically convex hulls can not be represented by such lines). Note that both clusters have at least one element in the intersecting area (empty dots and black dots, respectively), such that this intersecting area is also an overlapping region. Clearly, this is the real partition. If our clustering method misclassified the three black dots in the overlapping region, assigning them to the empty dots cluster, it

would reproduce an intersecting area, because black dots cluster would not have any element in the intersecting area.

To put it formally, in the case of two cluster, say k and k' with corresponding elements $i \in S_k$ and $j \in S_{k'}$, the overlapping region $OR_{kk'}$ can be defined as

$$OR_{kk'} = \{I_{kk'} | \exists \ (i \in S_k, j \in S_{k'}) \in I_{kk'}\}$$

Note that, by this way, we retain the hypothesis of null intersection between clusters considered as set of points, that is $k \neq k' \ \ S_k \cap S_{k'} = \emptyset \ \ k, k' = 1, ..., K$. In fact, the overlapping regions are regions containing points in two or more clusters, where each point belongs to one and only one cluster.

This is the so-called *hard* partition, as opposite to the *fuzzy* assignment procedure, where each point may belong to more than one cluster, as in the *Fuzzy* logic, see Chapter 2 for a discussion. Here, what we intend to stress is the fact that in the case of *bridge points* overlapping regions may be present in the *true* partition, whereas *fuzzy* "partition" may be just an abstract structure built to avoid bias in centroids estimation. In other words, a partition with overlapping regions may reflect the *true* partition we are looking for, while the *fuzzy* partition do not.

Thus, the ideal would be a method to catch and handle the overlapping regions, while based on a suitable notion of distance. But the issue of detecting overlapping regions is not a trivial one. In fact, for the same reason discussed above, such a task can not be achieved by none of the methods entirely based on metric distance: with two or more centroids and a point in a overlapping region, not only this point but also all the other points in a suitable neighborhood of the same point will be assigned to the nearest centroid (in the sense of the choosen distance), thus vanishing the possibility of detecting overlapping regions.

In other words, all of the mehods, based on a distance only, will induce a partition in the sense of clusters, that is a partition in the corresponding D-dimensional space, while overlapping regions do not fulfill this assumption. From this point of view, to handle the overlapping regions and to allow for a suitable notion of distance are two different tasks which could be in conflict: the first aim may not be achievable while pursuing the second.

This complex scenario is the starting point of our proposal: to combine a distance approach with another procedure (based on symmetry) to counterbalance these drawbacks. In this context, our work is intended to study how *bridge points* influence centroid estimates in the case of Gaussian clusters, and to propose a skewness-based method to improve these estimates. In this sense, we expect symmetry to

show two particular features at the same time: to detect Gaussian-shape clusters while improving allocation of the points in overlapping regions. The comparison in clustering results will be done with respect to the Gussian mixture approach as implemented by the *Mclust* algorithm (see Chapter 3 for details). In the case of Gaussian clusters, this is surely a challenging task, because Gussian mixture models reflect the underlying "truth".

Chapter 2

Prototype-based methods

Partitional clustering methods represent one of the earlier and most famous set of techniques in the clustering history. The name comes from their main feature: these methods start from an initial partition and modify it at each step of the process according to some criterion, until a given convergence rule is satisfied. In other words, they work essentially as an iterative relocation algorithm. According to Äyrämö and Kärkkäinen (2006), we can describe a general partitive clustering method as follows:

Input: The number of clusters K, and a database $\mathbf{X}$ containing n objects in $\mathbb{R}^D$
Output: A set of K clusters, which minimizes a criterion function $\mathrm{Q}(\mathbf{X}, K)$.
Step 1. Begin with initial K centers/prototypes as the initial solution.
Step 2. (Re)compute memberships for the data points using the current cluster centers.
Step 3. Update some/all cluster centers/prototypes according to the updated membersips of the data points.
Step 4. Repeat Step 2-3 until no convergence in terms of $\mathrm{Q}(\mathbf{X}, K)$ or no data point changes cluster membership.

In this general context, partitional clustering methods aim at estimating cluster centers, which denote representative quantities for the clusters. To this end, not only the mean, but also the medoid (an element of the cluster as a representative member) represents a typical choice. The best-known strategies in this sense are, respectively, the *K-means* and the more robust *K-medoids* (this one exploiting medoid, an element of the cluster as a representative member, rather than mean).

From this point of view, it is possible to distinguish between different partitional clustering algorithms on the basis of the quantities choosen as representative, which often are referred to as prototypes, as detailed in Xiao and Yu (2012): "Generally, partitional clustering algorithms suppose that the data set can be represented by a set of prototypes, therefore is also called prototype-based clustering method ... According to different definitions of prototypes, prototype-based clustering methhods

can be widely categorized into two groups: point-prototype-based clustering algorithms and prototype-based clustering algorithms using non-point prototypes, such as line, hyperplane, and hypersphere, generally called non-point-prototype-based clustering algorithms".

In the next, we will focus on point prototype-based clustering algorithms, because they are likely the most commonly use, and above all, our proposal is also point-prototype-based.

Essentially, point-prototype-based clustering defines cluster as a set represented by a point in the space of the features. In this sense, point prototype clustering methods may be distinguished into two types: "virtual" point prototype clustering and "actual" data point prototype clustering. Roughly speaking, prototypes in *virtual* point prototype clustering techniques do not necessarily belong to the original dataset, while prototypes in *actual* data point prototype clustering do. For instance, if we choose means as cluster prototypes, these will not be in the original dataset (apart from special cases, e.g. if a cluster contains only one observation, this will obviously coincide with the corresponding mean). So, in the following, we will first discuss virtual point prototype clustering methods and then proceed to the actual prototype clustering methods.

2.1. Virtual point prototype-based methods

The underlying idea of virtual point prototype clustering methods is essentially to minimize a given objective function, with each cluster represented by a virtual point prototype. Basically, it is possible to define a generic objective function as follows

$$\mathrm{Q}\left(\mathbf{X}, K\right) = \sum_{k=1}^{K} \sum_{i=1}^{n} w_{ik} \left\|\boldsymbol{x}_i - \boldsymbol{m}_k\right\|^p$$

where $w_{ik} \in \{0, 1\}$ denotes the membership value (boolean) of a data point i to cluster k, $\boldsymbol{m}_k$ is the prototype relative to that cluster k, the superscript p is the power used for the distance between the obserbation $\boldsymbol{x}_i$ and the k-th prototype $\boldsymbol{m}_k$. Clearly, many different manipulations of the objective function are also possible, see below for details. Among the earliest partitional clustering algorithms we can find the *K-means*, see Forgy (1965) and MacQueen (1967), which is a typical virtual point prototype based clustering approach, based on means as cluster prototypes,

i.e. $\boldsymbol{m}_k = \bar{\boldsymbol{x}}_{\boldsymbol{k}}$. In the next, we discuss the main aspects of this method and some of its variants.

2.1.1. K-means

Essentially, *K-means* is an iterative method that aims at splitting a dataset into K disjoint groups. *K-means* chooses the cluster means as ptototypes and the Euclidean distance as a measure of dissimilarity, i.e. $\boldsymbol{m}_k = \bar{\boldsymbol{x}}_{\boldsymbol{k}}$ and $p = 2$. Perhaps, its main distinctive feature is the objective function, which is based on the within-cluster squared error. This both measures the quality of the clustering result and rules the allocation process. Formally, in the case of a *D*-dimensional dataset, and a sample including n points $\boldsymbol{x}_i \in \mathbb{R}^D$, $i = 1, ..., n$ and for a choice of the number of cluster K, the criterion to be minimized is

$$\mathrm{Q}\left(\mathbf{X}, K\right) = \sum_{k=1}^{K} \sum_{i=1}^{n} w_{ik} \left\| \boldsymbol{x}_i - \boldsymbol{m}_k \right\|^2$$

In the previous expression $\sum_{i=1}^{n} w_{ik} = \sum_{i \in S_k}$, where S_k is the set of points assigned to the kth cluster, and $\boldsymbol{m}_k = \bar{\boldsymbol{x}}_{\boldsymbol{k}}$ is the centroid of the kth cluster. Note that the equality $\boldsymbol{m}_k = \bar{\boldsymbol{x}}_{\boldsymbol{k}}$ holds since, conditionally to a given partition, the aritmetic mean is a minimizer of the corresponding cluster within deviance (as it defines the center of order 2), and the sum of minima over S_k guarantees the total minimum for the objective function. So, in matrix notation, the above criterion corresponds to minimize $\sum_{k=1}^{K} \mathrm{trace}\left(W_k\right)$, where W_k is the covariance matrix for the kth cluster. In fact, the trace of a (square) matrix is the sum of its diagonal elements, i.e. in the case of W_k the variances for each dimension. The *D*-dimensional point which minimizes $\mathrm{trace}\left(W_k\right)$ corresponds to the D aritmetic means $\bar{\boldsymbol{x}}_{\boldsymbol{k}}$, and the expressions above coincide.

In this sense, *K-means* is also referred to as a variance minimization technique, see Kaufman and Rousseeuw (1990). Before the "official" *K-means* was proposed by Forgy (1965) and MacQueen (1967), a similar criterion was proposed by Ward (1963), in a hierarchical rather than a partitive context.

Hereafter a sketch of the *K-means* algorithm in the version described by Forgy (1965), for a dataset containing n objects $\boldsymbol{x}_i \in \mathbb{R}^D$, $i = 1, ..., n$ is given

Input: Choose a number of clusters, say K
Step 1. randomly initialize K centroids $\bar{\boldsymbol{x}}_{\boldsymbol{k}} \in \mathbb{R}^D$
Step 2. for $i = 1, ..., n$,
allocate $\boldsymbol{x}_i$ to the kth cluster according to the following criterion

$$w_{ik} = 1 \; if \quad k = \arg\min_k \|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{k}}\|$$

Step 3. Update centroids $\bar{\boldsymbol{x}}_{\boldsymbol{k}}$ according to the new w_{ik}, with a cluster cardinality equal to $|S_k|$, do
for $k = 1, ..., K$,
for $i = 1, ..., n$

$$\bar{\boldsymbol{x}}_{\boldsymbol{k}} = \frac{\sum_{i=1}^{n} w_{ik}\boldsymbol{x}_i}{\sum_{i=1}^{n} w_{ik}} = \frac{\sum_{i \in S_k} \boldsymbol{x}_i}{|S_k|}$$

Step 4. Repeat Step 2-3 until no data point i changes cluster membersip.
Output: A set of K clusters, which (locally) minimizes the objective function

$$\mathrm{Q}\left(\bar{\boldsymbol{x}}_{\boldsymbol{k}}; \boldsymbol{x}\right) = \sum_{k=1}^{K} \sum_{i \in S_k} \|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{k}}\|^2$$

The *K-means* is a greedy algorithm, that is at every run it produces the maximum decrease in the objective function $\mathrm{Q}\left(\bar{\boldsymbol{x}}_{\boldsymbol{k}}; \boldsymbol{x}\right)$, until it converges to a local minimum, see Jain (2010).

Actually, the same *K-means* criterion encompasses both strenghts and weaknesses. In fact, inducing compact clusters (i.e. with a low within-cluster variability) makes interpretability easier (highly homogeneous elements in each cluster). On the other hand, it fails every time *true* clusters are somewhat similar (i.e. not well-separated centroids and/or high variances, see further discussion in Chapter 4). The choice of the euclidean norm in the objective function makes clustering results sensitive to extreme values, so that the process shows a low level of robustness. The choice of medoids rather than means in the *K-medoids*, see Kaufman and Rousseeuw (1987), is intented to balance this drawback. Paragraph 2.1.1.2 is devoted to discuss in more details some of these drawbacks.

However, its implementational simplicity and computational efficiency makes it a still actual and popular clustering method. For the same reasons, it has been considered as an initialization technique for other computationally expensive methods, see e.g. Bradley and Fayyad (1998). Finally, *K-means* type algorithms have been

developed in a wide number of variants, the main of which are presented in the following.

2.1.1.1. Variants of the K-means method

Before the "official" release of the *K-means* metod, described by Forgy (1965) and MacQueen (1967), two early works, which are strictly related to those, have been introduced by Fisher (1958) and Lloyd (1957, 1982). As outlined in Äyrämö and Kärkkäinen (2006) "*K-means* type grouping has a long history. For instance, already in 1958, Fisher investigated this problem in one-dimensional case as a grouping problem", and even earlier, "Lloyd presented a quantization algorithms for pulse-code modulation (PCM) of analog signals. The algorithm is often referred to as Lloyd's algorithm and it is actually equivalent with the Forgy's *K-means* algorithm in a scalar case". Although Lloyd's paper was not published before 1982, the unpublished manuscript from 1957 is referred, for example, by Chen (1977) and Linde *et al.* (1980), respectively.

However, the first versions of the *K-means* method were published independently by Forgy (1965) and MacQueen (1967). There are two main differences between the two formulations:

1. Allocation step. In Forgy (1965), cluster centers are updated only after *all* observations are allocated to the closest centroid, while in the MacQueen (1967) release the centroids are updated *every time* a single data point is assigned to a cluster (clearly only the two clusters involved will be updated, i.e. the cluster which "looses" his data point and the one which "gains" it).

2. Convergence. In Forgy (1965) the algorithm runs until convergence is reached, generally in a time proportional to $O(nDKt)$ where *t* is the number of iterations, see Duda et al. (2001). Vattani (2011) showed that *K-means* may converge in an exponential time "even in the plane". In the release by MacQueen (1967) the basic algorithm runs only one time (until all data points are allocated to the K clusters, so that there's no iteration, see Äyrämö and Kärkkäinen (2006)). These features of *K-means* algorithms will also be addressed within the framework of our proposal, see Chapter 5.

2.1.1.2. Drawbacks of the K-means

As we noted before, the different versions of the *K-means* algorithm have some drawbacks that helped to kick-start new variants that have been developed in the last decades; for a review see e.g. Äyrämö and Kärkkäinen (2006). Among the most well known drawbacks we may recall:

DRAWBACKS RELATED TO THE ALGORITHM

1. *Sensitivity to initial configuration*. The basic algorithms are local search heuristics and the *K-means* cost function is non-convex; therefore the algorithm is very sensitive to the initial configuration and the resulting partition is often only suboptimal.

2. *Order-dependency*. The MacQueen's basic and converging variants are sensitive to the order in which the points are relocated. This is not the case for the batch versions (such as Forgy's one).

3. *Empty clusters*. The Forgy's batch version may lead to empty clusters due to poor initialization (while in MacQueen's formulation this usually does not occur; if the process leads to a cluster with a single observation, it would coincide with the mean, thus necessary remaining in the same cluster).

DRAWBACKS RELATED TO THE METHOD

4. *Lack of robustness*. The sample mean and variance are very sensitive to outliers. So-called breakdown point for the mean is zero (roughly speaking, the breakdown point is the proportion of outlying observations, which an estimator can handle before giving an incorrect result). This means that even only one huge error may completely bias the estimate. The obvious consequence is that the *K-means* method is highly non-robust as well.

5. *Unknown number of clusters*. Since *K-means* is in general a "non-hierarchical" method, it does not provide any information about the number of clusters, in the sense it is started for given and fixed K. However, it is possible to develop *ad-hoc* procedures to choose the optimal number of clusters. See, for instance, Hamerly and Elkan (2004).

6. *Only spherical clusters*. *K-means* is based on spherical components/clusters. Therefore, a large amount of "clean" data is usually needed for successful clustering.

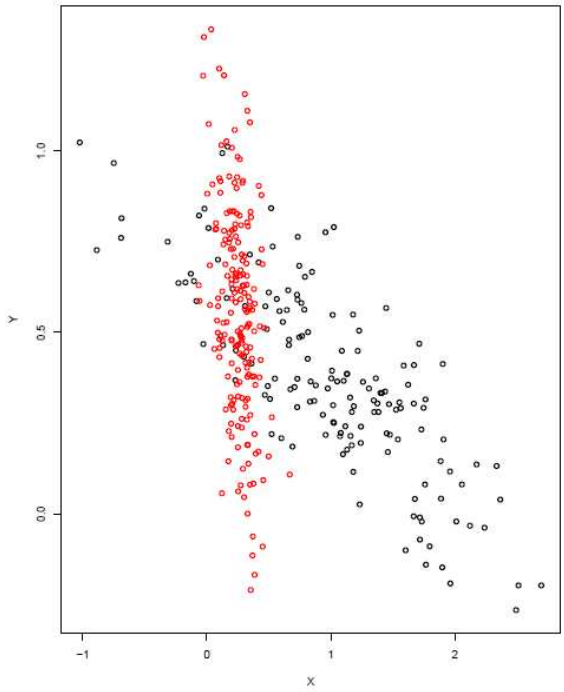
7. *Handling of nominal values*. The sample mean is not defined for nominal values. To solve this issue several variants for the original versions have been developed.

8. *Hard membership values.* The membership values in the *K-means* functions are "hard" in that they may assume only two values, 0 or 1, since each data point can be assigned only to a single cluster (the so-called *hard assignment*). According

to Xiao and Yu (2012), this drawback makes the *K-means* method not applicable to complex datasets which contain overlapping clusters or some data points that can not be easily allocated to one cluster.

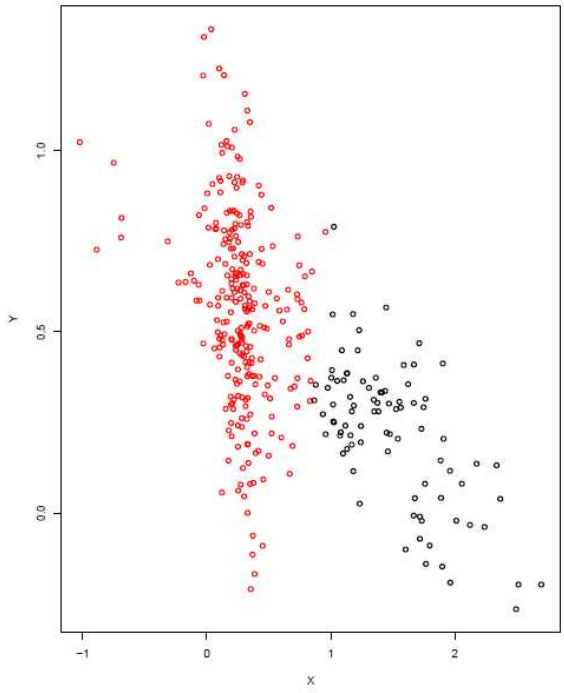
Actually, the real problem with overlapping clusters, in our opinion is not due to the *hard assignment* strategy, but rather to the objective function itself. In this sense, it seems hard to suppose that a distance based criterion can be suitable for the case of overlapping clusters. To catch this issue, let us consider the simple case of two clusters ($K = 2$) for a bidimensional dataset, as depicted in the following figure:

Fig. 2.1: Overlapping on a tail (*true* partition)



Now, let us assume that the overlapping region is delimited by the intersection of the central zone for the red cluster and a tail zone for the black cluster. Any objective function based on a metric distance (e.g. Manhattan or Euclidean) will assign the observations contained in the overlapping region to the red cluster (and also other observations in a neighborhood determined by the adopted metric distance), thus inducing bias in centroid estimates (see Chapter 5 for further discussion). For instance, basing on Euclidean distance only, the *K-means* provides the following solution:

Fig. 2.2: Overlapping on a tail (*K-means* partition)



In other words, when we approach such a problem, a metric distance-based criterion can not be adopted alone, but needs to be supplemented by a further kind of measure, which preferably accounts for other cluster features (such as skewness, like in our approach).

Here our interest is mainly focused on point 8, because relaxing the hypothesis of *hard membership*, i.e. letting w_{ik} vary between 0 and 1, we can introduce another class of virtual point prototype clustering methods, namely the *fuzzy C-means* method. In the next section, we are going to discuss the *fuzzy C-means* algorithm, originally developed by Dunn (1973) and Bezdek (1973,1981).

2.1.2. The Fuzzy C-means algorithm

Let us start again considering the D-dimensional dataset $\mathbf{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n)$, and suppose the aim is at finding K disjoint clusters, say $S_1, S_2, ..., S_K$, so that $\cup_k S_k = S$.

Therefore, a clustering result would be a partition $\{S_1, S_2, ..., S_K\}$ from the original dataset $\mathbf{X}$. Then, let us suppose the probability membership w_{ik} is such that

$w_{ik} \in \{0,1\}$, $\sum_k w_{ik} = 1$, that is each $\boldsymbol{x}_{\mathrm{i}}$ can belong only to a cluster. Thus, in this definition we have stated the following two hypotheses:

1. $w_{ik} \in \{0,1\}$, $\sum_k w_{ik} = 1$
2. $\cup_k S_k = S$

By relaxing one or both hypotheses, we may define a generalization of the formulation given in Chapter 1. By holding both conditions we could obtain the so-called *hard* (also known as *crisp*) clustering, in which every data point belongs to one and only one cluster. By relaxing the first one, we would obtain a so-called *soft* (also known as *fuzzy*) clustering, in which we have, in general, $w_{ik} \in [0,1]$, $\sum_k w_{ik} = 1$, i.e. every data point could belong to more than one cluster. Therefore the K subsets don't form a partition of S. This is the main feature of *fuzzy* clustering, which will be discussed in this chapter.

The second hypothesis, instead, deals with a sort of exhaustiveness property: if we relaxe it, we would obtain a subset of the sample S, since $S_1 \cup S_2 \cup ... \cup S_K \neq S$ implies $S_1 \cup S_2 \cup ... \cup S_K \subset S$. This context is typical of those robust clustering techniques which adopt a trimming approach to clustering, see e.g. *trimmed K-means* by Cuesta-Albertos, Gordaliza, and Matrán (1997) and the *Tclust* approach by García-Escudero, Gordaliza, Matrán and Mayo-Iscar (2008). Nonetheless, other types of robust clustering techniques have been proposed in the literature, which do not adopt a trimming approach, see e.g. *OTRIMLE* (*Optimally Tuned Robust Improper Maximum Likelihood Estimator*) by Coretto and Hennig (2013a,b). Clearly, many other important references could be found as well.

Both hypotheses can be jointly relaxed, thus generalizing the formulation given in Chapter 1 and leading, for example, to trimming-based *fuzzy* clustering algorithm, see e.g. Dotto, Farcomeni, García-Escudero and Mayo-Iscar (2017).

In contrast with the earliest versions of the *K-means*, *fuzzy* clustering allows for multiple membership data point (also in this case the literature uses the expression "overlapping clusters"). The seminal papers for *fuzzy* clustering tecniques are Dunn (1973) and Bezdek (1973,1981). See also Ruspini (1969 and 1970), where the concept of fuzzy sets in a clustering framework was already formulated.

The main idea behind this approach is to introduce a coefficient $w_{ik} \in [0,1]$, with $i = 1, ..., n$, $k = 1, ..., K$, which defines for each data point $\boldsymbol{x}_i$, its degrees of membership to the k-th cluster. Therefore, w_{ik} may be interpreted as a probability for the ith observation to belong to cluster k, where $\sum_{k=1}^{K} w_{ik} = 1$ holds $i = 1, ..., n$. In this context, the objective function needs to be modified as follows:

$$\mathrm{Q}\left(\mathbf{X}, K\right) = \sum_{k=1}^{K} \sum_{i=1}^{n} w_{ik}^{m} \left\| \boldsymbol{x}_i - \boldsymbol{m}_k \right\|^2$$

This function, for $m > 1$ and provided that $\boldsymbol{x}_i \neq \boldsymbol{m}_k$ for all i and k, is locally minimized if

$$\hat{w}_{ik}^{m} = \frac{1}{\sum_{c=1}^{K} \left(\frac{\left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{k}} \right\|}{\left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{c}} \right\|} \right)^{\frac{2}{m-1}}}$$

and

$$\hat{\boldsymbol{m}}_k = \bar{\boldsymbol{x}}_k = \frac{\sum_{i=1}^{n} w_{ik}^{m} \boldsymbol{x}_i}{\sum_{i=1}^{n} w_{ik}^{m}}$$

The (weighting) power parameter $m \in \mathbb{R}$, $m \geq 1$, is referred to as the *fuzziness* parameter. Note that $\bar{\boldsymbol{x}}_k$ (the centroid of the cluster k) is the weighted mean of all data points, with weights equals to the exponentiated degrees of membership w_{ik}^{m}, see Bezdek, Ehrlich and Full (1984).

So, in the *fuzzy* logical architecture the weighting parameter m becomes a quite important parameter, that significantly influences the *fuzziness* of the resulting partition. For instance, as $m \to 1^+$, the partition becomes *hard*, i.e $w_{ik} \in \{0, 1\}$, and $\bar{\boldsymbol{x}}_k$ are the "ordinary means". In fact, assuming $m \to 1^+$ we have for $i = 1, ..., n$:

$$\lim_{m \to 1^+} w_{ik}^{m} = \frac{1}{\sum_{c=1}^{K} \left(\frac{\left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{k}} \right\|}{\left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{c}} \right\|} \right)^{\frac{2}{1^+ - 1}}} = \frac{1}{\sum_{c=1}^{K} \left(\frac{\left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{k}} \right\|}{\left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{c}} \right\|} \right)^{+\infty}}$$

and this expression goes to 1, only for $k = k^*$ where $k^* = argmin_k \left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{k}} \right\|$:

$$\lim_{m \to 1^+} w_{ik^*}^{m} = \frac{1}{\sum_{c=1}^{K} \left(\frac{\left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{k}^*} \right\|}{\left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{c}} \right\|} \right)^{+\infty}} = 0 + ... + 1 + ... + 0 = 1$$

While, with $k \neq argmin_k \left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{k}} \right\|$ the above expression is identically null:

$$\lim_{m \to 1^+} w_{ik}^{m} = \frac{1}{\sum_{c=1}^{K} \left(\frac{\left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{k}} \right\|}{\left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{c}} \right\|} \right)^{+\infty}} = \frac{1}{0 + \infty + ... + 1} = 0$$

where the term in the denominator goes to 0 every time $\left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{k}} \right\| < \left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{c}} \right\|$, while it goes to $+\infty$ every time $\left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{k}} \right\| > \left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{c}} \right\|$ and it is equal to 1 if $k = c$, i.e $\left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{k}} \right\| = \left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{c}} \right\|$. So, for $m \to 1^+$ and $k = k^*$ with $k^* = argmin_k \left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{k}} \right\|$, $w_{ik}^{m} \to 1$, while for $m \to 1^+$ and $k \neq k^*$ $w_{ik}^{m} \to 0$.

Therefore, for $m \to 1^+$ the membership coefficient w_{ik}^m becomes a boolean variable which is equal to 1 if $k = argmin_c \|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_c\|$ and 0 otherwise, i.e. $w_{ik}^m = 1_{k=argmin_c\|\boldsymbol{x}_i-\bar{\boldsymbol{x}}_{\boldsymbol{c}}\|}$. So, relatively to the centroid, we have for $m \to 1^+$

$$\bar{\boldsymbol{x}}_{\boldsymbol{k}} = \frac{\sum_{i=1}^n w_{ik}^m \boldsymbol{x}_i}{\sum_{i=1}^n w_{ik}^m} = \frac{\sum_{i \in S_k} \boldsymbol{x}_i}{|S_k|}$$

Note that in the above formula we have put $\sum_{i=1}^n 1_{k=argmin_c\|\boldsymbol{x}_i-\bar{\boldsymbol{x}}_{\boldsymbol{c}}\|} = \sum_{i=1}^n 1_{i \in S_k}$ since, for a fixed k, S_k is the set of all objects $i = 1, ..., n$ such that $k = argmin_c \|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_c\|$. Actually, such a notation is not strictly necessary, but it helps to represent $\bar{\boldsymbol{x}}_{\boldsymbol{k}}$ as an "ordinary mean" calculated over the set S_k.

On the other hand, for $m \to +\infty$ the partition becomes completely *fuzzy*, i.e. $w_{ik} = 1/K$ for $i = 1, ..., n$ and $k = 1, ..., K$, with maximum eterogeneity in the cluster membership of all data points:

$$\lim_{m \to +\infty} w_{ik}^m = \frac{1}{\sum_{c=1}^K \left(\frac{\|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{k}}\|}{\|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{c}}\|} \right)^{0^+}} = \frac{1}{\sum_{c=1}^K 1} = \frac{1}{K}$$

With the same formulation, it is also straightforward to show that, for fixed m, the condition $\sum_{k=1}^K w_{ik} = 1$ holds for $i = 1, ..., n$. To show this, let us simplify notation, denoting $d_k = \|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k\|^{\frac{2}{m-1}}$, $k = 1, ..., K$. So, for an observed $\boldsymbol{x}_i$, we may rewrite $\sum_{k=1}^K w_{ik}^m$ as follows:

$$\sum_{k=1}^K w_{ik}^m = \sum_{k=1}^K \frac{1}{\sum_{c=1}^K \left(\frac{\|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{k}}\|}{\|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{c}}\|} \right)^{\frac{2}{m-1}}} = \sum_{k=1}^K \frac{1}{\sum_{c=1}^K \left(\frac{d_k}{d_c} \right)} = \sum_{k=1}^K \frac{1}{\left(\frac{d_k}{d_1} + \frac{d_k}{d_2} + ... + \frac{d_k}{d_K} \right)} =$$

$$\frac{1}{\left(\frac{d_1}{d_1} + \frac{d_1}{d_2} + ... + \frac{d_1}{d_K} \right)} + \frac{1}{\left(\frac{d_2}{d_1} + \frac{d_2}{d_2} + ... + \frac{d_2}{d_K} \right)} + ... + \frac{1}{\left(\frac{d_K}{d_1} + \frac{d_K}{d_2} + ... + \frac{d_K}{d_K} \right)} =$$

$$\frac{1}{\left(\frac{d_1 \prod_{c \neq 1, c=1}^K d_c}{\prod_{c=1}^K d_c} + \frac{d_1 \prod_{c \neq 2, c=1}^K d_c}{\prod_{c=1}^K d_c} + ... + \frac{d_1 \prod_{c \neq K, c=1}^K d_c}{\prod_{c=1}^K d_c} \right)} +$$

$$+ ... + \frac{1}{\left(\frac{d_K \prod_{c \neq 1, c=1}^K d_c}{\prod_{c=1}^K d_c} + \frac{d_K \prod_{c \neq 2, c=1}^K d_c}{\prod_{c=1}^K d_c} + ... + \frac{d_K \prod_{c \neq K, c=1}^K d_c}{\prod_{c=1}^K d_c} \right)} =$$

$$= \frac{\prod_{c=1}^{K} C_c \prod_{c\neq 1,c=1}^{K} C_c + \prod_{c=1}^{K} C_c \prod_{c\neq 2,c=1}^{K} C_c + ... + \prod_{c=1}^{K} C_c \prod_{c\neq K,c=1}^{K} C_c}{\prod_{c=1}^{K} C_c \left(\prod_{c\neq 1,c=1}^{K} C_c + \prod_{c\neq 2,c=1}^{K} C_c + ... + \prod_{c\neq K,c=1}^{K} C_c \right)} =$$

$$= \frac{\prod_{c=1}^{K} C_c \left(\prod_{c\neq 1,c=1}^{K} C_c + \prod_{c\neq 2,c=1}^{K} C_c + ... + \prod_{c\neq K,c=1}^{K} C_c \right)}{\prod_{c=1}^{K} C_c \left(\prod_{c\neq 1,c=1}^{K} C_c + \prod_{c\neq 2,c=1}^{K} C_c + ... + \prod_{c\neq K,c=1}^{K} C_c \right)} = 1$$

The *fuzzy C-means* algorithm can therefore be sketched as follows:

Input: Choose a value for K, the *fuzziness* parameter m, and the threshold $\varepsilon > 0$ (eventually a norm $\|\cdot\|$)

Step 1. At step $t = 0$ randomly assign k membership coefficients $w_{ik}^{(t=0)}$ to each point $\boldsymbol{x}_i$ subject to $\sum_{k=1}^{K} w^{(t=0)}{}_{ik} = 1$ for $i = 1, ..., n$,

Repeat for t=1,2,...

Step 2. for $k = 1, ..., K$, update the cluster prototypes (weighted means):

$$\bar{\boldsymbol{x}}_{\boldsymbol{k}}^{(\boldsymbol{t})} = \frac{\sum_{i=1}^{n} w_{ik}^{(t-1)} \boldsymbol{x}_i}{\sum_{i=1}^{n} w_{ik}^{(t-1)}}$$

Step 3. update the $n \times K$ distances $d_{ik}^{(t)}$ between each point $\boldsymbol{x}_i$ and the k centroids $\bar{\boldsymbol{x}}_{\boldsymbol{k}}$, i.e.

for $k = 1, ..., K$,

for $i = 1, ..., n$

$$d_{ik}^{(t)} = \left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{k}}{}^{(t)} \right\|$$

Step 4. update the $n \times K$ partition matrix $\mathbf{W}^{(t)}$ according to the updated $d_{ik}^{(t)}$, i.e.

for $k = 1, ..., K$,

for $i = 1, ..., n$

$$w^{(t)}{}_{ik} = \frac{1}{\sum_{c=1}^{K} \left(\frac{\left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{k}}{}^{(t)} \right\|}{\left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{c}}{}^{(t)} \right\|} \right)^{\frac{2}{m-1}}}$$

Until $\max_{i,k} \left\| \mathbf{W}_{ik}^{(t)} - \mathbf{W}_{ik}^{(t-1)} \right\| < \varepsilon$

Output: A set of K clusters, which (locally) minimizes the objective function

$$\mathrm{Q}\left(\mathbf{X}, K\right) = \sum_{k=1}^{K} \sum_{i=1}^{n} w_{ik} \left\| \boldsymbol{x}_i - \boldsymbol{m}_k \right\|^2$$

2.1.2.1. Drawbacks of the Fuzzy C-means

In the above scheme of the *fuzzy C-means* algorithm, a singularity can occur if, for some i and k, $\|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_{\boldsymbol{k}}\| = 0$, thus vanishing the calculation of the corresponding ${w^{(t)}}_{ik}$. This case is quite rare in practice, and clearly, there are many possibilities to overcome this drawback; for instance, Bezdek, Ehrlich and Full (1984) pointed out that "this eventuality to our knowledge, has never occurred in nearly 10 years of computing experience".

The above formulation of *fuzzy C-means* approach shows a direct link to the *K-means* method: both minimize intra-cluster variance and reach a local minimum when $\hat{\boldsymbol{m}}_k = \bar{\boldsymbol{x}}_k$. In both cases the results depend on the initial choices (weights for *fuzzy C-means* and centroids for the *K-means*).

Some comparisons between the two methods could be found in literature, see for instance recent papers from Cebeci and Yildiz (2015) or Yin, Sun, Yang and Guo (2014), where a comparison is carried out in the case of well separated cluster structures with regular patterns and in the arterial input function (AIF) detection, respectively.

When compared to *hard* assignment clustering methods, the *fuzzy C-means* provides more information about the structure of the data set, due to the varying degree of membership for each data point. Nevertheless, such a gain in information induce non negligible costs in term of computational complexity. In fact, with respect to Forgy (1965) version of *K-means*, which generally converges in a time of order $O(nDKt)$, the computational complexity of *fuzzy C-means* is $O(nDK^2t)$, which grows faster with the number of clusters K and, therefore, it may be not appropriate for large datasets.

The first formulation of the *fuzzy C-means* method assumes that the points in the dataset are equally important; clustering results are affected by outliers, see Xiao and Yu (2012).

To overcome these drawbacks, different versions of the standard algorithm have been proposed in the literature. For instance, to deal with the issue of sensitivity to noise, Ohashi (1984) proposed a *fuzzy C-means*-type algorithm by assuming that a separate outlier cluster is present. Menard et al. (2003) developed a *fuzzy*

generalized C-means (*FGCM*) algorithm, and a few years after Yu and Yang (2007) proposed the *generalized fuzzy clustering regularization* algorithm (*GFCR*). Naturally, many other versions of the *fuzzy C-means*-type algorithm can be found in the literature.

Despite the increasing number of papers focused on *fuzzy C-means* and its variants, an analogous attention to related software developments missed for a long time. This gap has been recently solved by Ferraro and Giordani (2015) with development of the R package $fclust$.

2.2 Actual data point prototype-based methods

As we noticed before, the main difference between virtual and actual point prototype clustering methods is that in the last only real set data points can be defined as cluster prototypes, while the aim of the two methods is the same: find a partition which minimizes a given objective function.

Some of the most important actual data point prototype methods were developed as alternative versions to *K-means* method, especially to overcome its lack of robustness. The earliest and perhaps the most famous methods are the ones included in the *K-medoids* family, developed since Kaufman and Rousseeuw (1987), who proposed the *PAM* (*Partitioning Around Medoids*) method starting from an idea introduced by Vinod (1969). Both *K-means* and *K-medoids* aim at partitioning a dataset in clusters that minimize the distance between observations and the corresponding prototypes. However, while *K-means* uses mean as cluster prototypes, *K-medoids* exploits actual data points as prototypes; these are referred to as *medoids*. The *medoid* of a cluster is the object for which the average dissimilarity (or equivalently the total dissimilariry) with respect to all the objects of that cluster is a minimum, see Kaufman and Rousseeuw (1987).

In the following, we discuss some of the most important *K-medoids* methods, starting from the *PAM*, see Kaufman and Rousseeuw (1987).

2.2.1 Partitioning Around Medoids (PAM)

PAM is based on the selection of an object as a representative (*medoid*) for a cluster. In this context, the distance is interpreted as the dissimilarity between a

generic object and the *medoid* of the cluster to which it belongs. To find an estimate for the parameters, that is a partition in K clusters, and a representative element for each cluster, one has to implement two types of actions:
1. The selection of K observations as representative objects of the K clusters (*medoids*). To this end, a boolean variable y_i is considered, which is equal to one if and only if the object i, $i = 1, ..., n$, is selected as a *medoid* (in the previous notation if $\boldsymbol{x}_i = \boldsymbol{m}_k$, that is if $\boldsymbol{x}_i$ is selected as prototype for cluster k).
2. The assignment of each observation $j = 1, ..., n$, to one of the K selected representative objects. In this sense, we consider a further boolean variable z_{ij}, equal to one if and only if data point j is assigned to the cluster for which the data point i is the *medoid*.

To put it formally, let us consider a data set of n observations $\mathbf{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n)$, the variables $y_i, z_{ij} \in \{0, 1\}$ and a measure of dissimilarity between two generic data points $\boldsymbol{x}_\mathrm{i}$ and $\boldsymbol{x}_\mathrm{j}$, $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Thus, the corresponding objective function can be describe as follows:

$$\mathrm{Q}(\mathbf{X}, K) = \sum_{i=1}^{n} \sum_{j=1}^{n} d(\boldsymbol{x}_i, \boldsymbol{x}_j, K)\, z_{ij}$$

subject to the following set of constraints, which makes *PAM* a so-called zero-one linear program:

$$\sum_{i=1}^{n} z_{ij} = 1 \qquad j = 1, ..., n$$

$$z_{ij} \leqslant y_i \qquad i, j = 1, ..., n$$

$$\sum_{i=1}^{n} y_i = K$$

The first constraint ensures that each data pont j is assigned to a single representative object (*medoid*, which represents a specific cluster, a *hard* assignment). Indeed, for a given j only one of the z_{ij} is equal to one and all other must be zero. The second constraint implies that an object j can only be assigned to a single object i if this last object has been selected as a *medoid*. In fact, if the i-th observation is not a *medoid*, then y_i is zero (remind that $y_i, z_{ij} \in \{0, 1\}$) and the constraint forces all z_{ij} to be zero (for every j). Viceversa, if the i-th observation is a *medoid*, then all the z_{ij} (for such an i) can be either zero or one, according to their membership,

that is z_{ij} will be equal to 1 for those j represented by *medoid* i (i.e. belonging to the cluster that *medoid* y_i represents) and zero otherwise. Finally, the last equality ensures that exactly K data point are to be chosen as representative objects.

Essentially, *PAM* works on the symmetric $n \times n$ dissimilarity matrix of the given data set, whose generic i, j entries are given by the above dissimilarity measure $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$. In other words, the aim of PAM is at finding that partition which minimizes the sum of all dissimilarities in the K clusters, summarized in the previous objective function

$$\mathrm{Q}(\mathbf{X}, K) = \sum_{i=1}^{n} \sum_{j=1}^{n} d(\boldsymbol{x}_i, \boldsymbol{x}_j, K)\, z_{ij}$$

To obtain such a goal, the *PAM* algorithm runs in two step. First, it computes the dissimilarity matrix between object i and j, and then searches the set of K data points that are optimal as cluster prototypes to minimize the objective function. The minimization is carried out by exchanging medoids with nonmedoids. Due to the exhaustive swapping operated in the last step, *PAM* algorithm provides clustering results which do not depend upon the order the observations have in the input data set, except if some of the distances between objects are tied, see Kaufman and Rousseeuw (1987).

It is worth notice that *PAM*, choosing *medoids* as prototypes, is more robust to outliers and noise points than *K-means* algorithm, see Xiao and Yu (2012). Nevertheless, with respect to the *K-means* algorithm, *PAM* shows a higher computational complexity, with a time to convergence of order $O\left(K\left(n-K\right)^2 Dt\right)$, which is not suitable for large datasets. An interesting feature of the *PAM* algorithm is the possibility for the user to choose different measures of dissimilarity in the function $d(\cdot, \cdot)$, such Euclidean or Manhattan distances. Moreover, user can give its own dissimilarity matrix, and *PAM* will work even if the values in the matrix do not respect the triangular inequality $d(\boldsymbol{x}_i, \boldsymbol{x}_h) \leqslant d(\boldsymbol{x}_i, \boldsymbol{x}_j) + d(\boldsymbol{x}_j, \boldsymbol{x}_h)$. This last feature makes *PAM* appropriate to deal also with data which are not necessarily on interval scale (binary, ordinal or nominal).

So, the use of the dissimilarity matrix allows for the analysis of non quantitative data (dissimilarities coming even from a subjective point of view), thus extending the *K-means* framework to those cases in which means are not defined.

2.2.2 The evolution of PAM: CLARA and CLARANS algorithms

A few years after introducing *PAM*, Kaufman and Rousseeuw (1990) proposed a new variant of the *PAM* algorithm which is suitable for large dataset, *CLARA* (*Clustering LARge Application*). Essentially, a single run of the *CLARA* algorithm works as follows: instead of the entire dataset, *CLARA* randomly chooses a subsample, as we are in a large data context. For the current subsample, *PAM* is applied to find medoids; the entire original dataset is considered to calculate the current dissimilarity matrix. If this is smaller than the one found in previous runs, then the current medoids are chosen as the solution. The whole process is iterated a prespecified number of times, with an ovearll time complexity of $O\left(Ks^2Dt + K\left(n-K\right)Dt\right)$, where s is the size of the subsample.

Another member of the *K-medoids* family is *CLARANS* (*Clustering Large Applications based on RANdomized Search*), which represent a further development of *CLARA*, introduced by Ng and Han (2002), exploiting a graph-strategy for spatial data mining with a complexity of $O\left(n^2Dt\right)$, thus showing more efficiency when compared to both *PAM* and *CLARA*, see e.g. Ng and Han (2002).

Both *CLARA* and *CLARANS* are intended to make the basic *PAM* algorithm more efficient to be used on potentially large datasets. It is worth noting that *K-medoids* type algorithms are less sensitive to outliers and noise when compared to the *K-means* one. They can also be proved to be invariant to translations and orthogonal transformations of the objects, but *not* to affine transformations that change the inter-object distances, see Kaufman and Rousseeuw (1990).

2.3 Open issues in point prototype-based methods

There are, at least, three main issue related to point prototype clustering methods we have discussed along this section:

1. How to initialize the algorithm.

2. Allocation of data points into clusters and updating of cluster parameters.

3. Unknown number of clusters. We briefly discuss these three issues in turn.

INITIALIZATION. The performance of a partitive clustering is related to the starting values choosen to initialize the algorithm. This is true for virtual point prototype methods (we have to choose initial values for centroids) and for actual point prototype methods (we have to choose K *medoids* to start the algorithm).

This fact derives from the non-convex nature of the criterion functions involved, so that these methods could be trapped into local minima. A naive solution is to

run the algorithm from several different starting points, to gain satisfactory clustering results based on some evaluation index. In fact, point prototype clustering algorithms often converge to local optimal solutions, if the initialization is not suitable for the data at hand. Thus, a good choice for the initial prototypes is a crucial issue that affects the final clustering result. In the literature, several methods have been developed to guide the prototype initialization. For instance, Duda and Hart (1973) suggested a recursive method by running K clustering problems; Fisher (1996) proposed a method based on an initial hierarchical clustering; Khan and Ahmhad (2004) developed a *Cluster Center Initialization Algorithm* (*CCIA*) specifically related to *K-means*. Other options are naturally available, see Pena, Lozano and Larranaga (2006). But, surprisingly, in a comparative study, Steinley and Brusco (2007) show that, among 12 different type of initialization strategies for the *K-means* algorithm, the random initialization outperforms other compared methods for general use. Thus, as pointed out by Xiao and Yu (2012), there is no universally accepted method for selecting initial cluster prototypes, and it is still an open problem.

ALLOCATION. The central iteration of the algorithm, i.e. the allocation of data points and the consequent updating of parameter estimates represents the step that improves the godness of fit for the clustering method with respect to the previous solution. For instance, in *K-means*-type algorithms, there are at least two main different kind of steps: *nearest centroid sorting pass* and *hill-climbing pass*, see Anderberg (1973).

The *Nearest centroid sorting pass* allocates observations to the nearest centroid. In this respect we may adopt two different strategies, see Aldenderfer and Blashfield (1984): *combinatorial*, as in MacQueen (1967) type of *K-means*, and *non-combinatorial*, as described by Forgy (1965). According to the first one, centroids are updated each time a data point is reassigned to a cluster. The *non-combinatorial* case is somewhat simplier and time saving, in that centroids are updated only after *all* the observations have been allocated to the closest cluster centroid. This requires only one center updating for a single run of the algorithm, instead of multiple updates as in the *combinatorial* case.

The *Hill-climbing* starts with an arbitrary initial condition (in our case an initial partition with its clusters centroids), and aims at improving solution by incrementally changing only one element of the solution (i.e. a single data point in the current partition). Only if the change defines a better solution, an incremental change is made to the current solution, and so the algorithms runs until no improvements can be produced. In other words, the *hill-climbing pass* allocates points to clusters

only if the move improves the value of a given criterion, see Äyrämö and Kärkkäinen (2006).

CHOICE OF K. The issue of the unknown number of clusters is crucial in all point prototype-based clustering algorithms. It is a relevant issue, because the number of clusters is an important and implicit part in all clustering frameworks. The point protoype-based methods we have discussed often work only conditionally to a fixed number of clusters. In other words, they need some (often external) criterion to choose the number of clusters. As a proof of the importance of this issue, it is worth noting that there is a huge amount of literature on this topic, see Dubes (1987), the yet cited Hamerly and Elkan (2004) for the case of *K-means*, or Li, Ng, Cheung and Huang (2008) for *Fuzzy C-means*. In this context, Rousseeuw (1987) proposes a graphical technique based on the comparison of cluster tightness and separation to estimate the number of clusters. However, in some fields (e.g. double-blind clinical trials) it makes sense to suppose that the real number of clusters is exactly known (e.g. we could know that only two drugs and a placebo were given to patients). In lack of a unique and objective definition of what a cluster is, the issue of the number of clusters declines a little bit in significance, while it still has its importance in those contexts where a rigorous and unambiguos definition of cluster is available.

Apart from partitive methods, other clustering approaches may be used, e.g. based on hierarchical methods, see for example Lance and Williams (1966). This class produces a set of solutions with different numbers of clusters, summarized in a hierarchical graphical structure (dendrogram), which may be used to choose a sensible number of clusters. But, as pointed out by Äyrämö and Kärkkäinen (2006), "(...) although the hierarchical methods provide some information about the number of clusters, they are not very feasible for data mining problems...quadratic memory requirement of the dissimilarity matrix is intractable for large data sets".

Chapter 3

Finite Mixture Models (with Gaussian kernel) for clustering

In this section we discuss the so-called model based approach to clustering, focusing on Gaussian Finite Mixture Models. So, the chapter will be organised essentially as follows: a first section is dedicated to Finite Mixture Models, with a brief history and some basic definitions. The second paragraph discusses Finite Mixture Models in a clustering framework; a further section deals with Finite Mixture Models and the related maximum likelihood approach to parameter estimation. We consider the EM (*Expectation-Maximisation*) algorithm to estimate mixture parameters. Finally, we discuss a specific implementation of the EM algorithm for the Gaussian case, included in the $Mclust$ package (developed in software R), looking to constraints on the covariance matrices and the case of overlapping clusters.

Note that the choice of dealing with Gaussian Finite Mixture Models in a separate section reflects the aim of this thesis: providing a comparison in clustering performance between Finite Mixture Models with Gaussian kernel (via the EM algorithm) and the proposed, skewness based, approach. Hereafter we will use "Mixture Model" for "Finite Mixture Model", since our discussion will exclusively address Finite Mixture Models.

3.1 Mixture Models

In the following we present an introductory approach to Mixture Models, consisting of a first part with some historic references and a second one where basic definitions are introduced.

3.1.1 A brief history

Although the first paper referred to is that by Pearson (1894), a few years before Simon Newcomb (1886) published an article dealing with an implementation of a Gaussians mixture model to overcome a robustness problem in sample mean estimation, see Stigler (1973). So, to our knowledge, Newcomb (1886) can be considered the first attempt to exploit a mixture model in a statistical framework, although earlier, but implicit, references to Mixture Models can be found in Quetelet (1846,1852) and Holmes (1892), see McLachlan and Peel (2000) for a discussion.

On the other hand, the implementation of a mixture model in Pearson (1894) seems somewhat more interesting to our purpose, since it deals with a sort of clustering problem, while Newcomb (1886) use Gaussian mixtures to solve an unrelated problem. According to Stigler (1973): "Newcomb (1886) provided the first sound, modern approach to robust estimation". So, one could say Newcomb (1886) started implementation of Finite Mixture Models, while Pearson (1894) realized one of the most important use of such a model: splitting a population into components, thus putting Mixture Models in a clustering framework.

His analysis entailed 1000 observations (ratio of forehead to body length of crabs) Pearson (1894) exploited a mixture of two normal eteroschedastic components (i.e. with different variances) to model the skewness in the dataset, provided by Weldon (1892,1893).

It's interesting to note that Pearson (1894) used the method of moments, obtaining results very similar to those achieved by the maximum likelihood approach, see McLachlan and Peel (2000).

Anyway, since the seminal attempt of Pearson (1894) many other works focused on the method of moments to estimate mixture model parameters, while the maximum likelihood approach was not adopted until the mid 20th century, due to related problems such as multiple maxima in the likelihood function or the unboundness of the likelihood function in the case of normal components with unequal covariance matrices. To our knowledge, the first attempt to fit Gaussian Mixtures via the likelihood approach is Rao (1948), for the simple case of a mixture of two univariate distributions with equal variances.

Only in the '60s the likelihood approach becomes an active topic in the literature on Mixtures models, thanks to some papers by Wolfe (1965,1967) and Day (1969), see McLachlan and Peel (2000). Nevertheless, it is thanks to the development of the EM algorithm (Dempster, Laird and Rubin (1977)) that the likelihood method took over in fitting Gaussian Mixtures for heterogeneous data. As we will see in a further

section, this algorithm greatly simplifies parameter estimation in theMixture Model framework.

The first classical paper in this sense was Dempster, Laird and Rubin (1977), but nearly 100 years before an implicit formulation of EM logic can be found in the same Newcomb (1886).

3.1.2 Mixture Models: some basic definitions

Let us consider a random sample of n observations $\boldsymbol{x}_i \in \mathbb{R}^D$, $i = 1, ..., n$, with probability density function $f(\boldsymbol{x}_i)$ and let denote $\mathbf{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n)'$, so that $\mathbf{X}$ is a n-tuple of points in $\mathbb{R}^D$. Suppose $\boldsymbol{x}_i$ is a continuous random vector (otherwise we mean $f(\cdot)$ as a probability mass function), and that we can write the density of $\boldsymbol{x}_i$, $f(\boldsymbol{x}_i)$ as follows:

$$f(\boldsymbol{x}_i) = \sum_{k=1}^{K} \pi_k f_k(\boldsymbol{x}_i)$$

Here $f_k(\cdot)$ are kernel densities and π_k are nonnegative quantities such that

$$\int_{\mathbb{R}^D} f_k(\boldsymbol{x})\, d\boldsymbol{x} = 1 \qquad k = 1, ..., K$$

$$0 \leq \pi_k \leq 1, \qquad \sum_{k=1}^{K} \pi_k = 1$$

In this formulation, π_k are referred to as the mixing proportions (or weights). Note that, since $f_k(\cdot)$, $k = 1, ..., K$, denotes a density, $f(\cdot)$ will be a density as well (a convex combination of probability density functions is still a probability density function). We shall refer to $f(\cdot)$ as the mixture density, with corresponding component specific densities $f_k(\cdot)$, $k = 1, ..., K$.

Note that the expression above implicitly holds for a fixed K; however, in real applications K is unknown and it needs to be estimated as well as π_k and other parameters involved (see section 1).

3.2 Mixture Models for clustering

Another way to represent a K-components Mixture Model is based on introducing component labels in the above formulation. Let $Z_i \in \{1, ..., K\}$ be a categorical random variable which assumes values with probabilities $\pi_1, ..., \pi_K$; this can be represented by the component indicator $\mathbf{z}_i = (z_{i1}, ..., z_{ik})'$ with $z_{ik} = 1$ iff the i-th unit comes from the k-th component. Let us suppose that $f_{\boldsymbol{x}_i|Z_i}$, i.e. the conditional density of $\boldsymbol{x}_i$ given $z_{ik} = 1$ is $f_k(\boldsymbol{x}_i)$, $k = 1, ..., K$. So, the marginal (unconditional) density of $\boldsymbol{x}_i$ will be:

$$m(\boldsymbol{x}_i) = \sum_{k=1}^{K} f_{\boldsymbol{x}_i|z_{ik}=1} \cdot \Pr(z_{ik} = 1) = \sum_{k=1}^{K} \pi_k f_k(\boldsymbol{x}_i) = f_{\mathrm{x}}(\boldsymbol{x}_i|z_{ik} = 1)$$

We can depict the original variable Z_i as a multinomial one, consisting of one draw on K categories (labels) with probabilities $\pi_1, ..., \pi_K$, that is

$$\Pr(\mathbf{Z}_i = \mathbf{z}_i) = \pi_1^{Z_{i1}} \pi_2^{Z_{i2}} ... \pi_K^{Z_{iK}}$$

$i = 1, ..., n$, that is

$$\boldsymbol{Z}_i \sim Mult_K(1, \boldsymbol{\pi})$$

where $\boldsymbol{\pi} = (\pi_1, ..., \pi_K)'$.

It's quite natural to put such a formulation of a Mixture Model within a clustering framework. To this end, consider a situation where $\boldsymbol{x}_i$ is drawn from a population consisting of K clusters, in proportions $\pi_1, ..., \pi_K$. Then, suppose that the density of $\boldsymbol{x}_i$ in the k-th cluster is $f_k(\boldsymbol{x}_i)$ for $k = 1, ..., K$. Therefore, the density of $\boldsymbol{x}_i$ will have the form $m(\boldsymbol{x}_i) = \sum_{k=1}^{K} \pi_k f_k(\boldsymbol{x}_i)$. In this case, the components refer to clusters.

This kind of representation is adequate in those cases where we know in advance that the population is actually a mixture of K distinct groups. This may be the case in biometric researches, where clusters are known *a priori* to exist in some physical sense, see McLachlan and Peel (2000).

However, there are many other cases where such an assumption does not hold (see the discussion in section 1). For instance, if we are interested in modeling heterogeneity in a population which we can not adequately represent by a single distribution, we could exploit the components of a Mixture Model to catch such a heterogeneity. However, in this case there would not be any objective or physical

existence of clusters, and the concept of cluster does not apply to components; these are simply used to model heterogeneity.

Actually, there is also a notable case where we really know in advance the number of clusters that is, when $K = n$, and all the $\pi_1, ..., \pi_n$ proportions are constant and equal to $1/n$. It is worth noting that, by this way, we would obtain a nonparametric kernel estimate of a density.

From this point of view, Mixture Models show great flexibility in several different fields of application. As pointed out by McLachlan and Peel (2000): "(...) it can be seen that mixture models occupy an interesting niche between parametric and nonparametric approaches to statistical estimation ... mixture models have much of the flexibility of nonparametric approaches, while retaining some of the advantages of parametric approaches, such as keeping the dimension of the parameter space down to a reasonable size. Mixture models therefore provide a convenient method of density estimation that lies somewhere between parametric models and kernel density estimators".

Nevertheless, the above formulation of Mixture Models in terms of component indicators does not fit yet to the standard clustering framework. In fact, clustering is referred to as *unsupervised learning*, as we really do not know anything *a priori* about the number of groups or about observations clusters membership, while we have implicitly supposed to know both of them in the above formulation. So, hereafter we will consider the number of groups K as fixed (but unknown) and the component-label vectors $\mathbf{Z}_i$ will be treated as unknown as well.

To put it formally we introduce an incomplete-data structure for the mixture problem. In this case, consider $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n$ as n data points in $\mathbb{R}^D$ coming from a realization of a random sample of n independent and identically distributed (i.i.d.) random vectors $\boldsymbol{X}_1, \boldsymbol{X}_2, ..., \boldsymbol{X}_n$ with density function $f(\boldsymbol{x}_i)$. Thus, for $i = 1, ..., n$ we can write

$$\boldsymbol{X}_i | Z_{ik} = 1 \sim F_k$$

where $F(\boldsymbol{X}_i)$ is the common distribution function of the n independent $\boldsymbol{X}_i$ corresponding to the mixture density $f(\boldsymbol{X}_i)$. Now, the observations $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n$ can be viewed as being incomplete by simply considering unknown (*missing*) the n component-label vectors $\boldsymbol{z}_1, \boldsymbol{z}_2, ..., \boldsymbol{z}_n$. That is, the complete data could be represented as follow:

$$\mathbf{Y} = (\boldsymbol{x}, \boldsymbol{z})$$

where

$$\boldsymbol{x} = (\boldsymbol{x}_1,\, \boldsymbol{x}_2, ...,\, \boldsymbol{x}_n)^{'}$$

is the observed but incomplete $n \times D$ matrix and

$$\boldsymbol{z} = (\boldsymbol{z}_1,\, \boldsymbol{z}_2, ...,\, \boldsymbol{z}_n)^{'}$$

is the unobserved $n \times K$ matrix of component indicators $\boldsymbol{z}_1,\, \boldsymbol{z}_2, ...,\, \boldsymbol{z}_n$; thus $\mathbf{Y}$ is the $n \times (D + K)$ matrix of complete data.

A way to further extend this notation is to consider some of the n observations $\boldsymbol{x}_i$ missing, but we will not focus on this topic, thus all the data points will be treated as completely known.

The $\boldsymbol{z}_1,\, \boldsymbol{z}_2, ...,\, \boldsymbol{z}_n$ in the above formulation are to be intended as realizations of the random vectors $\boldsymbol{Z}_1,\, \boldsymbol{Z}_2, ...,\, \boldsymbol{Z}_n$, so that in the case of independent data we can assume they are distributed as follow:

$$\boldsymbol{Z}_1,\, \boldsymbol{Z}_2, ...,\, \boldsymbol{Z}_n \overset{i.i.d.}{\sim} Mult_K\,(1,\, \boldsymbol{\pi})$$

where $\boldsymbol{\pi} = (\pi_1,\, ...,\, \pi_K)$. For $k = 1,\, ...,\, K$ the k-th mixing proportion π_k is meant to be the prior probability that an observation $\boldsymbol{x}_i$ belongs to the k-th cluster. On the other hand, the posterior probability that an observation i belongs to the k-th component, say $\boldsymbol{\tau}_k$ $k = 1,\, ...,\, K$ and $i = 1,\, ...,\, n$ can be expressed as:

$$\boldsymbol{\tau}_k\,(\boldsymbol{x}_i) = \Pr\,(i \in k\text{th component} |\, \mathbf{x}_i)$$

$$= \Pr\,(Z_{ik} = 1 |\, \mathbf{x}_i) = \frac{\pi_k f_k\,(\boldsymbol{x}_i)}{f\,(\boldsymbol{x}_i)} = \frac{\pi_k f_k\,(\boldsymbol{x}_i)}{\sum_{g=1}^{K} \pi_g f_g\,(\boldsymbol{x}_i)}$$

So, the posterior probability that an observation $\boldsymbol{x}_i$ belongs to the k-th component $\boldsymbol{\tau}_k$ is equal to the prior probability of k-th component membership multiplied by the k-th component density over the marginal density of $\boldsymbol{x}_i$. Intuitively, this tells us that, among the K densities, the one wich best accounts for a specific $\boldsymbol{x}_i$ (i.e. with the highest value) will determine the highest posterior probability $\boldsymbol{\tau}_k$. It is straightforward to show that the $\boldsymbol{\tau}_k$ sums to 1:

$$\sum_{k=1}^{K} \boldsymbol{\tau}_k\,(\boldsymbol{x}_i) = \frac{\sum_{k=1}^{K} \pi_k f_k\,(\boldsymbol{x}_i)}{f\,(\boldsymbol{x}_i)} = \frac{\sum_{k=1}^{K} \pi_k f_k\,(\boldsymbol{x}_i)}{\sum_{k=1}^{K} \pi_k f_k\,(\boldsymbol{x}_i)} = 1$$

If our aim is to cluster an observed random sample into K components, we have to infer the corresponding component labels $\boldsymbol{z}_1,\ \boldsymbol{z}_2, ...,\ \boldsymbol{z}_n$ on the basis of the observed data points $\boldsymbol{x}_1,\ \boldsymbol{x}_2, ...,\ \boldsymbol{x}_n$. By this way, it is straightforward to provide a *hard* clustering of the data. It suffices to choose some criterion to assign each $\boldsymbol{x}_i$ to one and only one component among the K available in the mixture (in the above notation, every $\boldsymbol{x}_i$ has to belong only to the k-th cluster for which the k-entry of $\boldsymbol{Z}_i$ is equal to 1, being 0 all the other $K-1$ entries). For instance, a naive solution could be to associate i to the component for which the posterior probability results the highest. In other words, we are estimating the i-th component label $\boldsymbol{z}_i$ through $\hat{\boldsymbol{z}}_i$, based on the rule

$$\begin{aligned} \hat{Z}_{ik} &= 1 \qquad \text{if } k = \arg\max_g \boldsymbol{\tau}_g\left(\boldsymbol{x}_i\right) \\ &= 0 \qquad \text{otherwise} \end{aligned}$$

for $k = 1,\ ...,\ K$ and $i = 1,\ ...,\ n$. Actually, this allocation criterion coincide with the so-called plug-in sample version of the Bayes rule, see McLachlan and Peel (2000), and it is often referred to as *maximum a posteriori* (*MAP*) rule. It is also worth noticing that

$$\boldsymbol{\tau}_{ik} = E\left(Z_{ik}|\boldsymbol{x}_i\right)$$

that is the posterior expectation of the unknown component indicator.

As we will see in the next section, a notable feature of the incomplete-data formulation for mixture problems is that it leads to implement the EM algorithm, for maximum likelihood estimation (MLE).

Finally, until now we've supposed a clustering scenario suitable for i.i.d. datasets. When the assumption $\boldsymbol{Z}_1,\ \boldsymbol{Z}_2, ...,\ \boldsymbol{Z}_n \overset{i.i.d.}{\sim} Mult_K\left(1,\ \boldsymbol{\pi}\right)$ holds, Titterington (1990) introduced the term "hidden multinomial " for the mixture model. On the other hand, if data are not independent different approaches, such as those based on Hidden Markov Chains in the longitudinal framework, can be adopted. In the particular case where the component-label vectors $\boldsymbol{z}_1,\ \boldsymbol{z}_2, ...,\ \boldsymbol{z}_n$ refer to some two-dimensional lattice for which a Markov random field can be considered, the model can be described as a Hidden Markov Random Field Model. The list of possibile applications of Mixture Models is huge, see for example Alfò and Viviani (2015), where mixtures of structured models are discussed. We will not pursue these topics, which are beyond our scope.

3.3 EM algorithm for Mixture Models

The advent of the EM algorithm greatly contributed to the diffusion of maximum likelihood estimation. In this section we present two formulations of the EM algorithm: the first is direct and intuitive, the second is based on the incomplete data formulation. Note that in the following, for the sake of notation simplicity, we will denote the ML estimation by $\hat{\boldsymbol{\psi}}$.

3.3.1. An early version of the EM algorithm

First, let us consider for a sample of n i.i.d. observations $\boldsymbol{x}_i$, $i = 1, ..., n$, the above mixture density $f(\boldsymbol{x}_i; \boldsymbol{\psi})$, defined by

$$f(\boldsymbol{x}_i; \boldsymbol{\psi}) = \sum_{k=1}^{K} \pi_k f_k(\boldsymbol{x}_i; \boldsymbol{\theta}_k)$$

where $\boldsymbol{\psi}$ is the set of unknow parameters associated to the Mixture Model, written as

$$\boldsymbol{\psi} = \{\pi_1, ..., \pi_{K-1}; \boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_K\}$$

where $\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_K$ are supposed to be distinct, and $\boldsymbol{\Omega}$ indicates the specified parameter space. The likelihood and the log-likelihood functions are defined accordingly

$$L(\boldsymbol{\psi}) = \prod_{i=1}^{n} f(\boldsymbol{x}_i; \boldsymbol{\psi}) = \prod_{i=1}^{n} \sum_{k=1}^{K} \pi_k f_k(\boldsymbol{x}_i; \boldsymbol{\theta}_k)$$

$$\ell(\boldsymbol{\psi}) = \log L(\boldsymbol{\psi}) = \sum_{i=1}^{n} \log f(\boldsymbol{x}_i; \boldsymbol{\psi}) = \sum_{i=1}^{n} \log \left\{ \sum_{k=1}^{K} \pi_k f_k(\boldsymbol{x}_i; \boldsymbol{\theta}_k) \right\}$$

the corresponding ML estimation $\hat{\boldsymbol{\psi}} = \arg\max_{\boldsymbol{\psi}} \ell(\boldsymbol{\psi})$, is defined as the solution to the following *likelihood equation*

$$\frac{\boldsymbol{\partial} \log L(\boldsymbol{\psi})}{\boldsymbol{\partial \psi}} = 0$$

In the '60s some authors, e.g. Wolfe (1965), Hasselblad (1966) and Day (1969) already solved the *likelihood equation* noticing that the elements in $\hat{\boldsymbol{\psi}}$ have to satisy two equalities

$$\hat{\pi}_k = \frac{\sum_{i=1}^{n} \tau_k \left(\boldsymbol{x}_i;\, \hat{\boldsymbol{\psi}} \right)}{n}$$

$$\sum_{i=1}^{n} \tau_k \left(\boldsymbol{x}_i;\, \hat{\boldsymbol{\psi}} \right) \frac{\partial \log f_k \left(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k \right)}{\partial \boldsymbol{\theta}_k} = 0$$

where

$$\tau_k \left(\boldsymbol{x}_i;\, \boldsymbol{\psi} \right) = \frac{\pi_k f_k \left(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k \right)}{\sum_{g=1}^{K} \pi_g f_g \left(\boldsymbol{x}_i;\, \boldsymbol{\theta}_g \right)}$$

denotes the posterior probability that the i-th observation belongs to the k-th component of mixture (see the above section). In the same papers it is recognized that the above two solutions can be embedded within an iterative process to find a *likelihood* estimate. The idea is to start with some initial value for $\boldsymbol{\psi}$, say $\boldsymbol{\psi}^{(0)}$, plug it in the two equalities to get a new estimate for $\boldsymbol{\psi}$, say $\boldsymbol{\psi}^{(1)}$, which in turn will be put in the same equalities, thus generating a new solution, $\boldsymbol{\psi}^{(2)}$, until convergence (see below). This kind of "direct" approach to the EM algorithm is essentially the same developed by Dempster et al. (1977), who explicitly derived the general increasing monotonicity of the solutions provided at every step by the algorithm. Dempster et al. (1977) cited a work by Haberman (1976), where the Author shows the same monotonic behaviour, but in a less general framework. Similar results can be found in Baum and Eagon (1967), who derive a similar result in hidden Markov chains.

3.3.2 The EM algorithm as a solution to the incomplete data problem

As we already observed, the most natural framework to define the EM algorithm is in the incomplete data framework, where the observations $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n$

are viewed as a part of the complete data that include the unknown (*missing*) corresponding component indicators $\boldsymbol{z}_1,\ \boldsymbol{z}_2, ...,\ \boldsymbol{z}_n$. Thus, for $i = 1,\ ...,\ n$, the complete-data could be represented as follow:

$$\mathbf{y}_i = (\boldsymbol{x}_i, \boldsymbol{z}_i)$$

The $\boldsymbol{z}_1,\ \boldsymbol{z}_2, ...,\ \boldsymbol{z}_n$ in the above formulation are to be intended as realizations of the random vectors $\boldsymbol{Z}_1,\ \boldsymbol{Z}_2, ...,\ \boldsymbol{Z}_n$, so that in the case of independent data they are assumed to be unconditionally multinomial distributed:

$$\boldsymbol{Z}_1,\ \boldsymbol{Z}_2, ...,\ \boldsymbol{Z}_n \overset{i.i.d.}{\sim} Mult_K\left(1,\ \boldsymbol{\pi}\right)$$

With this assumption, the complete-data likelihood is

$$L_c\left(\boldsymbol{\psi}\right) = f\left(\boldsymbol{x},\ \boldsymbol{z}|\ \boldsymbol{\psi}\right) = \prod_{i=1}^{n}\prod_{k=1}^{K} \pi_k^{z_{ik}} f_k\left(\boldsymbol{x}_i;\ \boldsymbol{\theta}_k\right)^{z_{ik}}$$

and the log-likelihood is given by

$$\ell_c\left(\boldsymbol{\psi}\right) = \sum_{i=1}^{n}\sum_{k=1}^{K} \log\left\{\pi_k^{z_{ik}}\left[f_k\left(\boldsymbol{x}_i;\ \boldsymbol{\theta}_k\right)\right]^{z_{ik}}\right\} = \sum_{i=1}^{n}\sum_{k=1}^{K} \log\left\{\pi_k f_k\left(\boldsymbol{x}_i;\ \boldsymbol{\theta}_k\right)\right\}^{z_{ik}} =$$

$$= \sum_{k=1}^{K}\sum_{i=1}^{n} z_{ik}\left\{\log \pi_k + \log f_k\left(\boldsymbol{x}_i;\ \boldsymbol{\theta}_k\right)\right\}$$

We have all the ingredients needed to formulate an EM-type algorithm. The acronym EM shows the two steps in the algorithm: E stands for expectation, while M stands for maximization. The first (E step) is intended to provide the conditional estimate of the unobservable data, $\boldsymbol{z}_1,\ \boldsymbol{z}_2, ...,\ \boldsymbol{z}_n$. To this end, the expectation of the complete-data log likelihood, $\ell_c\left(\boldsymbol{\psi}\right)$, is computed conditionally to data $\boldsymbol{X}$, with $\boldsymbol{\psi}$ fixed at the current estimate, say $\boldsymbol{\psi}^{(t)}$. So, at the generic E step, the following expectation is computed

$$Q\left(\boldsymbol{\psi}^{(t+1)};\ \boldsymbol{\psi}^{(t)}\right) = E_{\boldsymbol{\psi}^{(t)}}\left\{\ell_c\left(\boldsymbol{\psi}\right)|\mathbf{X}\right\}$$

Note that the complete-data log likelihood $\ell_c(\boldsymbol{\psi})$ is linear in the unobservable data z_{ik}, so that the E step, at the generic iteration $t+1$, implies the computation of the current conditional expectation of Z_{ik} given the observed data $\mathbf{X}$:

$$E_{\boldsymbol{\psi}^{(t)}}(Z_{ik}|\mathbf{X}) = \Pr_{\boldsymbol{\psi}^{(t)}}(Z_{ik}=1|\mathbf{X}) = \frac{\pi_k^{(t)} f_k\left(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k^{(t)}\right)}{f\left(\boldsymbol{x}_i;\, \boldsymbol{\psi}_k^{(t)}\right)} =$$

$$= \frac{\pi_k^{(t)} f_k\left(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k^{(t)}\right)}{\sum_{g=1}^{K} \pi_g^{(t)} f_g\left(\boldsymbol{x}_i;\, \boldsymbol{\theta}_g^{(t)}\right)} = \tau_k\left(\boldsymbol{x}_i;\, \boldsymbol{\psi}^{(t)}\right) = \tau_{ik}^{(t+1)}$$

Now, replacing z_{ik} by its estimate in the $\ell_c(\boldsymbol{\psi})$ yields to

$$Q\left(\boldsymbol{\psi};\, \boldsymbol{\psi}^{(t)}\right) = \sum_{k=1}^{K}\sum_{i=1}^{n} \tau_k\left(\boldsymbol{x}_i;\, \boldsymbol{\psi}^{(t)}\right)\{\log \pi_k + \log f_k(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k)\} =$$

$$= \sum_{k=1}^{K}\sum_{i=1}^{n} \tau_{ik}^{(t+1)}\{\log \pi_k + \log f_k(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k)\}$$

This is the expression to maximize in the M step, to gain an updated estimate for $\boldsymbol{\psi}$, say $\boldsymbol{\psi}^{(t+1)}$ that will be plugged in the successive E step and so on until convergence (see below). In particular, the M step provides (if possible) the global maximizer of $Q\left(\boldsymbol{\psi};\, \boldsymbol{\psi}^{(t)}\right)$ for $\boldsymbol{\psi}$ over the parameter space $\boldsymbol{\Omega}$, which is the required updated estimate $\boldsymbol{\psi}^{(t+1)}$, conditional on the weigths $\tau_{ik}^{(t+1)}$.

Note that $Q\left(\boldsymbol{\psi};\, \boldsymbol{\psi}^{(t)}\right) = \sum_{k=1}^{K}\sum_{i=1}^{n} \tau_{ik}^{(t+1)}\{\log \pi_k + \log f_k(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k)\}$ consists of two separated parts, so that maximization of $Q\left(\boldsymbol{\psi};\, \boldsymbol{\psi}^{(t)}\right)$ may be accomplished in two separated operations. For the first term $\log \pi_k$, which includes the mixing proportions π_k, if the $\boldsymbol{z}_1,\, \boldsymbol{z}_2, ...,\, \boldsymbol{z}_n$ were known, then the maximization would give the standard maximum likelihood estimate for a multinomial case, i.e.

$$\hat{\pi}_k = \frac{\sum_{i=1}^{n} z_{ik}}{n}$$

Instead, in the general case of unknown $\boldsymbol{z}_1,\, \boldsymbol{z}_2, ...,\, \boldsymbol{z}_n$, the estimates $\hat{z}_{ik} = \tau_k\left(\boldsymbol{x}_i;\, \boldsymbol{\psi}^{(t)}\right)$ are used to get

$$\pi_k^{(t+1)} = \frac{\sum_{i=1}^{n} \tau_k\left(\boldsymbol{x}_i;\, \boldsymbol{\psi}^{(t)}\right)}{n}$$

Note that in this operation each data point contributes to define the probability of component membership π_k. In other words, each data point has a certain degree of membership for all the K clusters, thus implying a sort of *fuzzy* logical in this framework.

The second term to maximize in $Q\left(\boldsymbol{\psi};\, \boldsymbol{\psi}^{(t)}\right)$, $\log f_k\left(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k\right)$, is the one that is used to get an updated estimate $\boldsymbol{\theta}_k^{(t+1)}$ solving the weighted likelihood equation

$$\sum_{i=1}^{n} \tau_{ik}^{(t+1)} \left(\boldsymbol{x}_i;\, \boldsymbol{\psi}^{(t)}\right) \frac{\partial \log f_k\left(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k\right)}{\partial \boldsymbol{\theta}_k} = 0$$

The updated estimates define the new global parameter vector $\boldsymbol{\psi}^{(t+1)}$, that will be used in the E step, to calculate $\tau_{ik}^{(t+2)}$ and so on until convergence (see Hathaway (1986) where, interpreting the EM algorithm for mixture problems as a coordinate descent method, it is shown the relationship of EM to some clustering techniques and global convergence properties are discussed outside of incomplete data framework).

Note that in practice a root for the equation above may exist in closed form, as in the case of Gaussian mixtures (see below). The convergence rule, instead, is of the following type

$$L\left(\boldsymbol{\psi}^{(t+1)}\right) - L\left(\boldsymbol{\psi}^{(t)}\right) < \varepsilon$$

with $\varepsilon > 0$ as an arbitrary small constant. This choice makes sense, since Dempster et al. (1977) showed that

$$L\left(\boldsymbol{\psi}^{(t+1)}\right) \geqslant L\left(\boldsymbol{\psi}^{(t)}\right)$$

that is, the likelihood function is non-decreasing in subsequent steps of the EM algorithm. However, see Lindstrom and Bates (1988) criteria where the relative changes of the parameter vector and/or log-likelihood indicate lack of progress rather than actual convergence, see also McNicholas et al. (2010). Using a general framework Wu (1983) showed that convergence in likelihood sequence $t = 0,\, 1,\, 2, ...$ does not necessarily imply the underlying convergence of the sequence $\boldsymbol{\psi}^{(t)}$, in the EM algorithm. Mengersen, Robert and Titterington (2011) noted that some problems in the convergence of sequence $\boldsymbol{\psi}^{(t)}$ may occur if $f\left(\boldsymbol{x};\, \boldsymbol{\psi}\right)$ contains a ridge. On the identification of ridges and the related problem of saddlepoints, see the interesting paper by Ray and Lindsay (2005), where it's been proved that "for any D-dimensional,

K-component normal mixture, we can define a dimensional surface, which is guaranteed to include all the critical points (modes, antimodes and saddlepoints) of the D-dimensional mixture density", that is the $(K-1)$-dimensional ridgeline manifold. Finally we note that, although not often mentioned, a first attempt to realize an EM-type of algorithm can be found in Newcomb (1886), see Stigler (1973) and McLachlan and Krishnan (2007).

3.4 Gaussian Mixture Models

In this section, we will discuss the Gaussian Mixture Models and some related issues, such as maximum likelihood estimation. To correctly address this topic, we first introduce the general framework of parametric Mixture Models, with Gaussian Mixtures as a particular case. Finally, we introduce the incomplete data formulation of the EM algorithm for Gaussian Mixture Models.

3.4.1 Parametric Mixture Models

In a clustering context, the component densities $f_k(\boldsymbol{x}_i)$ of mixture are often specified as members of some parametric family. By this way, we need to modify the above formulation in order to encompass parameters involved by the specified densities; in place of the generic $f_k(\boldsymbol{x}_i)$ we will adopt $f_k(\boldsymbol{x}_i; \boldsymbol{\theta}_k)$, where $\boldsymbol{\theta}_k$ is the vector of all the unknow parameters associated to the k-th component of the Mixture Model. Thus, the previous marginal density $m(\boldsymbol{x}_i)$ is now rewritten as follows:

$$m(\boldsymbol{x}_i) = f(\boldsymbol{x}_i; \boldsymbol{\psi}) = \sum_{k=1}^{K} \pi_k f_k(\boldsymbol{x}_i; \boldsymbol{\theta}_k)$$

where $\boldsymbol{\psi}$ is the set of unknown parameters associated to the Mixture Model,

$$\boldsymbol{\psi} = \{\pi_1, ..., \pi_{K-1}; \boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_K\}$$

In this framework, we adopt $\boldsymbol{\Omega}$ to indicate the *a priori* specified parameter space for $\boldsymbol{\psi}$. Note that $\boldsymbol{\Omega}$ depends on 3 factors we know or fix in advance: 1. the dimension D of the data points; 2. the parameter space of the involved parametric family; 3.

the number of components K, which is consider fixed in the above formulation. In the specification of $\boldsymbol{\psi}$ we have ruled out the last element of the mixing proportions π_K, since $\sum_{k=1}^{K} \pi_k = 1$, a unity constraint.

In many applications, the component densities $f_k(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k)$ are supposed to belong to the same parametric family. In this case, we may remove the subscript k from the $f_k(\cdot)$, and the mixture density $f(\boldsymbol{x}_i;\, \boldsymbol{\psi})$ is written as

$$f(\boldsymbol{x}_i;\, \boldsymbol{\psi}) = \sum_{k=1}^{K} \pi_k f(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k)$$

where $f(\cdot;\, \boldsymbol{\theta})$ denotes a generic member of the parametric family

$$\{f(\boldsymbol{x}_i;\, \boldsymbol{\theta}) :\ \boldsymbol{\theta} \in \boldsymbol{\Theta}\}$$

where $\boldsymbol{\Theta}$ represents the parameter space for $\boldsymbol{\theta}$.

Although the above formulation is standard in Mixture Models, it hides a general problem of identifiability in the family $\{f(\boldsymbol{x}_i;\, \boldsymbol{\psi}) :\ \boldsymbol{\psi} \in \boldsymbol{\Omega}\}$. To discuss this common issue, we first remind the usual definition of identifiability for a parametric family of densities, and then we show that it does not hold in the Mixture Model framework.

A parametric family of densities $\{f(\boldsymbol{x}_i;\, \boldsymbol{\psi}) :\ \boldsymbol{\psi} \in \boldsymbol{\Omega}\}$ is said to be identifiable if different values of $\boldsymbol{\psi}$ imply distinct members of the same parametric family, that is

$$f(\boldsymbol{x}_i;\, \boldsymbol{\psi}) = f(\boldsymbol{x}_i;\, \boldsymbol{\psi}^*)$$

if and only if $\boldsymbol{\psi} = \boldsymbol{\psi}^*$.

Then, turn to the case we are discussing (mixture components belonging to the same parametric family); let us suppose that $\boldsymbol{\psi}_c$ is a value of the global parameter set $\boldsymbol{\psi}$ such that we have

$$f(\boldsymbol{x}_i;\, \boldsymbol{\psi}_c)$$

as the density evaluated at $\boldsymbol{\psi}_c$. Define $\boldsymbol{\Omega}_c$ as the set of $\boldsymbol{\psi}$ in $\boldsymbol{\Omega}$ such that a.c. $f(\boldsymbol{x}_i;\, \boldsymbol{\psi}) = f(\boldsymbol{x}_i;\, \boldsymbol{\psi}_c)$. Now, it is easy to prove that the set $\boldsymbol{\Omega}_c$ include also elements such that $\boldsymbol{\psi}^* \neq \boldsymbol{\psi}_c$, thus implying unidentifiability of $\boldsymbol{\psi}$. This occurs because a permutation of the K component labels in $\boldsymbol{\psi}_c$ produces the same value for the density function $f(\boldsymbol{x}_i;\, \boldsymbol{\psi})$ for all $\boldsymbol{x}_i$. That is, $f(\boldsymbol{x}_i;\, \boldsymbol{\psi})$ is invariant under the $K!$ permutations of the labels in $\boldsymbol{\psi}$. To show this, remind that $f(\boldsymbol{x}_i;\, \boldsymbol{\psi})$ is defined as

$$f\left(\boldsymbol{x}_i;\, \boldsymbol{\psi}\right) = \sum_{k=1}^{K} \pi_k f\left(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k\right)$$

i.e. $f\left(\boldsymbol{x}_i;\, \boldsymbol{\psi}\right)$ is a convex combination, so that the sum is not altered by any of the $K!$ permutations of the terms $\pi_k f\left(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k\right)$, while $\boldsymbol{\psi}$ changes througout the $K!$ permutations ($\boldsymbol{\psi}$ it is a vector of elements, so that every change in ordering determine a different element $\boldsymbol{\psi} \in \boldsymbol{\Omega}$). In other words, we have that the model $\left\{f\left(\boldsymbol{x}_i;\, \boldsymbol{\psi}\right) :\ \boldsymbol{\psi} \in \boldsymbol{\Omega}\right\}$ is not identifiable.

This is probably the most common identifiability problem in clustering, the so called *label switching* problem. Since it is not a special issue to the Mixture Model framework, but rather an ubiquitous one in clustering, it is appropriate to modify the identifiability definition, in order to overcome this problem. To this end, let us fix the number of components K, and suppose

$$f\left(\boldsymbol{x}_i;\, \boldsymbol{\psi}\right) = \sum_{k=1}^{K} \pi_k f_k\left(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k\right)$$

and

$$f\left(\boldsymbol{x}_i;\, \boldsymbol{\psi}^*\right) = \sum_{k=1}^{K} \pi_k^* f_k\left(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k^*\right)$$

are any two elements in a parametric family of mixtures densities $\left\{f\left(\boldsymbol{x}_i;\, \boldsymbol{\psi}\right) :\ \boldsymbol{\psi} \in \boldsymbol{\Omega}\right\}$. Then, for $\boldsymbol{\psi} \in \boldsymbol{\Omega}$ the family $\left\{f\left(\boldsymbol{x}_i;\, \boldsymbol{\psi}\right) :\ \boldsymbol{\psi} \in \boldsymbol{\Omega}\right\}$ is identifiable if

$$f\left(\boldsymbol{x}_i;\, \boldsymbol{\psi}\right) = f\left(\boldsymbol{x}_i;\, \boldsymbol{\psi}^*\right)$$

for almost all $\boldsymbol{x}_i \in \mathbb{R}^D$ implies that we can find a permutation such that

$$\pi_k = \pi_k^* \quad \text{and} \quad f_k\left(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k\right) = f_k\left(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k^*\right) \qquad k = 1, ..., K$$

To overcome *label switching* problem, it is possible to choose different constraints on $\boldsymbol{\psi}$, inducing unique ordering in the labels. One solution suggested in Aitkin and Rubin (1985) is to adopt the following constraint on the mixing proportions:

$$\pi_1 \leqslant \pi_2 \leqslant ... \leqslant \pi_K$$

implementing the Mixture Model estimation without any further constraint. For a more in-depth and detailed discussion on Mixture Models identifiability see Titterington, Smith and Makov (1985).

3.4.2 Gaussian Mixture Models

In our proposal (see Chapter 5), the K component densities of mixture $f(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k)$ are assumed to be members of the Gaussian family. In the multivariate case, with $\boldsymbol{x}_i \in \mathbb{R}^D$, the generic k-th component density can be therefore written as

$$f(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k) = \phi(\boldsymbol{x}_i;\, \boldsymbol{\mu}_k,\, \boldsymbol{\Sigma}_k)$$

where

$$\phi(\boldsymbol{x}_i;\, \boldsymbol{\mu}_k;\, \boldsymbol{\Sigma}_k) = (2\pi)^{-\frac{D}{2}} |\boldsymbol{\Sigma}_k|^{-\frac{1}{2}} \exp\left\{ -\frac{1}{2} (\boldsymbol{x}_i - \boldsymbol{\mu}_k)' \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_k) \right\}$$

is the multivariate (D-dimensional) Gaussian density, $\boldsymbol{\mu}_k$ is the D dimensional vector of means and $\boldsymbol{\Sigma}_k$ is the $D \times D$ dimensional covariance matrix, $k = 1, ..., K$. In the case of homoschedastic components, the covariances matrices $\boldsymbol{\Sigma}_k$ are constant across components, that is $\boldsymbol{\Sigma}_k = \boldsymbol{\Sigma}$ $k = 1, ..., K$.

Once we have specified the parametric form of the kernel density, we have to discuss how to get estimates of model parameters in this case. Apart from the method of moments, which started with Pearson (1894), there've been various approaches to estimation of mixture distributions. As pointed out in McLachlan and Peel (2000): "Over the years, a variety of approaches have been used to estimate mixture distributions. They include graphical methods, method of moments, minimum-distance methods, maximum likelihood, and Bayesian approaches. As surmised by Titterington (1996), perhaps the main reason for the huge literature on estimation methodology for mixtures is the fact that explicit formulas for parameter estimates are typically not available. For example, the MLE for the mixing proportions and the component means and variances/covariances cannot be written down in closed form for normal mixtures".

This last issue is of particular interest, since we will focus on Maximum Likelihood estimation for Gaussian mixtures in a clustering framework.

3.4.3 Features and issues with Gaussian Mixture Models

Note that all the issues in the discussion above refer to a general parametric family of mixture densities $\{f(\boldsymbol{x}_i;\, \boldsymbol{\psi}) :\ \boldsymbol{\psi} \in \boldsymbol{\Omega}\}$ not a specific parametric form. Instead, our main interest is on Gaussian mixture distributions. So, in the following, we will deal with some issues (identifiability and other) with a specific focus on this parametric family.

First of all, a relevant question in MLE estimation for Gaussian distributions arises since the likelihood $L(\boldsymbol{\psi})$ is unbounded. In such a case, obviously, the MLE $\hat{\boldsymbol{\psi}}_{ML}$ may not exist as a global maximizer of the likelihood function (it may still exist as a local maximizer). To show this point, consider a univariate, K component based, Gaussian Mixture Model

$$f(x_i;\, \boldsymbol{\psi}) = \sum_{k=1}^{K} \pi_k f(x_i;\, \boldsymbol{\theta}_k)$$

where

$$f(x_i;\, \boldsymbol{\theta}_k) = \phi(x_i;\, \mu_k,\, \sigma_k)$$

and

$$\phi(x_i;\, \mu_k,\, \sigma_k) = \left(2\pi\sigma_k^2\right)^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}\left(\frac{x_i - \mu_k}{\sigma_k}\right)^2\right\}$$

As reported by Kiefer and Wolfowitz (1956), if we set any of the K means, say the k-th, μ_k equal to any of the n observation x_i, i.e. if we have $x_i = \mu_k$, and let the corresponding variance σ_k^2 go to zero, the likelihood will tend to infinity. Note that this problem occurs for any sample size n, thus compromising consistency of the maximum likelihood estimator $\hat{\boldsymbol{\psi}}_{ML}$.

Several papers have dealt with this issue, see for example Chen, Tan and Zhang (2008) where a penalized likelihood method is discussed and conditions provided to restore the optimal properties of the likelihood in the above case of unboundness. Zucchini and MacDonald (2009) suggest to "replace each density value in the likelihood by the probability of the interval corresponding to the recorded value"; Scholz (2006) states that a maximum likelihood estimate does not exist (see Scholz (2006), example 2).

As pointed out in McLachlan and Peel (2000), the unboundness and absence of a global maximizer for the likelihood function in heteroschedastic Gaussian mixtures

is not an intractable problem "as the essential aim of likelihood estimation is to find a sequence of roots of the likelihood equation that is consistent, and hence efficient if the usual regularity conditions hold".

On this topic, according to Lehmann (1980), for an i.i.d. sample, as $n \to \infty$ "there exists under suitable regularity conditions a sequence of solutions of the likelihood equation that is consistent and asymptotically efficient. However, this consistent solution is not necessarily the maximum likelihood estimate. Likelihood estimation should therefore emphasize the determination of a consistent sequence of solutions of the likelihood equations rather than maximizing the likelihood".

Here, the real question seems to be rather on how identifying this sequence of solutions, since the likelihood equation will have in general multiple roots in Mixture Models framework. McLachlan and Peel (2000) state that "even if it were known that there exists a sequence of roots of the likelihood equation with the desired asymptotic properties, there is the problem of identifying this sequence".

Nevertheless, some results on this issue are available. For instance, in the univariate Gaussian mixture case, we may impose the following conditions on $\boldsymbol{\psi}$

$$\pi_k \neq 0 \qquad k = 1, ..., K$$

$$(\mu_g, \sigma_g) \neq (\mu_k, \sigma_k) \qquad \forall g \neq k = 1, ..., K$$

In this way, we could ensure the existence of a sequence of roots corresponding to local maxima that is asymptotically normal and efficient; when the consistence is entailed these constraints are unnecessary, see McLachlan and Peel (2000). In the case one of the two constraints does not hold, even in the multivariate case, we would be rejected in the case analyzed in Feng and McCulloch (1996), where they prove that although $\boldsymbol{\psi}_t$ lies on the boundary of the parameter space $\boldsymbol{\Omega}$, and it is in a unidentifiable subset $\boldsymbol{\Omega}_t$, $\hat{\boldsymbol{\psi}}_{ML}$ converges to the same $\boldsymbol{\Omega}_t$, which contains the true value $\boldsymbol{\psi}_t$.

Another set of constraints, which is intended to avoid the singularity (when $x_i = \mu_k$ and $\sigma_k^2 \to 0$) is written as

$$\min_{g,k} (\sigma_g / \sigma_k) \geqslant C > 0 \qquad \forall g \neq k = 1, ..., K$$

see Hathaway (1985). The same type of condition was yet mentioned in Dennis (1981). According to this condition, Hathaway (1985) identifies a reduced parameter space $\boldsymbol{\Omega}_C$ of the form

$$\boldsymbol{\Omega}_C = \left\{ \boldsymbol{\psi} \in \boldsymbol{\Omega} : \min_{g,k} (\sigma_g / \sigma_k) \geqslant C > 0, \qquad g \neq k = 1, ..., K \right\}$$

where $\boldsymbol{\Omega}$ clearly denotes the unconstrained parameter space. Provided that the true value of $\boldsymbol{\psi}$, $\boldsymbol{\psi}_t$, belongs to $\boldsymbol{\Omega}_C$, Hathaway (1985) showed that considering $C \in (0,\, 1]$ and taking $n > K$, the global maximizer $\hat{\boldsymbol{\psi}}_C$ of $L(\boldsymbol{\psi})$ over $\boldsymbol{\Omega}_C$ exists and it is strongly consistent for $\boldsymbol{\psi}$. So, in a way analogous to Redner (1981), Hathaway (1985) works on a reduced parameter space to avoid problems related to the likelihood function. Redner (1981) quotient space was another kind of reduced parameter space. However, there is still the issue of choosing a value for C, such that the true value $\boldsymbol{\psi}_t$ satisfies the above constraints.
Another interesting issue addressed in Hathaway (1985) is a constrained version of the likelihood in the multivariate case, according to which all the eingenvalues of $\Sigma_g \Sigma_k^{-1}$ must be at least equal to some value $C > 0$, once again provided that true value $\boldsymbol{\psi}_t$ satisfies such a condition. Moreover, starting from the paper by Hathaway (1985), Rocci and Ingrassia (2007) developed a new set of constraints that can be applied directly in the EM algorithm. Finally, on the same topic it should be mentioned the paper by Chen (2017), where the consistency of ML estimates under mixture models is addressed in a general theoric framework, which encompasses different previous theorems such as Wald (1949), Kiefer and Wolfowitz (1956) and Redner (1981).

However, as we have been discussing above, it is always possible to establish a one-to-one correspondence between the mixture components and the clusters, so that this formulation finds a natural counterpart in clustering framework. In this sense, every Finite Mixture Model can be seen as a clustering method. For this reason, finite mixtures are often referred to as "model based clustering" approaches. According to McLachlan and Peel (2000) "it can be seen that this mixture likelihood-based approach to clustering is model based in that the form of each component density of an observation has to be specified in advance".

In the last decades, model-based clustering has been more and more considered. This is due essentially to two factors: the first is that model-based approaches provide well-defined mathematical tools and well-established statistical techniques, see Marriott (1974), Aitkin, Anderson, and Hinde (1981); the second is linked to the advent of EM algorithm, which makes likelihood estimation in the Mixture Models easier. To this point, we are going to dedicate next subsection.

3.4.3. The EM algorithm for Gaussian Mixture Models

As we noted before, we are mainly interested in Gaussian Mixtures. So, the above formulation of the EM algorithm should be slightly modified. Essentially we will replace $f(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k)$ by the corresponding Gaussian density $\phi(\boldsymbol{x}_i;\, \boldsymbol{\mu}_k,\, \boldsymbol{\Sigma}_k)$ and omit the subscript k on $f_k(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k)$, i.e. $f_k(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k) = f(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k) = \phi(\boldsymbol{x}_i;\, \boldsymbol{\mu}_k,\, \boldsymbol{\Sigma}_k)$, since the same parametric family is used for all the K components.

To this end, let us first recall the generic definition of a Gaussian Mixture for an i.i.d. sample with observations $\boldsymbol{x}_i \in \mathbb{R}^D$, $i = 1, ..., n$

$$f(\boldsymbol{x}_i;\, \boldsymbol{\psi}) = \sum_{k=1}^{K} \pi_k f(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k)$$

here $f(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k)$ is a multivariate Gaussian density

$$f(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k) = \phi(\boldsymbol{x}_i;\, \boldsymbol{\mu}_k,\, \boldsymbol{\Sigma}_k) == (2\pi)^{-\frac{D}{2}} |\boldsymbol{\Sigma}_k|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)\right\}$$

where $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ have been defined before. In the case of homoschedastic components, we would have $\boldsymbol{\Sigma}_k = \boldsymbol{\Sigma}$ for $k = 1, ..., K$.

No changes to the E step of the algorithm based on calculating the posterior probability that the i-th observation belongs to the kth component of mixture, $\tau_k(\boldsymbol{x}_i;\, \boldsymbol{\psi})$:

$$\hat{z}_{ik}^{(t)} = E_{\boldsymbol{\psi}^{(t)}}(Z_{ik}|\mathbf{X}) = \tau_{ik}^{(t)} = \frac{\pi_k \phi\left(\boldsymbol{x}_i;\, \boldsymbol{\mu}_k^{(t-1)},\, \boldsymbol{\Sigma}_k^{(t-1)}\right)}{\sum_{g=1}^{K} \pi_g \phi\left(\boldsymbol{x}_i;\, \boldsymbol{\mu}_g^{(t-1)},\, \boldsymbol{\Sigma}_g^{(t-1)}\right)}$$

$k = 1, ..., K$ and $i = 1, ..., n$.

Also, for the M step we will have the same estimate for π_k, that is

$$\pi_k^{(t)} = \frac{\sum_{i=1}^{n} \tau_{ik}^{(t)}}{n}$$

and we have to solve the equations

$$\sum_{i=1}^{n} \tau_{ik}^{(t)} \frac{\partial \log \phi(\boldsymbol{x}_i;\, \boldsymbol{\mu}_k,\, \boldsymbol{\Sigma}_k)}{\partial \boldsymbol{\mu}_k} = 0$$

$$\sum_{i=1}^{n} \tau_{ik}^{(t)} \frac{\partial \log \phi(\boldsymbol{x}_i;\, \boldsymbol{\mu}_k,\, \boldsymbol{\Sigma}_k)}{\partial \boldsymbol{\Sigma}_k} = 0$$

to obtain estimates for model parameters.

A notable feature in Gaussian Mixture Models is that such solutions exist in closed form, leading at the iteration t to the following estimates for $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$

$$\boldsymbol{\mu}_k^{(t)} = \frac{\sum_{i=1}^n \tau_{ik}^{(t)} \boldsymbol{x}_i}{\sum_{i=1}^n \tau_{ik}^{(t)}}$$

and

$$\boldsymbol{\Sigma}_k^{(t)} = \frac{\sum_{i=1}^n \tau_{ik}^{(t)} \left(\boldsymbol{x}_i - \boldsymbol{\mu}_k^{(t)}\right)\left(\boldsymbol{x}_i - \boldsymbol{\mu}_k^{(t)}\right)'}{\sum_{i=1}^n \tau_{ik}^{(t)}}$$

$k = 1, ..., K$. Note that these are the standard MLE estimates for vecor $\boldsymbol{\mu}_k$ and covariance matrix $\boldsymbol{\Sigma}_k$ except for the averaging on the posterior probability $\tau_{ik}^{(t)}$ (once again implying a sort of *fuzzy* logical in the estimates).

3.5 The R package Mclust

As we have already observed, in the last decades Gaussian Mixtures have been increasingly investigated, probably due to their huge flexibility of implementation. Recently there has been a development of suitable software packages for fitting Gaussian mixture. For instance, see McLachlan, Peel, Basford and Adams (1999), who proposed the *EMMIX* software (fitting also t-components mixtures), or Bouman, Shapiro, Cook, Atkins and Cheng (1997), who developed *Cluster*, for modeling Gaussian mixtures.

Here, we will focus on one of the most used software codes for fitting Gaussian mixture, namely *Mclust*, see Fraley and Raftery (1999). As it could be read in the Technical Report by Fraley and Raftery (2006), essentially *Mclust* "provides functions for parameter estimation via the *EM* algorithm for normal mixture models with a variety of covariance structures, and functions for simulation from these models. Also included are functions that combine model-based hierarchical clustering, *EM* for mixture estimation and the Bayesian Information Criterion (BIC) in comprehensive strategies for clustering, density estimation and discriminant analysis".

Actually, the first official version of *Mclust* has been described in Fraley and Raftery (1999), where are implemented some ideas yet proposed in Fraley and Raftery (1998), and, even farther back in time, in Banfield and Raftery (1993).

Probably, among the notable features of *Mclust*, the main one is the set of constraints applied to component specific covariance matrices Σ_k, $k = 1, ..., K$. Initially, in Banfield and Raftery (1993), the idea was to choose a sort of middle way between $\Sigma_k = \Sigma$, a constraint applied in Friedman and Rubin (1967), and $\Sigma_k = \Sigma_k$ with $k = 1, ..., K$, that is unconstrained estimation, see Scott and Symons (1971).

To put it formally, let us consider a K component mixture. In the multivariate case, for an i.i.d. sample with $\boldsymbol{x}_i \in \mathbb{R}^D$, the generic k-th component density can be written as follows

$$f\left(\boldsymbol{x}_i;\, \boldsymbol{\theta}_k\right) = \phi\left(\boldsymbol{x}_i;\, \boldsymbol{\mu}_k,\, \boldsymbol{\Sigma}_k\right)$$

$$\phi\left(\boldsymbol{x}_i;\, \boldsymbol{\mu}_k;\, \boldsymbol{\Sigma}_k\right) = (2\pi)^{-\frac{D}{2}} \left|\boldsymbol{\Sigma}_k\right|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}\left(\boldsymbol{x}_i - \boldsymbol{\mu}_k\right)' \boldsymbol{\Sigma}_k^{-1}\left(\boldsymbol{x}_i - \boldsymbol{\mu}_k\right)\right\}$$

Now, the corresponding log-likelihood can be written (up to a constant) as follows

$$\ell\left(\boldsymbol{\psi};\, \mathbf{X}\right) = \log L\left(\boldsymbol{\psi};\, \mathbf{X}\right) = -\frac{1}{2}\sum_{k=1}^{K}\left\{tr\left(S_k \boldsymbol{\Sigma}_k^{-1}\right) + n_k \log\left|\boldsymbol{\Sigma}_k\right|\right\}$$

where n_k is the cardinality of cluster k and S_k is the sample cross-product matrix for component k, that is

$$S_k = \sum_{i \in P_k}\left(\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k\right)\left(\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k\right)'$$

and P_k contains the indexes of observations belonging to cluster k, i.e. $P_k = \{i :\, z_{ik} = 1\}$.

Therefore, if we assume $\Sigma_k = \Sigma = \sigma^2 \boldsymbol{I}$, $k = 1, ..., K$ where $\boldsymbol{I}$ denotes the identity matrix, we obtain the already discussed sum of squares criterion (see section 1), that is the above log-likelihood would be maximized choosing the partition which minimizes $tr\left(S\right)$ with $S = \sum_{k=1}^{K} S_k$.

If we relax this assumption, assuming $\Sigma_k = \Sigma \neq \sigma^2 \boldsymbol{I}$ $k = 1, ..., K$, we apply the Friedman and Rubin (1967) criterion, i.e. minimization of $|S|$.

On the other hand, if we want to relax such hypotheses, and leave Σ_k totally unconstrained, we obtain the identity criterion $\Sigma_k = \Sigma_k$, the above log-likelihood will be maximized minimizing the term $\sum_{k=1}^{K} n_k \log\left|\frac{S_k}{n_k}\right|$.

In this framework, Banfield and Raftery (1993) aimed to "develop new criteria which are more general than that of Friedman and Rubin (1967), but based on more parsimonious models than that of Scott and Symons (1971)". In other words, the question is to model some features of cluster distributions (orientation, size and

shape) in a parsimonious way. To this end, Banfield and Raftery (1993) proposed a reparameterization based on an eigenvalue decomposition of the covariance matrix Σ_k. This will be considered by Fraley and Raftery (1999) and in further papers related to the development of the same *Mclust* package, and can be based on the following decomposition

$$\Sigma_k = D_k \Lambda_k D_k^{'}$$

where D_k is the eigenvectors matrix, while Λ_k is the diagonal eigenvalue of Σ_k. In particular, the eigenvectors in D_k establish the orientation of the principal components of Σ_k, while the shape of density contours is ruled by Λ_k. Moreover, in the case of D-variate Gaussian component densities, we may further specialized the decomposition above by posing $\Lambda_k = \lambda_k A_k$ where λ_k is the first eigenvalue of Σ_k, $k = 1, ..., K$ $A_k = \text{diag}\{\alpha_{dk}\}$ with $1 \geqslant \alpha_{1k} \geqslant \alpha_{2k} \geqslant ... \geqslant \alpha_{Dk} > 0$.

This last formulation helps to better understand how the eigenvalues matrix Λ_k impacts on the size and shape of the density contours. In fact, if the orientation of cluster k is determined by D_k, its volume and shape are established by λ_k and A_k respectively. In other words, while the first eigenvalue λ_k rules the size of cluster, the α_{dk}'s rules the shape of the cluster; if they are of similar magnitude the shape will be approximately hyperspherical. Otherwise, the cluster will be concentrated along the first few dimensions.

As pointed out in Banfield and Raftery (1993) "the criterion of Friedman and Rubin (1967) is based on the assumption that D_k, λ_k and A_k are the same for each cluster, while the criterion of Scott and Symons (1971) assumes them all to be different. By allowing some but not all of these quantities to vary between clusters, we obtain criteria that are appropriate for various intermediate situations". For example, it's straightforward to generalize the sum of squares criterion by posing $\Sigma_k = \lambda_k \boldsymbol{I}$. In fact, while all the K densities are spherical (Σ_k being a multiple of the identity matrix $\boldsymbol{I}$), the sizes of the clusters vary with the magnitude of λ_k. See Celeux and Govaert (1995) for the importance of allowing clusters to have different volumes.

Now, if we write the above decomposition into the explicit form

$$\Sigma_k = \lambda_k D_k A_k D_k^{'}$$

we exactly get the form of decompostion shown in the first release of the *Mclust* package, Fraley and Raftery (1999), which yields 6 different models in term of clusters orientation, size and shape suitable for the EM algorithm estimates. The 6 possible configurations for clusters orientation, size and shape have been extended

over the various updates of the package (to our knowledge the last one is currently version 5.3 released in May 2017); 16 different parametrizations are allowed for (respectively 2 for univariate and 14 for multivariate distributions), which are summarized in the following table (see Scrucca et al., 2016):

Table 3.1: Parameterizations of the covariance matrix Σ_k currently available in *Mclust*

Identifier	Model	Distribution	Volume	Shape	Orientation
E		univariate	equal		
V		univariate	variable		
EII	λI	Spherical	equal	equal	NA
VII	$\lambda_k I$	Spherical	variable	equal	NA
EEI	λA	Diagonal	equal	equal	coordinate axes
VEI	$\lambda_k A$	Diagonal	variable	equal	coordinate axes
EVI	λA_k	Diagonal	equal	variable	coordinate axes
VVI	$\lambda_k A_k$	Diagonal	variable	variable	coordinate axes
EEE	$\lambda DAD^{'}$	Ellipsoidal	equal	equal	equal
EVE	$\lambda DA_kD^{'}$	Ellipsoidal	equal	variable	equal
VEE	$\lambda_k DAD^{'}$	Ellipsoidal	variable	equal	equal
VVE	$\lambda_k DA_kD^{'}$	Ellipsoidal	variable	variable	equal
EEV	$\lambda D_k AD_k^{'}$	Ellipsoidal	equal	equal	variable
VEV	$\lambda_k D_k AD_k^{'}$	Ellipsoidal	variable	equal	variable
EVV	$\lambda D_k A_k D_k^{'}$	Ellipsoidal	equal	variable	variable
VVV	$\lambda_k D_k A_k D_k^{'}$	Ellipsoidal	variable	variable	variable

where the letters in the first column I, E, V stand, respectively, for Identity, Equal and Variable (in order of generality). The order in which they appear referes to λ_k, A_k and D_k, so that the label EVI, for instance, implies λA_k, that is an equal λ_k ($\lambda_k = \lambda$), a variable A_k ($A_k = A_k$) and an identity matrix for D_k ($D_k = I$, omitted because it doesn't alterate the product λA_k in any way). Note that in the second column, the models appear top to bottom in order of non decreasing complexity. Furthermore, the NA values in the last column imply that the corresponding models can not be evaluated (it is not possible, indeed, to define an orientation for the K spheres, either of equal volume or not). Finally, the blank spaces in the table have a clear meaning: in the univariate case we can not have neither different shapes nor orientation.

Now, once the model has been described in the *Mclust* framework, the question obviously remains on how to choose a specific model among the 16 available in a concrete clustering problem. To this end, until the fifth version of the *Mclust*, it was adopted the BIC, *Bayesian Information Criterion*, see Schwarz (1978), defined by

$$BIC_{\mathcal{M},K} = 2\ell_{\mathcal{M},K}\left(\mathrm{X}|\,\hat{\boldsymbol{\psi}}_{ML}\right) - \nu_K\,\log(n)$$

where $\ell_{\mathcal{M},K}\left(\mathrm{X}|\,\hat{\boldsymbol{\psi}}_{ML}\right)$ is the log-likelihood evaluated at the MLE $\hat{\boldsymbol{\psi}}_{ML}$ for a model $\mathcal{M}$ (among the 16) with K components, n is the sample size, and ν_K is the number of estimated parameters.

A higher value of $BIC_{\mathcal{M},K}$ is considered the best, so the pair $\{\mathcal{M}, K\}$ which maximises $BIC_{\mathcal{M},K}$ is selected. Note that $BIC_{\mathcal{M},K}$ encompasses a penalizing term, which increases linearly with the number of parameters ν_K and logarithmically with the sample size n. This is intended to calibrate between the fitting of a given model $\mathcal{M}$ (the more complex the higher its log-likelihood) and its complexity in terms of the number of parameters to be estimated (which depends on K).

Nevertheless, the *BIC* seems to suffer from some limitations in the clustering framework, as pointed out in its fifth version by the same *Mclust* developers Scrucca, Fop, Murphy and Raftery (2016): "However, BIC tends to select the number of mixture components needed to reasonably approximate the density, rather than the number of clusters as such", (see also Chen and Kalbfleisch (1996), where is also stressed the difference between a consistent estimation of the number of clusters and the same estimate for the number of mixture components). So, since fifth release they provide also a model selection criterion proposed by Biernacki et al. (2000), a BIC-based approximation of the ICL (*integrated complete-data likelihood*), which is defined as follows:

$$ICL_{\mathcal{M},K} \simeq BIC_{\mathcal{M},K} + 2\sum_{i=1}^{n}\sum_{k=1}^{K} c_{ik}\log(\tau_{ik})$$

where τ_{ik}, as above, is the posterior probability that the i-th observation comes from the component k, while $c_{ik} = 1$ if that observation is assigned to cluster k and 0 otherwise. Compared to the *BIC*, the *ICL* adds a term which is meant to penalize entropy in the clusters, that is groups with many low values in the τ_{ik} estimates. In fact, such a cluster would be typically represented by an high amount of observations with a low conditional probability to belong to it (and probably with similar values to belong to other clusters, since $\sum_{k=1}^{K}\tau_{ik} = 1$). This is the context known in literature as overlapping clusters, where the expression "overlapping" refers to

a situation in which observations could belong to many clusters with similar posterior probabilities. Note that this term correctly penalize the *BIC*, because all the τ_{ik} are bounded between 0 and 1, so that their logarithm takes only non positive values. As explained by Scrucca, Fop, Murphy and Raftery (2016): "*ICL* penalises the *BIC* through an entropy term which measures clusters overlap. Provided that clusters overlapping is not too strong, *ICL* has shown good performance in selecting the number of clusters, with preference for solutions with well-separated groups". Anyway, despite of the above considerations the *BIC* remains the default choice in model selection within the *Mclust* package. Finally, on the same issue it should be mentioned the paper by Bertoletti, Friel and Rastelli (2015), where it is derived an exact expression for *ICL* suitable for practical implementation in different frameworks.

Chapter 4

Symmetry based clustering algorithms

In this Chapter we present some contributes to symmetry-based partitional clustering, mostly developed in the context of computer science and engineering. This because none of these methods is involved in specific statistical hypothesis (e.g. assumptions on model structure), but rather they all aim at identifying symmetric shaped clusters. From a statistical point of view, they do not represent parametric approaches.

The following short literature review is of major importance to our purpose, because also our proposal is based on symmetry, although with some relevant differences that we'll be point out in section 5. Furthermore, the next discussion may be somewhat instructive, since symmetry-based methods have been developed in clustering framework by solving from time to time some drawbacks connected to the use of symmetry, which we were able to consider as warnings in the formulation of our proposal. Given their importance, these issues will be considered in a separate section.

However, the papers we're going to discuss have been published between 2001 and 2017, thus reflecting the recent interest in symmetry based methods for clustering.

4.1 Cluster symmetry and *K-means*

The first contribution in the field (*A Modified Version of the K-Means Algorithm with a Distance Based on Cluster Symmetry*) has been proposed by Mu-Chun Su and Chieng-Hsing Chou (2001). The Authors propose a modified version of the well known *K-means* algorithm to cluster data, adopting a nonmetric distance measure based on the idea of "point symmetry" (see below for details).

Such a method aims at overcoming critical aspects in applying conventional distance measures (e.g. the Minkowski one) to clustering problems. In fact, the Euclidean distance tends to identify hyperspherical-shaped clusters, and in general a specific choice for a specific Minkowski metric will induce a specific shape in clusters we are looking for. So, symmetry is a possible alternative choice, playing the role of a more flexible measure and allowing us to find different shapes in the same dataset. In this sense, the problem of clustering data through a distance criterion turns into the problem of identifying some kind of symmetry in the structures of clusters.

Based on this idea, Su and Chou (2001) assign units to a cluster center if they present a symmetrical structure with respect to the center. The immediate problem is how to find a measure for symmetry; furthermore from a computational point of view, the question is also how to define an algorithm that may efficiently impose a given symmetry with a minimum displacement. Since in the *K-means* algorithm, the cluster centroids represent the most important information, "point symmetry" to be applied in the *K-means* algorithm is not only the symmetry about a point, but in this case, the cluster center.

For such a purpose, the Authors propose a nonmetric distance based on the concept of point symmetry: let us consider n statistical units with feature $\boldsymbol{x}_i$, $i = 1, ..., n$, and a reference vector $\boldsymbol{c}$ (e.g., a cluster centroid); the "point symmetry distance" between an individual $\boldsymbol{x}_j$ and the reference vector $\boldsymbol{c}$ is defined as

$$d_s(\boldsymbol{x}_j, \boldsymbol{c}) = \min_{i=1,...,n, i \neq j} \frac{\| (\boldsymbol{x}_j - \boldsymbol{c}) + (\boldsymbol{x}_i - \boldsymbol{c}) \|}{\| \boldsymbol{x}_j - \boldsymbol{c} \| + \| \boldsymbol{x}_i - \boldsymbol{c} \|}$$

where the denominator is used to normalize the point symmetry distance so as to make the point symmetry distance insensitive to the Euclidean distances $\| \boldsymbol{x}_j - \boldsymbol{c} \|$ and $\| \boldsymbol{x}_i - \boldsymbol{c} \|$. If the right hand term of $d_s(\boldsymbol{x}_j, \boldsymbol{c})$ is minimized when $i = j^*$(and therefore $\boldsymbol{x}_i = \boldsymbol{x}_{j^*}$), the pattern $\boldsymbol{x}_{j^*}$ is denoted as the symmetrical pattern relative to $\boldsymbol{x}_j$ with respect to $\boldsymbol{c}$. Note that the theoretical minimum value for $d_s(\boldsymbol{x}_j, \boldsymbol{c})$ is 0, and it occurs when the pattern $\boldsymbol{x}_i = (2\boldsymbol{c} - \boldsymbol{x}_j)$ exists in the observed sample, since the numerator will be identically null. Note that the above distance is a nonmetric one, see Su and Chou (2001).

The idea of the point symmetry is very simple and intuitive. It is worth to observe the geometrical interpretation of the point symmetry distance. Fig. 1 gives the concept. Let us suppose that we have four points $\boldsymbol{x}_1 = (2, 0)'$, $\boldsymbol{x}_2 = (-2, 0)'$, $\boldsymbol{x}_3 = (0, 1)'$, $\boldsymbol{x}_4 = (1, -2)'$ and one reference vector $\boldsymbol{c} = (0, 0)'$. According to $d_s(\boldsymbol{x}_j, \boldsymbol{c})$, we can easily compute:

$$d_s(\boldsymbol{x}_1, \boldsymbol{c}) = \min_{i=1,...,4, i\neq 1} \frac{\| (\boldsymbol{x}_1 - \boldsymbol{c}) + (\boldsymbol{x}_i - \boldsymbol{c}) \|}{(\| \boldsymbol{x}_1 - \boldsymbol{c} \| + \| \boldsymbol{x}_i - \boldsymbol{c} \|)} = \frac{\| (\boldsymbol{x}_1 - \boldsymbol{c}) + (\boldsymbol{x}_2 - \boldsymbol{c}) \|}{(\| \boldsymbol{x}_1 - \boldsymbol{c} \| + \| \boldsymbol{x}_2 - \boldsymbol{c} \|)} = \frac{0}{2+2} = 0$$

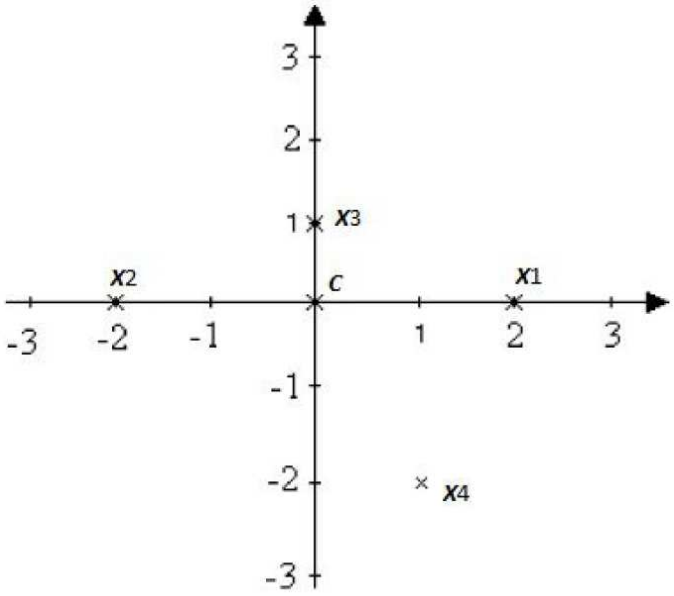
$$d_s(\boldsymbol{x}_2, \boldsymbol{c}) = \min_{i=1,...,4, i\neq 2} \frac{\| (\boldsymbol{x}_2 - \boldsymbol{c}) + (\boldsymbol{x}_i - \boldsymbol{c}) \|}{(\| \boldsymbol{x}_2 - \boldsymbol{c} \| + \| \boldsymbol{x}_i - \boldsymbol{c} \|)} = \frac{\| (\boldsymbol{x}_2 - \boldsymbol{c}) + (\boldsymbol{x}_1 - \boldsymbol{c}) \|}{(\| \boldsymbol{x}_2 - \boldsymbol{c} \| + \| \boldsymbol{x}_1 - \boldsymbol{c} \|)} = \frac{0}{2+2} = 0$$

$$d_s(\boldsymbol{x}_3, \boldsymbol{c}) = \min_{i=1,...,4, i\neq 3} \frac{\| (\boldsymbol{x}_3 - \boldsymbol{c}) + (\boldsymbol{x}_i - \boldsymbol{c}) \|}{(\| \boldsymbol{x}_3 - \boldsymbol{c} \| + \| \boldsymbol{x}_i - \boldsymbol{c} \|)} = \frac{\| (\boldsymbol{x}_3 - \boldsymbol{c}) + (\boldsymbol{x}_4 - \boldsymbol{c}) \|}{(\| \boldsymbol{x}_3 - \boldsymbol{c} \| + \| \boldsymbol{x}_4 - \boldsymbol{c} \|)} = \frac{\sqrt{2}}{1+\sqrt{5}} = 0.4$$

$$d_s(\boldsymbol{x}_4, \boldsymbol{c}) = \min_{i=1,...,4, i\neq 4} \frac{\| (\boldsymbol{x}_4 - \boldsymbol{c}) + (\boldsymbol{x}_i - \boldsymbol{c}) \|}{(\| \boldsymbol{x}_4 - \boldsymbol{c} \| + \| \boldsymbol{x}_i - \boldsymbol{c} \|)} = \frac{\| (\boldsymbol{x}_4 - \boldsymbol{c}) + (\boldsymbol{x}_3 - \boldsymbol{c}) \|}{(\| \boldsymbol{x}_4 - \boldsymbol{c} \| + \| \boldsymbol{x}_3 - \boldsymbol{c} \|)} = \frac{\sqrt{2}}{\sqrt{5}+1} = 0.4$$

Clearly, points $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are the most symmetrical pair when we consider the reference vector $\boldsymbol{c}$ in Fig. 4.1.

Fig. 4.1: An example of point symmetry distance



The algorithm developed by Su and Chou (2001) is called *SBKM* (*Symmetry Based K-Means*); its structure may be summarized as follows:

Step 1: Initialization.

Randomly choose K data points from the data set to initialize the K cluster centroids, let us denote them by $\boldsymbol{c}_1, \boldsymbol{c}_2, ..., \boldsymbol{c}_K$.

Step 2: Coarse-Tuning.

Use the ordinary *K-means* algorithm to update the K cluster centroids. After the K cluster centroids converge or some kind of stopping criterion is satisfied, we proceed to the following fine-tuning procedure.

Step 3: Fine-Tuning.

Consider a generic point $\boldsymbol{x}$, and find the nearest cluster centroid in the symmetrical sense. That is, we find the cluster centroid $\boldsymbol{c}_{k^*}$ which is nearest to $\boldsymbol{x}$ using the minimum-value criterion:

$$k^* = \arg\min_k d_s\left(\boldsymbol{x}, \boldsymbol{c}_k\right)$$

If the point symmetry distance $d_s\left(\boldsymbol{x}, \boldsymbol{c}_k\right)$ is smaller than a prespecified threshold θ, then we proceed to assign the point $\boldsymbol{x}$ to k^*th cluster. If not, the point is assigned to cluster centroid k^* using the following standard criterion:

$$k^* = \arg\min_k d_e\left(\boldsymbol{x}, \boldsymbol{c}_k\right)$$

where $d_e\left(\boldsymbol{x}, \boldsymbol{c}_k\right)$ is the Euclidean distance between $\boldsymbol{x}$ and the cluster centroid $\boldsymbol{c}_k$.

Step 4: Updating.

Compute the new centroids of the K clusters. The updating rule is given below:

$$\boldsymbol{c}_k^{(t+1)} = \frac{1}{n_k} \sum_{i \in S_k^{(t)}} \boldsymbol{x}_i$$

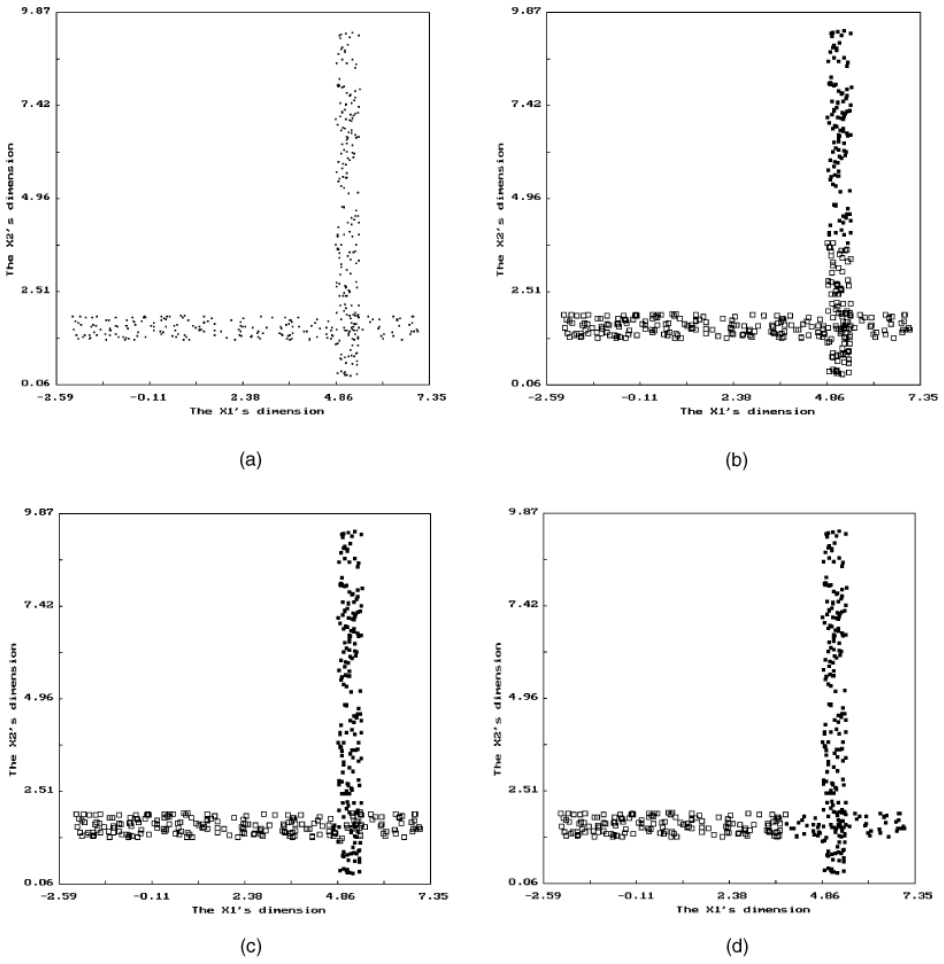
where $S_k^{(t)}$ is the set of points assigned to the kth cluster at iteration t with cardinality n_k.

Step 5: Continuation.

If no point changes cluster membership between two successive iterations or the number of iterations has reached a prespecified maximum, then stop. Otherwise, go to Step 3.

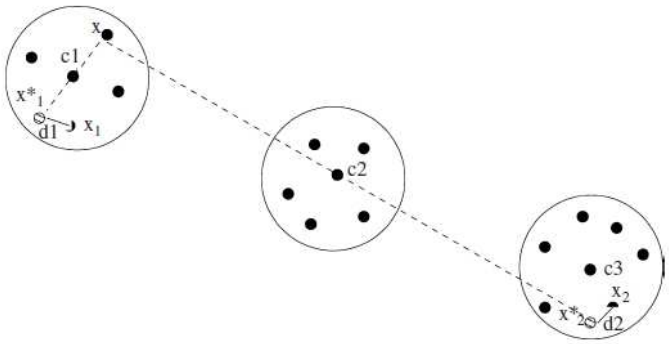
Su and Chou (2001) discuss encouraging experimental results of the *SBKM* when compared to *K-means* and their previous alghorithm *SBCL* (*Symmetry Based Competitive Learning*) proposed in Su and Chou (1999), either in clustering problems with differently shaped objects, or in the specific context of human face detection. Note that the Authors fix the parameter θ at the value of 0.18 for all the experiments. A sketch of the experimental comparison results is reported in the following figure, in a case of a two crossed objects data set:

Fig.4.2: (a) The data set contains a combination of two crossed lines. (b) The clustering result achieved by the *K-means* (c) by the *SBKM* algorithm. (d) and by the *SBCL* algorithm.



Nevertheless, there is a critical point in the *SBKM* algorithm. In particular, the underlying measure of point symmetry distance does not work well for situations where also the clusters are symmetric with respect to some intermediate point (see Fig. 4.3).

Fig. 4.3: An empirical case where point symmetry distance proposed by Su and Chou (2001) may fail.



In similar contexts, *SBKM* algorithm will produce unreasonable results, due to the symmetry distance function $d_s(\boldsymbol{x}_j, \boldsymbol{c}) = \min_{i=1,\ldots,n, i\neq j} \frac{\|(\boldsymbol{x}_j - \boldsymbol{c}) + (\boldsymbol{x}_i - \boldsymbol{c})\|}{(\|\boldsymbol{x}_j - \boldsymbol{c}\| + \|\boldsymbol{x}_i - \boldsymbol{c}\|)}$. In fact, as argued by Saha and Bandyopadhyay (2007) "minimization of $d_s(\boldsymbol{x}_j, \boldsymbol{c})$ means minimization of its numerator and maximization of its denominator. In effect, if a point $\boldsymbol{x}_j$ is almost equally symmetrical with respect to two centroids $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$, it will be assigned to that cluster that is the farthest. This is intuitively unappealing". Thus, in the example depicted in Fig. 4.3, *SBKM* will assign point $\boldsymbol{x}$ to $\boldsymbol{c}_2$, the middle cluster, thus invalidating our visual understanding.

In a subsequent paper, Su and Chou (2002) explicitly discuss this drawback and try to solve it by proposing a modification of the measure itself. But, according to Saha and Bandyopadhyay (2007) the modified measure has the same limitations of the previous one.

4.2 A Symmetry based clustering and MOO

Saha and Bandyopadhyay (2010) pose the problem of automatic clustering a data set in terms of solving a multiobjective optimization (MOO) problem. That is, their procedure aims at optimizing a set of cluster validity indices simultaneously.

Specifically, two cluster validity indices, the first based on the Euclidean distance, referred to as the *XB-index*, and the second based on the recently developed point symmetry distance, referred to as the *Sym-index*, are optimized simultaneously in order to determine the appropriate number of clusters in the observe data set. Saha and Bandyopadhyay (2010) also develop a novel point symmetry based distance, and use it as the allocation criterion to assign points to different clusters.

So, unlike the *SBKM* algorithm by Su and Chou (2001), this clustering technique is able to detect the proper number of clusters and the appropriate partitioning when the analyzed data sets have either hyperspherical clusters or point symmetric clusters. We will not focus on the number of clusters' issue, since the proposed clustering method does not involve this issue. Thus, the discussion will only point on the allocation problem, and the relative point symmetry distance.

Saha and Bandyopadhyay (2010) define their point symmetry distance $d_{ps}(\boldsymbol{x}, \boldsymbol{c})$, in the following way. Let us consider a point $\boldsymbol{x}$. The symmetrical (reflected) point of $\boldsymbol{x}$ with respect to a particular center $\boldsymbol{c}$ is $\boldsymbol{x}^* = 2 \times \boldsymbol{c} - \boldsymbol{x}$. Let $knear$ unique nearest neighbors of $\boldsymbol{x}^*$ be at Euclidean distances of d_i, $i = 1, 2, ..., knear$ from $\boldsymbol{x}^*$, that is we take the *knear* points nearest to the reflected point $\boldsymbol{x}^*$. Let us define the set of *knear* points by $N(\boldsymbol{x}^*)$, with cardinality equal to $knear = |N(\boldsymbol{x}^*)|$.

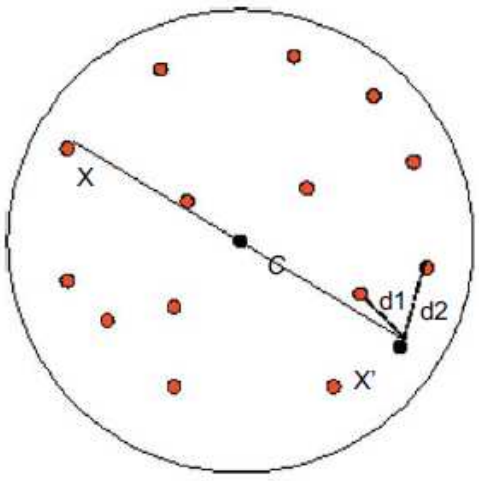
Thus, the point symmetry distance $d_{ps}(\boldsymbol{x}, \boldsymbol{c})$ can be expressed in the following way:

$$d_{ps}(\boldsymbol{x}, \boldsymbol{c}) = d_{sym}(\boldsymbol{x}, \boldsymbol{c}) \times d_e(\boldsymbol{x}, \boldsymbol{c}) = \frac{\sum_{i=1}^{knear} d_i}{|N(\boldsymbol{x}^*)|} \times d_e(\boldsymbol{x}, \boldsymbol{c})$$

with $\boldsymbol{x}$ being at the Euclidean distance $d_e(\boldsymbol{x}, \boldsymbol{c})$ from the center $\boldsymbol{c}$, and $d_{sym}(\boldsymbol{x}, \boldsymbol{c})$ is a symmetry measure of $\boldsymbol{x}$ with respect to $\boldsymbol{c}$.

To gain a visual concept of this measure, let us consider Fig. 4.4.

Fig. 4.4: Example of the point symmetry distance.



Here, we have a point $\boldsymbol{x}$ with symmetrical counterpart with respect to cluster center $\boldsymbol{c}$ denoted by $\boldsymbol{x}^* = 2 \times \boldsymbol{c} - \boldsymbol{x}$. Suppose we fix $knear = 2$ (see below for such a

choice) and the two nearest neighbors of $\boldsymbol{x}^*$ are at Euclidean distances d_1 and d_2, respectively. Thus, the point symmetry distance between $\boldsymbol{x}$ and $\boldsymbol{c}$ is $d_{ps}(\boldsymbol{x}, \boldsymbol{c}) = [(d_1 + d_2)/2] \times d_e(\boldsymbol{x}, \boldsymbol{c})$, where $d_e(\boldsymbol{x}, \boldsymbol{c})$ is the Euclidean distance between the point X and the cluster center $\boldsymbol{c}$.

As Saha and Bandyopadhyay note (2010), in the last equality of $d_{ps}(\boldsymbol{x}, \boldsymbol{c})$ expression, the set $N(\boldsymbol{x}^*)$ cannot be chosen to include only 1 point since if $\boldsymbol{x}^*$ exists in the data set then $d_{ps}(\boldsymbol{x}, \boldsymbol{c}) = 0$ and therefore the Euclidean distance will have no impact. On the other hand, large neighbour sets, that is large values for $|N(\boldsymbol{x}^*)|$ may not be suitable because they could lead to underestimate the amount of symmetry of a point with respect to a particular cluster center. Here $N(\boldsymbol{x}^*)$ is chosen to include 2 points. It may be noted that the proper choice for $N(\boldsymbol{x}^*)$ largely depends on the distribution in the data set. A fixed value for the corresponding $|N(\boldsymbol{x}^*)|$ may have many drawbacks. For instance, for very large clusters (with many points), two neighbors may not be enough as it is very likely that a few neighbors would have a distance close to zero. On the other hand, clusters with too few points are more likely to be scattered, and the distance of the two neighbors may be too large. Thus a proper choice of the cardinality $|N(\boldsymbol{x}^*)|$ is an important issue that needs to be addressed in the future.

Based on this point symmetry distance, the Author propose the following criterion to assign points to different clusters: for each point $\boldsymbol{x}_j$, $j = 1, 2, ..., n$, find the cluster center nearest to $\boldsymbol{x}_j$ in the symmetrical sense. That is, we find the cluster center k^* that is nearest to $\boldsymbol{x}_j$ using the minimum-value criterion:

$$k^* = \arg\min_k d_{ps}(\boldsymbol{x}_j, \boldsymbol{c}_k)$$

where $\mathbf{c}_k$ denotes the center of the kth cluster and $d_{ps}(\boldsymbol{x}_j, \boldsymbol{c}_k)$ is the point symmetry based distance between a particular point $\boldsymbol{x}_j$ and the cluster center $\mathbf{c}_k$. If the corresponding ratio $d_{ps}(\boldsymbol{x}_j, \boldsymbol{c}_k)/d_e(\boldsymbol{x}_j, \boldsymbol{c}_k)$ is smaller than a pre-specified threshold θ, we assign the point $\boldsymbol{x}_j$ to the kth cluster. Here $d_e(\boldsymbol{x}_j, \boldsymbol{c}_k)$ is the Euclidean distance between the point $\boldsymbol{x}_j$ and the cluster center $\boldsymbol{c}_k$. But if $d_{ps}(\boldsymbol{x}_j, \boldsymbol{c}_k)/d_e(\boldsymbol{x}_j, \boldsymbol{c}_k) > \theta$, assignment is done based on the minimum Euclidean distance criterion as in the standard *K-means* algorithm, i.e. assign $\boldsymbol{x}_j$ to k^* th cluster where $k^* = \arg\min_k d_e(\boldsymbol{x}_j, \boldsymbol{c}_k)$.

The reason for doing such an assignment is that, in the intermediate stages of the algorithm, when the centers are not yet properly evolved, then the minimum d_{ps} value for a point is expected to be quite large, since the point might not be symmetric with respect to any center. In such cases, the use of Euclidean distance for allocating a point to a cluster appears to be intuitively more appropriate.

The value of θ is kept equal to the maximum nearest neighbor distance among

all the points in the data set. It may be noted that if a point is symmetric with respect to some cluster center, the symmetrical distance computed in the above way will be small, and can be bounded as follows. Let d_{NN}^{max} be the maximum nearest neighbor distance in the dataset, $d_{NN}^{max} = \max_{i=1,\ldots,n} d_{NN}(\boldsymbol{x}_i)$, where $d_{NN}(\boldsymbol{x}_i)$ is the nearest neighbor distance for $\boldsymbol{x}_i$. Assume that $\boldsymbol{x}^*$, the reflected point of $\boldsymbol{x}$ with respect to the cluster center $\boldsymbol{c}$, lies within the data space, and recall that d_1 and d_2 are the Euclidean distances of the two nearest neighbors of $\boldsymbol{x}^*$. Then it may be noted that $d_1 \leq (d_{NN}^{max}/2)$ and $d_2 \leq (3 \times d_{NN}^{max}/2)$ resulting in $[(d_1 + d_2)/2] \leq d_{NN}^{max}$. Ideally, a point $\boldsymbol{x}$ is exactly symmetrical with respect to some $\boldsymbol{c}$ if $d_1 = 0$. However, considering the uncertainty in the location of a point as the sphere of radius d_{NN}^{max} around $\boldsymbol{x}$, Saha and Bandyopadhyay (2010) have kept the threshold θ equal to d_{NN}^{max}. Thus the computation of θ is automatic and does not require intervention from the user side.

Once the allocation has been done, the cluster centers are replaced by the mean points of the respective clusters. This is referred to as the *K-means* like update center operation.

The previous criterion can be expressed in the form of the following algorithm:

Step 1: Computation of all point symmetry distances.
For all data points $\boldsymbol{x}_i$, $1 \leqslant i \leqslant n$, compute

$$k^* = \arg\min_k d_{ps}(\boldsymbol{x}_i, \boldsymbol{c}_k)$$

Step 2: Assignment
if

$$d_{ps}(\boldsymbol{x}_i, \boldsymbol{c}_{k^*}) / d_e(\boldsymbol{x}_i, \boldsymbol{c}_{k^*}) < \theta$$

$\boldsymbol{x}_i$ is assigned to the k^*th cluster.
Otherwise, the data point is assigned to the k^*th cluster, where

$$k^* = \arg\min_k d_e(\boldsymbol{x}_i, \boldsymbol{c}_k)$$

Step 3: Centers update
Compute the new centroids of the K clusters as follows:

$$\boldsymbol{c}_k^{(t+1)} = \frac{1}{n_k} \sum_{i \in S_k^{(t)}} \boldsymbol{x}_i$$

where $S_k^{(t)}$ is the set of elements assigned to the kth cluster at time t and $n_k = |S_k|$.
Step 4: Continuation.

If no point changes cluster membership between two successive iterations or the number of iterations has reached a prespecified maximum, then stop. Otherwise, go to Step 1.

Compared to the method of Su and Chou (2001), this procedure shows a notable feature, namely it tries to avoid wrong allocations when clusters themselves are symmetrical with respect to some intermediate point (see above Fig. 2). According to Saha and Bandyopadhyay (2007), this is due to considering the second nearest neighbor. In fact, at least in the case where data can be supposed to be realizations of continuous random variables, the term $(d_1 + d_2)/2$ will never be equal to 0, and the effect of $d_e(\boldsymbol{x}_i, \boldsymbol{c}_k)$, the Euclidean distance, will always have an impact in the value of $\frac{\sum_{i=1}^{knear} d_i}{|N(\boldsymbol{x}^*)|} \times d_e(\boldsymbol{x}, \boldsymbol{c})$, thus reducing the problems discussed in Fig. 2.

In a previous work Saha and Bandyopadhyay (2007) show good performance for this method, when compare to the *K-means*, *SBKM* algorithm and its modified version, in sixteen scenarios. In this case, Saha and Bandyopadhyay (2010) propose a further comparison of their technique VAMOSA, with other multiobjective clustering technique, *MOCK* (see Handl and Knowles (2007)), two automatic clustering techniques based on single objective genetic algorithms, *VGAPS* (see Saha and Bandyopadhyay (2008)) and *GCUK* (see Bandyopadhyay and Maulik (2002)). The study of performance is shown for seven artificial data sets and six real-life data sets with varying complexities.

One of the external index chosen in Saha and Bandyopadhyay (2010) to evaluate clustering performance is the *Minkowski score*. This is a measure of the quality of a solution given the true clustering, see Ben-Hur and Guyon (2003). To put it informally, let T be the "true" solution and S our current solution to be evaluated. Denote by n_{11} the number of pairs of elements that are in the same cluster in both S and T. Denote by n_{01} the number of pairs of elements that are in the same cluster only in S, and by n_{10} the number of pairs of elements that are in the same cluster in T. Therefore, *Minkowski score* can be defined as follows:

$$MS(T, S) = \sqrt{\frac{n_{01} + n_{10}}{n_{11} + n_{10}}}$$

Note that it's a misclassification index, measuring the discordance between the true and another clustering solution in the numerator, such that a lower value represents a better solution (0 is the optimum value).

On the basis of such an index, Saha and Bandyopadhyay (2010) show the following table of comparison results

Table 4.1: Rankings (in brackets) for algorithms VAMOSA, MOCK, VGAPS and GCUK over 13 datasets, based on the *MS* value obtained

DataSet	VAMOSA	MOCK	VGAPS	GCUK
AD_5_2	0.25(1)	0.39(2)	0.25(1)	0.39(2)
AD_10_2	0.43(2)	1.01(4)	0.84(3)	0.09(1)
Mixed_5_2	0.00(1)	0.00(1)	0.00(1)	0.75(2)
Sym_3_2	0.12(1)	0.69(2)	0.12(1)	0.74(3)
Square1	0.19(1)	0.19(1)	0.20(2)	0.21(3)
Square4	0.51(1)	0.60(3)	0.52(2)	0.51(1)
Sizes5	0.14(1)	0.64(4)	0.22(2)	0.25(3)
Iris	0.80(2)	0.82(3)	0.62(1)	0.84(4)
Cancer	0.32(1)	0.39(4)	0.37(2)	0.38(3)
Newthyroid	0.57(1)	0.82(4)	0.58(2)	0.65(3)
Lungcancer	0.85(1)	0.97(3)	0.97(3)	0.94(2)
Wine	0.97(3)	0.90(1)	0.97(3)	0.93(2)
LiverDisorder	0.98(1)	0.98(1)	0.98(1)	0.99(2)
Average rank	1.31	2.538	1.846	2.384

4.3 A line symmetry based approach

Singh Vijendra and Sahoo Laxman (2015) present another multiobjective genetic clustering approach, where data points are assigned to clusters based on a *line* symmetry distance, rather than a point one. The Authors call their algorithm "multiobjective line symmetry based genetic clustering" (*MOLGC*). Similarly to the method proposed by Saha and Bandyopadhyay (2010) they exploit two objective functions, the *Davies-Bouldin index* (DB) and the line symmetry distance, while Saha and Bandyopadhyay (2010) used the *XB index* and the point symmetry distance.

Vijendra and Laxman (2015) stress that their "algorithm evolves near-optimal clustering solutions using multiple clustering criteria, without a priori knowledge of the actual number of clusters". So, even in the case of *MOLGC*, the output of the algorithm gives us an estimate of the number of clusters as well as an optimal (in some sense) allocation of points. Vijendra and Laxman (2015) begin their paper with a wide discussion on critical points of previous symmetrical approaches to clustering. We will discuss their arguments as they have a major interest for our

purpose, since they show using several examples the essential drawbacks which symmetry based methods may present.

4.3.1 Drawbacks of previous approaches

The first algorithm discussed is *SBKM* (Su and Chou, 2001). In this case, Vijendra and Laxman (2015) point out two critical aspects in the point symmetry distance (*PSD*)

(1) lacking the distance difference symmetry property.

(2) it may lead to an unsatisfactory clustering result for the case of symmetrical inter-clusters.

With regards to the first issue, the *PSD* measure favors the far data point when we have more than two symmetrical data points and this may degrade the symmetrical robustness.

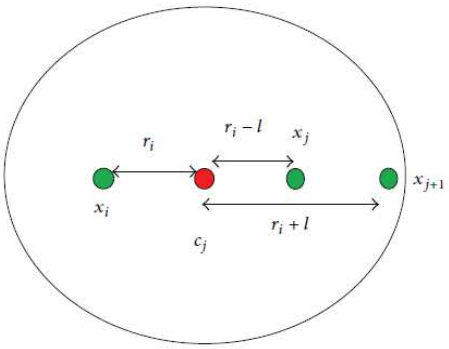
This kind of problem may be decribed using the following simple example: let $\boldsymbol{x}_j = (-5;\, 5)$, $\boldsymbol{x}_i = (7;\, -1)$, $\boldsymbol{x}_{i+1} = (10;\, -9)$, $\boldsymbol{c} = (0;\, 0)$; then find the most symmetry point of $\boldsymbol{x}_j$ relative to $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i+1}$,

$$d_s\left(\boldsymbol{x}_j, \boldsymbol{c}\right) = \min\left\{\frac{\parallel \left(\boldsymbol{x}_j - \boldsymbol{c}\right) + \left(\boldsymbol{x}_i - \boldsymbol{c}\right) \parallel}{(\parallel \boldsymbol{x}_j - \boldsymbol{c} \parallel + \parallel \boldsymbol{x}_i - \boldsymbol{c} \parallel)}, \frac{\parallel \left(\boldsymbol{x}_j - \boldsymbol{c}\right) + \left(\boldsymbol{x}_{i+1} - \boldsymbol{c}\right) \parallel}{(\parallel \boldsymbol{x}_j - \boldsymbol{c} \parallel + \parallel \boldsymbol{x}_{i+1} - \boldsymbol{c} \parallel)}\right\}$$

$$d_s\left(\boldsymbol{x}_j, \boldsymbol{c}\right) = \min\left\{\frac{\sqrt{20}}{\left(\sqrt{50} + \sqrt{58}\right)}, \frac{\sqrt{41}}{\left(\sqrt{50} + \sqrt{181}\right)}\right\} = \min\left\{0.32,\, 0.31\right\}$$

The data point $\boldsymbol{x}_{i+1}$ is selected as the most symmetrical point of $\boldsymbol{x}_j$ relative to the centroid $\boldsymbol{c}$, although the most symmetrical point with respect to $\boldsymbol{x}_j$ is clearly $\boldsymbol{x}_i$ (note that only numerator measures the symmetry degree, and we have for this example $\parallel \left(\boldsymbol{x}_j - \boldsymbol{c}\right) + \left(\boldsymbol{x}_i - \boldsymbol{c}\right) \parallel = \sqrt{20} < \parallel \left(\boldsymbol{x}_j - \boldsymbol{c}\right) + \left(\boldsymbol{x}_{i+1} - \boldsymbol{c}\right) \parallel = \sqrt{41}$, indicating that $\boldsymbol{x}_i$ is the most symmetrical point). It happens as a result of the denominator impact on $d_s\left(\cdot, \cdot\right)$ value, which is intended to take into account for euclidean distances. This shows that $d_s\left(\boldsymbol{x}_j, \boldsymbol{c}\right)$ favors the far data point when we have more than two data points and this may corrupt the symmetrical robustness (see the following Fig 4.5 for a simplified representation).

Fig. 4.5: An example for the distance difference symmetry.



The second drawback is the same pointed out by Saha and Bandyopadhyay (2007). It occurs when two clusters are symmetrical to each other with respect to the centroid of any third cluster, resulting in a strong biased clustering output (see Fig 2). Vijendra and Laxman (2015) refer to it as a "point symmetry interclusters distance". Hence, the Authors mention two possible alternative approaches to solve the issues involved in the *SBKM* algorithm: the proposals by Chung and Lin (2007) and Saha and Bandyopadhyay (2008).

4.3.2 The line symmetry based distance

Vijendra and Laxman (2015) introduce the line symmetry based distance taking the steps from a previous paper by Saha and Maulik (2011), who developed a line symmetry based automatic genetic clustering technique called "variable string length genetic line symmetry distance based clustering" (*VGALS-Clustering*). To measure the amount of line symmetry for a point x with respect to a particular line r_k, $d_{ls}\left(\boldsymbol{x}, r_k\right)$, the following steps are required:

1. For a particular data point $\boldsymbol{x}$, calculate the projected point $p_k\left(\boldsymbol{x}\right)$ on the relevant symmetrical line r_k.

2. Find $d_{sym}(\boldsymbol{x}, p_k)$ as

$$d_{sym}(\boldsymbol{x}, p_k) = \sum_{j \in N(\boldsymbol{x})} \frac{d_j}{|N\left(\boldsymbol{x}\right)|}$$

where $|N\left(\boldsymbol{x}\right)|$ nearest neighbors of $\boldsymbol{x}^* = 2 \times p_k\left(\boldsymbol{x}\right) - \boldsymbol{x}$ are at Euclidean distances of d_j, $j = 1, ..., |N\left(\boldsymbol{x}\right)|$ from point $\boldsymbol{x}^*$.

The amount of line symmetry for a particular point x with respect to that particular symmetrical line r_k of cluster k is calculated as

$$d_{ls}\left(\boldsymbol{x}, r_k\right) = d_{sym}(\boldsymbol{x},\, p_k) \times d_e(\boldsymbol{x},\, \mathbf{c}_k)$$

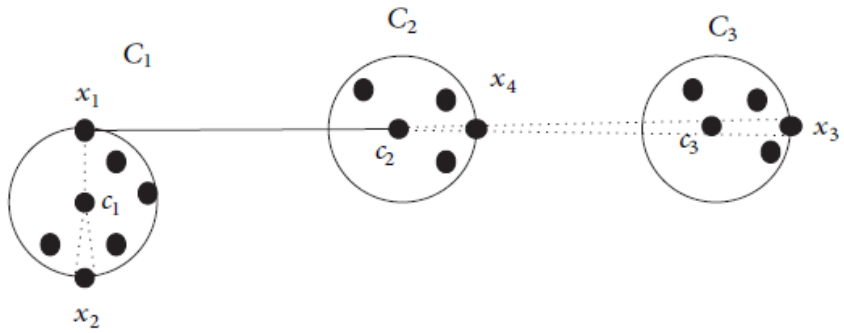
where $\mathbf{c}_k$ is the centroid of the specific cluster k and $d_e(\boldsymbol{x}, \mathbf{c}_k)$ is the Euclidean distance between points $\boldsymbol{x}$ and $\mathbf{c}_k$.

Roughly speaking, line symmetry based distance is somewhat different from the point one, in that symmetry is calculated with respect to a point projected on a line which passes trough the centroid, rather than with respect to the centroid itself. Vijendra and Laxman (2015) note however that all methods discussed, violate the closure property that paraphrasing the Authors may be defined as follows: if the data point x is currently assigned to the cluster k with centroid $\mathbf{c}_k$ in the current iteration, the determined most symmetrical point $p_k\left(\boldsymbol{x}\right)$ (relative to x, $\mathbf{c}_k$ and the line r_k) must have been assigned to the same cluster k in the previous iteration.

Intuitively this property avoid that one point would be assigned to a cluster on the basis of another point allocated in a different cluster (note that a previous definition comes from Chung and Lin, 2007).

The example discussed by Vijendra and Laxman (2015) can be better explained as follows: let us consider the data point $\boldsymbol{x}_1$ in Figure 4.6,

Fig. 4.6: Violation of the closure property.



which is originally not in cluster C_2, when the symmetry distance $d_{sym}(\boldsymbol{x}_1, \boldsymbol{c}_2)$ with respect to the cluster center $\boldsymbol{c}_2$ is the most symmetrical distance

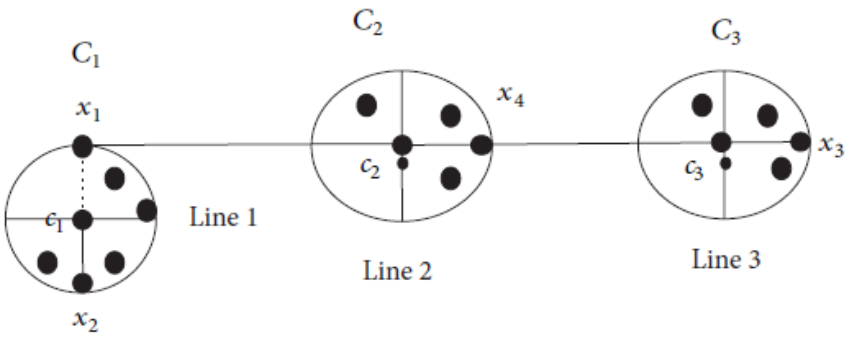
$$d_{sym}(\boldsymbol{x}_1, \boldsymbol{c}_2) < d_{sym}(\boldsymbol{x}_1, \boldsymbol{c}_1) < d_{sym}(\boldsymbol{x}_1, \boldsymbol{c}_3)$$

among all other symmetry distances, the point $\boldsymbol{x}_1$ should be assigned to cluster C_2.

But, as we can see by looking at figure 4.6, the most symmetrical point of $\boldsymbol{x}_1$ with respect to the centroid $\boldsymbol{c}_2$ is the data point $\boldsymbol{x}_3$, which is assigned to a different cluster (the third with centroid $\boldsymbol{c}_3$). Since the data point $\boldsymbol{x}_3$ has not been assigned to the same cluster of $\boldsymbol{x}_1$, the closure property is violated and an unsatisfactory clustering result is given.

To solve the problem, Vijendra and Laxman (2015) propose a new line symmetry based distance, with a constraint on the points to be considered. In particular, the Authors remove the candidate symmetrical points $\boldsymbol{x}_j \notin C_k$ for each symmetrical line k of cluster C_k. For the data point $\boldsymbol{x}_i$ and symmetrical line of cluster C_k, this restriction helps to search more suitable symmetrical point $\boldsymbol{x}_j$, since the candidate most symmetrical points $\boldsymbol{x}_j$ which are not in the cluster C_k are ignored. As depicted in the following Fig. 4.7,

Fig. 4.7: An example of line symmetry distance.



let us consider the point $\boldsymbol{x}_1$ with line symmetry distances $d_{ls}(\boldsymbol{x}_1, r_2) < d_{ls}(\boldsymbol{x}_1, r_1) < d_{ls}(\boldsymbol{x}_1, r_3)$ with respect to the line for cluster C_2 and the symmetrical point is $\boldsymbol{x}_3$; due to the above constraints, the proposed line symmetry distance method allocates the point $\boldsymbol{x}_1$ to cluster C_1. The assignment of $\boldsymbol{x}_1$ to C_1 is a reasonable assignment from our visual system. The same constraint is applied to the second symmetrical point, so that computation of line symmetry considers only points in the same cluster of the involved centroid, as depicted in Figure 4.7.

However, the main question in all methods based on line symmetry is the need for a rule to choose the orientation of the line passing trough centroids, referred to as *major axis*. The approach proposed by Vijendra and Laxman (2015) requires the computation of moments up to second order. The Authors discuss the procedure in the case of two-dimensional data. Let the data be denoted by $X = \{(x_1, y_1), (x_2, y_2), (x_n, y_n)\}$, the sample moment of order (p, q) is defined as follows

$$u_{pq} = \sum_{i=1}^{n} x_i^p y_i^q$$

The mean for the data is defined by

$$(\bar{x}, \bar{y}) = (u_{10}/u_{00},\ u_{01}/u_{00})$$

The central moment is defined by

$$m_{pq} = \sum_{i=1}^{n} (x_i - \bar{x})^p \, (y_i - \bar{y})^q$$

Clearly, the same quantities relative to the generic kth cluster would be obtained by restricting the above sums only to the points in the same cluster k, that is

$$u_{pq}^{(k)} = \sum_{i \in S_k} x_i^p y_i^q$$

$$\mathbf{c}_k = \left(\bar{x}_{(k)}, \bar{y}_{(k)}\right) = \left(u_{10}^{(k)}/u_{00}^{(k)},\ u_{01}^{(k)}/u_{00}^{(k)}\right)$$

$$m_{pq}^{(k)} = \sum_{i \in S_k} \left(x_i - \bar{x}_{(k)}\right)^p \left(y_i - \bar{y}_{(k)}\right)^q$$

According to the mean $\mathbf{c}_k = (\bar{x}, \bar{y})^{(k)}$ and $m_{pq}^{(k)}$ calculated over cluster k, the *major axis* for the same kth cluster can be determined by the following two conditions:

(a) the major axis of the cluster k must pass through the centroid $\mathbf{c}_k$.

(b) the angle between the *major axis* and the x-axis is equal to

$$0.5 \arctan\left(2m_{11}^{(k)}/(m_{20}^{(k)} - m_{02}^{(k)})\right)$$

Consequently, the corresponding *major axis* for the kth cluster, r_k, is thus expressed by

$$\left\{ \mathbf{c}_k,\ 0.5 \arctan\left(\frac{2m_{11}^{(k)}}{\left(m_{20}^{(k)} - m_{02}^{(k)}\right)} \right) \right\}$$

In other words, for a particular cluster the couple (centroid coordinates and slope) gives the equation for the line (*major axis*) we are looking for (note that it's well defined since for a fixed slope only one line could pass trough a particular point).

Once obtained the cluster *major axis*, the allocation procedure works as follows: "the *major axis* r_k is treated as the symmetric line of the relevant cluster. This line is used to measure the amount of symmetry for a particular point in that cluster. To measure the amount of line symmetry for a point $\boldsymbol{x}_i$ with respect to the line r_k of cluster C_k, $d_{ls}(\boldsymbol{x}_i, r_k)$ the following steps are required:

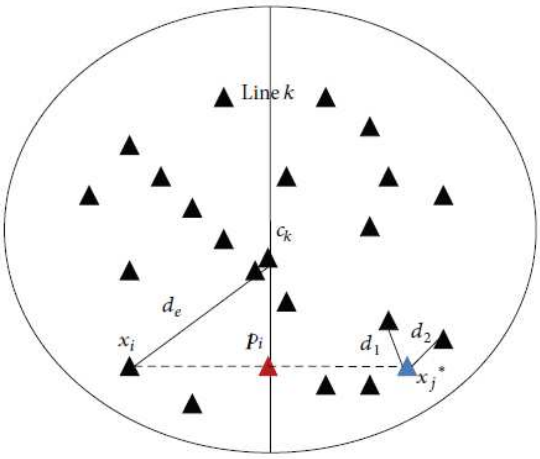
1. For the point $\boldsymbol{x}_i$, calculate the projected point $p_k(\boldsymbol{x}_i)$ on the relevant symmetrical line r_k for cluster C_k (as shown in Figure 4.8) and find out all possible symmetrical data point x_j relative to each symmetrical line k for $1 \leqslant i \leqslant n$, $1 \leqslant j \leqslant n$ and $1 \leqslant k \leqslant K$.

2. Find $d_{sym}(\boldsymbol{x}_i, p_k(\boldsymbol{x}_i))$ as

$$d_{sym}(\boldsymbol{x}_i, p_k(\boldsymbol{x}_i)) = \sum_{j \in N(\boldsymbol{x}_i)} \frac{d_j}{|N(\boldsymbol{x}_i)|}$$

where $|N(\boldsymbol{x}_i)|$ nearest neighbors of $\boldsymbol{x}_j = 2 \times p_k(\boldsymbol{x}_i) - \boldsymbol{x}_i$ are at Euclidean distances of d_j, $i = 1, ..., |N(\boldsymbol{x}_i)|$.

Fig. 4.8: An example for computing line symmetry distance.



As argued by the Authors, the role of the size of the neighboring points set $|N(\boldsymbol{x})|$ is intuitively easy to understand and it can be set by the user based on specific knowledge. As we have argued before, the choice for a fixed size irrespective

of the current aaplication may not be a good choice. For clusters with too few points, the points are likely scattered and the distance between two neighbors may be too large, while for very large clusters a fixed number of neighbors may not be enough because few neighbors would have a close to zero distance. Obviously, the parameter is related to the expected minimum cluster size and should be much smaller than the number of objects in the data; the Authors suggest to use the rule $|N(\boldsymbol{x})| \leq \sqrt{n}$.

Once obtained $d_{sym}(\cdot, \cdot)$, the amount of line symmetry for a particular point x_i with respect to a particular symmetrical line r_k is calculated as follows

$$d_{ls}(\boldsymbol{x}_i, r_k) = d_{sym}(\boldsymbol{x}_i, p_k(\boldsymbol{x}_i)) \times d_e(\boldsymbol{x}_i, \mathbf{c}_k)$$

where $\mathbf{c}_k$ is the centroid for cluster C_k and $d_e(\boldsymbol{x}_i, \mathbf{c}_k)$ denotes the Euclidean distance between data point $\boldsymbol{x}_i$ and cluster center $\mathbf{c}_k$.

The Authors present the allocation procedure using a scheme which could be described in algorithmic details as follows:

Algorithm details

Step 1: Identifying symmetrical points under closure property

Find two symmetrical points $\boldsymbol{x}_1^*$ and $\boldsymbol{x}_2^*$ of $\boldsymbol{x}_i$ relative to
projected point $p_k(\boldsymbol{x}_i)$ on the line r_k of cluster C_k/* To ensure the closure property */

Step 2: Computation of the line symmetry distance

Calculate the line symmetry-based distance $d_{ls}[C_k] = d_{ls}(\boldsymbol{x}_i, r_k)$, $k = 1, ..., K$
Find $C_{k^*} = \arg\min_k d_{ls}[C_k]$
if $d_{ls}(\boldsymbol{x}_i, r_{k^*}) \leqslant d_{ls}(\boldsymbol{x}_i, r_l)$, k^*, $l = 1, ..., K$ and $d_{ls}(\boldsymbol{x}_i, r_{k^*}) / d_e(x_i, \boldsymbol{c}_{k^*}) \leqslant \theta$
Allocate point $\boldsymbol{x}_i$ to cluster C_{k^*}
if $d_{ls}(\boldsymbol{x}_i, r_{k^*}) / d_e(x_i, \boldsymbol{c}_{k^*}) > \theta$
Allocate the point x_i to the cluster C_{k^*} based on the Euclidean distance
$C_{k^*} = \arg\min_k d_e(\boldsymbol{x}_i, \boldsymbol{c}_{k^*})$

Step 3: Centroids update

Compute new cluster centers as follows:

$$c_k^{new} = \frac{1}{n_k} \sum_{i \in C_k} \boldsymbol{x}_i$$

where n_k is the number of data points belonging to cluster C_k.

Step 4: Recursion and stopping rule

If no point changes cluster membership between two successive iterations or the number of iterations has reached a prespecified maximum, then stop. Otherwise, go to Step 1.

In the second part of the paper, Vijendra and Laxman (2015) show some experimental results based on several artificial and real datasets. They compare the proposed approach *MOLGC* with *SBKM* and *MOCK* clustering algorithms in terms of different quality measures. Here below an example of the comparison results, based on adjusted Rand index:

Table 4.2: Median values of adjusted Rand index for artificial and real data sets.

Data sets	SBKM	MOCK	MOLGC
Data set-1	0.7599	0.9880	0.9895
Data set-2	0.7510	0.9297	0.9555
Data set-3	0.5199	0.9560	0.9940
Data set-4	0.8685	0.9875	0.9850
Data set-5	0.7280	0.9885	0.9915
Data set-6	0.6625	0.9690	0.9825
Data set-7	0.6805	1.0000	1.0000
Data set-8	0.6375	0.9915	0.9980
Iris	0.7715	0.9380	0.9875
Cancer	0.7901	0.9565	0.9780
Wine	0.6610	0.9605	0.9615
Diabetes	0.7157	0.9870	0.9925

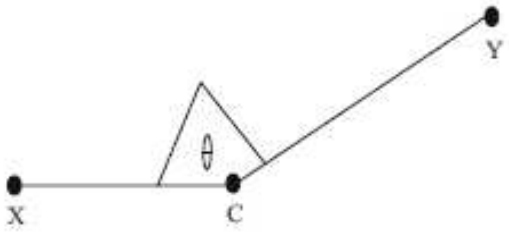
4.4 Enhancing point symmetry-based distance

Saha in February (2017) presents a new formulation of point symmetry distance, with further closure and symmetry properties.

The new point symmetry distance works as follows: let us consider two points

are $\boldsymbol{x}$ and $\boldsymbol{y}$ (as shown in Fig. 4.9) and let a cluster center be $\boldsymbol{c}$. Further, let us consider two vectors $\mathbf{a} = (\boldsymbol{x} - \boldsymbol{c})$ and $\mathbf{b} = (\boldsymbol{y} - \boldsymbol{c})$.

Fig. 4.9: Example of the new distance based on point symmetry.



Then, the symmetry-based distance between $\boldsymbol{x}$ and $\boldsymbol{y}$ with respect to cluster center $\boldsymbol{c}$ is computed as follows:

$$symlevel\,(\boldsymbol{x},\,\boldsymbol{y},\,\boldsymbol{c}) = \frac{0.5 - \frac{\mathbf{ab}}{2*|\mathbf{a}||\mathbf{b}|}}{1 + \frac{(||\mathbf{a}|-|\mathbf{b}||)}{\min(|\mathbf{a}|,|\mathbf{b}|)}}$$

Here, the numerator measures the angle between the vectors $\mathbf{a}$ and $\mathbf{b}$, which is given by $\frac{\mathbf{ab}}{|\mathbf{a}||\mathbf{b}|}$. If we note that $-1 \leq \frac{\mathbf{ab}}{|\mathbf{a}||\mathbf{b}|} \leq +1$, we obtain also that

$$-\frac{1}{2} \leq \frac{\mathbf{ab}}{2 * |\mathbf{a}|\,|\mathbf{b}|} \leq \frac{1}{2}$$

$$0 \leq 0.5 - \frac{\mathbf{ab}}{2 * |\mathbf{a}|\,|\mathbf{b}|} \leq 1$$

At the same time, the denominator measures 1 plus the relative difference between the length of the vectors $\mathbf{a}$ and $\mathbf{b}$. For the optimal case, i.e., when $\boldsymbol{y}$ is perfectly symmetrical with respect to $\boldsymbol{x}$, this difference should be equal to zero, so that the denominator is equal to 1. Note that in order to maximize the symmetry level between two points $\boldsymbol{x}$ and $\boldsymbol{y}$ with respect to the cluster center $\boldsymbol{c}$, the angular value between $\mathbf{a}$ and $\mathbf{b}$ should be near or equal to 180. This means that the numerator should be close to 1. Thus in order to maximize symmetry, the numerator has to be maximized. At the same time, in order to maximize the symmetry level between two points $\boldsymbol{x}$ and $\boldsymbol{y}$ with respect to the cluster center $\boldsymbol{c}$, the relative difference between the two vectors $\mathbf{a}$ and $\mathbf{b}$, should be minimum. In the ideal case, this should be equal to 0.

Thus in order to maximize symmetry level between the two points $\boldsymbol{x}$ and $\boldsymbol{y}$ with

respect to the cluster center $\boldsymbol{c}$, the numerator of the *symlevel* should be maximized and the denominator should be minimized. Thus larger value of *symlevel* indicates higher symmetry between these two points. To get a visual understanding of this formulation, let us consider once again Figure 4.9.

As Saha (2017) shows, *symlevel* is bounded between 0 and 1. In fact, its numerator $0 \leq 0.5 - \frac{\mathbf{ab}}{2*|\mathbf{a}||\mathbf{b}|} \leq 1$ while the denominator $1 + \frac{(||\mathbf{a}|-|\mathbf{b}||)}{\min(|\mathbf{a}|,|\mathbf{b}|)} \geq 1$. Thus, the ratio is always between 0 and 1, and reaches 0 if $0.5 - \frac{\mathbf{ab}}{2*|\mathbf{a}||\mathbf{b}|} = 0 \longleftrightarrow \frac{\mathbf{ab}}{2*|\mathbf{a}||\mathbf{b}|} = 0.5 \longleftrightarrow \frac{\mathbf{ab}}{|\mathbf{a}||\mathbf{b}|} = 1$ since denominator $1 + \frac{(||\mathbf{a}|-|\mathbf{b}||)}{\min(|\mathbf{a}|,|\mathbf{b}|)} \geq 1$, while the ratio reaches 1 if numerator and denominator are both equal to 1, that is $0.5 - \frac{\mathbf{ab}}{\mathbf{2}*|\mathbf{a}||\mathbf{b}|} = 1 \longleftrightarrow \frac{\mathbf{ab}}{\mathbf{2}*|\mathbf{a}||\mathbf{b}|} = -0.5 \longleftrightarrow \frac{\mathbf{ab}}{|\mathbf{a}||\mathbf{b}|} = -1$ and denominator $1 + \frac{(||\mathbf{a}|-|\mathbf{b}||)}{\min(|\mathbf{a}|,|\mathbf{b}|)} = 1 \longleftrightarrow \frac{(||\mathbf{a}|-|\mathbf{b}||)}{\min(|\mathbf{a}|,|\mathbf{b}|)} = 0 \longleftrightarrow |\mathbf{a}| = |\mathbf{b}|$, provided that neither a nor b is a null vector.

Moreover, *symlevel* is also a symmetrical measure. It is very easy to check that the symmetry level measure between points $\boldsymbol{x}$ and $\boldsymbol{y}$ with respect to the cluster center $\boldsymbol{c}$ is equal to the symmetry level measure between $\boldsymbol{y}$ and $\boldsymbol{x}$ with respect to the cluster center $\boldsymbol{c}$, that is *symlevel* does not depend on the order of its elements.

Nevertheless, as Saha (2017) notes, this symmetry measure does not only lack the closure property, but it is also not able to properly identify symmetrical interclusters. To overcome the first drawback, Saha (2017) introduces the following constraint, which ensures closure property:

$$symlevel^{'}\left(\boldsymbol{x},\, \mathbf{c}_k\right) = \max_{\boldsymbol{x}_j \in C_k} symlevel\left(\boldsymbol{x},\, \boldsymbol{x}_j,\, \mathbf{c}_k\right)$$

Using such an approach, only symmetrical point $\boldsymbol{x}_j$ in the same cluster will be considered. On the other hand, to make this measure able to properly identify symmetrical interclusters , Saha (2017) introduces the following pseudo-Euclidean distance $dist\left(\boldsymbol{x},\, \boldsymbol{y}^{'},\, \mathbf{c}_k\right)$ in the allocation procedure,

$$dist\left(\boldsymbol{x},\, \boldsymbol{y}^{'},\, \mathbf{c}_k\right) = \frac{d_e\left(\boldsymbol{x},\, \mathbf{c}_k\right) + d_e\left(\boldsymbol{y}^{'},\, \mathbf{c}_k\right)}{2}$$

where $\boldsymbol{x}$ is a generic point, $\boldsymbol{y}^{'}$ its symmetrical counterpart and $\mathbf{c}_k$ the centroid. As we can see below, in the algorithmic scheme the term $dist\left(\boldsymbol{x},\, \boldsymbol{y}^{'},\, \mathbf{c}_k\right)$ enables the proposed algorithm to detect properly symmetrical interclusters.

Point assignment based on *symlevel*

Step 1: Computation of *symlevel* under closure property
$mindist = \infty$

for each datapoint $\boldsymbol{x}$
for each cluster center $\mathbf{c}_k$, $k = 1, ..., K$
and $\boldsymbol{x}_j \in C_k$ /*ensuring closure property */ compute
$symlevel^{'}(\boldsymbol{x}, \mathbf{c}_k) = \max_{\boldsymbol{x}_j \in C_k} symlevel(\boldsymbol{x}, \boldsymbol{x}_j, \mathbf{c}_k)$
Step 2: Identifying symmetrical points
For $k = 1, ..., K$
$\boldsymbol{y}'_k = \arg\max_{\boldsymbol{x}_j \in C_k} symlevel(\boldsymbol{x}, \boldsymbol{x}_j, \mathbf{c}_k)$
$\boldsymbol{y}'_k$= symmetrical point of $\boldsymbol{x}$ with respect to $\mathbf{c}_k$
Step 3: Point allocation
if $\{symlevel(\boldsymbol{x}, \boldsymbol{y}'_k, \mathbf{c}_k) > 0.7538 \ \& \ dist(\boldsymbol{x}, \boldsymbol{y}'_k, \mathbf{c}_k) < mindist\}$
assign $\boldsymbol{x}$ to C_k
$mindist = dist(\boldsymbol{x}, \boldsymbol{y}'_k, \mathbf{c}_k)$
if $\boldsymbol{x}$ is still unassigned, assign it to the closest cluster center based on the Euclidean distance, i.e., assign it to cluster center $\mathbf{c}_k$ where
$k^* = \arg\min_k d_e(\boldsymbol{x}, \mathbf{c}_k)$ where d_e stands for the Euclidean distance.
Step 4: Centers update
Compute the new centroids of the K clusters as follows:

$$\boldsymbol{c}_k^{(t+1)} = \frac{1}{n_k} \sum_{i \in S_k^{(t)}} \boldsymbol{x}_i$$

where $S_k^{(t)}$ is the set of elements assigned to the kth cluster at time t and $n_k = |S_k|$.
Step 5: Continuation.
If no point changes cluster membership between two successive iterations or the number of iterations has reached a prespecified maximum, then stop. Otherwise, go to Step 1.

Here $dist\left(\boldsymbol{x}, \boldsymbol{y}^{'}, \mathbf{c}_i\right) = \frac{d_e(\boldsymbol{x}, \mathbf{c}_k) + d_e\left(\boldsymbol{y}^{'}, \mathbf{c}_k\right)}{2}$ and 0.7538 is a pre-specified threshold (see below). As we can see from the algorithm, the condition imposed on $dist\left(\boldsymbol{x}, \boldsymbol{y}^{'}, \mathbf{c}_i\right)$ enables the proposed algorithm to properly detect symmetrical interclusters.

So, Saha (2017) sets a lower bound for the symlevel at the value of 0.7538; below this threshold the allocation of point is done according to the Euclidean distance (i.e. point is assigned to the closest cluster center). As the Author points out, the specific threshold is determined on the basis of two worst cases:
1. The angle between the vectors (a and b) must not be too far from 180 degrees. If the perfect symmetrical point exists in the data set, then this value should be equal to 180. Here, the chosen threshold is 165, that is the value 0.7538 corresponds to this angle (under a further constraint, see below) where $\frac{\mathbf{ab}}{|\mathbf{a}||\mathbf{b}|} = -0.96$. Saha (2017) has also experimented with the range 180–160, but results show poorer

performance.

2. the relative difference in the absolute values must not exceed 30%, leading denominator in $symlevel \leqslant 1.3$.

Putting the threshold values in *symlevel* expression we get

$$threshold = 0.5-(-0.96/2)/1.3 = 0.7538$$

That is, the point is allocated if the angle is between 165 and 180 degrees and the relative difference in vector lenght does not exceed 30%.

Implementation results

Experimental results shown by Saha (2017) are interesting to our purpose, due to their extensive nature (14 artificial and 7 real-life data sets) and the comparison with an existing symmetry-based genetic clustering technique, *GAPS*, (see Bandyopadhyay and Saha (2007)) and with three popular and well-known clustering techniques, *K-means*, *EM* (Expectation-Maximization) and average linkage algorithm (AL). Saha (2017) chooses two indexes to evaluate clustering performances: *Minkowski Scores* (Jardine and Sibson 1971) and *F-Measure* (Bandyopadhyay and Saha 2013). The results achieved by Saha method, *GAnPS*, seem to be very encouraging (see Table below), suggesting that this field of research is actually a promising one.

Table 4.3: Rankings for *K-means*, *GAPS*, *GAnPS*, *EM* and *AL* over 21 data sets, based on the *Minkowski Score* (MS). The best MS values are marked in bold.

Data set	K-means	GAPS	GAnPS	EM	AL
Mixed_3_2	0.40(4)	0.18(3)	**0.00**(1)	0.08(2)	0.52(5)
Sym_3_2	0.91(5)	0.12(2)	**0.097**(1)	0.24(3)	0.80(4)
Sym_5_2	0.42(2)	**0.00**(1)	**0.00**(1)	0.00(1)	0.00(1)
AD_5_2	0.25(2)	0.51(5)	**0.00**(1)	0.50(4)	0.44(3)
Line_2_2	0.82(3)	**0.0**(1)	**0.0**(1)	0.37(2)	0.85(4)
Ellip_2_2	0.83(2)	**0.0**(1)	**0.0**(1)	0.87(3)	0.90(4)
Rect_3_2	0.26(2)	**0.00**(1)	**0.00**(1)	0.30(3)	0.00(1)
Sph_4_3	**0.00**(1)	**0.00**(1)	**0.00**(1)	**0.00**(1)	**0.00**(1)
Sph_6_2	**0.00**(1)	**0.00**(1)	**0.00**(1)	**0.00**(1)	**0.00**(1)
Sph_10_2	0.13(3)	0.43(5)	**0.05**(1)	0.08(2)	0.28(4)
Square1	0.25(4)	0.21(2)	**0.19**(1)	0.23(3)	0.26(5)
Square4	0.52(3)	0.51(2)	**0.50**(1)	0.51(2)	0.67(4)
Twenty	0.15(3)	0.12(2)	**0.10**(1)	0.22(4)	0.26(5)
Forty	0.20(3)	0.15(2)	**0.12**(1)	0.21(4)	0.36(5)
Iris	0.68(2)	**0.67**(1)	**0.67**(1)	0.88(4)	0.70(3)
Cancer	0.37(3)	0.36(2)	**0.34**(1)	0.42(4)	0.45(5)
Newthyroid	0.94(4)	**0.59**(1)	**0.59**(1)	0.65(2)	0.85(3)
Wine	1.40(4)	**0.91**(1)	0.92(2)	0.95(3)	0.92(2)
Glass	1.69(4)	**0.82**(1)	**0.82**(1)	1.41(2)	1.46(3)
LungCancer	1.45(5)	0.89(2)	**0.87**(1)	0.93(3)	0.96(4)
LiverDisorder	0.98(2)	**0.96**(1)	**0.96**(1)	0.99(3)	0.98(2)
Average rank	2.818	1.72	1.13	2.54	3.18

Chapter 5

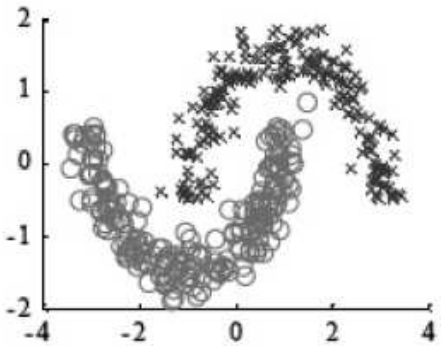
A skewness-based clustering method

In this Chapter we present a novel skewness and model based clustering method. A source of inspiration for this method could be found in the papers we have been discussing in the previous chapters. The results obtained by those approaches suggest that skewness-based techniques may represent a rapidly increasing and promising field of research. In particular, they have very good performance when compared to other, recently developed, clustering methods. Nevertheless, there are some relevant differences between skewness-based methods presented in the last chapter and the one we propose here below.

First of all, the field of application: all the clustering techniques discussed have been defined in a non parametric context, while the proposed method is introduced also in a model based clustering framework. Only the case of Gaussian densities is considered, even though, in the following we will suggest some extensions of the proposed method.

This fact leads us to a further, non-negligible, difference: despite of the generality of the prevoius approaches, here we focus on a particular cluster shape, namely the elliptical one, according to a general elliptical (cluster-specific) distribution. This means that our method could not be appropriate when the target is a different kind of symmetry, as depicted in the following figure, see Wang, Bo and Jiao (2006).

Fig. 5.1: Non-elliptical clusters



As we will see in the following, a natural choice for the objective function is a suitable skewness index, which should be able to account for elliptical (cluster-specific) distributions, with Gaussian shaped clusters as a special case. Moreover, in the final section we will discuss a possible way to encompass also non-elliptical distributions in a high-dimensional data framework.

Another relevant difference entails the competitors of the proposed method when compared to other skewness-based approaches: the methods discussed before aim at improving parameters estimates and classification performance and usually such perfomances are compared to genetic algorithms and other non-parametric competitors, such *k-means*. The proposed approach is developed to be compared also with other model based clustering techniques, such as the one implemented in the version 5 of *Mclust* package, see Scrucca et al. (2016).

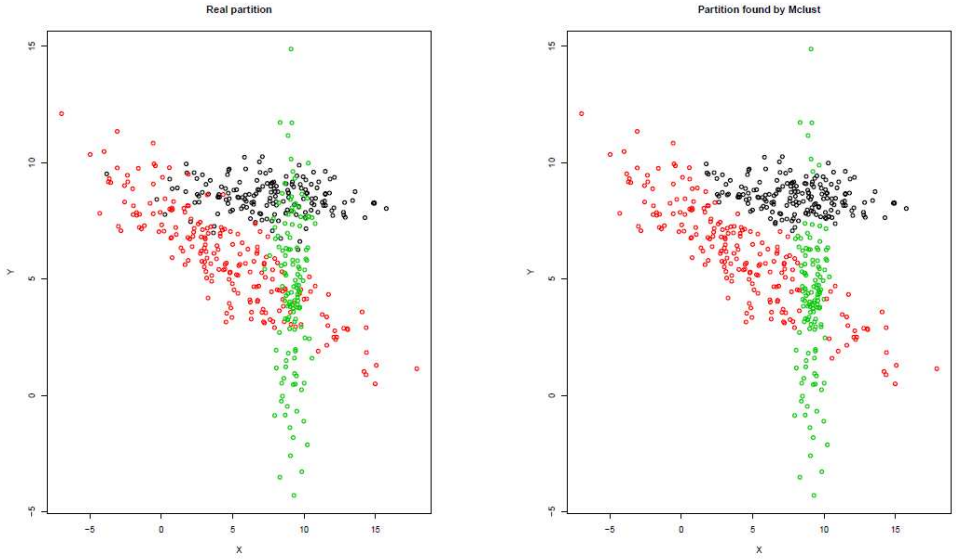
In particular, a starting point for this work was just some issues related to centroid estimates in *Mclust* when overlapping clusters are present. Therefore, to introduce the proposed approach, we start discussing this topic.

5.1. The case of overlapping clusters

The *EM* algorithm is one of the most reliable algorithms for ML estimation in Gaussian finite mixture models, and it is known to provide good estimates under quite general conditions (see the discussion in chapter 3). Nonetheless, as one could expect, the estimates may get worse in the case of intersecting clusters. As we pointed out in Chapter 1, the reason behind this particular feature is common to many different clustering techniques: they are based on a distance, and therefore they may not be able to detect overlapping regions. Consequently, it is not possible to provide a proper classification of the observations lying in those regions. Whatever the type of metric distance we adopt, the points in the intersecting region will be allocated to the nearest centroid ("nearest" in the sense of the adopted distance). This occurs also in the case of *mclust* algorithm, due to the underlying maximum likelihood method, based on a density which may be interpreted as prportional to a "kernel distance" (see the discussion in Chapter 1 and the discussion on the *Mclust* library in Chapter 3). Clearly, this circumstance can occur if the allocations are carried out by a *MAP* (*maximum a posteriori*) rule as in the case of $mclust$ (see Chapter 3), while in a *fuzzy* framework this can not occur because there is not

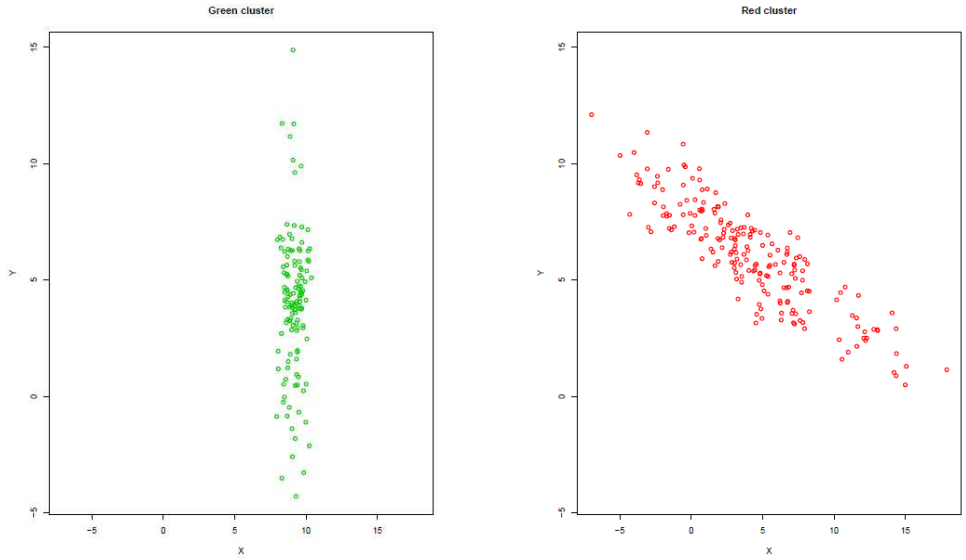
univoque membership for the observations. So, in the case of Gaussian mixtures, the maximum likelihood principle (via the *EM* algorithm) togheter with parameters estimates and a *hard* clustering allocation, induces some kind of "density designed" thresholds (see below), which establish a specific partition. It is clear that this feature may provide good estimates if clusters are not intersecting. But if this is not the case, it may induce a non-negligible bias in the parameters estimates, due to the chance of assigning points in the overlapping region to one and only one cluster, defined as the cluster with the nearest centroid (see the following Figure).

Fig. 5.2: Real partition and partition estimated by *Mclust*



Note that intersecting regions can be properly detected, because "Gaussian kernel distance" allows for the intersection of the clusters convex hulls (that is intersecting areas), but there are no overlapping regions, because all the points in the intersecting region are assigned only to one and only one cluster (see discussion in Chapter 1). So, despite of the allmost excellent contours detection, when we turn to allocate units to components, which correspond to clusters, via the MAP procedure, quite "unnatural" clusters are produced. In this case, a sort of "broken" ellipsoids (cluster red and green) arise, delimited by a sort of "density designed" thresholds, as depicted in the following figures.

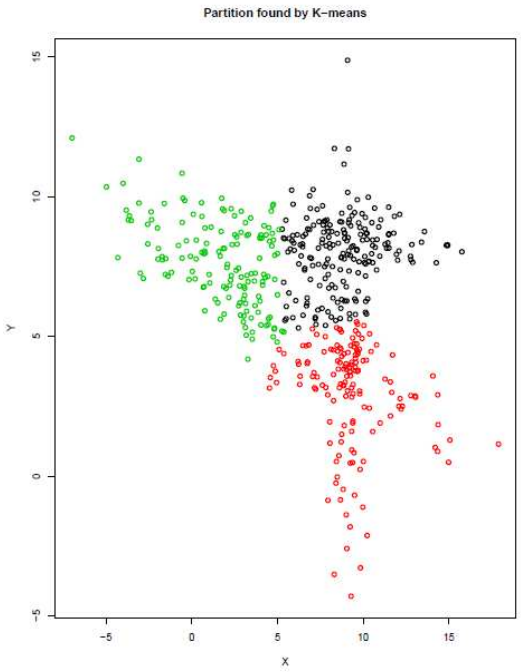
Fig. 5.3: “Broken” clusters in the partition provided by *Mclust*



Note that a case similar to the one depicted in Figure 2 occurs *every* time we consider overlapping clusters. Thus, when overlapping regions are present, the maximum likelihood method, based on a Gaussian density, may introduce (if used togheter with a *MAP* rule) a misclassification, which may in turn influence parameters estimates. The corresponding partition could be improved if we allocate points at random in the overlapping regions to the corresponding clusters! But this would not be a rationale approach. So, one possible solution is to exploit, togheter with a prespecified distance, a different criterion to allocate points lying in overlapping regions. As we will see in the following, this is what we try to do by introducing a suitable skewness function.

It can be observed that the partition provided by the maximum likelihood method (with *MAP* rule) is often more appropriate than the partition carried out by the criterion of minimum within clusters variance underlying the *K-means* solution, as depicted in the following Figure 5.4 (see below). It is in fact evident that *K-means* solution does not reproduce intersecting areas at all, while *Mclust* solution do (see above).

Fig. 5.4: Partition found by *K-means*



This may be understood as a consequence of two facts: first, we are considering Gaussian mixtures, and a Gaussian density-based approach should have some home-court advantage. The second reason, and the most interesting for our purpose, is related to the different distances involved, respectively, a Gaussian kernel distance and the Euclidean distance. The former is a generalization of the latter, considering also the information contained in the covariance matrix (it is a Mahalanobis-type distance). Note that this feature is particularly relevant in all cases of overlapping clusters, not only in the case of Gaussian mixtures. This clearly depends on the information provided by the covariance matrix, which implies a more flexible concept of neighborhood with respect to the one embedded in the Euclidean distance. As we can see from Figures 5.2(b) and 5.3 above, all the points to the right of the green cluster contour are properly allocated by $Mclust$ to the red cluster, even if such points are closer to the green cluster centroid than to the red cluster centroid (they are closer according to Euclidean distance). Should we have adopted the Euclidean distance, we would have allocated all these points simply to the nearest centroid (the green cluster centroid), thus increasing the misclassification rate and vanishing the possibility to catch the intersecting region, as it occurs in the *K-means* solution (see Figure 5.4 above). In this sense, *K-means* induces partitions which

exclude *a priori* the intersection of the clusters convex hulls (that is the intersecting regions), due to the underlying criterion of minimum within cluster variance. This criterion, in fact, induces a partitioning of the data space into *Voronoi* cells, that is, roughly speaking, regions containing all the points closer to that centroid than to any other centroid, as can be seen in the Figure 5.4 (clearly in the Figure there is not a "complete" *Voronoi* diagram, because only clusters sets are represented).

Nevertheless, the feature of the Gaussian kernel distance has an interesting "graphical consequence": it projects clusters in a higher dimension space. In fact, if we ideally put the Figure 5.2(b) in $\mathbb{R}^3$ we would see the "black" cluster as the nearest in front of us, hiding a region of the "green" cluster, and the same occurring between the "green" and the "red" one. In other words, it's like it would represent the clusters in different hyperplanes in $\mathbb{R}^3$, at different depth.

Anyway, this feature of the Gaussian kernel distance (togheter with the *MAP* allocation) does not allow for overlapping regions at all. This feature could be called a "hard-shield shaped classification", meaning that no points of a cluster can enter in the convex hull of another cluster. Clearly, this feature is also implied by the *Voronoi* cell partition of the *K-means* solution, with the further drawback of the *a priori* exclusion of intersecting regions. Note that, on the contrary, a "soft-shield shaped classification" would be in conflict with clusters compactness, a good feature usually required in clustering results. But this would be an appropriate feature when clusters are actually separated, whereas it would induce misclassification in the case of intersecting clusters.

Finally, note that the case depicted in Figure 5.2 is a sort of "best" case with respect to the $Mclust$ solution, where intersecting regions are properly reproduced, and we expect centroid estimates to be not severely biased. Often this is not the case, and the presence of overlapping clusters induces a significantly higher bias in the $Mclust$ clustering results. Nonetheless, for convenience of exposition, we choose to discuss the ability of Gaussian kernel distance to reproduce the intersection of cluster convex hulls, in opposition to the *Voronoi* cell partition induced by the *K-means* method.

This is the main idea of the present work: to develop a suitable criterion to detect Gaussian clusters, while allowing for overlapping clusters. So, from a non-parametric point of view, what we need is a sensible method to detect elliptical clusters, exploiting some suitable skewness based function to assign observations in the intersecting regions. Thus, in the following, we define a skewness function and the corresponding cluster validity index.

5.2 A skewness function and a related cluster validity index

Once established the particular cluster shape we are looking for, the question is: what kind of skewness function may we adopt to detect such clusters? As a starting point, we choose a non-metric distance based on the concept of point symmetry, as formulated in Su and Chou (2001). It is defined in Chapter 4, but for convenience of discussion we report it below. So, let us consider n observations $\boldsymbol{x}_i$, $i = 1, ..., n$, and a cluster centroid $\bar{\boldsymbol{x}}$. The point symmetry distance between an individual $\boldsymbol{x}_j$ and the cluster centroid $\bar{\boldsymbol{x}}$ is defined as:

$$d_s\left(\boldsymbol{x}_j, \bar{\boldsymbol{x}}\right) = \min_{i=1,...,n,\, i \neq j} \frac{\parallel \left(\boldsymbol{x}_j - \bar{\boldsymbol{x}}\right) + \left(\boldsymbol{x}_i - \bar{\boldsymbol{x}}\right) \parallel}{\left(\parallel \boldsymbol{x}_j - \bar{\boldsymbol{x}} \parallel + \parallel \boldsymbol{x}_i - \bar{\boldsymbol{x}} \parallel\right)}$$

If the right hand term of $d_s\left(\boldsymbol{x}_j, \bar{\boldsymbol{x}}\right)$ is minimized for a specific i, $\boldsymbol{x}_i$ is defined as the symmetrical pattern relative to $\boldsymbol{x}_j$ with respect to $\bar{\boldsymbol{x}}$. Note that the minimum value for $d_s\left(\boldsymbol{x}_j, \bar{\boldsymbol{x}}\right)$ is 0, occurring when the pattern $\boldsymbol{x}_i = 2\bar{\boldsymbol{x}} - \boldsymbol{x}_j$ exists in the observed sample, since in this case the numerator is identically null. Note that Su and Chou (2001) introduce the denominator to normalize the point symmetry distance, and to make the point symmetry distance insensitive to the Euclidean distances $\parallel \boldsymbol{x}_j - \bar{\boldsymbol{x}} \parallel$ and $\parallel \boldsymbol{x}_i - \bar{\boldsymbol{x}} \parallel$. Nonetheless, while the first task is achieved, the second one inevitably fails. In fact, as pointed out by Vijendra and Laxman (2015) the $d_s\left(\cdot, \cdot\right)$ measure favours data point that are far away (if we have at least two symmetrical observations to compare), and this may degrade the corresponding robustness. This circumstance is shown with the following example. Let $\boldsymbol{x}_j = (5; 5)$, $\boldsymbol{x}_i = (7; -1)$, $\boldsymbol{x}_{i+1} = (10; -9)$, $\bar{\boldsymbol{x}} = (0; 0)$; find the most symmetric point of $\boldsymbol{x}_j$ with respect to $\bar{\boldsymbol{x}}$ when $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i+1}$ are the possible choices,

$$d_s\left(\boldsymbol{x}_j, \bar{\boldsymbol{x}}\right) = \min\left\{\frac{\parallel \left(\boldsymbol{x}_j - \bar{\boldsymbol{x}}\right) + \left(\boldsymbol{x}_i - \bar{\boldsymbol{x}}\right) \parallel}{\left(\parallel \boldsymbol{x}_j - \bar{\boldsymbol{x}} \parallel + \parallel \boldsymbol{x}_i - \bar{\boldsymbol{x}} \parallel\right)}, \frac{\parallel \left(\boldsymbol{x}_j - \bar{\boldsymbol{x}}\right) + \left(\boldsymbol{x}_{i+1} - \bar{\boldsymbol{x}}\right) \parallel}{\left(\parallel \boldsymbol{x}_j - \bar{\boldsymbol{x}} \parallel + \parallel \boldsymbol{x}_{i+1} - \bar{\boldsymbol{x}} \parallel\right)}\right\}$$

$$d_s\left(\boldsymbol{x}_j, \bar{\boldsymbol{x}}\right) = \min\left\{\frac{\sqrt{20}}{\left(\sqrt{50} + \sqrt{58}\right)}, \frac{\sqrt{41}}{\left(\sqrt{50} + \sqrt{181}\right)}\right\} = \min\left\{0.32,\ 0.31\right\}$$

The data point $\boldsymbol{x}_{i+1}$ is selected as the most symmetrical point of $\boldsymbol{x}_j$ relative to the centroid $\bar{\boldsymbol{x}}$, although the most symmetrical point with respect to $\boldsymbol{x}_j$ is clearly $\boldsymbol{x}_i$ (note that only numerator measures the symmetry degree, and we have for this example $\parallel (\boldsymbol{x}_j - \bar{\boldsymbol{x}}) + (\boldsymbol{x}_i - \bar{\boldsymbol{x}}) \parallel = \sqrt{20} < \parallel (\boldsymbol{x}_j - \bar{\boldsymbol{x}}) + (\boldsymbol{x}_{i+1} - \bar{\boldsymbol{x}}) \parallel = \sqrt{41}$, indicating that $\boldsymbol{x}_i$ is the most symmetrical point). This result is due to the impact of the denominator on the $d_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}})$ value. This shows that $d_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}})$ favors data points that are far away (see also Figure 4.3 in Chapter 4).

For this reason, we simply decided to remove the denominator from the expression above, and defined the skewness function $f_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}})$

$$f_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}}) = \min_{i=1,\ldots,n,\, i \neq j} \parallel (\boldsymbol{x}_j - \bar{\boldsymbol{x}}) + (\boldsymbol{x}_i - \bar{\boldsymbol{x}}) \parallel$$

to avoid the drawbacks discussed above. Using $f_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}})$ (that is ignoring the denominator) on the example above, we would have the following result:

$$f_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}}) = \min \left\{ \sqrt{20},\, \sqrt{41} \right\} = \min \{4.47,\, 6.40\}$$

In the previous example, $f_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}})$ is able to properly recognize $\boldsymbol{x}_i$ as the most symmetrical point of $\boldsymbol{x}_j$. As we will see in the following section, when embedded in a suitable function, $f_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}})$ avoids also a further problem for the Su and Chou (2001) $d_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}})$, namely the lack of robustness to "point symmetry interclusters distance" (see Chapter 4). In fact, for similar values of point symmetry, $f_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}})$ will favour both centroids and observations close to the specific $\boldsymbol{x}_j$ for which we computed it.

Starting from this skewness function, we introduce a skewness-based index, SBI_k. Essentially, it works as an index measuring the skewness in a specific cluster. Thus, for cluster k, $k = 1, ..., K$, with cardinality $|S_k|$, SBI_k is defined as the sum of $f_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}}_k)$ over each $j \in S_k$ with respect to the centroid of the cluster $\bar{\boldsymbol{x}}_k$:

$$SBI_k = \sum_{j \in S_k} f_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}}_k) = \sum_{j \in S_k} \min_{i \in S_k, i \neq j} \parallel (\boldsymbol{x}_j - \bar{\boldsymbol{x}}_k) + (\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k) \parallel$$

On the basis of such cluster-specific index, we introduce a cluster validity index, SBI, which is simply the sum of the SBI_k, $k = 1, ..., K$:

$$SBI = \sum_{k=1}^{K} SBI_k = \frac{1}{(K \times D)} \sum_{k=1}^{K} \left\{ \sum_{j \in S_k} f_s\left(\boldsymbol{x}_j, \bar{\boldsymbol{x}}_k\right) \right\} =$$

$$= \frac{1}{(K \times D)} \sum_{k=1}^{K} \left\{ \sum_{j \in S_k} \min_{i \in S_k, i \neq j} \parallel \left(\boldsymbol{x}_j - \bar{\boldsymbol{x}}_k\right) + \left(\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k\right) \parallel \right\}$$

Clearly, SBI is intended to account for the overall skewness in a given partition. In this sense, the lower is the SBI, the better is the clustering performance that we expect to achieve (the corresponding clusters are less skewed). This is a sensible choice for clusters corresponding to elliptical distributions, which will be associated to a low amount of skewness, that is a low value for SBI. This is an important feature of the proposed method, as well as a relevant step in the proposed algorithm (see next section). In particular, apart from the proposed algorithm, in Chapter 6 we will test SBI in many different Gaussian mixtures scenarios, showing by simulations its quite encouraging ability to validate clustering results when $Mclust$ and *K-means* algorithms are compared.

Finally, we note that, obviously, there are many other possible choices for a skewness index (in addition to those considered here and in Chapter 4). Just to give an example, a well-known skewness index has been introduced by Mardia (1970), and it is defined as

$$\beta = E\left\{ \left[\left(\mathbf{X} - \boldsymbol{\mu}\right)' \boldsymbol{\Sigma}^{-1} \left(\mathbf{Y} - \boldsymbol{\mu}\right) \right]^3 \right\}$$

where $\boldsymbol{\mu}$ denotes the mean vector, $\boldsymbol{\Sigma}$ the covariance matrix, $\mathbf{X}$ and $\mathbf{Y}$ represent two independent and identically distributed random vectors from the same distribution. In the case of multivariate Gaussian distribution this index will be exactly zero, while its sample counterpart will be around zero for low-dimensional data sets, or more precisely $E\left\{ \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \left[\left(\boldsymbol{x_i} - \bar{\boldsymbol{x}}\right)' \boldsymbol{S}^{-1} \left(\boldsymbol{x_j} - \bar{\boldsymbol{x}}\right) \right]^3 \right\} = \frac{D(D+1)(D+2)}{n} + o(n^{-1})$; for a detailed proof see Sumikava, Koizumi and Seo (2014). However, due to computational inefficiency and instability (this index involves the inverse of the sample covariance matrix) we have faced in a simulation study, we did not consider it further. Anyway, for a brief review and analysis of other skewness indexes, see Balakrishnan and Scarpa (2012).

5.3. The skewness based objective function

Starting from the discussion above, we have developed a skewness-based allocation method, $SBAM$. The main problem, as usual, is to find a suitable function to allocate observations to different clusters.

Initially, we attempted to exploit different objective functions, all of them based on different formulations of the skewness function $f_s(\cdot,\cdot)$ defined above. Unfortunately, we did not find a way to make them work in case of well separated clusters, i.e. in the case of low variances and centroids well far away. This issue, actually, is an intrisic feature of the skewness functions: without specific constraints, skewness remains a *geometrical* property of the analyzed objects, not necessarily regarding distances between the same objects. To show this issue, we may consider the following example: consider two small sized samples in the univariate case, $X = 10, 100, 190$ and $Y = 90, 100, 110$. It is straighforward to observe that skewness will be exactly zero in both cases, regardless the differences in terms of distances from the centroid. Therefore, a skewness index may be used to define the geometrical shape for the clusters but not to fix the corresponding radius.
So, to overcome this drawback, we have at least the following two strategies:
1. choose a suitable threshold on the skewness values, beyond which we consider only a distance function to allocate the points.
2. find a linear combination of skewness function with a proper distance
Naturally, it is possible to merge both strategies; some examples of the first strategy have already been discussed in the previous section. We followed both, but as we will see later, the second one seems to work well in our context.

In a way, by neglecting distances, the skewness function tends to select clusters overdispersed with respect to the corresponding centroids, so we need to handle this drawback by means of a suitable distance function. Since we are in a model based context, where the Gaussian distribution is central, it seems natural to exploit the Mahalanobis (1936) distance, defined by

$$\sqrt{(\boldsymbol{x}-\bar{\boldsymbol{x}})' \mathbf{S}^{-1} (\boldsymbol{x}-\bar{\boldsymbol{x}})}$$

where $\bar{\boldsymbol{x}}$ denotes the centroid, $\mathbf{S}$ the sample covariance matrix, and $\boldsymbol{x}$ a random vector. Clearly, the choice of this distance is related to the elliptical shape we are

looking for. Not surprisingly, the distance itself is clearly related to the multivariate Gaussian density. It is, in fact, straightforward to show that constant density curves for a multivariate Gaussian are themselves ellipsoids centred at $\bar{\boldsymbol{x}}$, satisfying the equality

$$(\boldsymbol{x} - \bar{\boldsymbol{x}})' \, \mathbf{S}^{-1} \, (\boldsymbol{x} - \bar{\boldsymbol{x}}) = c^2$$

with c arbitrary real constant. Unfortunately, due to the need of calculating the inverse of the sample covariance matrix $\mathbf{S}$, the Mahalanobis distance resulted unstable in the simulations scenarios we have considered. To overcome this problem, we decided to exploit the Manhattan distance which, for an observation $\boldsymbol{x} \in \mathbb{R}^D$ and a centroid $\bar{\boldsymbol{x}} \in \mathbb{R}^D$, is defined as follows:

$$|\, \boldsymbol{x} - \bar{\boldsymbol{x}}| = \sum_{d=1}^{D} |\, x_d - \bar{x}_d|$$

The choice of the Manhattan distance is also related to the problem of outliers yet discussed in Chapter 2 (in *K-means* framework). This distance, in fact, is less sensitive to outliers when compared to the Euclidean one.

Before defining the objective function, we would note a relevant difference between the proposed strategy and the other skewness-based approaches described until now. We choose to consider a normalized version for both the Manhattan distance and the skewness function $f_s(\cdot, \cdot)$. To this end, let us consider an observation $\boldsymbol{x}_i$ and K centroids $\bar{\boldsymbol{x}}_k$, $k = 1, ..., K$. The normalized versions of the Manhattan distance and the skewness function, respectively $d^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k)$ and $f^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k)$, are defined as follows:

$$d^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k) = \frac{|\, \boldsymbol{x}_i - \bar{\boldsymbol{x}}_k|}{\sum_{k=1}^{K} |\, \boldsymbol{x}_i - \bar{\boldsymbol{x}}_k|} = \frac{\sum_{d=1}^{D} |\, x_{id} - \bar{x}_{kd}|}{\sum_{k=1}^{K} \sum_{d=1}^{D} |\, x_{id} - \bar{x}_{kd}|}$$

$$f^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k) = \frac{f_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}}_k)}{\sum_{k=1}^{K} f_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}}_k)} = \frac{\min_{i=1,...,n,\, i \neq j} \parallel (\boldsymbol{x}_j - \bar{\boldsymbol{x}}_k) + (\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k) \parallel}{\sum_{k=1}^{K} \{\min_{i=1,...,n,\, i \neq j} \parallel (\boldsymbol{x}_j - \bar{\boldsymbol{x}}_k) + (\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k) \parallel\}}$$

Thus, for each observation $\boldsymbol{x}_i$ and each measure we get two vectors with K values, each summing to 1, which we define as $\boldsymbol{d}^N(\boldsymbol{x}_i)$ and $\boldsymbol{f}^N(\boldsymbol{x}_i)$. It is worth noticing that this formulation allow for a *fuzzification* of the procedure, since it is

possible to assign different degrees of cluster membership to an observation on the basis of the corresponding normazlized values. There are several ways to implement such a procedure. A naive one is to take the complement to 1 for each of the K values, and divide for $K-1$, but we would not preserve the ratios between the normazlized values. A more sensible strategy could be the following. Let us suppose we are in the case $K=3$ clusters and an observation has the following normalized values with respect to its Manhattan distances from the K centroids $(0.1, 0.4, 0.5)$. We would transform these values in probability-type values (clearly they are not probabilities) just taking the inverse of the ratios between values and imposing that new values sum to 1, that is the first value would be four times the second and the third value would be $4/5$ with respect to the new second value, thus obtaining $(0.69, 0.172, 0.138)$ which sum to 1.

Nonetheless, *fuzzification* is not the reason why we adopt such a choice, since as pointed out in Chapter 1, we are mainly interested in *hard* partitioning, attempting to catch the overlapping regions. Rather we adopt this strategy to put both measures on the same scale, such that we could compare them. Essentially, Manhattan distance and skewness function can be regarded as two different sources of information, which may be in conflict. So, we want to exploit both, but without mixing them in a single term (see the following discussion). To this end, first we take the minimum over k for both the normalized expressions of Manhattan distance and skewness function, say respectively $d_{min}(\boldsymbol{x}_i)$ and $f_{min}(\boldsymbol{x}_i)$, defined as follows:

$$d_{min}(\boldsymbol{x}_i) = \min_k \frac{|\,\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k|}{\sum_{k=1}^{K} |\,\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k|} = \min_k \frac{\sum_{d=1}^{D} |\,x_{id} - \bar{x}_{kd}|}{\sum_{k=1}^{K}\sum_{d=1}^{D} |\,x_{id} - \bar{x}_{kd}|}$$

$$f_{min}(\boldsymbol{x}_i) = \min_k \frac{f_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}}_k)}{\sum_{k=1}^{K} f_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}}_k)} = \min_k \frac{\min_{i=1,\ldots,n,\, i\neq j} \parallel (\boldsymbol{x}_j - \bar{\boldsymbol{x}}_k) + (\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k) \parallel}{\sum_{k=1}^{K} \{\min_{i=1,\ldots,n,\, i\neq j} \parallel (\boldsymbol{x}_j - \bar{\boldsymbol{x}}_k) + (\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k) \parallel\}}$$

Note that, in general, the k-th centroid which minimizes the normalized Manhattan distance is not the same which minimizes the skewness function. This is an obvious consequence of the different information provided by the two functions: the centroid nearest to the observation $\boldsymbol{x}_i$ is not necessarily the one $\boldsymbol{x}_i$ is most symmetric to (in the sense of our skewness function). Actually, it can be seen that, in this context, the difference between the two terms $d_{min}(\boldsymbol{x}_i)$ and $f_{min}(\boldsymbol{x}_i)$ plays a relevant role. For this purpose, assume that the k-th centroid which minimizes the normalized Manhattan distance and the skewness function is the same. In this case, the two measures are not in conflict, and we are not interested in observing

the difference between $d_{min}(\boldsymbol{x}_i)$ and $f_{min}(\boldsymbol{x}_i)$, since the allocation of the point $\boldsymbol{x}_i$ is not ambiguos: the unit will be assigned to the k-th centroid which minimizes both $d^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k)$ and $f^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k)$. But, in general, this is not the case, and we will have two different centroids minimizing $d^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k)$ and $f^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k)$, say k_1 and k_2. The difference between the two terms, $d_{min}(\boldsymbol{x}_i)$ and $f_{min}(\boldsymbol{x}_i)$, is relevant, since it implicitly determines the relative discriminant power of each function in the allocation of the observation $\boldsymbol{x}_i$. To explain, let us suppose we are in a case with $K = 5$ clusters, and we have $d_{min}(\boldsymbol{x}_i) = 0.05$ and $f_{min}(\boldsymbol{x}_i) = 0.19$ (the maximum is clearly $1/K$ for both the functions). So, the difference $d_{min}(\boldsymbol{x}_i) - f_{min}(\boldsymbol{x}_i) = -0.14$ tells us that the distance has a strong discriminant power with respect to the skewness function (thus implying a smaller ambiguity in the allocation of $\boldsymbol{x}_i$). Given the importance of this information, we decided to embed it in the objective function, defining it as $\gamma_i = d_{min}(\boldsymbol{x}_i) - f_{min}(\boldsymbol{x}_i)$. Note that from the expression we get immediatly a maximum equal to $\frac{1}{K}$, which occurs when $d_{min}(\boldsymbol{x}_i) = \frac{1}{K}$ (that is no discriminant power, with K centroids perfectly equidistant from $\boldsymbol{x}_i$) and $f_{min}(\boldsymbol{x}_i) = 0$, that is maximally discriminant (with a centroid perfectly symmetric to $\boldsymbol{x}_i$ in the sense of our skewness function). For the same reason, the minimum is equal to $-\frac{1}{K}$, which occurs in the opposite case. This choice has also other advantages, which will be discussed below.

Now, linearly combining the two functions of skewness and Manhattan distance, we get for every $k = 1, ..., K$ the proposed objective function:

$$\mathrm{Q}(X, K) = \sum_{i=1}^{n} \left(d^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k) + \theta_i f^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k) \right) =$$

$$\sum_{i=1}^{n} \left(\frac{|\,\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k|}{\sum_{k=1}^{K} |\,\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k|} + \theta_i \frac{\min_{i=1,...,n,\, i \neq j} \parallel (\boldsymbol{x}_j - \bar{\boldsymbol{x}}_k) + (\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k) \parallel}{\sum_{k=1}^{K} \{\min_{i=1,...,n,\, i \neq j} \parallel (\boldsymbol{x}_j - \bar{\boldsymbol{x}}_k) + (\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k) \parallel\}} \right)$$

where

$$\theta_i = \exp(\gamma_i \cdot K \cdot \delta)$$

that is $\theta_i = \exp(\gamma_i \cdot K \cdot \delta)$ if $\gamma_i > 0$ such that skewness function will have a major impact in the allocation of $\boldsymbol{x}_i$, whereas $\theta_i = \frac{1}{\exp(\gamma_i \cdot K \cdot \delta)}$ if $\gamma_i < 0$ (the opposite case), and $\theta_i = 1$ if $\gamma_i = 0$. Note that we have specified the subscript i to stress the dependence of this term on the observation $\boldsymbol{x}_i$, while the reasons why we introduce

also K and $\delta \geq 0$ can be sketched as follows. When K increases, the difference between $d^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k)$ and $f^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k)$ becomes more relevant. In fact, a value of $\gamma_i = 0.05$, say, is not substantial in the case of $K = 2$ clusters, while the same value $\gamma_i = 0.05$ in a case of $K = 20$ clusters represents the maximum (equal to $\frac{1}{K} = 0.05$, corresponding to $f_{min}(\boldsymbol{x}_i) = 0$ and a $d_{min}(\boldsymbol{x}_i) = 0.05$, see above). So, for a given γ_i and δ, a high value of K will determine a rapid increase or decrease in θ_i (via the exponential function), depending on the sign of γ_i. The term δ has just a theoretic role (in our simulation study we ignore it, fixing $\delta = 1$). For a given K, it can be interpreted as a parameter which rules the sesitivity of θ_i to the values of γ_i. In fact, since $\gamma_i = d_{min}(\boldsymbol{x}_i) - f_{min}(\boldsymbol{x}_i)$, a high value of δ implies that even a small difference between $d^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k)$ and $f^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k)$ will produce a high value of θ_i if $d^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k) > f^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k)$ or a small value of θ_i if $d^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k) < f^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k)$. This means that a small difference in $d^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k)$ and $f^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k)$ will have a heavy impact on the linear combination, and then on the allocation of $\boldsymbol{x}_i$.

In particular, to clarify the role of δ we discuss two extreme cases: the first is obtained when $\delta = 0$. Here, we will have $\theta_i = 1$ regardless of the sign of γ_i. Thus, we would have a simple linear combination where each function has its weight in the objective function, proportional to the values of $d^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k)$ and $f^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k)$, but regardless of the difference γ_i, that is $\mathrm{Q}(\boldsymbol{x}_i, k) = d^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k) + f^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k)$. In the second case, which is probably the most relevant, if δ assumes an arbitrary high value (excluding the case $\gamma_i = 0$) we approach an objective function of the following type:

$$\mathrm{Q}(\boldsymbol{x}_i, k) = (1 - \xi)\, d^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k) + \xi f^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k)$$

with ξ boolean variable, equal to 1 iff $\gamma_i > 0$ and 0 otherwise. In fact, if $\gamma_i > 0$ and δ is arbitrary high, θ_i will be high, and only the skewness function impacts on the linear combination, such that $\boldsymbol{x}_i$ will be allocated to the cluster k^* for which $f^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_{k*}) = f_{min}(\boldsymbol{x}_i)$. Viceversa if $\gamma_i < 0$ and δ is arbitrary high, we have $\theta_i \simeq 0$ and only the Manhattan function will determine the allocation of $\boldsymbol{x}_i$, to the cluster k^* for which $d^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_{k*}) = d_{min}(\boldsymbol{x}_i)$.

Note that, excluding the case $\gamma_i \simeq 0$, this holds even for negligible values of γ_i. In other words, the allocation of $\boldsymbol{x}_i$ is enterely decided by the sign of γ_i. This rule corresponds to find the function with the minimum value between $d_{min}(\boldsymbol{x}_i)$ and $f_{min}(\boldsymbol{x}_i)$, that is $\min(d_{min}(\boldsymbol{x}_i), f_{min}(\boldsymbol{x}_i))$, and then apply to it a sort of mAP (*minimum a posteriori*) rule. To explain, let us suppose that $\min(d_{min}(\boldsymbol{x}_i), f_{min}(\boldsymbol{x}_i)) = d_{min}(\boldsymbol{x}_i)$, this rule assigns $\boldsymbol{x}_i$ to the cluster k^* for which $d^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_k)$ is minimum, that is $d^N(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_{k*}) = d_{min}(\boldsymbol{x}_i)$.

This example is intended to show that in the current framework, this kind of rule can be regarded as a special case of the above linear combination. Moreover, in this case the rule would imply a substancial loss of information: if we used it, we preserve only the sign of γ_i, while ignoring its magnitudo. As we pointed out above, this is a precious information, since Manhattan distance and skewness function could be in conflict. To show this issue, let us consider the following example in a case of $K = 3$ clusters. Suppose we have $\boldsymbol{d}^N(\boldsymbol{x}_i) = (0.1, 0.12, 0.78)$, $\boldsymbol{f}^N(\boldsymbol{x}_i) = (0.6, 0.11, 0.29)$, with $d_{min}(\boldsymbol{x}_i) = 0.1$, $f_{min}(\boldsymbol{x}_i) = 0.11$ and $\gamma_i = d_{min}(\boldsymbol{x}_i) - f_{min}(\boldsymbol{x}_i) = -0.01$. According to this kind of rule, we would assign $\boldsymbol{x}_i$ to the first cluster, while there is a strong concordance of both measures to assign it to the 2-th cluster, as our method would have done, considering almost equal weights for both measures in the objective function (with $\delta = 1$ we have $\theta_i = \frac{1}{\exp(\gamma_i \cdot K \cdot \delta)} = 0.97$). Note that also the opposite case of $\delta = 0$ implies a loss of information, since it implies $\theta_i = 1 \, \forall \gamma_i$.

Apart from these, there are other interesting features of the proposed objective function. Embedding $f_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}})$ in the linear combination, we are able to avoid another issue with the Su and Chou (2001) $d_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}})$ measure, the lack of robustness to "point symmetry interclusters distance" (see Chapter 4, Figure 4.3). For similar values of point symmetry, the proposed objective function will favour both the centroids and the observations that are close to the specific $\boldsymbol{x}_j$, due to the impact of the Manhattan distance on linear combination. For the same reason, we solved the drawback related to the violation of the "closure property" (see Chapter 4, Figure 4.6), without imposing a constraint. In fact, it hardly occurs to allocate a point $\boldsymbol{x}_j$ in a cluster on the basis of another point $\boldsymbol{x}_i$ which result to be far from the centroid $\bar{\boldsymbol{x}}$, because, if $\bar{\boldsymbol{x}}$ is near to $\boldsymbol{x}_j$, $f_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}})$ will reject $\boldsymbol{x}_i$ in favour of another point, say $\boldsymbol{x}_l$, closer to the centroid, and if $\bar{\boldsymbol{x}}$ is far from $\boldsymbol{x}_j$, $\boldsymbol{x}_j$ will not be associated to $\bar{\boldsymbol{x}}$ due to the impact of the Manhattan distance. Clearly, these drawbacks can occur also adopting the proposed method, when clusters are close to each other and/or overlapping; in these cases, the impact of distance will be smaller. But, in such cases the influence of these drawbacks on clustering results will be much smaller, and this is the inevitable price we pay to detect overlapping clusters.

5.4 Sbam (Skewness-Based Allocation Method)

Finally, we report a sketch of our method, $Sbam$, followed by some remarks:

1. Initialize K centroids (Model based or random choice)
2. For $t = 1, 2, ...$

For $i = 1, ..., n$ assign $\boldsymbol{x}_i$ to cluster k^* minimizing the objective function:

$$k^* = \text{argmin}_{\text{k=1,...,K}} \left(\frac{\left| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_k^t \right|}{\sum_{k=1}^{K} \left| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_k^t \right|} + \theta_i \frac{\min_{i=1,...,n,\, i \neq j} \parallel (\boldsymbol{x}_j - \bar{\boldsymbol{x}}_k^t) + (\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k^t) \parallel}{\sum_{k=1}^{K} \left\{ \min_{i=1,...,n,\, i \neq j} \parallel (\boldsymbol{x}_j - \bar{\boldsymbol{x}}_k^t) + (\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k^t) \parallel \right\}} \right)$$

where

$$\theta_i = \exp\left(\gamma_i \times K \times \delta\right)$$

with $\gamma_i = d_{min}\left(\boldsymbol{x}_i\right) - f_{min}\left(\boldsymbol{x}_i\right)$

3. Update the estimates of $\bar{\boldsymbol{x}}_k^{t+1} = \frac{1}{\left|S_k^t\right|} \sum_{i \in S_k^t} \boldsymbol{x}_i$

4. Repeat steps 2-3 until no data point membersip changes and $SBI^t \geqslant SBI^{t+1}$ where

$$SBI^t = \sum_{k=1}^{K} \left\{ \sum_{j \in S_k^t} f_s\left(\boldsymbol{x}_j, \bar{\boldsymbol{x}}_k^t\right) \right\}$$

In the previous section, we have discussed how the proposed method avoids some drawbacks without referring to clusters membership. In fact, neither the skewness function $f_s\left(\boldsymbol{x}_j, \bar{\boldsymbol{x}}\right)$ nor the Manhattan distance depends on the clusters sets, i.e. they both consider only the centroids $\bar{\boldsymbol{x}}_k$, $k = 1, ..., K$. This seems to be a desirable feature. In fact, it is well known the strong dependence of clustering results on starting settings (for centroids and/or partition), so that limiting the functions domain to pre-specified clusters did not seem a good choice, while obviously it was inevitable to choose initial values for centroids. We adopt such a restriction, rather, in the SBI step, when clusters are yet formed, and we aim at defying a measure of partition "goodness of fit" which depends on each cluster. Thus, this feature can be considered an advantage over all those methods which work starting from initial partitions, such as the other skewness-based approaches discussed in Chapter 4.

However, this is not the only advantage of the proposed clustering approach. A comparison with other skewness-based methods, dicussed in Chapter 4, may be instructive. For instance, both Saha and Bandyopadhyay (2010) and Vijendra and Laxman (2015) mix the two informations provided by the distance (Euclidean) and the skewness functions. In this sense, they are not able to distinguish, in their formulations, the differences in magnitudo of the two sources of information, distance (Euclidean) and skewness functions. Most papers discussed in Chapter 4, that is Saha and Bandyopadhyay (2010), Vijendra and Laxman (2015), Su and Chou (2001) and Saha (2017) introduce a fixed threshold, below which we may consider

the skewness to determine the allocations (see the discussion in Chapter 4). Apart from Su and Chou (2001), where this threshold is arbitrary, in all other cases this seems to be well justified. But the fact remains that fixing a threshold we loose flexibility.

From this point of view, the formulation we propose do have more flexibility along three dimensions: first, we separate the informations provided by the distance and the skewness functions, and we use both, without fixing a threshold which establish which one is to be considered. Second, we normalize both the Manhattan distance and the skewness function, and compare them extracting a further significant information. Last but not least, the weight θ_i is not fixed, but change according to the specific informations we receive for a particular $\boldsymbol{x}_i$ at each step of the algorithm.

To our knowledge, the proposed method is the only one exploiting a linear combination of a distance and a skewness function with a variable threshold, automatically and flexibly determined.

Essentially, the proposed objective function is meant to calibrate two opposite tendencies: the Manhattan distance strive for iper-concentrated clusters, while skewness function tends to induce overdispersed ones. The more the clusters are overlapping, the more we want the skewness to weigh heavily on the allocation process. We expect this to occur automatically, because highly overlapping clusters could imply closer centroids, so that the impact of the Manhattan distance will be reduced.

5.5 Beyond the gaussianity

It is worth noting that the attempt to detect elliptical shapes is suitable not only for Gaussian-shaped clusters, but in many other cases. This issue is relevant for our purposes, because it indicates a feasible direction for further developments of the proposed method. An interesting paper in this sense is Villaverde, Kosheleva and Ceberio (2010). As the title already states, "*Why Ellipsoid Constraints, Ellipsoid Clusters, and Riemannian Space-Time: Dvoretzky's Theorem Revisited*", the Authors provide mathematical justifications to the research of elliptical shapes in the clusters: "In many practical applications, we encounter ellipsoid constraints, ellipsoid-shaped clusters, etc. A usual justication for this ellipsoid shape comes from the fact that many real-life quantities are normally distributed, and for a multivariate normal distribution, a natural confidence set (containing the vast majority of the objects) is an ellipsoid. However, ellipsoids appear more frequently than normal

distributions. In this paper, we provide a new justication for ellipsoids based on a known mathematical result Dvoretzky's Theorem".

At the beginning of this paper, Authors discuss probabilistic and mathematical reason for such a statement, respectively involving Central Limit Theorem and Taylor expansion, but their most interesting claim is the Dvoretzky's Theorem, which proves Grothendieck's hypothesis, i.e. that convex sets in large dimensions have sections whose shape is close to ellipsoidal - the larger the dimension, the close this shape to the shape of an ellipsoid.

Even more interesting, the Authors point out the result provided by V. L. Milman (1971), who shows that "not only *there exists* an almost ellipsoidal shape, but also that *almost all* low-dimensional sections of a convex set have an almost ellipsoidal shape. Strictly speaking, he proved that for every $\varepsilon > 0$ the probability to get a shape which is more than ε-different from ellipsoidal goes to 0 as the dimension of the convex set increases".

This conclusion is particularly relevant in clustering problems, beacuse in this context one of the main issues is that "theoretically, each real-life object can be characterized by a point (vector) containing the results of measuring all possible quantities characterizing this object. In this theoretical description, objects are represented by points in a (very) high-dimensional space...However, in the real world, we only observe a few of these quantities. Thus, what we observe is a lower-dimensional section of a high-dimensional set – and we know that, according to Dvoretzky's theorem, this section is almost always almost ellipsoidal".

This kind of topic holds even in the form of a general physical experiment. In fact, as the Authors pointed out: "In general, a physical constraint actually has a form $g(x_1, ..., x_n, x_{n+1}, ..., x_N) \leqslant 0$ where $x_{n+1}, ..., x_N$ are quantities that we do not measure in this particular experiment. Thus, the corresponding n-dimensional constraint set $\{x = x_1, ..., x_n : g(x_1, ..., x_n) \leqslant 0\}$ is a *section* of the actual (unknown) multi-dimensional constraint set

$$\{x = x_1, ..., x_n, x_{n+1}, ..., x_N : g(x_1, ..., x_n, x_{n+1}, ..., x_N) \leqslant 0\}$$

and we already know that in almost all cases, such sections are almost ellipsoidal".

From this point of view, we can give at least two further directions for possible developments of the proposed method. The first deals with non Gaussian high-dimensional dataset, for which, according to the above discussion, we can expect low-dimensional profile with almost ellipsoidal shaped clusters. So, we could extend the field of application of our skewness-based clustering approach to these cases. In a sense, a sort of validation of this theory can be found in Chapter 7 where we handle real data sets. In those cases, as we will see, we consider also non

Gaussian data sets with a number of dimensions $D = 13$, and the good performance of our skewness-based method seems to confirm such a theory. The second further direction of development concerns a sort of skewness-based dimension-selection. In this sense, when we are in a non Gaussian high-dimensional data set framework, we can expect that at least some dimensions will influence negatively the symmetry of the clusters. So, the idea is to find a criterion that is able to detect such dimensions. Such an experiment will be shown in Chapter 7.

Chapter 6

Two simulation studies

This Chapter is devoted to the empirical evaluation of our proposal. In this sense, we provide two different simulation studies: the first one is intended to discuss the performance of the SBI (skewness-based index) as a cluster validation index; the second involves a comparison between clustering performances achieved by the R package $Mclust$ and the algorithm we have built for $Sbam$ (*skewness-based allocation method*). In both simulation studies we consider only Gaussian clusters in several different scenarios, where clustering complexity (i.e. overlapping degree) is a priori defined. To be fair, as pointed out before, we have considered K as known for all the competitors in both simulation studies; the development of an index for the choice of the number of clusters is an issue which deserves our attention in the next future.

Before we illustrate the simulation studies, we introduce two definitions respectively related to the overlap degree and to the error in centroid estimation. We use the first one to generate different scenarios in the simulation studies, while the second is exploited to evaluate clustering performance.

In detail, the outline of this Chapter is as follows: in a first section, we discuss a theoretical definition of overlapping cluster degree, on the basis of Maitra and Melnykov (2010); then we describe the R function $MixSim$, see Melnykov, Chen and Maitra (2012), used to simulate different Gaussian mixtures with a given degree of overlap. In this context, we will provide some bivariate graphical examples to illustrate the potentials of this function. Then, we introduce a definition of error regarding centroid estimation, which we use in both simulation studies. So, in the first simulation study we discuss the performance of skewness-based cluster validation index, SBI, throghout different simulation scenarios. Finally, in the second simulation study we provide a comparison between $Mclust$ and $Sbam$ in terms of clustering performances, with some concluding remarks.

6.1 Theoretical and practical tools for simulations

As noticed in Chapter 1, the complexity of a clustering scenario can be function of its overlapping degree, and it is possible to associate this to the misclassification probabilities. In a relevant paper Maitra and Melnykov (2010) give theoretical definitions of overlapping degree for different cases in a Gaussian mixtures framework; we will take it as a reference point. For the empirical part, we will refer to Melnykov, Chen and Maitra (2012), where a Gaussian mixtures generator is discussed to perform simulations in a very general framework, controlling the degree of clusters overlap.

These features are very important since, without such a formalized framework, we could only have provided real data examples, which can not adequately reflect the complexity of possible different clustering scenarios. In the following we report a brief discussion of the definitions, as provided by Dasgupta (1999), Maitra (2009), and Maitra and Melnykov (2010), ending with a short panoramic on the related Gaussian mixtures generator, developed in Melnykov, Chen and Maitra (2012).

6.1.1 A measure of the overlapping degree

The first notion of overlapping degree we consider comes from Dasgupta (1999). Actually, the definition proposed by Dasgupta (1999) deals with separation, rather than overlap, but we gain a measure for overlapping degree by inverting the argument. This definition, as well as all other formulations we will discuss, refers to Gaussian mixtures.

For this purpose, let us consider two D-variate Gaussian clusters, described by component densities $X_i|z_{i1} = 1 \sim N_D(\boldsymbol{\mu}_1; \boldsymbol{\Sigma}_1)$ and $X_i|z_{i2} = 1 \sim N_D(\boldsymbol{\mu}_2; \boldsymbol{\Sigma}_2)$. We say that these are c-separated when the following condition holds

$$S_{1,2} = \|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\| \geqslant c\sqrt{D \cdot \max\left(\lambda_{\max}\left(\boldsymbol{\Sigma}_1\right), \lambda_{\max}\left(\boldsymbol{\Sigma}_2\right)\right)}$$

where $\lambda_{\max}(\boldsymbol{\Sigma})$ is the largest eigenvalue of the positive definite matrix $\boldsymbol{\Sigma}$. According to Dasgupta (1999), we can consider significant to moderate to low overlap between two clusters when $c = 0.5$, $c = 1$, $c = 2$. The meaning of the expression above is intuitive: it represents a measure of the distance between two centroids in terms of the maximum scale magnitude of the two densities, represented by

$\max\left(\lambda_{\max}\left(\boldsymbol{\Sigma}_1\right), \lambda_{\max}\left(\boldsymbol{\Sigma}_2\right)\right)$. The more $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ are far with respect to their maximum scale value, the more the corresponding densities can be considered separated (i.e. with a low degree of overlap). Probably, this may turn to be somewhat clearer if we rewrite the condition as follows:

$$S_{1,2} = \frac{\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|}{\sqrt{D \cdot \max\left(\lambda_{\max}\left(\boldsymbol{\Sigma}_1\right), \lambda_{\max}\left(\boldsymbol{\Sigma}_2\right)\right)}} \geqslant c$$

Considering only $\frac{\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|^2}{D}$, this expression can be regarded as an average distance of $\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|$ in terms of the number of dimension D, while $\max\left(\lambda_{\max}\left(\boldsymbol{\Sigma}_1\right), \lambda_{\max}\left(\boldsymbol{\Sigma}_2\right)\right)$ can be interpreted as a weight defined by the maximum scale magnitude.

Due to the inequality sign, this expression does not measure overlapping degree. For this reason, Maitra (2009) suggests to modify this expression, obtaining the so-called "exact c-separation" between at least two clusters, where for $k, k' = 1, ..., K$, $k \neq k'$ $\|\boldsymbol{\mu}_k - \boldsymbol{\mu}_{k'}\| = c\sqrt{D \cdot \max\left(\lambda_{\max}\left(\boldsymbol{\Sigma}_k\right), \lambda_{\max}\left(\boldsymbol{\Sigma}_{k'}\right)\right)}$. Maitra (2009) stresses that both formulations are carried out regardless of the clusters orientation or mixing proportions, thus providing a partial information on the degree of clustering complexity.

For this purpose, Maitra and Melnykov (2010) formalize the overlapping degree using a different approach. In particular, overlap between two Gaussian clusters is defined as the sum of their misclassification probabilities. To put it formally, let us consider a data set consisting of n D-variate i.i.d. observations $\boldsymbol{x}_i$, $i = 1, ..., n$ coming from the mixture density $f\left(\boldsymbol{x}_i\right) = \sum_{k=1}^{K} \pi_k f_k\left(\boldsymbol{x}_i; \boldsymbol{\theta}_k\right)$, where $f_k\left(\boldsymbol{x}_i; \boldsymbol{\theta}_k\right)$ is a D-variate Gaussian density and $\boldsymbol{\theta}_k = \left(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)$ is the parameters vector, that is

$$f_k\left(\boldsymbol{x}_i | \boldsymbol{\mu}_k; \boldsymbol{\Sigma}_k\right) = (2\pi)^{-\frac{D}{2}} \left|\boldsymbol{\Sigma}_k\right|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}\left(\boldsymbol{x}_i - \boldsymbol{\mu}_k\right)' \boldsymbol{\Sigma}_k^{-1}\left(\boldsymbol{x}_i - \boldsymbol{\mu}_k\right)\right\}$$

Here, the aim is at finding a way to specify parameters $\{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$ such that the realizations $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n$ and the corresponding clusters are characterized by some pre-specified value of the overlap degree, which is intended to represent clustering complexity.

To this end, let us consider two clusters described by $f_1\left(\boldsymbol{x}; \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1\right)$ and $f_2\left(\boldsymbol{x}; \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2\right)$ with mixing proportions π_1 and π_2. We define overlap ω_{12} between these two clusters in terms of the sum of the corresponding misclassification probabilities $\omega_{2|1}$ and $\omega_{1|2}$, that is $\omega_{12} = \omega_{2|1} + \omega_{1|2}$, where

$$\omega_{2|1} = \Pr\left(\pi_1 f_1\left(\boldsymbol{x};\, \boldsymbol{\mu}_1,\, \boldsymbol{\Sigma}_1\right) < \pi_2 f_2\left(\boldsymbol{x};\, \boldsymbol{\mu}_2,\, \boldsymbol{\Sigma}_2\right) \mid z_{i1} = 1\right)$$

which, up to a constant leads to

$$\omega_{2|1} = \Pr\left[\left(\boldsymbol{x}-\boldsymbol{\mu}_2\right)' \boldsymbol{\Sigma}_2^{-1}\left(\boldsymbol{x}-\boldsymbol{\mu}_2\right) - \left(\boldsymbol{x}-\boldsymbol{\mu}_1\right)' \boldsymbol{\Sigma}_1^{-1}\left(\boldsymbol{x}-\boldsymbol{\mu}_1\right) < \log \frac{\left(\pi_2\right)^2 |\boldsymbol{\Sigma}_1|}{\left(\pi_1\right)^2 |\boldsymbol{\Sigma}_2|} \mid z_{i1} = 1\right]$$

and similarly

$$\omega_{1|2} = \Pr\left[\left(\boldsymbol{x}-\boldsymbol{\mu}_1\right)' \boldsymbol{\Sigma}_1^{-1}\left(\boldsymbol{x}-\boldsymbol{\mu}_1\right) - \left(\boldsymbol{x}-\boldsymbol{\mu}_2\right)' \boldsymbol{\Sigma}_2^{-1}\left(\boldsymbol{x}-\boldsymbol{\mu}_2\right) < \log \frac{\left(\pi_1\right)^2 |\boldsymbol{\Sigma}_2|}{\left(\pi_2\right)^2 |\boldsymbol{\Sigma}_1|} \mid z_{i2} = 1\right]$$

So, being $\omega_{12} = \omega_{2|1} + \omega_{1|2}$ this measure is symmetric, that is $\omega_{12} = \omega_{21}$. This measure has an intuitive meaning, which can be expressed as follows: conditionally on belonging to the first cluster, represented by $f_1\left(\boldsymbol{x};\, \boldsymbol{\mu}_1,\, \boldsymbol{\Sigma}_1\right)$, the probability of the event $f_1\left(\boldsymbol{x};\, \boldsymbol{\mu}_1,\, \boldsymbol{\Sigma}_1\right) < f_2\left(\boldsymbol{x};\, \boldsymbol{\mu}_2,\, \boldsymbol{\Sigma}_2\right)$ represents the probability that the unit is erroneously associated to the other cluster.

The link between this index and overlapping degree is direct: the more the densities will be similar to each other (with respect to $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$), the higher will be both $\omega_{2|1}$ and $\omega_{1|2}$ (and then ω_{12}) and the clusters overlapping degree. So, the index proposed by Maitra and Melnykov (2010) has the significant advantage of giving a measure of the overlapping degree based on misclassification probabilities.

Note that by this formulation two different measures can be derived for clusters overlapping degree, an average value (computed on all the possible pairs of clusters) and a maximum one (represented by the couple of clusters which exhibit the maximum value).

To put it formally, let us denote the average value by $\bar{\omega}$ and the maximum overlap by ω^{max}. To compute the average $\bar{\omega}$, we have first to determine the number of selections (without ordering and without repetition) of class 2 from the total number of cluster K, which is the binomial coefficient $\binom{K}{2} = \frac{K!}{2!(K-2)!}$. In this sense, $\bar{\omega}$ refers to the average value of $\omega_{kk'} = \omega_{k|k'} + \omega_{k'|k}\ k \neq k'$, and not to the single terms $\omega_{k|k'}$ and $\omega_{k'|k}$. Then, the average overlap $\bar{\omega}$ can be expressed as follows:

$$\bar{\omega} = \frac{\sum_{k>k'} \omega_{kk'}}{\frac{K!}{2!(K-2)!}} = \frac{\sum_{k>k'} \omega_{kk'}}{\frac{K(K-1)}{2}}$$

On the other hand, the maximum value ω^{max} can be defined as:

$$\omega^{max} = \max_{k} \omega_{kk'}$$

Note that from the expression above is straightforward to provide an upper bound for ω^{max} in terms of $\bar{\omega}$. In fact, the overall overlap degree, that is the sum of all the $\omega_{kk'}$ with $k > k^{'}$, is

$$\sum_{k>k'} \omega_{kk'} = \frac{K!}{2!\,(K-2)!}\bar{\omega} = \frac{K\,(K-1)}{2}\bar{\omega}$$

such that $\omega^{max} \leqslant \sum_{k>k'} \omega_{kk'} = \frac{K!}{2!(K-2)!}\bar{\omega} = \frac{K(K-1)}{2}\bar{\omega}$, and the equality implies that only a couple of clusters determines all the overlap degree of the mixture. On the other hand, we get immediatly the lower bound for ω^{max}, by simply considering that a maximum value can not be lower than the corresponding average value, that is $\omega^{max} \geqslant \bar{\omega}$, and the equality implies that all the $\omega_{kk'}$ with $k > k^{'}$ are forced to be equal, i.e. $\omega_{kk'} = \bar{\omega}$, $k > k^{'} = 1, ..., K$. Note that both the upper and lower bound for ω^{max} would imply a very special case in a simulation framework. In fact, if ω^{max} is setted to its maximum value, $\omega^{max} = \frac{K(K-1)}{2}\bar{\omega}$, only one couple of clusters will be overlapping while all the other will be necessarily strongly separated. On the other hand, if ω^{max} is setted to its minimum value, $\omega^{max} = \bar{\omega}$, all the clusters will be forced to have the same overlapping degree, that is $\omega_{kk'} = \bar{\omega}$, $k > k^{'} = 1, ..., K$. So, a sensible strategy is to choose for ω^{max} an average value between lower and upper bound, that is $\frac{\bar{\omega}+\bar{\omega}\left[\frac{K(K-1)}{2}\right]}{2} = \bar{\omega}\left[\frac{2+K(K-1)}{4}\right]$, to account for at least potentially different cases in a simulation framework. Thus, we adopt such a strategy.

As pointed out by Maitra and Melnykov (2010), analytic calculation of $\omega_{k|k'}$ and $\omega_{kk'}$ is impractical, so they carry out numerical computation using the algorithm developed by Davies (1980). According to the above measure, the R-package $MixSim$ originally developed in Melnykov, Chen and Maitra (2012), allow the user to specify maximum and/or average value of the index to hold in the simulated data, generated from a Gaussian mixture.

Maitra and Melnykov (2010) recommend to select more than one measure at a time, that is to fix jointly maximum and average values. As pointed out by the Authors "a single characteristic is unlikely to comprehensively capture overlap in a realization. For instance, the average overlap may come about from few cluster pairs with substantial overlap, or where many cluster pairs have overlap measures close

to each other (and the average). At the other end, the maximal overlap is driven entirely by one cluster pair (the one with largest overlap, which amount we control). Consequently, we may obtain scenarios with very varying clustering difficulty, yet summarized by the same characteristic". According to this suggestion, we decided to fix both the average and the maximum overlap degree for each simulation scenario (see the last two sections for details).

6.1.2 The R package MixSim. Overlapping Gaussian clusters generator

As noted above, Melnykov et al. (2012) provide a Gaussian mixtures generator with the $MixSim$ package, which allows us to perform simulations in a very general framework, controlling the measures of complexity we have discussed so far. In particular, based on Maitra and Melnykov (2010), the user may specify the desired value of the maximum and/or average complexity.

$MixSim(\cdot)$ and $simdataset(\cdot)$ are two of the main functions in the package we exploited in the simulation study. The first one essentially generates a finite mixture model with Gaussian components for prespecified levels of maximum and/or average complexity. The second, $simdataset(\cdot)$, essentially simulates a datasets of sample size n given the parameters of finite mixture model with Gaussian components, e.g. as returned by the previous function $MixSim(\cdot)$. An important feature of the $simdataset(\cdot)$ function is the possibility to simulate with both outliers and noise, so that an analysis of robustness for a particular clustering method is straightforward to be carried out.

6.1.3 Graphical examples

As noted before, the present analysis deals with Gaussian clusters only. Here, the aim is to give a graphical representation of the potentialities of the $MixSim$ package in generating clustering scenarios with different complexity. To this end, here we discuss some of the arguments of the function $MixSim(\cdot)$ we consider in the simulation study (see below). Clearly, for an obvious lack of space, we limit our choices only to some of the scenarios that will be considered in the simulation study.

In particular, for a given sample size of $n = 300$, we report a couple of bidimensional figures ($D = 2$) for each configuration we obtain combining the values reported below:

- $\bar{\omega} \in \{0.005, 0.025, 0.05\}$
- $K \in \{3, 5\}$
- $\omega^{max} = \bar{\omega} \left[\frac{2+K(K-1)}{4}\right]$

where $\bar{\omega}$ is the average overlap degree as defined above, and K is the number of components. We choose to provide a couple of figures for each configuration to give an idea of the variety in the scenarios, even conditionally to a specific configuration. So, we have 6 different scenarios for a total of 12 figures depicted in Fig. 5.1. The others arguments, instead, have been fixed at the following values (for a discussion of such a choice see the next section):

SETTING VALUES FOR ARGUMENTS IN $MixSim(\cdot)$

- $D = 2$
- *sph* = FALSE
- *hom* = FALSE
- *PiLow* = 0.07
- $int \in (0, 100)$

where D is the number of dimensions, sph, hom control the shape, orientation and volume of the Gaussian mixture components throughout suitable constraints on the component covariance matrices. In particular, $sph = FALSE$ provides hyperellissoidal clusters (not necessarily hyperspherical), while $hom = FALSE$ allows for different covariance matrices across the mixtures components, see also the discussion in Chapter 3. $Pilow$ defines the minimum value for the K components weights π_k, and int represent the side of the D-dimensional hypercube from which we sample the K components mean vectors $\boldsymbol{\mu}_k$. Thus, we generate bidimensional Gaussian mixtures of the type

$$f(\boldsymbol{x}_i) = \sum_k \pi_k f_k(\boldsymbol{x}_i | \boldsymbol{\mu}_k; \boldsymbol{\Sigma}_k)$$

where

- $\min_k \pi_k \geqslant 0.07$, $\sum_k \pi_k = 1$
- $\boldsymbol{\mu}_k \in \{(0, 100) \times (0, 100)\}$
- $K \in \{3, 5\}$

- $z_i = \{z_{ik}\}_{k=1,\ldots,K}$
- $X_i|z_{ik} = 1 \sim MVN_2\left(\boldsymbol{\mu}_k,\, \boldsymbol{\Sigma}_k\right)$
- $\boldsymbol{\Sigma}_k \neq \boldsymbol{\Sigma}_{k'},\, k \neq k'$

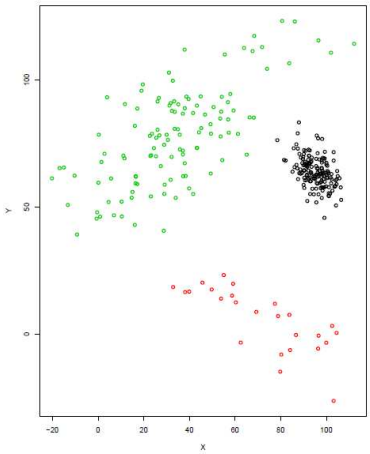
with density $f_k\left(\boldsymbol{x}_i|\boldsymbol{\mu}_k;\, \boldsymbol{\Sigma}_k\right)$ given by

$$f_k\left(\boldsymbol{x}_i|\boldsymbol{\mu}_k;\, \boldsymbol{\Sigma}_k\right) = (2\pi)^{-1}\left|\boldsymbol{\Sigma}_k\right|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}\left(\boldsymbol{x}_i - \boldsymbol{\mu}_k\right)' \boldsymbol{\Sigma}_k^{-1}\left(\boldsymbol{x}_i - \boldsymbol{\mu}_k\right)\right\}$$

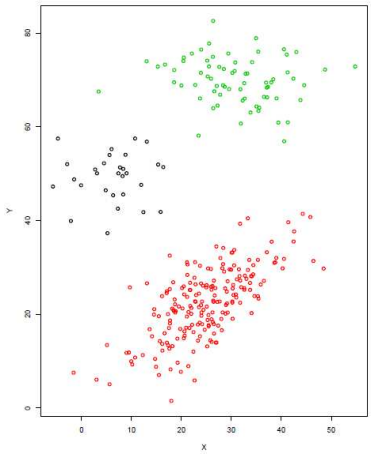
Thus, under this framework, combining the values of $\bar{\omega} \in \{0.005, 0.025, 0.05\}$ and $K \in \{3, 5\}$ we obtain the following 6 couples of figures:

Fig. 6.1: Clustering scenarios for different values of $\bar{\omega}$ and K.

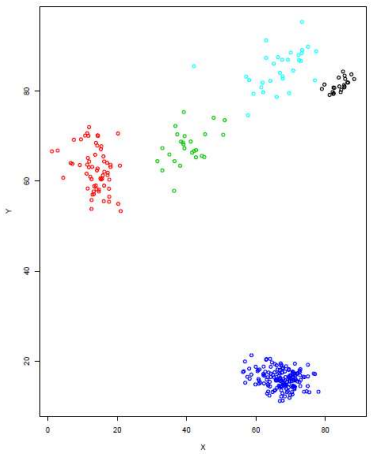
(a) $\bar{\omega} = 0.005$ with $K = 3$



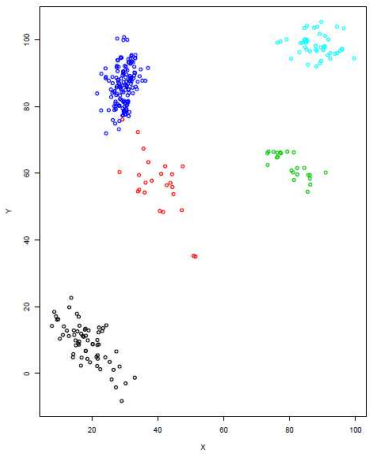
(b) $\bar{\omega} = 0.005$ with $K = 3$



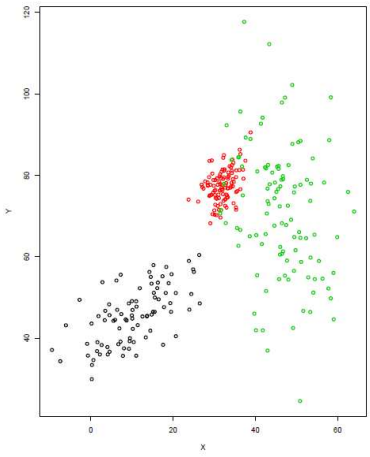
(c) $\bar{\omega} = 0.005$ with $K = 5$



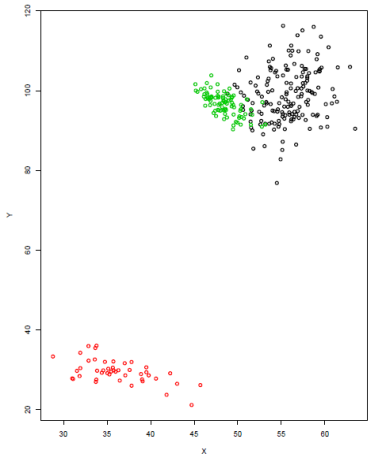
(d) $\bar{\omega} = 0.005$ with $K = 5$



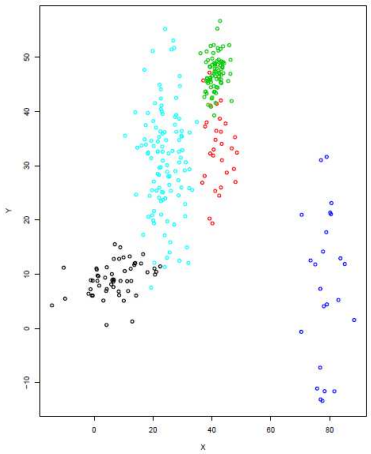
(e) $\bar{\omega} = 0.025$ with $K = 3$



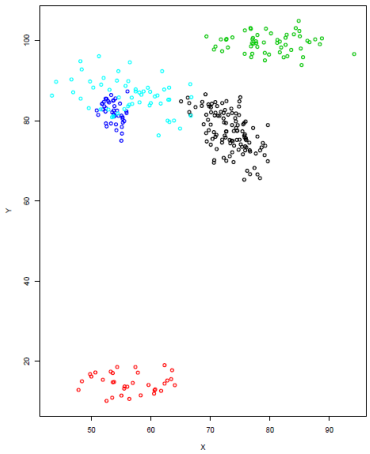
(f) $\bar{\omega} = 0.025$ with $K = 3$



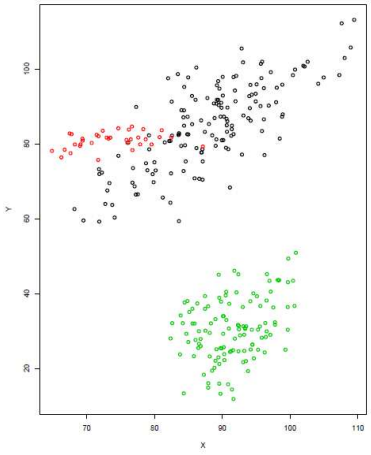
(g) $\bar{\omega} = 0.025$ with $K = 5$



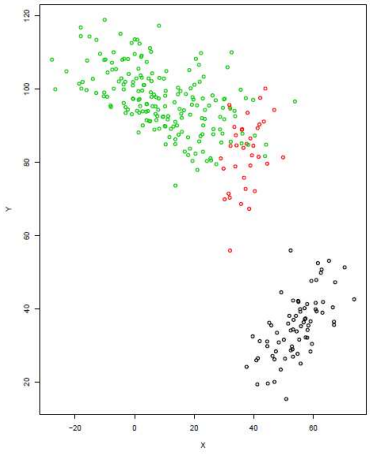
(h) $\bar{\omega} = 0.025$ with $K = 5$



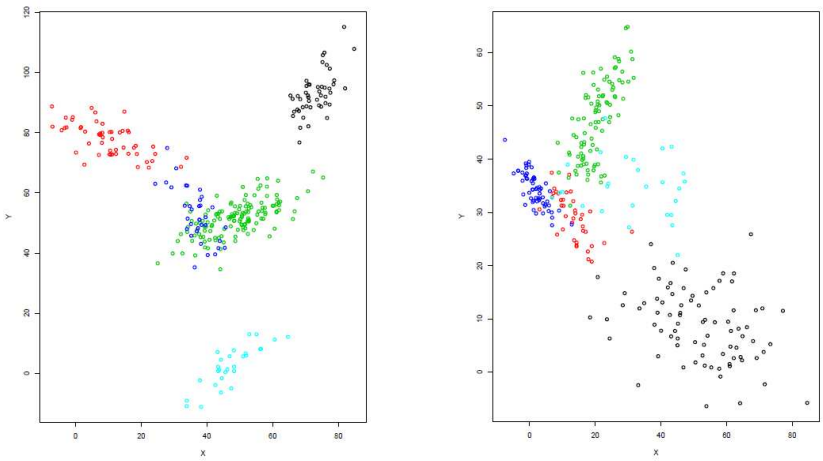
(i) $\bar{\omega} = 0.05$ with $K = 3$



(l) $\bar{\omega} = 0.05$ with $K = 3$



(m) $\bar{\omega} = 0.05$ with $K = 5$ (n) $\bar{\omega} = 0.05$ with $K = 5$



As it can be seen fromfigure 5.1, each parameter configuration $(K, \bar{\omega})$ reflects very different clustering scenarios. Probably, the most noticeable aspect is the increasing order of complexity, which, as expected, is directly related to the increasing values in $\bar{\omega}$. Nevertheless, a relevant variability in position, volume, shape and orientation can be also recognized for each couple belonging to the same configuration. This is an advantage, as it makes the simulation scenarios more sensible, generating quite different clustering while retaining a similar level of complexity (in the sense of overlapping). Another important feature is that the provided clustering scenarios represent each of the four major cases depicted in Chapter 1: well-separated clusters; clusters with low overlap and some *bridge points*; medium overlapping degree with influent *bridge points*; almost totally overlapping clusters.

6.2 Measures of performance

In this section, we introduce the centroid absolute error, adopted in both simulation studies to measure the performance in centroid estimates. Then, we report the expression for the proposed skewness-based index SBI, which we want to test in the first simulation study.

As noted before, we are firstly interested in centroid estimation. Thus, a significant issue is how to assess the goodness of fit for a given estimation method. In this sense, for a D-dimensional case ($d = 1, ..., D$) with K clusters, we define the centroid absolute error, CAE, as the sum of the absolute differences between the *true* and the estimated centroid, respectively $\boldsymbol{\mu}$ and $\bar{\boldsymbol{x}}$, that is

$$CAE = \frac{1}{(K \times D)} \sum_{k=1}^{K} \sum_{d=1}^{D} |\bar{x}_{kd} - \mu_{kd}|$$

where $\bar{x}_{kd}$ is the element corresponding to cluster k and dimension d, and similarly for μ_{kd}. Note that we rescale the above error with the term $\frac{1}{(K \times D)}$ because it increases with both K and D, as we will clarify below.

In such a definition, there are at least two main sources of bias to be considered:

1. different scale magnitudos with respect to different centroids coordinates (relative to the feature dimensions of the observations).

2. label switching.

The first issue is simply solved by rescaling the centroids coordinate for the sample standard deviation of the corresponding dimension, say s_d, that is the square root sign of the d-th element of the diagonal of the sample covariance matrix. Note that if we had ignored it, all of the errors relative to the variables with greater variability would have shown an artificially higher weight on the computation of the overall error CAE. We will denote respectively with $\boldsymbol{\mu}^{\sigma}$ and $\overline{\boldsymbol{X}}^{\sigma}$ the rescaled *true* and estimated centroid matrices, obtained by dividing each element in the d-th column for the quantity s_d. So, with this notation, the CAE will be written as

$$CAE = \frac{1}{(K \times D)} \sum_{k=1}^{K} \sum_{d=1}^{D} |\bar{x}^{\sigma}_{kd} - \mu^{\sigma}_{kd}| = \sum_{k=1}^{K} \sum_{d=1}^{D} \frac{|\bar{x}_{kd} - \mu_{kd}|}{s_d}$$

The second issue refers to the fact that if we can not find an objective ordering in the component labels, any comparison will be meaningless. In fact, without such ordering in the labels we can not understand wheter a centroid estimate actually corresponds to a given cluster, thus vanishing any possibility of comparison. In other words, in absence of ordering criteria any comparison will be a random one.

This topic was just addressed for theoretical purposes in section 2. Here, from a practical point of view, we propose the following naive ordering criterion, explained through an example: let us consider a $D = 2$ dimensional data set, with $K = 3$ clusters, with the (*true*) $K \times D$ centroid matrix

K\D	1	2
1	488.0460	283.4975
2	946.0679	304.0931
3	916.7745	333.4318

and let us suppose the centroid estimates are

K\D	1	2
1	488.4566	286.1510
2	488.3874	280.4233
3	933.8930	316.4190

We induce ordering in labels by ordering rows (i.e. labels) in both the *true* and the estimated $K \times D$ centroids matrices with respect to each dimension. In this case, we would have the following two possible orderings, respectively on the basis of first and second dimension:

True centroids

K\D	1	2
1	488.0460	283.4975
2	916.7745	333.4318
3	946.0679	304.0931

Centroids estimates

K\D	1	2
1	488.3874	280.4233
2	488.4566	286.1510
3	933.8930	316.4190

True centroids

K\D	1	2
1	488.0460	283.4975
2	946.0679	304.0931
3	916.7745	333.4318

Centroids estimates

K\D	1	2
1	488.3874	280.4233
2	488.4566	286.1510
3	933.8930	316.4190

By this way, we will in general have D potentially different orderings. Among these, we choose the one which minimizes the error as defined above. Now, let us denote with O_d the ordering with respect to dimension d, $d = 1, ..., D$, and, consequently, with CAE_{O_d} the CAE depending on the particular O_d. Note that this criterion acts on labels ordering (K rows of the corresponding matrix), while dimensions do not

vary (D columns are fixed). In other words, a particular O_d define a specific permutation of the K rows in the above matrix. Thus, in CAE expression, O_d acts only on the order of the K labels. We denote a specific permutation in the sequence of $k = 1, 2, ..., K$ as k_{O_d}, and consequently the corresponding sum as $\sum_{k=k_{O_d}}$. According to this, we define the centroids absolute error, CAE, as follows

$$CAE = \frac{1}{(K \times D)} \min_{d=1,...,D} CAE_{O_d} = \frac{1}{(K \times D)} \min_{d=1,...,D} \sum_{k=k_{O_d}} \sum_{d=1}^{D} |\bar{x}_{kd}^{\sigma} - \mu_{kd}^{\sigma}|$$

Note that in the example above we would have two different CAE_{O_d}, that is $CAE_{O_1} = 4.692497$ and $CAE_{O_2} = 3.777103$, with only two different orderings implied in this example. Thus, as D increases, the potential bias in CAE induced by *label switching* could be more and more significant. Finally, note that CAE is a sum of $(K \times D)$ terms, such that its magnitude depends on both K and D. So, to prevent artificial influence of these parameters on CAE we rescale it by the factor $\frac{1}{(K \times D)}$.

Now, we report the proposed skewness-based cluster validity index, SBI, which was introduced in Chapter 5. This skewness-based index is based on the skewness function $f_s(\cdot, \cdot)$, which for a D-dimensional point $\boldsymbol{x}_j$ and a cluster centroid $\bar{\boldsymbol{x}}_k$, $k = 1, ..., K$, is defined as

$$f_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}}_k) = \min_{i=1,...,n,\, i \neq j} \parallel (\boldsymbol{x}_j - \bar{\boldsymbol{x}}_k) + (\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k) \parallel$$

Thus, for $i, j \in S_k$, the skewness-based index SBI is defined as the sum over k of the corresponding K functions $f_s(\cdot, \bar{\boldsymbol{x}}_k)$, that is

$$SBI = \sum_{k=1}^{K} \left\{ \sum_{j \in S_k} f_s(\boldsymbol{x}_j, \bar{\boldsymbol{x}}_k) \right\} = \sum_{k=1}^{K} \left\{ \sum_{j \in S_k} \min_{i \in S_k, i \neq j} \parallel (\boldsymbol{x}_j - \bar{\boldsymbol{x}}_k) + (\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k) \parallel \right\}$$

6.3 Simulation studies scenarios

In this section, we describe the clustering scenarios, common to both simulation studies. To this end, we first report the values of reference that we set in our simulations for the function $MixSim(\cdot)$, and then we show the resulting Gaussian mixture distributions, as we did above for the graphical examples.

The relevant values we consider for the function $MixSim(\cdot)$ are the following

SETTING VALUES FOR ARGUMENTS IN $MixSim(\cdot)$
- $\bar{\omega} \in \{0.005, 0.01, 0.025, 0.05\}$
- $\omega^{max} = \bar{\omega} \left[\frac{2+K(K-1)}{4}\right]$,
- $K \in \{3, 5\}$
- $D \in \{3, 6, 12\}$
- *sph* = FALSE
- *hom* = FALSE
- *ecc*=0.9
- *PiLow* = 0.07
- $int \in (0, 100)$

As we saw in the previous graphical examples, the values choosen for $\bar{\omega}$, the average overlapping degree, reflect a wide range of clustering complexity, while the reason for the choice of ω^{max} was given above.

The other values are clearly intended to ensure the widest possible variety of clustering configurations. In fact, we choose a variable number of clusters and dimensions, while letting free the shape of clusters (*sph* = FALSE) and all the (notable) features related to a variable covariance structure (*hom* = FALSE). Moreover, also the maximum value of *ecc*, equal to 0.9, ensures a wide range of eccentricity for each cluster, while *PiLow* = 0.07 allows us to consider also (but not necessarily) unbalanced clustering contexts, where a cluster can have from 0.07 to $1 - [(K-1) \cdot 0.07]$ of the total number of observations n. For instance, in a case of $K = 3$ clusters and $n = 250$ observations, the value *PiLow* = 0.07 determines clusters of sizes across the range [18,214].

Finally, also the interval chosen for the side of the hypercube $int \in (0, 100)$, is intended to extend the possible differences in scale magnitudo between the feature dimensions, with respect to the $MixSim(\cdot)$ default interval, (0, 1).

Note that, if in the graphical examples we could claim a wide variability in scenarios complexity and features, this applies all the more to the current framework. In fact, with respect to the previous examples, here the set of possible value for $\bar{\omega}$ is extended to $\bar{\omega} \in \{0.005, 0.01, 0.025, 0.05\}$, and we consider also a wider range for the number of dimensions D, letting it vary in the set $\{3, 6, 12\}$ (we are not tied anymore to a graphical device). Thus, computing a suitable number of combinations between parameters values, we obtain 16 different clustering configurations, in place of the 6 for the previous graphical examples.

So, the resulting D-dimensional Gaussian mixtures are of the type

$$f(\boldsymbol{x}_i) = \sum_k \pi_k f_k(\boldsymbol{x}_i|\boldsymbol{\mu}_k;\, \boldsymbol{\Sigma}_k)$$

where

- $\min_k \pi_k \geqslant 0.07,\ \sum_k \pi_k = 1$
- $\boldsymbol{\mu}_k \in (0, 100)^D$
- $K \in \{3, 5\}$
- $z_i = \{z_{ik}\}_{k=1,\ldots,K}$
- $X_i|z_{ik} = 1 \sim MVN_D(\boldsymbol{\mu}_k,\, \boldsymbol{\Sigma}_k)$
- $\boldsymbol{\Sigma}_k \neq \boldsymbol{\Sigma}_{k'},\ k \neq k'$

with density $f_k(\boldsymbol{x}_i|\boldsymbol{\mu}_k;\, \boldsymbol{\Sigma}_k)$ given by

$$f_k(\boldsymbol{x}_i|\boldsymbol{\mu}_k;\, \boldsymbol{\Sigma}_k) = (2\pi)^{-\frac{D}{2}} |\boldsymbol{\Sigma}_k|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)' \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)\right\}$$

6.4 Simulation study: performance of SBI

In the following, we present and discuss some simulations to investigate the behaviour of the cluster validity index SBI that we introduced in Chapter 5. As we pointed out before, we are in a Gaussian mixtures framework, so that we expect *true* clusters showing low skewness values, and then low SBI values. Here, the aim is at evaluating performance of SBI as cluster validity index, independently from the clustering performance of the proposed $Sbam$ (it is not considered in this simulation study, while its performances will be evaluated in the second simulation study). Here our interest is mainly in accurately estimating centroids, rather than recovering the "true" partition (even if we provide also a measure to evaluate this point, see below).

To show it, in this simulation study we consider only the clustering performances of $Mclust$ and *K-means*, and we want to test whether the proposed SBI is able to predict the clustering performances of the two methods in terms of centroid absolute error, CAE. In this sense, we essentially analyze the correlation between CAE and SBI. We expect this correlation to be high and positive: the lower is the SBI achieved by a particular partition, the lower will be the error in terms of CAE associated to that partition (provided by a clustering method). In other words, comparing the two methods, we expect that, between $Mclust$ and *K-means*, the one who provides the lower SBI will have a better performance in terms of CAE. So, we account for the correlation between the difference in the CAE's of the corresponding methods $CAE_{Mclust} - CAE_{Kmeans}$ and the difference in the SBI's achieved by

the two methods, $SBI_{Mclust} - SBI_{Kmeans}$. For the sake of notation, we will denote CAE_{Mclust} and CAE_{Kmeans} with CAE_{Mc} and CAE_{Km}, and similarly SBI_{Mclust} and SBI_{Kmeans} with SBI_{Mc} and SBI_{Km}. Therefore, we expect the correlation between $CAE_{Mc} - CAE_{Km}$ and $SBI_{Mc} - SBI_{Km}$ to be high and positive.

To compare the "true" and the estimated partitions, we have used the well-known adjusted *Rand Index*, see Hubert and Arabie (1985). In fact, it is frequently used in the literature, see e.g. Vijendra and Sahoo Laxman (2015), with no relevant drawback. We do the same analysis with respect to the adjusted *Rand Index*, denoted by RI. So, we compute the correlation between $RI_{Mc} - RI_{Km}$ and $SBI_{Mc} - SBI_{Km}$. This time, we expect the correlation to be high and negative, such that lower values for SBI will be associated to higher (better) values for the *Rand Index*.

Finally, we consider also the CAE conditionally to the event $SBI_{Mc} > SBI_t$ and $SBI_{Mc} \leqslant SBI_t$, where SBI_t stands for SBI computed on the *true* partition. Here the aim is two-fold: from the one hand we want to show that the clustering performance of a method (in this case $Mclust$) is related to the reference value of SBI for the *true* partition, such that we expect a better performance if $SBI_{Mc} \leqslant SBI_t$ is true; from the other we want to stress that SBI_t acts really as a reference value, that is in general it is difficult to achieve a value of SBI lower than SBI_t. Thus, we expect that when a method provides a lower SBI, the difference with SBI_t will be negligible and its performance will be good, whereas when a method provides a higher SBI, the difference with SBI_t could be significant and the corresponding performance will get substantially worse. In other words, we want to show that around the SBI_t value, there are many similar partitions which represent good solutions for the clustering problem at hand. To complete the analysis, we report the correlation between the difference on $SBI_{Mc} - SBI_t$ and CAE_{Mc} and the correlation between the difference on $SBI_{Mc} - SBI_t$ and RI_{Mc}. Once again, for the same reason, we expect the first correlation to be high and positive, while the second high and negative.

For each of the 16 different clustering scenarios we generate 300 different Gaussian mixtures, thus considering a wide range of clustering complexity and features. So, on the basis of the above indexes CAE and SBI, for each of the 300 samples from a particular scenario we calculate:

- SBI for the *true* partition, SBI_t
- SBI for the partition returned by the function $Mclust$, SBI_{Mc}
- SBI for the partition returned by the function $Kmeans$, SBI_{Km}
- CAE for the centroid estimates from the function $Mclust$, CAE_{Mc}

- CAE for the centroid estimates from the function $Kmeans$, CAE_{Km}
- RI for the centroid estimates from the function $Mclust$, RI_{Mc}
- RI for the centroid estimates from the function $Kmeans$, RI_{Km}

We summarize these indexes in two separate table, which provide different analysis. In the first table, we report the results of the comparison between $Mclust$ and $K-means$. In the second table ,we report only quantities related to the performance of $Mclust$ conditionally to its values of SBI, that is SBI_{Mc}, when compared to those achieved by true partition, that is SBI_t.
In particular, for each of the 16 scenarios we report the following average quantities in table 6.1

- $ASBI_t$, average SBI_t
- $ASBI_{Mc}$, average SBI_{Mc}
- $ASBI_{Km}$, average SBI_{Km}
- $ACAE_{Mc}$, average CAE_{Mc}
- $ACAE_{Km}$, average CAE_{Km}
- ARI_{Mc}, average adjusted rand index for the partition returned by the function $Mclust$
- ARI_{Km}, average adjusted rand index for the partition returned by the function $Kmeans$
- $\rho^{CAE}_{Mc,Km}$, correlation between $(CAE_{Mc} - CAE_{Km})$ and $(SBI_{Mc} - SBI_{Km})$
- $\rho^{RI}_{Mc,Km}$, correlation between $(RI_{Mc} - RI_{Km})$ and $(SBI_{Mc} - SBI_{Km})$

In table 6.2, for each of the 16 scenarios we report the following average quantities

- $ACAE_{Mc}$ conditionally to the event $SBI_{Mc} > SBI_t$, $ACAE^{Mc}_{Mc>t}$
- $ACAE_{Mc}$ conditionally to the event $SBI_{Mc} \leqslant SBI_t$, $ACAE^{Mc}_{Mc\leqslant t}$
- $ASBI_{Mc}$ conditionally to the event $SBI_{Mc} > SBI_t$, $ASBI^{Mc}_{Mc>t}$
- $ASBI_{Mc}$ conditionally to the event $SBI_{Mc} \leqslant SBI_t$, $ASBI^{Mc}_{Mc\leqslant t}$
- $ASBI_t$ conditionally to the event $SBI_{Mc} > SBI_t$, $ASBI^{t}_{Mc>t}$
- $ASBI_t$ conditionally to the event $SBI_{Mc} \leqslant SBI_t$, $ASBI^{t}_{Mc\leqslant t}$
- $\rho^{CAE}_{Mc,t}$, correlation between $(SBI_{Mc} - SBI_t)$ and CAE_{Mc}
- $\rho^{RI}_{Mc,t}$, correlation between $(SBI_{Mc} - SBI_t)$ and RI_{Mc}

The simulation results are shown in the following two tables

Table 6.1: 16 scenarios depending on $\bar{\omega}$, K, D. A comparison of *Mclust* and *Kmeans* according to SBI index. "Sc" stands for "Scenarios".

Sc.	$\bar{\omega}$	K	D	SBI_t	$ACAE_{Mc}$	ARI_{Mc}	SBI_{Mc}	$ACAE_{Km}$	ARI_{Km}	SBI_{Km}	$\rho^{CAE}_{Mc,Km}$	$\rho^{RI}_{Mc,Km}$
1	0.005	3	3	0.1338	0.0870	0.9615	0.1354	0.1094	0.9000	0.1529	0.5948	-0.6840
2	0.005	3	6	0.8884	0.0997	0.9521	0.8948	0.1066	0.9174	0.9140	0.7641	-0.8223
3	0.005	3	12	3.5761	0.08361	0.9627	3.5813	0.0889	0.9459	3.5917	0.7470	-0.8282
4	0.005	5	3	0.0599	0.1032	0.9345	0.0614	0.0994	0.8972	0.0668	0.5297	-0.5620
5	0.01	3	3	0.1672	0.1125	0.9335	0.1710	0.1269	0.8598	0.1947	0.5281	-0.6581
6	0.01	3	6	1.0404	0.1070	0.9293	1.0495	0.1189	0.8843	1.0768	0.6888	-0.7566
7	0.01	3	12	4.4051	0.1125	0.9066	4.4303	0.1207	0.8711	4.4579	0.7982	-0.8697
8	0.01	5	3	0.0748	0.1161	0.8953	0.0755	0.1269	0.8422	0.0830	0.4537	-0.5321
9	0.025	3	3	0.2268	0.1111	0.8973	0.2234	0.1800	0.7632	0.2710	0.5904	-0.6027
10	0.025	3	6	1.4132	0.1477	0.8418	1.4250	0.1652	0.7721	1.4680	0.7079	-0.7354
11	0.025	3	12	6.1294	0.1408	0.7881	6.2158	0.1587	0.7286	6.2548	0.7381	-0.7980
12	0.025	5	3	0.1071	0.1769	0.7990	0.1058	0.1919	0.7412	0.1185	0.3941	-0.4485
13	0.05	3	3	0.3003	0.1456	0.8091	0.2964	0.2676	0.6041	0.3723	0.5307	-0.5727
14	0.05	3	6	1.9834	0.1946	0.7184	2.0005	0.2278	0.6091	2.0889	0.5726	-0.6761
15	0.05	3	12	8.7847	0.2029	0.5923	8.9738	0.2121	0.5529	8.9848	0.5654	-0.6263
16	0.05	5	3	0.1655	0.2483	0.6785	0.1611	0.2683	0.6303	0.1796	0.3249	-0.3863

Table 6.2: 16 scenarios depending on $\bar{\omega}$, K, D. Performance of *Mclust* conditionally to SBI_{Mc} and SBI_t. "Sc" stands for "Scenarios".

Sc.	$\bar{\omega}$	K	D	$ASBI^{Mc}_{Mc\leqslant t}$	$ASBI^{t}_{Mc\leqslant t}$	$ASBI^{Mc}_{Mc>t}$	$ASBI^{t}_{Mc>t}$	$ACAE_{Mc\leqslant t}$	$ACAE_{Mc>t}$	$\rho^{CAE}_{Mc,t}$	$\rho^{RI}_{Mc,t}$
1	0.005	3	3	0.1352	0.1382	0.1360	0.1228	0.0562	0.1663	0.8074	-0.7430
2	0.005	3	6	0.8753	0.8807	0.9343	0.9040	0.0767	0.1462	0.7938	-0.7991
3	0.005	3	12	3.5998	3.614	3.5648	3.5430	0.0802	0.0866	0.5501	-0.5995
4	0.005	5	3	0.0565	0.0588	0.0690	0.0617	0.0521	0.1819	0.6853	-0.6807
5	0.01	3	3	0.1665	0.1730	0.1779	0.1582	0.0573	0.1977	0.7626	-0.7819
6	0.01	3	6	1.0417	1.0531	1.0593	1.0243	0.0730	0.1503	0.7344	-0.6874
7	0.01	3	12	4.4457	4.4687	4.4180	4.3544	0.0857	0.1338	0.7972	-0.8079
8	0.01	5	3	0.0687	0.0735	0.0860	0.0768	0.0625	0.2000	0.5264	-0.6135
9	0.025	3	3	0.2208	0.2357	0.2283	0.2095	0.0753	0.1808	0.5727	-0.4686
10	0.025	3	6	1.3822	1.4132	1.4826	1.4131	0.0880	0.2279	0.7486	-0.7377
11	0.025	3	12	5.9977	6.0553	6.3122	6.1622	0.1065	0.1560	0.5954	-0.6312
12	0.025	5	3	0.1008	0.1109	0.1142	0.1008	0.1280	0.2591	0.4162	-0.3390
13	0.05	3	3	0.2932	0.3187	0.3021	0.2675	0.0959	0.2341	0.5300	-0.4880
14	0.05	3	6	1.9690	2.0286	2.0395	1.9274	0.1211	0.2856	0.5981	-0.5675
15	0.05	3	12	8.4606	8.5677	9.1734	8.8691	0.1496	0.2236	0.4172	-0.4690
16	0.05	5	3	0.1447	0.1625	0.1917	0.1712	0.2096	0.3200	0.3544	-0.3387

Some remarks are in order. As we pointed out before, the 16 clustering scenarios are quite different in terms of complexity, showing a significant variability in terms of shape, volume, orientation and covariance structure. The main results may be summarized for each table as follows:

TABLE 6.1

SBI increases with increasing average overlapping degree $\bar{\omega}$ for SBI_t, SBI_{Mc}, SBI_{Km}, as the corresponding average values $ASBI_t$, $ASBI_{Mc}$, $ASBI_{Km}$, show. This circumstance has the following possible explanation: with clusters getting more overlapping, symmetry acts like a further constraint in partial opposition to the primary aim to generate the desidered overlapping degree $\bar{\omega}$ (not all the clusters that fit a specific value for $\bar{\omega}$ will also fit a particular low value for skewness), but $MixSim$ function is built to satisfy overlapping degree constraints and not to generate necessarily low skewness clusters (this feature is rather an indirect one related to the Gaussianity assumption). Moreover, for a given n, it is easier to catch an high overlapping value with sparse clusters, which for a given size cover a wider area thus multiplying the possibilities of intersection with other clusters. But we expect that sparse clusters, assuming more variable shapes, would show an higher value of skewness. This seems to be confirmed by the increasing skewness values for the *true* partitions when $\bar{\omega}$ gets higher, whereas the similar increase in the partitions found by $Mclust$ and $Kmeans$ functions is to be regarded as a consequence of this fact. Note that $Mclust$ outperforms $K-means$ in all the 16 scenarios, as we expected, since we are in a Gaussian mixture framework. In this sense, the proposed SBI seems to work well: in all the 16 scenarios, in fact, $ASBI_{Mc} > ASBI_{Km}$, so that according to our index we would have expected such a result. In the same direction, it is quite evident the magnitude and the sign of the correlations considered in the last two columns. Both columns indicate that when one of the two methods provides partitions with lower values of SBI (with respect to the other) it achieves lower CAE and high RI.

Both these features have a relevant consequence: it means that the proposed SBI has good performances as *internal* cluster validation index (that is a measure of clustering performance independent from the *true* partition). In fact, all the above considerations tells us that if a method provides partitions with lower values of SBI it will probably outperform the other methods with higher values of SBI, independently from the corresponding value computed on the *true* partition, that is without knowing it.

Finally, note that, in a sense, the low skewness values achieved by $Kmeans$ when the clusters are well separated (that is for low values of $\bar{\omega}$) are to be considered as

accidental. In fact, such values occur because in case of well separated clusters $kmeans$ solutions approach the *true* partitions, which indeed are composed by low skewness Gaussian ellipsoids. In other words, as pointed out in section 1, basing on Euclidean distance, $kmeans$ can not detect skewness in itself. This is one of the main reasons that leads us to plug in a skewness index in our objective function.

TABLE 6.2

1. *Difference in $ASBI_t$ and $ASBI_{Mc}$.* Another interesting feature of SBI can be realized by comparing the difference $ASBI^{Mc}_{Mc>t} - ASBI^{t}_{Mc>t}$ and $ASBI^{Mc}_{Mc\leqslant t}$-$ASBI^{t}_{Mc\leqslant t}$ across all of the 16 scenarios. It is quite evident, indeed, that the first difference is always considerably higher (its mean is equal to 0.0577, with a maximum of 0.3043) than the second one (its mean is equal to -0.0246375, with a minimum of -0.1071). This means that when $Mclust$ is able to produce a lower skewness partition, this value will be close to the real skewness value of the *true* partition, whereas when $Mclust$ produce an higher skewness partition, this value will be probably much higher with respect to the real skewness value of the *true* partition. Thus, the skewness value of the *true* partition seems to be a sort of *baseline* value, such that it is difficult to produce partition with a lower value of skewness. In other words, a good partition will probably have a low skewness value (see also the following remarks, where it is shown that CEE is directly related to the skewness value in SBI).

2. *Conditional CEE*'s. It seems really significant that in all the configurations we considered, the error in centroid estimates provided by $Mclust$ when $SBI_{Mc} > SBI_t$ is always higher with respect to the same error when $SBI_{Mc} \leqslant SBI_t$, and that's true independently from the clustering scenarios complexity $\bar{\omega}$. Moreover, the same CEE's increases almost monotonically (for given K and D) with the same $\bar{\omega}$. Moreover, the ratio between $ACEE_{Mc>t}$ and $ACEE_{Mc\leqslant t}$ has a non negligible magnitudo: its average value is equal to 225%, with a maximum of 349% (scenario 4). This means that, if a clustering method provides a partition with a value of SBI similar to the SBI computed on *true* partition, we can expect a good clustering performance and viceversa.

3. *$\rho^{CAE}_{Mc,t}$ and $\rho^{RI}_{Mc,t}$.* Also in this case, the results in the correlations between $(SBI_{Mc} - SBI_t)$ and both CAE and RI show an high value (respectively positive and negative) across the different scenarios, although decreasing (in absolute value) with the average overlapping degree $\bar{\omega}$. This last feature is related to the first result analyzed, that is the direct relationship between $\bar{\omega}$ and the skewness of clusters. Higher skewness clusters will probably show more arbitrary shapes, thus weakening the power of our index in evaluating clustering performance. But the fact remains

that values of both correlations are significant. The interpretation of this circumstance is clearly related to the results basing on the conditional CAE's: there is an evident and direct relationship between the skewness values as reported by our index and the error in the centroid estimates, and an inverse relationship with respect to the adjusted rand index. So, the good news is that in a way our index is capable to catch the real $Mclust$ performances in terms of skewness: when the algorithm provide partition with an equal or lower skewness (with respect to the skewness of the *true* partition) we expect a lower CAE. This is an effort to the use of such an index when we need to validate clustering results in case of Gaussian mixtures, as indeed we do in our proposal. In this case, therefore, the proposed SBI index shows good performance as *external* cluster validation index (that is a measure of clustering performance related to quantities computed on the *true* partition).

RESULTS BASING ON BOTH TABLES IN TERMS OF CAE

Finally, it is also quite clear that not only skewness, but also CAE increases with the level of the average overlapping $\bar{\omega}$. Once we have esatblished that CAE is a sensible measure of error, this means that $\bar{\omega}$ index is actually a meaningful one, able to control the real complexity of a clustering scenario (note that our measure of error, CAE, does not have any relationship with the overlap degree). So, the overall results seem encouraging, suggesting that both $\bar{\omega}$ index and our cluster validation index SBI are reliable tools in Gaussian clustering evaluation.

6.5 Simulation study: performance of Sbam and Mclust

In this section we provide a comparison in clustering performance between $Mclust$ and the proposed $SBAM$. This comparison will be done with respect to the indexes introduced above, CAE and SBI, togheter with the well-known *adjusted Rand index*, see Hubert and Arabie (1985). In particular, from this simulation study we expect to show that a skewness-based method works better in scenarios with high overlapping degrees, possibly outperforming $Mclust$. Moreover, we expect that $Sbam$, due to the SBI index, is able to detect clusters with low values of skewness when $Mclust$ do not, and we expect that this occurs more often in presence of high overlapping scenarios, that is when $\bar{\omega} \geqslant 0.025$. To this end, we consider the number of times (over the 400 for each scenario) that $Sbam$ provides a lower SBI when $\bar{\omega}$ increases, and we analyze how this quantity impacts on the centroid estimates error (by CAE index). Thus, we expect that, for each scenario, if this

number increases, that is if $Sbam$ provides lower SBI's frequently, then its performances will get better with respect to those of $Mclust$. As we will see, this fact will be confirmed by our simulations, where in scenarios 7,11,14,15 $Sbam$ outperform $Mclust$ in both centroid estimate and classification.
To analyze these issues, for each of the 16 different clustering scenarios, we generate 400 different Gaussian mixtures, thus considering a wide range of clustering complexity and features. For each of the 400 simulations relating to a particular configuration we account for indexes CAE, SBI and $RAND$ (*adjusted Rand index*), with the following notation:

- SBI for the *true* partition, SBI_t
- SBI for the partition returned by the function $Mclust$, SBI_{Mc}
- SBI for the partition returned by the function $Sbam$, SBI_{Sb}
- CAE for the centroid estimates from the function $Mclust$, CAE_{Mc}
- CAE for the centroid estimates from the function $Sbam$, CAE_{Sb}
- $RAND$ Rand index (adjusted) for the partition returned by the function $Mclust$, $RAND_{Mc}$
- $RAND$ Rand index (adjusted) for the partition returned by the function $Sbam$, $RAND_{Sb}$

Then, we summarize in two tables the indexes by averaging over samples corresponding to a given scenario

TABLE 1

- average SBI_t, $ASBI_t$
- average SBI_{Mc}, $ASBI_{Mc}$
- average SBI_{Sb}, $ASBI_{Sb}$
- average CEE_{Mc}, $ACEE_{Mc}$
- average CEE_{Sb}, $ACEE_{Sb}$
- $ARAND$ Average Rand index (adjusted) for the partition returned by the function $Mclust$, $RAND_{Mc}$
- $ARAND$ Average Rand index (adjusted) for the partition returned by the function $Sbam$, $RAND_{Sb}$

TABLE 2

- occurrences of the event $(CAE_{Mc} > CAE_{Sb})$, written as $\#(E_{Mc} > E_{Sb})$
- occurrences of the event $(CAE_{Mc} < CAE_{Sb})$, written as $\#(E_{Mc} < E_{Sb})$
- occurrences of the event $(CAE_{Mc} = CAE_{Sb})$, written as $\#(E_{Mc} = E_{Sb})$

- occurrences of the event $(SBI_{Mc} > SBI_{Sb})$, written as $\#(S_{Mc} > S_{Sb})$
- occurrences of the event $(SBI_{Mc} = SBI_{Sb})$, written as $\#(S_{Mc} = S_{Sb})$

Below, we report separately the two tables, followed by the respective remarks.

Table 6.3: Performances of $Mclust$ and $Sbam$ in terms of $ACAE$ and $ARAND$

Sc	$\bar{\omega}$	ω^{max}	K	D	$ACAE_{Mc}$	$ACAE_{Sb}$	$ARAND_{Mc}$	$ARAND_{Sb}$
1	0.005	0.01	3	3	0.0920	**0.0910**	0.9568	0.9532
2	0.005	0.01	3	6	0.0908	**0.0890**	0.9654	0.9638
3	0.005	0.01	3	12	0.1305	**0.1290**	0.9116	**0.9139**
4	0.005	0.028	5	3	0.1352	0.1359	0.9157	0.9107
5	0.01	0.02	3	3	0.1031	**0.1028**	0.9373	0.9314
6	0.01	0.02	3	6	0.1095	**0.1053**	0.9289	0.9280
7	0.01	0.02	3	12	0.1356	**0.1335**	0.8776	**0.8817**
8	0.01	0.055	5	3	0.1740	**0.1669**	0.8693	0.8681
9	0.025	0.05	3	3	0.1224	**0.1211**	0.8817	0.8748
10	0.025	0.05	3	6	0.1503	**0.1421**	0.8469	0.8450
11	0.025	0.05	3	12	0.1730	**0.1703**	0.7516	**0.7523**
12	0.025	0.14	5	3	0.2347	**0.2293**	0.7586	0.7582
13	0.05	0.1	3	3	0.1500	**0.1456**	0.8075	0.7962
14	0.05	0.1	3	6	0.1981	**0.1863**	0.7136	**0.7162**
15	0.05	0.1	3	12	0.2269	**0.2221**	0.5689	**0.5779**
16	0.05	0.28	5	3	0.2871	**0.2848**	0.6554	**0.6560**

As we can see from Table 6.3, the $ACAE$ increases for both $Mclust$ and $Sbam$ as the dimensionality D and/or average overlap degree $\bar{\omega}$ increases (the impact of $\bar{\omega}$ is clear, since $\bar{\omega}$ represents an index of clustering complexity). Nonetheless, by looking at the $ACAE_{Mc}$ and $ACAE_{Sb}$ columns, $Sbam$ outperforms $Mclust$ in all of the 16 scenarios (except for the scenario 4, with a very negligible difference of 0.05%), improving center estimation up to a maximum of 6% (scenario 14). With regard to the classification, the overall performances of the two methods are quite similar, with a slight advantage of $Mclust$ (the mean of $ARAND_{Mc}$ and $ARAND_{Sb}$ are, respectively, 0.834 and 0.833). $Mclust$ performs better in 10 cases, improving classification up to a maximum of 1.4% (scenario 13), while $Sbam$ improve classification of 1.6% (scenario 15). Nonetheless Moreover, it can be seen that performances of $Sbam$ in centroid estimation with respect to those of $Mclust$ improve as the average overlap degree $\bar{\omega}$ increases, in the sense that the difference

$ACAE_{Mc} - ACAE_{Sb}$ increases with $\bar{\omega}$. On the other hand, classification gets worse as dimensionality D and/or $\bar{\omega}$ increases for both $Mclust$ and $Sbam$, but at the same time the difference $ARAND_{Sb} - ARAND_{Mc}$ increases with both dimensionality D and $\bar{\omega}$ (for the impact of D, note that, independently from $\bar{\omega}$, each time we have $D = 12$ $Sbam$ outperforms $Mclust$ in classification; to understand the influence of $\bar{\omega}$, note that in the last 4 scenarios, with the highest $\bar{\omega}$, $Sbam$ performs better than $Mclust$ in 3 cases). A possible reason for such a circumstance will be given below, referring to the analysis of the other features summarized in Table 6.4:

Table 6.4: Performances of $Mclust$ and $Sbam$ in terms of $ACAE$ and SBI

Sc	$\bar{\omega}$	ω^{max}	K	D	$\#(E_{Mc} > E_{Sb})$	$\#(E_{Mc} < E_{Sb})$	$\#(E_{Mc} = E_{Sb})$	$\#(S_{Mc} > S_{Sb})$	$\#(S_{Mc} = S_{Sb})$
1	0.005	0.01	3	3	72	70	258	142	258
2	0.005	0.01	3	6	110	117	173	227	173
3	0.005	0.01	3	12	171	160	69	331	69
4	0.005	0.028	5	3	66	77	257	143	257
5	0.01	0.02	3	3	70	81	249	151	249
6	0.01	0.02	3	6	147	136	117	283	117
7	0.01	0.02	3	12	189	177	34	366	34
8	0.01	0.055	5	3	85	61	254	146	254
9	0.025	0.05	3	3	85	101	214	186	214
10	0.025	0.05	3	6	178	149	73	327	73
11	0.025	0.05	3	12	212	182	6	394	6
12	0.025	0.14	5	3	92	57	251	149	251
13	0.05	0.1	3	3	111	98	191	209	191
14	0.05	0.1	3	6	222	138	40	360	40
15	0.05	0.1	3	12	233	165	2	398	2
16	0.05	0.28	5	3	75	67	258	142	258

Here, we observe first the behaviour of the $\#(E_{Mc} > E_{Sb})$ when $\bar{\omega}$ increases: the occurrence of a better performance of $Sbam$ tends to increase with $\bar{\omega}$, as we expected (the difference $\#(E_{Mc} > E_{Sb}) - \#(E_{Mc} < E_{Sb})$ tends to increase with $\bar{\omega}$, as one could see from scenarios 12-15, which represent the highest differences for all the combinations of K and D considered). But, probably, the most remarkable feature is the behaviour of $\#(S_{Mc} = S_{Sb})$: note that the event $(S_{Mc} = S_{Sb})$ implies that $Sbam$ could not find a better partition (that is less skewed) with respect to that

provided by $Mclust$, so returning the same partition. Looking at the table, we note that this evenience occurs evidently quite often when $Mclust$ provides a really good solution (with a smaller CAE and a higher $ARAND$, see also the previous table). Moreover, this occurrence is clearly and strongly negative correlated to the average overlap degree $\bar{\omega}$ and the dimensionality D: across scenarios with a fixed and given $\bar{\omega}$, $\#(S_{Mc} = S_{Sb})$ goes always monotonically from 258 to 69 (scenarios 1-4), from 258 to 34 (scenarios 5-8), from 251 to 6 (scenarios 9-12), from 258 to 2 (scenarios 13-16). This means that not only the dimensionality D but also the overlap degree $\bar{\omega}$ impacts quite strongly and negatively on the occurrence of the event $(S_{Mc} = S_{Sb})$, which is associated with better performance of $Mclust$. But we know that clustering performance of $Mclust$ get worse just when D and/or $\bar{\omega}$ increases. In other words, the proposed $Sbam$ is able to detect a worsening in $Mclust$ clustering performances, and if it is the case it provides an alternative and less skewed partition with a smaller error in centroid estimates. In this sense, $Sbam$ acts like a skewness-based corrective mechanism of the solutions provided by $Mclust$. But two issue remain:

1. why in general clustering performances get worse when D increases?

2. why $Sbam$ performance in terms of both centroid estimate and classification are less influenced by $\bar{\omega}$?

We discuss these issues in turn.

1. The first question is related to the multidimensional sparsity of data (see section relative to the definition of overlap degree). When D increases the observations become more and more sparse in the space of features, and the discriminant power of the metric distance decreases, so that both centroid estimates and classification get worse. As pointed out by Xiao and Yu (2012): "the distant measure become increasingly meaningless as the number of variables increases in the data set". This is in general the *subspace clustering* framework, a family of techniques implemented for high-dimenaional data. Essentially, *subspace clustering* attempts to find subsets of dimensions for different clusters (*hard* version) or different weights applied to dimensions for each cluster (*soft* version). This is a context where it would be interesting to evaluate clustering performance of $Sbam$. For instance, one could find a suitable subspace for a cluster, which minimizes its skewness, or the set of subspaces which minimizes the overall SBI. So, the *subspace clustering* framework could represent a further direction of work for the proposed method.

2. The second issue is, actually, related to the starting point of this work: as $\bar{\omega}$ increases, the corresponding overlapping areas extend across the space (and this is more the case as also D increases, due to the multidimensional sparsity of data). But, as we pointed out, the metric distance-based criterion underlying $Mclust$ is sistematically unable to detect such overlapping areas, allocating all the observations

in the corresponding intersection to one cluster only, thus deteriorating the classification (the *true* partition has actually overlapping clusters), while $Sbam$ is at least partially able to catch such regions. This would also explain the better performance of $Sbam$ when compared to $Mclust$ in both centroid estimates and classification for high value of average overlap degree $\bar{\omega}$.

Chapter 7

Performance analysis on real data

In this Chapter we discuss the application of the proposed skewness-based method to real data. In a sense, if one aim at testing a particular method, real data examples may be somewhat less informative when compared to the wide variability of a simulation study. A single real data set, although highly representative of some interesting features, is still a single one. This is even more the case in a parameric framework, where we are interested to study the performance under given distributional assumptions. Nevertheless, analysis of real data examples can be instructive in a further sense. For instance, real data examples may help analyze the behaviour of the proposed clustering method under a misspecified model, when clusters may not be elliptical. We should remark that, in each analyzed dataset, the clustering structure is completely and perfectly known.

This Chapter is organised as follows. In the first part we briefly introduce the real data sets considered, focusing only on basic features (year of reference and relative sources). Further, in the following subsections we provide a more detailed description of the datasets, and the performance achieved by the $Mclust$ and $Sbam$, in terms of CAE, SBI and *adjusted Rand Index*. Then, we propose a modified but provisional version of the $Sbam$ which we test (only as experiment) on the same real data examples. Finally, we conclude with some remarks about the performance of the two clustering methods in the real data context, and a proposal for further direction of development.

7.1 Application to real data

In this section, we introduce five real data examples, which will be considered to study the performance of $Mclust$ and $Sbam$ methods. As noted above, we can not

assume any known parametric distribution for the clusters in the five data sets. So, a major aim is to test clustering performance when we move away from the hypothesis of Gaussian mixtures. The five real data examples at hand are, in order of appearence, *Iris* by Anderson (1935), *Crabs* from Campbell and Mahon (1974), *Wine* by Aeberhard, Coomans and de Vel (1992), *Seeds* by Charytanowicz et al. (2010), and finally *Ecoli* by Horton & Nakai (1996). We describe them in the following subsections, and report a comparison in terms of performance for the $Mclust$ and the $Sbam$, while a discussion of the corresponding results is provided in the concluding section. All of these data sets can be found at the free access *UCI machine learning repository*, except for $Crabs$, which is embedded in the R package $MASS$.

7.1.1 Iris Data

Probably, $Iris$ is one of the best known dataset in the history of classification. Originally, these data were collected by Anderson (1935), and analyzed in a classic paper by Fisher (1936) on measurements in taxonomic problems.

The data entails 150 observations divided in 3 clusters of the same size (50 each). Each class refers to a different type of the iris plant, *Setosa*, *Versicolour* and *Virginica*. Below is a short list of the essential features of the dataset.

Number of classes: 3 (three species of *Iris* plant)
Classes: *Setosa*, *Versicolour* and *Virginica*
Number of Attributes: 4
Type of attributes: numeric
Attributes:
1. sepal length in cm
2. sepal width in cm
3. petal length in cm
4. petal width in cm

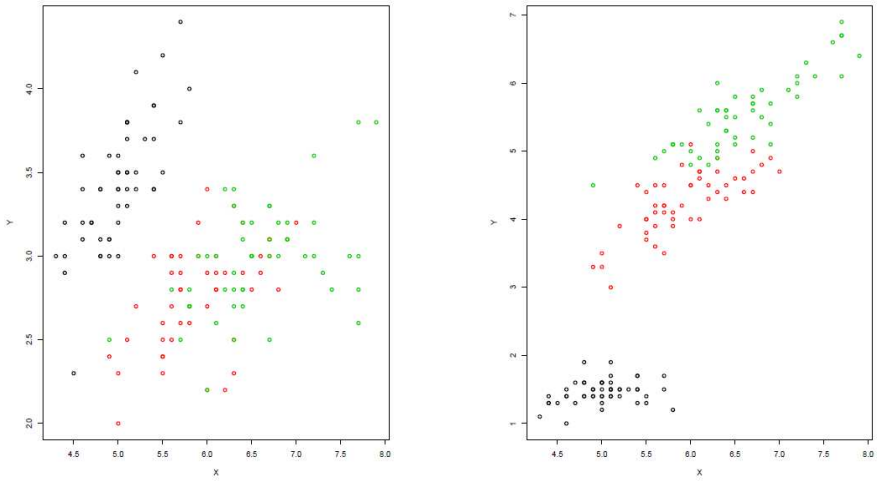
Number of Instances: 150 (50 in each of three clusters)

Note that each cluster has the same size of 50, so that data show an exactly balanced number of observations among clusters. Nonetheless, the main feature of these data is probably the partial overlapping which can be found in two of the

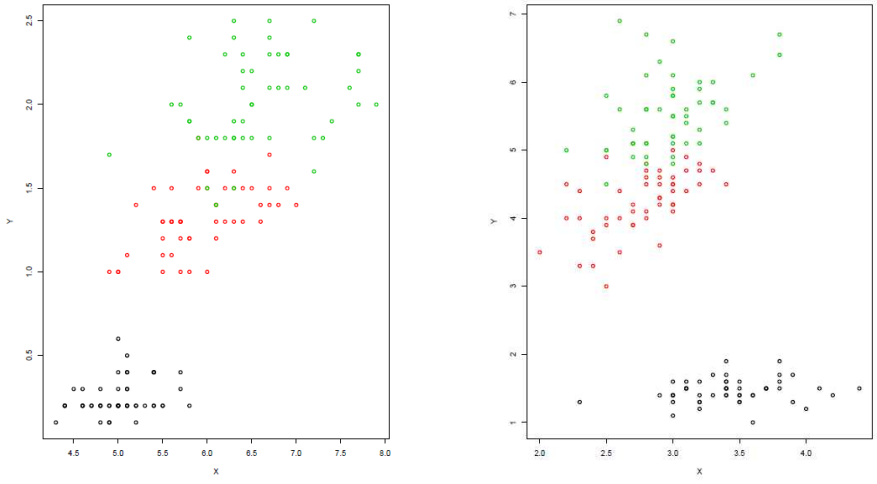
three clusters. To illustrate this issue, we report the following figures, representing all the bidimensional profiles of the multivariate distribution:

Fig. 7.1. Bidimensional profiles of data set *Iris* (1935), where X axis and Y axis are respectively: (a) sepal length, sepal width (b) sepal length, petal length (c) sepal length, petal width (d) sepal width, petal length (e) sepal width, petal width (f) petal length, petal width.

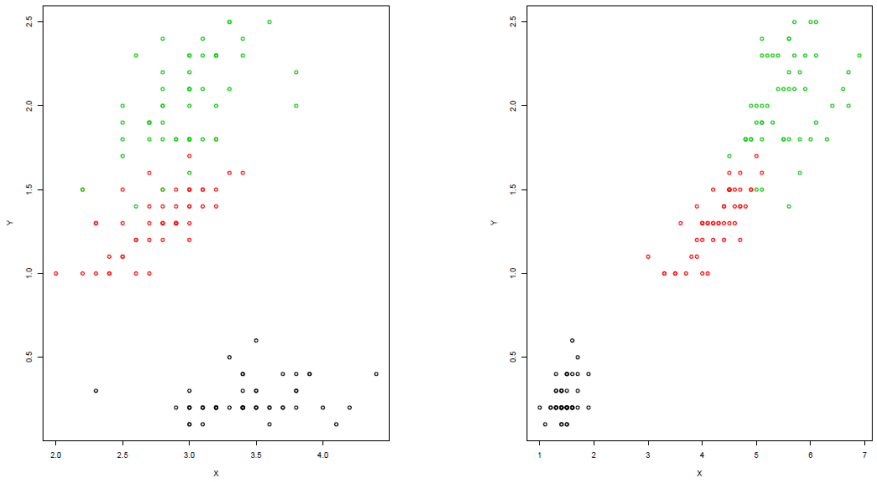
(a) (b)



(c) (d)



(e) (f)



It is easy to recognize that red and green clusters are not linearly separable (roughly speaking, there is no line dividing them) in any of the 6 bivariate profiles, while black cluster is clearly well separated from the others. So, in a sense, this case represents an example of overlapping clusters considered in this work. Below, we report the clustering performance achieved by $Mclust$ and $Sbam$ for this dataset in terms of CAE, SBI and RI.

K	D	SBI_t	SBI_{Mc}	SBI_{Sb}	CAE_{Mc}	CAE_{Sb}	RI_{Mc}	RI_{Sb}
3	4	1.0818	1.0097	1.0097	0.0258	0.0258	0.9039	0.9039

7.1.2 Crabs Data

This data set refers to morphological measurements on Leptograpsus Crabs, discussed in Campbell and Mahon (1974), who analyzed the variation in two species of rock crabs (*genus Leptograpsus variegatus*) divided by sex. In this case we have 4 clusters, each one corresponding to a specific combination of species and sex. We have 200 observations in total, divided in 4 classes, each one with 50 units. Also in this case data is exactly balanced with respect to the number of observations in each cluster. Below, we report the essential features of these data:

Number of classes: 4 (two species for each sex of crabs)
Classes: Blue (male), Blue (female),

Orange (male), Orange (female)
Number of Attributes: 5
Type of attributes: numeric
Attributes:
1. FL frontal lobe size (mm).
2. RW rear width (mm).
3. CL carapace length (mm).
4. CW carapace width (mm).
5. BD body depth (mm).
Number of Instances: 200 (50 in each of 4 clusters)

Below we report the clustering performance achieved by $Mclust$ and $Sbam$ for this dataset in terms of CAE, SBI and RI.

K	D	SBI_t	SBI_{Mc}	SBI_{Sb}	CAE_{Mc}	CAE_{Sb}	RI_{Mc}	RI_{Sb}
4	5	13.5879	12.8538	12.8538	0.3112	0.3112	0.3165	0.3165

7.1.3 Wine Data

This data set comes from a chemical analysis of wines from three different cultivars from the same region in Italy. The data consist of 13 constituents. So, we have 3 clusters, each one representing a different cultivar. In this case, we have an unbalanced data set with 59, 71 and 48 in each cluster, for a total of 178 observations. Actually, these data were discussed by Aeberhard, Coomans and de Vel (1992). They are on a substantially higher dimension when compared to the previous ones, with 13 attributes (chemical features measured on each observation). We provide a short list of the data features:

Number of classes: 3 (three types of wine)
Classes: 3
Number of Attributes: 13
Type of attributes: numeric
Attributes:
1. Alcohol
2. Malic acid
3. Ash

4. Alcalinity of ash
5. Magnesium
6. Total phenols
7. Flavanoids
8. Nonflavanoid phenols
9. Proanthocyanins
10. Color intensity
11. Hue
12. OD280/OD315 of diluted wines
13. Proline

Number of Instances: 178 (59, 71, 48)

Below we report the clustering performance achieved by $Mclust$ and $Sbam$ for this dataset in terms of CAE, SBI and RI.

K	D	SBI_t	SBI_{Mc}	SBI_{Sb}	CAE_{Mc}	CAE_{Sb}	RI_{Mc}	RI_{Sb}
3	13	4437.492	2754.034	2.754.034	0.05485	0.05485	0.8804	0.8804

7.1.4 Seeds Data

The dataset $Seeds$ results from an analysis of three different varieties of wheat: Kama, Rosa and Canadian, see Charytanowicz et al. (2010). These were randomly selected for the experiment. Internal kernel structure was detected in a high quality visualization using a soft *X-ray* technique (the images were recorded on 13x18 cm *X-ray* KODAK plates). The data set was constructed by measuring seven geometric parameters of wheat kernels, see Charytanowicz et al. (2010).

Once again, the data set is balanced, with 70 observations for each of the three clusters (three different types of wheat). Below we summarize some essential information on the dataset

Number of classes: 3 (3 varieties of wheat)
Classes: varieties of wheat (Kama, Rosa, Canadian)
Number of Attributes: 7
Type of attributes: numeric
Attributes:
1. area A

2. perimeter P
3. compactness C = 4*pi*A/P ^ 2
4. length of kernel
5. width of kernel
6. asymmetry coefficient
7. length of kernel groove

Number of Instances: 210, equally divided in 3 clusters of 70 observations each

Below we report the performance achieved by $Mclust$ and $Sbam$ for this dataset in terms of CAE, SBI and RI.

K	D	SBI_t	SBI_{Mc}	SBI_{Sb}	CAE_{Mc}	CAE_{Sb}	RI_{Mc}	RI_{Sb}
3	7	3.0068	2.6442	2.6442	0.1613	0.1613	0.6299	0.6299

7.1.5 Ecoli Data

The last dataset we consider is *Ecoli*, discussed in Horton & Nakai (1996). These data come from a study on classification systems for predicting the cellular localization sites of proteins. The aim is to properly classify 336 proteins on the basis of 8 different cellular localization sites (see below for details). Thus, the total number of observations is 336, while the numer of clusters is 8.

There are three main reasons for our interest in this dataset. The first one is that it shows a relatively high numer of clusters. The second one concerns the distribution of observations in the clusters, which is quite unbalanced, with 143, 77, 52, 35, 20, 5, 2, 2 units. The third and last one refers to the presence of binary variables (the fourth and fifth), whereas all the previous datasets have continuos variables only. This last feature, in particular, can be regarded as a further departure from the assumptions of a Gaussian mixture. Below we provide some basic information on this dataset

Number of classes: 8
Classes:
cp (cytoplasm)
im (inner membrane without signal sequence)
pp (perisplasm)

imU (inner membrane, uncleavable signal sequence)
om (outer membrane)
omL (outer membrane lipoprotein)
imL (inner membrane lipoprotein)
imS (inner membrane, cleavable signal sequence)
Number of Attributes: 8
Type of attributes: binary and numeric
Attributes:
1. Sequence Name: Accession number for the SWISS-PROT database (ignored in clustering implementation)
2. mcg: McGeoch's method for signal sequence recognition.
3. gvh: von Heijne's method for signal sequence recognition.
4. lip: von Heijne's Signal Peptidase II consensus sequence score.
Binary attribute.
5. chg: Presence of charge on N-terminus of predicted lipoproteins.
Binary attribute.
6. aac: score of discriminant analysis of the amino acid content of outer membrane and periplasmic proteins.
7. alm1: score of the ALOM membrane spanning region prediction program.
8. alm2: score of ALOM program after excluding putative cleavable signal regions from the sequence.
Number of Instances: 336, distributed among clusters with size 143, 77, 52, 35, 20, 5, 2, 2.

Below we report the clustering performance achieved by $Mclust$ and $Sbam$ for this dataset in terms of CAE, SBI and RI.

K	D	SBI_t	SBI_{Mc}	SBI_{Sb}	CAE_{Mc}	CAE_{Sb}	RI_{Mc}	RI_{Sb}
8	7	0.1191	0.1064	0.0974	0.7616	0.7374	0.4305	0.4356

7.2 A proposal for a further development of Sbam

As seen in the previous results, in these real data examples $Sbam$ improves the performance of $Mclust$ only in the case of *Ecoli*. Starting from this result, we investigate the possible reasons for such a circumstance. The main we consider is the lack of symmetry in one or more features of a dataset. In fact, if some features are substantially skewed, it is possible that $Sbam$ could not improve the partition provided by $Mclust$. From this point of view, we regard at such features as a source of noise, which artificially increase the overall skewness computed by SBI. Thus, the idea is to find such dimensions and exclude them from the dataset. We develop the following ad-hoc procedure:

- standardize the dataset
- compute SBI by removing a single dimension d (a column) from the dataset
- repeat for $d = 1, ..., D$
- remove permanently the dimension d^*, that, once excluded, leads to a case where SBI is minimum

To put it formally, let us consider SBI_{-d}, that is the SBI computed by removing the d-th dimension

$$SBI_{-d} = \frac{1}{K \cdot (D-1)} \sum_{k=1}^{K} \left\{ \sum_{j \in S_k} f_s \left(\boldsymbol{x}_j^{(-d)}, \bar{\boldsymbol{x}}_k^{(-d)} \right) \right\} =$$

$$= \frac{1}{K \cdot (D-1)} \sum_{k=1}^{K} \left\{ \sum_{j \in S_k} \min_{i \in S_k, i \neq j} \parallel (\boldsymbol{x}_j^{(-d)} - \bar{\boldsymbol{x}}_k^{(-d)}) + (\boldsymbol{x}_i^{(-d)} - \bar{\boldsymbol{x}}_k^{(-d)}) \parallel \right\}$$

where $\boldsymbol{x}_j^{(-d)}$, $\boldsymbol{x}_i^{(-d)}$ and $\bar{\boldsymbol{x}}_k^{(-d)}$ are, respectively, the $\boldsymbol{x}_j$, $\boldsymbol{x}_i$ and $\bar{\boldsymbol{x}}_k$ when the d-th dimension is not included in the dataset. Then, we compute the corresponding D quantities and remove permanently the dimension d^* that, once excluded, leads to the minimum SBI, that is

$$d^* = \arg\min_d SBI_{-d}$$

Note that we can compare the D different SBI_{-d}, since the denominator $\frac{1}{K \times (D-1)}$ is the same for all of them, and the scale influence is avoided by standardizing the

dataset. Moreover, this procedure can be iterated, if the noise features are supposed to be more than one.

In the following tables we report the performance of $Sbam$ and $Mclust$ on the same real data examples, where the first summarizes the previous results, while the second one reports the performance achieved by implementing the SBI_{-d} function. Some remarks can be found in the next section.

Table7.1: Performance of *Mclust* and *Sbam* in real data examples without implementing SBI_{-d}

$Data$	SBI_t	SBI_{Mc}	SBI_{Sb}	CAE_{Mc}	CAE_{Sb}	RI_{Mc}	RI_{Sb}
Iris	1.0818	1.0097	1.0097	0.0258	0.0258	0.9039	0.9039
Crabs	13.5879	12.8538	12.8538	0.3112	0.3112	0.3165	0.3165
Wine	4437.49	2754.03	2754.03	0.0549	0.0549	0.8804	0.8804
Seeds	3.0068	2.6442	2.6442	0.1613	0.1613	0.6299	0.6299
Ecoli	0.1191	0.1064	0.0974	0.7616	0.7374	0.4305	0.4356

Table 7.2: Performance of *Mclust* and *Sbam* in real data examples implementing SBI_{-d}

$Data$	SBI_t	SBI_{Mc}	SBI_{Sb}	CAE_{Mc}	CAE_{Sb}	RI_{Mc}	RI_{Sb}
Iris	0.8314	0.7103	0.7103	0.0199	0.0199	0.9410	0.9410
Crabs	12.9634	10.2501	10.2501	0.3833	0.3833	0.4378	0.4378
Wine	4318.02	4385.07	4385.07	0.0165	0.0165	0.9667	0.9667
Seeds	0.7052	0.7585	0.6510	0.0566	0.1191	0.8412	0.6544
Ecoli	0.1390	0.1057	0.0895	0.5226	0.5346	0.5291	0.5401

7.3 Concluding remarks

In the considered real data examples, the proposed $Sbam$ achieves the same clustering performance provided by $Mclust$, except for *Ecoli*, where *Sbam* outperforms $Mclust$. Nonetheless, the overall clustering performances seems to be not completely satisfactory, particularly for *Crabs*, *Seeds* and *Ecoli*. So, we decided to test the SBI_{-d} function in this context.

The result obtained implementing SBI_{-d} seems to be encouraging, improving substantially the performance for both *Mclust* and *Sbam*. Nonetheless, the results are to be considered carefully. There are, in fact, many open questions regarding the use of the SBI_{-d} function.

First of all, we do not know wheter *SBI* and *CAE* computed before and after the implementation of SBI_{-d} are comparable. In fact, the impact of SBI_{-d} on these measures is not clear. For instance, by looking at the tables, we note that we get an

apparently less skewed partition, but this is not the case for *Ecoli*. The same occurs with *CAE*, which is lower in all the cases except for *Crabs*. The problem lies at the heart of the SBI function: is it possible to compare measures for different (K, D) pairs? Here, we do not address this question, letting it to a further investigation.

Another relevant issue is how to choose the number of dimensions for a dataset to be removed. This is clearly not a trivial matter, since only if we solve it, we get a criterion to determine the proper number of iterations for SBI_{-d}, that is the proper number of "noise" dimensions (those dimensions that actually increase the overall skewness of a partition). For example, we choose only one iteration for the above examples, except for *Wine* and *Seeds* where we choose 2 iterations because removing only one variable we would get practically the same partition of the complete dataset. In these cases we have a complete knowledge of the datasets, but this is not the case in general, so it would be important to find a criterion to determine the proper number of iterations for SBI_{-d}.

Nonetheless, we note that the *adjusted rand index* are comparable, since labels are not altereted in any way by using the SBI_{-d} procedure. So, we can say that SBI_{-d} actually is able to improve clustering performance in the real examples considered. Note that SBI_{-d} seems to work well also when used before $Mclust$. In fact, by looking at the two tables we see that SBI_{-d} improves the clustering performances of $Mclust$ in all the datasets we have considered. In particular, the improvement carried out in the case of *Seeds* is substantial: thanks to the implementation of SBI_{-d}, $Mclust$ provides a partition with an *adjusted Rand Index* of 0.8412 (without SBI_{-d} it achieved a value of 0.6299) outperforming the proposed $Sbam$.

According to these features of the SBI_{-d}, we think that $Sbam$ can be further developed in the *subspace clustering* (see Chapter 5). This family of clustering methods works in a high-dimensional context, and discriminating between dimensions on the basis of skewness could be an interesting feature. For instance, one could try to find a proper subset of dimensions for each cluster on the basis of the SBI_{-d} function. This could represent a further direction of development for the $Sbam$.

Chapter 8

Concluding remarks

In this work, we have dealt with the issue of overlapping elliptical clusters. We have choosen such a framework, since we believe that clustering usefulness is someway inversely related to its complexity: there are several efficient techniques to handle non overlapping clusters, while there are a few methods that are designed to deal with high degrees of overlap. As we have seen, $Mclust$ may have problems in case of highly overlapping clusters, even in its "natural habitat", when Gaussian mixtures are considered. This was the starting point for our proposal of a skewness-based method, developed to handle such complex cases.

In Chapter 5 we have introduced the main features of $Sbam$, stressing the differences with respect to other skewness-based approaches, introduced in Chapter 4. In particular, we have shown the flexibility of the objective function, as well as its ability to exploit both sources of information, that is distance from the centroid and contribution to skewness.

In Chapter 6, we have studied the clustering performance of the proposed method in two ways: a simulation study on the SBI as a cluster validity index in the case of Gaussian clusters, and a simulation study where the performance of $Mclust$ and $Sbam$ are compared. We achieved encouraging results in both cases. In particular, in the first study, SBI showed good predicting properties, not only as *internal* validity index, but also as an *external* one. In the second simulation study, the proposed $Sbam$ performs pretty well when we consider the centroid estimates error, outperforming $Mclust$ in 15 of the 16 scenarios considered, while $Mclust$ works a little better in allocating units to clusters, even though the difference is quite negligible and $Sbam$ outperforms $Mclust$ not only in high-dimensional cases ($D = 12$) but also for high levels of overlapping degree (it occurs in 3 scenarios of 4 when $\bar{\omega} = 0.05$) .

Nonetheless, there are some relevant drawbacks related to the proposed method, so that there is still extensive room for improvement. We discuss some of them in the following.

The first evident drawback of the proposed method is the absence of a criterion to choose the number of clusters K. Surely, this is not a trivial matter, common to many other clustering approaches. This question is hard to solve in principle, since it depends on what we mean by cluster (see Chapter 1). Nonetheless, in a parametric framework, it is possible to build sensible criteria to choose the number of clusters, like the BIC (Schwartz, 1978) implemented in $Mclust$. In the case of $Sbam$, we could try to gain an estimate of K minimizing the SBI value obtained when K varies, but there is a warning in such a strategy. As we pointed out in Chapter 7, it is not clear whether it is possible to compare different SBI obtained under different pairs of (K,D). So, before trying this way, it is necessary to handle this issue.

A second issue is related to the robustness: the proposed methodology has been tested only on Gaussian mixtures, in absence of *extreme values* and/or *leverage points*. This is a critical point, since such cases occur quite often in real data. Morever, in our proposal there is no reference to the concept of noise, another relevant issue to deal with.

Clearly, each of the drawbacks just mentioned can be regarded as a further direction of development for the proposed method. It would be interesting to handle these issues starting from the concept of skewness. In this sense, one could try to define outliers and/or noise data observing their impact on clusters skewness. We did something similar when proposing SBI_{-d} function in real data example analysis. In fact, in such a frameworkthe SBI_{-d} is used to find and remove features which alter the overall skewness of the estimated partition. From this point of view, we indirectly treated those features as noise dimensions. As pointed out in Chapter 7, the same SBI_{-d} function could be used to approach *subspace clustering* strategies, which essentially are built to discriminate between dimensions in a high-dimensional context. For instance, on the basis of skewness, one could try to find a proper subset of dimensions for each cluster (e.g via a suitable modification of SBI_{-d}). So, we could find, also in this framework, a further direction of development for the proposed technique.

Beyond these general issues, there are drawbacks related to more specific features of the functions involved in our method.

For instance, the SBI lacks, when used as a cluster validity index, of an upper bound, due to the presence of the $\min(\cdot)$ in its expression, which is not *a priori* determined, and this is not a desirable property of an index.

On the other hand, also the $Sbam$ function could be improved in several ways. For instance, it could be extended to consider different definitions of skewness, so

that it would be able to detect further types of symmetry in the shapes of clusters (as well as other types of distributions).

Surely, there is room for improvement also for the initialization strategy. We tried only experimentally a random choice for the initial centroids, but we did not investigate sistematically this point. Once again, there is a huge literature on this topic, and consequently many methods to choose among, see Chapter 2 for a brief discussion. Nonetheless, given the sensitivity of clustering results to the initialization values, this is surely a relevant issue that needs to be sistematically analyzed. However, with respect to other skewness-based approaches, the proposed method is intended to be less sensitive to initialization values. In fact, as pointed out in Chapter 5, the proposed method does not need any initial partition to start (in this sense, an initial partition surely represent a further source of impact on clustering results).

Finally, we may call also for a systematic investigation on the convergence properties of the algorithm, which is surely a feature of interest, but beyond the scope of this work.

Despite of all these drawbacks, we believe that the encouraging results achieved by the proposed method well justify further work along one or more of these directions.

Bibliography

[1] Aeberhard, S., Coomans, D., & de Vel, O. (1992). The classification performance of RDA. Dept. of Computer Science and Dept. of Mathematics and Statistics, James Cook University of North Queensland, Tech. Rep, 92-01.

[2] Aitkin, M., & Rubin, D. B. (1985). Estimation and hypothesis testing in finite mixture models. Journal of the Royal Statistical Society. Series B (Methodological), 67-75. Aitkin, M., Anderson, D., & Hinde, J. (1981). Statistical modelling of data on teaching styles. Journal of the Royal Statistical Society. Series A (General), 419-461.

[3] Aldenderfer M. S. and Blashfield R. K. (1984), Cluster analysis, Sage Publications, London, England.

[4] Alfò and Viviani (2015), Finite Mixtures of Structured Models, in Hennig, C., Meila, M., Murtagh, F., & Rocci, R. (2015), Handbook of cluster analysis. CRC Press.

[5] Amendola C., Faugere J-C and Sturmfels B. (2016), Moment Varieties of Gaussian Mixtures, Journal of algebraic statistics, Vol. 7, No. 1, 2016, 14-28.

[6] Anderberg, M. R. (1973). Cluster analysis for applications (No. 519.53 A543). Academic Press.

[7] Anderson, E. (1935). The irises of the Gaspe Peninsula, Bulletin of the American Iris Society, 59, 2–5.

[8] Äyrämö, S., & Kärkkäinen, T. (2006). Introduction to partitioning-based clustering methods with a robust example. Reports of the Department of Mathematical Information Technology. Series C, Software engineering and computational intelligence 1/2006.

[9] Azzalini, A. (2001), Inferenza Statistica. Una Presentazione Basata sul Concetto di Verosimiglianza, Springer-Verlag Italia, Milano.

[10] Balakrishnan N., Scarpa B. (2012), Multivariate measures of skewness for the skew-normal distribution, Journal of Multivariate Analysis Volume 104, Issue 1, February 2012, Pages 73-87.

[11] Bandyopadhyay S, Saha S (2007) GAPS: a clustering method using a new point symmetry based distance measure. Pattern Recognit 40(12):3430–3451

[12] Bandyopadhyay, S., & Maulik, U. (2002). Genetic clustering for automatic evolution of clusters and application to image classification. Pattern recognition, 35(6), 1197-1208.

[13] Bandyopadhyay, S., & Saha, S. (2008). A point symmetry-based clustering technique for automatic evolution of clusters. IEEE Transactions on Knowledge and Data Engineering, 20(11), 1441-1457.

[14] Banfield J. D. and Raftery A. E. (1993), Model-based Gaussian and non-Gaussian clustering. Biometrics, 49:803–821.

[15] Baum, L. E. and Eagon, J. A. (1967). An inequality with applications to statistical estimation for proba- bilistic functions of Markov processes and to a model for ecology. Bull. Amer. Math. Soc., 73, 360-363.

[16] Ben-Hur, A., & Guyon, I. (2003). Detecting stable clusters using principal component analysis. Functional Genomics: Methods and Protocols, 159-182.

[17] Bertoletti, M., Friel, N., & Rastelli, R. (2015). Choosing the number of clusters in a finite mixture model using an exact integrated completed likelihood criterion. Metron, 73(2), 177-199.

[18] Bezdek J. C. (1973a). Fuzzy Mathematics in Pattern Classification, PhD Thesis, Cornell University, Ithaca, NY.

[19] Bezdek J. C. (1973b), Cluster validity with fuzzy sets JC Bezdek Taylor & Francis Group 3 (3), 58-73

[20] Bezdek J. C.(1981), Pattern Recognition With Fuzzy Objective Function Algorithms, Springer US.

[21] Bezdek, J. C., Ehrlich, R., & Full, W. (1984). FCM: The fuzzy c-means clustering algorithm. Computers & Geosciences, 10(2-3), 191-203.

[22] Biernacki, C., Celeux, G., & Govaert, G. (2000). Assessing a mixture model for clustering with the integrated completed likelihood. IEEE transactions on pattern analysis and machine intelligence, 22(7), 719-725.

[23] Bouman, C. A., Shapiro, M., Cook, G. W., Atkins, C. B., & Cheng, H. (1997). Cluster: An unsupervised algorithm for modeling Gaussian mixtures.

[24] Bradley P. S. and Fayyad U. M. (1998), Refining initial points for K-Means clustering, in Proc. 15th International Conf. on Machine Learning, Morgan Kaufmann, San Francisco, CA, pp. 91–99.

[25] C.H. Chou, M.C. Su, E. Lai (2002) Symmetry as a new measure for cluster validity, in: Second WSEAS International Conference on Scientific Computation and Soft Computing, 2002, pp. 209–213.

[26] Campbell, N.A. and Mahon, R.J. (1974) A multivariate study of variation in two species of rock crab of genus Leptograpsus. Australian Journal of Zoology 22, 417–425

[27] Cebeci, Z., & Yildiz, F. (2015). Comparison of K-Means and Fuzzy C-Means Algorithms on Different Cluster Structures. AGRÁRINFORMATIKA/JOURNAL OF AGRICULTURAL INFORMATICS, 6(3), 13-23.

[28] Celeux, G., & Govaert, G. (1995). Gaussian parsimonious clustering models. Pattern recognition, 28(5), 781-793.

[29] Charytanowicz, M., Niewczas, J., Kulczycki, P., Kowalski, P. A., Łukasik, S., & Żak, S. (2010). Complete gradient clustering algorithm for features analysis of x-ray images. In Information technologies in biomedicine (pp. 15-24). Springer, Berlin, Heidelberg.

[30] Chen D. (1977), On two or more dimensional optimum quantizers, in Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP '77., vol. 2, Telecommunication Training Institute, Taiwan, Republic of China, pp. 640–643.

[31] Chen, J. (2017). Consistency of the MLE under mixture models. Statistical Science, 32(1), 47-63.

[32] Chen, J., & Kalbfleisch, J. D. (1996). Penalized minimum-distance estimates in finite mixture models. Canadian Journal of Statistics, 24(2), 167-175.

[33] Chen, J., Tan, X., & Zhang, R. (2008). Inference for normal mixtures in mean and variance. Statistica Sinica, 443-465.

[34] Chu, S. C., Roddick, J. F., & Pan, J. S. (2001). A comparative study and extension to K-medoids algorithms. Contemporary Development Company.

[35] Chung, K.-L. and J.-S. Lin (2007), Faster and more robust point symmetry-based K-means algorithm, Pattern Recognition, vol. 40, no. 2, pp. 410–422.

[36] Coretto, P., & Hennig, C. (2013a). Consistency, breakdown robustness, and algorithms for robust improper maximum likelihood clustering. arXiv preprint arXiv:1309.6895.

[37] Coretto, P., & Hennig, C. (2013b). Finding approximately Gaussian clusters via robust improper maximum likelihood. arXiv preprint arXiv:1309.6895.

[38] Cuesta-Albertos, J. A., Gordaliza, A., & Matrán, C. (1997). Trimmed K-means: An attempt to robustify quantizers. The Annals of Statistics, 25(2), 553-576.

[39] Dasgupta, S. (1999). Learning mixtures of Gaussians. In Foundations of computer science, 1999. 40th annual symposium on (pp. 634-644). IEEE.

[40] Davies, R. B. (1980). Algorithm AS 155: The distribution of a linear combination of \chi ^ {2}

[41] Day, N. E. (1969). Estimating the components of a mixture of normal distributions. Biometrika, 56(3), 463-474.

[42] Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. Journal of the royal statistical society. Series B (methodological), 1-38.

[43] Dennis Jr, J. E. (1981). Algorithms for nonlinear fitting. Proceedings of the NATO Advanced. Research Symposium, Cambridge University, Cambridge.

[44] Dotto, F., Farcomeni, A., García-Escudero, L. A., & Mayo-Iscar, A. (2017). Robust Fuzzy Clustering via Trimming and Constraints. In Soft Methods for Data Science, pp. 197-204. Springer International Publishing.

[45] Dubes R. C. (1987), How many clusters are best? - an experiment., Pattern Recognition, 20, pp. 645–663.

[46] Duda, R. O., Hart, P. E., & Stork, D. G. (1973). Pattern classification (pp. 526-528). Wiley, New York.

[47] Dunn, J. C. (1973), A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters. Journal of Cybernetics. 3 (3): 32–57.

[48] Dvoretzky, A. (1961), Some results on convex bodies and Banach spaces, In: Proceedings of the 1960 International Symposium on Linear Spaces, pp. 123, Jerusalem Academic Press, Jerusalem, and Pergamon Press, Oxford.

[49] Estivill-castro V. (2002), Why so many clustering algorithms: A position paper, SIGKDD Explorations Newsletter, 4, pp. 65–75.

[50] Feng, Z. D., & McCulloch, C. E. (1996). Using bootstrap likelihood ratios in finite mixture models. Journal of the Royal Statistical Society. Series B (Methodological), 609-617.

[51] Ferraro, M. B., & Giordani, P. (2015). A toolbox for fuzzy clustering using the R programming language. Fuzzy Sets and Systems, 279, 1-16.

[52] Fisher W. D. (1958), On grouping for maximum homogeneity, Journal of the American Statistical Association, 53, pp. 789–798.

[53] Fisher, D. H. (1996). Iterative optimization and simplification of hierarchical clusterings. Journal of artificial intelligence research, 4(1), 147-179.

[54] Fisher, R. A. (1921). On the probable error of a coefficient of correlation deduced from a small sample. Metron, 1, 3-32.

[55] Fisher, R. A. (1922). On the mathematical foundations of theoretical statistics. Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character, 222, 309-368.

[56] Fisher, R. A. (1936) The use of multiple measurements in taxonomic problems. Annals of Eugenics, 7, Part II, 179–188.

[57] Forgy, E. W. (1965). Cluster analysis of multivariate data: efficiency vs interpretability of classifications. Biometrics 21, 768–769.

[58] Fraley C. and Raftery A. E. (1998), How many clusters? Which clustering method? - Answers via model-based cluster analysis. The Computer Journal, 41:578–588.

[59] Fraley, C., & Raftery, A. E. (1999). MCLUST: Software for model-based cluster analysis. Journal of classification, 16(2), 297-306.

[60] Fraley, C., & Raftery, A. E. (2006). MCLUST version 3: an R package for normal mixture modeling and model-based clustering. WASHINGTON UNIV SEATTLE DEPT OF STATISTICS.

[61] Fraley, C., Raftery, A., Scrucca, L., Murphy, T. B., Fop, M., & Scrucca, M. L. (2017). Package 'mclust'.

[62] Friedman, H. P., & Rubin, J. (1967). On some invariant criteria for grouping data. Journal of the American Statistical Association, 62(320), 1159-1178.

[63] García-Escudero, L. A., Gordaliza, A., Matrán, C., & Mayo-Iscar, A. (2008). A general trimming approach to robust cluster analysis. The Annals of Statistics, 1324-1345.

[64] Haberman, S. J. (1976). Iterative scaling procedures for log-linear models for frequency tables derived by indirect observation. Proc. Amer. Statist. Assoc. (Statist. Comp. Sect. 1975), pp. 45-50.

[65] Hall, D. J., & Ball, G. B. (1965). ISODATA: A novel method of cluster analysis and pattern classification. Technical Report, Stanford Research Institute, Menlo Park, California.

[66] Hamerly, G., & Elkan, C. (2004). Learning the k in k-means. In Advances in neural information processing systems (pp. 281-288).

[67] Handl, J., & Knowles, J. (2007). An evolutionary approach to multiobjective clustering. IEEE transactions on Evolutionary Computation, 11(1), 56-76.

[68] Hartigan, J. A. and Wong, M. A. (1979). A K-means clustering algorithm. Applied Statistics 28, 100–108.

[69] Hasselblad, V. (1966). Estimation of parameters for a mixture of normal distributions. Technometrics, 8(3), 431-444.

[70] Hathaway, R. J. (1985). A constrained formulation of maximum-likelihood estimation for normal mixture distributions. The Annals of Statistics, 795-800.

[71] Hathaway, R. J. (1986). Another interpretation of the EM algorithm for mixture distributions. Statistics & probability letters, 4(2), 53-56.

[72] Hathaway, R. J., Bezdek, J. C., & Hu, Y. (2000). Generalized fuzzy c-means clustering strategies using L/sub p/norm distances. IEEE Transactions on Fuzzy Systems, 8(5), 576-582.

[73] Hennig, C., Meila, M., Murtagh, F., & Rocci, R. (Eds.). (2015). Handbook of cluster analysis. CRC Press. Holmes, G.K. (1892), Measures of distribution. Journal of the American Statistical Association 3, 141-157.

[74] Horton, P., & Nakai, K. (1996). A probabilistic classification system for predicting the cellular localization sites of proteins. In Ismb (Vol. 4, pp. 109-115).

[75] Huang, Z. (1998). Extensions to the k-means algorithm for clustering large data sets with categorical values. Data mining and knowledge discovery, 2(3), 283-304.

[76] Hubert, L. and P. Arabie (1985). Comparing partitions. Journal of Classi¯cation 2, 193{218.

[77] Ingrassia, S., & Rocci, R. (2007). Constrained monotone EM algorithms for finite mixture of multivariate Gaussians. Computational Statistics & Data Analysis, 51(11), 5339-5351.

[78] Jain A., Dubes R. (1981). Algorithms for clustering data, Prentice-Hall.

[79] Jain, A. K. (2010). Data clustering: 50 years beyond K-means. Pattern recognition letters, 31(8), 651-666. K.-L.

[80] Kaufman, L. and Rousseeuw P. J. (1990), Finding groups in data: An introduction to cluster analysis, JohnWiley & Sons.

[81] Kaufman, L. and Rousseeuw, P.J. (1987), Clustering by means of Medoids, in Statistical Data Analysis, Y. Dodge, North-Holland, 405–416.

[82] Khan, S. S., & Ahmad, A. (2004). Cluster center initialization algorithm for K-means clustering. Pattern recognition letters, 25(11), 1293-1302.

[83] Kiefer, J., & Wolfowitz, J. (1956). Consistency of the maximum likelihood estimator in the presence of infinitely many incidental parameters. The Annals of Mathematical Statistics, 887-906.

[84] Krishnapuram, R., & Keller, J. M. (1993). A possibilistic approach to clustering. IEEE transactions on fuzzy systems, 1(2), 98-110.

[85] Lance, G. N., & Williams, W. T. (1966). A generalized sorting strategy for computer classifications. Nature, 212(5058), 218-218.

[86] Lehmann, E. L. (1980). Efficient likelihood estimators. The American Statistician, 34(4), 233-235.

[87] Li, M. J., Ng, M. K., Cheung, Y. M., & Huang, J. Z. (2008). Agglomerative fuzzy k-means clustering algorithm with selection of number of clusters. IEEE transactions on knowledge and data engineering, 20(11), 1519-153

[88] Linde Y., Buzo A. and Gray R. (1980), An algorithm for vector quantizer design, IEEE Transactions on Communications, 28, pp. 84–95.

[89] Lindstrom, M. J., & Bates, D. M. (1988). Newton-Raphson and EM algorithms for linear mixed-effects models for repeated-measures data. Journal of the American Statistical Association, 83(404), 1014-1022.

[90] Lloyd, S. P. (1957, 1982). Least squares quantization in PCM. Technical Note, Bell Laboratories. Published in 1982 in IEEE Transactions on Information Theory 28, 128–137.

[91] MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, eds L. M. Le Cam & J. Neyman, 1, pp. 281–297. Berkeley, CA: University of California Press.

[92] Mahalanobis, P. C. (1936). On the generalised distance in statistics. Proceedings of the National Institute of Sciences of India, 1936, 49-55.

[93] Maitra, R. (2009). Initializing partition-optimization algorithms. IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB), 6(1), 144-157.

[94] Maitra, R., & Melnykov, V. (2010). Simulating data to study performance of finite mixture modeling and clustering algorithms. Journal of Computational and Graphical Statistics, 19(2), 354-376.

[95] Mardia, K. V. (1970), Measures of multivariate skewness and kurtosis with applications. Biometrika, 57, 519–530.

[96] Marina M. and David H. (1998), An experimental comparison of several clustering and initialization methods, in Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence (UAI-98), San Francisco, CA, Morgan Kaufmann Publishers, pp. 386–395.

[97] Marriot F.H.C. (1974), The interpretation of multiple observations. Academic Press, London.

[98] McLachlan, G. J., Peel, D., Basford, K. E., & Adams, P. (1999). The EMMIX software for the fitting of mixtures of normal and t-components. Journal of Statistical Software, 4(2), 1-14.

[99] McLachlan, G. J., & Peel, D. (1999). Computing issues for the EM algorithm in mixture models. In Interface'99 (Vol. 3, pp. 421-430). Interface Foundation of North America.

[100] McLachlan G.J. and Peel D. (2000), Finite Mixture Models, New York: Wiley.

[101] McLachlan, G., & Krishnan, T. (2007). The EM algorithm and extensions (Vol. 382). John Wiley & Sons.

[102] McNicholas, P. D., Murphy, T. B., McDaid, A. F., & Frost, D. (2010). Serial and parallel implementations of model-based clustering via parsimonious Gaussian mixture models. Computational Statistics & Data Analysis, 54(3), 711-723.

[103] Melnykov, V., Chen, W. C., & Maitra, R. (2012). MixSim: An R package for simulating data to study performance of clustering algorithms. Journal of Statistical Software, 51(12), 1.

[104] Ménard, M., Courboulay, V., & Dardignac, P. A. (2003). Possibilistic and probabilistic fuzzy clustering: unification within the framework of the non-extensive thermostatistics. Pattern Recognition, 36(6), 1325-1342.

[105] Mengersen, K. L., Robert, C., & Titterington, M. (2011). Mixtures: estimation and applications (Vol. 896). John Wiley & Sons.

[106] Milman, V.D. (1971), A new proof of A. Dvoretzky's theorem on cross-sections of convex bodies. Functional Analysis and Its Applications 5(4), 28-37.

[107] Murtagh, F., & Kurtz, M. J. (2016). The Classification Society's Bibliography Over Four Decades: History and Content Analysis. Journal of Classification, 33(1), 6.

[108] Newcomb, S. (1886). A generalized theory of the combination of observations so as to obtain the best result. American Journal of Mathematics 8, 343-366.

[109] Ng, R. T., & Han, J. (2002). CLARANS: A method for clustering objects for spatial data mining. IEEE transactions on knowledge and data engineering, 14(5), 1003-1016.

[110] Ohashi, Y. (1984, March). Fuzzy clustering and robust estimation. In 9th Meeting, SAS Users Group International, Hollywood Beach, FL, USA (pp. 1-6).

[111] Pearson, K (1894), Contributions to the Mathematical Theory of Evolution. Philosophical Transactions of the Royal Society of London A, 185, 71-110.

[112] Pedrycz, W. (1998). Shadowed sets: representing and processing fuzzy sets. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 28(1), 103-109.

[113] Pena J. M. , Lozano J. A., and Larranaga P. (1999), An empirical comparison of four initialization methods for the k-means algorithm, Pattern Recognition Letters, 20.

[114] Piccolo, D. (2010), Statistica, Bologna, Il Mulino.

[115] Quetelet, A. (1846), Lettres à S.A.R. le duc régnant de Saxe-Coburg et Gotha, sur la théorie des probabilités, appliquée aux sciences morales et politiques. Brussels: Hayez.

[116] Quetelet, A. (1852). Sur quelques propriétés curieuses que présentent les résultats d'une serie d'observations, faites dans la vue de déterminer une constante, lorsque les chances de rencontrer des écarts en plus et en moins sont égales et indépendantés les unes des autres. Bulletins de l'Académie Royale des Sciences, des Lettres et des Beaux-arts de Belgique 19, 303-317.

[117] Rao, C.R. (1948). The utilisation of multiple measurements in problems of biological classification. Journal of the Royal Statistical Society B10,159-203.

[118] Ray, S., & Lindsay, B. G. (2005). The topography of multivariate normal mixtures. Annals of Statistics, 2042-2065.

[119] Redner, R. (1981). Note on the consistency of the maximum likelihood estimate for nonidentifiable distributions. The Annals of Statistics, 225-228.

[120] Rousseeuw P. J. (1987), Silhouettes: A graphical aid to the interpretation and validation of cluster analysis, Journal of Computational and Applied Mathematics, 20, pp. 53–65.

[121] Ruspini, E. H. (1969). A new approach to clustering. Information and control, 15(1), 22-32.

[122] Ruspini, E. H. (1970). Numerical methods for fuzzy clustering. Information Sciences, 2(3), 319-350.

[123] Saha S, Bandyopadhyay S (2008) Application of a new symmetry based cluster validity index for satellite image segmentation. IEEE Geosci Remote Sens Lett 5(2):166–170

[124] Saha S, Bandyopadhyay S (2009a) A new multiobjective simulated annealing based clustering technique using symmetry. Pattern Recognit Lett 30(15):1392–1403

[125] Saha S, Bandyopadhyay S (2009b) A new line symmetry distance and its application to data clustering. J Comput Sci Technol 24(3):544– 556

[126] Saha S, Bandyopadhyay S (2010a) A symmetry based multiobjective clustering technique for automatic evolution of clusters. Pattern Recognit 43(3):738–751

[127] Saha S, Bandyopadhyay S (2010b) A new multiobjective clustering technique based on the concepts of stability and symmetry. Knowl Inf Syst 23(1):1–27

[128] Saha S, Bandyopadhyay S (2011) On principle axis based line symmetry clustering techniques. Memet Comput 3(2):129–144

[129] Saha S, Bandyopadhyay S (2013) A generalized automatic clustering algorithm in a multiobjective framework. Appl Soft Comput 13:89–108

[130] Saha S, Maulik U (2011) A new line symmetry distance based automatic clustering technique: application to image segmentation. Int J Imaging Syst Technol 21(1):86–100

[131] Saha S, Spandana R, Ekbal A, Bandyopadhyay S (2015) Simultaneous feature selection and symmetry based clustering usingmultiobjective framework. Appl Soft Comput 29:479–486

[132] Scholz, F.W. (2006). Maximum likelihood estimation. In Encyclopedia of Statistical Sciences, second edition, S. Kotz, N. Balakrishnan, C.B. Read, B. Vidakovic and N.L. Johnson (eds.), 4629–4639. Wiley, Hoboken, NJ.

[133] Schwarz, G. (1978). Estimating the dimension of a model. The annals of statistics, 6(2), 461-464.

[134] Scott, A. J., & Symons, M. J. (1971). Clustering methods based on likelihood ratio criteria. Biometrics, 387-397.

[135] Scrucca, L., Fop, M., Murphy, T. B., & Raftery, A. E. (2016). mclust 5: Clustering, classification and density estimation using gaussian finite mixture models. The R Journal, 8(1), 289.

[136] Steinley, D., & Brusco, M. J. (2007). Initializing k-means batch clustering: A critical evaluation of several techniques. Journal of Classification, 24(1), 99-121.

[137] Stigler, S. M. (1973). Simon Newcomb, Percy Daniell, and the history of robust estimation 1885–1920. Journal of the American Statistical Association, 68(344), 872-879.

[138] Su, M. C., & Chou, C. H. (1999). A competitive learning algorithm using symmetry. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, 82(4), 680-687.

[139] Su, M. C., & Chou, C. H. (2001) A modified version of the K-means algorithm with a distance based on cluster symmetry. IEEE Trans Pattern Anal Mach Intell 23(6):674–680

[140] Sumikawa T., Koizumi K., Seo T. (2014), Measures of multivariate skewness and kurtosis in high-dimensional framework. SUT Journal of Mathematics, ISSN 0916-5746, Vol. 50, no 2, p. 483-511.

[141] Titterington, D. M., Smith, A. F., & Makov, U. E. (1985). Statistical analysis of finite mixture distributions. Wiley.

[142] Titterington, D. M. (1990). Some recent research in the analysis of mixture distributions. Statistics, 21(4), 619-641.

[143] Titterington, D.M. (1996). Mixture distributions (update). In Encyclopedia of Statistical Sciences, Vol. 5, S. Kotz, N.L. Johnson, and D. Banks (Eds.). New York: Wiley, pp.399-407.

[144] Vattani, A. (2011). K-means requires exponentially many iterations even in the plane. Discrete & Computational Geometry, 45(4), 596-616.

[145] Vijendra S., Laxman S. (2015) Symmetry Based Automatic Evolution of Clusters: A New Approach to Data Clustering. Computational Intelligence and Neuroscience Volume 2015

[146] Villaverde K., Kosheleva O. and Ceberio M. (2010), Why Ellipsoid Constraints, Ellipsoid Clusters, and Riemannian Space- Time: Dvoretzky's Theorem Revisited. Departmental Technical Reports (CS). Paper 19.

[147] Vinod Hrishikesh D. (1969), Integer Programming and the Theory of Grouping, Mathematica Journal of the American Statistical Association Vol. 64 , Iss. 326.

[148] Wald, A. (1949). Note on the consistency of the maximum likelihood estimate. The Annals of Mathematical Statistics, 20(4), 595-601.

[149] Wang L., Bo L., Jiao L. (2006), A Modified K-Means Clustering with a Density-Sensitive Distance Metric. In: Wang GY., Peters J.F., Skowron A., Yao Y. (eds) Rough Sets and Knowledge Technology. RSKT 2006. Lecture Notes in Computer Science, vol 4062. Springer, Berlin, Heidelberg.

[150] Ward Jr, J. H. (1963). Hierarchical grouping to optimize an objective function. Journal of the American statistical association, 58(301), 236-244.

[151] Weldon,W.F.R. (1892), Certain Correlated Variations in Crangon Vulgaris. Proceedings of the Royal Society of London,51,2-21.

[152] Weldon W.F.R. (1893), On Certain Correlated Variations in Carcinus Moenas. Proceedings of the Royal Society of London,54,318-329.

[153] Wolfe, J. H. (1965). A computer program for the maximum likelihood analysis of types (No. TB-65-15). U.S. Naval Personnel Research Activity, San Diego.

[154] Wolfe, J. H. (1967). NORMIX: computational methods for estimating the parameters of multivariate normal mixtures of distributions (No. NPRA-SRM-68-2). U.S. Naval Personnel Research Activity, San Diego.

[155] Wu, C. J. (1983). On the convergence properties of the EM algorithm. The Annals of statistics, 95-103. Wu, K. L., & Yang, M. S. (2002). Alternative c-means clustering algorithms. Pattern recognition, 35(10), 2267-2278.

[156] Xiao, Y., & Yu, J. (2012). Partitive clustering (K-means family). Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2(3), 209-225.

[157] Yin, J., Sun, H., Yang, J., & Guo, Q. (2014). Comparison of K-Means and fuzzy c-Means algorithm performance for automated determination of the arterial input function. PloS one, 9(2).

[158] Yu, J., & Yang, M. S. (2007). A generalized fuzzy clustering regularization model with optimality tests and model complexity analysis. IEEE Transactions on Fuzzy Systems, 15(5), 904-915

[159] Zucchini W. & MacDonald, I. L. (2009). Hidden Markov models for time series: an introduction using R. Chapman & Hall/CRC.