Università degli Studi di Napoli
Federico II

Non-hierarchical clustering methods on factorial subspaces

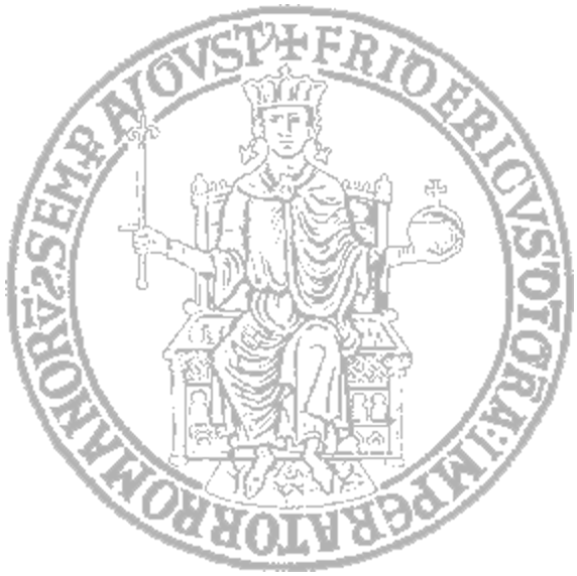
Cristina Tortora

Doctoral Thesis
in Statistics

XXIV Cycle

Dipartimento
di Matematica e Statistica
Università degli Studi di Napoli "Federico II"
via Cintia, Monte Sant'Angelo – 80126 Napoli

Non-hierarchical clustering methods on factorial subspaces



Napoli, December the 1^{st} 2011

III

Ringraziamenti

La mia padronanza e familiarità con i numeri non è neanche lontanamente paragonabile alla mia ostilità con le parole: scrivere dei ringraziamenti sinceri è più difficile di una tesi di dottorato!
Proverò ugualmente a ringraziare le persone che mi hanno permesso di raggiungere questo obiettivo:

Ringrazio il prof. Carlo Lauro che mi ha sempre aiutato con i suoi consigli accurati e precisi; è stato la stella polare che ha consentito di orientarmi, la sua presenza costante e la sua disponibilità a rispondere a ogni mia domanda mi hanno permesso di affrontare al meglio questi 3 anni.

I più sentiti ringraziamenti vanno al prof. Francesco Palumbo, il mio mentore, supporto e consigliere. In tutto il periodo di studi, fin dagli anni universitari, è sempre stato prodigo di consigli sia dal punto di vista professionale che umano fino ad essere il faro che ha aiutato a districarmi fra gli scogli e le nebbie del mio percorso.

Sull'altra sponda ecco comparire la Prof. Marina Marino, è anche grazie al suo supporto che l'attraversamento delle acque talvolta agitate e impetuose è divenuto d'improvviso un tranquillo navigare in acque calme spinti da una favorevole brezza.

Un indispensabile contributo è arrivato anche dall'effervescente cugina d'oltralpe, la prof. Mireille Gettler Summa, che con la sua esuberante energia mi ha stimolato per il raggiungimento di traguardi non immediatamente evidenti.

Tutto questo cammino non sarebbe stato possibile senza l'appoggio e il contributo delle due navigatrici, la prof.ssa Germana Scepi e la prof.ssa Simona Balbi sempre disponibili a guidarmi, consigliarmi e pronte a correggere le mie incertezze e deviazioni, riportandomi sulla rotta.

Questa traversata è stata accompagnata e resa più proficua e piacevole dai compagni di viaggio: Laura, Giorgio, Carlo, Agnieszka, Marcella, Nicole, Lidia, Maria, Stefania ed Enrico.

Infine un abbraccio agli sponsor di questa traversata, mamma e papà, a tutti i miei fratelli, a Fabio, a Sabrina e agli amici, indispensabile supporto morale di questo entusiasmante viaggio.

Cristina

Contents

List of Tables

List of Figures

List of Notations

X	data matrix of general element x_{ij}
n	number of units
J	number of variables
D	of elements d_{ij}, distance between the units i and j
P	of elements p_{ik}, probability that unit i belongs to cluster k
K	number of clusters
C	of elements c_{kj}, center of cluster k
x_i^k	units belonging to cluster k
W	within-cluster variance
B	between-cluster variance
I_k	distorsion error of cluster k
n_k	number of units belonging to cluster k
n_j	number of modality for variable j
J^*	total number of modality
Z	of elements z_{ij}, disjunctive coded data matrix
$\mathcal{B}$	Burt matrix
$f_{i,j}$	relative frequencies
$f_{.,j}$	marginal frequencies
F	coordinates of units in the factorial space
U	cluster membership
E	error components matrix
$\bar{Y}$	centroids matrix
JDF	Joint Distance Function

Introduction

Learning abilities derive by the human brain capability in recognizing and categorizing objects, where the word *object* must be interpreted in its wider acceptation. In this context, emotions, odorous, flavors, and whatever else represents a part of the knowledge, are considered objects. Steven Pinker wrote: *"An intelligent being cannot treat every object it sees as a unique entity unlike anything else in the universe. It has to put objects in categories so that it may apply its hard-won knowledge about similar objects encountered in the past, to the object at hand."* (How the Mind Works, 1997). When the human interest began to be addressed at the multiple faces of real World, this inborn human knowledge process was quickly coded in many fields of the Science. The *Taxonomy* exactly represents this process. However, many other examples could be found: libraries and bookstores, for example, arrange the books according to the topic and then by authors name. Actually, a classification scheme represents a convenient method for organising a large data set so it can be understood more easily and information retrieved more efficiently. The final aim consists in summarising the whole dataset by a small number of groups of homogeneous objects, then groups are labeled by concise descriptions of the similarities in each group.

Wherever objects could be measured and coded into numerical data (like

people, animals, chemical elements, sounds, etc.), the process based on the grouping of similar objects can be automated. Statistics and statisticians have largely contributed to the development of increasingly sophisticated classification methods.

The need to group data is increasingly important because of the growing of the whole quantity of digitally stored data. The exponential growth of the global amount of the stored data has strongly stimulated the scholar on clustering and classification: in statistics and in other disciplines, as well. The interest is proven by the large number of scientific articles published on the topic.

Some of these techniques have been independently proposed and implemented by different scholars in their respective disciplines. However, the different backgrounds have had a strong influence on the formal definitions and notations, rather than on the substantial methodological proposal. So we can observe that in biology classification is also called *Numerical taxonomy*, whereas psychologists use the term *Q Analysis*, in the artificial intelligence domain it is used the term *unsupervised pattern recognition* to refer to these methods. As the Statisticians have adopted the term *Cluster Analysis*, in the following chapters of this work we will use this definition. For sake of homogeneity, the same notation will be preserved, through all over the thesis, even illustrating methods that have been originally presented with an alternative (and may be more appropriate) notation.

When dealing with Cluster Analysis, in Statistics we are used to divide the methods in two main branches: *hierarchical* and *non–hierarchical* methods. The difference between these two big families depends on the clustering strategy. Methods in the former category have two possible starting points: *i*) n object grouped into n singleton clusters, where n is the total number

of objects to classify; *ii*) one cluster made by n objects. Depending on the starting configuration, the clustering algorithm aggregates into a cluster the two most similar objects, at each step; alternatively, the algorithm splits the most heterogeneous cluster into two more homogeneous clusters. Former ones are called *aggregative* methods and latter ones are called *divisive* methods. However, independently from the strategy, the clustering produces a nested clustering structure. Generally heterogeneity index permits to identify the best clustering configuration. Moving *up* and *down* along the clustering structure, it is possible to choose the most appropriate partition, and the number of clusters as a consequence.

Hierarchical clustering algorithms (in the most of cases) permit to have a graphical representation of the aggregative structure, they are preferably implemented when the total number of objects to classify is almost limited and when it is important to understand (and visualize) the dis/similarities between two objects or two clusters.

In this thesis the attention is focused on non-hierarchical clustering methods, which are also called partitioning methods. These methods assume *to be known* the number of clusters in the data and their goal is to achieve the *best* classification of the objects to the clusters. The quality of the classification is evaluated according to a given criterion, and the word *best* has no sense until the criterion has not been indicated by the analyst.

The clustering problem consists in grouping the objects so that the resulting value of the chosen criterion satisfies its optimal condition (which can be the maximum or the minimum according to the criterion itself). If the number of clusters k is assumed known the total number of partitions of n objects may be computed. However, it is clear and intuitive that this number quickly becomes large (and even very large!) as n becomes large.

From a numerical point of view, non-hierarchical clustering algorithms are considered Non-deterministic Polynomial-time hard problems (*NP-hard*), whose have no solutions in polynomial time. Only a very small n allows to perform the exhaustive search to choose the best one. If n is large most algorithms are likely to find a partition that is sub–optimal.

Moreover, in the most of the cases, the global number of available *features* is quite large and not all features are really necessary. In a geometrical optics, given a set of n statistical units represented as points in the multi-dimensional space spanned by p variables, clustering aims at finding dense regions in the original feature space or embedded in a properly defined subspace. However, when p is large, the global number of available variables is definitively greater than the number of really necessary variables. From a pure theoretical point of view, the number of dimensions that are really necessary to *separate* k clusters is equal to $\text{int}\left(\log_2(k)\right)$: the unidimensional space can separate two clusters, from 2 to 4 clusters a two dimensional space is needed, a three dimensional space permits to separate from $2^2 + 1$ to 2^3 clusters, four dimensions from $2^3 + 1$ to 16 clusters, an so on... In fact, the high dimensional space determines that points are sparse and it makes more difficult to find *dense* regions. Two solutions are possible: *i*) to select the features holding the "good" variables and discarding the others; *ii*) to define a set of q new variables, where $q \ll p$ and the new variables are linear combinations of the original ones. Each strategy hat its drawbacks and advantages, the thesis goes into more details on this aspects and mainly focuses the attention on the variables transformation.

In the recent years, a new category of non-hierarchical clustering algorithms has been introduced. The novelty aspect introduced with these algorithms consists in having one integrated strategy where the variables

transformation and the clustering phase are alternatively iterated satisfying the same common criterion. The algorithm stops when the criterion reaches its highest (lowest) value. These algorithms permit to refine the variable combination at each step increasing their separating capabilities.

The spread of these methods has been driven by the need to handle large amounts of data, their considerable increase is due to the ever growing improvement of data storage capability and of informatics devices diffusion. Applications as Internet or digital imaging or video surveillance are increasingly diffused. These determine the increase of high-dimensional datasets that can be spread through Internet or through portable device that are more and more capacious. These datasets can be easily shared; on internet everybody can have access to a huge number of datasets made by large number of variables. In many fields every day there are new datasets that can be shared with other people. The number of access to web pages is just an example: every day million of people surf on Internet and it is possible to know what website they have visited, which pages and how long they stay on each page. These datasets are very interesting, they are real gold mines for marketing managers, for a head of sales or for someone interested in social or psychological science. Facing with such a huge amount of data, how can we extract knowledge? How can we let the data "speak"?
This is a very typical problem in which statisticians, or someone interested in obtaining information are faced with. Useful methods are needed to manage high-dimensional datasets and to catch information contained in the data. Probably the need of these methods has led the birth of the Knowledge Discovery in Databases (KDD) and of data mining. Data mining refers to the process of selection and processing a large amount of data to discover hidden relationships, patterns and similarities, not known a pri-

ori. The word mining evokes the extraction of gold nuggets or precious stone from the mine, the choose of this term emphasizes how difficult can be the process.

Factorial clustering methods are based on two main approaches: probabilistic and geometric. Probabilistic clustering methods, also called model based clustering, start from the assumption that each cluster is a subpopulation that has his own model with his own parameters. The population is a mixture of all the subpopulations. The extension of this methods to a factorial clustering methods has led to the development of mixtures of factor analyzers where each subpopulation has been generated according to a factorial model. The estimation of factorial models, one for each subpopulation, and of cluster membership leads to a partition of unit and to a dimensionality reduction. Geometric clustering methods are based on the concept of distance. According to these methods a partition adequacy criterion is chosen, data points are projected on the factorial space that optimize the criterion. In this new space the partition that optimizes the same criterion is computed. The computation of the factorial space and of the cluster partition simultaneously is unfeasible, the two quantity are computed alternatively and iteratively. For this reason these methods are also called iterative two-step clustering methods.

This thesis is focused on clustering methods for high-dimensional datasets and in particular on factorial clustering methods. These methods give stable results dealing with large and very large datasets, however, they can have problems dealing with datasets characterized by some issues. Correlation between variables or heteroschedastic clusters can compromise the right clustering structure detection. The same difficulties can be caused by outliers, that can mask the real clustering structure. Some methods fail

dealing with arbitrarily shaped clusters, to cope with these problems a new factorial clustering method, *Factorial Probabilistic Distance Clustering*, is proposed. It performs a linear transformation of data and *Probabilistic Distance clustering (PD-clustering)* iteratively. PD-clustering is an iterative, distribution free, probabilistic, clustering method. Clusters centers are computed according to the constraint that the product between the belonging probability of a point to clusters and the distance of the point to clusters center is a constant. An iterative algorithm allows to compute the centers. When the number of variables is large and variables are correlated PD-Clustering becomes unstable and the correlation between variables can hide the real number of clusters. A linear transformation of original variables into a reduced number of orthogonal ones using common criteria with PD-Clustering can significantly improve the algorithm performance.

Thesis outline Chapter 1 presents a short historical overview of cluster analysis; followed by a brief excursus of the main domains where cluster analysis is used, putting the attention on the different purposes that cluster analysis can have. At each purpose corresponds a different field of application. In section 1.1 cluster analysis is detailed; the approach is a complex process that does not consist in the the mere execution of an algorithm. In the following subsections, each phase of the process is detailed.

There is a wide range of clustering methods, this thesis is focused on non hierarchical clustering ones that are the object of chapter 2. These methods require the a-priori choice of the number of clusters, some techniques the allow to choose the right number of clusters are detailed in section 2.1. Non hierarchical clustering methods are based on the optimization of a criterion chosen. A wide range of methods are based on the variance decomposition

that is detailed in section 2.2. The most simple and widely used non hierarchical clustering method is *k-means* that, under some optimality conditions, allows to obtain a good classification of data in an efficient way, k-means is detailed in section 2.2.1, 2.3 and 2.4. In section 2.5 methods for clusters representation are illustrated.

Chapter 3 is focused on cluster analysis of high-dimensional datasets. The first approach dealing with this type of data is two-step analysis that can be used dealing with quantitative data, section 3.1, and qualitative data, section 3.2. An improvement of these methods are the factorial clustering methods that can be probabilistic, section 3.3, or geometric, section 3.4. When clusters are of arbitrary shape, classical clustering methods can fail and sometimes is necessary to use methods based on the projection of points in a feature space, these methods are illustrate in section 3.5.

In chapter 4 a new factorial clustering method, *Factorial Probabilistic Distance Clustering* (FPDC), is detailed. FPDC is based on an already existing clustering method presented in section 4.1. FPDC is explained in section 4.2 and an example on a real dataset is presented in section 4.3.

Chapter 5 contains a computational study on Factorial Probabilistic Distance Clustering. Firstly in section 5.1 theoretical aspects of the simulation are explained and then, in section 5.2, there are the simulation results.

Chapter 1

Cluster Analysis

Many research fields present a common need; it's been given a dataset composed by many statistical units, the main need is to find homogenous groups of individuals, homogeneous is referred to individuals that present similar characteristics. This is the aim of cluster analysis, that had its origins in different areas at the same time. This phenomenon caused the birth of different denominations for the same subject; even the method cluster analysis is also called unsupervised classification. This circumstance can cause communication problems between different sectors, statisticians and "learners", specialized in machine learning, studied the same methods using different words. Especially in such a field where cluster analysis is applied, like biology or medicine, specialists in the field use a different language. Thus before coming into cluster analysis details a clear definition of concepts are important.

The term cluster analysis was first used by Robert Tryon [Tryon, 1939] in 1939. He developed the theory of cluster analysis and a particular set of cluster analysis methods in psychometric domain. However the first cluster

approach can be dated in 1932 [Driver and Kroeber, 1932] with Driver and Kroeber. They used a clustering approach studying the development of cultures. In 1951 Florek [Florek et al., 1951] proposed a new approach that is the father of modern, popular, hierarchical clustering. The development of cluster analysis started with the improvement of computational techniques and with the creation of modern computers in 1960. In 1967 MacQueen [MacQueen, 1967] proposed the K-means method, the most commonly used clustering method.

Nowadays cluster analysis has many field of applications: medicine, biology, social science, demography, psychology, ... The goal is to investigate the relationship between groups of units. Its aim is to establish whether exist groups of objects that have similar characteristics. The base of these methods is the definition of: i) a set of units; ii) a distance measure; iii) a clustering strategy; iv) a validation. The clusters can be an already known structure of the population that have to be revealed by the analysis, for example in case of clustering of flowers when the species are already known. The clusters and the number of clusters can also be unknown, like in the classification of web sites. The objective is to look for groups that are not already known, but that depends on: the variables taken into account, the parameters chosen and the method used during the analysis. Groups or clusters can be analyzed and defined from a geometric point of view; a cluster is a set of units that are close to each other , according to a specific criterion. In this case the chosen measure of distance have a central role. It is important to choose how to measure the distance between two clusters, linkage method, because each measure has his own characteristics and weaknesses. Many widely used methods are based on this geometric way of looking at clustering.

Clustering methods based on a geometric approach can be hierarchical or non hierarchical. Hierarchical methods aim at building a hierarchy of clusters that can be represented throughout a specific graph, called dendrogram, non hierarchical ones aim is to divide a set of object in K clusters. Another way to look at clustering is to define clusters as high density regions separated by low density regions.
In both cases the choice of search for groups of individuals in a datasets can arise from three main purposes:

- **Natural classification** or taxonomy, the objective is to identify the degree of similarity among shapes or organisms;
- **Images recognition** the aim is to recognize images or schemes into the data;
- **Find underlying structure** the objective is to find information into the data.

Taxonomy has been used since ancient times when Aristotle (384-322 B.C.), classified animals and plants. He proposed the *Five Predicables* to classify: genus, species, differences, properties and accidents. More than 2000 years later taxonomy is still a valid and used instrument in many fields. In chemistry the elements are clustered according to their characteristics; in geology the classification is applied on the earth structure, like: mineral, water, mountains, etc., and on the associated phenomena, like: earthquakes, magnetic field, etc.. These phenomena are still an interesting and popular field of study. Natural classification is used in human science too. In the linguistic field where words are classified by theme and in social and political sciences fields where a society function classification can be done.

Image recognition is used in many fields; images can be of different kind like letters or numbers or photos. The improvement of image recognition techniques are very useful for the automatically read of a license plate or for face recognition. Image recognition is obtained through specific clustering method based on machine learning or neuronal network.
With the increase of computer power there are always more datasets, often large dimensional dataset, in many field; this has led to a growing need for methods that find information into the data. On internet there are always more dataset containing the history of download of documents where the objective is to find interesting patterns. In genomic field there are many dataset containing few units and many variables and the objective is to catch the information in the data. This thesis is focused on this last purpose, to find underlying structure into the data.

1.1 Cluster analysis process

The application of a clustering method does not represent the whole clustering process: *When we refer to the term cluster analysis we are taking into account an entire process where clustering maybe only a step* [Gordon, 1999]. Cluster analysis can be split in 4 main steps:

- definition of the object matrix;
- transformation of the original data matrix;
- application of a clustering method;
- cluster validation.

Each step involves different choices that will be detailed in the following sections.

1.1.1 Definition of the object matrix

The first phase consists in the definition of the set of objects taken into account during the analysis. Considering a multivariate data matrix X, with n units and J variables, besides it some other informations about units can exist. Often nowadays there is a lot of information due to the use of automated methods of data collection and storage, these informations can be used in the analysis or it can be ignored. The choice depends on their correlation with the objective of the analysis. Moreover, the data matrix is often large with many variables that are not necessary related to the domain of interest. Sometimes it can be useful to select a subset of them or to reduce the dimension of the data matrix before starting the analysis.

Before choosing a way to treat the data, it is necessary to distinguish between type of variables. The most general way to distinguish variables is to divide them in three categories: numerical, ordinal and nominal. Numerical variables can be whether continuous or discrete. A variable is said continuous if it takes value in $\Re$ or a compact subset of $\Re$; it is possible to count, order and measure continuous variables. A variable is said discrete if the observations belonging to it are distinct and separate.
A variable is said ordinal if it is ranked or if it has a rating scale attached. It is possible to count and order, but not measure ordinal variables. The categories for an ordinal set of variables have a natural order, however, the distinction between neighboring points on the scale is not necessarily always the same.
A variable is said to be nominal or categorical if it can be assigned to a code as a numeric value where the numbers are simply labels. It is possible

to count but not order or measure nominal variables.
Each type of variable allows its own set of operations.

1.1.2 Transformation of original data matrix

Cluster analysis implies the tranformation from data matrix to: pattern matrix, distance matrix or (dis)similarity matrix. A pattern matrix X is a $n \times J$, objects $\times$ variables matrix, where the general element x_{ij} expresses the value of the variable j for the i unit with $i = 1, \ldots, n$ and $j = 1, \ldots, J$. In many clustering methods, as hierarchical ones, data matrix requires to be coded as distance matrix or (dis)similarity matrix, each cell of the matrix have to express a distance or a (dis)similarity measure between two elements. Distance or (dis)similarity matrix are symmetric $n \times n$ matrix where the general element $d_{i,j}$ provides a measure of the distance or (dis)similarity between the object i and the object j with $i, j = 1, \ldots, n$. Dealing with categorical data, distance measures used for numerical data cannot be used but some dissimilarity coefficients can be applied. A list of distance measures for numerical and categorical data can be found in Arabie et al. book [Arabie et al., 1996].

1.1.3 Applying a clustering method

This is the core of clustering process. Data are assigned to clusters where a cluster is *a set of one or more objet that we are willing to call similar to each other* [Romesburg, 2004]. A cluster can even be composed by only one object if there are not any other objects similar to it. Or a cluster can be composed by all the data units if all of them are similar to each other.
Traditional clustering methods can be divided into two main groups: hierarchical methods and nonhierarchical methods (fig. 1.1.3).

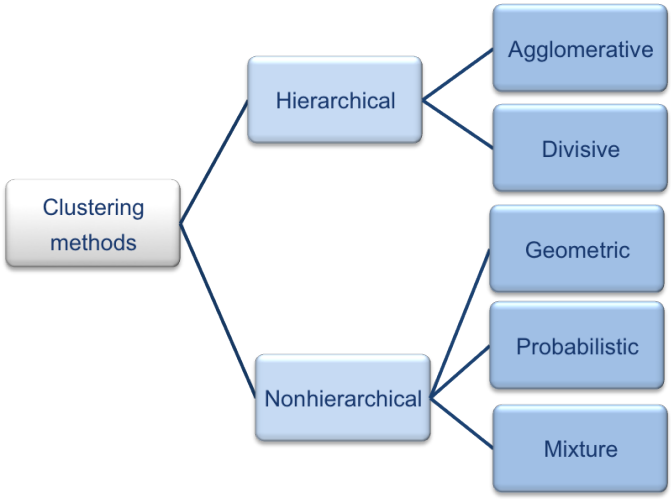


Figure 1.1: Clustering methods

The original purpose of hierarchical clustering was to reveal the taxonomy or the hierarchical structure among species of animals. Then their applications domain have been extended to marketing, in order to find a structure of products, nowadays these methods are used in a wide range of fields. Hierarchical clustering methods seek to build a hierarchy of clusters. To derive a hierarchical structure of n units, two type of strategies can be used: agglomerative and divisive.

Agglomerative methods start with n single element clusters and proceed by successively combination of clusters according to a proximity criterion until one single cluster is obtained. To use these methods it is necessary to define a linkage criterion between clusters or elements. A widely used criterion is Ward's linkage that agglomerates the clusters giving the minimum increase of the total sum of squares. To a complete detailed list of linkage criteria

refers to [Wedel and Kamakura, 1999].
Divisive methods start with only one cluster of all elements that are successively separated in smaller clusters until n clusters of a single element are obtained. In 1964 Macnaughton-Smith and others proposed splinter average distance method [Macnaughton-Smith et al., 1964]. In 1965 a method based on least squared criterion has been proposed, however this method has a high computational cost due to all possible divisions that had to be computed [Edwards and Cavalli-Sforza, 1965]. In case of binary data Monothetic methods are widely used.
After finding the hierarchy it can be represented through a dendrogram, it is a tree representation of clusters. The units are arranged along the bottom of the dendrogram and referred to as leaf nodes. Clusters are formed by gathering individual units. The vertical axis refers to a distance measure between units or clusters.

Non hierarchical clustering methods, also called partitioning methods, divide a set of n objects in K clusters where K is a priori defined. The best partition is obtained by computing all the possible partitions of the units in K clusters and by finding the one that optimize the chosen criterion. This is often unfeasible. If the dataset is composed by 100 units that have to be divided in 3 clusters, exist 161700 partitions. To calculate all the possible partitions and to compare them, it has a very high computational cost; this issue makes this method unfeasible, especially for large dataset.

To cope with this issue exist three possible strategies: geometric, probabilistic and mixture methods.
Geometric clustering methods are based on measure of distances between units. Using these techniques a measure of adequacy of a partition must be chosen and the aim is to find the partition that optimize the criterion.

Several different adequacy measures have been proposed in literature, one of the most commonly used nonhierarchical clustering method is K-means, that seeks to minimize the variance inside each cluster. For a more detailed development of nonhierarchical geometrical clustering methods refers to chapter 2.

When using probabilistic clustering methods and mixture models, the hypothesis in which a unit is assigned to one and only one cluster is relaxed. In probabilistic clustering methods, a membership function p_{ik} is assigned to each unit, it represents the probability that a generic unit i belongs to a generic cluster k. Two other assumption are entered: the membership function is a number in the interval $[0, 1]$ and the total sum of each unit membership is one (1). The advantage of these type of methods is finding clusters of different shapes. A well known probabilistic method is c-means cluster that minimize the sum of squared errors, like K-means and Ward's linkage criterion [Dunn, 1974], [Bezdek, 1974].

Using mixture models, probabilistic clustering assumptions are respected [McLachlan and Basford, 1988] and one more parametric assumption is done: K subgroups form the population, the proportion of the units that form each subpopulation in the sample is unknown. The goal is to identify the subgroups and to estimate the parameters of the density function underlying each subgroups. Maximum likelihood estimations are used to estimate the vector parameter. Density function of each subgroup must be specified, in many cases the subpopulations are assumed as normally distributed. To maximize the likelihood some complex routines are needed and it is not always possible. To avoid this computational problem, expectation maximization (EM) algorithms are used to estimate mixture models [McLachlan and Krishnan, 2008], for details see section 3.3.

Mixture models are a particular case of a greater set of methods: cluster-wise regression models. According to these models, clusters are considered a subset of the data points that can be modeled by a distribution. Each class has its own reference distribution, consequently a different model can be used for each class [Hennig, 1999]. Cluster-wise regression models can be divided in two subgroups: mixture models and fixed partition models. The main difference of fixed partition is that there are n parameters that specify the cluster membership of each point.

1.1.4 Cluster validation

Cluster analysis is commonly used in exploratory data analysis. The aims of these analysis are: finding underlying structure of data, finding important variables, suggesting hypothesis about the causes of observed phenomena, making assumptions on which statistical inference will be based on and supporting the selection of the appropriate statistical tools and techniques. Exploratory data analysis are usually followed by a post hoc interpretation of results. However, sometimes these techniques can realease inappropriate results. For this reason is necessary a validation step.
The first verification of results is the visualization of the datasets but this is possible only if the dataset has maximum three variables. Otherwise three types of validity criteria can be verified: internal, external and relative. Internal criteria compare the classification with the original dataset. External criteria use the information not used during the classification to test its validity, this validity criteria require a pre-specified structure imposed on the dataset. Relative criteria compare the results obtained with different parameters. The validity can be verified on the whole structure or on a partition or on a single cluster.

Internal validity criteria The null hypothesis in cluster validity is that the units are randomly structured, to test this hypothesis different models can be used. Some models are briefly illustrated afterwords for a more complete list of methods referring to [Gordon, 1999]. A simple model is Random labels model, the basic assumption is that all the permutations of the labels are equally likely. All the $n!$ permutations are computed and a statistical test is done in order to compare the observed value with an statistic computed on the permutations. Another frequently used criterion is the Monte Carlo one. A validity index must be chosen, the density function of this index is computed through Monte Carlo techniques. In order to simulate the density function, a large amount of normally distributed data are simulated, on each of this simulated dataset the value of the index is computed. A density distribution of the index is obtained and it can be used to test whether the obtained index on the original dataset is in the not acceptation zone. In this case the null hypothesis is rejected; the units structured with the clustering technique are not randomly grouped.

External validity criteria External validity criteria can be used only if a pre-structure of data exists. The pre-existing structure is compared with the structure obtained with the clustering techniques. The number of units that belong to the same cluster in both classifications is denoted with a. The number of units that belong to the same cluster using clustering technique but not in the pre-existing classification is denoted with b. The number of units that belong to different clusters using clustering technique but are part of the same cluster in the pre-existing classification, is denoted with c. The

number of points that belong to different clusters in both classifications is denoted with d. Two index can be used to compute the degree of similarity between the two structures: Simple matching, $1 - \frac{a+d}{a+b+c+d}$, and Jaccard coefficient, $1 - \frac{a}{a+b+c}$.
Some other specific index can be found in [Halkidi et al., 2002].

Relative validity criteria The aim of relative clustering validity criteria is to evaluate a clustering structure by comparing it with other structures obtained by the same algorithm with different parameters. Clustering algorithm is run in the same dataset for different number of clusters K between a minimum value K_{min} and a maximum value K_{max}. For each value of the parameter K an index q is obtained. The evaluation of the values of the index q changing K allows to choose the optimal value of K. A wide range of index exist in literature. A well known and often used one is the intra-cluster variance; it measures the homogeneity of the clusters and it is used with non-overlapping clusters. The intra-cluster variance or within variance is the sum of the squared distances between units and the center of the clusters that they belong to.

However each clustering method optimize its own objective function that depends on the criterion optimized by the method. The evaluation of the values of the objective function changing K allows to choose the optimal value of the parameter K.
Dealing with categorical or binary data CATANOVA method [Singh, 1993] can be used to evaluate clustering methods. This method is the generalization of ANOVA method to the case of categorical data.

These indices cannot be used to evaluate different clustering methods because each method optimizes a different criterion and, consequently, a

different objective function. For example, applying two different clustering methods on the same dataset, the partitions l_1 and l_2 are obtained. The partition l_1 is the best according to the first method criterion, the partition l_2 is the best according to the second method criterion. So the values of the objective functions cannot be used to compare two different methods. A different index can be chosen, but the best partition is obtained by applying another clustering method whose criterion optimize the chosen index.

Chapter 2

Non hierarchical clustering methods

This chapter deals with methods to solve the problem of partitioning n objects in K clusters, also known as non hierarchical clustering methods. Defining with X a $n \times J$ data matrix, the aim is to find K clusters of the n objects that are similar inside each cluster and not similar to the objects of other clusters. The way to obtain the best partition would be to investigate all possible partitions of the n objects in K clusters: it is often unfeasible for the very high computational cost. An alternative way is to define a measure of partition adequacy and to find clusters that optimize this measure. In literature a wide range of measure of partition adequacy have been proposed and consequently many non hierarchical clustering methods exist. A lot of these can be brought back to the most used and well known K-means [MacQueen, 1967]. K-means success is due to its simplicity and to its computational advantages; the computational complexity of the method is very low thus it can be easily applied dealing with a large number of units. This

method is based on the variance decomposition and on the minimization (maximization) of the within group (between groups) variance.

2.1 Definition of the number of clusters

Non hierarchical clustering methods aim at obtaining K clusters starting from n units. The number of clusters K can be a priori known, but it is very rare; more often it is a researcher choice. The most commonly used way to choose K is to obtain the partitions from a range of the parameter K, $1 \leq K \leq n$ and to compare the partitions obtained. Dealing with dataset composed by a small number of variables, clustering results can be visualized and compared through a scatter plot. A more formal method consists of finding stopping rules that can be global or local. Global rules use a measure of goodness of a partition $G(K)$ that evaluates the whole partition structure; the purpose is to identify the value of K for witch $G(K)$ is optimal. Global rules are usually based on within-cluster and between-cluster variance, they cannot be used in case of only one cluster consequently they cannot be used to evaluate if the data should be partitioned or not. Local rules are based on a part of the data, their objective is to evaluate if two clusters should be agglomerated or if a cluster must be partitioned; these indeces can be used only with hierarchical clustering, therefore they will be not detailed in this context. In literature a lot of global stopping rules exist [Milligan and Cooper, 1985], the most used are shown in the following formulas. Defining with B the between-cluster sum of squared distances and with W the total within-cluster sum of squared distances of the centroids; the Caliński and Harabasz's index (1974) is defined as:

$$G_1(K) = \frac{B/(K-1)}{W/(N-K)}. \tag{2.1}$$

Goodman and Kruskal in 1954 proposed a modified version of their gamma index γ. It is based on the comparison between all the within-cluster and the between-cluster distances. If a within-cluster distance is strictly shorter than a between-cluster distance their are said concordant and their are indicated with S_+. In the opposite case they are said discordant and they are indicated with S_-; the index is defined as the following formula:

$$G_2(K) = \frac{S_+ - S_-}{S_+ + S_-}. \tag{2.2}$$

A third well known index standardizes the sum of all within-cluster distances, $D(K)$, for a fixed value of K.

$$G_3(K) = \frac{D(K) - min(D(k)}{max(D(k) - min(D(k)}, \tag{2.3}$$

with $k = 1, \ldots, K$.
The value of K that maximizes $G_1(K)$ and $G_2(K)$ or that minimizes $G_3(K)$ is the optimal number of clusters; normally only small values of K are investigated.

2.2 Variance decomposition

A measure of partition adequacy [Edwards and Cavalli-Sforza, 1965] is the within-cluster variance, the idea is to minimize the within-cluster variance and, consequently, to maximize the between-cluster variance. Defining with X the original data matrix, with c_{kj} the center of cluster k, with x_i^k the i^{th} object belonging to cluster k, with n_k the number of elements belonging to the cluster k, where $i = 1, \ldots, N$, $j = 1, \ldots, J$ and $k = 1, \ldots, K$, the within-cluster $J \times J$ variance matrix, W and the between-cluster $J \times J$ variance matrix B are defined with the following formulas:

$$W_k = \sum_{i=1}^{n_k} (x_i^k - c_k)(x_i^k - c_k)', \forall k = 1, \ldots, K;$$

$$W = \sum_{k=1}^{K} W_k \frac{n_k}{n}; \tag{2.4}$$

$$B = \sum_{k=1}^{K} n_k (c_k - \bar{x})(c_k - \bar{x})'; \tag{2.5}$$

where $\bar{x}$ is the global mean. The sum of W and B is the total variance hence to find the centers of clusters that minimize W is equivalent to find the centers of clusters that maximize B. In case of only one cluster the problem consists in finding a point that minimize the distances between n points. In case of more than one cluster the problem becomes more complex because K points must be found; several alternative ways are feasible [Everitt et al., 2011].

An extension of the minimization of the sum of squares criterion is the minimization of the sum of squares of the within groups variances, that is equivalent to minimize the trace of W or equivalently to maximize the trace of B. This criterion was first applied by Ward in the context of hierarchical clustering [Joe and Ward, 1963]. This criterion is used in many clustering methods, it was used explicitly by Singleton and Kautz but it was also implicitly used by Forgy [Forgy, 1965], Jancey [Jancey, 1966], MacQueen [MacQueen, 1967] and Ball and Hall [Ball and Hall, 1967].
To verify whether there is a difference between mean group vectors can be used the ratio between the determinant of the total and within groups variances matrix: $\frac{det(T)}{det(W)}$. This ratio is as larger as mean vectors difference increase, thus a clustering criterion can be the maximization of

this ratio that corresponds to the maximization of the determinant of W [Friedman and Rubin, 1967]. Friedman and Rubin suggested the use of the trace of (BW^{-1}) as an indicator of the differences among mean group vectors.
The most widely used criterion is the minimization of the trace of W even if it is affected by some problems. The minimization is not obtained deriving the objective function, thus can be lead to local minima and not a global one; several iterations of the same method can bring different results. The method is scale dependent and finally this criterion can find clusters of spherical form but it can not identify clusters of arbitrary form. To cope with these problems Friedman and Rubin proposed the criterion previously mentioned.

2.2.1 K-means method

K-means is the most known and used non-hierarchical clustering method thanks to its simplicity. It has been proposed in 1967 by MacQueen and it is based on the minimization of the trace of W.
The optimization problem consists in finding K centers c_{kj} that minimize the trace of W where W is defined in 2.4. The problem can not be solved analytically, and is required an iterative procedure.
The procedure starts with a random initialization of center matrix C, Euclidean distance between each unit of the matrix X and each center C is computed and each unit is assigned to the closer center. According to clustering structure new centers are computed as centroid of each cluster. Distances of units from the new cluster centers are computed and each unit is assigned to the closer center. The algorithm is iterated until obtaining convergence. There are three main convergence criteria:

- two consequently iterations define the same partition;
- inertia reduction is lower than a fixed level ϵ;
- maximum number of iterations is reached.

K-means is structured as: i) choice of K random centers; ii) attribution of each unit to the nearest cluster; iii) computation of centers of the new clusters; iv) iteration of step ii and iii until obtaining convergence.
The method is detailed in the algorithm 1.

Algorithm 1 K-means

function K-MEANS(X, K)
  $C \leftarrow \text{rand}(K, J)$ ▷ Matrix $C_{K,J}$ is randomly initialised
  $D \leftarrow distance(X, C)$ ▷ Initialise the array D as distances of X from C
  $l \leftarrow assing(X, D)$ ▷ $assign(X, D)$ assign each point to the closer center
  $W \leftarrow T$ ▷ Initialise the array W to the maximum
  repeat
    $C \leftarrow C^*$ ▷ Centers are updated as centroid of the clusters
    for $k = 1, K$ **do**
      $D_k \leftarrow distance(X, C(k))$ ▷ D_k distances of all units from the centre k
    end for
    $W0 = W$ ▷ Current W is stored in $W0$
    $l \leftarrow assing(X, D)$ ▷ $assign(X, D)$ assign each point to the closer center
    $W \leftarrow withinvariance(X, l)$ ▷ $withinvariance(X, l)$ implements the 2.4
  until $W0 > W + \epsilon$
  return C, l
end function

K-means requires three user-specified parameters: center initialization, number of clusters K (see section 2.1), and distance metric [Jain, 2009]. The initial choice of centers significantly affects the results. Diday in 1971 proposed the dynamic clouds method in order to cope with this problem [Diday, 1971]. According to this method the initial centers are not a unit but each center is computed as a centroid of q units. However, both meth-

ods allow to find a local optimum and not a global one [Steinley, 2003]. The best way to cluster a dataset using K-means is to perform the method several times on the same dataset and to look for units that are always together in the same cluster.

2.3 K-means advantages

K-means is the most widely used non hierarchical clustering algorithm; it is due to its simplicity and mostly to its computational propriety. In practice K-means converges very fast thus, even if the global convergence is not guaranteed, the method can be iterated several times. Duda et al. state that the number of iterations until the clustering stabilizes is often linear and yet the worst-case running time is exponential [Duda et al., 2000] (Sec. 10.4.3). The computational complexity is of order $O(nKl)$ where n is the number of units, K is the number of clusters, and l is the number of iterations taken by the algorithm to converge. Often, K and l are fixed so the algorithm has linear time complexity according to the size of the dataset. Even the space complexity is small: $O(K + n)$, however it requires additional space to store the data matrix [Jain et al., 1999].
Another advantage, K-means does not need parametric hypothesis, however, in presence of normal distributed clusters, it is still an optimal clustering method.

These conditions make the K-means the most well-known and widely used clustering method.
Under the following conditions K-means is the optimal clustering method:

- small number of variables J ($J < n$);

- orthogonal variables (spherical clusters):
- clusters having the same variance;
- absence of outliers.

2.4 Violation of optimality of conditions

K-means is very simple and very fast, these characteristics make it the most widely used clustering method. Nevertheless this method is not the optimal one dealing with datasets that present some characteristics: large numbers of variables, correlation between variables or clusters having different variance, and in presence of outliers. Moreover, the convergence to a global minimum is not guaranteed and this can cause stability problems.

2.4.1 Clusters stability

K-means has tendency to converge to a local minimum. This issue can significantly affect the results, different iterations of the same method on the same dataset, can bring to different solutions. A part of this is due to the initial random choice of centers because if a center is initialized in an inappropriate position it can not move into an optimal one. In 1997 Fritzke [Fritzke, 1997] proposed a solution to this issue. He introduced a measure of distortion errors; defined with x_0 the center of the Euclidean space and with x_i^k the units belonging to the cluster k with $k = 1, \ldots, K$, the distortion error I_k is:

$$I_k = \sum_{i=1}^{n_k} (x_i^k - x_0)^2 - n_k(c_k - x_0)^2 \tag{2.6}$$

where n_k is the number of elements that belongs to cluster k.
After computing a clustering structure by using K-means, Fritzke proposes to verify if there is a cluster that can be moved into an other position reducing the sum of distortions. If so, the cluster center is moved and K-means method is applyed starting from the new centers. He proposes to jump the center of the cluster with the least distortion error to a random position in the cluster with the most distortion error. When a center is removed all the elements of the cluster will belong to the second nearest cluster, the increment of the total distortion error is equal to the sum of the squared distances between the units of the cluster and the second nearest center. This measure does not consider that the center of the nearest cluster moves when the points are added to the cluster. Pelleg and Moore proposed to start K-means method with a smaller number of clusters K_0, then the number of cluster is doubled by adding other centers in suitable positions [Pelleg and Moore, 2000]. The problems of this method are that each cluster is divided in two parts independently from the others and the method does not guarantee distortion errors minimization. In 2004 incremental K-means has been proposed by Pham and al. [Pham et al., 2004]. The incremental algorithm starts by putting $K = 1$ and increasing K of 1 at each iteration until a fixed value is reached. In each iteration K centers are obtained by using K-means algorithm; when K-means clusters are obtained, the new center is inserted in the cluster with the highest distortion error. In order to take into account the changes of clustering structure, when a cluster is suppressed, the variation of distortion errors are taken into consideration.

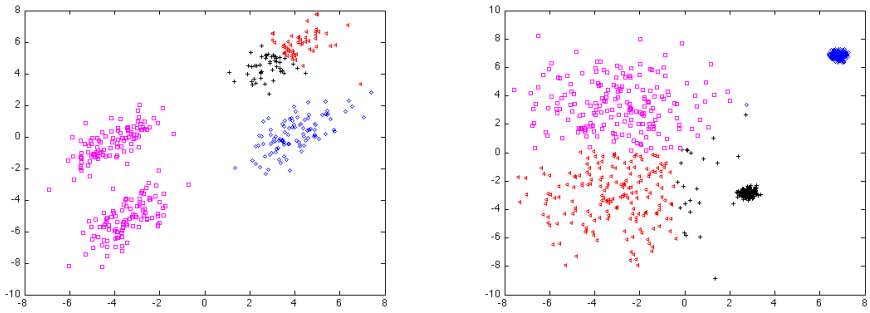
2.4.2 High dimensional data

High dimensional data can be characterized by: large number of units n or large number of variables J, or both. The definition of large and very large have varied (and will constantly continue so) with improvement in technology and computational fields (e.g., memory and processing time). In the 1960s, large meant several thousand patterns. Nowadays, there are applications where millions of patterns of high dimensionality have to be clustered; taking into account the genomic field a dataset can be composed by more than 1 million variables. K-means computational coast is of order $O(nKl)$ where n is the number of units, K the number of clusters and l the number of iterations taken by the algorithm to converge [Jain, 2009]. The number of variables does not affect K-means computational coast, however, the method works well increasing the number of units but it can have stability problem dealing with large number of variables. Several methods have been proposed in order to face this problem, they will be detailed in chapter 3.

2.4.3 Variable correlation

When clusters are spherical K-means is one of the best clustering algorithm thanks to the fastness of the convergence. However K-means can not separate clusters that are non-linearly separable. This can happen dealing with: clusters of arbitrary form, correlation between variables and variance changing among clusters. Fig. 2.4.3 shows two examples of applications of K-means when variables are correlated (a) and variance changes among clusters (b). The figure represent two simulated dataset where four clusters have been normally generated. In the first case, variable correlation, the

method do not catch the right clustering structure, in the second case there are 6% of misclassified.



(a) variables correlated

(b) variance changing

Figure 2.1: Scatter plot of two simulated datasets clustered using K-means

Two approaches have been proposed for tackling such a problem. One is kernel K-means; before clustering, points are mapped to a higher-dimensional feature space using a nonlinear function, and then kernel K-means divides the points by linear separators in the new space. The other approach is spectral clustering algorithms, which use the eigenvectors of an affinity matrix to obtain a clustering of the data [Dhillon et al., 2004]. Kernel characteristics are detailed in chapter 3.5.

2.4.4 Outliers

Outliers are some elements of the datasets that are far from all the others elements. In the univariate case many methods already exist to avoid the effects of outliers, but they can still cause problems in the multivariate case. In cluster analysis application there are two types of outliers: external

outlier, an element of the group that is located far from other groups or internal outlier that compared with the distance of the other data in the same group, is located far from the centroid. In both cases the outliers can cause some issues in cluster analysis. K-means is based on the arithmetic mean, it is known that the arithmetic mean is sensitive to outliers; this issue is reflected on K-means method. A simulated dataset, composed by four normally distributed clusters, has been created by adding 10% of outliers; K-means failed detecting the right cluster structure, see fig. 2.4.4.

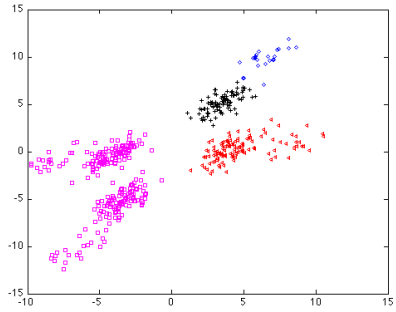


Figure 2.2: Scatter plot of a simulated datasets with 10% of outliers clustered using K-means

Trimmed K-means methods have been proposed to robustify K-means algorithm. In these algorithms outliers are detected and are excluded from the analysis before applying K-means [Cuesta-Albertos et al., 1997] [Ordonez, 2003]. Trimmings depend only on the joint structure of the data and not on arbitrarily selected directions or zones for removing data. For trimming details refer to [Gordaliza, 1991].

2.5 Clusters representation

There is not an unique approach to evaluate clustering partitions, however the quality of the results assesses the ability of the clustering procedure.
A partition can be evaluated by different point of view like: stability, homogenity and separability. A visual representation can help in the evaluation of the clustering structure and for the interpretation of results. A widely used method to visualize a partition is the scatter plot; units are projected on a Cartesian plane and different colors or symbols are used according to the belonging cluster. This method can be used only when dataset have 2 or 3 variables (fig. 4.4). An extension is the splom, where a different scatter plot is built for every pair of variables (fig. B.9). This representation is useful for the interpretation of results, however when the number of variables is large these methods can not be used. Box plots are a valid instrument for the interpretation of results too. For every variable a box plot for each cluster can be built, this allows to evaluate the differences between medians and distributions of each cluster (fig. 4.3).
In order to evaluate the stability of results an evaluating index must be chosen; the value of the index can be easily represented through an histogram or a bar graphic (fig. 4.7).
Evaluating the separability of the clusters is a difficult task. Several evaluating method are based on some measure of distance between objects and clusters. One exploratory tool hanging on this idea is the silhouette information [Rousseeuw, 1987]. The silhouettes can be used only when the proximities are on a ratio scale, in case of K-means method the metric used is the Euclidian one that respect the condition. The objective is to evaluate how compact and separated the clusters are. It is worth to note that for

evaluating the goodness of a partition, the criterion used should be consistent with the clustering method adopted to produce that partition.
To construct the silhouettes two quantities are needed: the partition and the matrix containing the distances between all objects. For each object the quantity $s(i)$ is measured, it depends on $a(i)$ and $b(i)$. Having a generic unit i belonging to cluster A:

$a(i)$ = average dissimilarities between i and all unit belonging to A

$d(i, C)$ = average dissimilarities between i and all unit belonging to $C \neq A$

$$b(i) = \min_{C \neq A} d(i, C). \tag{2.7}$$

The cluster, for which the minimum value of $d(i, C)$ is obtained, is called neighbor of i. The quantity $s(i)$ is computed as the follow formula:

$$s(i) = \frac{b(i) - a(i)}{max(a(i), b(i))}. \tag{2.8}$$

When the cluster A contains one single object $s(i) = 0$. It can be seen that $-1 \leq s(i) \leq 1$. When $s(i)$ is equal to 1, $a(i)$ is much smaller than $b(i)$ the dissimilarity within the cluster is much smaller than the dissimilarity between the clusters, the point is well clustered. When $s(i)$ is equal to 0 the point has the same distance from both clusters, when $s(i) = -1$, $b(i)$ is much smaller than $a(i)$ the dissimilarity between the clusters is much smaller than the dissimilarity within the cluster, i is misclassified. By ranking the silhouettes of each cluster in a decreasing order they can be represented on a plot, fig 2.5 represents silhouettes of the simulated dataset represented in fig. 2.4.3a.

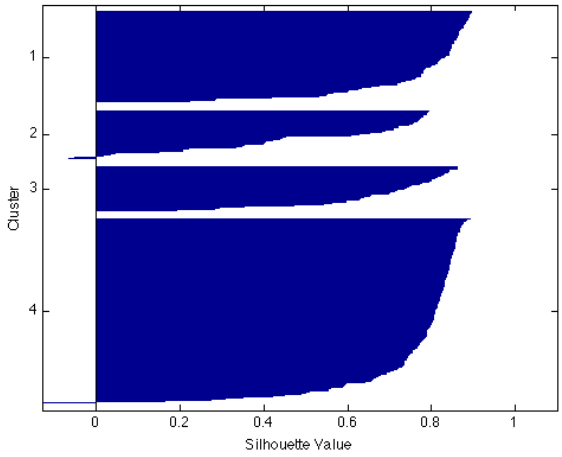


Figure 2.3: Silhouette plot of a simulated dataset clustered using K-means

Chapter 3

Clustering in high dimensional space

Thanks to the ever-growing informatics capability, data storage ability becomes bigger and bigger every time. Computers are largely used in many fields and large databases are more frequent; large can be referred to both units and variables. Classical statistics and data analysis methods work well dealing with a large number of units however they can have problem dealing with a large number of variables. When the aim is to cluster a dataset the number of variables affects the quality of results; each variable can be seen as an axis of the $\Re^J$ data-space, in a J dimensional problem the space can be divided into a number of region equal to 2^J. For every axis a generic point on it allows to obtain two regions, each units can belong to only one region. Taking into account two variables the number of regions becomes 2^2, dealing with three variables the number of regions becomes 2^3. The number of regions increases exponentially as the number of variables increases.

The exponentially relationship between variables and number of regions can cause stability problem even dealing with a relatively small number of variables. Bellman's in 1961 coined the phrase "curse of dimensionality" referring to the exponential growth of hyper-volume (number of distances to compute) as a function of dimensionality (number of variables) [Bellman, 1961]. In many fields the number of variables is even large or very large. A particular example can be obtained looking at genomic datasets where the number of units is even smaller than the number of variables. In genomic datasets the variables are nucleotides, little part of chromosome, and the units are tissues. The number of nucleotides is huge, each chromosome is composed by millions of nucleotides. However, the number of tissues is low because the cost to obtain a chromosome complement is very high. This is an extreme case but datasets with a very large number of variables are frequent in many sectors.

To find information into an high dimensional database it is a very difficult task. Generally clustering methods can have stability problems dealing with large datasets. Several clustering strategies have been proposed in order to cope with this issue. These strategies are based on dimensionality reduction before applying a clustering technique. Furthermore clusters obtained in a lower dimensional space can be easily interpretable and can be graphically represented.

There are two main techniques: features selection and features transformation.

Feature selection attempts to find variables that are more relevant for the objective of the analysis. According to this method some sets of variables are evaluated according to a chosen criterion. At first feature selection was used for supervised learning where variables are selected according to their

closeness with the variable to be predicted. In unsupervised learning there is not a variable to be predicted neither an universally measure of partition adequacy. Parsons and al. [Parsons et al., 2004] proposed some entropy measures adapted for the scope, however a universal criterion does not yet exist. Moreover, the condition to apply a feature selection is that exists a group of few variables that explain the phenomenon under study.
Feature transformation is based on the use of factorial techniques and clustering techniques, these methods are called two-steps clustering methods. Dealing with categorical datasets these methods can have two scopes: dimensionality reduction and variables quantification; a method can pursue the two scope at the same time. To apply a feature transformation two quantities have to be computed: a linear transformation of the dataset and a clustering partition; they cannot be computed at the same time. Two strategies are feasible: two-steps or iterative two-steps methods. Two-steps methods, also called tandem analysis, minimizes two different functions that can be in contrast between them and the first factorial step can in part obscure or mask the clustering structure ([Chang, 1983], [DeSarbo et al., 1990]). This issue can be overcame. De Soete and Carroll in 1994 proposed an alternative method [De Soete and Carroll, 1994]; this method is an alternative K-means procedure, after a clustering phase, it represents the centroids in a lower dimensional space chosen such that the distances between centroids and points belonging to the cluster are minimized. Then all points are projected in this low-dimensional space obtaining a low-dimensional representation of points and clusters. This method can fail in finding the real clustering structure when the data have a lot of variance in directions orthogonal to the one capturing the interesting clustering.

Iterative factorial clustering methods overcome these issues; they perform a factorial step and a clustering step iteratively, optimizing a common criterion.
Iterative methods are used when the direct solution is unfeasible or cannot be computed. These methods attempt to solve a problem by finding successive approximations to the solution, starting from an initial guess.
Factorial clustering methods can be divided in two type: probabilistic and geometric methods.

The chapter is organized as the following description. Firstly two-step methods are introduced. A section is dedicated to clustering of categorical data; it will be done even a hint of clustering methods for categorical data that does not require quantification. After the chapter is focused on iterative two-step methods that can be probabilistic or geometric. In section 3.3 probabilistic factorial clustering methods are detailed, in section 3.4 the attention is focused on geometric ones. Geometric factorial clustering methods description is organized as; at first a factorial methods for quantitative data, Factorial K-means [Vichi and Kiers, 2001] is introduced, after the attention is focused on a factorial methods for qualitative data [Hwang et al., 2006] and at the end a factorial methods for binary data is described [Iodice D'Enza and Palumbo, 2010]. When clusters are characterized by typical structures, like nested clusters, even factorial methods can fail; in this case an efficient strategy is to project points in a feature space. These type of methods are detailed in section 3.5.

3.1 Two-step methods

Non hierarchical methods easily deal with a large number of units n, however they can fail when the number of variables J becomes large or very large, and when the variables are correlated. The converge of these methods, like K-means, has been empirically demonstrated, yet they can converge to a local solution: a different solution at each iteration of the method on the same dataset [Bottou and Bengio, 1995]. To cope with these issues, French school of data analysis [Lebart et al., 1984] suggested a strategy to improve the overall quality of clustering that consists in two phases: variables transformation through a factorial method and application of a clustering method on transformed variables. Arabie and Hubert in 1996 [Arabie et al., 1996] formalized the method and called it *tandem analysis*. Tandem analysis exploits the factor analysis capabilities that consist in obtaining a reduced number of uncorrelated variables which are linear combination of the original ones. This method gives more stability to the results and makes the procedure faster. The choice of the factorial method is an important and tricky phase because it will affect the results. It depends on the type of data [Le Roux and Rouanet, 2004]. The main factorial method for quantitative data is the Principal Component Analysis (PCA) [Hotelling, 1933]. Factorial axes obtained according to PCA are ordered according to the explained variability, so a reduced number of factors can be retained. Several criteria exist to choose the number of factors:

- the minimum number of factors that gives a significant value of the explained variability;
- the number of factors that corresponds to eigenvalues larger than one, otherwise the factor explains less variability than original variables;

- to use a scree-test, choosing the factors that precede the regularization of eigenvalues' histogram.

The second phase of the *tandem analysis* consists in applying clustering methods.
This technique has the advantage of working with a reduced number of variables that are orthogonal and ordered respecting to the borrowed information. Moreover, dimensionality reduction permits to visualize the cluster structure in two or three dimensional factorial spaces [Palumbo et al., 2008].

3.2 Clustering categorical data

Categorical data clustering and classification presents well known issues. Categorical data can be combined forming a limited subspace of data space. This type of data is consequently characterized by non-linear association. Moreover, when dealing with variables having different number of categories, the usually adopted complete binary coding leads to very sparse binary data matrices. Dealing with a large dataset it is necessary to reduce the dimensionality of the problem before applying clustering algorithms. When there are linear associations between variables, suitable transformations of the original variables or proper distance measures allow to obtain satisfactory solutions [Saporta, 1990]. However, when data are characterized by non-linear association, the interesting cluster structure remains masked to these approaches.

There are two main strategies to cope with the clustering in presence of categorical data:

- to transform categorical variables into continuous ones and then to perform clustering on the transformed variables;

- to adopt non-metric matching measures.

It is worth noticing that matching measures become less effective as the number of variables increases.

3.2.1 Quantification

Quantification of categorical variables consists on a transformation of them into continuous ones. Some factorial methods can be used in order to obtain a categorical variables quantification:

- binary data
  - Principal Component Analysis (PCA);
  - Correspondence Analysis (CA)[Benzécri, 1973];
  - Multiple Correspondence Analysis (MCA) [Greenacre, 2007];
- nominal data
  - Multiple Correspondence Analysis (MCA);

One of the advantages of factorial methods is that factorial axes are sorted according to the explained variability; a number of variables lower than the number of original ones can be used without great lost of information. In the case of binary data, dealing with sparseness, few factorial axes can represent a great part of the variability of the data. Techniques for the choice of the number of factors are illustrated in chapter 3.1. In case of categorical data it is worth noticing that inertia rate (percentage of variability explained by each factor) is a pessimistic measure of factors

explicative power, due to disjunctive coding. For this reason Benzécri proposed a corrective factor [Benzécri, 1979] to determinate the effectiveness explained variability of first factors.

PCA is based on the Euclidean distance between points so it can be applied on binary data but it cannot be used dealing with categorical data. Nominal data are characterized by non-linear relations, PCA does not detect this type of relations. Multiple Correspondence Analysis (MCA) permits to combine the categorical variables into continuos variables preserving the non-linear association structure and reducing the number of variables. The main difference between the two methods is that using PCA the linear transformation is applied on the raw data matrix, using MCA on a transformed data matrix. The effect of this transformation is detailed in the following section. Anyhow dimensionality reduction is very useful with both type of data; dealing with categorical data the dimensionality growth as the number of category increase.

The use of MCA at first and the application of a clustering method on the first factorial axes is a widely used approach, there are several example of MCA combined with different clustering techniques ([Green et al., 1988], [Lebart, 1994], [Arimond and Elfessi, 2001], [Marino and Tortora, 2009]).

ACM propriety MCA has some important proprieties.
Defined with:

- X data matrix of elements x_{ij}, with $i = 1, \ldots, n$ and $j = 1, \ldots, J$;
- n_j number of modality for each variable J with $j = 1, \ldots, J$;
- $J^* = \sum_{j=1}^{J} n_j$ total number of modality;

- Z data matrix in disjunctive coding of elements z_{ij}, with $i = 1, \ldots, n$ and $j = 1, \ldots, J^*$;
- $\mathcal{B}$ Burt matrit $\mathcal{B} = Z'Z$;
- $z_{i,j} = 1$ if i have chosen j, if not 0;
- $f_{i,j} = \frac{z_{i,j}}{np}$
- $f_{\cdot,j} = \frac{n_{\cdot,j}}{J^*}$ marginal frequencies of columns.

Matrices X and Z can be represented as:

Table 3.1: Original data matrix X of dimension $n \times J$.

	V_1	...	V_j	...	V_J
i	1		3		3
i'	2		1		2

Table 3.2: Disjunctive coding data matrix Z of dimension $n \times J^*$.

	V_{11}	V_{12}	V_{13}	...	V_{j1}	V_{j2}	V_{j3}	...	V_{J1}	V_{J2}	V_{J3}	
i	1	0	0		0	0	1		0	0	1	
i'	0	1	0		1	0	0		0	1	0	J
	$n_{\cdot,j}$											nJ

Starting from these definitions the distance between two generical points i, i' in the original data space can be computed using χ^2 distance with metric $M = diag(\frac{1}{f_{.j}})$.

$$d^2(i,i')_{\chi^2} = n^2 \sum_{j=1}^{P} \frac{(f_{i,j} - f_{i',j})^2}{f_{.j}} \tag{3.1}$$

Applying Multiple Correspondence Analysis new orthonormal factorial axes are defined, as well as a matrix, F, that contains principal coordinates of rows on the factorial axes. MCA gives $\mathcal{B} = U\Lambda V$.

Defining with s the number of not null eigenvalues, the first s rows of matrix U, of general element $\mu_{\alpha j}$ with $\alpha = 1, \ldots, s$ and $j = 1, \ldots, J$, are taken into account.
Defining with:
f_J^i the vector $f_j^i = \frac{f_{i,j}}{f_{i.}}$;
f_J the vector $f_{.j}$;
$\mu_{\alpha J}$ the vector $\mu_{\alpha j}$;
with $j = 1, \ldots, J^*$. The factorial axes are orthonormal so it can be applied the reconstruction formulas [Le Roux and Rouanet, 2004]:

$$(f_J^i - f_J) = \sum_{\alpha=1}^{s} F_\alpha(i)\mu_{\alpha J} \tag{3.2}$$

The equation (3.2) represents the coordinates of point i on factorial axes. On the factorial axes, distances between points can be computed using Euclidean distance:

$$d^2(i,i')_e = \sum_{\alpha=1}^{s} (F_\alpha(i) - F_\alpha(i'))^2. \tag{3.3}$$

THEOREM 1: The euclidean distance between two genetical points i and i' 3.3 on the factorial axes is equal to the χ^2 distance between i and i' in the original space.

$$d(i,i')_e = d(f_J^i, f_J^{i'})_{\chi^2} \tag{3.4}$$

PROOF: χ^2 distance in the original data space can be defined as:

$$d(f_J^i, f_J^{i'})^2_{\chi^2} = ||f_J^i - f_J^{i'}||^2. \tag{3.5}$$

Starting from the (3.2) the distance between the projection of two generic points i and i' on the factorial axes can be computed:

$$(f_J^i - f_J) = \sum_{\alpha=1}^{s} (F_\alpha(i)\mu_{\alpha J}) \tag{3.6}$$

$$(f_J^i - f_J) - (f_J^{i'} - f_J)) = (f_J^i - f_J^{i'}) \tag{3.7}$$

Using (3.7) it can be said:

$$\begin{aligned} ||f_J^i - f_J^{i'}||^2 = ||\sum_{\alpha=1}^{s} F_\alpha(i)\mu_{\alpha J} - F_\alpha(i')\mu_{\alpha J}||^2 = \\ = \sum_{\alpha=1}^{s} (F_\alpha(i) - F_\alpha(i'))^2 ||\mu_{\alpha J}||^2 + \\ +2 \sum_{\alpha \neq \alpha'} (F_\alpha(i) - F_\alpha(i'))(F_{\alpha'}(i) - F_{\alpha'}(i'))\langle \mu_{\alpha J}\mu_{\alpha' J}\rangle \end{aligned} \tag{3.8}$$

The factorial axes are, for construction, orthogonal and they have an unitary norm, it is equal to say: $||\mu_{\alpha J}||^2 = 1$ and $\langle \mu_{\alpha J}\mu_{\alpha' J}\rangle = 0$ if $\alpha \neq \alpha'$, so (3.8) became:

$$||f_{i,j} - f_{i',j}||^2 = \sum_{\alpha=1}^{s} (F_\alpha(i) - F_\alpha(i'))^2 \tag{3.9}$$

Because (3.3) and (3.9), then (3.4) is verified.

■

χ^2 distance between two generic row profile i and i' in the original space is equal to the usual euclidean distance computed on the not null MCA factors.

3.2.2 Non-metric matching measures

An alternative strategy to cluster categorical data is to adopt non-metric matching measures. However these measures becomes less effective as the number of variables increases, the main object of this thesis are large dataset. Non-metric matching measures can not be used dealing with this type of dataset, so the method is only briefly presented.
According to this approach items they can be grouped depending on the rules they generate. A typical example of rules can be found in the market-basket domain; $A \rightarrow B$ means that if someone buy the set of items A he will probably buys the set of items B. There is a difference between the association rule, that is based on the cooccurrences of the two groups of items and gives asymmetric importance to A and B, and the logical implication $A \Rightarrow B$ or the equivalence $A \Leftrightarrow B$. In a categorical database a large quantity of rules can be found, the objective is to find interesting rules; thus an appropriate measure of interestingness must be chosen in order to filter the rules. There are two types of measures: user driven and data driven. The first type of measures are subjective, the second one objective and are

based on the use of some filters of rules.
Defined a rule as an assertion $A \rightarrow B$ where A and B are two itemsets and $A \cap B = \emptyset$, in order to filter the rules some quality measure must be taken into account [Lenca et al., 2008]:

- the support of $A \rightarrow B$ is the percentage of items that have the value 1 for variables A and B;
- the confidence of $A \rightarrow B$ is the ratio of the number of items that have the value one for variables A and B against the number of items that have the value one for variable A.

Agrawal and al. [Agrawal et al., 1993] proposed a two-step algorithm to cluster data using non-metric matching measures:

1. to find frequent itemsets X (frequency higher than a minimum threshold) by finding all the possible item-sets of size from 2 to n_k;
2. to generate rules from frequent itemsets and filter them with the minimum confidence threshold.

However this method finds a large number of rules; among them strong rules can be found. A wide range of algorithms to select strong rules exist in literature [Lenca et al., 2008]. It is worth noticing that matching measures become less effective as the number of variables increases.

3.3 Probabilistic factorial clustering methods

These methods perform a model based clustering and a dimensionality reduction simultaneously. They can be parametric or non-parametric; among

parametric methods the most used one is the Mixtures of Factor Analyzers (MFA) proposed by McLachlan and Peel in 2000 [McLachlan and Peel, 2000] [McLachlan et al., 2003]. Model based clustering consists on assuming that the data come from several subpopulations. Each subpopulation has its own model, the population is a mixture of the subpopulations. The global model is a finite mixture model, where the observations x_i with $i = 1, \ldots, n$ are assumed to be a sample from a probability distribution with density:

$$p(x_i|K, \theta_K) = \sum_{k=1}^{K} p_k \phi(x_i|a_k); \tag{3.10}$$

where p_k, such that $0 < p_k < 1$ and $\sum_{k=1}^{K} p_k = 1$, is the mixing proportion, $\phi(.|a_k)$ is the parametrized density, a_k is the vector of parameters of the k^{th} cluster and $\theta_k = (p_1, \ldots, p_{K-1}, a_1, \ldots, a_K)$.
The most used density probability is a d-dimensional normal distribution, in this case $a_k = (\mu_k, \Sigma_k)$ where μ_k is the mean of the k^{th} cluster and Σ is the variance and covariance matrix of the k^{th} cluster. The mixture model is an incomplete structure, the corresponding complete structure is:

$$y = (y_1, \ldots, y_n) = ((x_1, u_1), \ldots, (x_n, u_n)); \tag{3.11}$$

$u = (z_1, \ldots, u_n)$, with $u_i = (u_{i1}, \ldots, u_{iK})$, is the membership vector such that $u_{ik} = 1$ if x_i arise from group k.
Model based clustering methods have several advantages [Celeux, 2007]: many versatile or parsimonious models are available and many algorithm available. However, when the number of variables increases the number of parameters to estimate is very high.

For this reason mixtures of factor analyzers (MFA) have been proposed. This method can be viewed as a dimensionally reduced model-based clustering approach.
The basic assumption is that the data are generated according to a factor model with a certain a priori probability. The number of parameters can be consequently reduced and the method can be applied to a higher dimensional dataset [Montanari and Viroli, 2011].
According to MFA the distribution of each J-dimensional vector y_i of the observed data can be modeled with an ordinary factor analysis model with a certain probability π_i:

$$y = \mu_i + \Lambda_i z + e_i, \tag{3.12}$$

where μ_i is a location vector of length J, z is a q-dimensional vector of common latent variables or factors, with $q < J$, and Λ_i is a $J \times q$ matrix of factor loadings. The factors z are assumed to be distributed as a standardized q-dimensional Gaussian, $\phi(q)(0, I_q)$, independently from the errors e_i, which are distributed as a J-dimensional Gaussian, $\phi(p)(0, \Psi_i)$, where Ψ_i is a diagonal matrix. The density of the observed data is given by:

$$f(y;\theta) = \sum_{i=1}^{K} \pi_i \phi^{(J)}(y; \mu_i, \Lambda_i \Lambda_i' + \Psi_i), \tag{3.13}$$

where $\theta = (\pi', \theta_1' \ldots, \theta_K')'$. The weights vector π, of generic element π_i, is a $K-1$ dimensional vector and $\theta_i = (\mu_i, vec\Lambda_i, diag\Psi_i)$, with $i = 1, \ldots, K$. The method can be seen as a globally nonlinear latent variable model, K factor models with Gaussian factors are fitted on the data.
The total number of parameters to be estimated is lower than the number

of parameters needed in the estimation of an unconstrained model.
In an unconstrained model the number of parameters to be estimated is: $\frac{1}{2}J(J+1)$ for each component. The number of parameters quadratically increases as J increases, using MFA the number of parameters is: $KJ(q+1)$. In order to successfully perform an MFA q must be chosen lower than $\frac{J-1}{2}$.

The weight π_i are the a priori probabilities of belonging to each subpopulation. Bayes theorem can be used to compute the posterior probability $\tau_{ij}(y_i|\theta)$:

$$\tau_{ij}(y_j|\theta) = \frac{\pi_i \phi^{(J)}(y_j; \mu_i, \Lambda_i \Lambda_i' + \phi_i)}{\sum_{k=1}^{K} \pi_k \phi^{(J)}(y_j; \mu_k, \Lambda_k \Lambda_k' + \phi_k)} \tag{3.14}$$

The estimation of the whole model is obtained using an Expectation Maximization (EM) algorithm, readers interested in algorithm details can refer to [Montanari and Viroli, 2011],

3.4 Geometric factorial clustering methods

A first contribution to geometric two-step iterative methods was given by Vichi and Kiers [Vichi and Kiers, 2001] with Factorial K-means analysis for two-way data. Related methods have been developed to cope with categorical and binary data. Some relevant methods are: multiple correspondence analysis for identifying heterogeneous subgroups of respondents [Hwang et al., 2006] and Iterative Factorial Clustering of Binary Data [Iodice D'Enza and Palumbo, 2010].

All the methods listed above require the choice a priori of the number R of factors and the number K of clusters. A way to choose the number of factor is to apply once the factorial technique used in the factorial clus-

tering method and to choose the number of factors using the techniques illustrated in chapter 3.1 of the current thesis. To chose the number of clusters a possible way is to apply the factorial clustering method with different input values of K, for a fixed value of R, and to choose the best value using the techniques described in chapter 2.1. It is recommended that the number of clusters is greater than the number of dimensions [Van Buuren and Heiser, 1989]. A widely used non-statistical criterion is to evaluate the usefulness and relevance of clusters.

3.4.1 Factorial K-means

The aim of this method is to identify the best partition of the objects and to find a subset of factors that best describe the classification according to the least squares criterion. An alternating Least Squares (ALS) two-step algorithm solves this problem.
Let's define it with:

- X a data matrix with n units and J variables;
- A a $J \times R$ orthonormal matrix, whose elements are a linear combination of observed variables, R is the number of factor to be used;
- U a $n \times K$ matrix, of generic element u_{ij}, that represents the binary membership to each cluster, with K number of clusters;
- E a $n \times J$ error components matrix;
- $\bar{Y}$ a $K \times R$ centroids matrix.

Factorial K-means model is defined as:

$$XAA' = U\bar{Y}A' + E \tag{3.15}$$

The model is an orthogonal projection of units on a subspace spanned by the columns of the orthonormal matrix A. The objective is to minimize f according to A, where$f$ is:

$$f = ||XA - U\bar{Y}||^2. \tag{3.16}$$

The minimization problem is:

$$\begin{aligned} &\min_A ||XA - U\bar{Y}||^2 \\ &\text{subject to} \qquad A'A = I \\ &\qquad\qquad\quad \sum_{j=1}^{J} u_{ij} = 1 \ \forall i \end{aligned} \tag{3.17}$$

Centroids matrix $\bar{Y}$ can be expressed as:

$$\bar{Y} = (U'U)^{-1}U'XA, \tag{3.18}$$

so the expression minimized in the (3.16) becomes:

$$f = ||XA - U(U'U)^{-1}U'XA||^2.$$

This quantity can be decomposed in two parts and the function to be minimized becomes:

$$\min[tr(A'X'XA) - tr(A'X'U(U'U)^{-1}U'XA)]. \tag{3.19}$$

The first component of the (3.19) is the total deviance of XA, the second one is the between classes deviance. An ALS algorithm can be used to solve the (3.19) [de Leeuw et al., 1976]; the algorithm minimizes the object function in two steps. In the first step U is computed applying K-means

algorithm; in the second step the value of A that minimizes the (3.19), given U, is found. The value of A is obtained by finding the first R eigenvectors of B:

$$B = X'(U(U'U)^{-1}U' - I_n)X, \tag{3.20}$$

where I_n is an n dimensional identity matrix.
The algorithm is detailed in Algorithm 2.
The advantage of Factorial K-means is that the two steps optimize a single

Algorithm 2 Factorial K-means

function FACTORIAL K-MEANS(X, K, R)
  $U \leftarrow rand(n, K)$ ▷ matrix $U_{n,K}$is randomly initialised
  $A \leftarrow rand(J, R)$ ▷ Matrix $A_{J,R}$ is randomly initialised
  $\bar{Y} \leftarrow centroids(U, A)$ ▷ *centroids(U,A)* implements the 3.18
  $f \leftarrow max$ ▷ initialise f to the maximum
  repeat
    $U \leftarrow K - means(A, K)$ ▷ U is computed according to algorithm 1
    $B \leftarrow b(X, U)$ ▷ $b(X, U)$ implements the 3.20
    $A \leftarrow svd(B)$ ▷ $svd(B)$ take the first R eigenvectors of B
    $\bar{Y} \leftarrow centroids(U, A)$ ▷ *centroids(U,A)* implements the 3.18
    $f_o \leftarrow f$ ▷ Current f is stored in f_o
    $f \leftarrow f(A, U, \bar{Y})$ ▷ $f(A, U, \bar{Y})$implements the 3.16
  until $f < f_o$
  return A, U
end function

objective function. Another advantage is that clustering method is applied on factorial axes. If the number of chosen factors is lower than the number of variables the method is external robust [Vichi and Saporta, 2009].

3.4.2 Multiple correspondence analysis for identifying heterogeneous subgroups of respondents

In 2006 Hwang and al. proposed to combine MCA and K-means in a unified framework [Hwang et al., 2006]. K-means is easily compatible with the MCA's distribution free optimization problem. The optimization criteria of the two methods are combined creating a single criterion.

Let's define it with:

- Z a $n \times J^*$ disjunctive coding data matrix;
- F a $n \times R$ matrix of R-dimensional representation of J categorical variables, R is the number of factor to be used;
- U a $n \times K$ matrix, of generic element u_{ij}, that represents the binary membership to each cluster, with K number of clusters;
- $\bar{Y}$ a $K \times R$ centroids matrix;
- W_j a $J^* \times R$ matrix of weights, with $j = 1, \ldots, J$;
- α_1, α_2 non-negative scalar weights.

The object of the method is to find a low-dimensional representation of variables and, at the same time, to identify clusters of respondents that are relatively homogenous in the low-dimensional representation. This is equivalent to:

$$\min_{U,W} \alpha_1 \sum_{j=1}^{J} SS(F - Z_j W_j) + \alpha_2 SS(F - U\bar{Y}) \qquad (3.21)$$

$$\text{subject to} \quad F'F = I$$

where $\alpha_1 + \alpha_2 = 1$ and SS indicates the sum of squares of $(F - Z_j W_j)$. When $\alpha_1 = 1$ the criterion is equal to the standard criterion of MCA; when $\alpha_2 = 1$ the criterion is equal to the standard criterion of K-means. When both α_1 and α_2 are non null, minimizing the criteria is equivalent to find a low-dimensional representation of data F such data the cluster structure is recognized. The values of α_1 and α_2 are a priori chosen, if $\alpha_1 = \alpha_2 = 0.5$ MCA and K-means are balanced. If someone wants to give more weight to one of two methods, $\alpha_1 \neq \alpha_2$. To minimize the (3.21) F, W_j, U and $\bar{Y}$ must be estimated; an ALS algorithm is used [de Leeuw et al., 1976]. At first F and U are fixed; W_j and $\bar{Y}$ are updated:

$$\hat{W_j} = (Z_j' Z)^{-1} Z_j' F, \tag{3.22}$$

$$\hat{\bar{Y}} = (U'U)^{-1} U' F. \tag{3.23}$$

The first step consists in the estimation of F. Let's define it with: $\omega_j = Z_j (Z_j' Z)^{-1} Z_j'$ and $\Psi = U(U'U)^{-1}U'$, the value of F that optimize the (3.21) is obtained as eigenvalues decomposition of:

$$\alpha_1 \sum_{j=1}^{J} \Omega_j + \alpha_2 \Psi. \tag{3.24}$$

In the second step U is computed using K-means algorithm, for given values of F, W_j and $\bar{Y}$. The algorithm is summarized in Algorithm 3.
Using the ALS algorithm the value of 3.21 monotonically decreases; however, the convergence to a global minimum is not guaranteed. In particular the K-means is sensitive to local minimum. A strategy to overcome this issue consists in running the all algorithm several times starting from different initial parameters.

Algorithm 3 MCA for identifying heterogeneous subgroups of respondents

function MCA K-MEANS($Z, K, R, \alpha_1, \alpha_2$)
  $U \leftarrow rand(n, K)$ ▷ matrix $U_{n,K}$is randomly initialized
  $F \leftarrow rand(n, R)$ ▷ Matrix $F_{n,R}$ is randomly initialized
  $\bar{Y} \leftarrow centroids(U, F)$ ▷ *centroids(U,F)* implements the 3.23
  for $j = 1, J$ **do**
    $W_j \leftarrow weight(Z, F)$ ▷ $weight(Z, F)$ implements the 3.22
  end for
  $f \leftarrow max$ ▷ initialise f to the maximum
  repeat
    $B \leftarrow b(\Omega_j, \Psi)$ ▷ $b(\Omega, \Psi)$ implements the 3.24
    $F \leftarrow svd(B)$ ▷ $svd(B)$ take the first R eigenvectors of B
    $U \leftarrow K - means(F, K)$ ▷ U is computed according to algorithm 1
    $\bar{Y} \leftarrow centroids(U, F)$ ▷ *centroids(U,F)* implements the 3.23
    for $j = 1, J$ **do**
      $W_j \leftarrow weight(Z, F)$ ▷ $weight(Z, F)$ implements the 3.22
    end for
    $f_o \leftarrow f$ ▷ Current f is stored in f_o
    $f \leftarrow f(F, U, W_j, \bar{Y})$ ▷ $f(A, U, W_j \bar{Y})$implements the 3.21
  until $f < f_o$
  return F, U
end function

3.4.3 Iterative Factorial Clustering of Binary data

Iodice D'Enza and Palumbo in 2010 [Iodice D'Enza and Palumbo, 2010] proposed a new factorial clustering method, iterative Factorial Clustering of Binary data (i-FCB), in order to cope with clustering of high-dimensional binary data. Dealing with this type of data the main issue is the sparseness. Sparsity in high-dimensional binary databases causes two main effects: cluster solutions are unstable and they consist of a single large group and further very small residual groups. The i-FCB find stable clusters of binary data.

Let's define with:

- X a $n \times 2J$ disjunctive coded binary data matrix;
- X_j with $j = 1, \ldots, J$ binary attributes;
- $\mathbf{U}$ a $n \times K$matrix of binary membership to each cluster, with K number of clusters;
- $H = \mathbf{U}^T X$ a $K \times J$ matrix of general element h_{kj} frequency of the j^{th} attribute in the k^{th} group;
- x_j the general column vector of X with $j = 1, \ldots, J$.

The aim of i-FCB is to seek to organize information about variables so that K homogeneous groups of statistical units can be formed. Each of the group is coded via an indicator variable. Thus there will be K of such indicators U_k, $k = 1, \ldots, K$, with $U_k = 1$ if the i^{th} statistical unit is in the group k, $U_k = 0$ otherwise.
Considering n trials, the random vector $\mathbf{U} = (\mathbf{U}_1, \mathbf{U}_2, \ldots, \mathbf{U}_K)$ follows a multinomial distribution with parameters $(n; \pi_1, \pi_2, \ldots, \pi_K)$, with $\pi_k = Pr(U_k = 1)$. Assuming the parameters $(\pi_1, \pi_2, \ldots, \pi_K)$ to be known, the

optimal criterion for the allocation of the n statistical units into the K groups is to randomly assign units to groups proportionally to the corresponding π_k parameters.
The aim of i- FCB is to estimate π_k in order to maximize the heterogeneity among groups with respect to the X_j attributes. Each of the attributes X_j is Bernoulli distributed (with x indicating success and $\bar{x}$ failure) with parameter π_X . According to the same criterion, new statistical units are processed in order to update both the clustering solution and the binary attribute quantifications.
For the sake of simplicity the criterion maximization is illustrated in the simple case of one single binary attribute X, and its generalization to the J attributes case afterwards. Consider $\mathbf{U}$ as a qualitative variable with $k = 1, \ldots, K$ categories and X a binary variable with attributes $\{x, \bar{x}\}$.

Let H be a cross-classification table with general element h_{kl} being the co-occurence number of the categories k and l, with $l = 1, 2$; the row margin h_{k+} is the number of occurrences of the category k ($k = 1, \ldots, K$) and the column margin h_{+l} is the number of occurrences of the category l ($l = 1, 2$), with $h_{++} = n$ being the grand total of the table. The qualitative variance, or heterogeneity, of U can be defined by the Gini index

$$G(U) = 1 - \sum_{k=1}^{K} \left(\frac{h_{k+}}{n} \right)^2 = 1 - \sum_{k=1}^{K} \frac{h_{k+}^2}{n^2}. \tag{3.25}$$

Within the category of x (the same occurs for $\bar{x}$) the variation is

$$G\left(U \mid x\right) = 1 - \sum_{k=1}^{K} \frac{h_{k1}^2}{h_{+1}^2}. \tag{3.26}$$

The variation of Z within the categories of the variable X is obtained by

averaging $G\,(U \mid x)$ and $G\,(U \mid \bar{x})$ and it is denoted by $G\,(U \mid X)$, formally

$$G(U \mid X) = \sum_{l=1}^{2} \frac{h_{+l}}{n} \left(1 - \sum_{k=1}^{K} \frac{h_{kl}^2}{h_{+l}^2}\right) = 1 - \frac{1}{n} \sum_{k=1}^{K} \sum_{l=1}^{2} \frac{h_{kl}^2}{f_{+l}}. \tag{3.27}$$

Then the variation of U explained by the categories of X is

$$\begin{aligned} G(U) - G(U \mid X) &= 1 - \sum_{k=1}^{K} \frac{h_{k+}^2}{n^2} - \left(1 - \frac{1}{n} \sum_{k=1}^{K} \sum_{l=1}^{2} \frac{h_{kl}^2}{h_{+l}}\right) = \\ &= \frac{1}{n} \sum_{k=1}^{K} \sum_{l=1}^{2} \frac{h_{kl}^2}{h_{+l}} - \frac{1}{n} \sum_{k=1}^{K} \frac{h s_{k+}^2}{n}. \end{aligned} \tag{3.28}$$

The groups heterogeneity corresponds to the qualitative variance between the K levels of the Z variable, then with n statistical units described by J binary attributes $X_1, X_2, \ldots, X_j, \ldots, X_J$, the quantity to maximize is

$$\sum_{j=1}^{J} (G(U) - G(U \mid X_j)). \tag{3.29}$$

This expression represents the sum of variances explained by each of the attributes X_j and it is the generalization of expression 3.28 to the $J-$attributes case gives the expression 3.29.
The algebraic formalization of the 3.29 is:

$$\begin{aligned} tr\left[\frac{1}{n} H(\Delta)^{-1} H^T - \frac{J}{n^2}(U^T 1 1^T U)\right] = \\ tr\left[\frac{1}{n} U^T X(\Delta)^{-1} X^T U - \frac{J}{n^2}(U^T 1 1^T U)\right], \end{aligned}$$

where $\Delta = diag(X^T X)$, 1 is a n-dimensional vector of ones.
The solution is obtained solving the following equation:

$$\frac{1}{n}\left[U^T X(\Delta)^{-1} X^T U - \frac{J}{n}(U^T 1 1^T U)\right] V = \Lambda V, \tag{3.30}$$

where V and U are unknown.
The problem corresponds to a classical eigenanalysis problem where the diagonal matrix Λ contains eigenvalues while the matrix V contains eigenvectors. A direct solution is unfeasible, an iterating strategy can be used to compute V and U alternatively. Given the matrix ψ, such that:

$$\psi = \left(X(\Delta)^{-1} X^T - \frac{J}{n} 1 1^T\right) U V \Lambda^{1/2}, \tag{3.31}$$

the eigenvalue decomposition of ψ permits to compute the values of V and U.

The algorithm can be summarized as shown in Algorithm 4.

Algorithm 4 i-FCB

function I-FCB(X, K)
  $U \leftarrow rand(n, K)$ ▷ matrix $U_{n,K}$is randomly initialised
  $f \leftarrow 0$ ▷ initialize f to the minimum
  repeat
    $\Delta \leftarrow diag(X^T X)$
    $\psi \leftarrow svd(X, \Delta, U)$ ▷ $svd(X, \Delta, U)$ implements the 3.31
    $U \leftarrow K - means(\psi, K)$ ▷ U is computed according to algorithm 1
    $f_o \leftarrow f$ ▷ Current f is sored in f_o
    $f \leftarrow f(X, Z, U, \Lambda)$ ▷ $f(X, Z, U, \Lambda)$ implements the 3.30
  until $f_o < f$
  return U
end function

3.5 Clustering in a feature space

The methods presented before solves a lot of clustering problems; however real dataset are characterized by non linear relations that sometimes remain invisible to these methods too. The non linearity is frequently dealing with categorical dataset. In order to face this problem a used approach consists in projecting points in a higher dimensional feature space where the relations between variables are linearized. Kernel functions are used in order to project points in the feature space.

3.5.1 Feature space

A feature space $F = \{\phi(x)|x \in X\}$ is an abstract t-dimensional space where each statistical unit is represented as a point. Being given data matrix X, units $\times$ variables, with general term x_{ij}, $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, J$, any generic row or column vector of X can be represented into a feature space using a non linear mapping function. Formally, the generic column (row) vector x_j (x_i') of X is mapped into a higher dimensional space F trough a function

$$x = (x_1, \ldots, x_n) \rightarrow \varphi(x_j) = (\phi_1(x_j), \phi_2(x_j), \ldots, \phi_t(x_j)),$$

with $t > J$ ($t > n$ in the case of row vectors) and $t \in n$, see fig. 3.5.1 [Muller et al., 2001].
If the feature space has been properly chosen data points, that are not linearly separable in the original space, it can be linearly separable.
This transformation can be used for both: reducing or enlarging the original data space. An example of second degree function $\varphi(x)$ that project points from a two dimensional space to a three dimensional space is: $(x_1, x_2) \rightarrow$

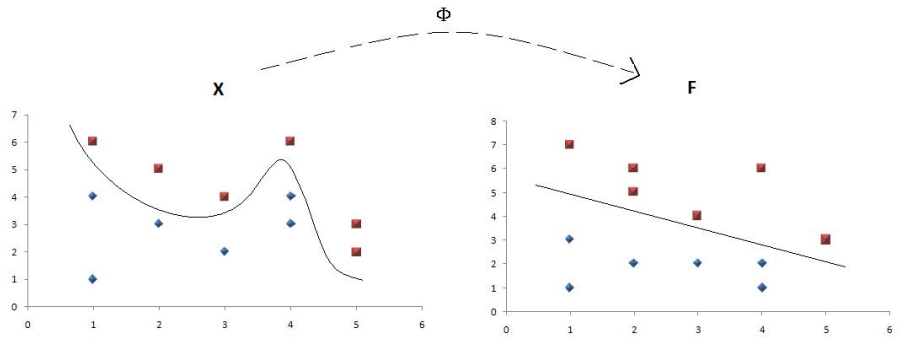


Figure 3.1: Projection of original data point X in the feature space through function $\varphi(x)$.

$\varphi(x) = (x_1^2, x_2^2, x_1 x_2)$. Points can be projected in a generical $\begin{pmatrix} n+d-1 \\ d \end{pmatrix}$ dimensional space by using a d^{th} degree function.
When points projection objective is to cluster , according to a support vector approach, the computation of the function $\varphi(\cdot)$ is unnecessary, the function only appears in products. The dot products $\varphi(x_j) \cdot \varphi(x_{j'})$ can be computed using an appropriate kernel function $K(x_j, x_{j'})$.
A Kernel function is a function of K, so that for every $x, y \in X$

$$K(x, y) = \langle \varphi(x) \cdot \varphi(y) \rangle . \tag{3.32}$$

3.5.2 Kernel functions

Kernel functions permit to reduce the computational cost because the feature space must be only implicitly defined. Thanks to the use of kernel function the function $\varphi(x_i)$ and its inner product $\varphi(x_j) \cdot \varphi(x_{j'})$ do not have to be computed.

The inner product can be defined as:

$$K(x,y) = \langle \varphi(x) \cdot \varphi(y) \rangle = \sum_{n=1}^{\infty} a_n^2 \varphi_n(x) \varphi_n(y). \tag{3.33}$$

It is worth noticing that not all the type of function define a feature space or equivalently the kernel function. Necessary but not sufficient conditions so that $K(x,y)$ is a kernel function are:

- symmetry

$$k(x,y) = \langle \varphi(x) \cdot \varphi(y) \rangle = \langle \varphi(y) \cdot \varphi(x) \rangle = K(y,x)$$

- Cauchy - Schwarz inequality

$$k(x,y)^2 = \langle \varphi(x) \cdot \varphi(y) \rangle^2 \leq \|\varphi(x)\|^2 \|\varphi(y)\|^2 =$$

$$= \langle \varphi(x) \cdot \varphi(x) \rangle \langle \varphi(y) \cdot \varphi(y) \rangle = K(x,x)K(y,y)$$

A sufficient condition to define a kernel function is given by Mercer theorem: In the simplest case when $K(x,y)$ is a simmetric function defined on $X = (x_1, \ldots, x_n)$, K is a matrix as:

$$K = (K(x_i, x_j))_{i,j=1}^n. \tag{3.34}$$

If K is simmetric than it exists an orthogonal matrix V such that $K = V \Lambda V'$, where Λ is a diagonal matrix of eigenvalues λ_t of K corresponding to eigenvectors $v_t = (v_{ti})_{i=1}^n$ columns of matrix V. Assuming that the eigenvalues are nonnegative:

$$\varphi(x) : x \to (\sqrt{\lambda_t} v_{ti})_{t=1}^n \in \Re^n, i = 1, \ldots, n. \tag{3.35}$$

then:

$$\langle \varphi(x_i) \cdot \varphi(x_j) \rangle = \sum_{t=1}^{n} \lambda_t v_{ti} v_{tj} = (V \Lambda V')_{ij} = k_{ij} = K(x_i, x_j) \tag{3.36}$$

$K(x_i, x_j)$ is the kernel function corresponding to $\varphi(x)$. Eigenvectors must be positive, assuming by contradiction that λ_s is negative than it will exist a point z such that:

$$z = \sum_{t=1}^{n} v_{si} \varphi(x_i) = \sqrt{\Lambda V'} v_s \tag{3.37}$$

The norm of z in the feature space is:

$$\|z\|^2 = \langle z \cdot z \rangle = v_s'(V)\sqrt{\Lambda}\sqrt{\Lambda}(V)' v_s = v_s'(V)\Lambda(V)' v_s = v_s' k v_s = \lambda_s < 0$$

it is impossible. Thus it can be said that:
Given a finite input space X and a simmetric function $K(x, y)$ of X, $K(x, y)$ is a kernel function if and only if:

$$K = (K(x_i, x_j))_{i,j=1}^{n}$$

is positive semidefined (eigenvectors are not negative).
Most used Kernel functions are [Abe, 2005]:

- linear: $K(x_i, x_j) = \langle x_i \cdot x_j \rangle$
- polinomial: $k(x_i, x_j) = (\langle x_i \cdot x_j \rangle + 1)^d$ con $d \in \mathbb{N}$ e $d \neq 0$
- gaussian: $k(x_i, x_j) = exp(-q \|x_i - x_j\|^2 / 2\sigma^2)$

Distances in the feature space Through kernel functions the inner product of $\varphi(x)$ and the distance between two generic points in the space can be computed without computing $\varphi(x)$
Defined with P the generic point in the feature space F defined by $\varphi(x)$:

$$P = \sum_{i=1}^{l} \alpha_i \varphi(x_i) = (\alpha_i, x_i)_{i=1}^{l}. \tag{3.38}$$

Defined with $Q = (\beta_i, z_i)_{i=1}^{l}$ a point in the same space; the distance between the two points is defined as follow:

$$\begin{aligned} \|P - Q\|_F^2 &= \\ &= \langle P - Q \cdot P - Q \rangle \\ &= \sum_{i,j} \alpha_i \alpha_j K(x_i, x_j) - 2 \sum_{i,j} \alpha_i \beta_j K(x_i, z_j) + \sum_{i,j} \beta_i \beta_j K(z_i, z_j). \end{aligned} \tag{3.39}$$

3.5.3 Support Vector Clustering

Support Vector Clustering (SVC) is a non parametric cluster method based on support vector machine that maps data points from the original variable space to a higher dimensional feature space trough a proper kernel function (Muller et *al.*, 2001).
The solution of the problem implies the identification of the minimal radius hypersphere that includes the images of all data points; points that are on the surface of the hypersphere are called *support vectors*. In the data space the support vectors divide the data in clusters. The problem is to minimize the radius under the constraint that all the points belong to the hypersphere: $r^2 \geq \|\varphi(x_j) - a\|^2 \quad \forall j$, where a is the center of the hypersphere and $\|\cdot\|$ denotes tha Euclidean norm.

To avoid that only the most far point determines the solution, slack variables $\xi_j \geq 0$ can be added:

$$r^2 + \xi_j \geq \|\varphi(x_j) - a\|^2 \quad \forall j.$$

This problem can be solved by introducing the Lagrangian:

$$L(r, a, \xi_j) = r^2 - \sum_j (r^2 + \xi_j - \|\varphi(x_j) - a\|^2)\beta_j - \sum_j \xi_j \mu_j + C \sum_j \xi_j \quad (3.40)$$

where $\beta_j \geq 0$ and $\mu_j \geq 0$ are Lagrange multipliers, C is a constant and $C \sum_j \xi_j$ is a penality term. To solve the minimization problem we set to zero the derivate of L with respect to r, a and ξ_j and we get the following solutions:

$$\begin{aligned} \sum_j \beta_j &= 1 \\ a &= \sum_j \beta_j \varphi(x_j) \\ \beta_j &= C - \mu_j \end{aligned}$$

We remind that Karush-Kuhn-Tucker complementary condition implies:

$$\begin{aligned} &\xi_j \mu_j = 0 \\ &(r^2 + \xi_j - \|\varphi(x_j) - a\|^2)\beta_j = 0 \end{aligned}$$

The Lagrangian is a function of r, a and μ_j. Turning the Lagrangian into the more simple Wolfe dual form, which is a function of the variables β_j, we obtain:

$$W = \sum_j \varphi(x_j)^2 \beta_j - \sum_{j,j'} \beta_j \beta_{j'} \varphi(x_j) \cdot \varphi(x_{j'}) \quad \forall \{j, j'\}, \quad (3.41)$$

with the constraints $0 \leq \beta_j \leq C$.
It is worth noticing that in (3.41) the function $\varphi(\cdot)$ only appear in products. The dot products $\varphi(x_j) \cdot \varphi(x_{j'})$ can be computed using an appropriate kernel function $K(x_j, x_{j'})$. The Lagrangian W is now written as:

$$W = \sum_j K(x_j, x_j)\beta_j - \sum_{j,j'} \beta_j \beta_{j'} K(x_j, x_{j'}). \tag{3.42}$$

The SVC problem requires the choice of a kernel function. The choice of the kernel function remains a still open issue (Shawe-Taylor and Cristianini 2004). There are several proposal in the recent literature:

- *Linear Kernel* ($k(x_i, x_j) = \langle x_i \cdot x_j \rangle$);
- *Gaussian Kernel* ($k(x_i, x_j) = exp(-q \left\| x_i - x_j \right\|^2 / 2\sigma^2)$);
- *polynomial Kernel* ($k(x_i, x_j) = (\langle x_i \cdot x_j \rangle + 1)^d$.

These Kernel functions (with $d \in \mathbb{N}$ and $d \neq 0$) are among the most largely used functions. In the present work we adopt a polynomial kernel function; the choice was based on the empirical comparison of the results (Abe 2005). Most important for the final clustering result is the choice of the parameter d because this parameter affects the number of clusters.
To simplify the notation, we indicate with $K^*(\cdot)$ the parametrised kernel function: then in our specific context the problem consists in maximising the following quantity with respect to β

$$W = \sum_m K^*(y_m, y_m)\beta_m - \sum_{m,m'} \beta_j \beta_{m'} K^*(y_m, y_{m'}), \tag{3.43}$$

where y_m represents the generic coordinate obtained via MCA, $1 \leq m \leq q$. This involves a quadratic programming problem solution, the objective

function is convex and has a globally optimal solution (Ben–Hur et *al.*, 2001).
The distance of the image of each point in the feature space and the center of the hypersphere is:

$$R^2(y) = \|\varphi(y) - a\|^2 \tag{3.44}$$

Applying previous results the distance is obtained as:

$$R^2(y) = K^*(y,y) - 2\sum_j K^*(y_j,y)\beta_j + \sum_{j,j'} \beta_j\beta_{j'}K^*(y_j,y_{j'}). \tag{3.45}$$

Points, whose distance from the surface of the hypersphere is less than ξ, are the support vectors and they define a partition of the feature space. These points are characterized by $0 < \beta_i < C$; points with $\beta_i = C$ are called bounded support vectors and they are outside the feature-space hypersphere. If $\beta_i = 0$ the point is inside the feature-space hypersphere. The number of support vectors affects the number of clusters, as the number of support vectors increases the number of clusters increases. The numbers of support vectors depend on d and C: as d increases the number of support vectors increases because the contours of the hypersphere fit better the data; as C decreases the number of bounded support vectors increases and their influence on the shape of the cluster contour decreases.
The (squared) radius of the hypersphere is:

$$r^2 = \{R(y_i)^2 | y_i \text{ is a support vector}\}. \tag{3.46}$$

Cluster labeling The last clustering phase consists in assigning the points projected in the feature space to the classes. It is worth reminding that the analytic form of the mapping function $\varphi(x) = (\phi_1(x), \phi_2(x), \ldots, \phi_t(x))$

is unknown, so that computing points coordinates in the feature space is an unfeasible task. Alternative approaches permit to define points memberships without computing all coordinates. Traditional SVC cluster labeling algorithm is Complete Graph (CG) [Ben-Hur et al., 2001]. Given two generic points x_i and x_j, where $\overline{x_i x_j}$ is the line segment connecting the two points, the two points belong to the same cluster if, for all point y belonging to the segment $\overline{x_i x_j}$, the projection on the feature space $\varphi(y)$ belong to the hypersphere. if it exists at least one point of the segment whose projection falls outside the hypersphere the two points belong to different clusters. Starting from this principle an adjacency matrix A, of general element a_{ij}, can be defined.

$$a_{ij} = \begin{cases} 1 & \text{if, } \forall y \text{ belonging to } \overline{x_i x_j} \text{ , } R(y) > R \\ 0 & \text{otherwise} \end{cases} \tag{3.47}$$

In practice points y are a sample of all the points belonging to the segment $\overline{x_i x_j}$. CG method has an high computational cost, order of N^2 time complexity. Same alternative method have been proposed:

- Support Vector Graph (SVG) [Ben-Hur et al., 2001]: based on the same approach of CG, according to this method the adjacency matrix is built considering the distances between support vectors and all data points x_i, order of nn_{sv} time complexity;
- Proximity Graph (PG) [Lee and Lee, 2005]: adjacency matrix is built on a different graph, order of n time complexity;
- Gradient Descend (GD) [Yang et al., 2002]: Stable Equilibrium Points (SEP) are found and the adjacency matrix is built on the SEPs, order of n_{SEP} time complexity;

- Cone Cluster Labeling(CCL) [Lee and Danels, 2006]: explained in the following, order of n_{sv^2} time complexity;

where n_{sv} is the number of Support Vectors and n_{SEP} the number of Stable Equilibrium Points.

The cone cluster labeling is different from classical methods because it is not based on distances between pairs of point. This method look for a surface that cover the hypersphere, this surface consists of a union of coned-shaped regions. Each region is associated with a support vector's features space image, the phase of each cone $\Phi_i = \angle(\phi(v_i)Oa)$ is the same, where v_i is a support vector, a is the center of the minimal hypersphere and O is the feature space origin. The image of each cone in the data space is an hypersphere, if two hypersphere overlap the two support vectors belong to the same class. So the objective is to find the radius of these hyperspheres in the data space $\|v_i - g_i\|$ where g is a generic point on the surface of the hypersphere. It can be demonstrated that $K(v_i, g_i) = \sqrt{1 - r^2}$ (Lee et Daniels 2006) so in case of polynomial kernel we obtain:

$$\begin{aligned} K(v_i, g_i) &= ((v_i g_i') + 1)^d, \\ \sqrt{1 - r^2} &= ((v_i g_i') + 1)^d. \end{aligned} \tag{3.48}$$

Starting from (3.48) it is possible to compute the coordinate of g_i: $g_i' = [(1 - r^2)^{\frac{1}{2d}} - 1]v_i'$ and consequently the value of $\|v_i - g_i\|$. If distances between two generic support vector is less than the sum of the two radius they belong to the same cluster.

3.5.4 Support Vector Clustering for categorical data

In this framework we have proposed a clustering approach based on a multistep strategy [Marino et al., 2009]: *i*) Factor Analysis on the raw data

matrix; *ii*) projection of the first factor coordinates into a higher dimensional space; *iii*) clusters identification in the high dimensional space; *iv*) clusters visualisation in the factorial space [Marino and Tortora, 2009]. The core of the proposed approach consists of steps *i* and *ii* This section aims at motivating the synergic advantage of this mixed strategy.

When the number of variables is large, projecting data into a higher dimensional space is a self-defeating and computationally unfeasible task. In order to carry only significant association structures in the analysis, dealing with continuos variables, some authors propose to perform a Principal Component Analysis on the raw data, and then to project first components in a higher dimensional feature space [Ben-Hur et al., 2001]. In the case of categorical variables, the dimensionality depends on the whole number of categories, this implies an even more dramatic problem of sparseness. Moreover, as categories are a finite number, the association between variables is non-linear.

Multiple Correspondence Analysis (MCA) on raw data matrix permits to combine the categorical variables into continuos variables that preserve the non-linear association structure and to reduce the number of variables, dealing with sparseness few factorial axes can represent a great part of the variability of the data. Indicating with F the $n \times q$ coordinates matrix of n points into the orthogonal space spanned by the first q MCA factors. Mapping the first factorial coordinates into a feature space permits to cluster data via a Support Vector Clustering approach.

Chapter 4

Factorial Probabilistic Distance Clustering

In this chapter a new factorial clustering method, Factorial Probabilistic Distance Clustering (FPDC), is introduced [Tortora et al., 2011b]; it is classified as a non-parametric probabilistic method. FPDC is a generalization of Probabilistic Distance (PD) Clustering [Ben-Israel and Iyigun, 2008] to a two steps iterative clustering method. PD-Clustering is an iterative, distribution free, probabilistic, clustering method that assigns units to a cluster according to their probability of belonging to that cluster. The probabilities are computed according to the constraint that the product between the probability and the distance of each point to any cluster center is a constant. When the number of variables is large and variables are correlated, PD-Clustering becomes unstable and the correlation between variables can hide the real number of clusters. A linear transformation of original variables into a reduced number of orthogonal ones using common criterion with PD-Clustering can significantly improve the algorithm performance.

The objective of Factorial PD-Clustering is to perform a linear transformation of the original variables and a clustering on transformed variables optimizing the same criterion.

4.1 Probabilistic Distance Clustering

PD-clustering [Ben-Israel and Iyigun, 2008] is a non hierarchical algorithm that assigns units to clusters according to their belonging probability to the cluster. Being given some random centers, the probability of any point to belong to each class is assumed to be inversely proportional to the distance from the centers of the clusters. Being given an X data matrix with n units and J variables, given K clusters that are assumed not empty, PD-Clustering is based on two quantities: the distance of each data point x_i from the K clusters centers c_k, $d(x_i, c_k)$, and the probabilities for each point to belong to a cluster, $p(x_i, c_k)$ with $i = 1, \ldots, n$ and $k = 1, \ldots, K$. The relation between them is the basic assumption of the method. Let's consider the general term x_{ij} of X and a center matrix C, of elements c_{kj} with $i = 1, \ldots, n$, $j = 1, \ldots, J$ and $k = 1, \ldots, K$, their distance can be computed according to different criteria, the squared norm is one of the most commonly used. The probability $p(x_i, c_k)$ of each point to belong to a cluster can be computed according to the following assumption: the product between the distances and the probabilities is a constant, $JDF(x_i)$, depending on x_i.
To simplify the notation we use $p_{ik} = p(x_i, c_k)$ and $d_k(x_i) = d(x_i, c_k)$; PD-clustering basic assumption is expressed as:

$$p_{ik} d_k(x_i) = JDF(x_i). \tag{4.1}$$

for a given value of x_i and for all $k = 1, \ldots, K$.

At decreasing the closeness of point from cluster center the belonging probability of the point to the cluster decreases. The constant depends only on the point and does not depend on the cluster k.
Starting from the 4.1 it is possible to compute p_{ik}. Indeed

$$p_{im}d_m(x_i) = p_{ik}d_k(x_i);\ p_{im} = \frac{p_{ik}d_k(x_i)}{d_m(x_i)};\ \forall m = 1, \ldots, K \tag{4.2}$$

The term p_{ik} is a probability so, under the constraint $\sum_{m=1}^{K} p_{im} = 1$, the sum over m of 4.2 becomes:

$$p_{ik} \sum_{m=1}^{K} \left(\frac{d_k(x_i)}{d_m(x_i)} \right) = 1,$$

$$p_{ik} = \left(\sum_{m=1}^{K} \left(\frac{d_k(x_i)}{d_m(x_i)} \right) \right)^{-1} = \frac{\prod_{m \neq k} d_m(x_i)}{\sum_{m=1}^{K} \prod_{k \neq m} d_k(x_i)}, k = 1, \ldots K. \tag{4.3}$$

Starting from the 4.1 and using 4.3 it is possible to define the value of the constant $JDF(x_i)$:

$$JDF(x_i) = p_{ik}d_k(x_i), k = 1, \ldots K,$$

$$JDF(x_i) = \frac{\prod_{m=1}^{K} d_m(x_i)}{\sum_{m=1}^{K} \prod_{k \neq m} d_k(x_i)}. \tag{4.4}$$

The quantity $JDF(x_i)$, also called *Joint Distance Function* (JDF), is a measure of the closeness of x_i from all clusters' centers. The JDF measures the classificability of the point x_i with respect to the centers c_k with $k = 1, \ldots, K$. If it is equal to zero, the point coincides with one of the clusters' centers, in this case the point belongs to the class with probability 1. If all

the distances between the point and the centers of the clusters are equal to d, $JDF(x_i) = d/k$ and all the belonging probabilities to each cluster are equal to: $p_{ik} = 1/K$. The smaller the JDF value is, the higher the probability for the point to belong to one cluster.
The whole clustering problem consists in the identification of the centers that minimizes the JDF.
Without loss of generality, the PD-Clustering optimality criterium can be demonstrated according to $K = 2$:

$$\begin{aligned} &\min \left(d_1(x_i)p_{i1}^2 + d_2(x_i)p_{i2}^2\right) \\ &\text{subject to} \qquad p_{i1} + p_{i2} = 1 \\ &\qquad\qquad\quad p_{i1}, p_{i2} \geq 0 \end{aligned} \tag{4.5}$$

The probabilities are squared because it is a smoothed version of the original function. The Lagrangian of this problem is:

$$\mathcal{L}(p_{i1}, p_{i2}, \lambda) = d_1(x_i)p_{i1}^2 + d_2(x_i)p_{i2}^2 - \lambda(p_{i1} + p_{i2} - 1) \tag{4.6}$$

Setting to zero the partial derivates with respect to p_{i1} and p_{i2}, substituting the probabilities 4.3 and considering the principle $p_{i1}d_1(x_i) = p_{i2}d_2(x_i)$ we obtain the optimal value of the Lagrangian.

$$\mathcal{L}(p_{i1}, p_{i2}, \lambda) = \frac{d_1(x_i)d_2(x_i)}{d_1(x_i) + d_2(x_i)}. \tag{4.7}$$

This value coincides with the JDF, the matrix of centers that minimizes this principle minimizes the JDF too. Substituting the generic value $d_k(x_i)$ with $\|x_i - c_k\|$, it can be found the equations of the centers that minimize the JDF (and maximize the probability of each point to belong to only one

cluster).

$$c_k = \sum_{i=1,\ldots,N} \left(\frac{u_k(x_i)}{\sum_{j=1,\ldots,N} u_k(x_j)} \right) x_i, \tag{4.8}$$

where

$$u_k(x_i) = \frac{p_{ik}^2}{d_k(x_i)}. \tag{4.9}$$

As showed before, the value of JDF at all centers is equal to zero and it is necessarily positive elsewhere. So the centers are the global minimizer of the JDF. Other stationary points may exist because the function is not convex neither quasi-convex, but they are saddle points [Iyigun, 2007].

There are alternative ways for modeling the relation between probabilities and distances, for example the probabilities can decay exponentially as distances increase. In this case the probabilities p_{ik} and the distances $d_k(x_i)$ are related by:

$$p_{ik} e^{d_k(x_i)} = E(x_i), \tag{4.10}$$

where $E(x_i)$ is a constant depending on x_i.
Many results of the previous case can be extended to this case by replacing the distance $d_k(x_i)$ with $e^{d_k(x_i)}$. Interested readers are referred to Ben-Israel and Iyigun [Ben-Israel and Iyigun, 2008].

The optimization problem presented in 4.5 is the original version proposed by Ben-Israel and Iyigun. Notice that in the optimization problem the probabilities p_{ik} are considered in squared form. The authors affirm that it is possible to consider $d_k(x_i)$ as well $d_k^2(x_i)$. Both choices have some advantages and drawbacks. Squared distances offer analytical advantages due

to linear derivates. Using simple distances endures more robust results and the optimization problem can be re-conducted to a Fermat-Weber location problem. The Fermat-Weber location problem aims at finding a point that minimizes the sum of the Euclidean distances from a set of given points. This problem can be solved with the Weiszfeld method [Weiszfeld, 1937]. Convergence of this method was established by modifying the gradient so it is always defined [Khun, 1973]. The modification is not carried out in practice. The global solution is guaranteed only in case of one cluster. Dealing with more than one cluster, in practice, the method converges only for a limited number of centers depending on the data.

In this thesis we consider the squared form:

$$d_k(x_i) = \sum_{j=1}^{J} (x_{ij} - c_{kj})^2, \tag{4.11}$$

where $k = 1, \ldots, K$ and $i = 1, \ldots, N$. Starting from the 4.11 the distance matrix D of order $n \times K$ is defined, where the general element is $d_k(x_i)$. The final solution $J\hat{D}F$ is obtained minimising the quantity:

$$JDF = \sum_{i=1}^{n} \sum_{k=1}^{K} d_k(x_i) p_{ik}^2 = \sum_{i=1}^{n} \sum_{j=1}^{J} \sum_{k=1}^{K} (x_{ij} - c_{kj})^2 p_{ik}^2, \tag{4.12}$$

$$J\hat{D}F = \arg\min_{C} \sum_{i=1}^{n} \sum_{j=1}^{J} \sum_{k=1}^{K} (x_{ij} - c_{kj})^2 p_{ik}^2. \tag{4.13}$$

Where c_k is the generic center and $d_k(x_i)$ is defined in 4.11.

The solution of PD-clustering problem can be obtained through an iterative algorithm.

Algorithm 5 Probabilistic Distance Clustering Function

```
1: function PDC(X, K)
2:     C ← rand(K, J)                          ▷ Matrix C_{K,J} is randomly initialised
3:     JDF ← 1/eps                             ▷ JDF is initialised to the maximum
4:     D ← 0                                   ▷ Initialise the array D, of dimension n × K, to 0
5:     p ← 1/k                                 ▷ Initialise to 1/k the probability vector p of n elements
6:     repeat
7:         for k = 1, K do
8:             D_k ← distance(X, C(k))         ▷ D_k distances computed according to 4.11
9:         end for
10:        JDF0 ← JDF                          ▷ Current JDF is stored in JDF0
11:        C ← C*                              ▷ Centres are updated according to formula 4.8
12:        JDF ← jdf(D)                        ▷ ← jdf(D) implements the formula 4.4
13:    until JDF0 > JDF
14:    P ← compp(D)                            ▷ function compp implements the formula 4.3
       return C, P, JDF
15: end function
```

The algorithm convergence is demonstrated in [Iyigun, 2007].
Each unit is then assigned to a cluster according to the highest probability that is computed a posteriori using the formula in equation 4.3.

4.2 Factorial PD-Clustering

When the number of variables is large and variables are correlated, PD-Clustering becomes very unstable and the correlation between variables can hide the real number of clusters. A linear transformation of the original variables into a reduced number of orthogonal ones can significantly improve the algorithm performance. Combination of PD-Clustering and variables linear transformation implies a common criterion [Tortora et al., 2011a].

This section shows how the Tucker3 method [Kroonenberg, 2008] can be properly adopted for the transformation into the Factorial PD-Clustering;

an algorithm is then proposed to perform the method.

4.2.1 Same theoretical aspects of Factorial PD-clustering

Firstly it is demonstrated that the minimization problem in 4.12 corresponds to the Tucker3 decomposition of the $n \times J \times K$ distance matrix G of general elements $g_{ijk} = |x_{ij} - c_{kj}|$, where $i = 1, \ldots, n$ indicates the units, $j = 1, \ldots, J$ the variables and $k = 1, \ldots, K$ the clusters. For any c_k a $n \times J$ G_k distances matrix is defined. In matrix notation:

$$G_k = X - hc_k \tag{4.14}$$

where h is an $n \times 1$ column vector with all terms equal to 1; X and c_k have been already defined in section 4.1.
Tucker3 method decomposes the matrix G in three components, one for each mode, in a full core array Λ and in an error term E.

$$g_{ijk} = \sum_{r=1}^{R} \sum_{q=1}^{Q} \sum_{s=1}^{S} \lambda_{rqs}(u_{ir} b_{jq} v_{ks}) + e_{ijk}, \tag{4.15}$$

where λ_{rqs} and e_{ijk} are respectively the general terms of the three way matrices Λ of order $R \times S \times Q$ and E of order $n \times J \times K$. The elements u_{ir}, b_{jq} and v_{ks} are respectively the general terms of the matrices U of order $n \times R$, B of order $J \times Q$ and V of order $K \times S$, with $i = 1, \ldots, n$, $j = 1, \ldots, J$, $k = 1, \ldots, K$.
As in all factorial methods, factorial axes in Tucker3 model are sorted according to the explained variability. The first factorial axes explain the greatest part of the variability, latest factors are influenced by anomalous data or represent the ground noise. For this reason the choice of a number of factors lower than the number of variables makes the method

robust. According to Kiers and Kinderen [Kiers and Kinderen, 2003] the choice of the parameters R, Q and S is a ticklish problem because they define the overall explained variability. The interested readers are referred to [Kroonenberg, 2008] for the theoretical aspects concerning this choice. In the present thesis it is used an heuristic approach to cope with this crucial issue: it is suggested to choose the minimum number of factors that corresponds to a significant value of the explained variability.

The coordinates x^*_{iq} of the generic unit x_i into the space of variables obtained through Tucker3 decomposition are obtained by the following expression:

$$x^*_{iq} = \sum_{j=1}^{J} x_{ij} b_{jq}. \tag{4.16}$$

Finally on these x^*_{iq} coordinates a PD-Clustering is applied in order to solve the clustering problem.

Let's start considering the expression 4.12; it is worth noting that minimizing the JDF:

$$\min_C \sum_{i=1}^{n} \sum_{j=1}^{J} \sum_{k=1}^{K} (x_{ij} - c_{kj})^2 p_{ik}^2 \tag{4.17}$$

$$\text{subject to } \sum_{i=1}^{n} \sum_{k=1}^{K} p_{ik}^2 \leq n, \tag{4.18}$$

is equivalent to:

$$\max_c \sum_{i=1}^{n} \sum_{j=1}^{J} \sum_{k=1}^{K} (x_{ij} - c_{kj})^2 p_{ik}^2, \tag{4.19}$$

$$\text{subject to } \sum_{i=1}^{n} \sum_{k=1}^{K} p_{ik}^2 \leq n. \tag{4.20}$$

Taking into account the following Proposition 1 (proved in section 4.2.3) and the lemma given below, we demonstrate that the Tucker3 decomposition is a consistent linear variable transformation that determines the best subspace according to the PD-clustering criterion.

Proposition 1 *Given an unknown matrix B of generic element b_{im} and a set of coefficients $0 \leq \psi_{im} \leq 1$, with $m = 1, \ldots, M$ and $i = 1, \ldots, n$ maximize*

$$-\sum_{m=1}^{M}\sum_{i=1}^{n} b_{im}\psi_{im}^2,$$

s.t. $\sum_{m=1}^{M}\sum_{i=1}^{n}\psi_{im}^2 \leq n$ *is equivalent to solve the equation*

$$\sum_{m=1}^{M}\sum_{i=1}^{n} b_{im}\psi_{im} = \mu\sum_{m=1}^{M}\sum_{i=1}^{n}\psi_{im},$$

where $\mu \geq 0$.

Lemma. *Tucker3 decomposition permits to define the best subspace for the PD-clustering.*

Considering the *Proposition 1* where:

$$\begin{aligned} M &= K \\ b_{ik} &= \sum_{j=1}^{J}(x_{ij} - c_{kj})^2 \\ \text{and} & \\ \psi_{ik} &= p_{ik}, \quad \text{with } i = 1, \ldots, n: \ k = 1, \ldots, K \end{aligned} \tag{4.21}$$

Let us assume that c_{kj} and p_{ik} are known, replacing $(x_{ij} - c_{kj})$ with g_{ijk} in 4.17 we obtain the following squared form:

$$\max \left(-\sum_{k=1}^{K} \sum_{i=1}^{n} \left(\sum_{j=1}^{J} g_{ijk}^2 \right) p_{ik}^2 \right) \qquad \text{subject to} \quad \sum_{k=1}^{K} \sum_{i=1}^{n} p_{ik}^2 \leq n$$

according to the Proposition 1 we obtain:

$$\sum_{k=1}^{K} \sum_{i=1}^{n} \left(\sum_{j=1}^{J} g_{ijk}^2 \right) p_{ik} = \mu \sum_{k=1}^{K} \sum_{i=1}^{n} p_{ik} \tag{4.22}$$

The value of μ that optimize the 4.22 can be find trough the singular value decomposition of the matrix G, which is equivalent to the following Tucker3 decomposition:

$$g_{ijk} = \sum_{r=1}^{R} \sum_{q=1}^{Q} \sum_{s=1}^{S} \lambda_{rqs} (u_{ir} b_{jq} v_{ks}) + e_{ijk}, \tag{4.23}$$

with $i = 1, \ldots, n$, $j = 1, \ldots, J$, $k = 1, \ldots, K$.
Where R is number of components of U, Q the number of components of B and S the number of components of V.
The 4.23 can be rewrite in matrix notation as:

$$G = U\Lambda(V' \otimes B') + E \tag{4.24}$$

■

The Proposition 1 and the Lemma 1 demonstrate that the Tucker3 transformation of the distance matrix G minimizes JDF. Subsection 4.2.3 presents an iterative algorithm to alternatively calculate c_{kj} and p_{ik} on one

hand, and b_{jq} on the other hand, until the convergence is reached. In section 5.2 it is empirically demonstrated that the minimisation of the quantity in the formula 4.17 converges at least to local minima.

4.2.2 Factorial PD-clustering iterative algorithm

Let's start considering the equation 4.13, where it is applied the linear transformation $x_{ij}b_{jq}$ to x_{ij} according to 4.16:

$$J\hat{D}F = \arg\min_{C;B} \sum_{i=1}^{n} \sum_{q=1}^{Q} \sum_{k=1}^{K} (x_{iq}^{*} - c_{kq})^2 p_{ik}^2. \tag{4.25}$$

Let's note that in formula 4.25: x_{ij} and b_{jq} are the general elements of the matrices X and B that have been already defined in section 4.2.1; c_{kq} is the general element of the matrix C (see eq. 4.8).
It is worth to note that C and B are unknown matrices and p_{ik} is determined as C and B are fixed. The problem does not admit a direct solution and an iterative two steps procedure is required.
The two alternative steps are:

- linear transformation of original data;
- PD-Clustering on transformed data.

The procedure starts with a pseudorandomly defined centre matrix C of elements c_{kj} with $k = 1, \ldots, K$ and $j = 1, \ldots, J$. Then a first solution for probabilities and distance matrices is computed according to 4.14. Given the initial C and X, the matrix B is calculated; once B is fixed the matrix C is updated (and the values p_{ik} are consequently updated). Last two steps are iterated until the convergence is reached: $J\hat{D}F^{(t)} - J\hat{D}F^{(t-1)} > 0$,

where t indicates the number of iterations.
In algorithm 6 the procedure is presented according to the usual flow diagram notation.

Algorithm 6 Factorial Probabilistic Distance Clustering

function FPDC(X, K)
  $JDF \leftarrow 1/\text{eps}$ ▷ JDF is initialised to the maximum
  $G \leftarrow 0$ ▷ Initialise the array G, of dimension $n \times J \times K$, to 0
  $P \leftarrow \frac{1}{k}$ ▷ Initialise to $\frac{1}{k}$ the probability vector p of n elements
  $C \leftarrow \text{rand}(K, J)$ ▷ Matrix $C_{K,J}$ is randomly initialised
  repeat
    for $k = 1, K$ **do**
      $G_k \leftarrow distance(X, C(k))$ ▷ G_k distances of all units from the centre k
    end for
    $B \leftarrow Tucker3(G)$ ▷ Tucker3 fun. in MatLab Toolbox N-way [Chen, 2010]
    $X^* \leftarrow XB$
    $JDF0 \leftarrow JDF$ ▷ Current JDF is stored in $JDF0$
    $(C, P, JDF) \leftarrow PDC(X^*, K)$ ▷ PDC() function is defined by the algorithm 5
  until $JDF0 > JDF$
  return C, P
end function

4.2.3 Proof of the Proposition 1

Proposition 1 *Given an unknown matrix B of generic element b_{im} and a set of coefficients $0 \leq \psi_{im} \leq 1$, with $m = 1, \ldots, M$ and $i = 1, \ldots, n$.* Maximising

$$-\sum_{m=1}^{M}\sum_{i=1}^{n} b_{im}\psi_{im}^2,$$

s.t. $\sum_{m=1}^{M}\sum_{i=1}^{n}\psi_{im}^2 \leq n$ *is equivalent to solve the equation*

$$\sum_{m=1}^{M}\sum_{i=1}^{n} b_{im}\psi_{im} = \mu\sum_{m=1}^{M}\sum_{i=1}^{n}\psi_{im},$$

where $\mu \geq 0$.

Proof (Proposition 1)**.** *To prove the proposition we introduce the Lagrangian function:*

$$\mathcal{L} = -\sum_{m=1}^{M}\sum_{i=1}^{n} b_{im}\psi_{im}^2 + \mu(\sum_{m=1}^{M}\sum_{i=1}^{n}\psi_{im}^2 - n)$$

where μ *is the Lagrange multiplier. Let us consider the first derivative of* $\mathcal{L}$ *w.r.t.* ψ_{im} *equal to 0 :*

$$\frac{\partial \mathcal{L}}{\partial \psi_{im}} = -2\sum_{m=1}^{M}\sum_{i=1}^{n} b_{im}\psi_{im} + 2\mu\sum_{m=1}^{M}\sum_{i=1}^{n}\psi_{im} = 0$$

which is equivalent to

$$\sum_{m=1}^{M}\sum_{i=1}^{n} b_{im}\psi_{im} = \mu\sum_{m=1}^{M}\sum_{i=1}^{n}\psi_{im}$$

■

4.3 Example on a real quantitative dataset

In order to evaluate the performances of Factorial PD-clustering it has been applied on the dataset used in [Vichi and Kiers, 2001]. The dataset *latest short-term indicators and economic performance indicators*[1] contains 6 macroeconomic variables measured on 20 countries members of the OECD. Variables are: Gross Domestic Product (GDP), Leading Indicator (LI), Unemployment Rate (UR), Interest Rate (IR), Trade Balance (TB), Net National Savings (NNS). Table 4.1 contains the dataset.

[1] OECD, Paris 1999

Country	**Label**	**GDP**	**LI**	**UR**	**IR**	**TB**	**NNS**
Australia	A-lia	4.80	8.40	8.10	5.32	0.70	4.70
Canada	Can	3.20	2.50	8.40	5.02	1.60	5.20
Finland	Fin	3.90	-1.00	11.80	3.60	8.80	7.70
France	Fra	2.30	0.70	11.70	3.69	3.90	7.30
Spain	Spa	3.60	2.50	19.00	4.83	1.20	9.60
Sweden	Swe	4.10	1.10	8.90	4.20	7.00	4.00
United States	USA	4.10	1.40	4.50	5.59	-1.40	7.00
Netherlands	Net	2.90	1.60	4.20	3.69	7.00	15.80
Greece	Gre	3.20	0.60	10.30	11.70	-8.30	8.00
Mexico	Mex	2.30	5.60	3.20	20.99	0.00	12.70
Portugal	Por	2.80	-7.50	4.90	4.84	-8.70	14.00
Austria	A-tria	1.10	0.60	4.70	3.84	-0.60	9.40
Belgium	Bel	1.40	-0.10	9.60	3.64	4.50	12.40
Denmark	Den	1.00	1.50	5.30	4.08	3.30	5.00
Germany	Ger	0.80	-2.00	9.50	3.74	1.50	7.70
Italy	Ita	0.90	-0.40	12.30	6.08	4.30	8.20
Japan	Jap	0.10	5.40	4.20	0.74	1.20	15.10
Norway	Nor	1.40	0.90	3.30	4.47	7.10	15.10
Switzerland	Swi	1.10	2.10	3.80	1.84	4.40	13.20
United Kingdom	UK	1.20	4.90	6.40	7.70	-0.50	4.80

Table 4.1: Six macroeconomic performance indicators of twenty OECD countries (percentage change from the previous year, September 1999)

The first step of Factorial PD-clustering is choosing the number of clusters K that has been fixed equal to 3.

The choice of the number of factors is a ticklish well known issue. An empirical technique has been used for this choice, FPDC is iterated varying the number of factors for units and variables and at each iteration the explained variability is measured. The number of factors for clusters dimension is fixed equal to $K - 1$. The values of the explained variability are plotted on a graphic where the horizontal axis is the number of factors for units and the vertical one is the value of the explained variability. The

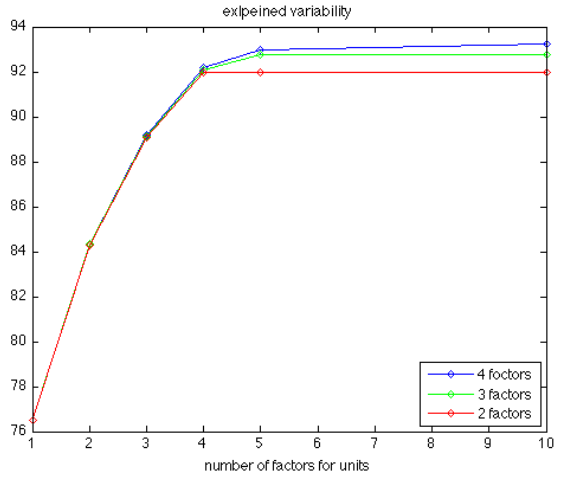


Figure 4.1: Explaiend variabilty varying the number of Tucker 3 factors

number of factors for variables is fixed. A different line is plotted for every number of factor for variables chosen. We choose the minimum number of factors that corresponds to a significant value of the explained variability. In this case the plot is represented in fig. 4.3. The number of factor for clusters is chosen equal to $K-1=2$, the significant value for the explained variability is chosen equal to 92% of the variability. Looking at fig. 4.3 92% of explained variability corresponds to 4 factors for the units. The minimum number of factors for variables that corresponds to 92% of explained variability is 2. Summarizing the number of factors that have been chosen are: 2 factors for the variables, 4 factors for the units and 2 factors for the clusters.

The factors correspond to the values of R, Q and S respectively in the (4.23).

The method has been iterated 50 times, to test the stability of the results.

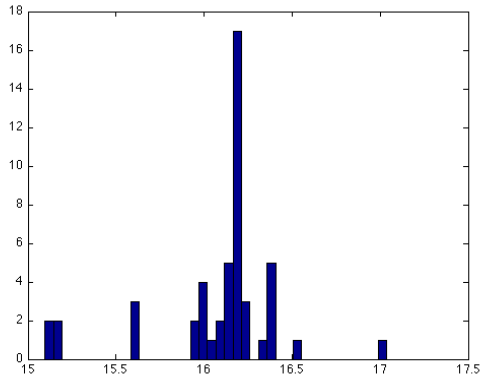


Figure 4.2: Frequency of JDF on 200 iterations of the Factorial PD-clustering

The JDF index has been measured at each iteration, fig. 4.7 represents the JDF value. In the best case the value of JDF is 15.12, it occurs in 4% of the cases. The modal value 16.19 occurs in 35% of cases.
Table 4.3 displays Factorial PD-clustering iterations summary statistic[2].
The partition of units in clusters is:

- **cluster 1** Mexico, Austria, Denmark, Japan, Norway, Switzerland, United Kingdom;
- **cluster 2** Finland, France, Spain, Netherlands, Portugal, Belgium, Germany, Italy;
- **cluster 3** Australia, Canada, Sweden, United States, Greece.

[2]The time has been measured by using Matlab R2007a on a Mac os X Snow leopard, Ram 4gb 1067Mhz DDR3, processor 2.26 GHz Intel Core 2 Duo.

JDF	Frequency	mean execution time	mean number of iterations
15,12	4,08%	0,39	8
15,17	4,08%	0,46	12
15,61	6,12%	0,34	6
15,95	4,08%	0,18	5
15,99	8,16%	0,67	17,75
16,04	2,04%	0,3	8
16,09	4,08%	0,33	8
16,14	10,20%	0,33	8,6
16,19	34,69%	0,45	10,23
16,24	6,12%	0,29	6,33
16,33	2,04%	0,32	7
16,38	10,20%	0,24	5,6
16,53	2,04%	0,38	5
17,01	2,04%	0,12	3

Table 4.2: Summary statistics on 200 iterations of Factorial PD-clustering

To describe the separating power of our variables in the following explanation are shortly described results illustrated in figures 4.3 to 4.6.
The differences between the medians have been evaluated for each cluster and for each variable, to understand which are the variables that mostly contribute to the class separability. The results can be better understood looking at the box-plots in fig. 4.3. The variable Net National savings presents the highest difference between the medians. NNS variable separates the second cluster from the others, where the values of the variable are lower, in cluster 1 the variable has a high variability. The variable Unemployment Rate presents a high difference between clusters medians: in cluster 1 the values of this variables are smaller than in the other clusters; in cluster 2 UR presents a small variability. The variable Gross Domestic Product presents high values in cluster 2 and small values in cluster 1. The variable Trade Balance is low in cluster 2. The medians of the variables

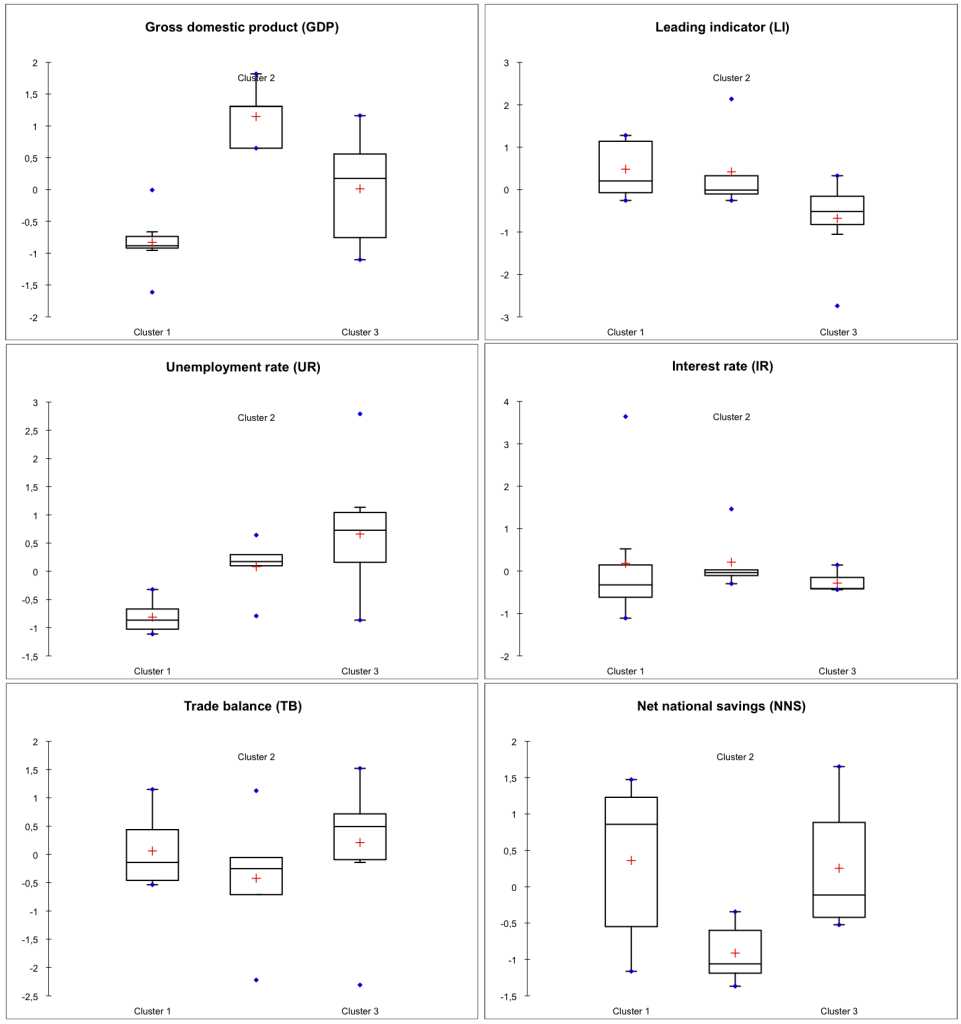


Figure 4.3: Box-plot of variables for each cluster obtained with Factorial PD-clustering

Interest Rate and Leading Indicator are not significantly different among clusters.

Fig. 4.4, 4.5 and 4.6 represent scatter-plots of units on the variables ordered according to discriminating power. Leading indicator and Interest rate have not discriminating power for the cluster partition obtained, see fig. 4.6.
Scatter-plots in figures 4.4 and 4.5 show that:

- Mexico, Austria, Denmark, Japan, Norway, Switzerland and United Kingdom have a low variation of Gross Domestic Product but they have a low Unemployment Rate and hight Trade Balance;
- Australia, Canada, Sweden, United States and Greece have a high Gross Domestic Product variations, low Unemployment Rate but they have low values of the Net National Savings variable, Greece is different from other elements of the cluster for some aspects;
- Finland, France, Spain, Netherlands, Portugal, Belgium, Germany and Italy have average values of Gross Domestic Product and Unemployment Rate and high values of Net National Savings.

The clusters description, presented in the foregoing, corresponds to the most frequent case of the JDF. Taking into account all iterations, it is worth noticing that three stable groups of countries are always together in the same cluster:

- Australia, Canada, United States;
- Finland, France, Spain;
- Belgium, Germany, Italy.

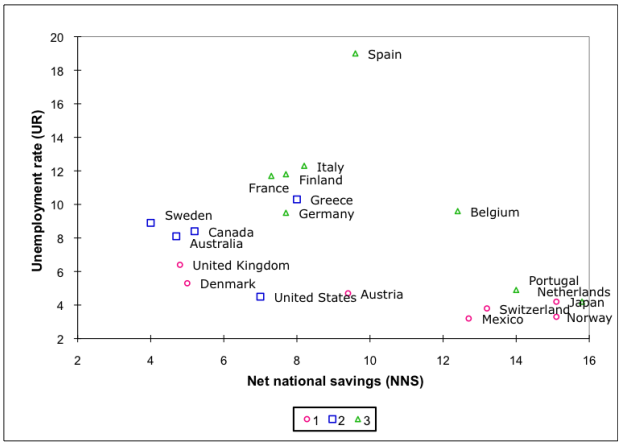


Figure 4.4: Scatter-plot of units divided in clusters obtained with Factorial PD-clustering

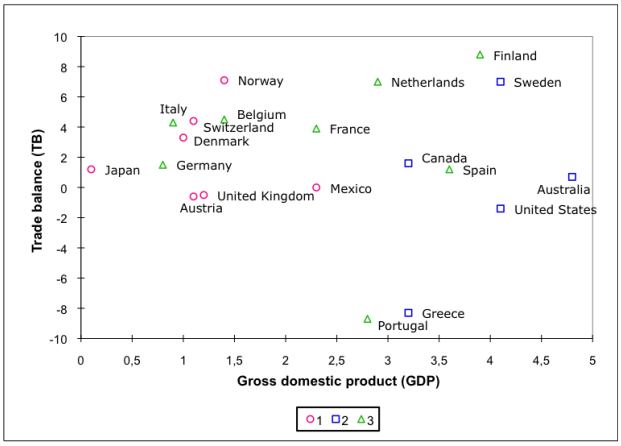


Figure 4.5: Scatter-plot of units divided in clusters obtained with Factorial PD-clustering

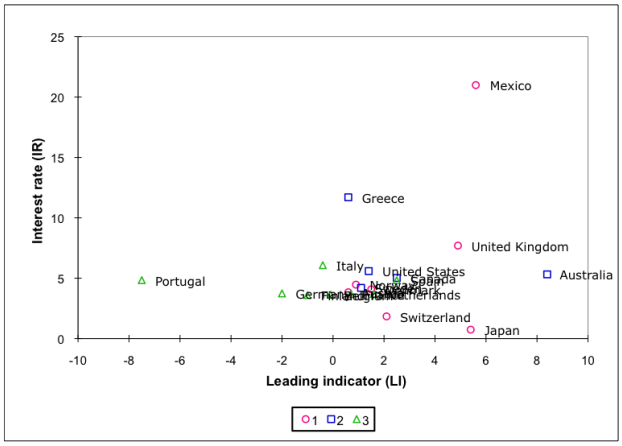


Figure 4.6: Scatter-plot of units divided in clusters obtained with Factorial PD-clustering

4.3.1 A comparison with Factorial K-means

In order to evaluate Factorial PD-clustering results, the method has been compared with Factorial K-means discussed in section 3.4.1. The method have been iterated 200 times.

Whitin variance	Frequency	mean execution time	mean number of iterations
5865,59	16,53%	0,004	2,63
961,92	23,14%	0,004	2,64
970,9	3,31%	0,004	2,75
999,52	54,55%	0,004	2,27
1256,42	2,48%	0,004	2

Table 4.3: Summary statistics on 200 iterations of Factorial K-means

Using the same scheme adopted for Factorial PD-Clustering result description, in the following Factorial K-means results are shortly described. The

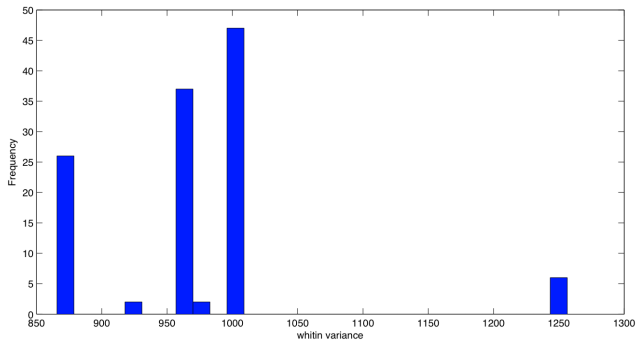


Figure 4.7: Frequency of within variance on 200 iterations of Factorial K-means

method has been applied on the same dataset; the results shown are consistent with those presented in the original Vichi and Kiers' paper.
To identify the most discriminating variables, in this case, the differences between the mean values of the clusters have been evaluated. This is because the K-means maximizes the differences between cluster centroids.
In order to easily compare the results, the same graphics are represented: box-plots in fig. 4.8; Scatter-plots indicating the cluster membership are represented in fig.4.9, 4.10, 4.11. Variables have not the same order used to describe FPDC results because the discriminating power is different.
It is important to notice that Factorial K-means results are consistent with the Factorial PD-clustering ones: specially the method identifies the same stable groups of countries. The most significative difference is in cluster 2: Factorial K-means identifies a cluster composed by Portugal, Greece and Mexico; Factorial PD-clustering assigns Portugal, Greece and Mexico in three different clusters. Looking at the scatter plot in fig. 4.9, where

statistical units are represented according to the two most separating variables, these three countries appear as the most different from the global mean. It is not surprising that the K-means algorithm separates this three points from the others, because it minimizes the variance within the clusters, and as a consequence it gives maximum importance to the variables Trade Balance and Interest Rate.

To summarize: Factorial K-means has found a cluster of few elements and large variability and two clusters having a larger number of elements (7 and 10 elements, respectively) and a small variability. This partition emphasizes the differences between the variables: Interest Rate, Trade Balance and Net National Savings. Differently Factorial PD-clustering have divided the space in three regions defining three clusters having almost the same variability and almost the same number of elements (7,8 and 5 elements, respectively). The clusters emphasize the differences between the variables: Net National Savings, Unemployment Rate and Gross Domestic Product.

The results obtained with Factorial K-means are different in terms of discriminating variables; the two methods emphasize different aspects of the same phenomenon.

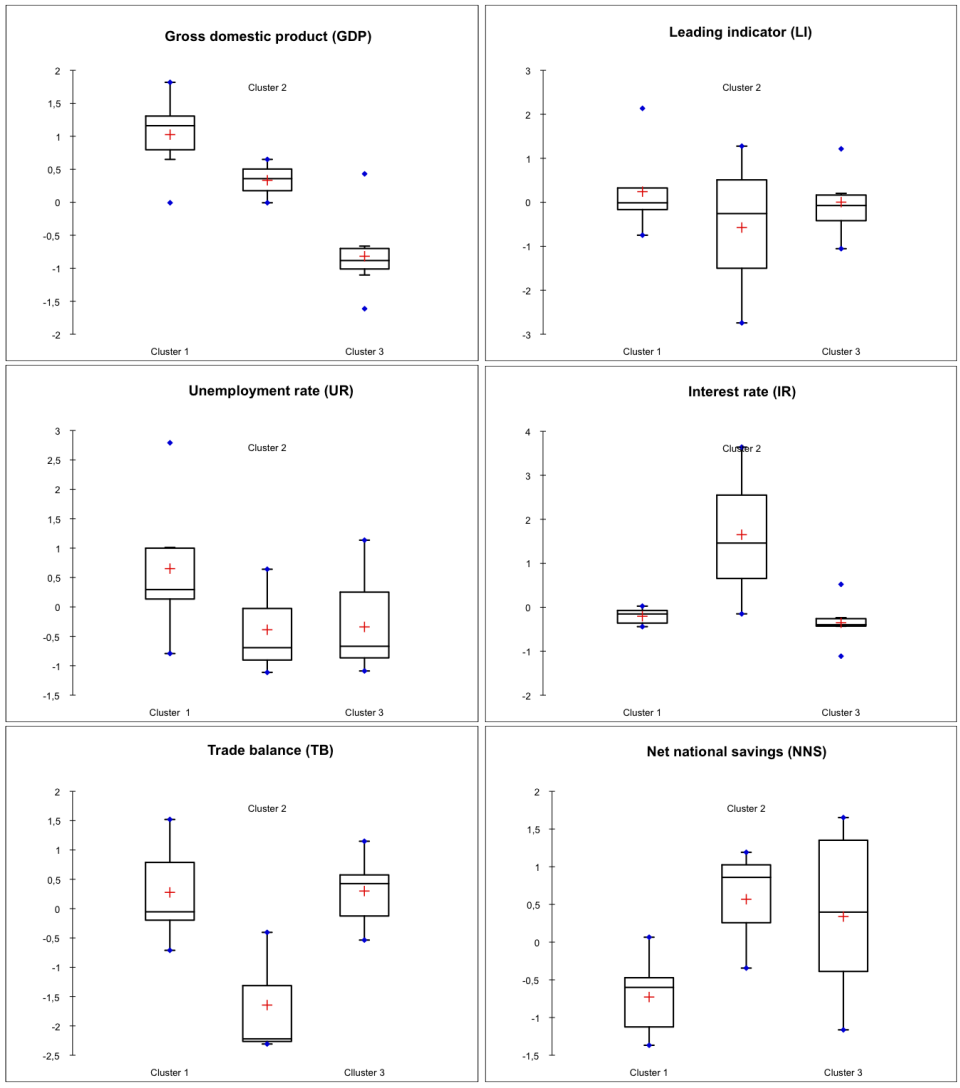


Figure 4.8: Box-plot of variables for each cluster obtained with Factorial K-means

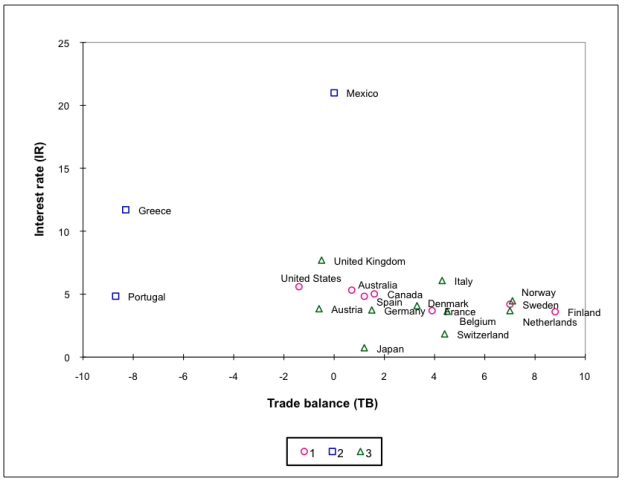


Figure 4.9: Scatter-plot of units divided in clusters obtained with Factorial K-means

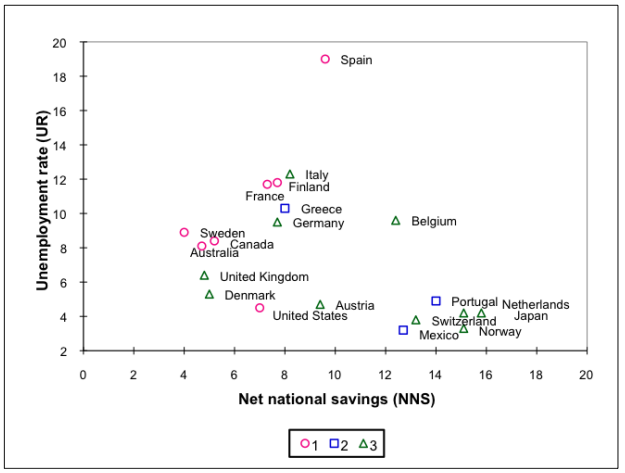


Figure 4.10: Scatter-plot of units divided in clusters obtained with Factorial K-means

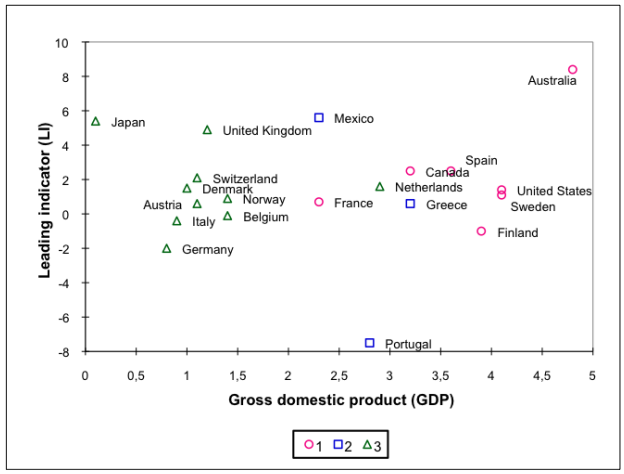


Figure 4.11: Scatter-plot of units divided in clusters obtained with Factorial K-means

Chapter 5

Computational aspects

This chapter involves a simulation study in order to evaluate the performance of the Factorial Probabilistic Distance Clustering (FPDC). The simulation design is based on the study made by Maronna and Zamar in 2002 [Maronna and Zamar, 2002].
The results presented in the following sections have been produced with Matlab. The toolbox *N-way* have been integrated in Matlab code to obtain Tucker3 decomposition. The toolbox is freely available at Matlab Central web site.[1].
The Matlab code used for the realization of this simulation study is in Appendix A.2.

5.1 Simulation design

Clusters structure and shape affects the performance of clustering methods. The objective of this section is to evaluate FPDC behavior facing the

[1] http://www.mathworks.com/matlabcentral/

following issues:

1. within-cluster variance changing among clusters;
2. clusters with different number of elements;
3. correlation among the variables;
4. outliers;
5. clusters of different shape.

Point 2 is a weak point of Classical PD-clustering [Iyigun, 2007], the algorithm works well when the number of elements in each cluster is similar but it can have some problem in finding the right clustering structure in the opposite case. In order to evaluate the performance of FPDC dealing with these issues, ad hoc datasets have been simulated. In each dataset every cluster have been obtained generating uncorrelated normal data $x_i \sim N_k(0, I)$ where I is a $J \times J$ identity matrix. For every cluster n_k J dimensional vectors x_i have been generated, where n_k is the number of elements of each cluster, with $k = 1, \ldots, K$, the number of clusters $K \in [2, 4]$. After, every cluster have been centered on a different center, the centers are uniformly distributed on a hypersphere. In order to study the performance of FPDC facing each issue presented above, 4 specific datasets for every issue have been generated. Datasets structures are: 2 clusters and 3 variables, 2 clusters and 6 variables, 3 clusters and 2 variables,4 cluster and 2 variables. The simulation have been obtained also for a different number of variables but for sake of brevity only significant results have been detailed. To evaluate the performances of FPDC when within-cluster variance changes among clusters (issue 1) each element x_i of every cluster have been transformed according to: $y_i = G_k x_i$, where G_k^2 is a covariance matrix with

$G_{jj} = v_{jk}$ and $G_{jr} = 0$ for $r \neq j$ and v_{jk} is the variance of variable j in cluster k. In this case variables are supposed to be uncorrelated, G_k^2 is a diagonal variance matrix.

To examine the performances of FPDC when clusters have different number of elements (issue 2) n_k changes varying the value of k.

To control the effect of the correlation among the variables (issue 3) on the performance of FPDC, each element of every cluster x_i have been transformed according to: $y_i = G_k x_i$, where G_k^2 is a covariance matrix with $G_{jj} = 1$ and $G_{jr} = \rho$ for $r \neq j$.

To evaluate whether the presence of outliers (issue 4) affects the results, the dataset have be contaminated at a fixed level ϵ. The levels of contamination investigated are: 10% and 20%. The contamination is obtained by generating $x_i \sim N_k(ra_0\sqrt{J}, G_k)$ where a_0 is a unitary vector generated orthogonal to $(1, 1, \ldots, 1)^T$ and r measures the distance between the outliers and the cluster center. In this simulation design the parameter r varies between r_{min} and 9 where $r_{min} = \frac{\left(1.2\sqrt{\chi^2_{J,1-\alpha}} + \sqrt{\chi^2_{J,1-\alpha}}\right)}{\sqrt{J}}$ to avoid that outliers overlap the elements of the clusters.

To control the effect of clusters of different shapes (issue 5) on the performance of FPDC two dataset have been generated. Each dataset is composed by two clusters, the first one normally generated, $x_i \sim N(c_1, I)$, the second one log-normally generated, $x_i \sim logN(0, I)$.

For each simulation the method has been iterated 50 times in order to evaluate the stability of the results.

Structure	N	Characteristics	exp. var.	iter.	error
	400	standard	65.83%	3	0
	400	within variance changing	64%	4	0.25%
	300	different number of elements	72.62%	3	0.3%
2 clusters	400	variable cor.	87.57%	29	0
3 variables	440	variable cor. $r = min$ $\epsilon = 10\%$	80.42%	16	0.91%
	440	variable cor. $r = max$ $\epsilon = 10\%$	79.19%	11	1.14%
	480	variable cor. $r = min$ $\epsilon = 20\%$	80.1%	7	1.04%
	480	variable cor. $r = max$ $\epsilon = 20\%$	79.98%	10	1.04%
	400	different shape	66.7%	3	4.5%
	400	standard	94.7%	5	0
	400	within variance changing	$93,5\%$	6	2%
	300	different number of elements	94.7%	5	0
2 clusters	400	variable cor.	91.7%	5	0
6 variables	440	variable cor. $r = min$ $\epsilon = 10\%$	95.7%	13	0
	440	variable cor. $r = max$ $\epsilon = 10\%$	97.7%	13	0
	480	variable cor. $r = min$ $\epsilon = 20\%$	$96,5\%$	14	0
	480	variable cor. $r = max$ $\epsilon = 20\%$	97.9%	14	0
	400	different shape	93.8%	19	1.5%
	600	standard	91%	3	0
	600	within variance changing	91.64%	3	1%
	450	different number of elements	92.84%	3	0.67%
3 clusters	600	variable cor.	90.53%	29	0.02%
2 variables	660	variable cor. $r = min$ $\epsilon = 10\%$	91.4%	3	0.3%
	660	variable cor. $r = max$ $\epsilon = 10\%$	92.30%	3	0.3%
	720	variable cor. $r = min$ $\epsilon = 20\%$	93.63%	3	0.28%
	720	variable cor. $r = max$ $\epsilon = 20\%$	92.15%	3	0.28%
	400	standard	98.32%	3	0
	400	within variance changing	97.81%	3	2%
	450	different number of elements	95.72%	3.72	0.89%
4 clusters	400	variable cor.	95.98%	3	1%
2 variables	440	variable cor. $r = min$ $\epsilon = 10\%$	96.24%	3	0.68%
	440	variable cor. $r = max$ $\epsilon = 10\%$	96.39%	3	0.91%
	480	variable cor. $r = min$ $\epsilon = 20\%$	93.31%	3	0.63%
	480	variable cor. $r = max$ $\epsilon = 20\%$	96.47%	4.78	0.63%

Table 5.1: Results on 50 iterations of FPDC

5.2 Simulation results

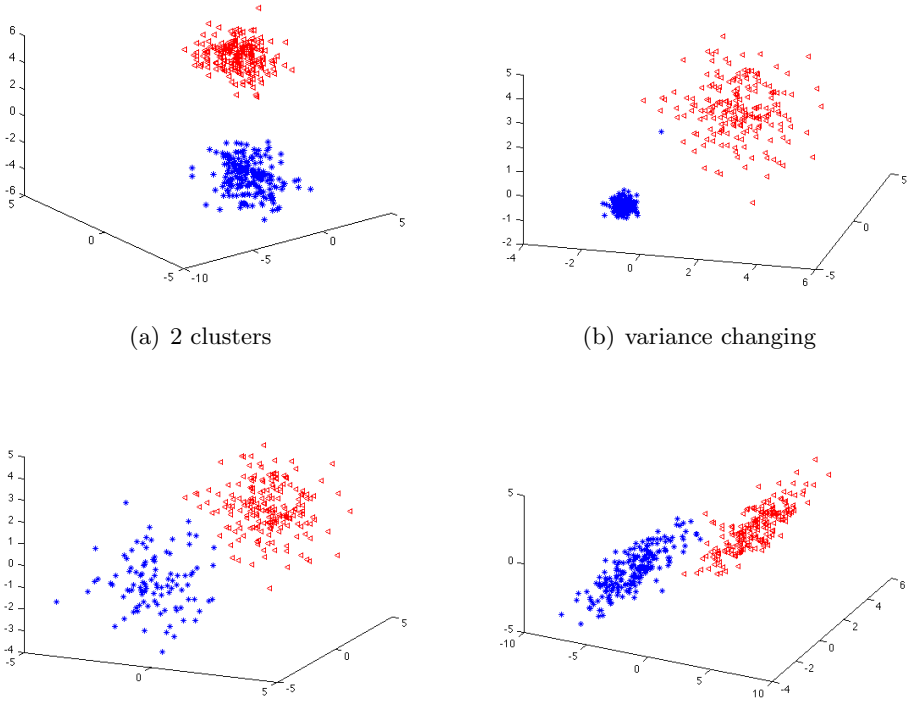
The results of the application of FPDC on the datasets described in the section 5.1 are described in table 5.1 . For each simulation the following attribute are described:

- **N**: number of units;
- **exp. var.**: explained variability of factorial axes;
- **iter.**: average number of FPDC iterations on 50 iterations of the method;
- **error**: percentage of misclassified units.

The average time elapsed to run FPDC on 50 iterations is between 0.12 and 3.4 seconds. The time has been measured by using Matlab R2007a on a Mac os X[2].

The first simulated dataset is composed by 2 normal distributed clusters, each cluster is composed by 200 elements and 3 variables. fig. 5.1a. FPDC catch the right clustering structure. In fig. 5.1b the two clusters are normally distributed, the within variance of the red cluster is 2, the within variance of the blu cluster is 0.5, in this case there is 1 misclassified unit. In 5.1c the two clusters are normally distributed, the red cluster is composed by 200 elements, the blu one by 100 elements, 1 misclassified unit is obtained. In 5.1d there is correlation between variables and there are not misclassified units.

[2]Snow leopard, Ram 4gb 1067Mhz DDR3, processor 2.26 GHz Intel Core 2 Duo



(a) 2 clusters

(b) variance changing

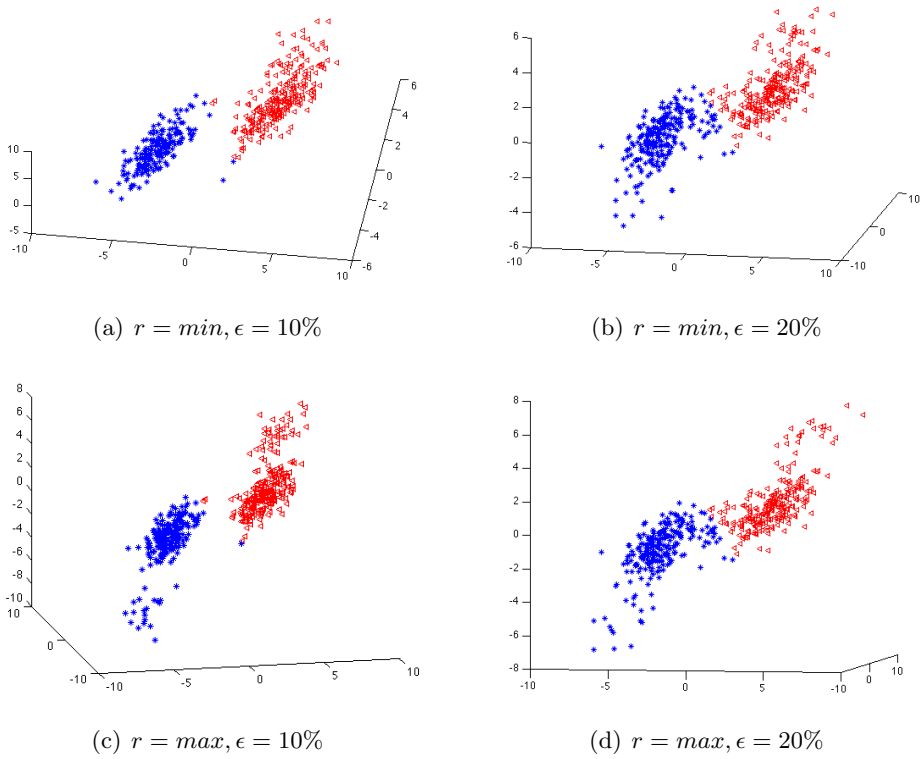
(c) different number of elements

(d) variables correlated

Figure 5.1: Clustering obtained applying FPDC on the dataset composed by 2 clusters and 3 variables

In the dataset in fig. 5.2 outliers have been added, with a 10% of outliers the number of the misclassified units is between 4 and 5. By adding 20% of outliers the number of misclassified units is 5.

Fig. 5.3 represents the results obtained by applying FPDC on the dataset composed by 6 variables. The points are plotted on the first two



(a) $r = min, \epsilon = 10\%$

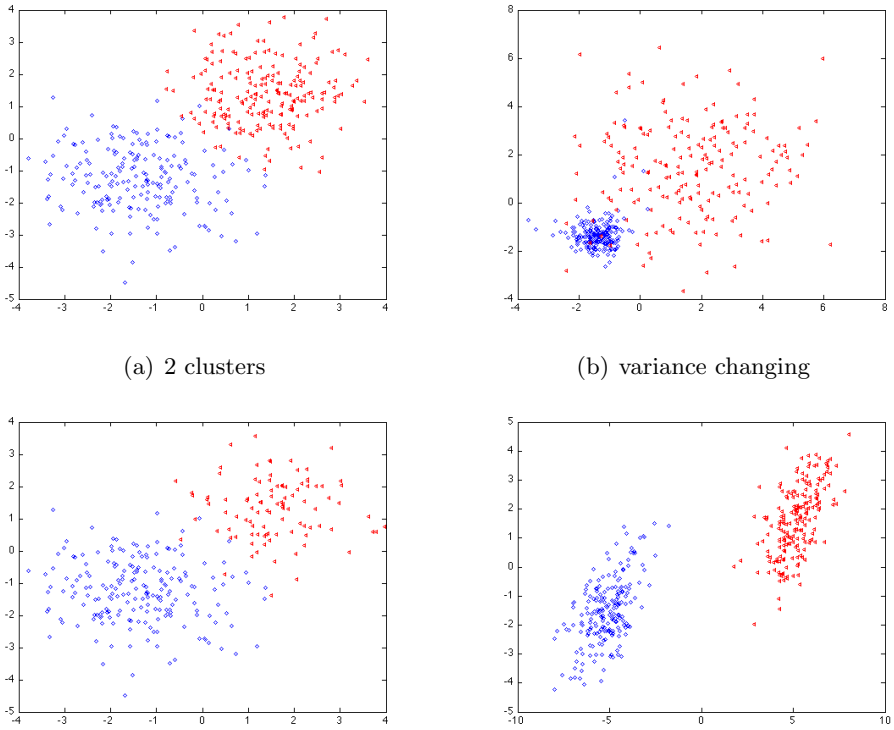
(b) $r = min, \epsilon = 20\%$

(c) $r = max, \epsilon = 10\%$

(d) $r = max, \epsilon = 20\%$

Figure 5.2: Clustering obtained applying FPDC on the dataset composed by 2 clusters and 3 correlated variables

axes, to see the results on the six axes refers to Appendix B. In fig. 5.3a the two clusters are normally distributed and there are not misclassified units. In fig. 5.3b the two clusters are normally distributed, the within variance of the red cluster is 2, the within variance of the blu cluster is 0.5, there are 8 misclassified units. In the other cases, clusters composed by a different



(a) 2 clusters

(b) variance changing

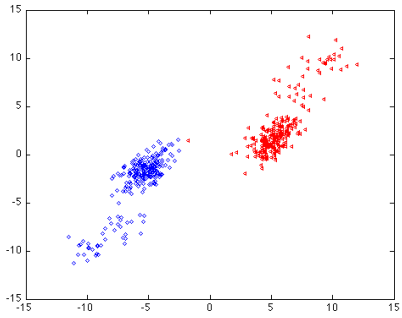
(c) different number of elements

(d) variables correlated

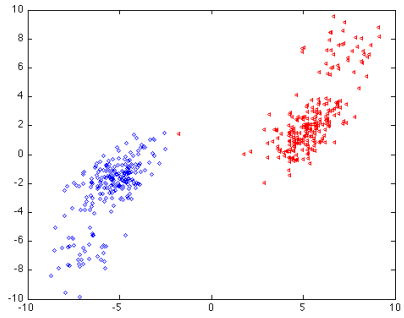
Figure 5.3: Clustering obtained applying FPDC on the dataset composed by 2 clusters and 6 variables, scatter plot of two first dimentions.

number of elements and in presence of the correlation among the variables, the algorithm catches the right clustering structure.

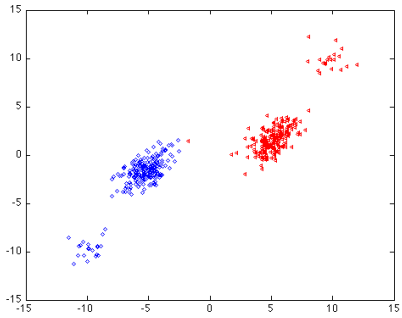
In the dataset In fig. 5.4 outliers have been added; each cluster has been contaminated at a level ϵ equal to 10% in fig. a and c, 20% in b and d. In all cases the algorithm catch the right clustering structure.



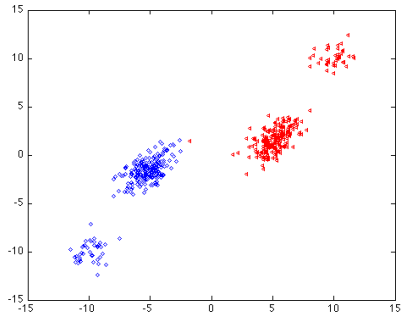
(a) $r = min, \epsilon = 10\%$



(b) $r = min, \epsilon = 20\%$



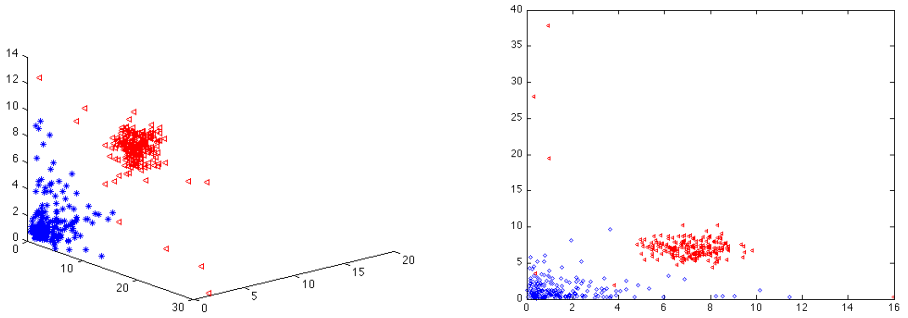
(c) $r = max, \epsilon = 10\%$



(d) $r = max, \epsilon = 20\%$

Figure 5.4: Clustering obtained applying FPDC on the dataset composed by 2 clusters and 6 correlated variables, scatter plot of two first dimentions.

In fig. 5.5 the red clusters are normally distributed, the blu ones are log-normally distributed; each cluster is composed by 200 elements and the within-cluster variance is 1. In the first case points are in a 3 dimensional space and there are 18 misclassified points, 5.5a. In the second case the original space is a 6 dimensional space, 6 misclassified units have been ob-



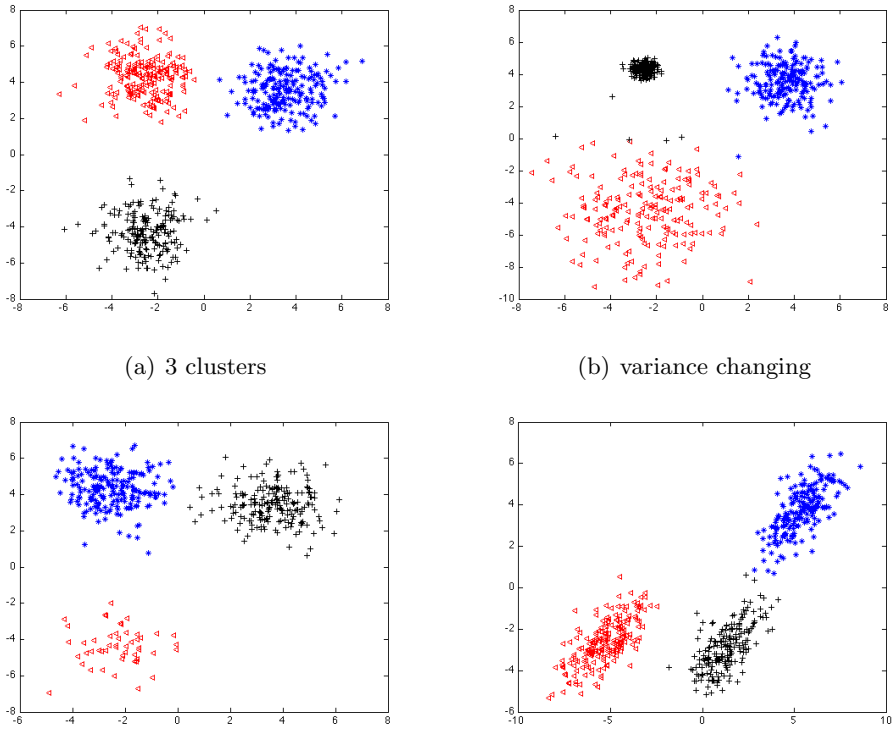
(a) 3 variables

(b) 6 variables

Figure 5.5: Clusters of different shape; red clusters generated according to a normal distribution; blu clusters generated according to a lognormal distribution

tained; in fig. 5.5b are shown clusters obtained on the first two factorial axes, to see the results on all factorial axes refers to Appendix B.9. The relatively high number of misclassified units is due to the presence of points far from the centers, belonging to the log-normally generated cluster. However, their belonging probability to the wrong cluster is low, between 40% and 49% and in the worst case the error rate is 4.5%.

The dataset in fig. 5.6 is composed by 2 variables; In fig. 5.6a the 3 clusters are normally distributed and well clustered. In fig. 5.6b the within-cluster variance, v_k changes among clusters: red $v_1 = 2$, blu $v_2 = 1$, black $v_3 = 0.5$. The number of misclassified units is 6. In fig. 5.6c the number of elements, n_k changes among clusters: red $n_1 = 100$, blu $n_2 = 200$, black $n_3 = 150$; there are 3 misclassified units. In fig. 5.6d there is 1 misclassified unit, in this case variables are correlated. In the worst case the error rate is 1%.



(a) 3 clusters

(b) variance changing

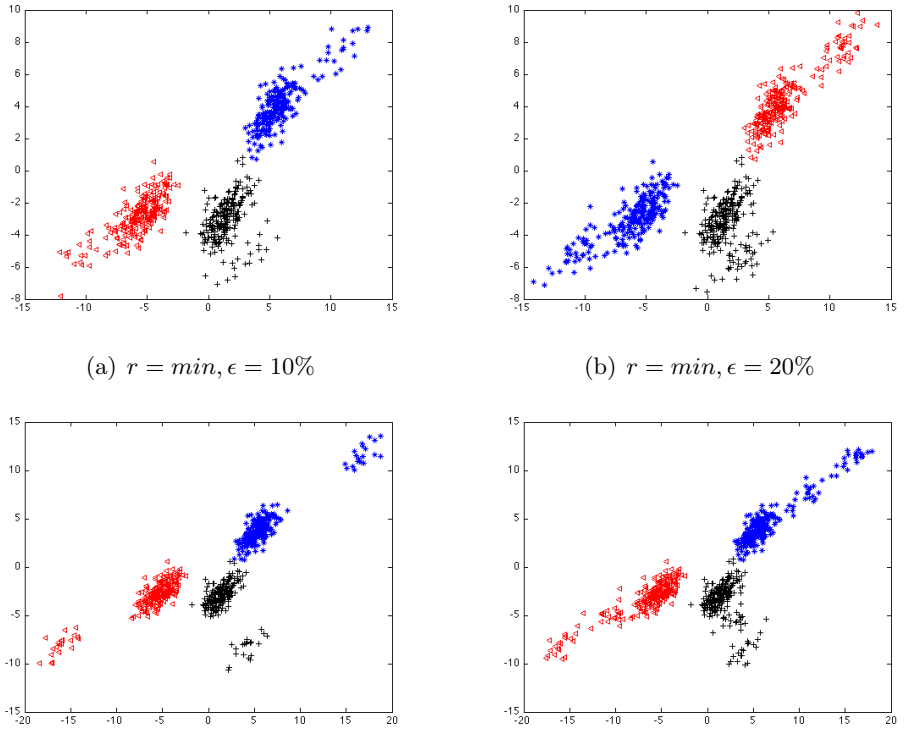
(c) different number of elements

(d) variables correlated

Figure 5.6: Clustering obtained applying FPDC on the dataset composed by 3 clusters and 2 variables

In the dataset represented in fig. 5.7 outliers have been added, the contamination level ϵ is equal to 10% in fig. a and c, 20% in b and d. The number of misclassified points is 2.

Four normally distributed cluster are represented in fig. 5.8. By varying the value within-cluster variance, v_k, among clusters ($v_1 = v_2 = 2$ in clusters



(a) $r = min, \epsilon = 10\%$

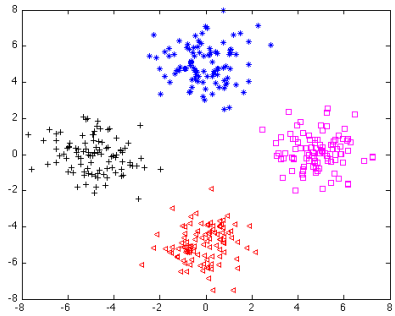
(b) $r = min, \epsilon = 20\%$

(c) $r = max, \epsilon = 10\%$

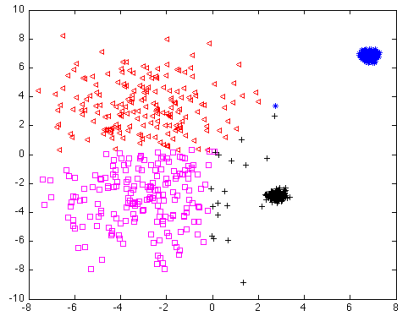
(d) $r = max, \epsilon = 20\%$

Figure 5.7: Clustering obtained applying FPDC on the dataset composed by 3 clusters and 2 correlated variables

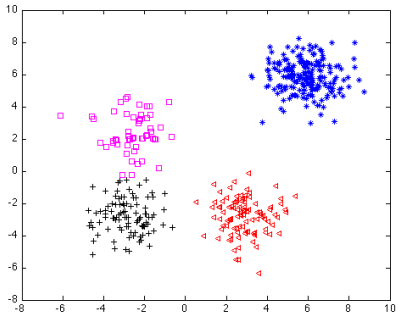
red and magenta, $v_3 = v_4 = 0.5$ in clusters blue and black) 8 misclassified units have been obtained, fig. 5.8b. Varying the number of elements, n_k, among clusters (magenta $n_1 = 50$, black and red $n_2 = n_3 = 100$, blue $n_4 = 200$) 4 misclassified points have been obtained, fig. 5.8c. In case of correlated variables, fig. 5.8d, the number of misclassified units is 4.



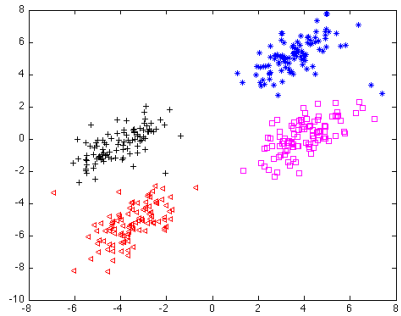
(a) 4 clusters



(b) variance changing



(c) different number of elements

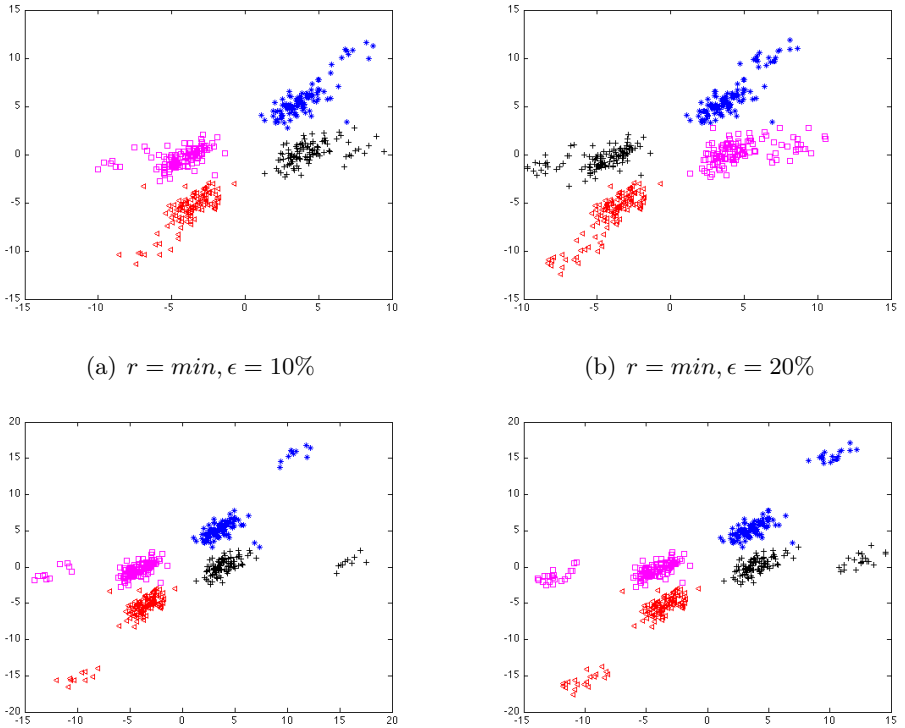


(d) variable correlated

Figure 5.8: Clustering structure obtained on the dataset composed by 4 clusters and 2 variables

In dataset in fig. 5.9 outliers have been added, with 10% of outliers the number of misclassified units is between 3 and 4. By adding 20% of outliers the number of misclassified points is 3.

The graphic 5.10 shows the value of the JDF at each iteration of FPDC on 50 iterations. The first iteration is not counted. The convergence of the algorithm has not been analytically demonstrated. In the simulation study



(a) $r = min, \epsilon = 10\%$

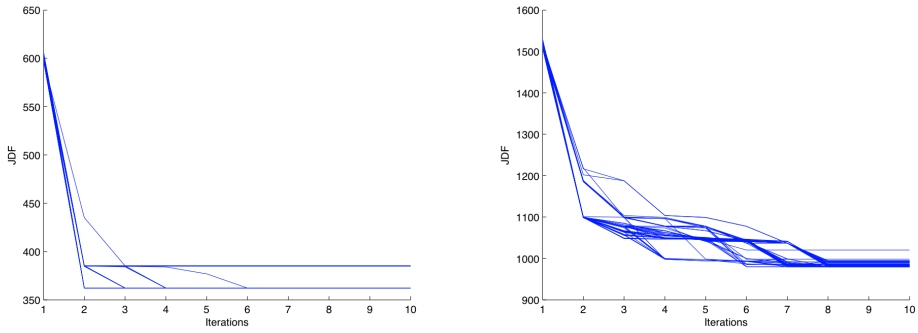
(b) $r = min, \epsilon = 20\%$

(c) $r = max, \epsilon = 10\%$

(d) $r = max, \epsilon = 20\%$

Figure 5.9: Clustering obtained applying FPDC on the dataset composed by 4 clusters and 2 correlated variables

the algorithm always converges to the same solution. The convergence is not shown for all the datasets for the sake of brevity, the algorithm always converges to the same solution with the exception of dataset showed in fig 5.8c. In this case some local minima have been obtained. The value of the JDF at each iteration of the algorithm is shown in 5.10a. In order to show



(a) 2 variables 4 clusters

(b) 5 variables 4 clusters

Figure 5.10: JDF behavior at each iteration of FPDC algorithm obtained along 50 iterations on two simulated datasets

another example of convergence problems a simulated dataset has been created. It is a 5 dimensional dataset composed by 4 normally distributed clusters. Variables are correlated and there is a contamination of $\epsilon = 20\%$ and $r = max$. At each iteration of the algorithm the value of the JDF has been measured. The results are shown in fig. 5.10b. In both cases the algorithm converges to a minimum.

Summarizing, the algorithm works well in presence of outliers, it can detect the right clustering structure in presence of 20% of outliers. The difference of variance among clusters does not affect the algorithm efficiency. Looking at two clusters dataset, although the error rate is still relatively low, it can be noticed a significant difference between the case of 2 variables and 6 variables. In the first case a dimensionality reduction is not needed and all dimension are taken into account to obtain the partition. Dealing with the 6 dimensional dataset, clustering is applied on few dimensions, this allows to avoid that ground noise interferes with the analysis and to

improve the algorithm performance. In practice FPDC is applied only on large datasets, in other cases a factorial clustering algorithm is not needed. Classical PD-clustering becomes less efficient when the number of elements in each cluster is different, FPDC results are not affected by this issue. The worst case is obtained on the dataset made by 4 clusters where the error rate is 0.89%.

The biggest error rate is obtained on the dataset where one cluster is normally distributed and the other one is log-normally distributed. In case of two variables dataset the error rate obtained is 4.5%, on the dataset made by 6 variables the error rate becomes 1.5%. As the number of variable increases the performance of the algorithm improves.

Conclusions

The increasing diffusion of large datasets in many fields has lead to the development of new clustering techniques that face problems connected to high dimensional datasets. Classical clustering techniques, indeed, can become unstable dealing with such a kind of datasets. One of the first solution has been the use of two-step clustering methods. These methods reduce the dimensionality of the dataset before applying a clustering method. An evolution of these techniques was possible by the improvement in computational capabilities; it consists in iterative two-step methods, or factorial methods, that perform a factorial technique and a clustering technique iteratively, by optimizing a common criterion.

In this thesis a new factorial two-step clustering method has been brought up: Factorial Probabilistic Distance Clustering (FPDC). It performs iteratively a linear transformation of data and Probabilistic Distance clustering (PD-clustering). PD-clustering is an iterative, distribution free, probabilistic, clustering method. Clusters centers are computed according to the constraint that the product between the probability and the distance of each point to any cluster center is a constant. An iterative algorithm allows to compute the centers. When the number of variables is large and variables are correlated PD-Clustering becomes unstable and the correlation between

variables can hide the real number of clusters. A linear transformation of original variables into a reduced number of orthogonal ones using common criteria with PD-Clustering can significantly improve the algorithm performance. In the thesis it has been demonstrated that Tucker3 transformation of the distance matrix permits to obtain a subspace that optimizes the same criterion of PD-clustering. PD-clustering is applied on the projection of units in this new space where clusters are better separated. The method can be summarized as: i) random initialization of centers matrix ii) computation of distances matrix iii) Tucker 3 decomposition of distances matrix iv) PD-clustering of Tucker 3 factors. Steps ii-iv are iterated until convergence is obtained.

Factorial PD-clustering allows to work with large datasets improving the stability and the robustness of the method. A computational study on several simulated dataset have been done in order to test FPDC performance facing some issues. The method performs well in presence of outliers, it can detect the right clustering structure in presence of 20% of outliers. Difference of variance among clusters does not affect the algorithm efficiency. Classical PD-clustering becomes less efficient when the number of elements in each cluster is different, FPDC results are not affected by this issue. To test whether different shaped clusters are detected by the method, two dataset have been generated, satisfactory results have been obtained. The simulation study has the aim to test the effect of different number of variables on the results too. The results obtained show that as the number of variables increase the number of misclassified units decrease.

Results presented in this thesis have been obtained applying FPDC on quantitative datasets. Some applications of FPDC on binary data have lead to interesting results, however the method can not be applied directly on

categorical datasets. An important issue in future researches is the FPDC generalization to the case of categorical data. Dealing with big nominal and binary data matrices, the sparseness of data and the non-linearity in the association can be more prejudicial to the overall cluster stability.

Appendix A

Routines in Matlab Language

A.1 Support Vector Clustering

For the kernel computation the toolbox SVMKM has been used[1].

```
function  [beta,sv,label,r2,a,b]= svc(xapp,kernel,kerneloption,Cpen,betaall)
%Support vector clustering using cone cluster labeling
%%%%%%%%%INPUT%%%%%%%%%
% xapp input data
%Kernel  kernel function
%kerneloption kernel parameters
% Cpen between 0 and 1, weight for outliers
%betaall sum of beta
%kernel     : kernel function
%       Type                        Function                Option
%       Polynomial                  'poly'                  Degree (<x,xsup>+1)^d
%       Homogeneous polynomial      'polyhomog'             Degree <x,xsup>^d
%       Gaussian                    'gaussian'              Bandwidth
%       Heavy Tailed RBF            'htrbf'                 [a,b]   %see
%                                                           Chappelle  1999
%       Mexican 1D Wavelet          'wavelet'
%       Frame kernel                'frame'             'sin','numerical'...
%
%  kerneloption : scalar or vector containing the option for the kernel
% 'gaussian' : scalar gamma is identical for all coordinates
%              otherwise is a vector of length equal to the number of
%              coordinate
```

[1] http://asi.insa-rouen.fr/enseignants/ arakotom/toolbox/index.html

```
%
%
% 'poly' : kerneloption is a scalar given the degree of the polynomial
%          or is a vector which first element is the degree of the polynomial
%           and other elements gives the bandwidth of each dimension.
%          thus the vector is of size n+1 where n is the dimension of the
%          problem.

%initialization
q=kerneloption;
if nargin < 6
    betaall=[];
end;
if nargin < 5
     Cpen=inf;
end;

%computation of kernel function
if ¬isempty(xapp);

    [kapp]=svmkernel(xapp,kernel,kerneloption);
    if  Cpen≠0
        kapp=kapp+(1/ Cpen)*eye(size(kapp));
    end;
elseif isfield(kerneloption,'matrix');
    kapp=kerneloption.matrix+(1/ Cpen)*eye(size(kerneloption.matrix));
else
    error('No ways for processing the radius. Check inputs...');
end;

%computation of beta
size(kapp)
D=diag(kapp);
A = ones(size(D));
b=1;
lambda=1e-7;
verbose=0;
size(kapp)
[betaaux,lagrangian,pos]=monqpCinfty(2*kapp,D,A,b,lambda,verbose,[],[],betaall);
beta=zeros(size(D));
beta(pos)=betaaux;
r2=-betaaux'*kapp(pos,pos)*betaaux + betaaux'*D(pos);
n=length(kapp);

%support vectors computation
nsv=size(pos,1);
sv=zeros(nsv,size(xapp,2));
for i=1:nsv
    sv(i,:)=xapp(pos(i),:);
end

%graphics
if size(xapp,2)==2
    figure
    plot(sv(:,1),sv(:,2),'r*')
    hold on
    plot(xapp(:,1),xapp(:,2),'k.')

elseif size(xapp,2)==3
    figure
    plot3(sv(:,1),sv(:,2),sv(:,3),'ro')
    hold on
    plot3(xapp(:,1),xapp(:,2),xapp(:,3),'b.')
elseif size(xapp,2)==4
    figure
```

```
    plot(sv(:,1),sv(:,2),'r*')
    hold on
    plot(xapp(:,1),xapp(:,2),'k.')

    figure
    plot(sv(:,3),sv(:,4),'r*')
    hold on
    plot(xapp(:,3),xapp(:,4),'k.')
elseif size(xapp,2)==5
    figure
    plot(sv(:,1),sv(:,2),'r*')
    hold on
    plot(xapp(:,1),xapp(:,2),'k.')

    figure
    plot3(sv(:,1),sv(:,2),sv(:,3),'r*')
    hold on
    plot3(xapp(:,1),xapp(:,2),xapp(:,3),'k.')
elseif size(xapp,2)==6
    figure
    plot3(sv(:,1),sv(:,2),sv(:,3),'r*')
    hold on
    plot3(xapp(:,1),xapp(:,2),xapp(:,3),'k.')

    figure
    plot3(sv(:,4),sv(:,5),sv(:,6),'r*')
    hold on
    plot3(xapp(:,4),xapp(:,5),xapp(:,6),'k.')
else
    figure
    plot3(sv(:,1),sv(:,2),sv(:,3),'r*')
    hold on
    plot3(xapp(:,1),xapp(:,2),xapp(:,3),'k.')
end


%adiacency matrix
z=sqrt((-log(sqrt((1-r2)))/q));
ls=size(sv,1);
a=zeros(ls,ls);
%i=1;
%j=1;
for i=1:ls
    for j=1:ls
        if (norm(sv(i,:)-sv(j,:)))^2<2*z
            a(i,j)=1;
        else
            a(i,j)=0;
        end;
    end;
end;


%transformation of adiaceency matirx

b=zeros(ls,ls);
for i=1:ls
    k=1;
    for j=1:ls
        if i≠j
            if a(i,j)==1
                b(i,k)=j;
                k=k+1;
            end;
        end;
```

```
    end;
end;

% clusters computation using cone cluster labeling
c=zeros(ls,ls);
d=ones(1,ls);
e=zeros(1,ls);

idc=0;

 while isequal(d,e)==0;
    idc=idc+1;
    v=find(d);
    i=v(1);
    c(idc,1)=i;
    j=1;
    d(i)=0;
       while((j<(ls))&&( b(i,j)≠0))
        c(idc,j+1)=b(i,j);
        j=j+1;
       end;
       l=2;
    while ((c(idc,l)≠0)&&(l<ls))
         i=c(idc,l);
         d(i)=0;
         k=1;
         while ((b(i,k)≠0)&&(k<ls))
              if notin(b(i,k),c(idc,:))
                   c(idc,j+1)=b(i,k);
              end
              k=k+1;
         end
         l=l+1;
    end
 end
%labeling
lab=zeros(ls,1);
for k=1:ls;
    for i=1:ls;
         for j=1:ls;
              if c(i,j)==k
                   lab(k)=i;
              end;
         end;
    end;
end;
label=zeros(n,1);
 for i=1:n
       for j=1:ls
            dist(j)=norm(xapp(i,:)-sv(j,:));
       end
       [y,ji]=min(dist);
       label(i)=lab(ji);
 end
```

A.2 Factorial PD-clustering

In order to apply Tucker 3 decomposition Tucker3 function in MatLab Toolbox N-way [Chen, 2010] has been used.

```
%TUCKER multi-way tucker model
%
% function [Factors,G,ExplX,Xm]=
% =tucker(X,Fac[,Options[,ConstrF,[ConstrG[,Factors[,G]]]]]);
%
% Change: True LS unimodality now supported.
%
% This algorithm requires access to:
% 'fnipals' 'gsm' 'inituck' 'calcore' 'nmodel' 'nonneg' 'setopts' 'misssum'
% 'missmean' 't3core'
%
% See also:
% 'parafac' 'maxvar3' 'maxdia3' 'maxswd3'
%
% ------------------------------------------------------------
%              The general N-way Tucker model
% ------------------------------------------------------------
%
% [Factors,G,ExplX,Xm]=tucker(X,Fac,Options,ConstrF,ConstrG,Factors,G);
% [Factors,G,ExplX,Xm]=tucker(X,Fac);
%
% INPUT
% X         : The multi-way data array.
% Fac       : Row-vector describing the number of factors
%             in each of the N modes. A '-1' (minus one)
%             will tell the algorithm not to estimate factors
%             for this mode, yielding a Tucker2 model.
%             Ex. [3 2 4]
%
% OPTIONAL INPUT
% Options   : See parafac.
% ConstrF   : Constraints that must apply to 'Factors'.
%             Define a row-vector of size N that describes how
%             each mode should be treated.
%             '0' orthogonality (default)
%             '1' non-negativity
%             '2' unconstrained
%             '4' unimodality and non-negativity.
%             E.g.: [0 2 1] yields ortho in first mode, uncon in the second
%             and non-neg in the third mode.
%             Note: The algorithm uses random values if there are no
%             non-negative components in the iteration intermediates. Thus,
%             if non-negativity is applied, the iterations may be
%             non-monotone in minor sequences.
% ConstrG   : Constraints that must apply to 'G'.
%             '[]' or '0' will not constrain the elements of 'G'.
%             To define what core elements should be allowed, give a core that
%             is 1 (one) on all active positions and zero elsewhere - this boolean
%             core array must have the same dimensions as defined by 'Fac'.
%
% OUTPUT
% Factors   : A row-vector containing the solutions.
% G         : Core array that matches the dimensions defined by 'Fac'.
```

```
% ExplX     : Fraction of variation (sums of squares explained)
% Xm        : Xhat (the model of X)
%
% This algorithm applies to the general N-way case, so
% the array X can have any number of dimensions. The
% principles of 'projections' and 'systematic unfolding
% methodology (SUM)' are used in this algorithm to provide
% a fast approach - also for larger data arrays. This
% algorithm can handle missing values if denoted
% by NaN's. It can also be used to make TUCKER2/1 models by
% properly setting the elements of 'Fac' to -1.
%
% Note: When estimating a Tucker model on data using non-orthogonal factors,
%       the sum of square of the core may differ between models of the
%       same dataset. This is in order since the factors may
%       thus be correlated. However, the expl. var. should always be the same.
%
```

A.2.1 PD-clustering

```
function [l,cnew,p,dis,cont,deu,JDF]=pdcluster(data,k)
% Cluster the data whit d-clustering Algoritmh
%
%
%%%%%%%INPUT%%%%%%%%%
%data=input data
%k=number of cluster
%
%
%%%%%%%OUTPUT%%%%%%%%
%cnew=cluster's center
%l=class label
%p=nxk matrix probability to belong to each class
%cont=number of iterations until convergence
%JDF jonit distance function


%center intialization
cnew(1,:)=min(data);
cnew(2,:)=max(data);
for i=3:k
cnew(i,:)=rand(1,size(data,2));
end
n=size(data,1);
ver=100;
cont=0;


while(ver>0.001 && cont<1000);%( ver>eps )nobb≤obb
    cont=cont+1;
    c=cnew;
    %STEP 1
    %distance matrix computation
    for i=1:n;
        for j=1:k;
            dis(i,j)=norm(data(i,:)-c(j,:));
        end;
    end;
    %STEP 2
    %center matrix computation
    nu2=ones(n,k);

    for  i=1:n;
        for h=1:k;
            for j=1:k;
                if j≠h;
                    nu2(i,h)=dis(i,j)*nu2(i,h);
                end

            end
        end
    end
    deu=sum(nu2,2);
    %probability matrix computation
    for i=1:n
       for j=1:k
           p(i,j)=nu2(i,j)/deu(i);
            u(i,j)=p(i,j)^2/dis(i,j);
        end
```

```
      end
     som=sum(u,1);
     for   j=1:k
          par=zeros(1,size(data,2));
          for  i=1:n
              par=par+(u(i,j)/som(j))*data(i,:);
          end
          cnew(j,:)=par;
     end
     %STEP 3
     %check convergence

     for  j=1:k
     ver1(j)=norm(cnew(j,:)-c(j,:));
     end
     ver=sum(ver1);

end
if  cont==1000
h = warndlg({'Convergence not reached'}); %#ok<NASGU>
end
%Points assigned to the closer center
[m,l]=max(p,[],2);
for  i=1:n
JDF=dis(i,1)'*p(i,1);
end

end
```

A.2.2 Number of Tucker3 factors

```
function [expleined]=FattoriTucker(dataO,nc,mi)
%heuristic approach to cope with the choice of Tucker 3 factors.
%we choose the minimum number of factors that corresponds to a significant
%value of the explained variability.
%
%%%%%%%%%INPUT%%%%%%%%%%
%dataO dataset
% nu number of clusters
%mi minimum number of factors
%%%%%%%%%OUTPUT%%%%%%%%%%
%expleined expleined variability


%if not specifyed minimum number of factors is 1
if nargin<3
    mi=1;
end

[n,ppp]=size(dataO);
if(ppp>1000)
    mi=2;
end
m1=0;
m2=0;
%data standardization
data=zscore(dataO);

%Computation of distance matrix G
[la,c]=dcluster1ALG(data,nc);
dist=[];
    ddt=[];
    dd=[];
    for j=1:nc;
        for i=1:n;
            dd(i,:)=abs(data(i,:)-c(j,:));
            dist(i,j)=norm(data(i,:)-c(j,:));
        end;
        ddt=[ddt;dd];
    end;
tway=reshape(ddt,n,ppp,nc);
%nf= number of factors for variables
%nu= number of factors for units

%graph settings
expleined=zeros(8,3);

nf=4;
expleined(1:8,2)=4;
expleined(1,1)=round(n/2);
expleined(4,1)=5;
expleined(5,1)=4;
expleined(6,1)=3;
expleined(7,1)=2;
expleined(8,1)=1;

%Computation of explained variability
nu=round(n/2);
dim=[nu,nf,nc-1]; %Computation of explained variability
[tuk,g,expl]=tucker1(tway,dim); %tucker3decomposition
```

```
expleined(1,3)=expl;%explaned variability is stored

%Computation of explained variability
cont=3;
for i=5:-1:mi
    cont=cont+1;
    nu=i;
    dim=[nu,nf,nc-1];%Computation of explained variability
    [tuk,g,expl]=tucker1(tway,dim);%tucker3decomposition
    expleined(cont,3)=expl;%explaned variability is stored
end


%graph settings
 nf=3;
expleined(9:16,2)=3;
expleined(9,1)=round(n/2);
expleined(12,1)=5;
expleined(13,1)=4;
expleined(14,1)=3;
expleined(15,1)=2;
expleined(16,1)=1;

nu=round(n/2);
dim=[nu,nf,nc-1];
[tuk,g,expl]=tucker1(tway,dim);
expleined(9,3)=expl;


cont=11;
for i=5:-1:mi
    cont=cont+1;
    nu=i;
    dim=[nu,nf,nc-1];
    [tuk,g,expl]=tucker1(tway,dim);
    expleined(cont,3)=expl;
end


%graph settings
 nf=2;
expleined(17:24,2)=2;
expleined(17,1)=round(n/2);
expleined(20,1)=5;
expleined(21,1)=4;
expleined(22,1)=3;
expleined(23,1)=2;
expleined(24,1)=1;

%Computation of explained variability
nu=round(n/2);
dim=[nu,nf,nc-1];
[tuk,g,exp]=tucker1(tway,dim);
expleined(17,3)=exp;

%Computation of explained variability
cont=19;
for i=5:-1:mi
    cont=cont+1;
    nu=i;
    dim=[nu,nf,nc-1];
    [tuk,g,expl]=tucker1(tway,dim);
    expleined(cont,3)=expl;
end
```

```
%Computation of explained variability
nf=2;
if round(n/8)>5
expleined(19,1)=round(n/8);
nu=round(n/8);
dim=[nu,nf,nc-1];
[tuk,g,expl]=tucker1(tway,dim);
expleined(19,3)=expl;
else
    expleined(19,:)=[];
end

%Computation of explained variability
if round(n/4)>5;
expleined(18,1)=round(n/4);
nu=round(n/4);
dim=[nu,nf,nc-1];
[tuk,g,expl]=tucker1(tway,dim);
expleined(18,3)=expl;
else
    expleined(18,:)=[];
end

%Computation of explained variability
nf=3;
if round(n/8)>5
expleined(11,1)=round(n/8);
nu=round(n/8);
dim=[nu,nf,nc-1];
[tuk,g,expl]=tucker1(tway,dim);
expleined(11,3)=expl;
else
    m1=1;
    expleined(11,:)=[];
end

%Computation of explained variability
if round(n/4)>5;
expleined(10,1)=round(n/4);
nu=round(n/4);
dim=[nu,nf,nc-1];
[tuk,g,expl]=tucker1(tway,dim);
expleined(10,3)=expl;
else
    m2=1;
    expleined(10,:)=[];
end

%Computation of explained variability
nf=4;
if round(n/8)>5
expleined(3,1)=round(n/8);
nu=round(n/8);
dim=[nu,nf,nc-1];
[tuk,g,expl]=tucker1(tway,dim);
expleined(3,3)=expl;
else
    expleined(3,:)=[];
end

%Computation of explained variability
if round(n/4)>5;
expleined(2,1)=round(n/4);
nu=round(n/4);
dim=[nu,nf,nc-1];
```

```
[tuk,g,expl]=tucker1(tway,dim);
expleined(2,3)=expl;
else
    expleined(2,:)=[];
end


%graph settings
if m1==1;
    ascisse =(5:-1:1);
    ascisse=[10,ascisse];
    ord1=expleined(1:6,3);
    ord2=expleined(7:12,3);
    ord3=expleined(13:18,3);
elseif m2==1;
    ascisse =(6:-1:1);
    ascisse=[10,ascisse];
    ord1=expleined(1:7,3);
    ord2=expleined(8:14,3);
    ord3=expleined(15:21,3);
else
    ascisse =(7:-1:1);
    ascisse=[10,ascisse];
    ord1=expleined(1:8,3);
    ord2=expleined(9:16,3);
    ord3=expleined(17:24,3);
end

%graph
plot(ascisse, ord1,'bd-')
hold on
plot(ascisse, ord2,'gd-')
plot(ascisse, ord3,'rd-')
legend('4 foctors','3 factors','2 factors','location','SouthEast')
xlabel('number of factors for units')
title('exlpeined variability')
```

A.2.3 FPDC

```
function [la,iter,t,JDF,JDFv,expl,tuk,ddt,dist,p,c,nc]=iterDc(dataO,nc,nf,nu,s)
%%%%%%%%%%%%%%%Performs Factorial PD-Clustering%%%%%%%%%%%%%%%%%%
%%%%%%%%INPUT%%%%%%%%%
%data=data matrix  nxp
%nc= number of clusters
%nf= number of factors
%s= standardize 1=yes
%%%%%%%OUTPUT%%%%%%%%%%
%la=cluster labels
%iter= number of iterations
%t= elapsed time
%dist=distance matrix
%p=probability matrix,
%expl=explained variability
%c= center matrix
%JDF= Joint distance function
%nc= number of clusters


[n,ppp]=size(dataO);

%Data Standardization

if s==1
    data=zscore(dataO);
else
    data=dataO;
end

%initialization
t=0;
iter=0;
JDF=1;
JDFo=0;
JDFv=[];

%center initialization
 c(1,:)=min(data)+1;
 c(2,:)=max(data)-1;
 for i=3:nc
     c(i,:)=rand(1,size(data,2));
 end

%Itaration: stop if JDF stop decreasiong or 100 iterations are reached
while(abs(JDF-JDFo)>0.01 && iter<100)
    tic
    JDFo=JDF;
    JDFoo=JDFo;
    co=c;

    %Computation of distance matrix (n*nc)x p

    %computation of distances as squared norm
    dist=[];
    ddt=[];
    dd=[];
    p=zeros(n,nc);
    for j=1:nc;
        for i=1:n;
```

```
        dd(i,:)=abs(data(i,:)-c(j,:));
        dist(i,j)=norm(data(i,:)-c(j,:));%needed in probability computation
    end;
    ddt=[ddt;dd];
end;

%probability matrix computation
nu2=ones(n,nc);
for  i=1:n;
    for h=1:nc;
        for j=1:nc;
            if j≠h;
                nu2(i,h)=dist(i,j)*nu2(i,h);
            end
        end
    end

end
deu=sum(nu2,2);
for i=1:n
    for j=1:nc
        p(i,j)=nu2(i,j)/deu(i);
    end
end



%Tucker3
tway=reshape(ddt,n,ppp,nc);
dim=[nu,nf,nc-1];
[tuk,g,expl(iter)]=tucker1(tway,dim);
tuk2=tuk{2};
%data projection in the new space
tuk=data*tuk2;

%check for NaN
numc=nc;

%PD-clustering
[la,c]=dcluster1ALG(tuk,nc);
cp=c;
for i=1:numc
if (isnan(c(i,1)))==1
     numc=numc-1;
     if numc≠1
         cp(i,:)=[];
         %cambio=1;
         crim=i;
         co(crim,:)=[];
     else
         numc=numc+1;
     end
nc=numc;
end
end
c=cp;

%New center computation
c=c*tuk2';
t=t+toc;
%labeling
[mmmm,la]=max(p,[],2);

%check if all the center are different form each other
```

```
    if nc>2
    for k=1:(nc-1)
        numc=nc;
        for kk=nc:-1:(k+1)
            if abs(c(k,:)-c(kk,:))<0.000001
                numc=numc-1;
                if numc~=1
                    ct=[c(k,:);c(kk,:)];
                    c(k,:)=mean(ct);
                    c(kk,:)=[];
                else

                numc=numc+1;
                end
            end
        end
        nc=numc;
    end
    end

    %convergence criteria update
    JDF=dist(:,1)'*p(:,1);
    JDFv=[JDFv,JDF];
    %check for loop
    if abs(JDFoo-JDF)<1
        cp=c;
        for i=1:nc
            cc=[co(i,:);c(i,:)];
            cp(i,:)=mean(cc);
        end
        c=cp;

    end

end
```

Appendix B

Displays

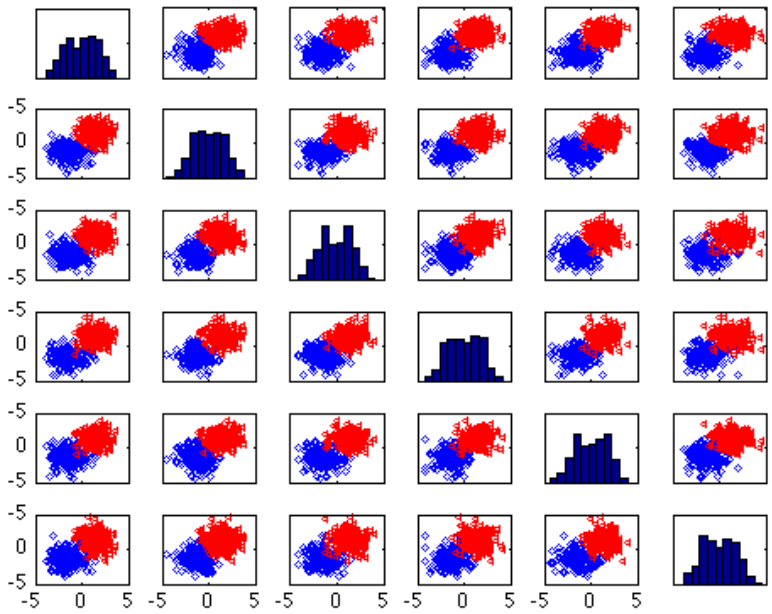


Figure B.1: Clustering structure obtained on the dataset composed by 6 variables.

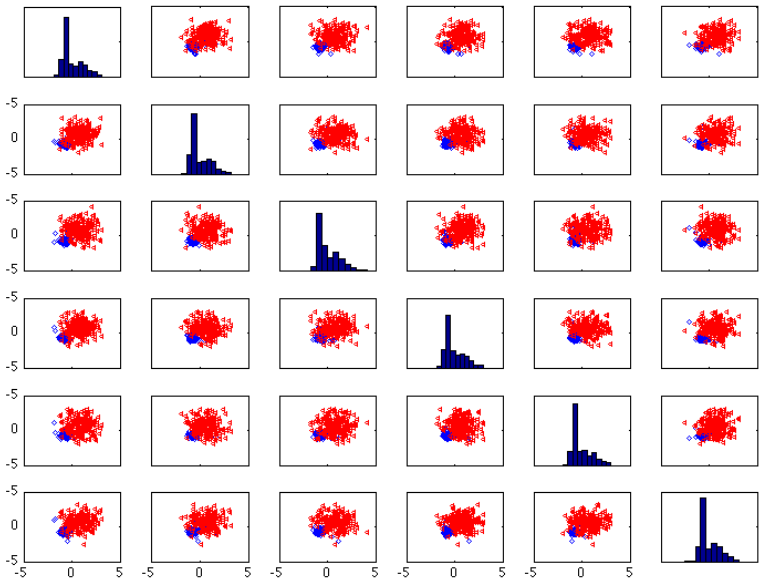


Figure B.2: Clustering structure obtained on the dataset composed by 6 variables, clusters have been normally generated with different value of variance

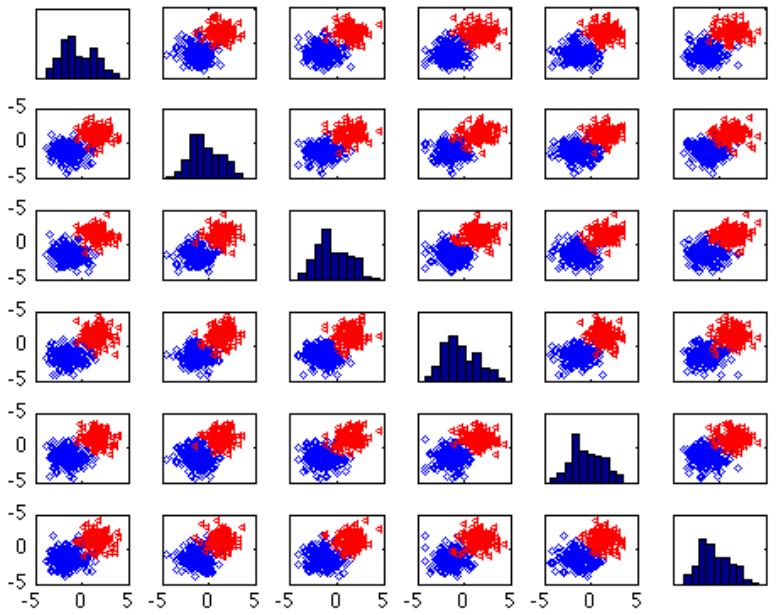


Figure B.3: Clustering structure obtained on the dataset composed by 6 variables, the clusters have been normally generated each cluster is composed by a different number of elements

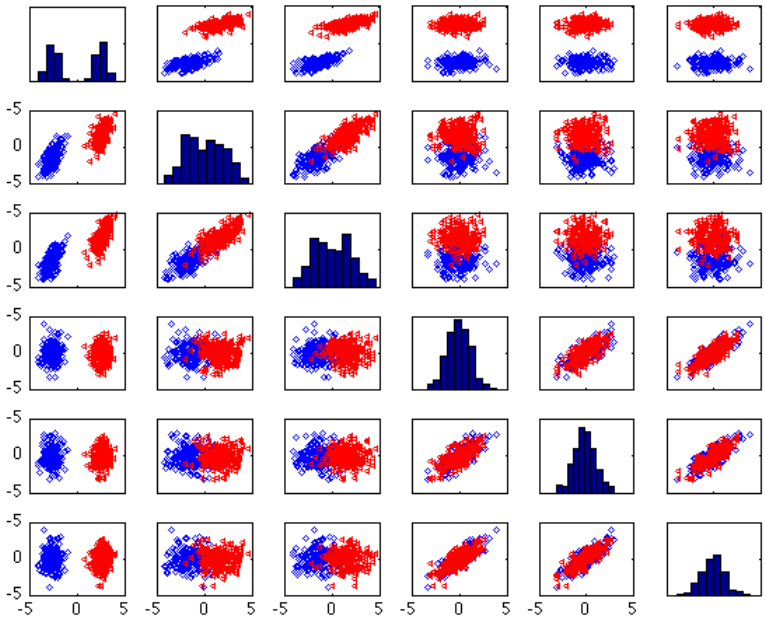


Figure B.4: Clustering structure obtained on the dataset composed by 6 variables, clusters have been normally generated variables are correlated

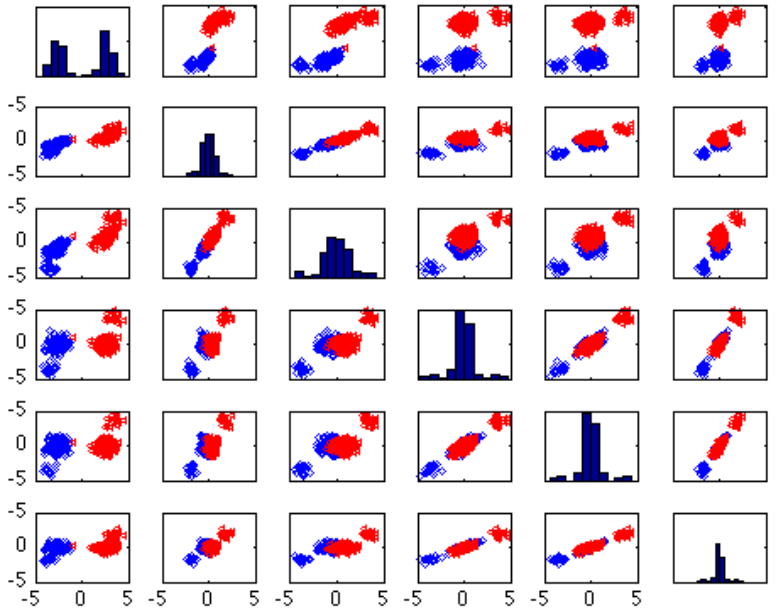


Figure B.5: Clustering structure obtained on the dataset composed by 6 variables, clusters have been normally generated variables are correlated, 10% of outliers with $r = min$

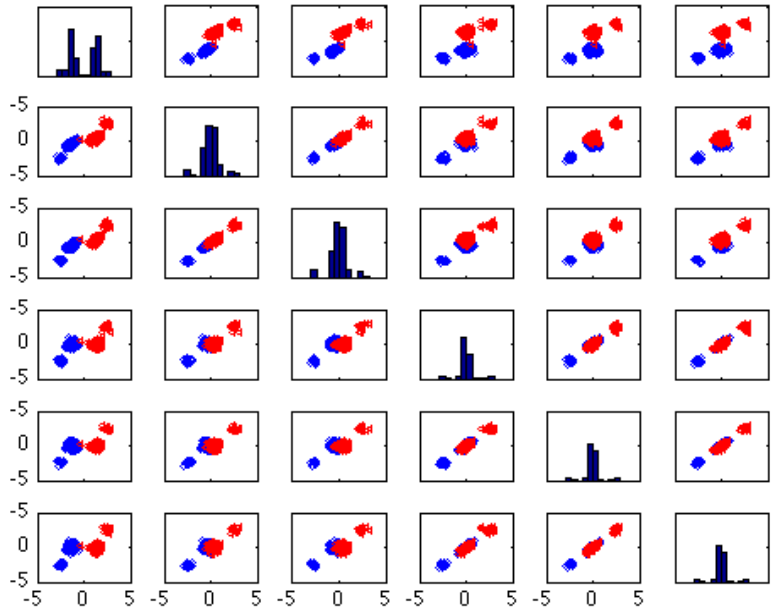


Figure B.6: Clustering structure obtained on the dataset composed by 6 variables, clusters have been normally generated variables are correlated, 10% of outliers with $r = max$

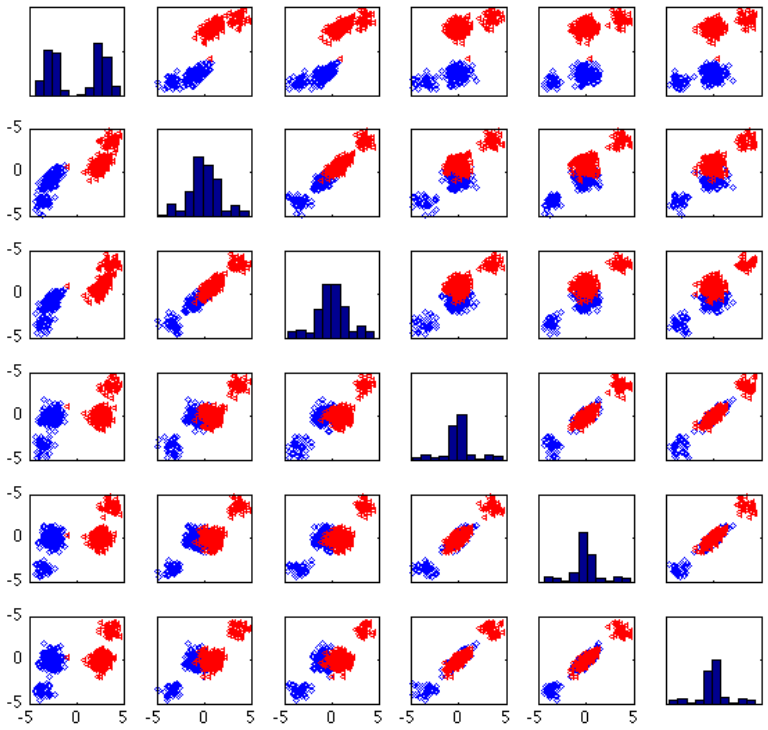


Figure B.7: Clustering structure obtained on the dataset composed by 6 variables, clusters have been normally generated variables are correlated, 20% of outliers with $r = min$

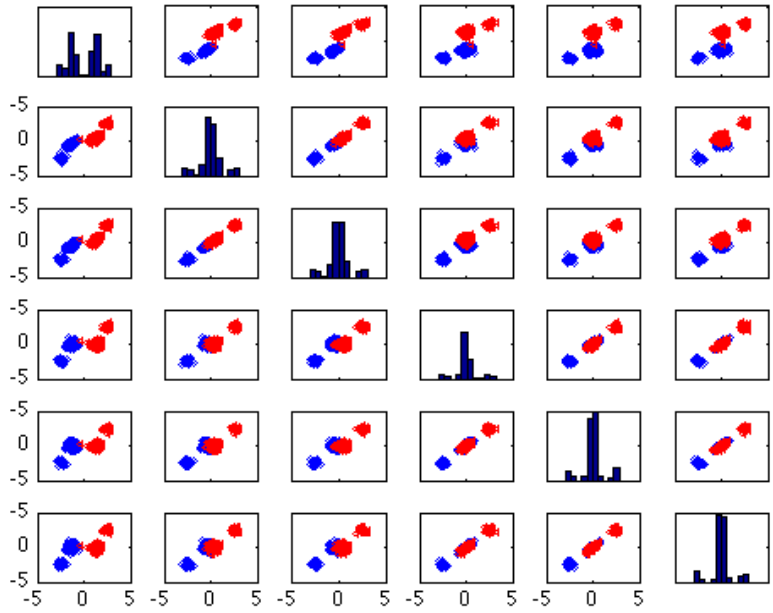


Figure B.8: Clustering structure obtained on the dataset composed by 6 variables, clusters have been normally generated variables are correlated, 20% of outliers with $r = max$

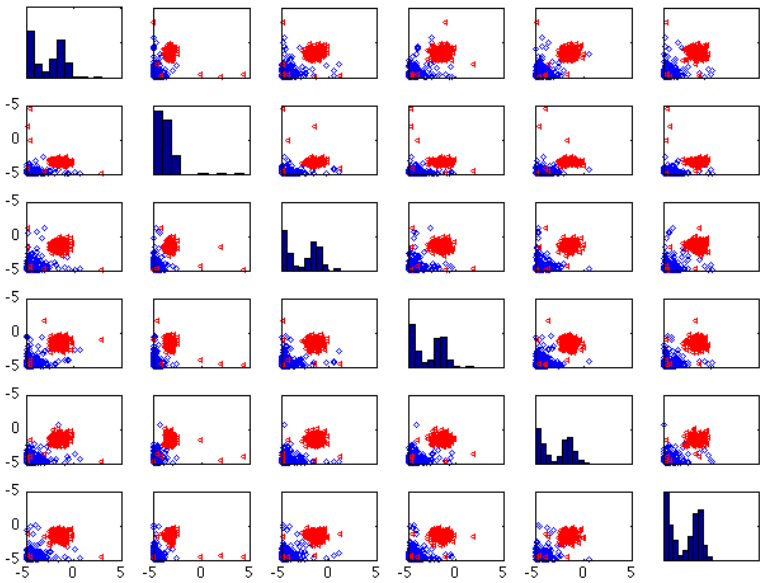


Figure B.9: Clustering structure obtained on the dataset composed by 6 variables, the first cluster has been normally genereted, the second one has been log normally generated

Bibliography

[Abe, 2005] Abe, S. (2005). *Support vector machine for pattern classification.* Springer.

[Agrawal et al., 1993] Agrawal, R., Imielinski, T., and Swami, A. (1993). Mining association rules between sets of items in large databases. *SIGMOD Rec.*, 2:207–2016.

[Arabie et al., 1996] Arabie, P., Hubert, L. J., and De Soete, G. (1996). *Clustering and Classification.* Word Scientific, River Edge, NJ.

[Arimond and Elfessi, 2001] Arimond, G. and Elfessi, A. (2001). A clustering method for categorical data in tourist market segmentatio research. *Journal of travel Research*, 39:391–397.

[Ball and Hall, 1967] Ball, G. and Hall, D. (1967). A clustering technique for summarizing multivariate data. *Behavioral Science*, 12:1099–1743.

[Bellman, 1961] Bellman, R. (1961). Adaptive control process: A guided tour. *Princeton University Press.*

[Ben-Hur et al., 2001] Ben-Hur, A., Horn, D., Siegelmann, H. T., and Vapnik, V. (2001). Support vector clustering. *Journal of machine learning research*.

[Ben-Israel and Iyigun, 2008] Ben-Israel, A. and Iyigun, C. (2008). Probabilistic d-clustering. *Journal of Classification*, 25(1):5–26.

[Benzécri, 1973] Benzécri, J. (1973). *L'analyse des Données*. Dunod.

[Benzécri, 1979] Benzécri, J. P. (1979). Sur le calcul des taux d'inertie dans l'analyse d'un questionnaire. *Le Cahiers de l'Analyse des Données*, 4(4):377–378.

[Bezdek, 1974] Bezdek (1974). Numerical taxonomy with fuzzy sets. *Journal of Mathematical Biology*, 1(1):57–71.

[Bottou and Bengio, 1995] Bottou, L. and Bengio, Y. (1995). Convergence properties of the k-means algorithms. *Advances in Neural Information Processing Systems 7*.

[Celeux, 2007] Celeux, G. (2007). Mixture models for classification. *Advances in Data Analysis*, pages 3–14.

[Chang, 1983] Chang, W. (1983). On using principal components before separating a mixture of two multivariate normal distributions. *Applied Statistics*, 32:267–275.

[Chen, 2010] Chen, H. (2010). N-way toolbox for matlab. *http://www.models.life.ku.dk/algorithms*, Accessed 20 June 2011.

[Cuesta-Albertos et al., 1997] Cuesta-Albertos, J. A., Gordaliza, A., and Matran, C. (1997). Trimmed k-means: An attempt to robustify quantizers. *The Annals of Statistics*, 25(2):553–576.

[de Leeuw et al., 1976] de Leeuw, J., Young, F., and Takane, Y. (1976). Additive structure in qualitative data: An alternating least squares method with optimal scaling features. *Psychometrika*, 41:471–503.

[De Soete and Carroll, 1994] De Soete, G. and Carroll, J. (1994). k-means clustering in a low-dimensional euclidean space. *In: Diday, E., et al. (Eds.), New Approaches in Classification and Data Analysis. Springer, Heidelberg*, pages 212–219.

[DeSarbo et al., 1990] DeSarbo, W. S., Jedidi, K., Cool, K., and Schendel, D. (1990). Simultaneous multidimensional unfolding and cluster analysis: An investigation of strategic groups. *Marketing Letters*, 2:129–146.

[Dhillon et al., 2004] Dhillon, I., Guan, Y., and Kulis, B. (2004). Kernel k-means: spectral clustering and normalized cuts. *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 551–556.

[Diday, 1971] Diday, E. (1971). La méthode des nuées dynamiques. *Revue de Statistique Appliquée*, 19(2):19–34.

[Driver and Kroeber, 1932] Driver, H. E. and Kroeber, A. L. (1932). Quantitative expression of cultural relationships. *University of California Publications in American Archaeology and Ethnology*, 31:211–256.

[Duda et al., 2000] Duda, R. O., Hart, P. E., and Stork, D. G. (2000). *Pattern Classification*. John Wiley & Sons.

[Dunn, 1974] Dunn, J. C. (1974). Well-separated clusters and optimal fuzzy partitions. *Cybernetics and Systems: An International Journal*, 4(1):95–104.

[Edwards and Cavalli-Sforza, 1965] Edwards, A. W. F. and Cavalli-Sforza, L. L. (1965). A method for cluster analysis. *Biometrics*, 21(2):362–375.

[Everitt et al., 2011] Everitt, B. S., Landau, S., Leese, M., and Stahl, D. (2011). *Cluster Analysis*. Wiley Series in probability and statistics.

[Florek et al., 1951] Florek, K., Lukaszewicz, J., Perkal, J., Steinhaus, H., and Zubrzycki, S. (1951). Sur la liaison et la division des points d'un ensemble fini. *Colloquium Mathematicae*, 2:282–285.

[Forgy, 1965] Forgy, E. W. (1965). Cluster analysis of multivariate data : efficiency versus interpretability of classifications. *Biometrics*, 21:768–769.

[Friedman and Rubin, 1967] Friedman, H. P. and Rubin, J. (1967). On some invariant criteria for grouping data. *Journal of the American Statistical Association*, 62(320):1159–1178.

[Fritzke, 1997] Fritzke, B. (1997). The lbg-u method for vector quantization–an improvement over lbg inspired from neural networks. *Neural Processing Letters*, 5(1):35–45.

[Gordaliza, 1991] Gordaliza, A. (1991). On the breakdown point of multivariate location estimators based on trimming procedures. *Statist. Probab. Lett.*, 11(387-394).

[Gordon, 1999] Gordon, A. D. (1999). *Classification*. Chapman and Hall/CRC, Boca Raton, 2nd edition.

[Green et al., 1988] Green, P. E., Schaffer, C. M., and Patterson, K. M. (1988). A reduced-space approach to the clustering of categorical data in a reduced-space approach to the clustering of categorical data in market segmentation. *Journal of the Market Research Society*, 30:267–288.

[Greenacre, 2007] Greenacre, M. J. (2007). *Correspondence Analysis in Practice, second edition*. Chapman and Hall/CR.

[Halkidi et al., 2002] Halkidi, M., Batistakis, Y., and Vazirgiannis, M. (2002). Cluster validity methods. *ACM Sigmod Record*.

[Hennig, 1999] Hennig, C. (1999). Models and methods for clusterwise linear regression. *Classification in the Information Age*, pages 179–187.

[Hotelling, 1933] Hotelling, H. (1933). Analysis of a complex of statistical variables into components. *Journal of Educational Psychology*, 24:417–441.

[Hwang et al., 2006] Hwang, H., Dillon, W. R., and Takane, Y. (2006). An extension of multiple correspondence analysis for identifying heterogenous subgroups of respondents. *Psychometrika*, 71:161–171.

[Iodice D'Enza and Palumbo, 2010] Iodice D'Enza, A. and Palumbo, F. (2010). Clustering and dimensionality reduction to discover interesting patterns in binary data. In Heidelberg, S. V., editor, *Advances in Data Analysis, Data Handling an Business Intelligence*, Studies in Classification, Data Analysis and Knoledge Organization, pages 45–55. Fink et al.

[Iyigun, 2007] Iyigun, C. (2007). *Probabilistic Distance Clustering*. Ph.D. thesis at, New Brunswick Rutgers, The State University of New Jersey.

[Jain, 2009] Jain, A. K. (2009). Data clustering: 50 years beyond k-means,òÜ. *Pattern Recognition Letters*, pages 1–16.

[Jain et al., 1999] Jain, A. K., Murty, M. N., and Flynn, P. J. (1999). Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323.

[Jancey, 1966] Jancey, R. (1966). Multidimensional group analysis. *Australian Journal of Botany*, 14:127–130.

[Joe and Ward, 1963] Joe, H. and Ward, J. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244.

[Khun, 1973] Khun, H. W. (1973). A note on fermat's problem. In *Mathematical programming*, volume 4, pages 98–107. Spinger.

[Kiers and Kinderen, 2003] Kiers, H. and Kinderen, A. (2003). A fast method for choosing the numbers of components in tucker3 analysis. *British Journal of Mathematical and Statistical Psychology*, 56(1):119–125.

[Kroonenberg, 2008] Kroonenberg, P. (2008). *Applied multiway data analysis*. Ebooks Corporation, Hoboken, New Jersey.

[Le Roux and Rouanet, 2004] Le Roux , B. and Rouanet, H. (2004). *Geometric data analysis*. Kluwer Academic Publishers, Dordrecht.

[Lebart et al., 1984] Lebart, A., Morineau, A., and Warwick, K. (1984). *Multivariate Statistical Descriptive Analysis*.

[Lebart, 1994] Lebart, L. (1994). Complementary use of correspondence analysis and cluster analysis. *In M. J., Greenacre and J. Blasius*

(Eds.), Correspondence Analysis in the Social Sciences. London: Academic Press, pages 162–178.

[Lee and Lee, 2005] Lee, J. and Lee, D. (2005). An improved cluster labeling method for support vector clustering. *IEEE transaction on pattern analysis and machine intelligence.*

[Lee and Danels, 2006] Lee, S.-H. and Danels, K. M. (2006). Cone cluster labeling for support vector clustering. In *Proceeding of the sixth SIAM internationa conference on data mining.* Bethesda.

[Lenca et al., 2008] Lenca, P., M.Patrik, Benoît, V., and Stéphane, L. (2008). On selecting interestingness measures for association rules: User oriented descripition and multiple criteria decison aid. *European journal of Operational Research.*

[Macnaughton-Smith et al., 1964] Macnaughton-Smith, P., Williams, W. T., Dale, M. B., and Mockett, L. G. (1964). Dissimilarity analysis: a new technique of hierarchical sub-division. *letters to nature*, 2:1034–1035.

[MacQueen, 1967] MacQueen, J. (1967). Some methods for classification and analysis some methods for classification and anlysis of multivariate observations. *Proceedings of the fifth Berkeley symposium*, 1:281–297.

[Marino et al., 2009] Marino, M., Palumbo, F., and Tortora, C. (2009). Clustering in feature space for interesting pattern identification of categorical data. In *extendedpaper proceedings of the SIS 2009 statistical conference on Statistical Methods for the analysis of large data-sets, Pescara ISBN 978-3-642-21036-5.*

[Marino and Tortora, 2009] Marino, M. and Tortora, C. (2009). A comparison between k-means and support vector clustering of categorical data. *Statistica applicata*, 21(1):5–16.

[Maronna and Zamar, 2002] Maronna, R. A. and Zamar, R. H. (2002). Robust estimates of location and dispersion for high-dimensional datasets. *Technometrics*, 44(4):307–317.

[McLachlan and Basford, 1988] McLachlan, G. J. and Basford, K. E. (1988). *Mixture models: inference and applications to clustering*, volume 84. Statistics, textbooks and monographs.

[McLachlan and Krishnan, 2008] McLachlan, G. J. and Krishnan, T. (2008). *The EM algorithm and extensions*. Wiley interscience.

[McLachlan and Peel, 2000] McLachlan, G. J. and Peel, D. (2000). Mixtures of factor analyzers. *Langley, Proccedings of the seventeenth International Conference on Machine Learning. Morgan Kaufman, San Francisco*, pages 599–606.

[McLachlan et al., 2003] McLachlan, G. J., Peel, D., and Bean, R. W. (2003). Modelling high-dimensional data by mixtures of factor analyzers. *Computational Statistics and Data Analysis*, 41:379–388.

[Milligan and Cooper, 1985] Milligan, G. and Cooper, M. (1985). An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, 50(2):159–179.

[Montanari and Viroli, 2011] Montanari, A. and Viroli, C. (2011). Maximum likelihood estimation of mixtures of factor analyzers. *Computational Statistics and Data Analysis*, 55:2712–2723.

[Muller et al., 2001] Muller, K. R., Mika, S., Ratsch, G., K.Tsuda, and Scholkopf, B. (2001). An inroduction to kernel-based learning algorithms. *IEEE transaction on neral networks*, 12.

[Ordonez, 2003] Ordonez, C. (2003). Clustering binary data streams with k-means. *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, pages 12–19.

[Palumbo et al., 2008] Palumbo, F., Vistocco, D., and Morineau, A. (2008). *Huge Multidimensional Data Visualization: Back to the Virtue of Principal Coordinates and Dendrograms in the New Computer Age*, volume Handbook of Data Visualization, pages 349–387. Springer.

[Parsons et al., 2004] Parsons, L., Haque, E., and Liu, H. (2004). Subspace clustering for high dimensional data: a review. *ACM SIGKDD Explorations Newsletter*, 6(1):90–105.

[Pelleg and Moore, 2000] Pelleg, D. and Moore, A. (2000). X-means: Extending k-means with e cient estimation of the number of clusters. *ICML-2000*.

[Pham et al., 2004] Pham, D., Dimov, S., and Nguyen, C. (2004). An incremental k-means algorithm. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 218(7):783–795.

[Romesburg, 2004] Romesburg, H. C. (2004). *Cluster analysis for researcher*. LULU Press North Carolina.

[Rousseeuw, 1987] Rousseeuw, P. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65.

[Saporta, 1990] Saporta, G. (1990). *Probabilités Analyse des Données et Statistiques*. Technip, Paris.

[Singh, 1993] Singh, B. (1993). On the analysis of variance method for nominal data. *Sankhyā: The Indian Journal of Statistics, Series B*.

[Steinley, 2003] Steinley, D. (2003). Local optima in k-means clustering: What you don't know may hurt you. *Psychological Methods*, 8:294–302.

[Tortora et al., 2011a] Tortora, C., Gettler Summa, M., and Palumbo, F. (2011a). Factorial pd-clustering. In submitted, editor, *Proceedings of the Symposium of the German Classification Society*.

[Tortora et al., 2011b] Tortora, C., Palumbo, F., and Gettler Summa, M. (2011b). Factorial pd-clustering. http://hal.archives-ouvertes.fr/hal-00591208/fr/.

[Tryon, 1939] Tryon, R. C. (1939). Cluster analysis. *Ann Arbor: Edwards*.

[Van Buuren and Heiser, 1989] Van Buuren, S. and Heiser, W. J. (1989). Clustering n objects into k groups under optimal scaling of variables. *Psychometrika*, 54:699–706.

[Vichi and Kiers, 2001] Vichi, M. and Kiers, H. (2001). Factorial k-means analysis for two way data. *Computational Statistics and Data Analysis*, 37:29–64.

[Vichi and Saporta, 2009] Vichi, M. and Saporta, G. (2009). Clustering and disjoint principal component analysis. *Computational Statistics & Data Analysis*, 53(8):3194–3208.

[Wedel and Kamakura, 1999] Wedel, M. and Kamakura, W. A. (1999). *Market segmentation*. Kluwer Academic Publishers.

[Weiszfeld, 1937] Weiszfeld, E. (1937). Sur le point par lequel la somme des distances de n points donnés est minimum. *Tohoku Mathematics journal*, 43:355–386.

[Yang et al., 2002] Yang, J., Estivill-Castro, V., and Chalup, S. (2002). Support vector clustering through proximity graph modeling. *Proceedings of 9th International Conference on Neural Information Processing*, pages 898–903.

Index