POLITECNICO DI MILANO

Facoltà di Ingegneria dei Sistemi

Corso di Laurea Specialistica in Ingegneria Matematica

Functional Sparse K-Means Clustering

Relatore
Prof. Piercesare Secchi

Laureando
Davide Floriello (matr. 734820)

Anno Accademico 2010-2011

The question, O me! so sad, recurring - What good amid these, O me, O life?

Answer.

That you are here - that life exists, and identity;
That the powerful play goes on, and you will contribute a verse.

O Me! O Life!, Walt Whitman

Chance and chance alone has a message for us. Everything that occurs out of necessity, everything expected, repeated day in and day out, is mute. Only chance can speak to us. We read its message much as gypsies read the images made by coffee grounds at the bottom of a cup.

The Unbearable Lightness of Being, Milan Kundera

Contents

List of Figures

List of Tables

Abstract

When one faces a clustering problem, typically *unsupervised*, it is probable that only a limited number of variables causes the differences between the groups. For this reason new statistical methods, denominated "sparse", are born, which, at the same time, select relevant features and classify the data. The purpose of the following work is the extension, in a functional environment, of a result, recently appeared in [23], which defines a method of this type in case of vectorial K-means. If the data are functions, we propose a method able to select Borel subsets of the domain, where the clusters distinguish the most and able to classify through a functional K-means. This is obtained thanks to the constrained maximization of a functional and the optimization is to be done over the set of possible clusters and over a set of admissible functions, responsible for feature selection. It is proven the existence and uniqueness of the solution to this problem and, under a weak strengthening of the hypotheses, the convergence in L^2 and $[\mu]-$a.e. of the solution function to an object known from the problem. Then it is derived an inequality on the committed error and a numerical algorithm is deducted. Successively, this method is tested firstly on simulated cases and then on real datasets. The first real case is the dataset Growth, on the growth curves of 93 children; the analises are conducted both on aligned and misaligned curves, in order to obtain a better clusterization with respect to standard methods and some aspects already found by evolutionists are observed. Finally, after a further extension of this method to the case of vector of functions, it is used to a study, even *supervised*, of the geometry of the internal carotid of 65 patients.

Introduzione

Lo scopo principale del seguente lavoro è lo sviluppo di un metodo, nel caso di dati funzionali, di clustering sparso, ossia, un procedimento che riesca a classificare i dati e, contemporaneamente, selezioni variabili rilevanti che differenzino i gruppi.
Nel caso di dati di tipo vettoriale, questo problema si colloca nel campo denonimato *unsupervised learning* e un gran numero di tecniche, ormai note, sono state sviluppate e studiate per ottenere tale fine. Esempi possono essere trovati in algoritmi come il *COSA* (Clustering Objects on a Subset of Attributes, cfr. [3]), in alcune particolari versioni di verosimiglianze penalizzate, come quelle proposte da Pan e Shen (2007) o, nel metodo che vogliamo estendere al caso funzionale, il K-Means sparso, cfr. [23].
Il problema della classificazione nasce quando ci si trova a dover analizzare un nuovo dataset, ma nessuna informazione sulla distribuzione di probabilità che genera i dati, è disponibile. I ricercatori sono interessati a stimare la distribuzione che dà origine alle osservazioni e alcuni utili parametri ad essa connessi, come la sua media o la sua varianza. Un gran numero di metodi esiste per trattare questo caso: da stime nonparametriche della densità fino a metodi locali, ognuno dei quali presenta propri vantaggi e svantaggi. Può tuttavia accadere che diversi gruppi di dati siano generati da differenti, anche se incognite, distribuzioni; perciò, prima di condurre qualsiasi analisi, è molto importante riconoscere i possibili clusters di osservazioni, in modo da isolare insiemi di dati generati dalla stessa distribuzione. Accade spesso che non tutte le variabili caratteristiche dei dati contribuiscano in ugual modo alla distinzione dei clusters. L'identificazione di quelle variabili responsabili della diversificazione è un problema di estrema importanza in statistica. Nel caso di dati vettoriali, tale questione è stata già affrontata e recentemente risolta con un metodo basato sul K-Means, da R. Tibshirani e D. Witten, [23]. Essi hanno proposto un K-Means pesato dove un vettore di elementi, maggiori o uguali a zero, riesce a identificare variabili rilevanti per la classificazione, dimostrando, inoltre, che questa è l'unica soluzione di un problema di ottimizzazione ben posto. Essi trovano una forma analitica del vettore

soluzione, che è data dall'operatore Soft-thresholding e strettamente legata al metodo di Lasso.
L'Analisi di Dati Funzionali, o FDA, è una recente e nuova area di ricerca in statistica. In un crescente numero di applicazioni, i dati raccolti sono quantità molto vicine nel tempo o nello spazio ed essi possono naturalmente essere pensati come curve o funzioni. Esempi si possono trovare in medicina, fisica, economia, meteorologia e molte altre scienze. La novità dell'approccio consiste nel considerare ogni curva come un singolo dato; in questo modo tutte le caratteristiche funzionali sono interamente mantenute. Sono già stati messi a punto algoritmi che svolgono clustering funzionale, ma, in questo lavoro, ci concentriamo sullo sviluppo di un metodo per clustering funzionale di tipo sparso. Notevoli sono le motivazioni per una ricerca in questa direzione: selezione di variabili e miglioramenti nella classificazione, o inferenze e criteri di decisione ne rappresentano solo un numero limitato. Non possiamo, tuttavia, pensare di applicare direttamente i risultati trovati in [23]: quelli erano stati trovati solo per dati vettoriali e perciò, quel contesto, è assolutamente inappropriato per dati funzionali. Di conseguenza, si sente l'esigenza di un metodo completamente funzionale. Estenderemo il procedimento proposto in [23], al caso funzionale.
La tesi è strutturata come segue:
Capitolo 1: questo è un capitolo di riepilogo dei risultati noti in spazi finito-dimensionali. Discuteremo il metodo proposto in [23] e il conseguente algoritmo. Infine, concluderemo mostrando un'applicazione di questo procedimento ad un esempio, modificato, e già presentato dagli autori.
Capitolo 2: questa sezione è prevalentemente teorica e introduce i nuovi risultati della tesi. Cominceremo discutendo le proprietà dei dati funzionali e i problemi ad essi collegati. In seguito definiremo il contesto rigoroso e formale del problema, che è proposto come una massimizzazione vincolata di un opportuno funzionale, sull'insieme dei cluster possibili e sull'insieme delle funzioni peso ammissibili. Nel Teorema 2.1, dimostreremo l'esistenza e l'unicità della soluzione del problema. Inoltre mostreremo che, se rafforziamo leggermente le ipotesi sulle proprietà topologiche dei dati, riusciamo ad ottenere la convergenza in L^2 e $[\mu]$ −a.e. della funzione peso ottima $w^*(x)$ ad una funzione d'interesse, data dal problema. Deriveremo un algoritmo che svolga quanto asserito dal Teorema 2.1. Alla fine del capitolo, proporremo un facile esempio di applicazione del metodo appena sviluppato, in cui riusciamo ad effettuare tutti i calcoli in modo analitico.
Capitolo 3: questo capitolo è completamente orientato a mettere alla prova, con simulazioni, il nuovo algoritmo. Gli esempi proposti sono di complessità crescente e sono utilizzati per osservare l'aderenza con i risultati teorici dimostrati. Per ogni caso i risultati sono confrontati con l'algoritmo classico del

K-Means e viene osservata l'eventuale convergenza delle soluzoni. Il nuovo algoritmo migliora costantemente le classificazioni, rispetto al K-Means classico.
Capitolo 4: viene qui illustrata la prima applicazione del nuovo metodo, a dei dati reali. É lo studio delle curve di crescita di 93 bambini. Questo famoso dataset è reso disponibile con il pacchetto `fda` di `R`. Innanzitutto, analizzeremo le curve non registrate. Noteremo che il novo algoritmo riconosce perfettamente i bambini e le bambine dalle loro altezze, ma non riesce a fare altrettanto con le curve della velocità e dell'accelerazione della crescita. Successivamente, dopo una breve presentazione delle principali tecniche di registrazione, passeremo ad analizzare le curve registrate. Come mostrato in un precedente lavoro, cfr. [19], possiamo individuare solo un cluster, perciò ci concentreremo su un'analisi supervisionata per stimare la differenza tra gli orologi biologici tra maschi e femmine.
Capitolo 5: qui consideriamo una seconda applicazione ad un caso reale; si tratta dello studio della geometria della carotide interna destra di 65 pazienti, alcuni dei quali sono affetti da aneurisma. Lo scopo è la classificazione di diverse possibili forme delle carotidi. Prima di ciò, tuttavia, dobbiamo estendere l'algoritmo al caso di vettori di funzioni. Dimostreremo, pertanto, un Teorema analogo al Teorema 2.1 e descriveremo un algoritmo che esegue un clustering sparso funzionale congiuntamente su tutti gli elementi dei vettori di funzioni dati. In seguito cercheremo classificazioni ragionevoli per la geometria delle carotidi e, infine, useremo il nuovo metodo per un'analisi supervisionata, volta a trovare gruppi di variabile che distinguono gruppi interessanti di pazienti.

Introduction

The primary aim of this work is the development of a functional sparse clustering method for functional data, i.e. a clustering method that selects only relevant features while it clusters the data.

In case of vector data, this particular problem lies within the research area of *unsupervised learning* and a large number of well known techniques have been already developed and studied in order to achieve that purpose. Examples include methods like *COSA* (Clustering Objects on a Subset of Attributes, cfr. [3]), some particular versions of a penalized log-likelihood, like those presented by Pan and Shen (2007) or the method which we here want to extend to functional data, namely Sparse K-means, cfr. [23].

The problem of clustering arises when a new dataset is to be analyzed, but no information about the distribution generating it is available. Researchers are interested in estimating the distribution generating the data and some useful parameters related to it, such as its mean or variance. A large number of methods exist to treat this case: from nonparametric estimation of the densities to local methods, each of which has advantages and drawbacks. It might however happen that different groups of data are generated by different, albeit unknown, distributions; thus, before conducting any analysis, it is very important to recognize possible clusters of the observations, aiming at isolating groups of data generated by the same distribution. It is often the case that not all the features describing the observations equally contribute to distinguish the clusters. Identifying those features responsible for the discrimination is a problem of high interest in statistics. In the case of vectorial data, the problem has been already attacked and recently solved, with a K-means based method, by R. Tibshirani and D. Witten, [23]. They have proposed a weighted K-means, where a vector of nonnegative weights are able to identify important discriminating features proving, moreover, that this is the unique solution to a well posed optimization problem. They find an analytical form for the resulting vector of weights, that is given by the Soft-Thresholding operator and strictly related to the Lasso method.

Functional Data Analysis is a recent new area of research in Statistics. In an ever increasing number of applications, the data collected are quantities very close in time or space, which can be naturally thought of as curves or functions. Examples are found in medicine, physics, economics, meteorology and many others sciences. The innovation consists in considering each curve as a single datum; in this way the functional features are all maintained.

Methods performing clustering with functional data already exist, but in this thesis we are concerned with the development of a sparse functional clustering method. There are many motivations for an interest in this direction: features selection, more interpretable results and improvements in the classification, or inference and decision making processes are only a limited number of them. However, we can not think of directly applying the results found in [23]: they were developed only for vectors and so that framework is inappropriate to functional data. Therefore, the need for a completely functional method arose. We will extend the method proposed in [23], to a functional setting.

The thesis is organized as follows:

Chapter 1: this is a review chapter about known results in finite dimension spaces. We will discuss the method proposed in [23] and go through the explanation of their algorithm. Finally we will end showing an application of this method to a modified example already discussed by the authors.

Chapter 2: this section is mostly theoretic and introduces the new results advanced in the thesis. We will begin by discussing the properties of functional data and the problems related to them. Then we will move on to define the rigorous setting of the problem, which is proposed as the constrained maximization of a functional over the set of possible clusters and the set of admissible weight functions. In Theorem 2.1, we will prove the existence and uniqueness of the solution to this problem. Moreover we will show that if we slightly strengthen the hypotheses on the topological properties of the data, we are able to obtain the convergence in L^2 and $[\mu]-$a.e. of the optimal weight function $w^*(x)$ to a contingent function of interest. We will derive a numerical algorithm that performs what Theorem 2.1 claims. In the end of the chapter, we will go through an easy example of application of the method, where we can develop all the computations in a completely analytical way.

Chapter 3: this chapter is completely devoted to test the new algorithm to some simulated cases. The examples examined are of increasing complexity and are used to see the adherence to the theorical results. For each case we make a comparison with classical K-means and eventually observe the convergence of the solutions. Our algorithm is constantly found to improve on the classifications.

Chapter 4: The first real data analysis considered in the thesis is here illus-

trated. It is the study of the growth curves of 93 children. This quite famous dataset is made available with the `R` package `fda`. Firstly we will analyze the misaligned curves. We will note that our algorithm perfectly recognizes the boys and the girls form the height curves, but fails in identifying the clusters from the velocities and the accelerations. Then, after a brief presentation of the alignment techniques, we will move to the analysis of the aligned curves. As showed in a previous work, cfr. [19], we can identify only one cluster, thus we will concentrate on a supervised analysis in order to estimate the shift in the biological clocks between boys and girls.
Chapter 5: We here consider a second application to real data; it is the study of the geometry of the right internal carotid of 65 patients, some of whom are affected by aneurysms. The purpose is the classification of the possible different shapes of the carotids. However, before going through this, we have to extend the algorithm to the case of vector of functions. We will prove a Theorem analogous to Theorem 2.1 and derive an algorithm that performs joint sparse functional clustering. Then we will search for some reasonable classifications of the geometry of the carotids and, finally, we will use our method to perform a supervised analysis, in order to find the features that mostly distinguish some interesting groups of patients.

Chapter 1

Vectorial Sparse Clustering

1.1 Introduction

When we face a clustering problem, in general, we can not expect that the different groups generating the data, distinguish themselves along the whole set of features. It is far more probable that only a restricted number of variables cause the data to be different. Some procedures, such as the *COSA method*, e.g. [3], some particular versions of a penalized log-likelihood or weighted clustering methods, like those presented in [10], were designed to take into account this fact, in order to obtain more precise clusters. We first begin this chapter by developing a particular method, among those just cited, in the finite dimensional case. We will derive the formal problem and find the solution, which is explicit, in this context, and given by the Soft-thresholding operator. Moreover we can obtain a simple algorithm, that could be implemented in specific softwares, so to find the solution when the data and the number of features are so many that we can not solve analitically the problem. The use of this particular algorithm leads to the problem of how to choose the right values of a parameter that is a specifical constraint on the ℓ_1 norm of the weight vector. The solution is found thanks to a particular use of the Gap Statistics through a permutation procedure.

1.2 Sparse Clustering in Finite Dimension

Let us first take a brief overview of some possible ways proposed for sparse clustering. These techniques are usually adopted when data are in high dimensional feature spaces.
Let X denote an $n \times p$ matrix, where n is the number of observations and p is

the number of features and each $x_{ij} \in \mathbf{R}$. One way to reduce the dimensionality of the data before clustering is by performing a matrix decomposition. The data matrix X can be approximated by $X \approx AB$, where A is a $n \times q$ matrix and B a $q \times p$ matrix, with $q \ll p$. Then, one can cluster the observation using A, rather than X. Several authors, for instance Ghosh and Chinnaiyan (2002) and Liu et al. (2003), proposed this way to solve the problem and the A matrix of reduced dimensionality they used was obtained by performing a Principal Component Analysis; then the rows of A can be clustered. Similarly, someone else, see Tamayo et al. (2007), suggetsed a decomposition of X using the nonnegative matrix factorization and then, again, cluster the rows of A. These proposals, however, present a number of drawbacks. First of all, the resulting clustering is not sparse in the features, since each of the columns of A is, in general, a function of the whole set of features. Moreover, there is no guarantee that we are able to separate the true groups, because the principal components with largest eigenvalues do not necessarily provide the best separation between subgroups.
Sparse clustering, instead, has two great advantages:

I) if it is true that only a small number of variables separate the clusters, then it might result a more accurate identification of the groups if compared with standard clustering;

II) it helps the interpretation of the procedure that leads to the formation of different groups and so gives a simpler way to recognize them in future applications.

Clustering procedures require the concept of dissimilarity measures between pairs of observations.

Definition 1.1. *A function $d : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}^+$ is a dissimilarity measure if it satisfies the two following properties:*

1) $d(\boldsymbol{x}, \boldsymbol{y}) = d(\boldsymbol{y}, \boldsymbol{x}), \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^p$;

2) $d(\boldsymbol{x}, \boldsymbol{x}) = 0, \forall \boldsymbol{x} \in \mathbb{R}^p$.

From this definition we see that any dissimilarity measure looks like a distance. Indeed every distance is a particular case of a dissimilarity measure: a distance is a dissimilarity that satisfies, also, the triangular inequality. We assume, moreover, that, the particular chosen dissimilarity measure is additive in the feature. For instance, take d to be the squared Euclidean distance, so to have:

$$d(\mathbf{x}, \mathbf{y}) = \sum_{j=1}^{p} d_{i,i',j},$$

where
$$d_{i,i',j} = \left(\mathbf{x}_{ij} - \mathbf{x}_{i'j}\right)^2 .$$
The model based clustering framework has been studied extensively in recent years and many of the proposals for feature selection and dimensionality reduction for clustering fall in this setting; for example the articles of McLachlan and Peel (2000) and Fraley and Raftery (2002) are considerable in this field. The basic idea is as follows: one can model the rows of X as indipendent multivariate observations drawn from a mixture model with K components; usually a mixture of Gaussians is used. That is, given the data, the log-likelihood is
$$\sum_{i=1}^{n} \log \left[\sum_{k=1}^{K} \pi_k f_k(\mathbf{x}_i; \mu_k, \Sigma_k) \right], \tag{1.1}$$
where f_k is a Gaussian density parametrized by its mean μ_k and covariance matrix Σ_k, π_k are the *a priori* probabilities and $\mathbf{x}_j \in \mathbb{R}^p$ is the vector of the jth row of X. The EM algorithm can be used to fit this model. When $p \approx n$ or $p \gg n$, however, problems arise, because the $p \times p$ covariance matrix Σ_k can not be estimated from only n observations. If we suppose that that the observations lie in a low-dimensional latent factor space, we can overcome the problem, but we end only with a dimensionality reduction and not with sparsity. It turns out that model based clusterings lends itself easily to feature selection. Rather than seeking μ_k and Σ_k that maximizes the previous log-likelihood, one can, instead, maximize the log-likelihood subject to a penalty that is chosen to yield sparsity in the features. For example, if we assume that the features of X are centered to have mean zero, then, the proposed penalized log-likelihood is
$$\sum_{i=1}^{n} \log \left[\sum_{k=1}^{K} \pi_k f_k(\mathbf{x}_i; \mu_k, \Sigma_k) \right] - \lambda \sum_{k=1}^{K} \sum_{j=1}^{p} |\mu_{kj}|, \tag{1.2}$$
where $\Sigma_1 = \ldots = \Sigma_K$ is taken to be a diagonal matrix. That is, a lasso penalty is applied to the elements of μ_k. When the nonnegative tuning parameter λ is large, then some of the elements of μ_k will be exactly equal to zero. If, for some variable j, $\mu_{kj} = 0$, $\forall k = 1, \ldots, K$, then the resulting clustering will not involve feature j, Hence this yields a clustering that is sparse in the features.
Friedman and Meulman (2004) proposed *Clustering Objects on Subsets of Attributes (COSA)*. Let C_k denote the indices of the observations in the kth of K clusters. Then, the *COSA* criterion is
$$\min_{(C_1,\ldots,C_K),\mathbf{w}} \sum_{k=1}^{K} a_k \sum_{i,i' \in C_k} \sum_{j=1}^{p} (w_j d_{i,i',j} + \lambda w_j \log w_j)$$

$$\text{subject to} \qquad \sum_{j=1}^{p} w_j = 1, \qquad w_j \geq 0, \ \forall j. \tag{1.3}$$

Actually, the *COSA* criterion allows different feature weights within each cluster. Here a_k is some function of the number of elements in cluster k, $\mathbf{w} \in \mathbb{R}^p$ is a vector of feature weights and $\lambda \geq 0$ is a tuning parameter. It can be observed that this criterion is related to a weighted version of K-means clustering. Unfortunately, this proposal, does not truly result in a sparse clustering, since all variables have nonzero weights for $\lambda > 0$. An extension of (1.3) is proposed, by the same authors in [3], in order to generalize the method to other types of clustering, such as hierarchical clustering. The proposed algorithm is quite complex and involves multiple tuning parameters, that should be avoided, as they require to be determined in an appropriate way.

1.3 The Proposed Sparse Clustering Framework

The method we want to extend is the one proposed by D. Witten and R. Tibshirani in [23], so we first analyze it more deeply.
We want to cluster the observations and we suspect that the true underlying groups differ only with respect to a restricted group of the variables. The method proposed allows to choose, adaptively, a subset of features to cluster the observations. Let $\mathbf{x}_j \in \mathbb{R}^n$ denote the vector of the jth feature. Many clustering methods can be restated as an optimization problem of the form:

$$\max_{\Theta \in D} \sum_{j=1}^{p} f_j(\mathbf{x}_j, \Theta), \tag{1.4}$$

where $f_j(\mathbf{x}_j, \Theta)$ is some appropriate function that involves only the$j$th feature of the data and Θ is a parameter restricted to lie in a set D. Two examples of such methods are K-means and hierarchical clustering. In K-means f_j turns out to be the between cluster sum of squares for feature j and Θ is a partition of the observation into K disjoint groups. The K-means method is precisely what we are going to implement. In [23], the following definition is given.

Definition 1.2. *We define sparse clustering the solution to the following problem:*

$$\max_{\boldsymbol{w};\Theta \in D} \sum_{j=1}^{p} w_j f_j(\boldsymbol{x}_j, \Theta)$$

$$\textit{subject to } \|\boldsymbol{w}\|_{\ell^2}^2 \leq 1, \ \|\boldsymbol{w}\|_{\ell^1} \leq s, \ w_j \geq 0 \ \forall j, \tag{1.5}$$

where w_j is a weight corresponding to the jth feature and s is a tuning parameter, with $1 \leq s \leq \sqrt{p}$.

Let us make a few comments on this problem.
If we take $w_1 = w_2 = \ldots = w_p$ in (1.5), then we obtain a simple, non-weighted clustering criterion. The constraint on the ℓ^1 norm of the vector $\mathbf{w}$ is truly what forces the sparsity on the components of $\mathbf{w}$ for small values of the tuning parameter s, meaning that s is a kind of measure of the number of features involved in the clustering. The ℓ^2 penalty is also important, since, without it, at most one element of $\mathbf{w}$ would be nonzero in general. The presence of those weights w_j helps interpretating the results: the value of each w_j can be thought of as the contribution of the jth feature to the final sparse clustering; therefore a large value of some w_j means that those features contribute greatly to the diversification, whereas $w_j = 0$ simply states that the jth feature is not involved in the clustering. Finally, we can observe that, for (1.5) to result in a nontrivial sparse clustering, it is necessary that $f_j(\mathbf{x}_j, \Theta) > 0$ for some or all j. Indeed, if we have $f_j(\mathbf{x}_j, \Theta) \leq 0, \forall j$, then the solution to the problem is simply $w_j = 0, \forall j$. If $f_j(\mathbf{x}_j, \Theta) > 0$, then the nonnegativity constraint on that feature has no effect.
The solution is found using an iterative algorithm: holding $\mathbf{w}$ fixed, (1.5) is optimized with respect to Θ and, holding Θ fixed, it is optimized with respect to $\mathbf{w}$. This procedure assures us a monotonicity on the value of the objective function. Note, finally, that we can rewrite the previous problem as a linear problem:

$$\max_{\mathbf{w}} \mathbf{w}^T \mathbf{a}$$

$$\text{subject to } \|\mathbf{w}\|_{\ell^2}^2 \leq 1, \ \|\mathbf{w}\|_{\ell^1} \leq s, \ w_j \geq 0 \ \forall j, \tag{1.6}$$

where $a_j = f_j(\mathbf{x}_j, \Theta)$. In this case, we have an explicit solution, found, by the two authors in [23].

Proposition 1.1.
The solution to the convex problem (1.6) is

$$\boldsymbol{w} = \frac{S(\boldsymbol{a}_+, \Delta)}{\|S(\boldsymbol{a}_+, \Delta)\|_{\ell^2}},$$

where x_+ denotes the positive part of x and where $\Delta = 0$ if that results in $\|\boldsymbol{w}\|_{\ell^1} \leq s$; otherwise, $\Delta > 0$ is chosen to yield $\|\boldsymbol{w}\|_{\ell^1} = s$. Here S is the Soft-thresholding operator, defined as $S(x, c) = sign(x)(|x| - c)_+$ and $1 \leq s \leq \sqrt{p}$.

Behind this result, there is the fundamental and technical assumption that there is a unique maximal element of $\mathbf{a}$. The proof of the propostion

follows from the Karush-Kuhn-Tucker's Theorem.
Now we are going to show that K-means clustering optimizes criteria of the form (1.4). Then we propose a sparse version of K-means clustering using (1.5). The resulting criterion is easily optimized and involve a single tuning parameter s that controls the number of features used in the clustering.

1.3.1 The Sparse K-Means Method

K-means clustering minimizes the *Within Cluster Sum of Squares (WCSS)*. That is, it seeks to partition the n observation into K sets, or clusters, such that the WCSS

$$\sum_{k=1}^{K} \frac{1}{2n_k} \sum_{i,i' \in C_k} \sum_{j=1}^{p} d_{i,i',j} \tag{1.7}$$

is minimal, where n_k is the number of observations in cluster k and C_k contains the indices of the observations in cluster k. In general $d_{i,i',j}$ can denote any dissimilarity measure between observations i and i' along feature j. In order to have a K-means procedure, it is required to take $d_{i,i',j} = (\mathbf{x}_{ij} - \mathbf{x}_{i'j})^2$; for this reason we refer to (1.7) as the within cluster sum of squares. Note that if we define the *Between Clusters Sum of Squares (BCSS)* as

$$\sum_{j=1}^{p} \left(\frac{1}{2n} \sum_{i=1}^{n} \sum_{i'=1}^{n} d_{i,i',j} - \sum_{k=1}^{K} \frac{1}{2n_k} \sum_{i,i' \in C_k} d_{i,i',j} \right), \tag{1.8}$$

then minimizing the WCSS is equivalent to maximizing the BCSS.
We could try to develop a method for sparse K-means clustering by optimizing a weighted WCSS, subject to constraints on the weights, that is:

$$\max_{(C_1,\ldots,C_K),\mathbf{w}} \sum_{j=1}^{p} w_j \left(\sum_{k=1}^{K} -\frac{1}{2n_k} \sum_{i,i' \in C_k} d_{i,i',j} \right)$$
$$\text{subject to} \qquad \|\mathbf{w}\|_{\ell^2}^2 \leq 1,\ \|\mathbf{w}\|_{\ell^1} \leq s,\ w_j \geq 0\ \forall j. \tag{1.9}$$

Here, s, is a tuning parameter. Since, however, each element of the weighted sum is negative, the maximum occurs when all are weights are set to zero, regardless of the value of s. Thus, we have to maximize a weighted BCSS, subject to constraints on the weights. Therefore, the sparse K-means clustering criterion is:

$$\max_{(C_1,\ldots,C_K),\mathbf{w}} \sum_{j=1}^{p} w_j \left(\frac{1}{2n} \sum_{i=1}^{n} \sum_{i'=1}^{n} d_{i,i',j} - \sum_{k=1}^{K} \frac{1}{2n_k} \sum_{i,i' \in C_k} d_{i,i',j} \right)$$
$$\text{subject to} \qquad \|\mathbf{w}\|_{\ell^2}^2 \leq 1,\ \|\mathbf{w}\|_{\ell^1} \leq s,\ w_j \geq 0\ \forall j. \tag{1.10}$$

The weights will be sparse for an appropriate choice of the tuning parameter s, which has to satisfy $1 \leq s \leq \sqrt{p}$. Note that if $w_1 = \ldots = w_p$, then (1.10) simply reduces to the standard K-means criterion. Moreover, we observe that (1.8) and (1.10) are special cases of (1.4) and (1.5), where $\Theta = (C_1, \ldots, C_K)$, $f_j(\mathbf{x}_j, \Theta) = \frac{1}{2n} \sum_{i=1}^{n} \sum_{i'=1}^{n} d_{i,i',j} - \sum_{k=1}^{K} \frac{1}{2n_k} \sum_{i,i' \in C_k} d_{i,i',j}$, and D denotes the set of all possible partitions of the observations into K clusters.
The criterion (1.10) assigns a weight to each feature, based on the increase in BCSS that the feature can contribute. First, consider the criterion with the weights $w_1, \ldots, w_p$ fixed. It reduces to a clustering problem, using a weighted dissimilarity measure. Second, consider the criterion with the clusters $C_1, \ldots, C_K$ fixed. Then a weight will be assigned to each feature based on the BCSS of that feature; features with larger BCSS will be given larger weights. This argument suggests how to develop an iterative algorithm for maximizing (1.10).

Algorithm for sparse K-means clustering

1. Initialize $\mathbf{w}$ as $w_1 = \ldots = w_p = \frac{1}{\sqrt{p}}$.
2. Iterate until convergence:
   (a) Holding $\mathbf{w}$ fixed, optimize (1.10) with respect to $C_1, \ldots, C_K$. That is,
   $$\min_{C_1,\ldots,C_K} \sum_{k=1}^{K} \frac{1}{2n_k} \sum_{i,i' \in C_k} \sum_{j=1}^{p} w_j d_{i,i',j} \tag{1.11}$$
   by applying the standard K-means algorithm to the $n \times n$ dissimilarity matrix with (i, i') element $\sum_j w_j d_{i,i',j}$.
   (b) Holding $C_1, \ldots, C_K$ fixed, optimize (1.10) with respect to $\mathbf{w}$ by applying the Proposition (1.1): $\mathbf{w} = \frac{S(\mathbf{a}_+, \Delta)}{\|S(\mathbf{a}_+, \Delta)\|_{\ell^2}}$, where
   $$a_j = \left(\frac{1}{2n} \sum_{i=1}^{n} \sum_{i'=1}^{n} d_{i,i',j} - \sum_{k=1}^{K} \frac{1}{2n_k} \sum_{i,i' \in C_k} d_{i,i',j} \right) \tag{1.12}$$
   and $\Delta = 0$ if that results in $\|\mathbf{w}\|_{\ell^1} < s$; otherwise, $\Delta > 0$ is chosen so that $\|\mathbf{w}\|_{\ell^1} = s$.
3. The clusters are given by $C_1, \ldots, C_K$ and the feature weights corresponding to this clustering are given by $w_1, \ldots, w_p$.

When d is squared Euclidean distance, Step 2(a) can be optimized by performing K-means on the data after scaling each feature j by $\sqrt{w_j}$. In this implementation of sparse K-means, Step 2 is iterated until the stopping criterion

$$\frac{\sum_{j=1}^{p} |w_j^r - w_j^{r-1}|}{\sum_{j=1}^{p} |w_j^{r-1}|} < 10^{-4} \tag{1.13}$$

is satisfied, where $\mathbf{w}^r$ indicates the set of weights obtained at iteration r. However, the Tibshirani and Witten, in [23], have noticed that, in general, the algorithm will not converge to the global optimum of the criterion (1.10), since the criterion is nonconvex and uses in Step 2(a) the algorithm for K-means clustering, which is not guaranteed to find a global optimum.
Note the similarity between the COSA criterion (1.3) and (1.10): when $a_k = \frac{1}{2n_k}$ in (1.3), then both criteria involve minimizing a weighted function of the WCSS, where the feature weights reflect the importance of each feature in the clustering. However, (1.3) does not result in weights that are exactly equal to zero unless $\lambda = 0$, in which case only one weight is nonzero. The combination of ℓ_1 and ℓ_2 constraints in (1.10) yields the desired effect.

1.3.2 Selection of Tuning Parameter for Sparse K-Means

We now discuss briefly a method that can be used to choose an adequate value for the tuning parameter. We have seen that the sparse K-Means clustering algorithm depends on one tuning parameter s, which is the ℓ_1 bound on $\mathbf{w}$ in (1.10). We assume that the number of clusters K is fixed. The problem of finding the appropriate number of clusters is a quite complex one and requires the use of the "Gap Statistics".
First of all, we note that we can not simply select s to maximize the objective function in (1.10), since, as s is increased, the objective function is increased as well. Moreover we have previously observed the role the tuning parameter assumes: it controls the sparsity in clustering, so, the more we increase s, the less the number of variables will result equal to zero. Thus we have to apply a different method: a permutation approach that is closely related to the Gap Statistics.

Algorithm to select tuning parameter s for sparse K-Means

1. Obtain permuted datasets $\mathbf{X}_1, \ldots, \mathbf{X}_B$ by indipendently permuting the observation within each feature.

2. For each candidate tuning parameter value s:

   (a) Compute $O(s) = \sum_j w_j \left(\frac{1}{2n} \sum_{i=1}^{n} \sum_{i'=1}^{n} d_{i,i',j} - \sum_{k=1}^{K} \frac{1}{2n_k} \sum_{i,i' \in C_k} d_{i,i',j} \right)$, the objective obtained by performing sparse K-Means with tuning parameter value s on the data $\mathbf{X}$.

   (b) For $b = 1, 2, \dots, B$, compute $O_b(s)$ the objective obtained by performing sparse K-means with tuning parameter value s on the data $\mathbf{X}_B$.

   (c) Calculate $Gap(s) = \log(O(s)) - \frac{1}{B}\sum_{b=1}^{B}\log(O_b(s))$.

3. Choose s^* corresponding to the largest value of $Gap(s)$. Alternatively, one can choose s^* equal to the smallest value for which $Gap(s^*)$ is within one standard deviation of $\log(O_b(s))$ of the largest value of $Gap(s)$.

Note that even if there could be strong correlations between the features in the original data $\mathbf{X}$, the features in the permuted datasets $\mathbf{X}_1, \dots, \mathbf{X}_B$ are uncorrelated with each other. The Gap Statistics measures the strength of the clustering obtained on the real data relative to the clustering obtained on null data that does not contain subgroups. The optimal tuning parameter value occurs when this quantity is greatest.

Observation 1.1.
An important question is how to choose the permuted datasets at Step 1. of the previous algorithm. The permuted datasets are obtained in this way: a particular column, say $\bar{j}$, is selected; then the elements $x_{i\bar{j}}$, with $i = 1, \dots, n$, are permuted. Then other columns are chosen and the procedure is repeated, independently from the previous permutations. Thus, it is as we are generating new observations, but with the values of the original dataset. The new datasets $\boldsymbol{X}_1, \dots, \boldsymbol{X}_B$, are randomly chosen among the $(n \times p)!$ possible permutations of the original set of data.

1.4 An Example

Now, we present an example showing sparse clustering in the vectorial case. This case is taken from the booklet explaining the usage of the package `sparcl` in R, but we have made it more complicated. We have two groups, both generated from a Multivariate Normal random variable, with p, the number of variable, equal to 300. The cardinality of the groups is the same: 25 units each. Therefore we have a matrix X, belonging to $\mathbb{R}^{50\times 300}$. To make the groups different between each other, we simply add a random quantity to the observations of the first group, and only to the first 15 variables. Then we rescale the matrix obtained and permute it. The random quantity added is given by $1 + y_i$, where the y_i's are samples from a normal random variable Y_i, with $i = 1, \dots, 15$.
In this case, we know that the tuning parameter s has to satisfy: $1 \leq s \leq \sqrt{300}$. We know exactly what the real parameter s should be; anyway we run

the method implemented with the gap statistics to find its plausible value. The optimal tuning parameter found is 3.857143, very close to the right value, with a gap statistic equal to 0.7489, leading to a number of nonzero weights of 22. In figure 1.1, we have shown the computed Gap Statistics as a function

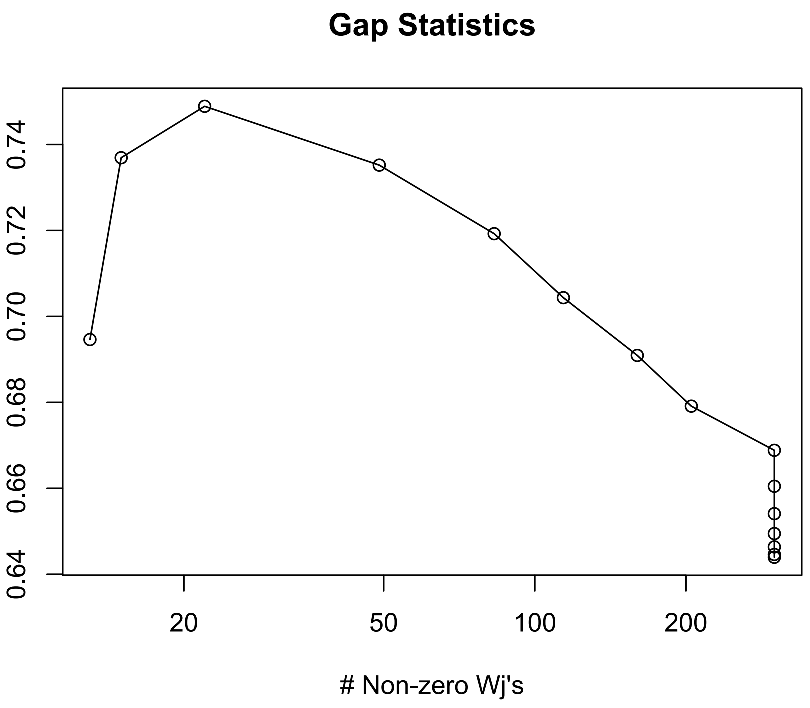

Figure 1.1: Plot of the Gap Statisitc as a function of the number of nonzero w_j's.

of the number of nonzero weights. Note that it overestimate the number of significative variables that differ between the groups. This is a drawback constantly found in every example we have run. However this overestimation is not so large: only seven more than variables, than the right ones, are considered to help distinguishing the groups. Unfortunately, the entity of the overestimation gets larger and larger as the number of variables increases. The reasons for this are not very clear. We propose three motivations that could be responsible for that:

- problems internal to the algorithm used;
- sometimes the way used to weight the variables is not properly correct;
- the fact that, if we augment the number of variables, the probability that the significative features increase is raising.

After these considerations, we apply the Sparse algorithm and we find the number of nonzero weight to be 22, $\|\mathbf{w}\|_{\ell^1} = 3.857141$ and the vector of clusters is perfectly consistent with the permutation, i.e. the groups are perfectly recognized. In figure 1.2, we have reported the plot of the computed

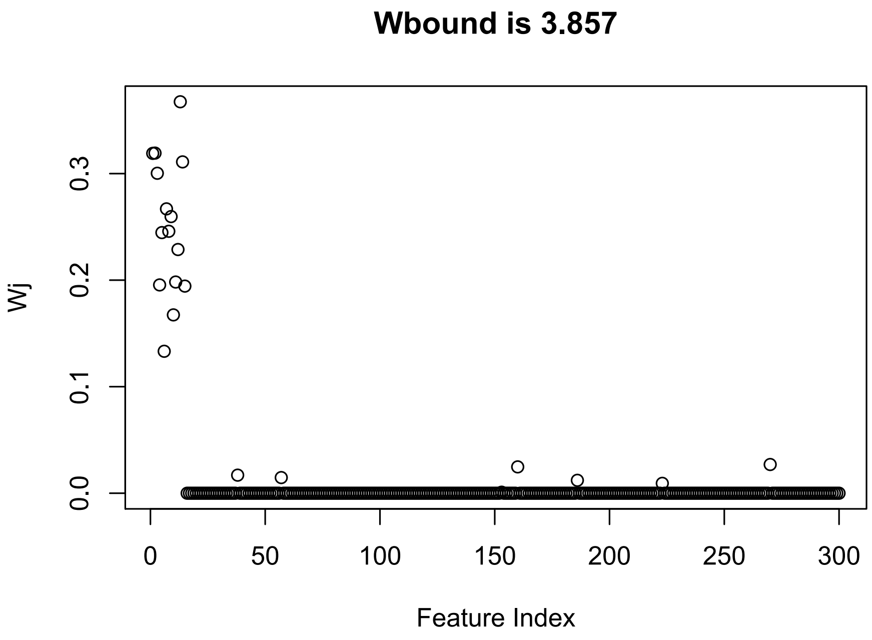

Figure 1.2: Plot of the calculated values of w_j's.

values of the vector $\mathbf{w}$. The components equal to zero are those irrelevant to the clusterization. The maximal component is found to be the thirteenth. If we repeat the calculations with the real parameter s, we find exaclty the same results.
Finally, we have applied the normal K-means algorithm to the same dataset. It misclassifies three data, but, it still performs overall well.

Chapter 2

Functional Sparse Clustering

2.1 Introduction

In this chapter, we will introduce the field of Functional Data Analysis, arisen only few years ago. We will see a definition of functional data and functional dataset. Successively, we will give a short overview on the properties of functional variables, some statistically interesting quantities and their sample counterparts. We will also discuss some problems related to infinite dimensional spaces where functional data lie and we have to work with. Besides problems of mathematical nature, we occur in another great difficulty given by the necessary discretization of functions operated by all calculators. Then we will pass to extend the result found in finite dimensions to this functional case. We will see what changes from the vectorial case and what remains unaltered, the kind of solution we are able to find, the procedure that brings us to it and its properties. Then, we will show, under regular hypotheses of functional data, the convergence of the solution to a particular function of interest. This suggests us how we can write a code for an algorithm able to find the desired solution and, finally, we will try to investigate the error, eventually committed, by this new algorithm. The last section is about an easy example with which we can do, analytically, the all computations needed and verify the properties stated in the previous parts of this chapter.

2.2 Functional Data

There is actually an increasing number of situations coming from different fields of applied sciences, such as biometrics, medicine or econometrics, in which the collected data are curves. In particular, for a single phenomenon, it can be observed a very large set of variables. For instance, we can consider this usual situation where some random variable can be observed at several different times in the range (t_{min}, t_{max}). An observation can, then, be expressed by the random family $\{X(t_j)\}_{j=1,\dots,J}$, in other terms, as if we were sampling from a stochastic process. In modern statistics, we make the grid becoming finer and finer, meaning that consecutive instants are closer and closer. One way to take this into account is to consider the data as observations of the continuous family $\mathcal{X} = \{X(t) : t \in (t_{min}, t_{max})\}$. To fix ideas, we give the following, rather general, definition:

Definition 2.1. *A random variable $\mathcal{X}$ is called functional variable if it takes values in an infinite dimensional, or functional, space $(E, \mathcal{E})$. An observation χ of $\mathcal{X}$ is called a functional data. Here, $\mathcal{E}$ is an appropriate σ-algebra on E.*

This definition, as given in [2], is voluntarily unprecise, to take into account the fact that, with the term functional variable, we are, really, considering a very large variety of objects: it can be a simple real function, or a surface or, in general, a n-dimensional (with $n \geq 2,\ n \in \mathbb{N}$) vector of functions. When $\mathcal{X}$, respectively χ, denotes a random curve, respectively its observation, we implicitly make the following identification $\mathcal{X} = \{\mathcal{X}(t) : t \in T\}$, respectively $\chi = \{\chi(t) : t \in T\}$. In this situation, the functional feature comes directly from the observations. The case when the variable is a curve is associated with an unidimensional set $T \subset \mathbb{R}$. If a functional variable is a random surface, like for instance the grey levels of an image or a vector of curves, then, in this case, T is a bidimensional set $T \subset \mathbb{R}^2$ and therefore we are sampling from what is called a random field or, moreover, we can consider any other more complicated infinite dimensional mathematical object.
Having specified what a functional variable is, we now need a precise definition of a functional dataset.

Definition 2.2. *A functional dataset $\chi_1, \dots, \chi_n$ is the observation of n functional variables $\mathcal{X}_1, \dots, \mathcal{X}_n$ identically distributed.*

This definition covers many situations, the most popular being curves datasets. We will not investigate the question of how these functional data have been collected, which is linked with the discretization problems. According to the kind of the data, a preliminary stage consists in presenting them in a way which is well adapted to functional processing. The problem if the grid of the measurements is fine enough, is a first important stage and usually involves some numerical approximation techniques. In other standard cases, classical smoothing methods can be invoked. There exist some other situations which need more sophisticated smoothing techniques, for instance when the repeated measures per subjects are very few (sparse data) and/or with irregular grid. This is obviously a parallel and complementary field of research but, from now on, we will assume that we have at hand a sample of functional data.

2.2.1 Some properties of Functional Data

One of the interesting facts in considering functional data is that we have a complete representation of a phenomenon. We can take note of *peaks* or *valleys* or even where the function crosses a determined level, which, usually, corresponds to a significant variable to be observed. Each of these functional features are associated with a specific value of the argument of the data, i.e. most features are characterized by a *location*. Moreover, other informations, specifically *amplitude* and *width* are associated to peaks and valleys. This means that in errorless circumstances we could ideally use three measurements to a local reconstruction of our functions and this comes from the idea that in minima or in maxima a function looks like a parabola, which is defined by three coefficients. We call *dimensionality* the amount of information needed to estimate a functional feature.
This suggests the notation of the *resolving power* or *resolution* of a set of data. It is inversely related to the width of the narrowest event that can be estimated to our satisfaction. As an example, we mean by "high resolution data", that they can tell small events. The resolution leads, in turn, to the concept of the *dimensionality* of a function. We can roughly consider it as simply the sum, across all functional features, of the number of informations required to define each of the features. Functions are object potentially infinite dmensional. If we have a funcion that has an infinite number of peaks or valleys in any interval, no matter how small, then we will need an infinite resolving power. An example of this particularly frustrating case is given by the Brownian motion.

Let us be given a functional variable $\mathcal{X} : \Omega \to (E, \mathcal{E})$, where E and $\mathcal{E}$ are, respectively, a functional space and its Borel σ-algebra. Then:

Definition 2.3. *The Mean Function is a function $\mu \in E$, such that:*

$$\mu(t) = \mathbb{E}[\mathcal{X}(t)].$$

Definition 2.4. *The Autocovariance, Variance and Standard Deviation Functions are, respectively:*

- $\Sigma(t,s) = cov\,(\mathcal{X}(t), \mathcal{X}(s)) = \mathbb{E}[(\mathcal{X}(t) - \mu(t))(\mathcal{X}(s) - \mu(s))]$;
- $\sigma^2(t) = \mathbb{E}[(\mathcal{X}(t) - \mu(t))^2]$;
- $\sigma(t) = \sqrt{\sigma^2(t)}$.

These quantities are defined exactly as if we were managing random variables, but, obviously, we have to consider the dipendence on the t variable. Now, given the functional dataset $\chi_1, \dots, \chi_n$, we list the sample counterparts of the previous definitions.

Definition 2.5. *The Sample Mean Function is given by:*

$$\bar{\chi}(t) = \frac{1}{n}\sum_{i=1}^{n} \chi_i(t),$$

Definition 2.6. *The Sample Autocovariance, Sample Variance and Sample Standard Deviation Functions are given by:*

- $S(t,s) = \frac{1}{n-1}\sum_{i=1}^{n}\left[(\chi_i(t) - \bar{\chi}(t))(\chi_i(s) - \bar{\chi}(s))\right]$;
- $s^2(t) = \frac{1}{n-1}\sum_{i=1}^{n}\left[\chi_i(t) - \bar{\chi}(t)\right]^2$;
- $s(t) = \sqrt{s^2(t)}$.

We will focus our attention on data, rather than on the distribution that generated them. This leads us to consider an entire function as a single datum. The opposite choice is typical, instead, to the field of *nonparametric functional statistics*, but we will not go through that way.

2.2.2 Some problems connected with Functional Data

When we are dealing with functional data, we immediately face some important statistical problems. The larger is the space E in which the variables are taking their values, the sparser are the data. In case of functional data, we know, by the nature itself of the data, that E is an infinite dimensional space. So we have to consider essential problems of high (i.e., infinite) dimensional data.
As in Statistics one is usually interested in variations between phenomena, the sparseness notion is strongly linked with the way used to measure closeness between data. So, the first concept to be made precise is that of distance. In the case of finite dimensions this is usually done by choosing a norm in that space, but this choice is not crucial (except, at least, for some constants), because of the equivalence between norms in $\mathbb{R}^n$. Even if we take a

Mahalanobis distance, we encounter no problems, because it is induced by a positive definite matrix and we can define a suitable scalar product, giving us a corresponding norm still equivalent to any other in that space. On the other hand, in an infinite dimensional space it is no more true that every norm is equivalent to any other and the presence of a Hamel basis complicates the situation. Therefore we have to pay more attention to the selection of the norm to use and different choices lead to different results or could not ensure convergence results any more. In some cases (see for example [2]), a choice of a seminorm, or a semimetric, can be even more appropriate. A seminorm, respectively a semimetric, is a function satisfying all the properties of norms, respectively distances, except for the fact that, now, it is no more true that $\|x\|_E = 0$ implies $x = 0$, respectively $d(x, y) = 0 \Rightarrow x = y$. If the data considered are regular functions in $L^2(E)$, i.e. provided with all the derivatives we need, an example of seminorm could be

$$\left(\int |\chi_i^{(m)}(t) - \chi_j^{(m)}(t)|^2 dt \right)^{\frac{1}{2}},$$

that is, the pairwise difference in $L^2(E)$ of the m-th derivative of the functions $\chi_i(t)$ and $\chi_j(t)$. However, there is never a unique answer to the question of what kind of norm or seminorm should be used: it all depends on the particular analyses we want to carry out.
The sparsity of the data in high dimensional spaces leads to another problem well known by statisticians: *the curse of dimensionality*. It is, geometrically, the disposing, of the data, close to the boundaries of the space when the number of the observed variables p becomes large. Indeed it is very interesting to ask what happens with the curse of dimensionality if we work with functional data. If we have n observations lying in $\mathbb{R}^p$, one way to illustrate the curse of dimensionality is to count the number $N(p)$ of units falling into a subset, of fixed size, of $\mathbb{R}^p$ when p takes successive values $(1, 2, \ldots)$. Following the same idea, if we have n functional observations lying in a (semi)metric space (E, d), we will count the number N_d of units falling into a subset, of fixed size, of E. If we take into account the functional feature of the data, we can compute the quantity N_d defined as:

$$N_d = \sum_{i=1}^{n} I_{\left\{ \frac{d(\chi_i(t),0)}{\max d(\chi_i(t),0)} < 0.1 \right\}}(\chi_i(t)),$$

where, I is the usual indicator function and d denotes the chosen functional measure of closeness. It appears, in practice, that the curse of dimensionality does not affect one-dimensional functional data when they show a high correlation structure, but it continues to be a serious problem in the other

cases. Obviously, a crucial challenge is pointed out here: the choice of the measure of closeness d, which is, we remember, strictly related to practical considerations. Other considerations about these questions could be find, for instance, in [2].
Another complication, worth to be mentioned and still linked to the curse of dimensionality, is related to the computational aspect of the kind of data we have to manage. As we have already noticed, we could need a huge amount of informations to treat, on a calculator, functional data, or, said in other terms, a high resolution. This leads to the problem of the choice of the grid used to handle numerically the data and, obviously, to issues concerning memory, time and velocity of the software for functional data analysis. Evidently, a finer grid brings to a higher resolution, but it can also result in more memory and time used for the computation. On the other hand, a coarser grid means less time and memory spent, but also a loss of informations in the discretization process of a function.

2.3 Sparse Clustering in Infinite Dimensions

Now we extend the same sparse clustering method in an infinite dimensional space. Precisely, we assume that the data we have to deal with are functions. The complexity we now face with is greatly augmented with respect to the finite dimension case, because every point in the domain of the data is to be considered a feature, so we actually have an uncountable infinity of variables we are looking at.
Let us precise the setting of this new problem. For simplicity, we assume that the whole set of functions data are defined on the same domain $D \subset \mathbb{R}$, which is a Borel set and the image space is one dimensional:

$$f_j : D \to \mathbb{R},\ \forall j \in J, \tag{2.1}$$

where J is a set of indices. To avoid problems concerning how to measure sets in an infinite dimensional space, we will always suppose that J is a set of finite cardinality. This means that the number of data is finite, which is so in every statistical application and we consider the number of observations to be equal to $N \in \mathbb{N}$, with $N < \infty$. Given the data, we have to define an appropriate dissimilarity measure. The straightforward extension of the finite dimensional case is to consider the squared L^2 norm of pairwise differences of functions, that is:

$$d_{i,j} = d(f_i, f_j) = \int_D (f_i - f_j)^2 \, d\mu,\ \forall i, j = 1, \ldots, N,\ i \neq j, \tag{2.2}$$

which we can rewrite as:

$$d_{i,j} = d(f_i, f_j) = \|f_i - f_j\|^2_{L^2(D)}, \ \forall i, j = 1, \dots, N, \ i \neq j. \tag{2.3}$$

We chose the L^2 norm because we want to perform a K-means procedure, which is defined by taking Euclidean distances between points in the space (see Pollard, [13]) and because it gives particular properties to the space that other norms would not guarantee. It is not, naturally, the unique choice of distance we could take, but different alternatives lead to different methods.
From this definition of dissimilarity measure, we have to consider every f_j belonging to $L^2(D)$, giving us a Hilbert space structure. We consider the functions f_j's, which, to be rigorous we should have indexed also with k, as realizations of some functional random variables $\mathcal{X}_k : \Omega \to L^2(D)$ with $k = 1, \dots, K$, where K is the number of clusters that we, for the moment, hold fixed. That is, we are supposing that K functional random variables are generating the whole set of data and we have to cluster the observations basing on this information. The $\mathcal{X}_k$'s are defined on the same probability space $(\Omega, \mathcal{F}, \mathbb{P})$, where Ω is the event space, $\mathcal{F}$ is a suitable σ-algebra on Ω and $\mathbb{P}$ is an appropriate probability measure on $\mathcal{F}$.
The measure μ used in the definition of the dissimilarity measure should be positive. For instance Lebesgue's measure $\mathcal{L}$ or any other positive measure absolutely continuous with respect to $\mathcal{L}$. From now on, we will always assume μ to be the Lebesgue's measure. We moreover assume that $\mu(D) < \infty$, so that, μ, could be made a probability measure after a normalization. Thanks to Hölder's inequality we immediately get, also: $f_j \in L^1(D)$ for every $j = 1, \dots, N$.
We could also consider the image space to be $\mathbb{R}^n$, with $n > 1$; in this case we have to consider $f_j \in L^2(D; \mathbb{R}^n)$ and the dissimilarity measure becomes the sum, across the components, of the squared L^2 norm pairwise differences:

$$d_{i,j} = d(f_i, f_j) = \sum_{m=1}^{n} \int_D \left(f_i^m - f_j^m\right)^2 d\mu, \ \forall i, j = 1, \dots, N, \ i \neq j. \tag{2.4}$$

We focus, for the moment, on the case of a one dimensional image space.
In the functional case we define the Total Within Clusters Sum of Squares (WCSS) as

$$\sum_{h=1}^{K} \frac{1}{|2C_h|} \sum_{i,j \in C_h} \int_D (f_i - f_j)^2 \, d\mu, \tag{2.5}$$

where $|C_h|$ is the cardinality of the group C_h and the coefficient 2 is taken to avoid the sum of two equal terms. As a consequence, the Between Clusters

Sum of Squares (BCSS) becomes:

$$\int_D \left\{ \frac{1}{2N} \sum_{i,j=1}^{N} (f_i - f_j)^2 \, d\mu - \sum_{h=1}^{K} \frac{1}{|2C_h|} \sum_{i,j \in C_h} (f_i - f_j)^2 \right\} d\mu, \tag{2.6}$$

that is the Total Sum of Squares, represented by the first term, to which we subtract the total WCSS. We note that this allows us to write the following expression about the Variance Decomposition:

$$BCSS = TSS - WCSS.$$

If we were to solve the following problem:

$$\max_{(C_1,\dots,C_K)} \int_D \left\{ \frac{1}{2N} \sum_{i,j=1}^{N} (f_i - f_j)^2 \, d\mu - \sum_{h=1}^{K} \frac{1}{|2C_h|} \sum_{i,j \in C_h} (f_i - f_j)^2 \right\} d\mu, \tag{2.7}$$

we would find the classical functional K-Means solution. We are, instead, interested in sparse clustering, so we have to make some feature selection and integrate it into the formulation of the problem. As we are dealing with an uncountable infinity of variables, we notice that we need a weight function $w(x)$ to do this work. Precisely:

$$w : D \to \mathbb{R}, \text{ with } w(x) \geq 0, \ \mu - a.e. \tag{2.8}$$

w is defined on the same domain of the functions f_j's and nonnegative almost everywhere on D; if some point $\bar{x} \in D$ or an entire Borel set $B \subset D$ is not involved in the clustering process, or, said in an alternative way, the data are almost indistinguishable on that point or on that set, then $w(\bar{x}) = 0$ or $w(x) = 0$ for every $x \in B$. Otherwise, if some points or entire sets have a role in the clustering, then w will be strictly positive and as much greater as that point or that set are important in the distinction of the groups. Thus if we write the functional sparse clustering problem, we obtain:

$$\max_{w(x),(C_1,\dots,C_K)} \int_D w(x) \left\{ \frac{1}{2N} \sum_{i,j=1}^{N} (f_i - f_j)^2 \, d\mu - \sum_{h=1}^{K} \frac{1}{|2C_h|} \sum_{i,j \in C_h} (f_i - f_j)^2 \right\} d\mu. \tag{2.9}$$

We must pose some constraints on w. The first requirement is that the weight function belongs to $L^2(D)$ and, specifically that it belongs to the closed ball of radius 1:

$$\|w(x)\|^2_{L^2(D)} \leq 1. \tag{2.10}$$

If the norm were not limited, the solution to the problem (2.9) would not exist: indeed, we could always find a function increasing the value of the functional considered and this process would not end, resulting in a maximum growing to infinity and in useless clusters: every randomly chosen grouping is optimum in the same way. The choice of the unit closed ball comes from a normalization reason: w should increase the value of the functional in (2.9) only by feature selection or functions reassignements and not by augmenting its own values, as, evidently, the greater the norm of a function is, the greater are the values it can assume.
The second constraint is what truly determines the sparsity in the clustering. Note that, by Hölder's inequality, we readily have:

$$\|w(x)\|_{L^1(D)} = \int_D w(x) d\mu \leq \sqrt{\mu(D)} \|w(x)\|_{L^2(D)} \leq \sqrt{\mu(D)}. \tag{2.11}$$

As in the finite dimensional case, the constraint on the L^1 norm of the weight functions is what truly specifies sparsity in clustering. Indeed, if we define the set W_s as:

$$W_s = \left\{ u \in L^2(D) : \|u(x)\|_{L^1(D)} \leq s,\ \|u(x)\|^2_{L^2(D)} \leq 1, u(x) \geq 0\ \mu - a.e. \right\}, \tag{2.12}$$

we see that

$$W_t \subset W_s, \text{ if } 0 < t < s, \tag{2.13}$$

and if we let

$$W := \bigcup_s W_s,$$

where the union is done over all admissible s, specifically, $0 < s \leq \sqrt{\mu(D)}$, we obtain

$$W = \left\{ u \in L^2(D) : u(x) \geq 0\ \mu - a.e.,\ \|u(x)\|^2_{L^2(D)} \leq 1 \right\},$$

which, in particularly, contains the solution of a nonsparse K-Means. From this we deduce, again, that, the greater is s, the less the sparsity will result in the final clustering. The tuning parameter s, therefore, assumes again the role of "measure of sparsity" of the resulting clustering. This happens because (and will be even more clear in the construction of the weight function) the L^1 norm direclty controls the amplitude and the extension, over the domain, of $w(x)$.
Having described the context and the path that leads to the specific properties we ask to the weight function w, we can formally define what we mean by *Sparse Functional Clustering*.

Definition 2.7. *We define Sparse Functional K-means Clustering the solution to the following problem:*

$$\max_{w(x),(C_1,\dots C_k)} \int_D w(x) \left(\frac{1}{2N} \sum_{i,j=1}^{N} (f_i - f_j)^2 - \sum_{h=1}^{k} \frac{1}{|2C_h|} \sum_{f_i, f_j \in C_h} (f_i - f_j)^2 \right) d\mu,$$

$$\textit{subject to: } \|w(x)\|^2_{L^2(D)} \leq 1,\ \|w(x)\|_{L^1(D)} \leq s,\ w(x) \geq 0\ \mu - \textit{a.e.} \tag{2.14}$$

Before going deeply in the solution of this problem, let us make some brief comments.
We note, again, that, if we take $w(x)$ constant over the domain D and

$$w(x) = \min \left\{ \frac{1}{\sqrt{\mu(D)}}, \frac{s}{\mu(D)} \right\},$$

we find the solution to the classical K-means. If the measure μ considered is the counting measure, we readily fall in the finite dimensional case discussed in (1.10). The tuning parameter s is assigned basing on specific outer informations about the functions or by other computational methods. Note that, the more we increase s, the greater will be the value of the objective functional and the less the sparsity will result, leading to a method that confounds with the classical K-means. As a result, there may need a little effort in choosing the right L^1 constraint. A good weight function solution to the problem (2.14) should, besides helping the classification, identify subsets, if any, of the domain where it is useless clustering the functions because they have the same distribution, at least blurred by some noise.

2.3.1 The Main Result

The complexity of the problem forces us to look for an algorithm to find the solution, but, first we need to know if the solution of (2.14) exists and it is unique.
Fortunately, we have the following result.

Theorem 2.1.
Let $D \subset \mathbb{R}$ be a Borel set, such that $\mu(D) < \infty$, $f_i \in L^2(D), f_i \in L^\infty(D)\ \forall i = 1, \dots, N$ with $\|f_i\|_{L^\infty(D)} \leq M\ \forall i$, with $M \in \mathbb{R}$. Then there exists a unique solution to the following problem:

$$\max_{(w(x),C_1,\dots C_k)} \int_D w(x) \left(\frac{1}{2N} \sum_{i,j=1}^{N} (f_i - f_j)^2 - \sum_{h=1}^{k} \frac{1}{|2C_h|} \sum_{f_i, f_j \in C_h} (f_i - f_j)^2 \right) d\mu, \tag{2.15}$$

where $k < N$ is the (fixed) number of clusters, C_i is the i-th cluster and the function $w(x)$ must satisfy the following constraints:

$$\|w(x)\|^2_{L^2(D)} \leq 1, \;\|w(x)\|_{L^1(D)} \leq s, \; w(x) \geq 0 \; \mu - a.e.$$

and s, the tuning parameter, is fixed.

Proof.
The proof will proceed in three steps.
1^{st} **step.**
Let $C_1, \dots, C_k$ be a fixed clustering of functions. Let us consider a sequence of simple functions $w_n(x)$ defined in this way:

$$w_n(x) = \alpha \sum_{l=1}^{n} c_l I_{A_l}(x), \tag{2.16}$$

where $I_{A_l}(x)$ is the indicator function of the subset A_l,

$$c_l = \left(\frac{1}{2N} \sum_{i,j=1}^{N} \frac{1}{\mu(A_l)} \int_{A_l} (f_i - f_j)^2 - \sum_{h=1}^{k} \frac{1}{|2C_h|} \sum_{f_i, f_j \in C_h} \frac{1}{\mu(A_l)} \int_{A_l} (f_i - f_j)^2 \, d\mu \right) \tag{2.17}$$

and the A_l are such that $A_i \cap A_j = \emptyset$ if $i \neq j$, $\bigcup_{l=1}^{n} A_l = D$ and $\mu(A_l) = \frac{\mu(D)}{n}$. The parameter α is to be chosen to make the resulting function $w_n(x)$ satisfy the constraints $\|w_n(x)\|_{L^2(D)} \leq 1, \; \|w_n(x)\|_{L^1(D)} \leq s, \; w_n(x) \geq 0 \; \mu - a.e.$. Substantially we are building the maximizing sequence in such a way that it results proportional, on subintervals of the domain, to the local Between Clusters Sum of Squares (BCSS). Now, the problem we face becomes:

$$\max_{\alpha} \int_D w_n(x) \left(\frac{1}{2N} \sum_{i,j=1}^{N} (f_i - f_j)^2 - \sum_{h=1}^{k} \frac{1}{|2C_h|} \sum_{f_i, f_j \in C_h} (f_i - f_j)^2 \right) d\mu, \tag{2.18}$$

with α making the $w_n(x)$'s satisfying the previous constraints. The optimization problem so derived, is simply a constrained linear maximization problem. Indeed, if we treat the function to be maximized as a function of α, we see that it is a line with a positive angular coefficient and, for every fixed n, we have an explicit solution in terms of α:

$$\int_D w_n(x) \left(\frac{1}{2N} \sum_{i,j=1}^{N} (f_i - f_j)^2 - \sum_{h=1}^{k} \frac{1}{|2C_h|} \sum_{f_i, f_j \in C_h} (f_i - f_j)^2 \right) d\mu = \tag{2.19}$$

$$\alpha \int_D \sum_{l=1}^{n} c_l I_{A_l}(x) \left(\frac{1}{2N} \sum_{i,j=1}^{N} (f_i - f_j)^2 - \sum_{h=1}^{k} \frac{1}{|2C_h|} \sum_{f_i, f_j \in C_h} (f_i - f_j)^2 \right) d\mu = \tag{2.20}$$

$$\alpha \sum_{l=1}^{n} \frac{1}{\mu(A_l)} \left(\frac{1}{2N} \sum_{i,j=1}^{N} \int_{A_l} (f_i - f_j)^2 \, d\mu - \sum_{h=1}^{k} \frac{1}{|2C_h|} \sum_{f_i, f_j \in C_h} \int_{A_l} (f_i - f_j)^2 \, d\mu \right)^2 . \tag{2.21}$$

It suffices to choose the maximum value such that the constraints are satisfied:

$$\alpha^* = \min \{\alpha_1, \alpha_2\}, \tag{2.22}$$

where

$$\alpha_1 = \frac{s}{\sum_l \left(\frac{1}{2N} \sum_{i,j=1}^{N} \int_{A_l} (f_i - f_j)^2 - \sum_{h=1}^{k} \frac{1}{|2C_h|} \sum_{f_i, f_j \in C_h} \int_{A_l} (f_i - f_j)^2 \right)}, \tag{2.23}$$

$$\alpha_2 = \frac{1}{\sqrt{\sum_l (d_l)^2 \mu(A_l)}} \tag{2.24}$$

and we have indicated with d_l the term in brackets in (2.17). Let us call the $w_n(x)$ with the optimum α^*, $\hat{w}_n^*(x)$ and consider, in $L^2(D)$, the sequence $\{\hat{w}_n^*(x)\}_n$. From the boundedness of the $\hat{w}_n^*(x)$ in L^2, thanks to the Banach-Alaoglu's Theorem, we have that we can extract a subsequence, which we still call $\hat{w}_n^*(x)$ weakly converging to a function $w^*(x) \in L^2(D)$, depending on the particular clusters taken and still solving the maximum problem, with $\|w^*(x)\|_{L^2(D)} \leq 1$ and, moreover, with $w^*(x) \in L^1(D)$, from the finitedness of $\mu(D)$. We have to verify that the limit $w^*(x)$ still satisfies the constraints. From the definition of weak convergence:

$$\int_D \hat{w}_n^*(x)\varphi(x) d\mu \to \int_D w^*(x)\varphi(x) d\mu, \; \forall \varphi \in L^2(D).$$

Thanks to the finitedness of $\mu(D)$, we have that the indicator function of the domain D and, of course, of every Borel set B contained in D, belongs to L^2 whence:

$$\int_B \hat{w}_n^*(x) d\mu \to \int_B w^*(x) d\mu, \; \forall B \in \text{Borel}(D),$$

which assures us the positivity $\mu-$a.e. of the limit $w^*(x)$ and also:

$$s \geq \|\hat{w}_n^*(x)\|_{L^1(D)} = \int_D \hat{w}_n^*(x) d\mu \to \int_D w^*(x) d\mu = \|w^*(x)\|_{L^1(D)}.$$

At a first sight the limit function $w^*(x)$ seems to be depending on the particular partition $\{A_j\}_j$ chosen to decompose the domain D. In reality, it can be shown that not only a maximizing sequence should be constructed in the way we proceeded, but it is also independent of the partition of D taken. This is the reason why we did not add a second index in the definition of the A_j's to take into account of the particular choice of the partition. Finally, for every fixed choice of the clusters, the limit function is unique thanks to this and the uniqueness of weak limits.

<u>2nd **step.**</u>

Now let $w_n(x)$ be fixed and satisfying the requested constraints. Let us consider the problem:

$$\max_{(C_1,\dots C_k)|w_n(x)} \int_D w_n(x) \left(\frac{1}{2N} \sum_{i,j=1}^{N} (f_i - f_j)^2 - \sum_{h=1}^{k} \frac{1}{|2C_h|} \sum_{f_i, f_j \in C_h} (f_i - f_j)^2 \right) d\mu.$$

This is just a weighted clustering problem and we know how to solve it. Indeed it is only a maximization over a finite number of possible groupings; then we choose the clusters giving us the maximum value of that integral. The best clustering always exists (because of the finite number of possible cases) and, moreover, we assume that it is unique. This assumption is not restrictive: it is confirmed in the applications, except, eventually, for pathological cases, or, even if we occurr in a non unique best clustering, we simply choose one and go on.

<u>3rd **step.**</u>

Finally we have to prove that there exists a solution to the original problem with $w_n(x)$ and the clusters both varying. To this purpose, we start by setting $w_1(x) = \frac{1}{\sqrt{\mu(D)}}$ everywhere on D and we look for the solution of the problem with that function. It is just a normal (i.e. non weighted) clustering problem and we obtain the optimal clusters. Given this optimal clustering, we search for the optimal function and then we proceed in this manner iteratively. In the end, we want to show that this algorithm converges to an optimal solution. Focus on the term in brackets in (2.15), which we now call $h(x, C_1, \dots, C_k)$. Considered as a function of the only x, with a little abuse of notation, it belongs to $L^2(D)$. Indeed:

$$\int_D (h(x))^2 \, d\mu < \int_D 4M^4 d\mu = 4M^4 \mu(D) < \infty, \tag{2.25}$$

thanks to our hypotheses on the data functions. We can see that, for how we constructed the algorithm defined by the above iterative procedure and the uniqueness assumption made in the second step, we obtain sequences $\{w_n^*(x)\}_n$ (we have now indicated with $w_n^*(x)$ the weak-limit function $w^*(x)$ found at the n-th application of the algorithm) and $\left\{(C_{1,n}^*, \dots, C_{k,n}^*)\right\}_n$ of optimal clusters, satisfying, at the n-th step:

$$\begin{aligned} \int_D w_1^*(x) h(x, C_{1,1}^*, \dots, C_{k,1}^*) d\mu &\le \int_D w_2^*(x) h(x, C_{1,2}^*, \dots, C_{k,2}^*) d\mu \quad (2.26) \\ \le \dots &\le \int_D w_n^*(x) h(x, C_{1,n}^*, \dots, C_{k,n}^*) d\mu. \end{aligned}$$

Let us suppose, for a moment, that the sequence of the optimal clusters keeps varying at every stage of the algorithm. Surely, because of the finite number of possible groupings, there are at least two optimal clusterings repeating. So, suppose that, at steps n and m, with $n < m$, we have found the same clusters and let us compare the corresponding optimal functions: $w_n^*(x)$ and $w_m^*(x)$. Then, it must happen that $w_n^*(x) = w_m^*(x)$ for how we defined the maximizing sequence and the uniqueness of weak limits. For if we suppose, for example, $w_n^*(x) < w_m^*(x)$, then $w_n^*(x)$ can not be the optimal function of the n-th passage, because we can augment the value of the functional (2.15) by simply put $w_m^*(x)$ instead of $w_n^*(x)$ and we can do this, because the clusters are the same. Therefore the algorithm must end after a finite number of passages and the proof is so concluded. □

Let us make some comments on the Theorem. We consider as given the parameter s and the number of clusters K. The hypotheses requested about the functions f_i's are quite natural: firstly they have to belong to the space $L^2(D)$, because that is our functional setting where we are supposing the variables $\mathcal{X}_k$'s are taking their values, then we want their L^∞ norm to be limited, as the data are thought to be measurements, over a limited period of time or some other continuum, of a physical process and, therefore, they have to be equibounded. Otherwise there would be need an infinite quantity of energy to observe the process and this is obviously impossible.
The constructive way in which we proved the theorem suggests us how to build an algorithm to find the desired solution with a computer:

1. Maximize with respect to the weight function w holding the clusters fixed. In particular, the first maximization of this kind is given taking the clusters coming from an ordinary K-means procedure;

2. Maximize with respect to the K clusters holding fixed the weight function given by the previous step.

This scheme is repeated until convergence of the function w. Practically this is achieved by a pre-determined stopping criterion which is given by normalizing the L^1 norm of the difference of two weight functions at two consecutive stages of the algorithm and assessing if this quantity is less than a given threshold.
Moreover, the optimal weight function is proportional to the local BCSS on given sets of the domain and, at every step of our procedure, those sets are splitted again into more subsets, so that the solution detects better the difference between the clusters.
If we refer back to the vectorial case, however, we immediately see that we

have an explicit analytical form for the solution vector $\mathbf{w}$. We can, anyway, say something about the optimal weight function in the functional case if we slightly strengthen the hypotheses on the data.

Theorem 2.2.
Let f_i's be functions such that: $f_i \in L^2(D)$, $f_i \in L^\infty(D)$, with $\|f_i\|_{L^\infty(D)} \leq M$, $\forall i = 1, \dots, N$. Suppose, moreover, $f_i \in C(D)$ for every $i = 1, \dots, N$. Define

$$b(x) := \frac{1}{2N} \sum_{i,j=1}^{N} (f_i - f_j)^2 (x) - \sum_{h=1}^{k} \frac{1}{|2C_h|} \sum_{f_i, f_j \in C_h} (f_i - f_j)^2 (x), \tag{2.27}$$

which is well defined thanks to the hypotheses on the f_i's and $b(x) \in L^2(D)$. Then

$$\sum_{l=1}^{n} c_l I_{A_l}(x) \longrightarrow b(x) \text{ in } L^2(D). \tag{2.28}$$

Note, also, that $\| \sum_{l=1}^{n} c_l I_{A_l}(x) \|_{L^1(D)} = \|b(x)\|_{L^1(D)}$.

Proof.
We have that, as $n \to \infty$,

$$\sum_{l=1}^{n} c_l I_{A_l}(x) \to b(x) \ \mu - a.e., \tag{2.29}$$

because $[\mu]$ −a.e. $x \in D$ is a Lebesgue's point to the f_i's (actually, every point, thanks to continuity). Then:

$$|\sum_{l=1}^{n} c_l I_{A_l}(x) - b(x)|^2 \leq |\sum_{l=1}^{n} c_l I_{A_l}(x)|^2 + |b(x)|^2 \leq 8M^4 I_D(x) \tag{2.30}$$

and $8M^4 I_D(x) \in L^2(D)$. Then (2.28) follows from Lebesgue's Dominated Convergence Theorem. □

Thus we have that the solution function, i.e. the optimal weak limit, is proportional to what we have called $b(x)$ that is the function that follows exactly the BCSS point by point (it is clear that $b(x)$ depends also on the particular clusters taken, but we have supressed it in the notation). This happens because if $f_n \to f$ weakly in L^p, with $1 \leq p \leq \infty$ and $f_n \to g$ μ−a.e., then $f = g$. Naturally, because $L^2(D) \subset L^1(D)$, whenever D has finite measure, we readily have also that $\sum_{l=1}^{n} c_l I_{A_l}(x) \to b(x)$ in $L^1(D)$. The good definition of $b(x)$ comes from the continuity of the f_i's: we need functions which assume a value in *every point* of the domain to have such a fair posedness. If we know that D is a compact subset of $\mathbb{R}$, the hypothesis

to belong to the spaces L^2 and L^∞ becomes redundant. The requests of Theorem 2.2 are quite weak. Indeed, if our dataset is composed by discrete observations of a physical or some other process and we want to treat them as functions, we can smooth the data at an arbitrary degree, in order to have very regular curves.
In the end of this section, we want to look at the error, eventually made, by this method. We can surely state, given the optimal weigth function $w(x)$ and the optimal clusters $(C_1, \dots, C_K)$, that:

$$f_i \in C_k \Leftrightarrow \frac{1}{|C_k|} \int_D w(x) \sum_{j \in C_k} (f_j - f_i)^2 d\mu \leq \frac{1}{|C_h|} \int_D w(x) \sum_{j \in C_h} (f_j - f_i)^2 d\mu, \tag{2.31}$$

for every $h, k = 1, \dots, K$, with $h \neq k$. Thus, summing the first term of the inequality over all functions and all clusters, we obtain the quantity

$$\sum_{k=1}^{K} \frac{1}{|2C_k|} \int_D w(x) \sum_{i,j \in C_k} (f_j - f_i)^2 d\mu, \tag{2.32}$$

which is just the weighted $WCSS$ and therefore we can say that problem (2.15) is equivalent to minimize this quantity, over the set of admissible functions and the set of possible clusters. Note that, if we had started by trying to minimize the weighted $WCSS$, we could have found a useless result: a $w(x)$ constantly equal to 0 and any randomly chosen clusterization from the set of all possible groups. Moreover, it is easier to find and characterize the solution with the maximization of the $BCSS$. So, we want the weighted $WCSS$ to be as small as possible and this suggests us that we can somehow interpret it as a measure of the departure of the functions from their hypothetical original groups, or, said in another way: as a measure of "committed error in the clustering process". This interpretation allows us to state an important property about the error.

Proposition 2.1.
The committed error is monotone nonincreasing.

Proof.
The proposition follows rather immediately thanks to inequalities (2.26) □

Moreover, if we think that at the first stage of the algorithm we have the solution to classical K-means, we have, also, that the committed error is less than, or equal to, that committed by the classical method.
On the contrary, we can claim that the optimal value of the functional considered in our problem is greater than, or equal to, that of the same functional with $w(x)$ constant, which is the solution to classical K-means, otherwise the

new method would be useless.
There is a drawback affecting both methods, coming from the fact that they do not take into account the stochastic nature of the problem, but focus only on distances between realizations of the functional variables rather than to the functional variables themselves which stand at the origin of the problem. For this reason it could be possible the two methods assign data to the wrong cluster. Not only: suppose there exists a subset N of the domain D such that the trends of the various clusters are very similar, in other words such that the distances are almost zero, but there also exists a particular partition $(G_1, \dots, G_K)$ of the data into groups, which wrongly results in assigning positive distances between groups in that subset. Then we can write, given the optimal weight function u for this clusterization:

$$\begin{aligned}\int_D u(x)b(x; G_1, \dots, G_K)d\mu &= \int_N u(x)b(x; G_1, \dots, G_K)d\mu + \\ &+ \int_{D \setminus N} u(x)b(x; G_1, \dots, G_K)d\mu \end{aligned} \tag{2.33}$$

and both terms are positive. Now, given the real clusters $(C_1, \dots, C_K)$ and the real optimal weight function w, it could be possible that:

$$\int_D w(x)b(x; C_1, \dots, C_K)d\mu \leq \int_D u(x)b(x; G_1, \dots, G_K)d\mu, \tag{2.34}$$

because $\int_N w(x)b(x; C_1, \dots, C_K)d\mu$ is negligible and thus the method would select the wrong solution $u(x)$ and $(G_1, \dots, G_K)$. However, if this happens for weighted K-means it is even more so for classical K-means. A proposed solution to overcome this drawback will be stated at the end of Chapter 3.

2.4 An Analytical Example

We now want to test our method on an easy example where we can execute analytically all the computations needed. This particular case allows us to verify the previous results and confirm the predictions we can make before getting through any calculation. The data are the following:

- Group 1: $\{f(x) : \; f_i(x) = \mathcal{M}_i x, \forall i = 1, \dots, N\}$
- Group 2: $\{g(x) : g_i(x) = \mathfrak{M}_i x, \forall i = 1, \dots, N\}$,

that is, the groups are simply lines, considered on the compact interval $D = [0, 1]$, passing through the origin of axes, but distinguishing for their angular coefficients, which are random variables, and N is taken equal to 20. We only

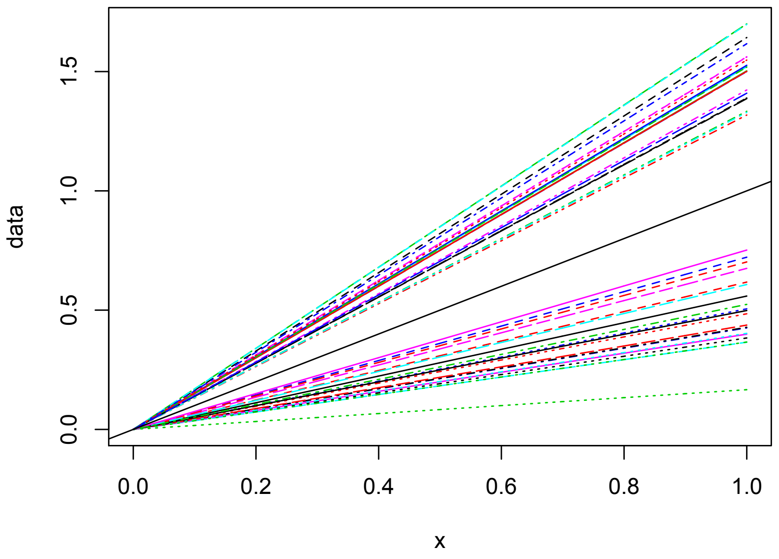

Figure 2.1: The analytical case.

have to specify the distribution of the $\mathfrak{M}_i$'s and the $\mathcal{M}_i$'s. We, therefore, consider: $\mathfrak{M}_i \overset{i.i.d.}{\sim} \mathcal{N}(\mu_1, \sigma^2)$, $\mathcal{M}_j \overset{i.i.d.}{\sim} \mathcal{N}(\mu_2, \sigma^2)$, with $\mu_1 > \mu_2$ and $\mathfrak{M}_i$ independent of $\mathcal{M}_j$, for every $i, j = 1, \ldots, N$. Finally, we indicate with M_i and m_j, respectively, $i, j = 1, \ldots, N$ a random sample from the $\mathfrak{M}_i$'s and the $\mathcal{M}_j$'s.

Figure 2.1 shows a plot of this case. We have taken $\mu_1 = \frac{3}{2}$, $\mu_2 = \frac{1}{2}$ and $\sigma^2 = \frac{1}{8}$. We have traced, in black, the bisector of the first quadrant, that is the ideal line dividing the two clusters. The choice of these particular values for the parameters is justified by the fact that we already know the right clusters and because we want to focus, for the moment, on the solution function $w(x)$.

Before finding the maximum of that functional, we can try to make some predictions on the solution. It will surely start from the origin: in that point the two groups are perfectly coincident, therefore, 0, is not a point that occur in the clustering process. Then we see that, as long as we move from the origin, the two families of functions are more and more distinct, whence w will be increasing.

Now, the problem we have to solve is:

$$\max_{w(x),(C_1,C_2)} \int_D w(x) \frac{1}{N} \sum_{i,j=1}^{N} (f_i - g_j)^2 dx + \\ - \int_D \frac{w(x)}{2} \left(\frac{1}{|C_1|} \sum_{f_i, f_j \in C_1} (f_i - f_j)^2 + \frac{1}{|C_2|} \sum_{g_i, g_j \in C_2} (g_i - g_j)^2 \right) dx, \tag{2.35}$$

with the usual constraints on w: $w(x) \geq 0$ a.e., $\|w(x)\|_{L^1(D)} \leq 1$ and $\|w(x)\|^2_{L^2(D)} \leq 1$. Note that, here, s equals 1, because the two groups are different essentially on every point in D and this, besides the well-separateness of the curves, forces the optimum clusters to coincide with those found by the classical K-means and the real ones. The previous expression, after having written in explicit form the functions and having managed the terms, becomes:

$$\max_{w(x),(C_1,C_2)} \int_D w(x) \frac{1}{N} \sum_{i,j=1}^{N} (M_i - m_j)^2 x^2 dx + \\ - \int_D w(x) \left(\frac{1}{2N} \sum_{i,j} (m_i - m_j)^2 + \frac{1}{2N} \sum_{i,j} (M_i - M_j)^2 \right) x^2 dx. \tag{2.36}$$

For simplicity, we rewrite the problem as:

$$\max_{w(x)} \int_D w(x) \eta x^2 dx, \tag{2.37}$$

where $\eta = \frac{1}{N} \sum_{i,j=1}^{N} (M_i - m_j)^2 - \frac{1}{2N} \sum_{i,j} (m_i - m_j)^2 - \frac{1}{2N} \sum_{i,j} (M_i - M_j)^2$. As it represents a distance, η is always positive and, theorically, we could even calculate its distribution. Note, that, in general, η should depend explicitly on the clusters, which are unknown, but, in this example, we are allowed to reduce to (2.37).

Finding the solution to the maximization (2.37) is a fact from real analysis, see, for example, [7]. The requested function is of the type $w(x) = \alpha \eta x^2$, where the coefficient α is taken in order to satisfy the constraints on w, precisely

$$\alpha = \min \left\{ \sqrt{\frac{5}{\eta}}, \frac{3}{\eta} \right\}$$

where, these values are obtained by forcing, the solution function, to satisfy the inequalities on its norms. In figure 2.2 we can see that the weigth function

is just a rescalement of the function we called $b(x)$, which, in this example, is simply ηx^2. The calculated α in this case is approximately 0.0015.
To be even more sure of the similarity between the two functions, we have plotted in figure 2.3 the values taken by $w(x)$ against those taken by $b(x)$. Moreover we have overlapped the theorical line these values should follow and we note that they lie very close to that line. The stair-shaped curve described by the points $(w(x), b(x))$, comes from the grid chosen and from the way the algorithm was written.

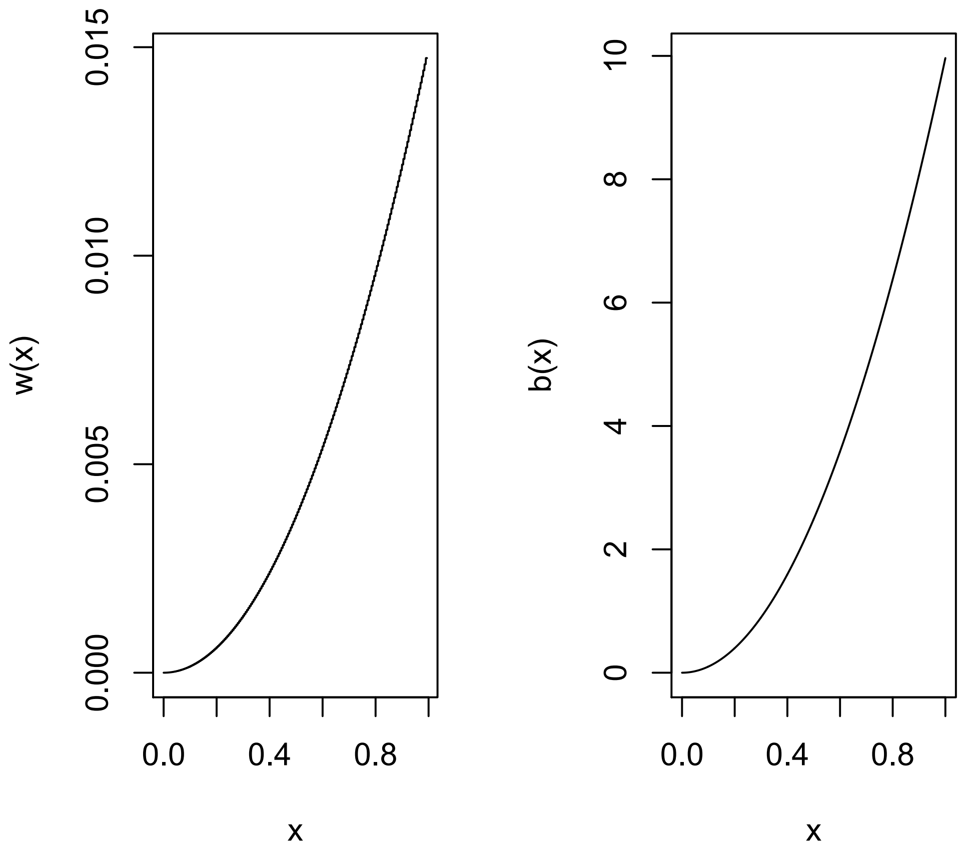

Figure 2.2: Comparison between $w(x)$ and $b(x)$.

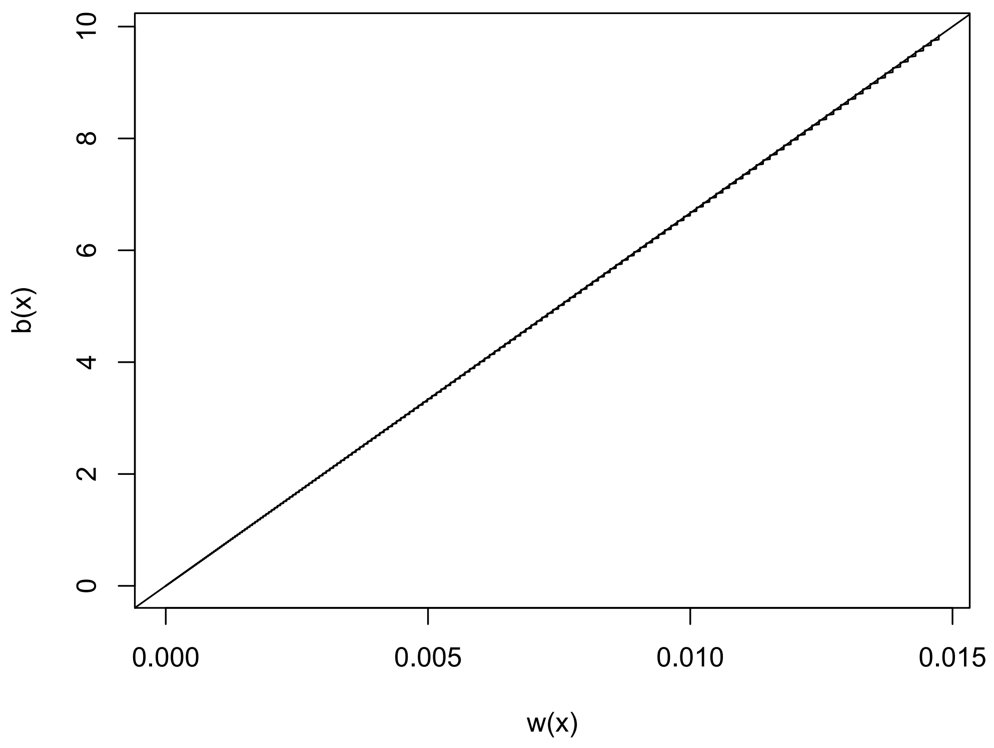

Figure 2.3: Plot of $w(x)$, on the x-axis against $b(x)$, on the y-axis. We can note the characteristic stair-shaped curve and see the hypothetical line these points should follow.

Chapter 3

Simulations

3.1 Introduction

In this chapter we want to test our algorithm on some simulated cases of increasing complexity and compare the obtained clusters to those got by classical K-means. With these *ad hoc* built examples, we can also check the adherence to the real groups and observe the convergence of the (nonnormalized) weight function to what we have called $b(x)$. Moreover, we know, exactly, which is the right tuning parameter s.
Every example will have the same following scheme: 40 curves are simulated, from two clusters, over the interval $[0, 1]$, half coming from a group and the remaining half from the other and we consider only continuous functions for the first two cases, but, then, we will allow the curves to be discontinuous. The grid over which we evaluated the functions is composed by 1000 points. The first two examples are quite easy, since they are composed by functions with a well defined graph. In the first case the groups are formed by positive curves only on a subinterval of the domain, whereas in the second one, they are positive on two different and disjont subsets of the domain. The following examples show a more complicated kind of complexity: besides being different on a part of the set of definition, a noise is added to the curves of one group, then to both groups, but firstly only on a subset of the domain and then in every part of it. The best improvements are seen on these last cases, which is optimal, as we are trying to build a method able to identify different underlying shapes even in very complex cases.

3.2 Simulation Studies

3.2.1 Case 1

This is the simplest case of all, therefore we have chosen the first group to be composed by piecewise linear (affine) continuous functions on a subinterval of the domain and then set equal to zero on the remaining part, so that, the data have a triangular shape. The peak of the triangles are generated from independent random variables $Y_i \sim \mathcal{N}(2, 1/2),\ i = 1, \dots, 20$ and then the coefficients m_1, m_2 and q are computed, for every replication, in order to have continuous functions. Precisely the curves of the first group will have the following expression:

$$f_i(x) = \begin{cases} m_1(Y_i)x & \text{if } x \leq x_0, \\ m_2(Y_i)x + q(Y_i) & \text{if } x_0 < x \leq x_1, \\ 0 & \text{if } x > x_1, \end{cases} \tag{3.1}$$

with $m_1, q > 0$ and $m_2 < 0$.
The other group is composed by parabolas starting from zero, having the maximum in $x = x_0$, then continuously rejoining to zero at $x = x_1$ and, finally, identically equal to zero from $x = x_1$ to 1. Again, the maximum of the parabolas are samped from independent random variables $Z_j \sim \mathcal{N}(\frac{5}{2}, \frac{1}{2}),\ j = 1, \dots, 20$ and the coefficients a and b are calculated in order to have continuos curves:

$$g_j(x) = \begin{cases} a(Z_j)x^2 + b(Z_j)x & \text{if } 0 \leq x \leq x_1, \\ 0 & \text{if } x > x_1. \end{cases} \tag{3.2}$$

We have run this first example with various decreasing values of x_0 and x_1, respectively: $\frac{1}{4}, \frac{1}{2}$, then $\frac{1}{8}, \frac{1}{4}$ and, finally, with $\frac{1}{20}$ and $\frac{1}{10}$. We want the values taken by the random variables of maxima of the two groups to be slightly different from each other, in order to have well shuffled data, so that we can not distinguish them by only looking at them and this motivated the choice for the parameters of the distributions generating the data. We have only reported the last case, that is, that where $x_0 = \frac{1}{20}$ and $x_1 = \frac{1}{10}$. The motivation for this comes from the fact that the larger is the subinterval over which the functions are different, the more the solution clusters are similar to those found by classical K-means, which is obvious, since we are augmenting the part of the domain where the groups are distinct and this implies that the number of points involved in the clustering process is increasing. Indeed, when we run the case with $x_0 = 1/4$ and $x_1 = 1/2$, the two algorithms find almost always the same clusters, with only sligth improvements performed by weighted K-means, as we have observed the reallocation of, at most, three

curves. It is observed that the proposed algorithm performs better than classical K-means, when the domain over which the data are different becomes small, or, said in other terms, when the sparsity becomes small. In figure 3.1, we see a plot of this case. Looking at the curves, it seems very hard, but only for a few number of functions, to tell between the two groups. This effect, though, is useful, in order to test the strength of our algorithm.
For the case shown in figure 3.1, we report a table comparing the performance of the two algorithms.

	Real Group 1 (lines)	Real Group 2 (parabolas)
K-means $\rightarrow$ 1	12	7
K-means $\rightarrow$ 2	8	13

Table 3.1: K-means vs Real clusters in Case 1.

	Real Group 1 (lines)	Real Group 2 (parabolas)
WK-means $\rightarrow$ 1	16	4
WK-means $\rightarrow$ 2	4	16

Table 3.2: Weighted K-means vs Real clusters in Case 1.

In figure 3.2, we can observe the graphs of the two functions $w(x)$ and $b(x)$ and they have the same shape. Obviously, from the point $x_1 = 1/10$, the two curves are identically equal to zero; $b(x) = 0$, $\forall x \geq x_1$ because the two groups are equal and, consequently, $w(x) = 0$, $\forall x \geq x_1$ as those points are not involved in the clustering process. Note, moreover, the symmetrical shape, of the two functions, around the point $x_0 = 1/20$. This particular profile is completely justified by the type of data used: they both start from zero, then have a peak at the point x_1 and, finally, go down again to zero. The distance between the two groups increases up to a maximum, located at 0.034, and this happens thanks to the different shape of the generating prototype functions. Then, the same distance, becomes smaller at the maxima of the two groups, thus we have a local minimum in the graphs of b and w. Beyond this minimum, the distance between groups increases again, with another maximum at $x = 0.066$, but then it decreases definitely to zero. Looking at the figure, it seems that the maxima and the local minimum of w and b are angular points and so not points of differentiability. Indeed these two

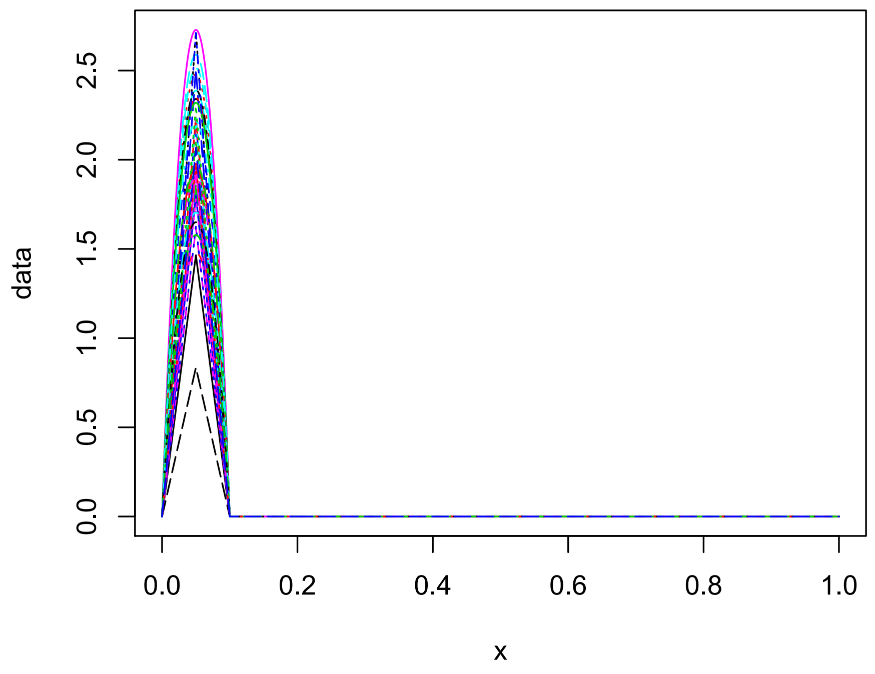

Figure 3.1: A simulation for Case 1

functions are not differentiable in $x = 1/20$ and this comes from the resolution of the analytical problem, but they are differentiable in correspondence of maxima. Thus b and w are only continuous and the functional is not a regularizing one.

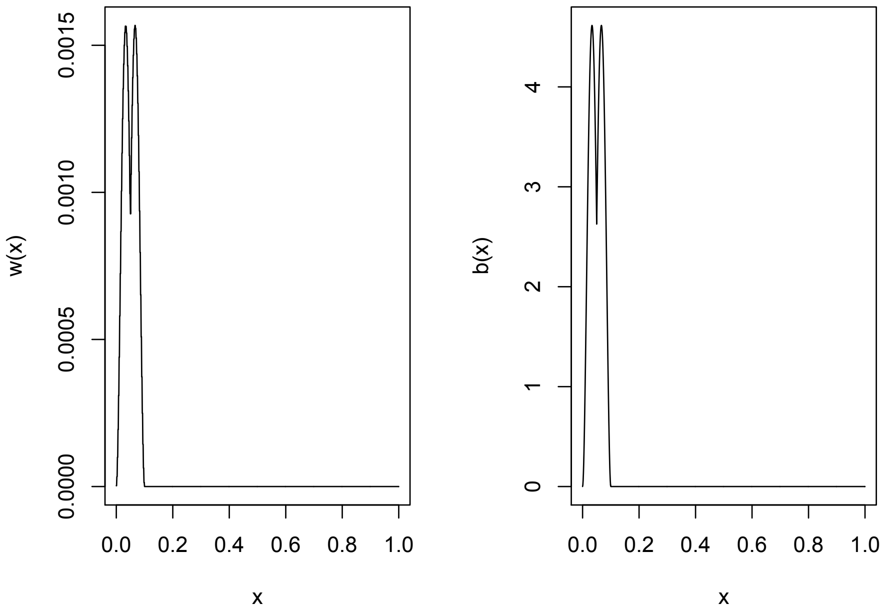

Figure 3.2: Comparison between $w(x)$ and $b(x)$ for Case 1.

3.2.2 Case 2

In this example we have considered a little more complicated kind of functions. Precisely, we keep considering the same prototypes of the previous case, but, now, we divide the domain $D = [0, 1]$ into four intervals of different measure. On two of these subintervals, the functions are equal to zero, but, on the remaining parts, the data have their own shape: piecewise affine linear or parabolas.
As in the previous example, the maxima of the functions are generated by independent gaussian random variables and then, every parameter m_r, q_r, a_s, b_s and c, with $r = 1, 2, 3, 4,\ \ s = 1, 2$, is calculated in order to have continuous functions. The maxima are located at points 1/20 and 5/8 and the intervals where the groups are different from zero are $[0, 1/10]$ and $[1/2, 3/4]$. Moreover, the maxima, are so distributed:

$$f_i\left(\frac{1}{20}\right) = Y_i \sim \mathcal{N}\left(2, \frac{1}{4}\right) \text{ and } f_i\left(\frac{5}{8}\right) = Z_i \sim \mathcal{N}\left(6, 1\right);$$

$$g_j\left(\frac{1}{20}\right) = U_j \sim \mathcal{N}\left(\frac{5}{2}, \frac{1}{4}\right) \text{ and } g_j\left(\frac{5}{8}\right) = V_j \sim \mathcal{N}\left(4, 4\right),$$

with $i, j = 1, \dots, 20$. Analytically:

- Group 1:

$$f_i(x) = \begin{cases} m_1(Y_i)\,x & \text{if } x \leq 1/20, \\ m_2(Y_i)x + q_1(Y_i) & \text{if } 1/20 < x \leq 1/10, \\ 0 & \text{if } 1/10 < x \leq 1/2, \\ m_3(Z_i)x + q_2(Z_i) & \text{if } 1/2 < x \leq 5/8, \\ m_4(Z_i)x + q_3(Z_i) & \text{if } 5/8 < x \leq 3/4, \\ 0 & \text{if } x > 3/4; \end{cases} \tag{3.3}$$

- Group 2:

$$g_j(x) = \begin{cases} a_1(U_j)x^2 + b_1(U_j)x & \text{if } x \leq 1/10, \\ 0 & \text{if } 1/10 < x \leq 1/2, \\ a_2(V_j)x^2 + b_2(V_j)x + c(V_j) & \text{if } 1/2 < x \leq 3/4, \\ 0 & \text{if } x > 3/4. \end{cases} \tag{3.4}$$

In figure 3.3 we can see a possible case for the second simulation. The two groups are more similar on the first subinterval than on the other where they are different from zero. Therefore we expect that the weight function will give more importance to the second subinterval. Moreover, in $[0, 1/10]$, the parabolas are a bit taller than the piecewise linear functions, while, in $[1/2, 3/4]$, apart from a rather unusual outlier, the contrary is true. That is why we also expect a different shape of the functions $b(x)$ and $w(x)$ on these two intervals. Specifically, in $[0, 1/10]$, their shape should be very similar to that of the previous example, but it should be something different on the other set.
In figure 3.4, we can see the optimal weight function $w(x)$ and $b(x)$. As we thought, the two curves are higher on the second interval and they have different shape on the two subsets. Note that on the first part there is the same symmetrical profile of the previous example, whereas, on the second set, it is different: it is monotonically increasing with a peak and then monotonically decreasing to zero. The reason for this is due to the fact that the first group is taller than the second on that set. Again, the points of local minimum and absolute maximum are not points of differentiability for b and w.

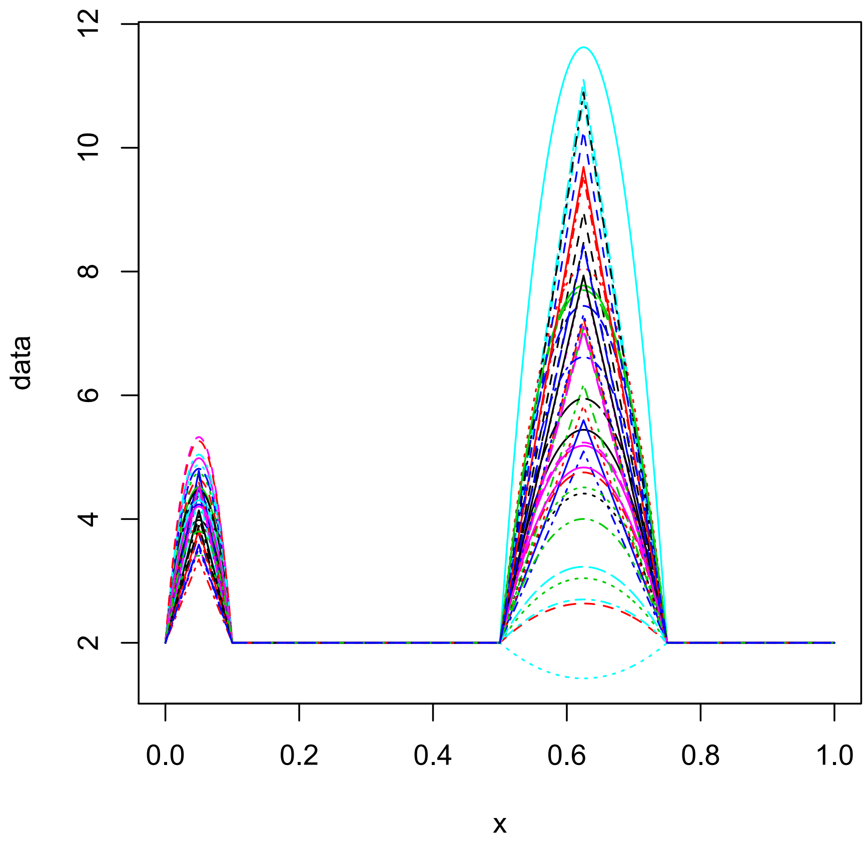

Figure 3.3: A simulation for Case 2.

Finally, we have reported two tables comparing the clusters obtained to the real groups. We can observe the improvement made by our method: six data are reallocated to the correct clusters. If we look at the misclassified data added to the first group, we find that one is the outlier parabola. One reason for this error could be that, as this parabola has a very high maximum, it is regarded as a line, because its curvature is less distinguishable from that of a line.

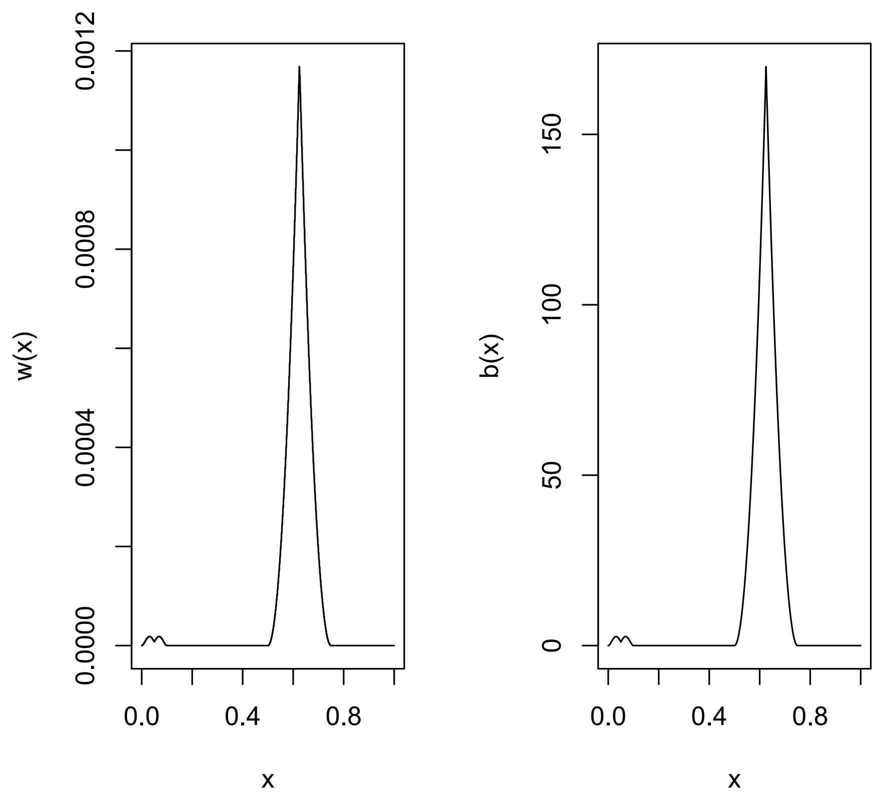

Figure 3.4: Comparison between $w(x)$ solution of the problem and $b(x)$ for Case 2.

3.3 Simulations With Noise

In the next cases we are going to consider more complicated data. Precisely, we have substituted the piecewise linear group with a completely non-linear kind of functions. Then, the other group, is composed by curves following those of the first group from one point to the end of the domain, but with the shape modified on the first part of the interval. Finally, we have added a gaussian noise in every point of the domain to the functions of the first group.
The explicit expressions of the prototype functions are, thus:

- Group 1:

$$f(x) = (2\sin(3\pi x) + 3)\left(-\frac{3}{2}x + 3\right) + S_1 + \varepsilon(x) \tag{3.5}$$

	Real Group 1 (lines)	Real Group 2 (parabolas)
K-means $\to$ 1	11	6
K-means $\to$ 2	9	14

Table 3.3: K-means vs Real clusters, Case 2.

	Real Group 1 (lines)	Real Group 2 (parabolas)
WK-means $\to$ 1	16	5
WK-means $\to$ 2	4	15

Table 3.4: Weighted K-means vs Real clusters, Case 2.

- Group 2:

$$g(x) = \begin{cases} a(Y)x^2 + b(Y)x & \text{if } x \leq x_0, \\ (2\sin(3\pi x) + 3)\left(-\frac{3}{2}x + 3\right) + S_2 & \text{if } x > x_0. \end{cases} \tag{3.6}$$

Here S_i's, $i = 1, 2$, are independent random variables, used to generate the data, thus giving us a particular kind of variability in the shift of the functions. $\varepsilon(x)$ is another independent random variable which is added in every point of the domain to the functions from the first group, thus resulting in a shape reflecting a Brownian motion. For the second group, the maximum of the parabola, Y is, again, an independent gaussian random variable. Therefore we have two kinds of variability for the curves in the two groups: one coming from the shifts among the curves in the same cluster, which is generated by the same random variable and the other, coming, for Group 1, from the noise ε and, for Group 2, from the variability in the maxima of parabolas. We do not require to have continuous functions for the second group, that is, it need not be true that

$$\lim_{x \to y^-} f(x) = \lim_{x \to y^+} f(x),$$

for every $y \in D$ and, moreover, that

$$\lim_{x \to x_0^-} g(x) = \lim_{x \to x_0^+} g(x).$$

Note, moreover, that the random variables S_i's have the same distribution for both groups, even if they are independent.
In this case we want to test the ability, of the algorithm, to extrapolate a trend from the noise and we wonder if it is able to recognize the same functional structure in the two groups, thus giving a low weight in the second part of the

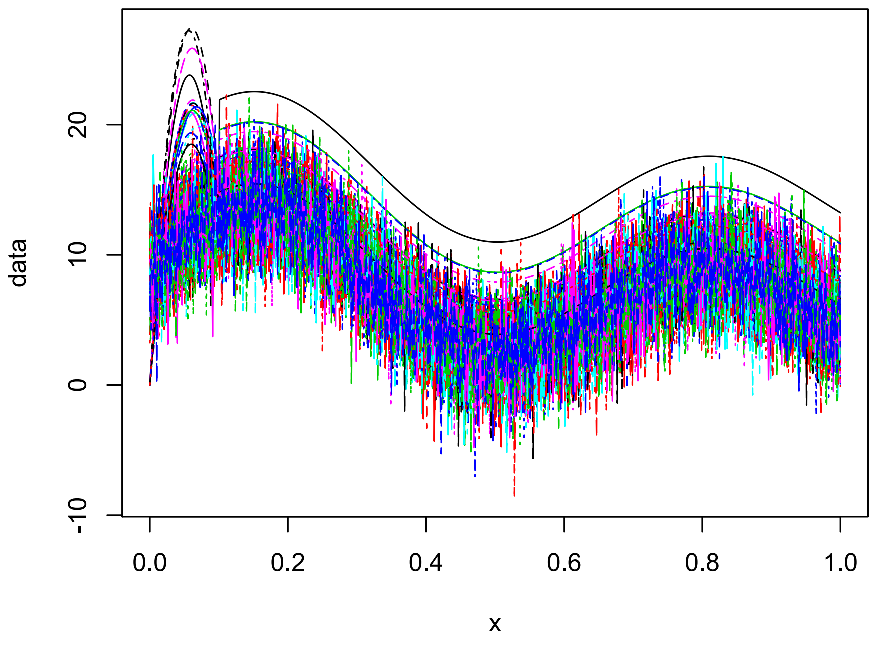

Figure 3.5: A simulation for the noisy case, 3.3.1.

domain. Obviously we expect that, in some sense, this ability will depend on the variables used to generate the data, but, we will see that our algorithm performs well with acceptable values of variability of those random variables.

3.3.1 A First Example

We will see that weighted K-means performs much better than classical K-means.

For these simulations, we have considered $S_i \sim \mathcal{N}\left(0, \frac{25}{4}\right)$, while, the maximum of the parabolas are sampled from independent gaussian random variabls with mean equal to 20 and variance equal to 9. In figure 3.5 the data for the considered case are plotted. The tuning parameter chosen is $\frac{1}{10}$, which is the length of the interval on which the two groups are different. The abscissa of the maximum of the parabolas is taken to be at $x = 1/20$, while, the point from which the two groups have the same trend is $x = 1/10$.

In the two tables 3.5 and 3.6 we report the comparison between the two

	Real Group 1 (noisy)	Real Group 2 (par. + sine)
K-means → 1	14	13
K-means → 2	6	7

Table 3.5: K-means vs Real clusters, 3.3.1.

algorithms used. We can evidently see that classical K-means can not tell well between the two groups, assigning most functions to the first cluster. This strange assignment suggests us to find, using some statistics, the optimum number of clusters for normal K-means. Therefore we try, using the following quantity:

$$S(k) = \frac{\frac{BCSS(k)}{k-1}}{\frac{WCSS}{N-k}}, \; k = 2, \ldots, N-1 \tag{3.7}$$

and then look for the value k^* which maximize $S(k)$. In figure 3.6, we can

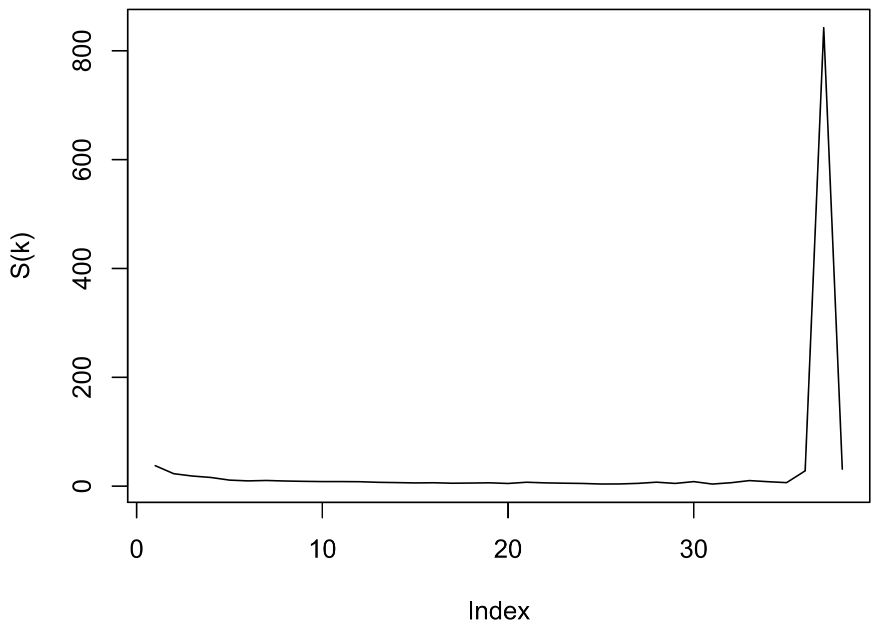

Figure 3.6: The graph of $S(k)$ using normal K-means as clustering method (the index starts from 2), example 3.3.1.

see the peak at the right end of the plot. In this case, the value k^* that maximizes $S(k)$ is found to be 36. Considering that we have 40 curves, this means that classical K-means is not able to recognize the same pattern lying behind the two groups of functions.

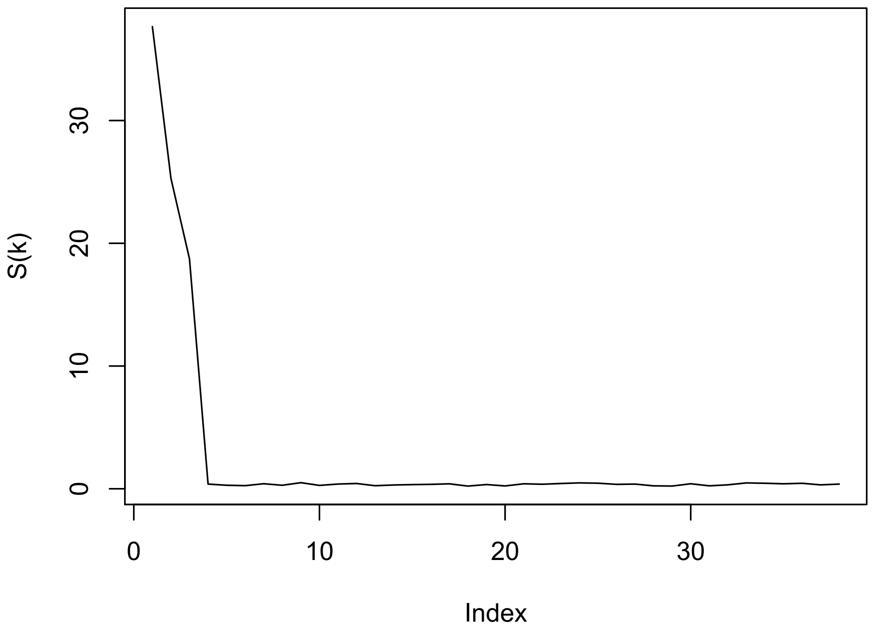

Figure 3.7: The graph of $S(k)$ using weighted K-means as clustering method (the index starts from 2), example 3.3.1.

Instead, with weighted K-means, things go better. All the functions from the first group are correctly classified, but there are four curves from the second cluster which are assigned to the wrong set of data. These misclassified functions are the most similar to those of the first group, in the sense that they have a lower maximum and are completely included in the noisy band. That is the probable reason for the committed errors. Note that it seems the algorithm rightly takes into account the fact that the second group starts from $(0, 0)$. If we compute the same statistic $S(k)$ for weighted K-means, we find that the maximum is assumed in correspondence of $k^* = 2$ and in figure 3.7 there is the plot of that quantity.

In figure 3.8 we can observe the very irregular shape found for the func-

	Real Group 1 (noisy)	Real Group 2 (par. + sine)
WK-means → 1	20	4
WK-means → 2	0	16

Table 3.6: Weighted K-means vs Real clusters, 3.3.1.

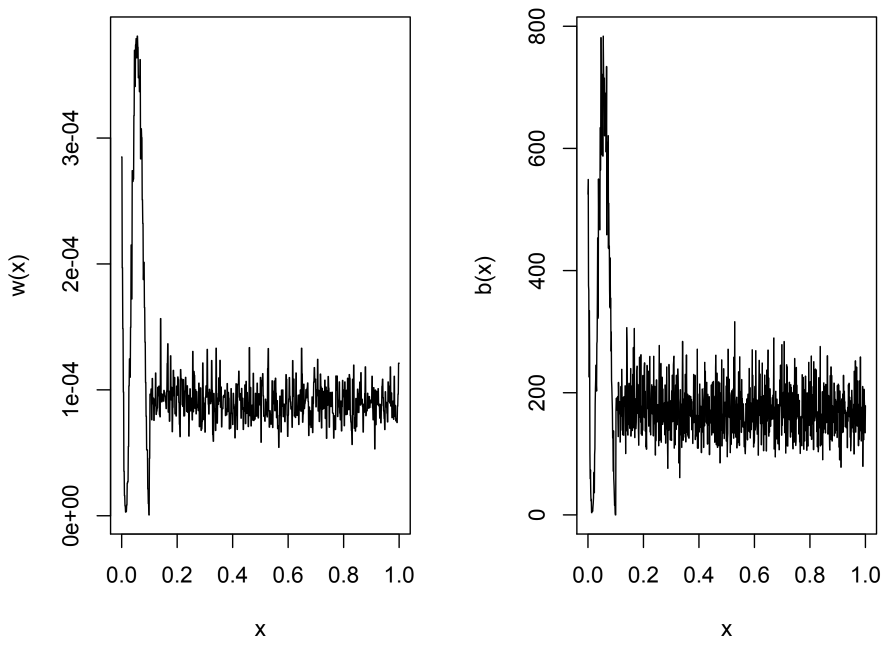

Figure 3.8: The very irregular shape of w and b, example 3.3.1.

tions $b(x)$ and $w(x)$. First of all we have to say that, since we are also considering functions which are not necessarly continuous, the result found in Theorem 2.2 does not hold. However it still continues to be valid in subsets of continuity and, moreover, we have a.e. convergence of the two considered functions. We can readily notice the importance given, by the weight function, to the first part of the interval. There is a first local maximum in correspondence of $x = 0$, since that point is important in the distinction, and an absolute maximum in correspondence to the maximum of the parabolas. It also seems to have two minima, but, the first, at $x = 0.014$ is only a local minimum, whereas, the second, located at $x = 1/10$ is an absolute minimum. The local minimum is the mean of the abscissas of the point at which the conjunction between functions of different clusters happens. The absolute

minimum, instead, is the point at which the curves from the second cluster change their shape: from parabolas, they become sines. To explain why this point is an absolute minimum, we have to recall that the functions of the second group are not necessarily continuous, thus, in that point, they are very similar to the noisy functions of the first group.
Now, we can focus on the remaining part of the interval. There, $w(x)$ still shows an irregular shape, but we can guess it is a nearly flat trend, i.e., apart from some noise, in that set the distance between groups is nearly constant. In the ideal case, if the algorithm had classified correctly all data, we would have expected that "plateau" close to zero. Instead it is not so. However, this is justifiable if we look at those four curves misclassified. That erroneous assignment makes the distance nearly constant, but at a value different from zero. Anyway, we can still collect some useful informations for the classification: $w(x)$ is not zero, but has a very low value and the flat trend suggests us that, in that subset, there are no evident characteristics distinguishing the two groups. It is found, and it is obvious that it should be so, that these fluctuations around the plateau increase with the increasing of the variance of $\varepsilon(x)$. Finally, the obtained functions $w(x)$ and $b(x)$ are, evidently, still in L^2.
In the last figure are plotted the cluster mean functions for the two groups found using weighted K-means.

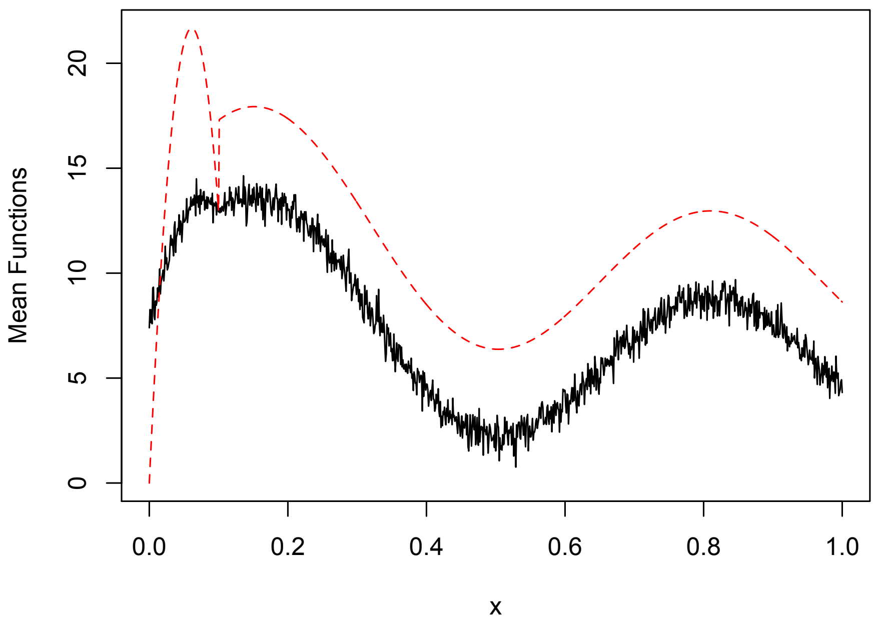

Figure 3.9: Mean functions for the clusters in example 3.3.1.

3.3.2 A Noisier Example

In the following case, we are going to complicate further the situation, by introducing a new source of noise even in the second group. Actually, we have added some noise, to every point of the first subinterval, to the functions of the second group. Therefore, in formulas, the analytical expressions of the curves generating the two groups become:

- Group 1:

$$f(x) = (2\sin(3\pi x) + 3)\left(-\frac{3}{2}x + 3\right) + \varepsilon(x) \tag{3.8}$$

- Group 2:

$$g(x) = \begin{cases} a(Y)x^2 + b(Y)x + \xi(x) & \text{if } x \le x_0, \\ (2\sin(3\pi x) + 3)\left(-\frac{3}{2}x + 3\right) + S & \text{if } x > x_0. \end{cases} \tag{3.9}$$

The variable S has the same distribution of the previous example: $S \sim \mathcal{N}\left(0, \frac{25}{4}\right)$. The variables $\varepsilon(x)$ and $\xi(x)$ are again independent gaussian random variables, but they have different parameters, precisely: $\varepsilon(x) \sim \mathcal{N}\left(0, \frac{25}{4}\right)$ and $\xi(x) \sim \mathcal{N}(0, 1)$. Finally, the maximum of the parabolas, Y, is sampled, idependently from all the other random variables, from a gaussian random variable with mean equal to 20 and variance equal to 4. Even in this example the functions of the second group are not supposed to be necessarly continuous. The tuning parameter, the location of the maximum of the parabolas and the point where the shape of the curves from the second group changes are the same of the previous example. In figure 3.10 are plotted the functions considered for this example. It is very difficult telling the two groups, even if one could note, in the first part of the interval, a different shape for some functions.

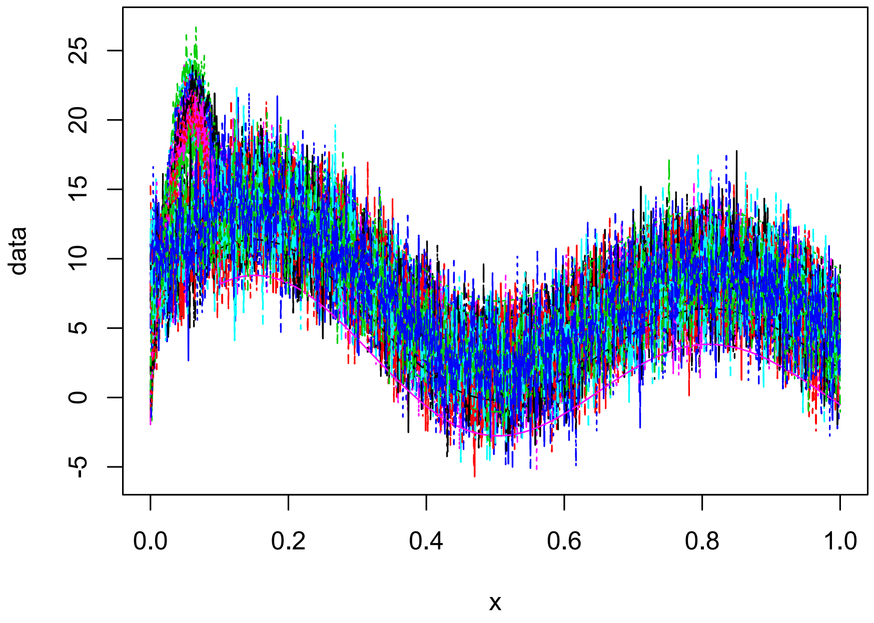

Figure 3.10: Data for example 3.3.2.

	Real Group 1	Real Group 2
K-means $\rightarrow$ 1	15	15
K-means $\rightarrow$ 2	5	5

Table 3.7: K-means vs Real clusters, example 3.3.2.

	Real Group 1	Real Group 2
WK-means $\rightarrow$ 1	11	9
WK-means $\rightarrow$ 2	9	11

Table 3.8: Weighted K-means vs Real clusters, example 3.3.2.

In tables 3.7 and 3.8, respectively, we report the cluster obtained using classical and weighted K-means. The classical method assigns the 75% of the data to the first group. Following the same reasoning of the previous case, we plot the statistic $S(k)$ and seek for the k^* maximizing it. In the same way, we find hard interpretable results, with $k^* = 35$. In figure 3.11, we have

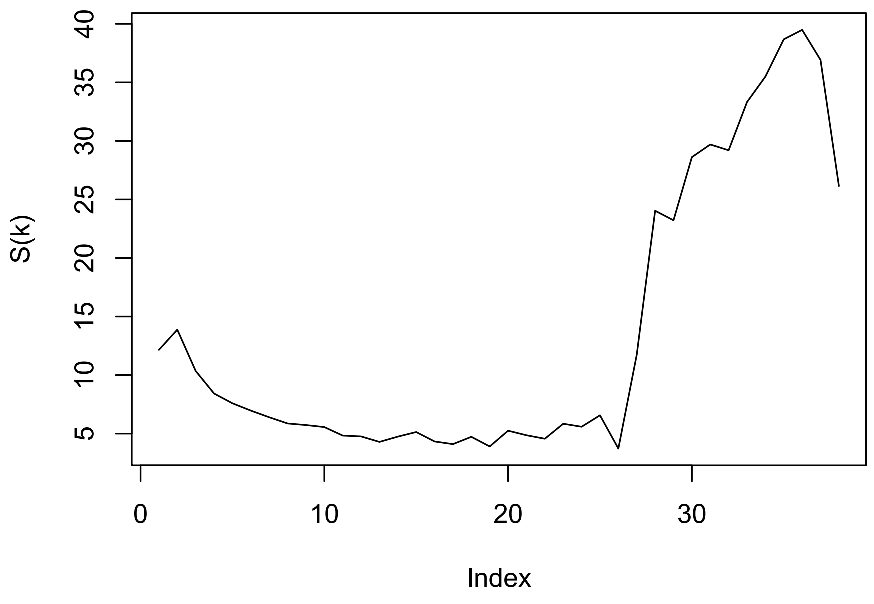

Figure 3.11: The statistics $S(k)$ computed with classical K-means as clustering method (the index starts from 2), example 3.3.2.

reported the quantity $S(k)$ for every value of k. The high value for k^* means that, again, the classical method is not able to recognize the same patterns for the groups and in figure 3.12 are riported the mean functions obtained with normal K-means and two clusters. We can note that, probably, classical K-mean distinguishes between the two clusters only by a criterion based on height: the highest functions constitute a group and the remaining form the other.

Looking at the results found with weighted K-means, it seems that it pro-

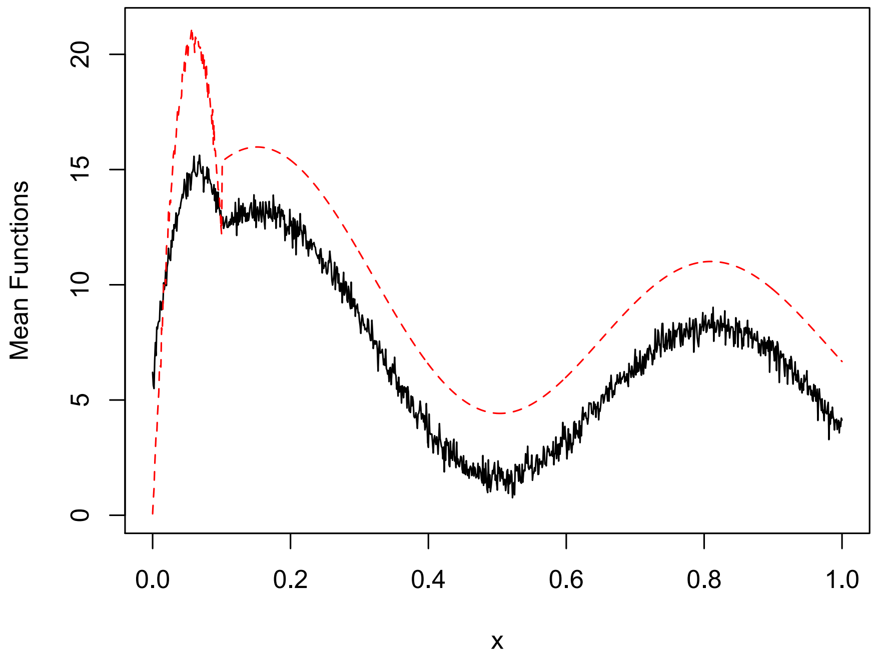

Figure 3.12: Mean functions found with normal K-means for the data in example 3.3.2.

vides an improvement but not so evident. However, the misclassifications are understood because of the lack of the variable S in the first group, which confounds the situation even more. If we consider the figure 3.14, we immediately understand the reason for so many misclassified data: the trend in the second part of the interval are perfectly overlapping, thus it is very probable the method confounds the groups, if they are very similar even in the first part of the domain. Note the difference with respect to the mean functions found by classical K-means. In figure 3.13 the statistic $S(k)$ is plotted for weighted K-means and k^* is correctly found at 2.

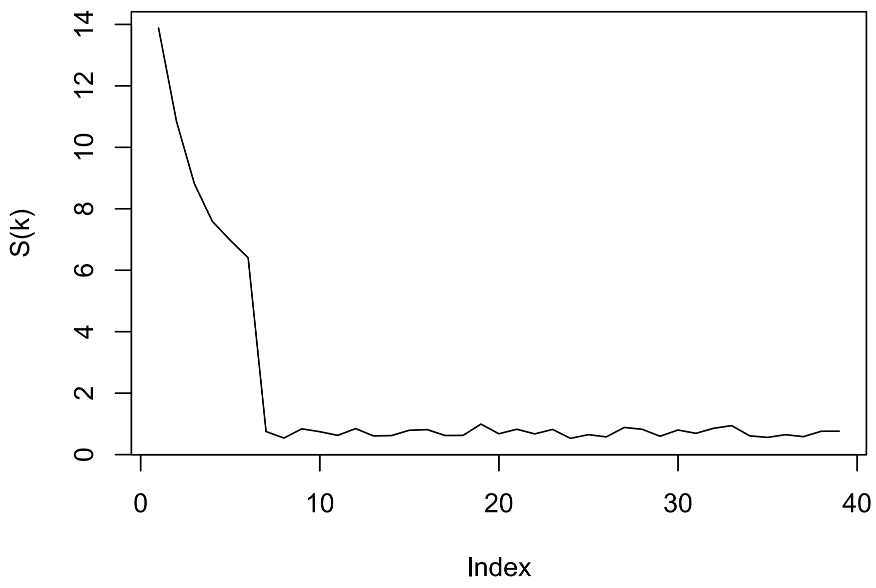

Figure 3.13: The statistics $S(k)$ computed with weighted K-means as clustering method (the index starts from 2), example 3.3.2.

In figure 3.15, we can see the plots of $w(x)$ and $b(x)$ found by the algorithm. The shape in the first part of the domain is very similar to that of the previous example. In this case, however, it must be a bit more noisy, because of the random variable $\xi(x)$ added to the parabolas. For this first set, we can notice the same characterisics as in the case without noise added to the second group: $w(x)$ starts with a local maximum in zero, has two, now both local, minima, situated, respectively at $x = 0.019$ and at $x = 0.113$ and there is an absolute maximum at $x = 0.055$. Now the absolute minimum has moved to the flat part, and, precisely, at $x = 0.968$. Even if the clusters do not reflect the real groups, the plateau, this time, if found around zero. This is absolutely expected because of the same trend characterizing the groups, therefore, the distance between the clusters is almost zero.

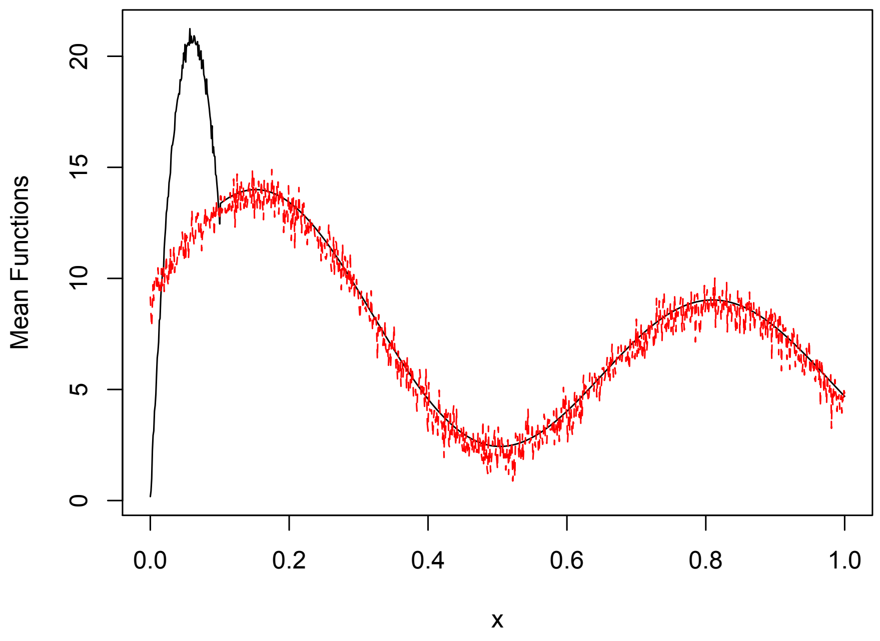

Figure 3.14: Mean functions found with weighted K-means for the data in example 3.3.2.

3.3.3 The Last Case

Now, as the last simulation, we are going to consider the most complicated case of all. To this purpose, we have added some noise to every group and in every point of the domain and then we have tried to cluster these data. The analytical expressions of the prototype functions are the same of the two previous examples, but we have to consider, also, the new sources of noise.

- Group 1:

$$f(x) = (2\sin(3\pi x) + 3)\left(-\frac{3}{2}x + 3\right) + S + \varepsilon(x) \tag{3.10}$$

- Group 2:

$$g(x) = \begin{cases} a(Y)x^2 + b(Y)x + \xi(x) & \text{if } x \le x_0, \\ (2\sin(3\pi x) + 3)\left(-\frac{3}{2}x + 3\right) + T + \xi(x) & \text{if } x > x_0. \end{cases} \tag{3.11}$$

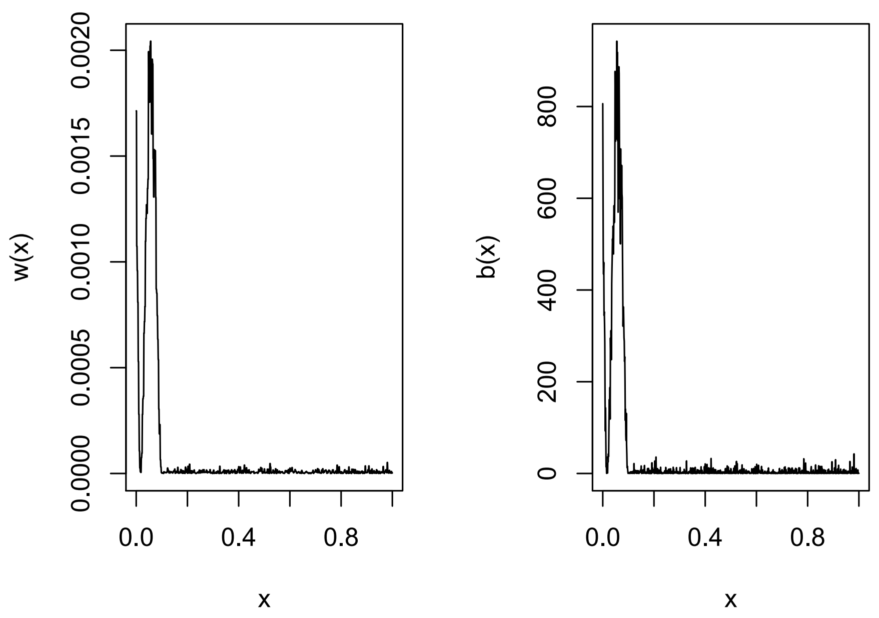

Figure 3.15: The functions w and b, for example 3.3.2.

Here, $S, T, \varepsilon(x)$, $\xi(x)$ and Y are independent gaussian random variables. S an T are responsibles for the shift between functions of the same group, whereas $\varepsilon(x)$ and $\xi(x)$ are responsibles for the noise added in every point of the domain. To the functions of the second group, we have used random variables with the same distributions. For the example specifically reported we have considered the following distributions:

- $S \sim \mathcal{N}(2,4)\,,\ \varepsilon(x) \sim \mathcal{N}\left(0, \frac{25}{4}\right);$
- $T \sim \mathcal{N}\left(0, \frac{25}{4}\right),\ \xi(x) \sim \mathcal{N}(0,1)\,.$

The maximum of parabolas, Y, comes from a $\mathcal{N}(20,4)$. So we have different kind of variability in each group: one coming from the shift and the other one coming from the punctual noise. Moreover, in the second group we have also an amplitude variability for the maximum of the parabolas. In figure 3.16, we can see the data generated in the considered example. It is almost impossible distinguishing the two groups, apart, at most, for one or two parabolas. We expect that the two algorithms do not perform well, because of these confused data. Moreover we still have the curves of the groups not

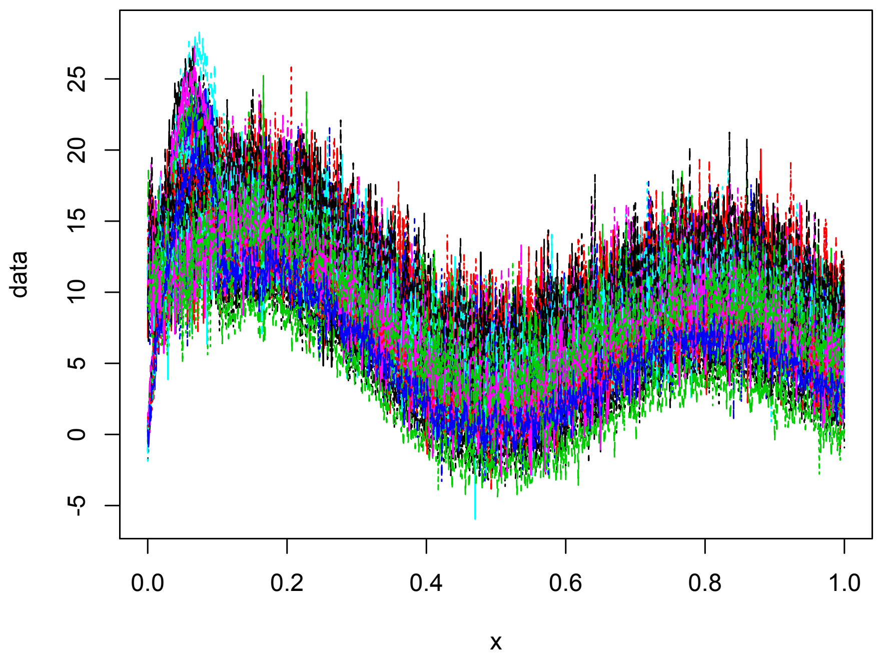

Figure 3.16: Data for example 3.3.3.

	Real Group 1	Real Group 2
K-means $\rightarrow$ 1	4	7
K-means $\rightarrow$ 2	16	13

Table 3.9: K-means vs Real clusters, example 3.3.3.

continuous.

In tables 3.9 and 3.10 we can compare the clusterization obtained with classical and weighted K-means. The classical algorithm does not return an efficient result, assigning about the 75% of data to one group and the remaining functions to the other. Moreover, if we look at the misclassified curves, it is not well understood the criterion used to distinguish between the groups. Then we repeat the criterion of maximizing the statistic $S(k)$, to find the optimal number of clusters with classical algorithm. Unexpectedly, we find that it is correctly maximized for $k = 2$, as we can see in figure 3.17.

	Real Group 1	Real Group 2
WK-means $\to$ 1	13	7
WK-means $\to$ 2	7	13

Table 3.10: Weighted K-means vs Real clusters, example 3.3.3.

However, we can also note that the graph is very irregular, meaning that, in this case, it is not a very reliable result. In figure 3.18, we have plotted the

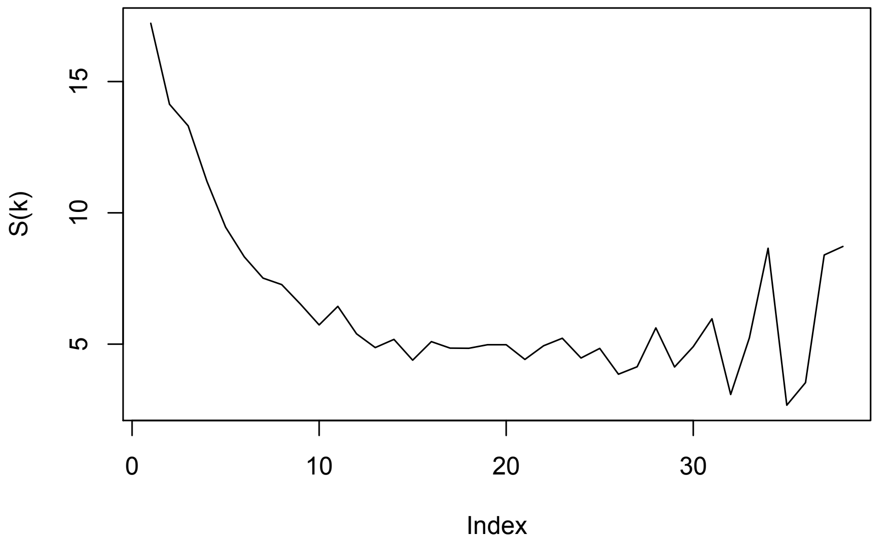

Figure 3.17: The statistic $S(k)$ using classical K-means as clustering procedure in example 3.3.3.

mean functions obtained with classical K-means. Note that, in the first part of the domain, each group shows a maximum. This means that the algorithm has not recognized that only one group has a maximum in the first part of the interval. Thus it mixes the functions coming from both groups. Even the mean function with the highest maximum has an irregular shape in the descending part just following the maximum itself. Thus we have to conclude that classical K-means is not very useful in this case.

Now we focus on the result found with weighted K-means. It correctly assigns thirteen functions to the right clusters, but it still misclassifies seven

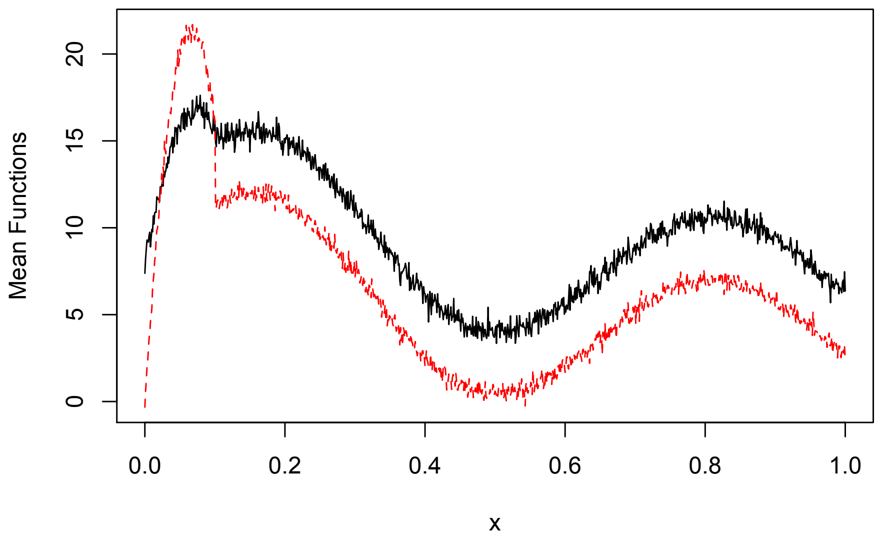

Figure 3.18: The Mean functions found using classical K-means as clustering procedure in example 3.3.3.

curves in each group. However, it is a better result with respect to classical K-means, because the correctly classified curves really come from that specifically group. In figure 3.19, we have reported the graph for the statistic $S(k)$ computed using the new method for the classification. The maximum is again correctly assumed in $k = 2$ and the shape of $S(k)$ is more regular than the previous one, suggesting that the result is more robust. To confirm the fact that the found result is fairer than that found with classical K-means, we plot the mean functions in figure 3.20. We can note some differences between the mean functions found in the two cases. The most interesting fact is the presence of the maximum in only one group and also the shape of the parabola is more regular. Moreover the distance between the groups, in the part with common shape, is smaller with weighted K-means than with classical K-means, which is right, looking at how we have constructed the curves in the simulation.

Finally, we have shown, in figure 3.21 the graph of $w(x)$ solution of the problem and that of $b(x)$ found with the weighted algorithm. They both rightly starts with a maximum, which has now become an absolute maximum, whereas, what was, in the previous two examples an absolute maximum, has now become a local maximum. This happended because, now, the

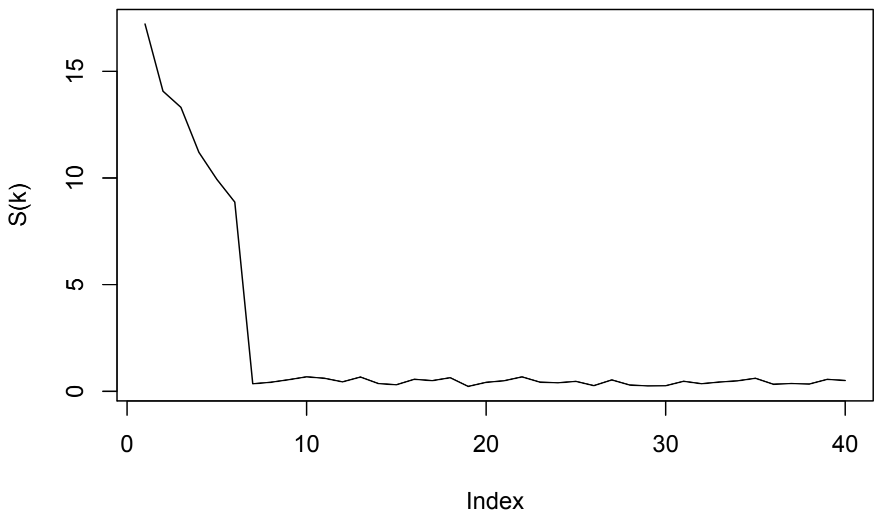

Figure 3.19: The statistic $S(k)$ using weighted K-means as clustering procedure in example 3.3.3.

point which distinguishes the two groups the most is $x = 0$. The added noise, instead, smoothes the distances between the parabolas and the other curves in correspondence of the maxima. The first minimum, now, is the absolute one, whereas the second is a local minimum. The first is found at $x = 0.023$ and the second at $x = 0.1$. Then we have the average flat trend. The fact the plateau is not at zero is due to the misclassified functions, but it still communicates the fact that those points are not so useful for the clustering procedure. It seems that the shape of $w(x)$ is not just a rescalement of that of $b(x)$. This was quite expected because the functions are discontinuous in every point of the domain. However $w(x)$ is still useful to see what are the most distiguishing sets or points between the groups.

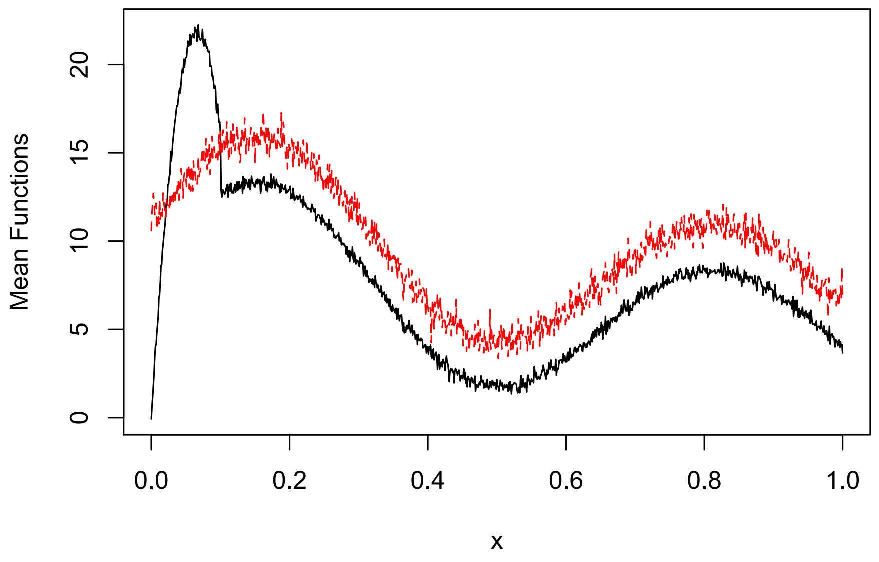

Figure 3.20: The Mean functions found using classical K-means as clustering procedure, example 3.3.3.

3.4 Final Comments

At the end of these comparisons, we can say that, at least for the cases here considered, weighted K-means performs better than classical K-means and the improvements are as more evident as the data are noisier and the algorithm has to identify an underlying shape defining the different groups. However, it still shows some drawbacks affecting every K-means procedure. The most serious one is represented by the fact that it is a method depending only on distances. Therefore, data which are close, in the norm used, to functions of a determined group are assigned to it, even if they belong to another cluster. For example, if we consider a datum, which is an outlier, it will surely be assigned to a wrong cluster, even if it is an outlier only on a subset of the domain, but if that set is such that it makes the distance between that datum and a wrong group smaller than the distance between the same datum and the real group to which it should belong, then we will have a wrong assignment.
The first way we can propose to overcome this problem is to make K-means methods to take into account the probability model underlying the generation of the data oserved. Thus, we have to consider a probability measure $\mathbb{P}$

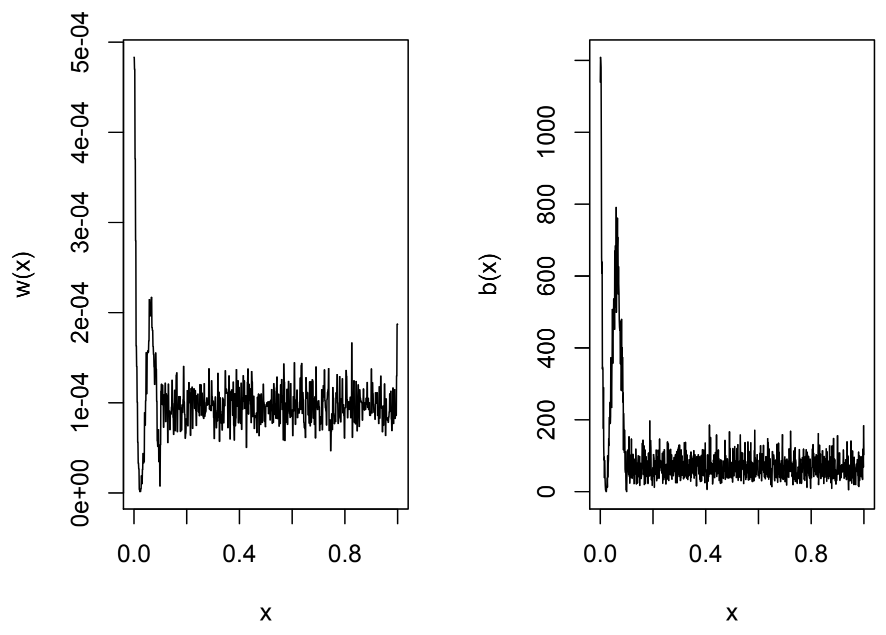

Figure 3.21: The graphs of $w(x)$ and $b(x)$, example 3.3.3.

which depends on the functional variables generating the data and also on the points of the domain, so, we write $\mathbb{P}(\mathcal{X}_k(\omega, x), x)$. Therefore, setting $\mathbf{C} = (C_1, \ldots, C_K)$, we can rewrite our problem in the following way:

$$\max_{w(x),(C_1,\ldots,C_K)} \int_D w(x) b(x, f_{i,k}, \mathbf{C}) d\mathbb{P}(\mathcal{X}_k(\omega, x), x), \tag{3.12}$$

with $i = 1, \ldots, N$, $k = 1, \ldots, K$, $w(x)$ satisfying the usual constraints and where we have explicited the dependence of b on the domain, the data $f_{i,k}$'s and the clusters C_k's. However, there is still the complication due to the knowledge of the distribution of the functional variables acting in the problem.

We can try to investigate another direction to improve the solution to the clustering problem. We are dealing with curves, therefore, if it is true that the functions come from different distributions, they will be different not only in their own shape, but also in that of their derivatives, if they exist. So, we can try to cluster simultaneously the functions and their derivatives. We can, then, define the following functional:

$$\int_D \Big\langle \mathbf{w}(x), \mathbf{b}(x, f_{i,k}, f'_{i,k}, \mathbf{C}) \Big\rangle \, d\mu, \tag{3.13}$$

where, $\mathbf{w}(x)$ is a vectorial weight function and $\mathbf{b}(x, f_{i,k}, f'_{i,k}, \mathbf{C})$ is the BCSS, but now considering also the derivatives of the data, i.e. the first component of this vector is the usual BCSS, while, the second component, is the BCSS computed with the derivatives f'_i's. Now, we could try to maximize it over $\mathbf{w}(x)$ and $\mathbf{C}$. We would like to have $\mathbf{w}(x) = (w_1(x), w_2(x))$, with, eventually, the relationship $w_2(x) = w'_1(x)$. The function $w_1(x)$ has the same role $w(x)$ had previously and $w_2(x)$ is another function used to weight the BCSS computed with the derivatives of the data. Certainly, we have that $w_2(x)$ has to satisfy some constraints as well: $\|w_2(x)\|_{L^1} \leq t$, $\|w_2(x)\|^2_{L^2} \leq 1$ and, finally, $w_2(x) \geq 0$ μ-a.e. Here, t, is another tuning parameter used to measure the sparsity, but, now, referring to derivatives. However, the nature of t and the connections with s should be investigated, because it is not necessary that the sparsity structure on the derivatives coincides with that of the functions. If we want to consider this new problem, we have to modify the theorical setting in some ways. First of all, we should have functions belonging to the Sobolev space $H^1(D)$. However, some questions arise quite naturally: how could we find the optimum? Is it a global optimum? Moreover: what are the real connections between $w_1(x)$ and $w_2(x)$? Could it be possible that $w_2(x)$ is the (weak) derivative of $w_1(x)$? The answer to this last question is that, in general, $w_2(x)$ can not be the (weak) derivative of $w_1(x)$. The proof of Theorem 2.1 was based, heavily, on the coincidence of the weak and the weak* topology in L^2, fact which does not hold in H^1 and, moreover, we do not even know what is the dual space of H^1.

Chapter 4

Analysis of Growth Curves

4.1 Introduction

In this chapter we are going to analyze deeply the functional dataset "Growth", already provided in the `fda` package. This dataset is quite famous in the field of functional statistics. It is what is called a "benchmark dataset", that is a set of data to which apply new methods in order to test them, as the analyses provided can be readily compared with previous results. For instance it has been already studied deeply by Ramsay and Silverman in [16]. The dataset is composed by the heights, in cm, of 93 children, of which 54 girls and 39 boys, measured quarterly from 1 to 2 years, annually from 2 to 8 years and then biannually from 8 to 18 years. The data, thus, are discrete, but it is reasonable to think them as sample points, in time, of a determined curve representing the particular height of a child. Therefore, before proceeding in any analysis, we have, firstly, to smooth them. This is done through standard techniques, requiring the definition of particular basis functions. After having obtained regular (i.e. twice differentiable) curves, it is reasonable that children of the same sex will have similar functions describing their own height and so we will try to make some cluster analysis on them. Thanks to studies conducted by physicians, we know how human growth develops and we want to use the method just defined to see if we are able to find the same results.
We have two kind of curves on which we can conduct our analyses: those coming directly from the smoothing, called *misaligned* or *unregistered* and then the same functions but considered after a procedure called *alignment* or *registration*. This process consists in finding some nonlinear trasformations of the temporal variable such that the fictitious (in some sense) variability between data is deleted.

Our study starts with the analysis of misaligned data and then, after a brief presentation of the registration techniques, we will move on to study the aligned curves.

4.2 The Growth Dataset

In this section we describe, in details, the characteristics of the dataset. The grid over which the measurements of the heights are taken, is not equally spaced in time, since they are more frequent in the early years of children's life and then become more and more distant. What we are really observing,

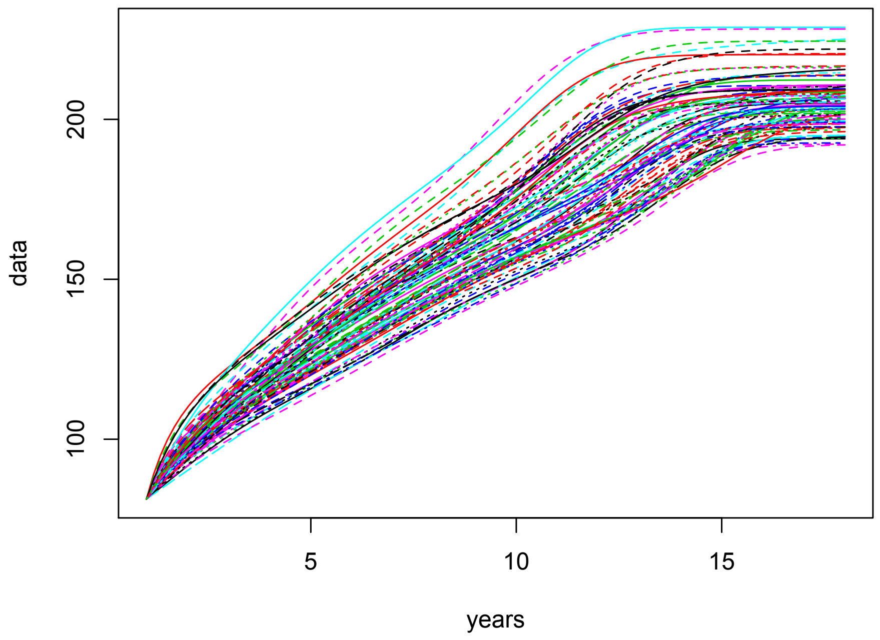

Figure 4.1: The estimated functions describing the developments of the heights of the 93 children.

when we measure the heights of the children, are the consequences of growth. The "derivatives" of the height data represents the velocities of growth and what should correctly called "growth", as they are the change in height per unit time. Indicating with $H(t)$ the "height function" and with $V(t)$ its "derivative", we could estimate, at time points t_i, the velocity by the difference ratio

$$V(t_i) = \frac{H(t_{i+1}) - H(t_i)}{t_{i+1} - t_i}. \tag{4.1}$$

This, anyway, is a bad idea from a statistical perspective, since even a small amount of noise in the height measurements will have a huge effect on the ratio, and this problem only gets worse as the time points get closer together. It is much better to fit the height data with an appropriate smooth curve, and then estimate velocity by finding the slope of this smooth curve. To smooth these data, monotonic cubic regression splines were implemented, using the `R` function `smooth.monotone`. Spline functions are the most common choice of approximation system for non-periodic functional data or parameters. In figure 4.1, the reconstructed height curves are plotted. They are all quite similar: they are all monotone, start from 80 cm and reach a flat shape at about 15 years, meaning that, there is a particular age from which the height is almost constant. In particular, the monotonicity of the heights implies that the derivatives are all positive, starting from high values and then decreasing to zero.
We can get more understanding of the growth process by studying the rate of change in velocity; this is the acceleration in height, denoted by $A(t)$. From the previous observation about the shape of the velocities, we can readily state that accelerations will be mostly negative. In figures 4.2 and 4.3 the reconstructed velocities and accelerations are shown. We can observe and confirm the predictions made about their shape. In both cases, the curves have similar trends, with some differences in the points at which minima or maxima are assumed. This fact will be eliminated successively with the procedure of alignment.
A way to look at growth could be represented by the following expression:

$$V(t_{i+1}) - V(t_i) = w_i V(t_i)(t_{i+1} - t_i). \tag{4.2}$$

It relates the velocity change over the interval $[t_i, t_{i+1}]$ to three factors:

1. $t_{i+1} - t_i$ itself. The smaller this time interval, the less change there will be, and in the limit $\Delta t \to 0$, velocity will not change. This says that over very small time scales growth is essentially a smooth process, an assertion that seems beyond question since a jump in the rate of growth over an arbitrarily small time interval would seem inconceivable in terms of the body's physiology;

2. $V(t_i)$, a term that measures growth changes on a percentage or relative basis. This is particularly useful in allowing for variations in height over the population, and, for instance, allows for comparison of growth patterns independently of people's ultimate adult height;

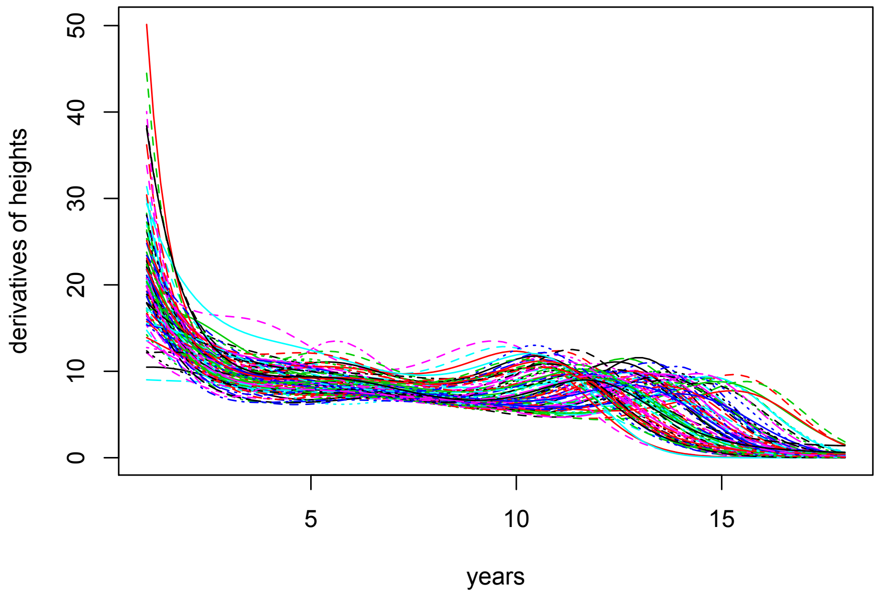

Figure 4.2: The estimated velocities of the heights of the 93 children.

3. w_i, a factor that determines the change in velocity. We make this factor depend on t_i because we imagine that this factor itself will change with time. This is the factor that really specifies how growth varies. Note that w_i will be positive if velocity is increasing at age t_i, zero if there is no change, and negative if velocity is decreasing.

Equation 4.2 could be rearranged as:

$$\frac{V(t_{i+1}) - V(t_i)}{t_{i+1} - t_i} = w_i V(t_i). \tag{4.3}$$

The left side of this equation is just an estimate of the instantaneous rate of change of $V(t)$, and becomes the acceleration $A(t)$ when $t_{i+1} - t_i \to 0$. Therefore, rather than defining w_i to satisfy (4.2) and (4.3) exactly, we replace it by a function $w(t)$ defined by:

$$A(t) = w(t)V(t), \text{ or } w(t) = \frac{A(t)}{V(t)}. \tag{4.4}$$

The continuously defined function $w(t)$ is now the ratio of acceleration to velocity, or what we can call *relative acceleration*, meaning acceleration of

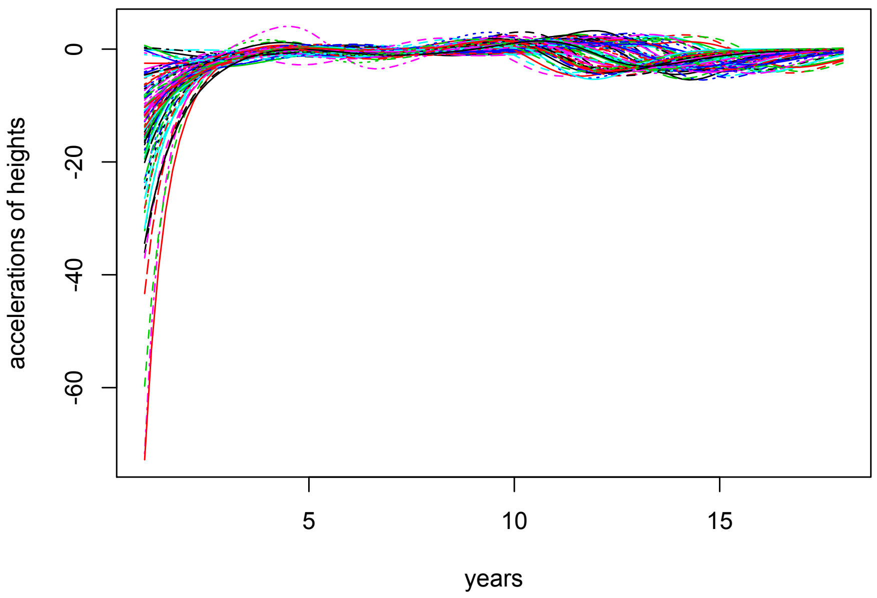

Figure 4.3: The estimated accelerations of the heights of the 93 children.

height measured as a fraction of velocity. We can rewrite (4.4) as the differential equation:

$$\frac{d^2H(t)}{dt^2} = w(t)\frac{dH(t)}{dt}. \tag{4.5}$$

The general solution to this equation is given by:

$$H(t) = C_0 + C_1 \int_0^t e^{\int_0^u w(v)dv} du. \tag{4.6}$$

In this expression, C_0 and C_1 are arbitrary constants that will need to be estimated from data.
This model was proposed by Ramsay and Silverman (2002). Equation (4.5) may be described as the fundamental equation of growth, in the sense that any intrinsically smooth growth process may be expressed in this way. The relative acceleration $w(t)$ is the functional parameter of growth. A possible approach to think about growth is to model this function, rather than the height function itself. Once it has been estimated $w(t)$, one can check it against the data by using equation (4.6). However, neither the authors, nor us, will go deep through this argument.

4.3 Analysis On Misaligned Data

The first step of our analysis, begins with the clusterization on misaligned data, that is, with nonmodified reconstructed curves. We are supposing that the growth process distinguishes between girls and boys and we want to test our algorithm on this dataset. We want to see if it is able to detect the differences in heights of boys and girls and find the age at which these differences are more significant.
The most important problem, before running the algorithm, consists in finding the right tuning parameter, defining sparsity. We have no outer informations helping us: it is not evident which interval of time distinguishes the clusters the most and, moreover, even medical sciences can not provide useful notions, apart from the so called *pubertal spurt*, which is, unfortunately, not easy to determine, as it happens at different ages to different children and it could have different duration. Therefore, we need other quantitative methods to estimate the right s. To do so, we define an equally spaced grid of 100 points from 0 to 18, then we evaluate the curves on this grid and collect the values in a matrix belonging to $\mathbb{R}^{93\times100}$. Now we apply a reasoning similar to that of the Gap Statistic discussed in chapter 1, but modified, in order to take into account the functional nature of the original data and the right connections between the norms of functions involved. This procedure provides $s^* = 4.123106$, that is a value very close to the maximum allowable value of $\sqrt{18} \approx 4.242641$. This proximity between the two values could suggest that there is not a well defined set where the clusters are well separated, as we had already observed looking at figure 4.1. As a consequence, the weight function will be interpreted as the difference, in every point of the domain, between the two groups, meaning that eventual peaks or valleys indicate where the two clusters differ, respectively, the most or the least.
After having found the optimal s, we run our algorithm. We find that it recognizes correctly the two clusters: one, of cardinality 54, composed by the heights of the girls and the other, of cardinality 39, composed by those of the boys. In figure 4.4, we have reported the calculated mean functions found for the corrispectively clusters. As we could have expected, the mean curve of the boys is higher than that of the girls. Even if it is not so easy, looking carefully at the figure, a difference in the shape could be observed. Indeed, there is a small interval, centered just after 10 years where the mean function of the girls has a negative curvature, whereas the mean function of the boy has a positive one.

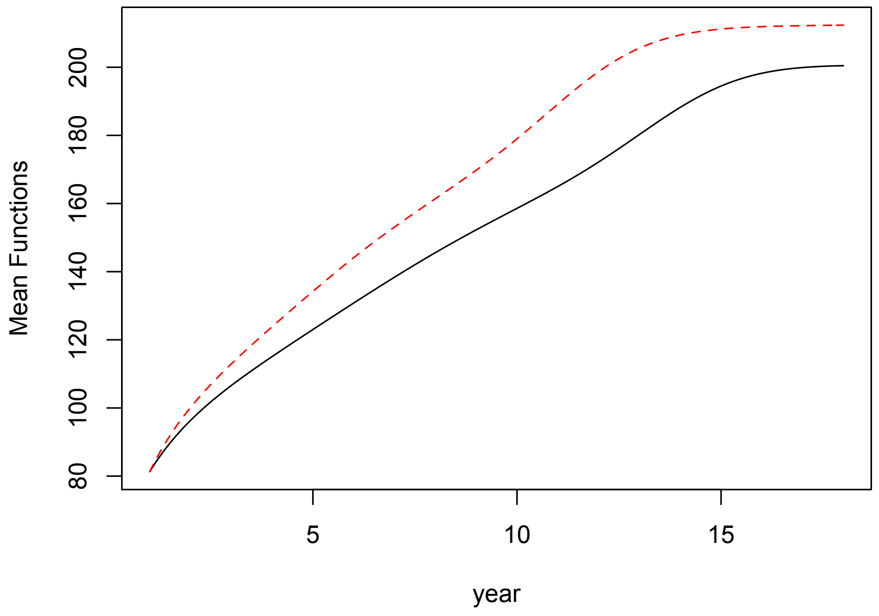

Figure 4.4: The computed mean functions for the two clusters: the red dashed curve is the mean for the male heights, while the black curve is the mean for the female heights.

To see better this characteristic, we have reported, in figures 4.5 and 4.6, the functions $b(x)$ and $w(x)$ found for this clusterization. We can immediately note the absolute maximum of these functions and, when we look at the abscissa in correspondence of the maximum, we find it is 12.25. This fact has an immediate feedback in the real dynamic of growth in children: it is the *pubertal spurt*. As confirmed by auxologists and evolutionists, in [15], girls grow faster than boys and the peak of the process happens at about 12-13 years, age at which girls undergo to radical changes in their body. The same process, instead, happens later to boys, usually at 14-15 years. Thus, the maximum found in the two functions, really indicates the presence of the pubertal spurt and suggests that, while girls are going through changes, boys are still rather late in their evolution. Then, going on in years, the differences smooth and we can guess that $w(x)$ and $b(x)$ have a right horizontal asymptote, having a value given by the difference between the mean height of boys and girls. Before occurring in the maximum, we can see that both functions are monotonically increasing, meaning that height really diversify

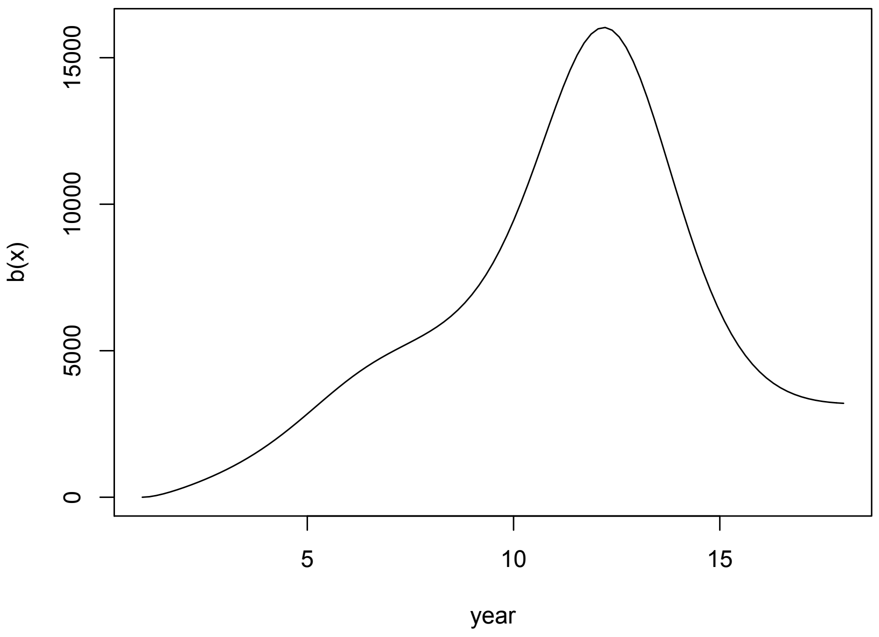

Figure 4.5: $b(x)$ found for the clusterization on nonregistered data.

between sexes, because of a different type of evolution, until, at least, 12 years. The early pubertal spurt in girls could be evidentiated looking at the corresponding velocities and accelerations, i.e. looking at the corresponding first and second derivatives of heights. Indeed, we expect to see a maximum in the female velocities occurring at about 12 years and to find it later if we refer to the male velocities. In figure 4.7, we have plotted the mean velocities for the functions assigned to the clusters found in this case. We can readily note the different years where we find the maximum, as we expected.

Observation 4.1.
The shape of the two curves is very similar: it seems that the mean male velocity is just the female mean velocity, but stretched. In particular, this suggests that human growth follows the same path in boys and girls but with different times.

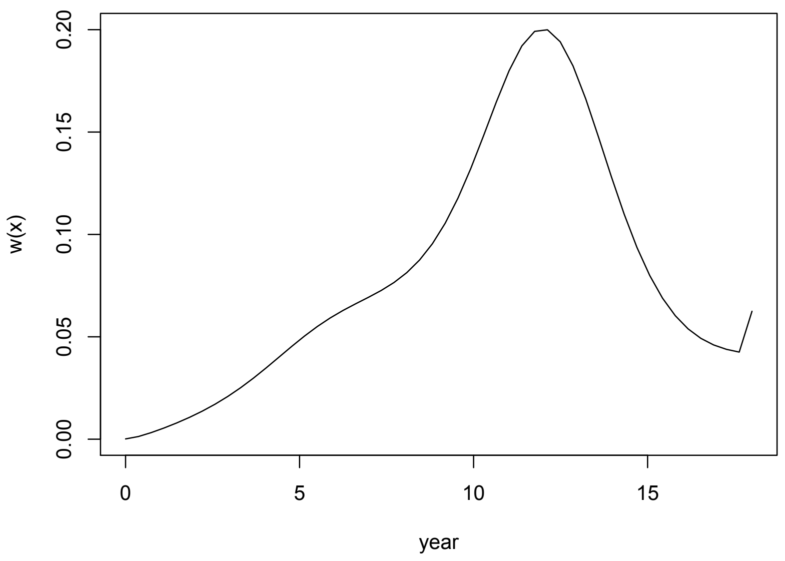

Figure 4.6: $w(x)$ solution to the problem (the point at the right end of the interval is due to possible problems of the algorithm at the boundaries).

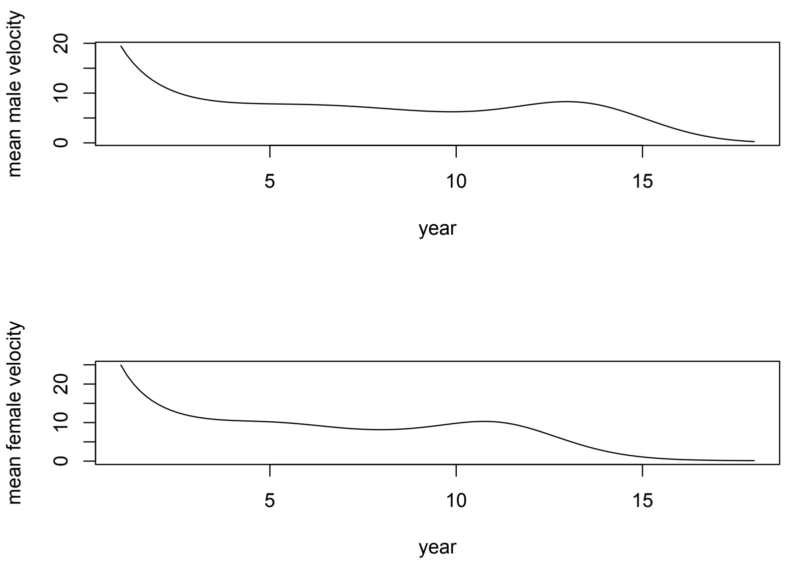

Figure 4.7: In the upper and the lower panel, the mean velocity for boys and girls respecively are plotted.

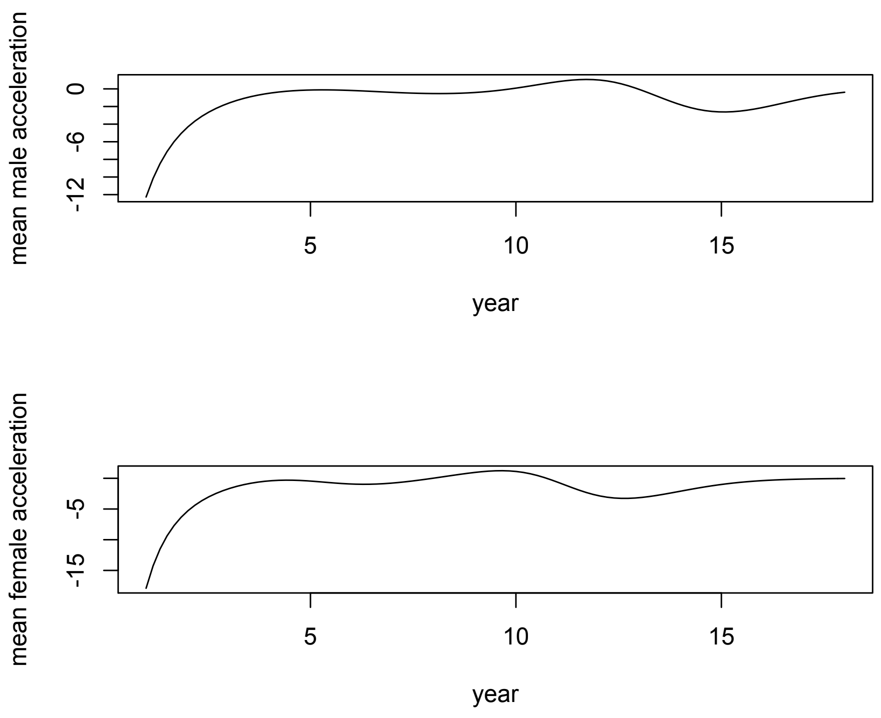

Figure 4.8: In the upper and the lower panel, the mean accelerations for boys and girls respecively are plotted.

Finally, in figure 4.8, we have shown the mean acceleration functions for boys and girls. Even in this case, we can note the different years where we observe the maximum and the shape of the two curves, which is, again, very similar. After the analysis of the heights themselves, we try the clusterization of the derivatives and then of the accelerations. Even in this cases, we do not have outer important informations suggesting the appropriate tuning parameter defining sparsity; therefore, we use the same modified Gap Statistic used previously. For both cases, the optimal s is equal to 4.1231 leading to a corresponding value for the Gap Statistic of 0.0062 and 0.3826.
When we apply weighted K-means to the first and second derivatives of the height function, we do not find good performances. The clusters do not reflect the distributions of the boys and girls found with heights. When applied to the derivatives, the algorithm finds a group of cardinality 18 and, as a consequence, the other one is composed by 75 functions, whereas, when applied

to the accelerations the division in clusters becomes 12 and 81. Moreover, the found clusters are not homogeneous between sexes, so that we can neither state that there are some peculiarities conditioning only boys or only girls. A reason for these erroneous classifications could be found in the observation 4.1. The process of human growth is the same, independently of sex, it only happens at different times, therefore the shapes of the functions are the same and the algorithm easily makes mistakes. Eventually, we could use the analyses on velocities and accelerations to find anomalous process of growing.

In figure 4.9, we have plotted $w(x)$ and $b(x)$ found by application of our

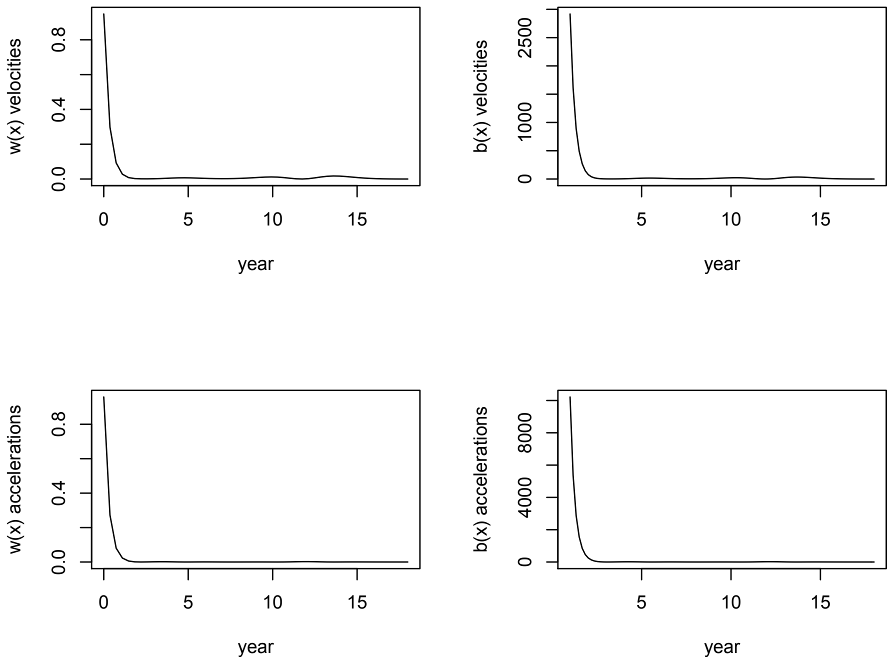

Figure 4.9: The functions $w(x)$ and $b(x)$ found for velocities (the upper panels) and accelerations (the lower panels).

method to velocities and accelerations. The most evident fact is the flat plateau near zero. It is not exactly zero, but the values are extremely small in that interval: $b_{vel}(x)$ is between 7.59×10^{-4} and 7.46×10^{-1}, while $b_{acc}(x)$ is between 8.92×10^{-4} and 9.79×10^{-1}. This means that the two found clusters are composed by almost similar functions from the estimated age of 1.85 years. Before this moment, the two clusters are significantly different, but they do not reflect the separation between sexes.

4.4 Alignment

In this section we will introduce the notion of *data alignment*. The methods here discussed can be found, with more details, in [16].
In general, functional data can exhibit two types of variability: *amplitude variability*, that pertains to the sizes of particular features such as the velocity peak in the pubertal growth spurt, ignoring their timings and *phase variability*, that is variation in the timings of the features without considering their sizes. We have seen, discussing the simulations of Chapter 3, that variation in functional observations involves both phase and amplitude, and that confounding these two leads to many problems. Before we can get a useful measure of a typical growth curve, we must separate these two types of variation, so that features such as the pubertal spurt occur at roughly the same time for all children. The need to transform curves by transforming their arguments, which is what is called *curve registration*, can be motivated as follows. The rigid metric of physical time may not be directly relevant to the internal dynamics of many real-life systems. Rather, there can be a sort of biological time scale that can be nonlinearly related to physical time, and can vary from case to case. Human growth, for example, is the result of many physiological events that are not assured to happen at the same time in everyone. The intensity of the pubertal growth spurts of two children should be compared at their respective ages of peak velocity rather than at any fixed age.
In more abstract terms, the values of two or more function values $f_i(t_i)$ can in principle differ because of two types of variation. The first is the more familiar vertical variation, or amplitude variation, due to the fact that two functions $f_1(t)$ and $f_2(t)$ may simply differ at points of time t at which they are compared, but otherwise exhibit the same shape features at that time. But they may also exhibit phase variation in the sense that functions f_1 and f_2 should not be compared at the same time t because they are not exhibiting the same behavior. Instead, in order to compare the two functions, the time scale itself has to be distorted or transformed.
The simplest case of all is represented by *shift registration*. It means that curves differ between each other because they are only translations of the same prototype function, thus we only need to move them horizontally to eliminate this kind of variability. In this case, we want to find, for each f_i, a parameter δ_i, in order to have the following transformation of data:

$$f_i^*(t) = f_i(t + \delta_i).$$

The estimation of a shift or an alignment requires a criterion that defines when several curves are properly registered. One possibility is to identify

a specific feature or *landmark* for a curve, and shift each curve so that this feature occurs at a fixed point in time. However, the registration by landmark or feature alignment has some potentially undesirable aspects: the location of the feature may be ambiguous for certain curves, and if the alignment is only of a single point, variations in other regions may be ignored.
Instead, we can define a global registration criterion for identifying a shift δ_i for curve i as follows. First we estimate an overall mean function $\hat{\mu}(t)$ for t in the domain. If the individual functional observations f_i are smooth, it usually suffices to estimate $\hat{\mu}$ by the sample average $\bar{f}$. We can now define our global registration criterion by

$$REGSSE = \sum_{i=1}^{N} \int_D [f_i(t+\delta_i) - \hat{\mu}(t)]^2 dt. \tag{4.7}$$

Thus, our measure of curve alignment is the integrated or global sum of squared vertical discrepancies between the shifted curves and the sample mean curve. The target function for transformation in (4.7) is the unregistered cross-sectional estimated mean $\hat{\mu}$. However, one of the goals of registration is to produce a better estimate of this mean function. We therefore expect to proceed iteratively: beginning with the unregistered cross-sectional estimated mean, argument values for each curve are shifted so as to minimize $REGSSE$, then the estimated mean $\hat{\mu}$ is updated by re-estimating it from the registered curves f_i^* , and a new iteration is then undertaken using this revised target.
The second method used to align curves is called *registration landmark.* A landmark or a feature of a curve is some characteristic that one can associate with a specific argument value t. These are typically maxima, minima, or zero crossings of curves, and may be identified at the level of some derivatives as well as at the level of the curves themselves. The landmark registration process requires for each curve f_i the identification of the argument values $t_{i,j}$, $j = 1, \dots, J$ associated with each of J features. The goal is to construct a nonlinear transformation h_i for each curve such that the registered curves with values

$$f_i^*(t) = f_i(h_i(t))$$

have more or less identical argument values for any given landmark. The estimated functions h_i's are called *warping functions* and have to satisfy only two conditions: $h_i(0) = 0$ and h_i must be strictly monotone.
After having found the right warping functions, we have to compute the registered function values $f_i^*(t) = f_i(h_i(t))$, but this requires two more steps:

1. estimate the *inverse warping function* h_i^{-1} such that $h_i^{-1}(h_i(t)) = t$;

2. smooth or interpolate the relationship between $h_i^{-1}(t)$ plotted on the abscissa and $f_i(t)$ plotted on the ordinate.

The third and last method used to align functional data is the *continuous registration*. The least squares criterion (4.7) worked well for simple shift registration, but can bring some problems for more general warping functions. When two functions differ in terms of amplitude as well as phase, the least squares criterion uses time warping to also minimize amplitude differences by trying to squeeze out of existence regions where amplitudes differ. Put another way, the least squares fitting criterion is intrinsically designed to assess differences in amplitude rather than phase.
Suppose two curves f and g differ only in amplitude but not in phase. Then, if we plot the function values f and g against each other, we will see a straight line. Amplitude differences will then be reflected in the slope of the line, a line at 45° corresponding to no amplitude differences. Now thinking about a line as a one-dimensional set of points on a plane, we can turn to principal components analysis as just the right technique for assessing how many dimensions are required to represent the distribution of these points. This technique will yield only one positive eigenvalue if the point spread is, in fact, one-dimensional. That is, the size of the smallest eigenvalue measures departures from unidimensionality.
Let us consider now evaluating both the target function f and the registered function f^* at a fine mesh of n values of t to obtain the pairs of values $(f(t), f[h(t)])$. Let the n by two matrix $\mathbf{X}$ contain these pairs of values. Then the two-by-two crossproduct matrix $\mathbf{X}'\mathbf{X}$ would be what we would analyze by principal components.
The following order two matrix is the functional analogue of the crossproduct matrix $\mathbf{X}'\mathbf{X}$.

$$\mathbf{T}(h) = \begin{bmatrix} \int f(t)^2 dt & \int f(t)f(h(t))dt \\ \int f(t)f(h(t))dt & \int f(h(t))^2 dt \end{bmatrix}.$$

We see that the summations over points implied by the expression $\mathbf{X}'\mathbf{X}$ have here been replaced by integrals. Otherwise this is the same matrix. We have expressed the matrix as a function of warping function h to remind ourselves that it does depend on h.
Consequently, we can now express our fitting criterion for assessing the degree to which two functions are registered as follows:

$$MINEIG(h) = \lambda_2(\mathbf{T}(h)),$$

where the function λ_2 is the size of the second eigenvalue of its argument, which is an order two symmetric matrix. When $MINEIG(h^*) = 0$, we have achieved registration, and h^* is the wanted warping function that does the job. Finally, if we want the warping function h to be endowed of more regularity or smoothness, we can consider a penalized version of $MINEIG(h)$.

4.5 Analysis On Aligned Data

As we had anticipated in the introduction of this chapter, after having introduced the techniques of data registration, we, now, move to an analysis of the same data, but, this time, they are aligned.
The registration of the curves of the dataset is done using the last method seen: continuous registration. The class of warping functions $h(t)$ used to do the job, was chosen *apriori*: linear affine functions. This choice was made, fundamentally, for two reasons. The first motivation follows from what we have stated in observation 4.1: the curves look all quite similar; therefore it is reasonable to think that, thanks to a linear affine rescalement of the biological time of each child (since the data seems only traslations and dilatations of the same prototype), we will able to decouple the amplitude and phase variabilities. The second motivation is due to the possibility of making a comparison with the article [19]. In that paper, the authors, develops a method for making, at the same time, alignment and classification of data. They use a functional optimized to linear affine warping functions and prove that the similarity index there defined satisfies a number of properties. Considering the same class of warping functions puts us in condition to compare the results.
As we can see in figure 4.10, the aligned curves are all very similar. This fact does not surprise us, because we had already observed that growth is a process that is the same in every human being. The registration of data only confirms this statement. In figure 4.11, we have reported the warping functions used for alignment. The warping functions for the boys, typically, are the highest lines, but, we are not able to divide exactly those used for boys from those used for girls. There is not a clear distinction between sexes. Indeed, when we try to cluster the aligned data, we do not obtain useful results. Our algorithm, and classical K-means as well, can not recognize the presence of two clusters and the process does not even start, because the two procedures try to generate two cluster centres from what is seen as one group. Therefore, we opted for conducting a supervised analysis on the aligned dataset; we decided to make clearer the real difference between sexes, by applying a shift registration: we traslated the argument of the aligned functions to evidentiate the presence of two distinct groups. The result of this operation is shown in figure 4.12. With this analysis, we can estimate easily the shift

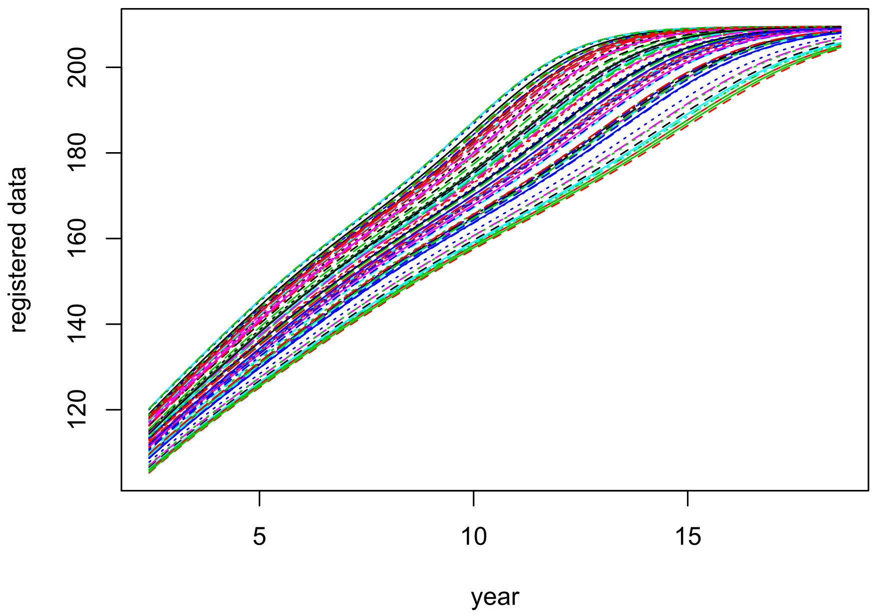

Figure 4.10: The aligned height curves, using linear affine warping functions.

in the biological clocks of boys and girls. Obviously, the highest functions represent the male group and, in this case, the algorithm performs better: it correctly recognizes the two clusters. With this data the advantage of the method proposed in [19] is the immediate identification of only one cluster, thanks to the contemporary process of alignment and clusterization. Moreover, they show that, eventually, we could identify two or more clusters, but this detail does not justify the computational effort.

Considering the aligned velocities and accelerations of growth leads to the same problems: there are no definite different clusters. Therefore, we apply another shift registration on both aligned velocities and accelerations. In figure 4.14 the so obtained velocities are plotted. From these two plots, the same shape for the functions of boys and girls is even more evident: with only horizontal movements, we can overlap them. This is another evidence confirming the same structure of growth, that joins every human, with no regards of sex. The only aspect differentiating between sexes is the time at which we find maxima or minima in the functions, as we have already noted for the pubertal spurt. Thus, the shift applied to distinguish the two groups

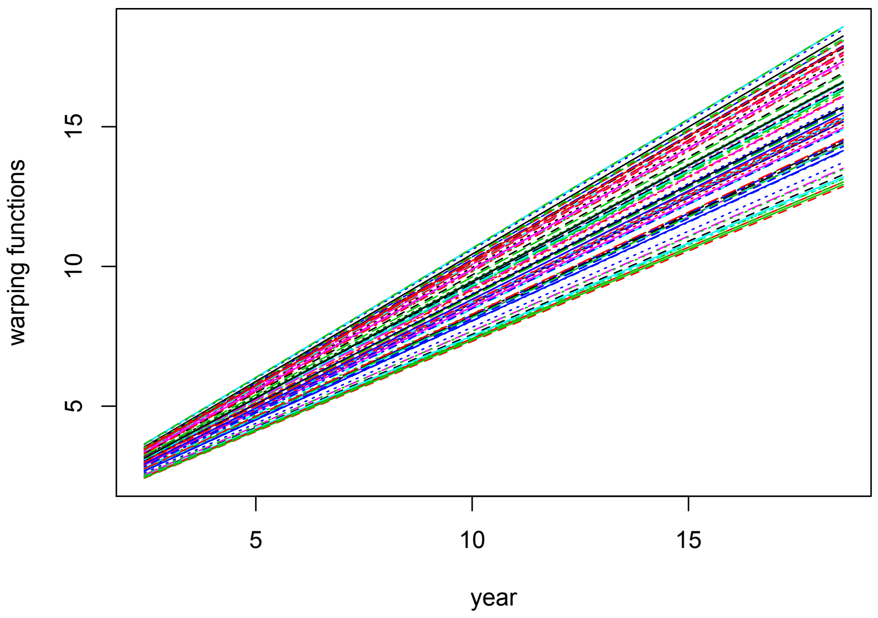

Figure 4.11: The warping functions used for alignment.

means that the biological clock of girls is just the same as that of boys, but it is forward of a constant quantity. The calculated mean shift is equal to 1.978 years.

In figure 4.16 we have plotted the registered accelerations, but after having shifted the two groups. Again, we can note the same shape in the functions; the only difference between them is a horizontal traslation. The algorithm correctly clusters boys and girls. In figure 4.17, we have shown the computed mean acceleration functions for boys and girls. The calculated mean shift between boys and girls is about 2.00 years, a value close to that found for velocities and really reflecting the different biological times of the two sexes.

In the end, we have found almost the same results of the article [19]: the structure of human growth is the same for everyone, the only difference between sexes is a shift in the biological clocks, even if, the differences between boys and girls, in [19], were found in terms of warping functions, but this comes from the algorithm there adopted. Therefore, after having aligned the data, we can consider the existence of only one distribution generating the curves. However, this is achieved, in that article, by jontly registrating and cluterizing, a method that proves that sometimes useful results can not come

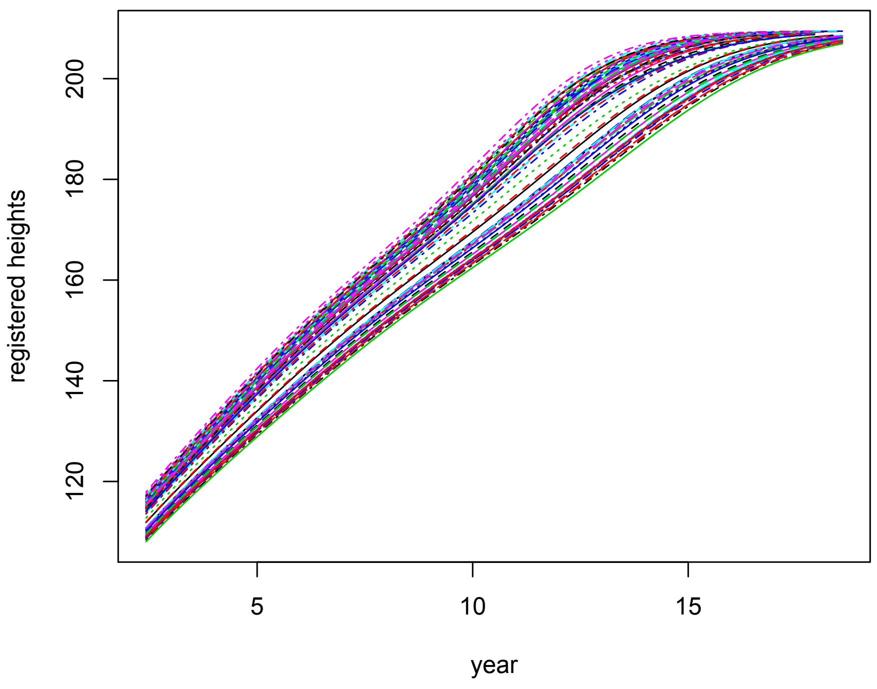

Figure 4.12: The aligned curves, but shifted, in order to distinguish better the presence of different sexes.

from solely clustering or solely alignment. Indeed, this is pointed out, also, from the fact that, when we applied our algorithm to misaligned velocities and misaligned accelerations, we came out with not so useful informations about the existence of two groups. Anyway, our method performs well even with misaligned height curves. The reason could lie in the fact that it is true that the process of growth is the same in everyone, but the consequences of the process, manifestating in the height curves, are different because of the shift between the biological clocks of boys and girls. Moreover weighted functional K-means has the property, as viewed in the previous chapter, of being only weakly affected of the eventual noise of data, because it considers also the shape of curves. This is an aspect that could contribute in recognizing the effective presence of two groups even with misaligned data, but the same trends in velocities and accelerations confounds the method.
We have found the same results of [19], even if we have faced the problem from another point of view: we had a functional that considers the differences

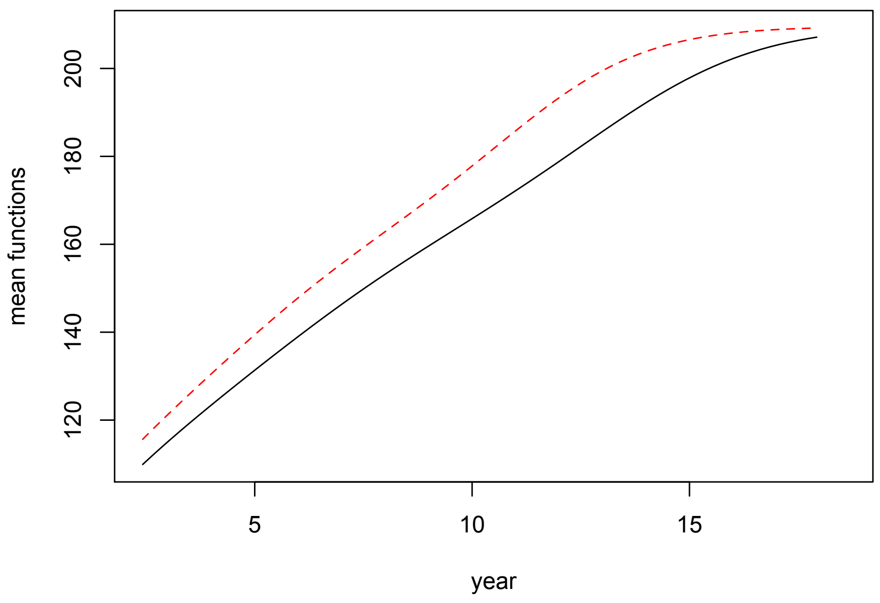

Figure 4.13: The computed mean function for the two clusters: the dashed function is the mean height for the boys, while, the other one is that of girls.

between data, so we had a measure of dissimilarity between functions; on the other side, the functional of that paper is a measure of similarity. However it is well known that, given a dissimilarity measure, a similarity could always be defined. The contrary is not always true. So, we could define a similarity measure starting from our functional and we would come out with the same results.
The jointly combination of aligment and clustering process could suggest us new directions of research: trying to do the same with the functional here considered is probably the most immediate; after that we could seek for classes of warping functions, if they exist, such that our functional remains unaltered. Another interesting path that could be followed, is the investigation of the selection of the tuning parameter when this new method is adopted: does the conjunction of alignment and clustering change the determination of the parameter s? How is the notion of sparsity modified? Could we define new methods for parameters selection? Are they modified, if we consider jointly alignment and clustering? However we will not go through these questions here.

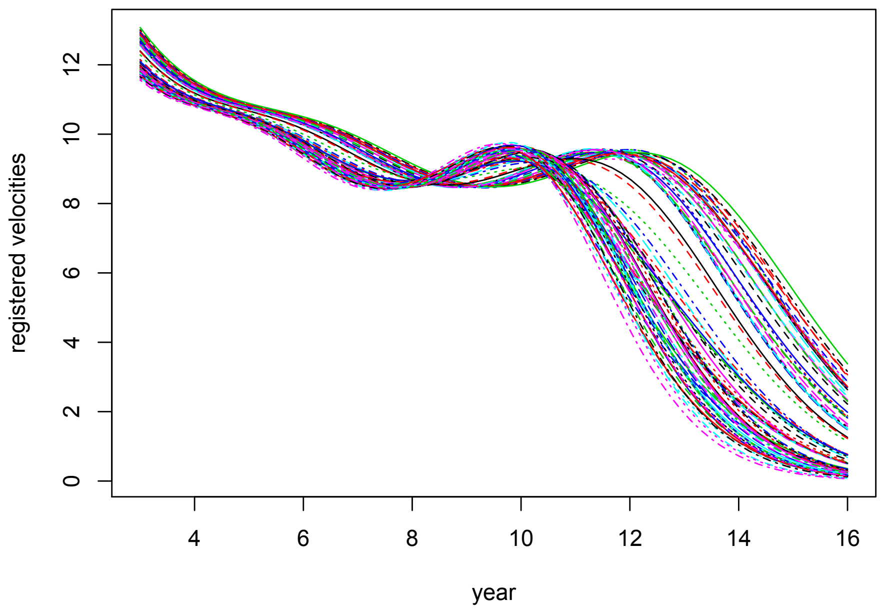

Figure 4.14: The aligned velocities, but shifted between different sexes.

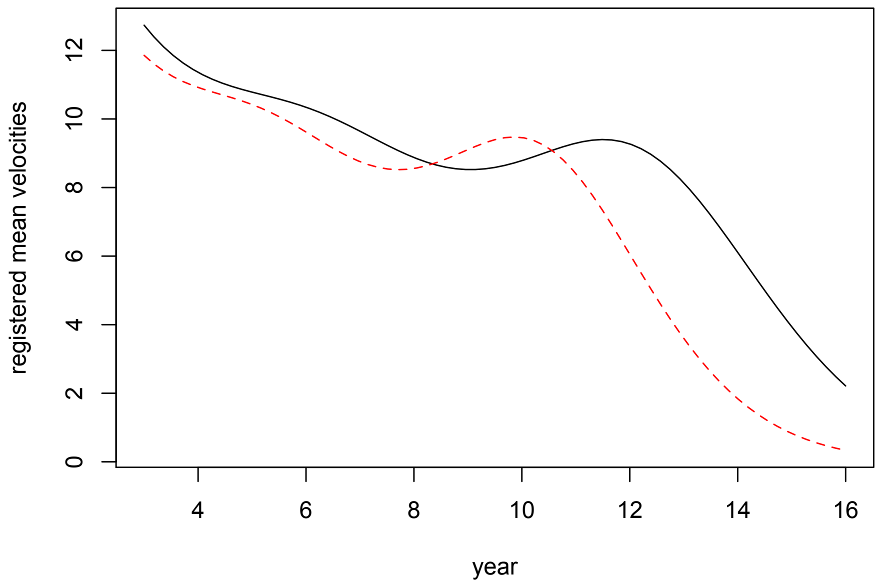

Figure 4.15: The mean functions for the velocities. The dashed curve is the mean for the girls and note the maximum occurring earlier than the maximum for the boys.

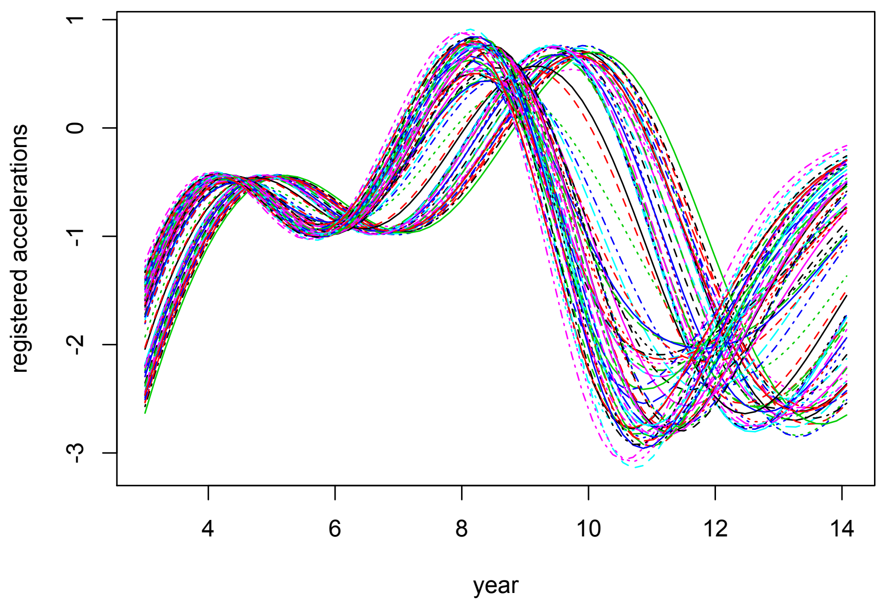

Figure 4.16: The aligned accelerations, but shifted between different sexes.

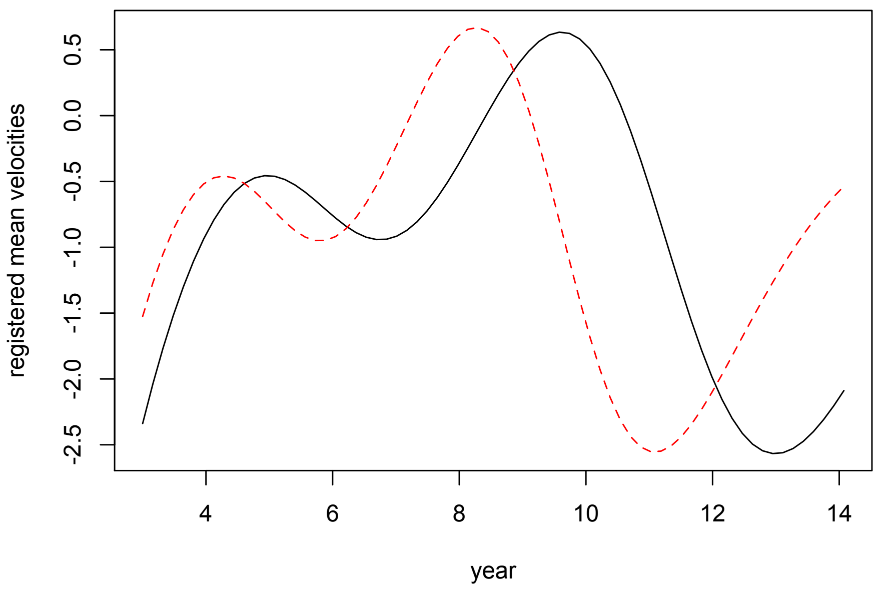

Figure 4.17: The mean acceleration functions.

Chapter 5

Analysis of Three-Dimensional Cerebral Vascular Geometries

5.1 Introduction

In this chapter, we will analyze a functional dataset coming from a real study. It is composed of curves describing the carotid of 65 patients, taken in the contest of the AneuRisk Project. To every patient three functions are associated: the x, y and z coordinates of the centerline of their carotid. The aim of the analysis is to classify and recognize similar shapes of the carotids. Therefore, we want to apply our method to this dataset and, thanks to the weight function, we want to see what differentiates the possible clusters the most. However, before proceeding in the analysis, we have to extend the method defined in chapter 2 to vectors of functions. We will extend Theorem 2.1 and propose another algorithm able to compute that result numerically. After that, we will take a closer look at the dataset and finally we will apply weighted functional K-means to it. In the end of the final section of this chapter, we will use our method to perform *supervised learning*: indeed, knowing particular groups of interest of types of patients, we will look at those features which contributes in distinguishing the most those clusters.

5.2 Multidimensional Weighted Functional K-Means

Now, we want to extend the result found in Theorem 2.1, to the case of vectors of functions. We are now supposing that the data at our disposal are vectors in $\mathbb{R}^n$, $n > 1$, and every component is a function, i.e.:

$$\mathbf{f}(x) = \begin{pmatrix} f_1(x) \\ f_2(x) \\ \vdots \\ f_n(x) \end{pmatrix}.$$

The functional setting of our problem undergoes some little modifications. We are now considering functional variables $\mathcal{X}_k$'s such that:

$$\mathcal{X}_k : \Omega \longrightarrow L^2(D;\mathbb{R}^n),\ \forall k = 1, \dots, K. \tag{5.1}$$

Therefore, the data are functions such that:

$$\mathbf{f}(x) : D \longrightarrow \mathbb{R}^n, \tag{5.2}$$

which belongs to $L^2(D;\mathbb{R}^n)$ and when we compute the distances between data we have to consider the pairwise component distances:

$$d_{i,j} = d(\mathbf{f}^i, \mathbf{f}^j) = \sum_{m=1}^{n} \int_D \left(f_m^i - f_m^j\right)^2 d\mu. \tag{5.3}$$

This slightly different setting modifies, also, the BCSS we have to consider in our problem, but the extension is straightforward:

$$\mathbf{b}(x, \mathbf{f}, \mathbf{C}) = \begin{pmatrix} b_1(x, \mathbf{f}, \mathbf{C}) \\ b_2(x, \mathbf{f}, \mathbf{C}) \\ \vdots \\ b_n(x, \mathbf{f}, \mathbf{C}) \end{pmatrix},$$

where

$$b_m(x, \mathbf{f}, \mathbf{C}) = \left(\frac{1}{2N} \sum_{i,j=1}^{N} \left(f_m^i - f_m^j\right)^2 - \sum_{h=1}^{k} \frac{1}{|2C_h|} \sum_{\mathbf{f}^i, \mathbf{f}^j \in C_h} \left(f_m^i - f_m^j\right)^2 \right), \tag{5.4}$$

for every $m = 1, \dots, n$, $\mathbf{C} = (C_1, \dots, C_K)$, each cluster has cardinality $|C_h|$ and N is the total number of observations. The weights we have to consider become a vector of weight functions:

$$\mathbf{w}(x) = \begin{pmatrix} w_1(x) \\ w_2(x) \\ \vdots \\ w_n(x) \end{pmatrix}.$$

Here we have that w_1 will weight the first component of the data, thus it will be multiplicated to the first component of the vector $\mathbf{b}$, w_2 to the second component and so on. All the weight functions have to belong to $L^2(D)$ and precisely to the unit ball of that space. We want them all to be nonnegative and finally, they should perform sparse clustering. As we had already observed in the second chapter, the constraint on the $L^1(D)$ norm of the weight functions does the job. The sparsity structure, however, could be different for each component examined. So, we have to request that:

$$\|w_1\|_{L^1(D)} \leq s_1, \; \|w_2\|_{L^1(D)} \leq s_2, \; \dots, \; \|w_n\|_{L^1(D)} \leq s_n, \tag{5.5}$$

with $s_1, s_2, \dots, s_n > 0$.
Now, we are ready to define the new sparse clustering problem.

Definition 5.1.
We define Multidimensional Sparse Functional K-means Clustering to be the solution to the following problem:

$$\max_{\mathbf{w}(x),\mathbf{C}} \int_D \langle \mathbf{w}(x), \mathbf{b}(x, \mathbf{f}, \mathbf{C}) \rangle \, d\mu, \tag{5.6}$$

$$\textit{subject to: } \begin{cases} \|w_1\|_{L^1(D)} \leq s_1, \; \|w_2\|_{L^1(D)} \leq s_2, \; \dots, \; \|w_n\|_{L^1(D)} \leq s_n, \\ \|w_1\|^2_{L^2(D)} \leq 1, \; \|w_2\|^2_{L^2(D)} \leq 1, \; \dots, \; \|w_n\|^2_{L^2(D)} \leq 1, \\ w_1(x) \geq 0, \; w_2(x) \geq 0, \dots, w_n(x) \geq 0 \; \mu - a.e. \end{cases} \tag{5.7}$$

The parameters s_i's are to be determined, if they are not known *apriori*. Anyway, we suppose we know the right value for each of them. Then, we can state the following result:

Theorem 5.1.
Let $D \subset \mathbb{R}$ be a Borel set, such that $\mu(D) < \infty$, $\mathbf{f}^i$ be N vectorial functions, s.t. $f^i_m \in L^2(D), f^i_m \in L^\infty(D)$ $\forall i = 1, \dots, N$, $\forall m = 1, \dots, n$ and suppose there exists an $M \in \mathbb{R}$ such that $\|f^i_m\|_{L^\infty(D)} \leq M$ $\forall i$ and $\forall m$. Then there exists a unique solution to the following problem:

$$\max_{\mathbf{w}(x),\mathbf{C}} \int_D \langle \mathbf{w}(x), \mathbf{b}(x, \mathbf{f}, \mathbf{C}) \rangle \, d\mu, \tag{5.8}$$

with $K < N$ is the (fixed) number of clusters, $\mathbf{C}$ is the clustering vector and the vectorial function $\mathbf{w}(x)$ must satisfy the following constraints:

$$\begin{array}{l} \|w_1\|_{L^1(D)} \leq s_1, \; \|w_2\|_{L^1(D)} \leq s_2, \; \dots, \; \|w_n\|_{L^1(D)} \leq s_n, \\ \|w_1\|^2_{L^2(D)} \leq 1, \; \|w_2\|^2_{L^2(D)} \leq 1, \; \dots, \; \|w_n\|^2_{L^2(D)} \leq 1, \\ w_1(x) \geq 0, \; w_2(x) \geq 0, \dots, w_n(x) \geq 0 \; \mu - a.e. \end{array} \tag{5.9}$$

Proof.
The proof has the same structure of that of Theorem 2.1. For each of the integrals coming from the explicit writing of the scalar product in (5.8), after having fixed $\mathbf{C}$, we know that the solution exists, it is unique and we know, also, how to build it and its properties. Thus we have a vector of weight functions solution to this first step.
Then, holding fixed an admissible vector of weight functions, there exists a solution in terms of clusters: since we have a limited, though potentially very large, number of possible clusters, we can, after enumerating them, choose the combination which maximizes the functional in (5.8).
Finally, we have to show that the solutions exists when both $\mathbf{w}(x)$ and $\mathbf{C}$ change. So, we start with a vector of functional weights constant over D and then, iteratively, apply the two previous steps. We end with inequalities:

$$\int_D \langle \mathbf{w}_1^*(x), \mathbf{b}(x, \mathbf{f}, \mathbf{C}_1^*) \rangle \, d\mu \leq \ldots \leq \int_D \langle \mathbf{w}_r^*(x), \mathbf{b}(x, \mathbf{f}, \mathbf{C}_r^*) \rangle \, d\mu, \tag{5.10}$$

where we have called with $\mathbf{w}_r^*(x)$ and $\mathbf{C}_r^*$, respectively, the optimal weight vector and the optimal clusters at the r-th step. Now, since the number of possible clusters is finite, there will be an inequality where two clustering are the same, i.e.: $\mathbf{C}_r^* \equiv \mathbf{C}_p^*$, for two indeces r and p. Then, supposing $r > p$, considering these two iterations, we can write:

$$\int_D \left\langle \mathbf{w}_r^*(x) - \mathbf{w}_p^*(x), \mathbf{b}(x, \mathbf{f}, \mathbf{C}_r^*) \right\rangle d\mu \geq 0. \tag{5.11}$$

Since the second term in the scalar product is composed by always nonnegative components, we must have that the functional vector $\mathbf{w}_r^*(x) - \mathbf{w}_p^*(x)$ must be composed by always nonnegative components as well. Now, suppose there exists at least one component, say j, for which it holds that: $w_r^{j*}(x) - w_p^{j*}(x) > 0$. Then we would end with an absurd: the way in which we built the solution function and the uniqueness of the weak limit force $\mathbf{w}_p^*(x)$ not to be the optimal solution to the p-th passage. So, either $\mathbf{w}_r^*(x) - \mathbf{w}_p^*(x) \equiv \mathbf{0}$, or the solution at the r-th passage is the global optimum and the algorithm ends after a finite number of steps. □

The reasoning behind a numerical algorithm performing Theorem 5.1 is very similar to that behind the algorithm for the univariate case. The only detail we have to investigate more carefully, is the selection of the optimal solution clusters at each stage of the procedure. There do not already exist numerical methods performing joint clustering, so we have to implement a possible algorithm from scratch. The solution we propose to this problem is the following. The idea is to compare the values coming from the optimization of the functional. We begin conducting separate clusterings on the different components of the data, and, for each of them, we record the optimal clusters. Then, we put the various optimal groups in each of the integrals of the functional in (5.8) and calculate the optimal weight functions associated. Finally, we compute the value of that functional with every optimal clustering and the associated weight functions and we consider optimal the clusters to which is associated the highest value of the functional with this procedure. Note that, first of all, this procedure is replicated at every step of the algorithm and not only at its final step and, finally, this criterion is optimal from the point of view of the maximization of the objective functional, even though it could miss the global optimum.

5.3 The Dataset of the AneuRisk Project

In this section we will briefly discuss the data considered before proceeding in the analysis.
The AneuRisk Project is a joint research program that aims at evaluating the role of vascular geometry and hemodynamics in the pathogenesis of cerebral aneurysms, see, for instance [18] for further details. The data considered in the analysis here presented are the three $x(s)$, $y(s)$ and $z(s)$ spatial coordinates, measured in mm, of 65 Internal Carotid Artery (ICA) centerlines, measured on a fine grid of points along a curvilinear abscissa s, decreasing from the terminal bifurcation of the ICA towards the heart. These 3D reconstructions of the carotids were taken thanks to 3D angiographies and the data were estimated with three dimensional free knot regression splines. We are interested in clustering the three-dimensional centerlines, in order to classify ICA's possible different morphological shapes. Since the shape of the ICA influences the pathogenesis of cerebral aneurysms through its effects on the hemodynamics, such a classification could in fact be helpful in the determination of the risk level of a given patient. In reality, we will consider the derivatives of the estimated curves and in figure 5.1 we have plotted the data. However, the curves could not have the same length: the angiographies were not taken starting from the same point and this resulted in having curves defined on different domains; on the other hand, different patients have different features in different points of their own carotid. This fact complicates the situation, but we decided, following [19], to perform the analysis only on the common part of the domains. We are able to do so because there is a set, coming from the intersection of all domains, where every function is defined and, moreover, that interval coincides with interesting features to be observed in the geometry of the carotids, otherwise, it would be unuseful the recording of that particular part. Naturally, the diversity of the domains

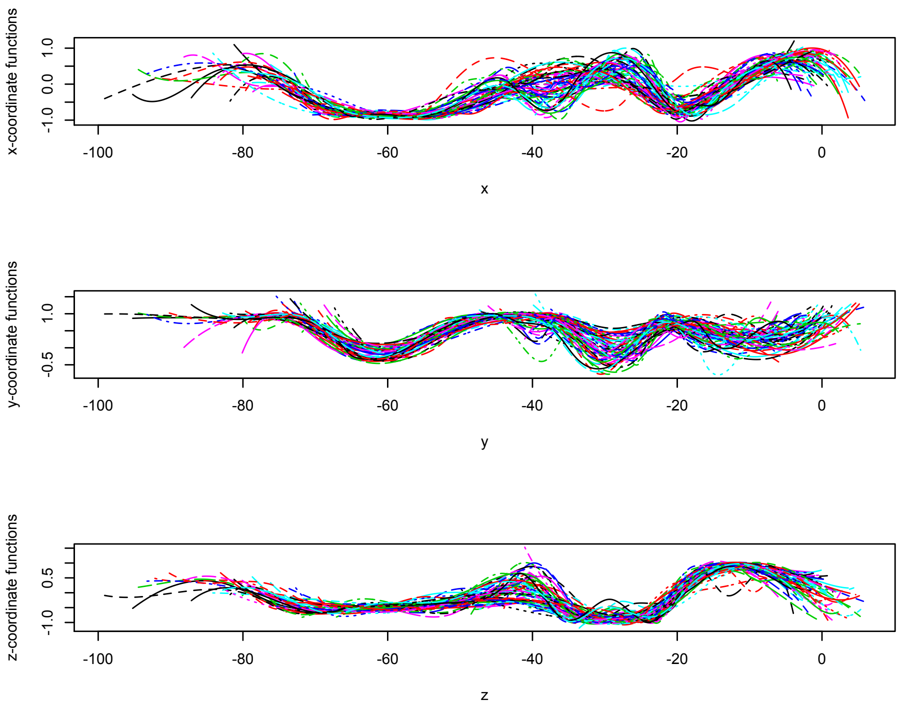

Figure 5.1: The three spatial derivatives coordinate functions.

implies a high misalignment of the data. Therefore, we have to register them before conducting any analysis. The curves in figure 5.1 are already registered; note that the different lengths of the functions are very evident.
Brain aneurysms represent a very serious illness which is associated with a high rate of mortality and disability, even though ruptured aneurysms are relatively uncommon. A brain aneurysm is a weak bulging spot on the wall of a brain artery very much like a thin balloon or weak spot on an inner tube. Over time, the blood flow within the artery pounds against the thinned portion of the wall and aneurysms form silently from wear and tear on the arteries. As the artery wall becomes gradually thinner from the dilation, the blood flow causes the weakened wall to swell outward. This pressure may cause the aneurysm to rupture and allow blood to escape into the space around the brain. A ruptured brain aneurysm commonly requires advanced surgical treatment. There are two kinds of aneurysms: saccular aneurysms are the most common type of aneurysm and account for 80% to 90% of all intracranial aneurysms and are the most common cause of nontraumatic subarachnoid hemorrhage (SAH). This aneurysm looks like a sac or berry forming at the bifuraction or the "Y" segment of arteries. It has a neck and

stem. These small, berry-like projections occur at arterial bifurcations and branches of the large arteries at the base of the brain, known as the Circle of Willis. The other kind of aneurysm is the fusiform aneurysm and is a less common type of aneurysm. It looks like an outpouching of an arterial wall on both sides of the artery or like a blood vessel that is expanded in all directions. The fusiform aneurysm does not have a stem and it seldom ruptures. We now report few facts and statistics about cerebral aneurysms [1]

- The estimated number of people having an unruptured brain aneurysm, is 1 every 50 people.
- There is a brain aneurysm rupturing every 18 minutes. Ruptured brain aneurysms are fatal in about 40% of cases. Of those who survive, about 66% suffer some permanent neurological deficit.
- Approximately 15% of patients with aneurysmal subarachnoid hemorrhage (SAH) die before reaching the hospital. Most of the deaths from subarachnoid hemorrhage are due to rapid and massive brain injury from the initial bleeding which is not correctable by medical and surgical interventions.
- 4 out of 7 people who recover from a ruptured brain aneurysm will have disabilities.
- Women, more than men, suffer from brain aneurysms at a ratio of 3:2.
- Accurate early diagnosis is critical, as the initial hemorrhage may be fatal, may result in devastating neurologic outcomes, or may produce minor symptoms. Despite widespread neuroimaging availability, misdiagnosis or delays in diagnosis occurs in up to 25% of patients with subarachnoid hemorrhage (SAH) when initially presenting for medical treatment. Failure to do a scan results in 73% of these misdiagnoses. This makes SAH a low-frequency, high-risk disease.
- There are almost 500,000 deaths worldwide each year caused by brain aneurysms and half the victims are younger than 50.

From all these facts we see that an early analysis of the carotids is fundamental in order to avoid future diseases and this kind of investigation could help further the work of physicians.

[1] Statistics taken from The Brain Aneurysm Foundation website.

5.4 Analysis of the Geometry of the Carotids

Now we will proceed in the analysis of the dataset.
To begin with, we will perform separate analyses on the three coordinates derivatives and compare the results so obtained, to see if these classifications are enough to gather some informations on the carotids. Then we will cluster jointly the functions and compare the result with a previous work on the same dataset and, finally, we will use our method in the case of supervised learning, knowing the groups of patients with pathology on the Circle of Willis, those with pathology on ICA or healthy.
First of all we have to determine the right tuning parameters s_1, s_2 and s_3 that bring to the right sparse clusterization. We have no specifical outer informations to decide the right values, therefore we have to calculate them via the same previously adopted numerical procedure. Unfortunately, we do not even know how many clusters we have to seek for. Thus we calculate the statistic $S(k)$ for the three coordinates, using the aligned data on the common domain.
Even if the plot of the statistic shows, in each case, some fluctuations (as we can see in figure 5.2, mainly on the z coordinate), in each case $S(k)$ is maximized for $k^* = 2$ and this fact confirms the number of groups found by the authors in [19]. From these plots we can already think that clustering separately the three coordinates will not provide the fairest results and, mainly, the z coordinate seems to be that which gives the most problematic situation.

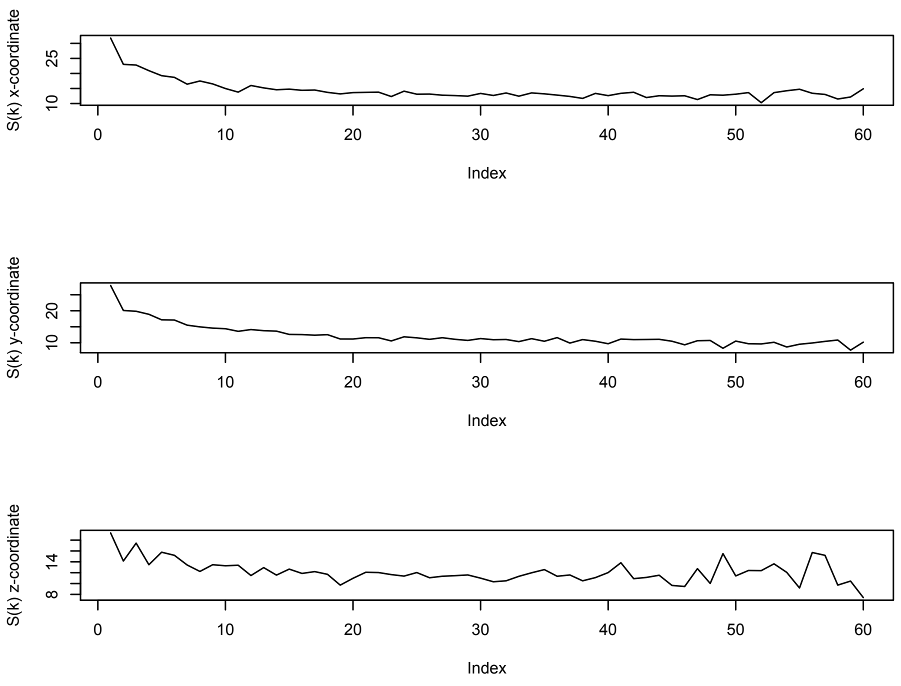

Figure 5.2: The three statistics $S(k)$ computed for each coordinate. Note that in each case, the optimal number of clusters is 2.

After these preliminary observations, we can calculate the optimal tuning parameters in case there were only two groups. The resulting values are very similar for each coordinate: 5.080 with Gap Statistic value of 2.2774 for x, 5.074 with Gap Statistic value of 2.1007 for y and 5.080 again with Gap Statistic value of 1.7941 for z.
Now we can apply our algorithm to the three separate coordinates. We immediately see that the three clusterizations do not produce the same groups, even if those coming from the x and y coordinates are the most similar. The vector containing the calculated groups of the z coordinate, instead, is completely different from the others, the clusters are less defined and this confirms what we have observed about the graph of the statistic $S(k)$ in this case.

In figure 5.3, we can see the group means in case of separate clusterizations. The plot of the means for the z coordinate is not informative: the shape of the two groups is very similar and it seems that they differentiate each other only for the values of the functions themselves. The other graphs are more interesting. We can observe that the first part of the domain is what differentiate the clusters the most. The remaining final part seems not

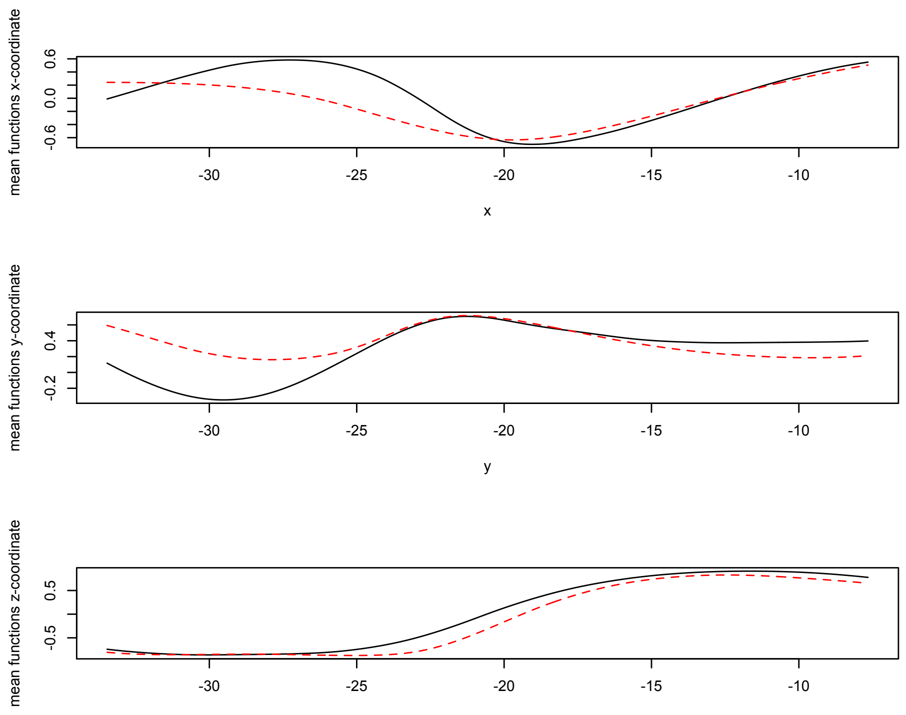

Figure 5.3: The three computed cluster means when the clustering is conducted separately on the three coordinates.

so important in the diversification: the clusters only differ for the values assumed by the functions, but not for their shape. Moreover, in the y and, even more prevalently, in the x coordinate, we can recognize a distinctive shape in the two clusters: one group (indicated with a continuous line) is composed by curves with an evident maximum and an evident minimum, while, the other (indicated with a dashed line) is composed by only a minimum, in case of the x coordinate and less prominent maximum and minimum in the y coordinate.

In figures 5.4, 5.5 and 5.6 we have reported the plots of the b and w functions in the three cases. Actually, we can observe the fact that for the first two coordinates the initial part of the interval is important to the differentiantion and in case of y coordinate, even a final part of it is not negligible. A case on its own is represented by the remaining coordinate. The two plots show many peaks and an absolute maximum located at the right end of the interval, but, the importance given to the first part of the domain is small. This suggests that y and z take more into account the part close to the bifurcation. The cardinality of the found clusters is the same in case of x and

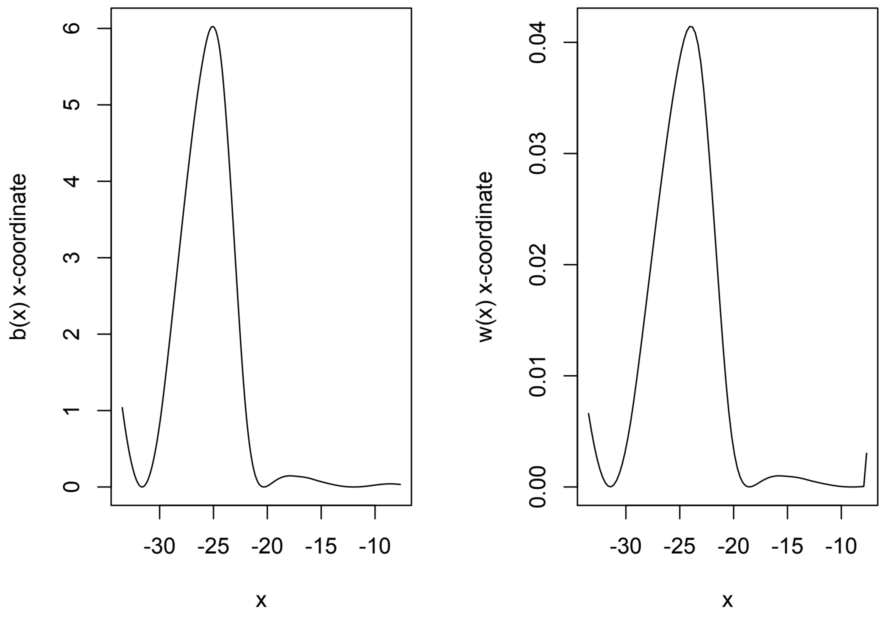

Figure 5.4: The computed functions b and w for the x coordinate, in the separate analysis.

y coordinates: 31 and 34, even if they do not coincide, whereas, for the z coordinate the two groups have cardinality 30 and 35. Only three functions differentiate the groups when the clustering process is conducted separately on the x and y coordinates.
Now we can move on to focus on the joint clustering of the coordinates. The lackness of deep studies in the field of joint clustering does not provide us with methods suitable to compute a statistic that is, in some sense, the "multivariate" version of $S(k)$. Thus we are in trouble in finding the right number of clusters. However, since for each of the coordinates we obtained that the estimated number of clusters is 2, we keep holding this assumption. We have to calculate, also, the vector $\mathbf{s} = (s_1, s_2, s_3)^T$ of the optimal tuning parameters. Even in this case, if we refer back to the previous analysis, we see that the values found in those cases were almost equal and very close to the maximum allowed value: $\sqrt{25.814}$, where the number under the square root is the measure of the common part to the all domains. Therefore we have decided to take $s_1 = s_2 = s_3 = \sqrt{25.814}$.
After having precised all these questions, we can apply our modified algo-

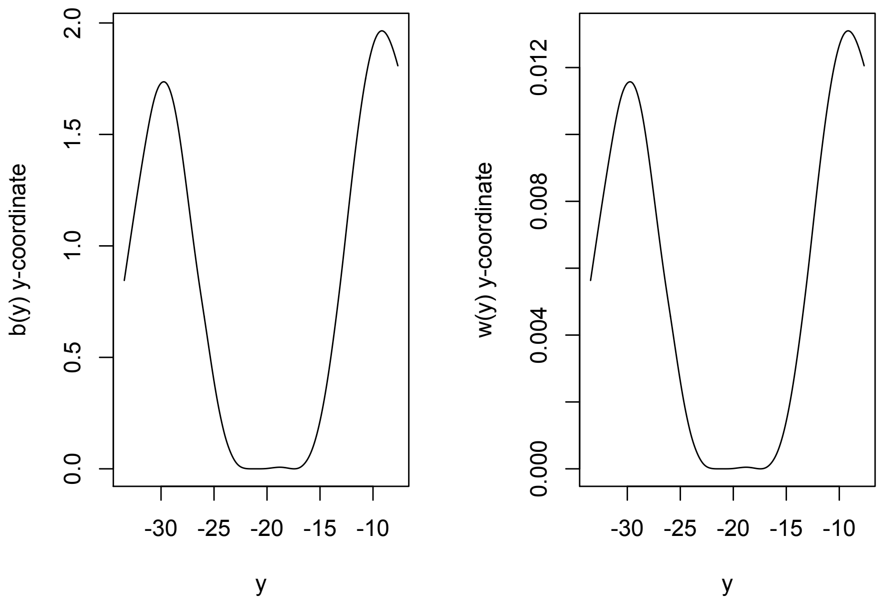

Figure 5.5: The computed functions b and w for the y coordinate, in the separate analysis.

rithm to perform joint clustering. We obtain a clustering vector that is different from all those previously found with separate clusterizations. The clustering vector found with the application of the univariate algorithm to the x coordinate is the most similar to the new found vector, with only 7 curves reassigned, followed by that of the y coordinate, with 19 reassignments and, finally, the most different is that of the z coordinate, with the reassignments of all but 6 functions. In figure 5.7 we have reported the new calculated mean functions.

We can immediately note some aspects distinguishing the new mean functions. First of all in the case of x coordinate, it seems there are not so many changes: the shapes look quite the same, except in the interval $[-20, -15]$, where we can note a more evident separation with the separate clustering. Instead, with the joint algorithm this difference is smoothed. However it is quite simple to understand why this is happening. In separate clustering, the algorithm has to take into account only one set of data and, therefore, it focuses on the differences of only that dataset, without considering the other coordinates.

The mean functions for the other two coordinates undergo to more distinct

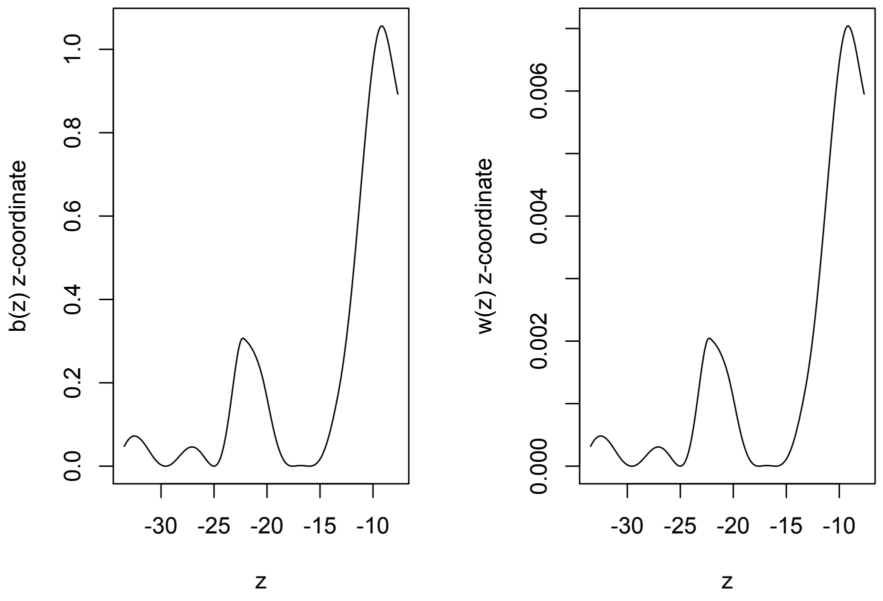

Figure 5.6: The computed functions b and w for the z coordinate, in the separate analysis.

changes. For the y coordinate, in the initial part of the domain the mean functions are quite the same. However, approximatively from the point $x = -24$, they change. The differences in the clusters are underlined: the dashed line, previously almost equal to the continuous one, is now a bit under it in the inteval $[-24, -20]$ and a bit above in $[-20, -13]$. Note, that, now, the two mean functions intersecate two more times than before.
Finally, we can look at the z coordinate. Here the changes happen mostly in the first part of the domain, whereas the differences between the groups smooth in last part. Even if it not so evident, the continuous line now shows a maximum and the gap in $[-25, -15]$ previously found is almost eliminated in the joint clusterization. Morever we can deduce, from the fact that there are more intersections than before between the mean functions, that this new clusterization is more reliable: the algorithm has distinguished mainly basing on the shape of data, rather than the values they assumed.
There is a common aspect in all these plots: the mean functions mainly differ on the first part of the domain. Therefore we can infer that it suffices to consider only this part to decide to which group assign a new observation. The analysis, here conducted, was completely *unsupervised*, meaning that

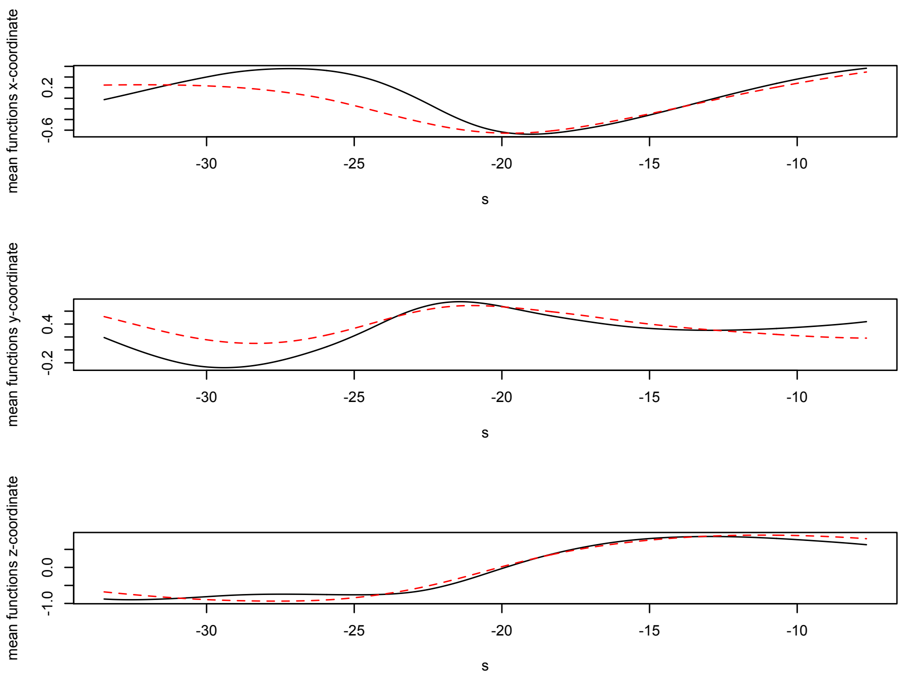

Figure 5.7: The computed mean functions with the joint clusterization.

we had no previously collected informations. Specifically, we did not know what to expect in the shape of the groups found. However, we can still note the presence of two distinct shapes and already found in literature, e.g. [19]. These shapes are called Ω and S shape, depending on the number of syphons of the carotid: one in case of Ω and two in case of S.
We can compare the clusters here found to those calculated in [19] with jointly applying K-mean method and alignment of data.

	K-means + alignment	weighted K-means
Ω	35	35
S	30	30

Table 5.1: Comparison between joint K-means and alignment of data with weighted K-means

The groups found are exactly coincident. However we have, also, a quantitative measure of the difference between clusters and a criterion of decision. In the following three figures the function $b(s)$ is plotted for each coordinate.

As we have already noted, the graph of $b_x(s)$ is the most similar to the separate case. There are only little changes: the absolute maximum, from

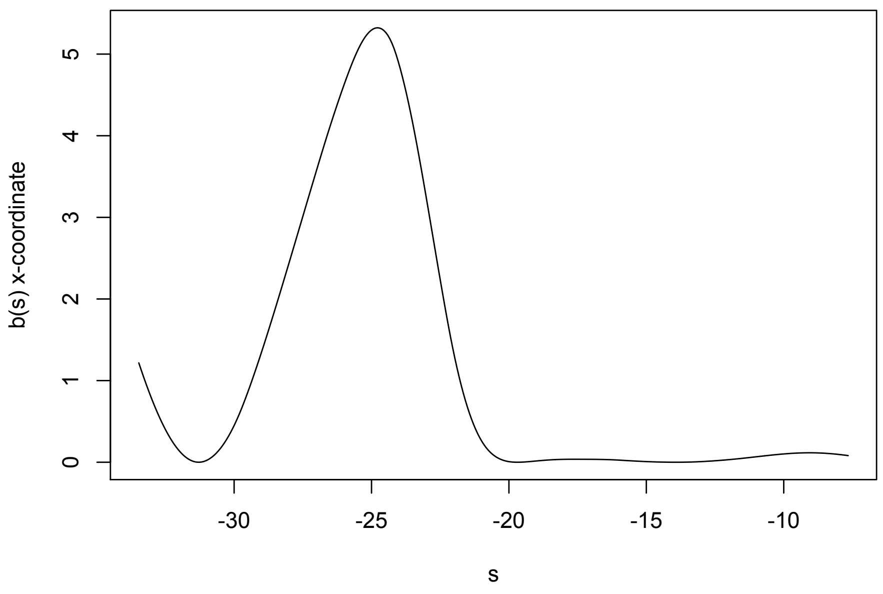

Figure 5.8: The computed function $b(s)$ for the x coordinate.

6, has moved to 5 and also the successive local maxima have lowered. The other two cases show the most evident modifications. In the separate analysis, $b_y(s)$ had two evident maxima, the first local and the second absolute, a flat part near zero with another local maximum. Now the absolute maximum has moved to be the first one and raised to 3. The remaining part of the function has completely modified, it is near zero, with two local maxima and it ends monotone increasingly.
The function $b_z(s)$ is the one which shows the most evident changes. The absolute maximum, located in the final part of the domain, is now found precisely at the first point of the interval. Even if, in this case, some importance is still given to the ending part of the domain, the most significative part has become the initial one.
As the final step of our analysis, we want to use our method, to perform *supervised learning*. The study conducted so far did not take into account any kind of knowledge about the data collected and we tried to cluster and then, *aposteriori*, we induced some useful informations about them. Now, we know, among the patients, who had an aneurysm, where it was located and if the aneurysm ruptured or not. For every datum, there are associated two more details: where the aneurysm was found, if the patient had any, and if

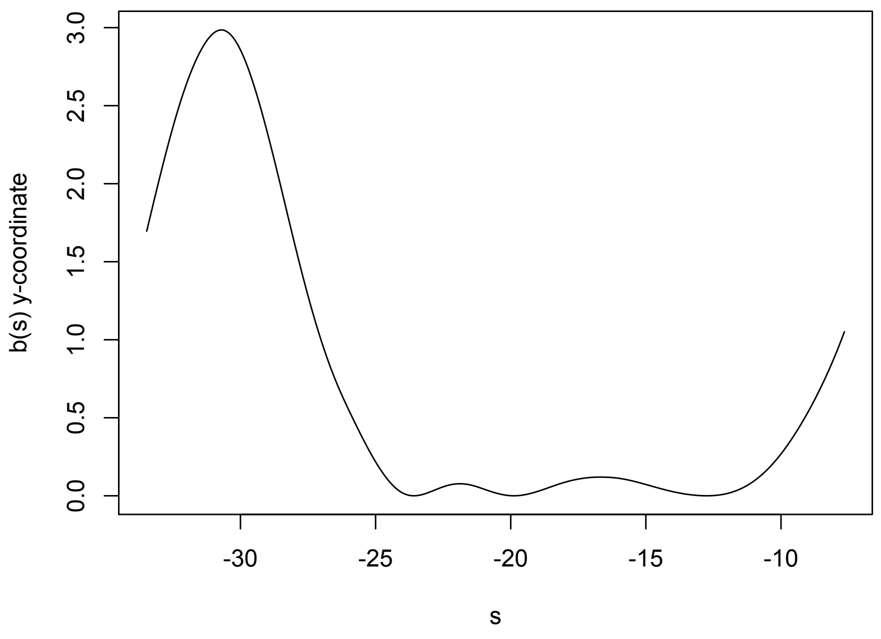

Figure 5.9: The computed functions $b(s)$ for the y coordinate.

the aneurysm ruptured. About the location of the anerysm, we can identify five classes of patients:

1. Patients without any aneurysm;
2. Patients with an aneurysm along the internal carotid far from the bifurcation;
3. Patients with an aneurysm along the internal carotid close to the bifurcation;
4. Patients with an aneurysm at the end of the internal carotid;
5. Patients with an aneurysm after the end of the internal carotid.

Of all these distinctions, classes 4. and 5. are the aneurysm commonly indicated as those happening on or after the Circle of Willis, while those in classes 2. and 3. are located in the internal carotid. Therefore, we want to find out if we can individuate particular aspects distinguishing all these possible groups. To begin with, we will consider three groups: patients without

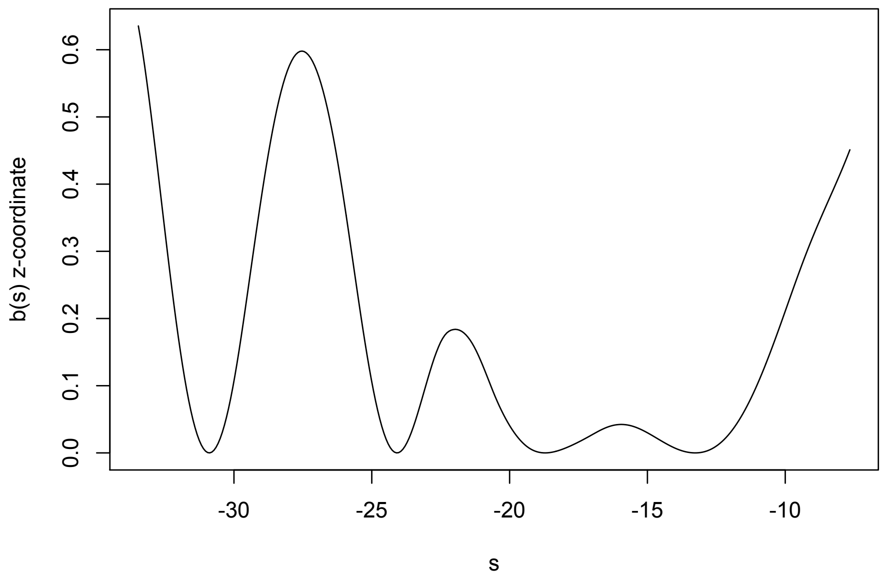

Figure 5.10: The computed functions $b(s)$ for the z coordinate.

any kind of aneurysm, those belonging to classes 2. and 3. and those belonging to the last two classes. In figure 5.11, we have shown $b_x(s), b_y(s)$ and $b_z(s)$ in this first case. The shape of $b_x(s)$ and $b_y(s)$ is similar to that found when we have jointly clustered. Even the abscissa of the absolute maximum is almost the same. Morever it seems, again, that the first half of the interval is what really diversify the clusters. If we look at the z coordinate, we can see that, even if some importance is given to the first half of the interval, what is most useful to the grouping is the final half of the set and with more and more significance as long as s augments. If we restrict our attention to the first half of the domain, we find the absolute maximum of $b_z(s)$ in correspondence of that of $b_y(s)$. The absolute minimum of $b_x(s)$ and $b_z(s)$ is found at values very close to one another: -21.208 and -21.437, respectively, while the absolute minimum of $b_y(s)$ is found at -13.8.
This kind of comparison, however, does not tell us, for example, which features mostly distinguish patients with an aneurysm along the internal carotid from those with an aneurysm on the Circle of Willis. Therefore, to perform this analysis, we remove, from our dataset those patients who do not suffer of any aneurysm and consider only those remaining. This will help us to seek for differences between the two classes of aneurysms. Only seven patients do

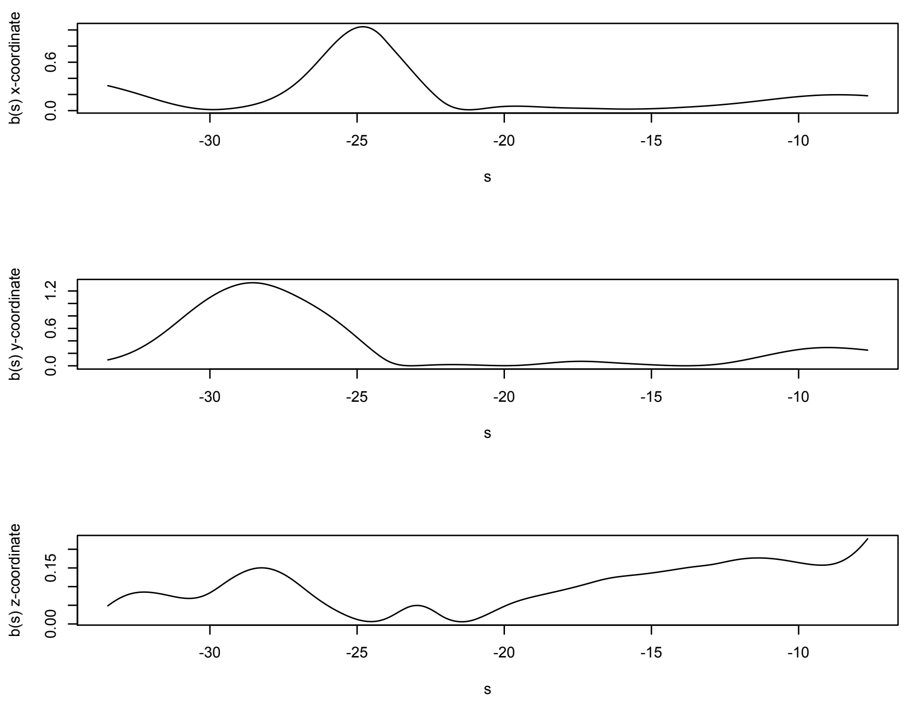

Figure 5.11: The computed vector $\mathbf{b}(s, \mathbf{f}, \mathbf{C})$ for each coordinate with three groups: $\{1\}$, $\{2, 3\}$ and $\{4, 5\}$.

not show any kind of aneurysm, so we do not consider them and remain with 58 curves. We have decided to divide the data into these groups: $\{2, 3\}$ and $\{4, 5\}$.

In figure 5.12, the components of the resulting vector $\mathbf{b}(s, \mathbf{f}, \mathbf{C})$ are reported. The most similar to the previous cases, here, is the component associated to the y coordinate. The other two coordinates, show different patterns. Surprisingly, even the x coordinate has a different shape if compared to the previous analyses. More relevance is assigned to the ending part of the domain. It is reasonable, however, since the neighborhood of the bifurcation could really contribute in determining the type of aneurysm. In particular, we can note two main areas that distinguish the aneurysms: from the beginning to $x = -30.422$ and from $x = -30.422$ to $x = -21.513$. The shape of the function in this second interval has modified lightly, with respect to the previous cases and we can also note that more importance is given to two more subsets: from -21.513 to -16.792 and, finally, from here to the end of the domain. The z coordinate changes its shape once again. Now it assumes a pattern similar to the y coordinate, but with inverted importance

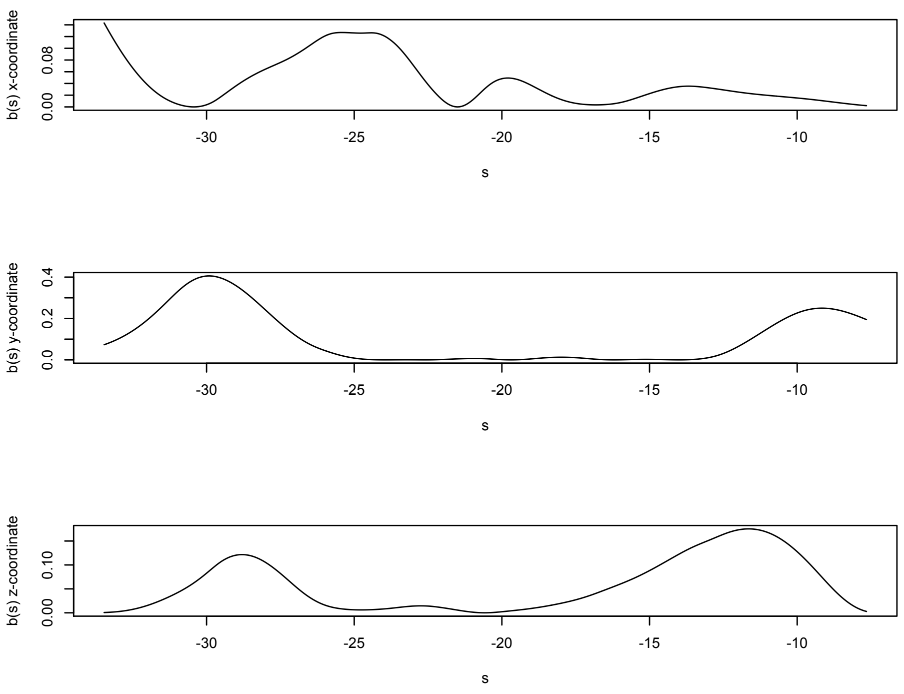

Figure 5.12: The computed vector $\mathbf{b}(s, \mathbf{f}, \mathbf{C})$ for each coordinate with groups: $\{2, 3\}$ and $\{4, 5\}$. In this way we can search for differences between those showing different types of aneurysms.

of maxima: the absolute maximum is the second one. This is a constant characteristics of the z coordinate: it always gives more relevance to the ending parts of the domains.
In particular, if we look at the two groups, we can note some differences in the shape and in the values of the centerlines of the carotids. Those with an aneurysm on the internal carotid have centerlines with higher values and a more pronounced maximum at about 30 mm from the bifurcation, on the x coordinate. The converse is true for the y coordinate: those with an aneurysm on the Circle of Willis have a more relevant maximum at about 20 mm from the bifurcation and, typically, higher values of their centerlines, especially in the final part of the domain. For the z coordinate, instead, the two groups show the same shape. The only aspect differentiating them is in the values of their centerlines; indeed the patients with an aneurysm on the Circle of Willis have higher values on the final part of the carotid, that is near the bifurcation.
The last analysis we can afford is the comparison of the centerlines between

the patients who do not have any kind of aneurysm and those who have it, with no regards of the type. In figure 5.13, we have reported the computed vector $\mathbf{b}(s, \mathbf{f}, \mathbf{C})$, in this last case. For what concerns the x coordinate, we

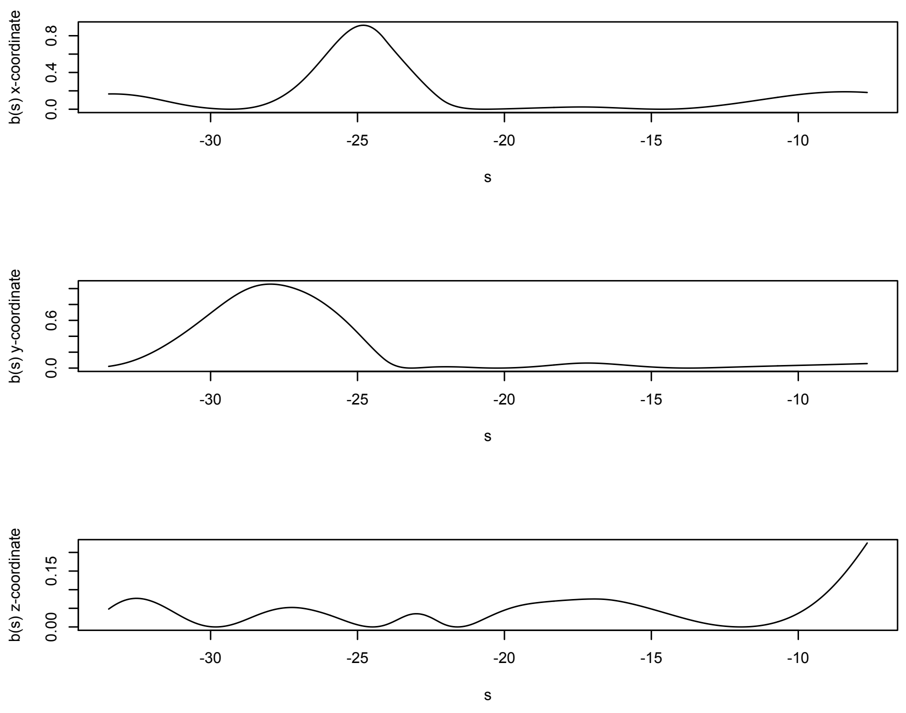

Figure 5.13: The computed vector $\mathbf{b}(s, \mathbf{f}, \mathbf{C})$ for each coordinate with groups: $\{1\}$ and $\{2, 3, 4, 5\}$. In this way we can search for differences between those patients showing an aneurysm and those without any kind of it.

can note a higher maximum and a steeper descent in the healthy patients than in the sick ones, but, we have to say that, in this second group, the differences can be mitigated by the presence of the two types of aneurysm. In the y coordinate, the two groups show the same shape, except that the values of the centerlines of the healthy patients seem to be more extreme: the initial minimum is lower and the final flat part is higher than in the second group. However, no evident differences of pattern in the two clusters are found. On the contrary, the shapes of the z coordinate are more distinct. The sick patients have a flat part near zero up to 25 mm from the bifurcation and then the centerline increase. The others, instead, have a little maximum between 30 and 25 mm; this causes the maximum between -30 and -25 in the plot of $b_z(s)$. The ending part also differentiate the two clusters: the centerlines of both groups increase and then decrease again, but the centerlines of the healthy patients decrease more rapidly and this explains the increasing final part of $b_z(s)$.

Appendix A

Codes

In this appendix, we report the codes used to implement the algorithms used. All the functions and the programs are written in `R` language. All the commands are commented, in order to explain their meaning.

This first function, called "UpdateWsfun" is that used to the update of the discretized weight function. The returned vector is as long as the number of the current iteration and it is constructed exactly as it is shown in Theorem 2.1. It needs, as arguments, the matrix containing the data, the vector containing the updated groups, the tuning parameter, the number of the current iteration and, finally, the vector containing the points of the domain.

```
UpdateWsfun <- function(x, Cs, L1bound, niter, xpoints){
# x is the matrix containing the data;
# Cs is the updated vector containing the groups;
# l1bound is the tuning parameter;
# niter is the number of the current iteration;
# xpoints is the vector containing the points of the domain.
m <- dim(x)[[2]]
len <- length(xpoints) # length of the interval
# measure of the subinterval generated at every step:
subint <- (xpoints[len] - xpoints[1])/niter
ws <- numeric(niter)
bcss <- numeric(niter)
# update of the distances between groups per subinterval:
b <- GetWCSS(x[,1:(floor(len/niter))], Cs)$bcss.perfeature
bcss[1] <- sum(b)
for(i in 2:(niter - 1)){
```

```
b <- GetWCSS(x[,(1 + (i-1)*floor(len/niter)):
     (i*floor(len/niter))], Cs)$bcss.perfeature
bcss[i] <- sum(b)
}
b <- GetWCSS(x[,(1 +
     (niter-1)*floor(len/niter)):m], Cs)$bcss.perfeature
bcss[niter] <- sum(b)
# calculation of the BCSS in every subinterval:
locbcss <- bcss*subint
sum.locbcss <- sum(locbcss) # global BCSS
# choice of the optimal alpha:
alpha <- min(L1bound/sum.locbcss, 1/(sqrt(sum(locbcss^2))))
# update of the weight "vector":
for(i in 1:niter){
ws[i] <- alpha * (subint) * bcss[i]
}
return(ws)
}
```

The second function we report is that which, at every iteration, reassigns the data to the correct clusters. Its arguments are: the data matrix, the number of clusters, the updated weight vector, the clusters calculated at the previous iteration, the current iteration and the maximum number of allowable iterations.

```
UpdateCsfun <- function(x, K, ws, Cs, niter, maxiter){
# x is the matrix containing the data;
# K is the number of clusters;
# ws is the updated weight vector;
# Cs is the clustering vector of the previous iteration;
# niter is the number of the current iteration;
# maxiter is the maximum allowed number of iteration.
n <- dim(x)[[1]]
m <- dim(x)[[2]]
# weighting of the data matrix (at the second iteration):

if((niter == 2) && (niter <= maxiter)){
X1 <- x[,1:(floor(m/2))]
x[,1:(floor(m/2))] <- sqrt(ws[1])*X1
X2 <- x[,(1 + floor(m/2)):m]
x[,(1 + floor(m/2)):m] <- sqrt(ws[2])*X2
}
# weighting of the data matrix:

if((niter > 2) && (niter <= maxiter)){
X <- x[,1:(floor(m/niter))]
x[,1:(floor(m/niter))] <- sqrt(ws[1])*X
for(j in 2:(niter-1)){
X <- x[,(1 + (j-1)*floor(m/niter)):((j)*floor(m/niter))]
x[,(1 + (j-1)*floor(m/niter)):((j)*floor(m/niter))]
       <- sqrt(ws[j])*X
}
X <- x[,(1 + (niter-1)*floor(m/niter)):m]
x[,(1 + (niter-1)*floor(m/niter)):m] <- sqrt(ws[niter])*X
  }
# clusters assignment:

nrowx <- nrow(x)
mus <- NULL
if(!is.null(Cs)){
  for(k in unique(Cs)){
if(sum(Cs==k)>1) mus <- rbind(mus, apply(x[Cs==k,],2,mean))
if(sum(Cs==k)==1) mus <- rbind(mus, x[Cs==k,])
    }
  }
  if(is.null(mus)){
km <- kmeans(x, centers=K, nstart=10)
}
else {
    distmat <- as.matrix
    (dist(rbind(x, mus)))[1:nrowx, (nrowx+1):(nrowx+K)]
    nearest <- apply(distmat, 1, which.min)
    if(length(unique(nearest))==K){
     km <- kmeans(x, centers=mus)
}
else {
     km <- kmeans(x, centers=K, nstart=10)
    }
  }
  return(km$cluster)
}
```

With these functions we have substantially built, respectively, the first and the second steps of the proof of Theorem 2.1. Now, we need a third function gathering the previous steps iteratively; i.e. it has to perform the third step of the proof of the same theorem. This is done with the following function. It receives as arguments: the data matrix, the number of clusters, the bound on the L^1 norm of the solution function, the points of the domain, the number of restarts of the algorithm and the maximum number of iterations.

```
FKMeansSparseCluster <- function(x, K, wbounds, xpoints,
              nstart, maxiter) {
# x is the data matrix;
# K is the number of clusters;
# wbounds is the bound on the L^1 norm of w(x)
   (it could be less or equal to the tuning parameter);
# xpoints is the vector of points of the domain;
# nstart is the number of restart of the algorithm;
# maxiter is the maximum number of iterations.
# nstart default = 20
# maxiter default = 10
# cycles of controls:
if(is.null(nstart)){
nstart <- 20
}
if(is.null(maxiter)){
maxiter <- 10
}
# measure of the domain
mis <- xpoints[length(xpoints)] - xpoints[1]
# maximum value of the tuning parameter:
L1bound <- sqrt(mis)
    if (is.null(wbounds)){
        wbounds <- L1bound
        }
    if (wbounds > L1bound){
        stop("wbounds should be less than or equal to L1bound")
        }
    l <- length(xpoints)
    wbounds <- c(wbounds)
    out <- list()
# initialization of the weight vector:
        WS <- rep(sqrt(1/mis), times = l)
        WS.old <- (1/2)*WS
        store.bcss.ws <- NULL
# initialization of the clustering vecor:
         C <- kmeans(data, K)$clusters
                niter <- 2
# iterative calculation of weight and clustering vectors:
while((sum(abs(WS - WS.old))/sum(abs(WS.old))) > 1e-04
         && (niter <= maxiter)){

WS.old <- WS
ws <- UpdateWsfun(x, C, wbounds, niter, xpoints)
Cs <- UpdateCsfun(x, K, ws, C, niter, maxiter)
C <- Cs
it <- niter
niter <- niter + 1
WS <- numeric(l)
WS[1:floor(l/it)] <- ws[1]
for(i in 2:(it-1)){
WS[(1 + (i-1)*floor(l/it)):((i)*floor(l/it))] <- ws[i]
}
WS[(1 + (it-1)*floor(l/it)):l] <- ws[it]
                 }
clusters.mean <- diag(0, K, l)
b <- (GetWCSS(x, Cs)$bcss.perfeature) * WS
wcss <- GetWCSS(x, Cs, WS)$wcss.perfeature
bcss <- GetWCSS(x, Cs)$bcss.perfeature
        for(j in 1:K){
         clusters.mean[j,] <- colMeans(x[which(C == j),])
         }
out <- list(ws = ws, Cs = C, iteration = it,
    fv = sum(b), wcss = wcss, tss = sum(b + wcss),
    clusters.mean = as.matrix(clusters.mean),
    bcss = bcss, WS = WS)
    return(out)
}
```

The objects returned by this fuction are: the final weight function (`ws`), the final cluster vector (`Cs`), the number of iterations (`iteration`), the optimal value of the objective functional (`fv`), the WCSS, the TSS and the BCSS (`wcss, tss, bcss`) and, finally, the clusters mean functions (`clusters.mean`).

Bibliography

[1] P. Billingsley (1999): *Convergence of Probability Measures*, New York, Wiley & Sons Inc.

[2] F. Ferraty, P. Vieu (2006): *Nonparametric Functional Data Analysis. Theory and Practice*, New York, Springer.

[3] J. H. Friedman, J. J. Meulman (2004): "Clustering Objects On a Subset of Attributes", *Journal of the Royal Statistical Society, Ser. B.*, 66, 815-849

[4] P. Halmos (1974): *Measure Theory*, New York, Springer.

[5] J. A. Hartigan (1975): *Clustering Algorithms*, New York, Wiley & Sons Inc.

[6] T. Hastie, R. Tibshirani, J. Friedman (2010): *The Elements of Statistical Learning. Data Mining, Inference and Prediction*, New York, Springer.

[7] E. Hewitt, K. Stromberg (1965): *Real and Abstract Analysis*, New York, Springer.

[8] J. Jacod, P. Protter (2004): *Probability Essential*, Berlin - Heidelberg, Springer.

[9] R. A. Johnson, D. W. Wichern (2007): *Applied Multivariate Statistical Analysis*, Upper Saddle River, NJ, Prentice Hall.

[10] W. Pan, X. Shen (2007): "Penalized Model-Based Clustering With Application to Variable Selection," *Journal of Machine Learning Research*, 8, 1145-1164.

[11] D. Pollard (1981): "Strong Consistency of K-Means Clustering", *The Annals of Statistics*, Vol. 9, No. 1, 135-140.

[12] D. Pollard (1982): "Quantization and the Method of K-Means", *IEEE TRANSACTION ON INFORMATION THEORY*, Vol. IT-28, No. 2, 199-205.

[13] D. Pollard (1984): *Convergence of Stochastic Processes*, New York, Springer.

[14] J. O. Ramsay, G. Hooker, S. Graves (2009): *Functional Data Analysis with R and MATLAB*, New York, Springer.

[15] J. O. Ramsay, B. W. Silverman (2002): *Applied Functional Data Analysis: Methods and Case Studies*, New York, Springer.

[16] J. O. Ramsay, B. W. Silverman (2005): *Functional Data Analysis*, New York, Springer.

[17] W. Rudin (1987): *Real and Complex Analysis*, Singapore, McGraw-Hill Book Co.

[18] L. M. Sangalli, P. Secchi, S. Vantini, A. Veneziani (2009): "A Case Study in Exploratory Fuctional Data Analysis: Geometrical Features of the Internal Carotid Artery", *Journal of the American Statistical Association*, vol 104, No. 485, 37-48.

[19] L. M. Sangalli, P. Secchi, S. Vantini, V. Vitelli (2010): "K-Mean Alignment for Curve Clustering", *Computational Statistics and Data Analysis*, 54, 1219-1233.

[20] A. N. Shiryaev (1996): *Probability*, New York, Springer.

[21] R. Tibshirani, G. Walther (2005): "Cluster Validation by Prediction Strength", *Journal of Computational and Graphical Statistics*, 14 (3), 511-528.

[22] R. Tibshirani, G. Walther, T. Hastie (2001): "Estimating the Number of Clusters in a Dataset via the Gap Statistic", *Journal of the Royal Statistical Society, Ser. B.*, 32 (2), 411-423.

[23] R. Tibshirani, D. Witten (2010): "A Framework for Feature Selection in Clustering", *Journal of American Statistical Association*, Vol. 105, No. 490, 713-726.

Acknowledgments

At the end of this work, I would really like to thank a huge number of people for having supported, helped or even entertained me through all these months.
First of all, I have to thank my advisor, prof. Piercesare Secchi, to which I owe so much. He has made me fond of this difficult discipline that is Statistics, but also he has stimulated me with challenges contributing to make this thesis a better work and to think about a great variety of subjects. His teachings goes beyond the mere academic environment and my regard of him is very high. There is another person to whom I owe a great debt: Valeria, for having very patiently followed and helped me with `R`, even if she was very busy with all her deadlines... and also, if I was able of getting through the algorithms and all the oddities of the computational parts of "our" work, a great contribution is up to her.
Then, I have to thank another long list of persons... Starting from my family my brother Paolo, my dad and my mom, for having sustained me all this time and without whom all this would have not been possible. I can't omit all my adventure-mates, that have shared with me all these years, the sad and the happier moments. I will always remember with a smile the laughes made during the lectures, the moments spent in explanations, those spent in studying and solving, sometimes quite unuseful, problems and, mainly, the parties after the exams and the "recreational" moments. I can't nominate the totality of people, but, among them, I want to remember some people in particular. Emma, because of her great patient shown with me, in bearing my oddities and ideas, for taking care of me like a mom and for her great work (doing also mine) in all the projects made together. Pamela and Andrea, for the plays made instead of studying and the "happy hours", Fez and Martino, for their continuously computer skills, Il Cosso, Stefano, Fabio, Il Portos, Alessio, Federica and Chiara and all the others, that I can't report, but whom I distinctly remember. And finally, I would like to thank Stefania, for she has teached me a lot...
Many other people, outside the university, contributed indirectly to this work.

Camilla, for sharing my same dreams and fears, Tommy, Paolo, Martina and Mario, funny travel-and-poor-figure-mates, but also all the friends of a lifetime: Fra, Lucia, Fede, Claudio, Anna, the-at-an-$\varepsilon/2$-from-being-my-favorite Gaia and many others... for they have constantly trusted in me. Elisabetta, because I promised her, for her "medical consultations" and all the members of the mighty Team Delfo, mainly Davide and his philosophycal discussions... and finally Gaia, about whom I have nothing to say, unless that "I'm learning all about my life, by looking through her eyes".