UNIVERSITÁ DEGLI STUDI DI NAPOLI FEDERICO II

Ph.D. Thesis in Statistics (XXVII ciclo)

CONTRIBUTIONS IN CLASSIFICATION:

VISUAL PRUNING FOR DECISION TREES,

P-SPLINE BASED CLUSTERING OF CORRELATED SERIES,

BOOSTED-ORIENTED PROBABILISTIC CLUSTERING OF SERIES.

Carmela Iorio

Department of Economics and Statistics

April, 2015

Promotor:
Prof. dr. Roberta Siciliano Universitá degli studi di Napoli Federico II
Co-Promotor:
Prof. dr. Massimo Aria Universitá degli studi di Napoli Federico II

External Supervisor:
Prof. dr. Mark de Rooij Leiden University
Prof. dr. José Fernando Vera Granada University

A mia madre

CONTENTS

Contents i

1 Introduction 1

2 Preliminary tools and concepts 5

2.1 Supervised classification 5

2.1.1 Splitting criteria 7

2.1.2 Two Stage splitting criterion 10

2.1.3 FAST splitting criterion 12

2.1.4 Stopping rules and assignment of the response classes/values to the terminal nodes 14

2.1.5 Pruning 15

2.2 Unsupervised classification 18

2.2.1 Partitioning methods 20

2.2.2 Hierarchical methods 24

2.2.3 Density-based methods 25

2.2.4 Grid-based methods 27

2.2.5 Model-based methods 29

2.2.6 Dissimilarity measure 31

2.3 Penalized Spline 41

2.3.1 B-Spline 42

2.3.2 Penalized regression 44

2.3.3 Smoothing parameter selection 47

3 Visual Pruning for Decision Trees 55

3.1 Tree based recursive partitioning method and tree model selection: an overview 55

3.2 Decision tree visualization 57

3.3 Visual tree model selection 59
3.4 An application of visual tree 61
3.5 Concluding remarks 62

4 P-Spline based Clustering of Correlated Series 65
4.1 Introduction 65
4.2 Time series clustering 68
4.2.1 Notations and definitions 68
4.2.2 Clustering algorithms 69
4.2.3 Distance measure 72
4.3 The key idea 74
4.3.1 Series parametrization 74
4.3.2 Smoothing parameter selection 77
4.3.3 Clustering procedure 78
4.4 Simulation study 78
4.5 Clustering of financial time series 82
4.6 Concluding remarks 88

5 Boosted-Oriented Probabilistic Clustering of Series 91
5.1 Introduction 91
5.2 Boosted-oriented probabilistic clustering of time series 96
5.2.1 The key idea 96
5.2.2 A brief introduction to P-splines 97
5.2.3 PD clustering approach 99
5.2.4 The algorithm 99
5.3 Experimental evaluation 102
5.3.1 Simulated data 103
5.3.2 Synthetic data set 105
5.4 A real data example 109
5.5 Concluding remarks 111

6 Conclusions and perspective 113

Bibliography 117

Acknowledgments 127

1 INTRODUCTION

The aim of this work is to provide a methodological contribution and application on both supervised and unsupervised classification.
This work consists of three papers written during my Ph.D. period. The thesis consists of five chapters. In **chapter 2** the basic building blocks of our works are introduced. In particular we briefly recall the concepts of classification (supervised and unsupervised) and penalized spline.

In **chapter 3** we present a paper whose idea was presented at Cladag 2013 Symposium. Within the framework of recursive partitioning algorithms by tree-based methods (Breiman et al., 1984), this paper provides a contribution on both the visual representation of the data partition in a geometrical space and the selection of the decision tree. In our visual approach the identification of both the best tree and of weakest links is immediately evaluable by the graphical analysis of the tree structure without considering the pruning sequence. The results in terms of error rate are really similar to the ones returned by the Classification And Regression Trees procedure, showing how this new way to select the best tree is a valid alternative to the well known cost-complexity pruning

In **chapter 4** we present a paper on parsimonious clustering of correlated series whose idea was presented at ERCIM 2014 Symposium. Clustering of time series has become an important topic, motivated by the increased interest in these type of data. Most of the time, these procedures do not facilitate the removal of noise from data, have difficulties handling time series with unequal length and require a preprocessing step of the data considered, i.e. by modeling each series with an appropriate model for time series. In this work we propose a new clustering data (time) series studying, which can be considered as model and feature based approach (Liao, 2005). The proposal consists of model each series by penalized spline (P-spline) smoothers

(Eilers and Marx, 1996) and to perform a clustering directly on spline coefficients. The P-spline coefficients are close to the fitted curve and present the skeleton of the fit (Eilers and Marx, 2010). Series with different length can be handled by P-spline due to the extrapolation properties. Thus, summarizing each series by coefficients reduces the dimensionality of the problem, improving significantly computation time without reduction in performance of clustering procedure. To select the smoothing parameter we adopt a V-curve procedure proposed by Frasso and Eilers (2015). The performance of our clustering approach is evaluated analyzing a simulated data set as described in Coffey et al. (2014). An application of our proposal on financial time series is also presented.

In **Chapter 5** we present a paper that proposes a fuzzy clustering algorithm that is independent from the choice of the fuzzifier. It comes from two approaches, theoretically motivated for respectively unsupervised and supervised classification cases. The first is the Probabilistic Distance (PD) clustering procedure defined by Ben-Israel and Iyigun (2008). The second is the well known Boosting philosophy. From the PD approach we took the idea of determining the probabilities of each series to any of the k clusters. As this probability is unequivocally related to the distance of each series from the cluster centers, there are no degrees of freedom in determine the membership matrix. From the Boosting approach (Freund and Schapire, 1997) we took the idea of weighting each series according to some measure of badness of fit in order to define an unsupervised learning process based on a weighted re-sampling procedure. Our idea is to adapt the boosting philosophy to unsupervised learning problems, specially to non hierarchical cluster analysis. In such a case there not exists a target variable, but as the goal is to assign each instance (i.e. a series) of a data set to a cluster, we have a target instance. The representative instance of a given cluster (i.e. the center of a cluster) can be assumed as a target instance, a loss function to be minimized can be assumed as a synthetic index of the global performance, the probability of each series to belong to a given cluster can be assumed as the individual contribution of a given instance to the overall solution. In contrast to the boosting approach, the higher is the probability of a given series to be member of a given cluster, the higher is the weight of that instance in the re-sampling process. As a learner we use a P-spline smoother (Eilers and Marx,

1996). To define the probabilities of each series to belong to a given cluster we use the PD clustering approach. This approach allows us to define a suitable loss function and, at the same time, to propose a fuzzy clustering procedure that does not depend on the definition of a fuzzifier parameter. The global performance of the proposed method is investigated by three experiments (one of them on simulated data and the remaining two on data sets known in literature) evaluated by using a fuzzy variant of the Rand Index (Hullermeier et al., 2012).

Chapter 6 concludes the thesis.

2 PRELIMINARY TOOLS AND CONCEPTS

This chapter introduces some preliminary concepts that was used in the following chapters. The concept of classification (supervised and unsupervised) and Penalized Spline are presented.
***Keywords**: Classification and Regression Tree, Cluster analysis, Dissimilarity measure, B-splines, P-splines*

2.1 *Supervised classification*

Classification is a supervised learning problem of assigning an object to one of several pre-defined categories based upon the attributes of the object. Recursive partitioning tree procedures have been the subject of extensive research in the past. Specially tree-based methods have been proposed for both prediction and exploratory purposes.

Binary segmentation procedure consists of a recursive binary partition of a set of objects described by some explanatory variables (either numerical or and categorical) and a response variable. In the following, CART procedure (Breiman et al., 1984) is followed.

The data are partitioned by choosing at each step a variable and a cut point along it according to a goodness of split measure which allows to select that variable and cut point that generates the most homogeneous subgroups respect to the response variable. The procedure results in a nice and powerful graphical representation known as decision tree which express the sequential grouping process. Because of the evident analogy with the graph theory, a subset of observations is called node and nodes that are not split are called terminal nodes or leaves (see figure 2.1).

Each node has a number such that generic node t generates the left node $2t$ and the right node $(2t+1)$. This approach was proposed by authors of statis-

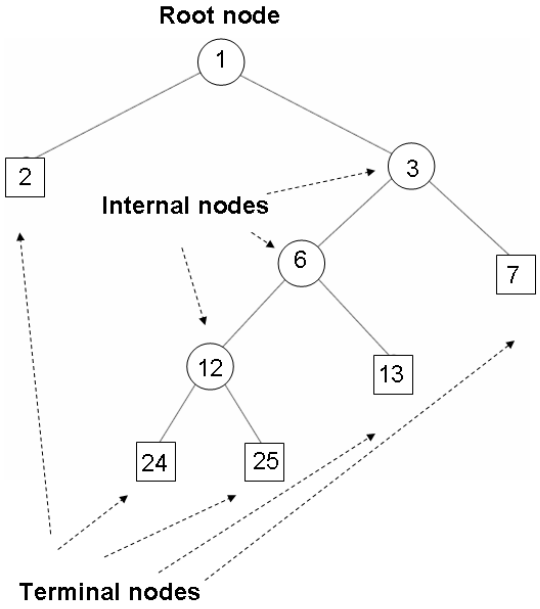

Figure 2.1: Tree-based structure

tical software SPAD (Cisia Institute, France). In this way, it is always possible to recognize the position of each node given its number deriving the path from the node to the root node and vice versa. In example, in the above figure, the node 6 is the left node of its parent node 3 which is the right node of its parent node 1 (the root node).

Once the tree is built, a response value or a class label is assigned to each terminal node. According to its nature (categorical or numerical) a distinction is made between Classification Tree (for the categorical response case) and Regression Tree (for the numerical response case). In the former case, when the response variable takes value in a set of previously defined classes, the node is assigned to the class which presents the highest proportion of observations (by voting). In the latter case, the value assigned to cases in a given terminal node is the average of the response variable values associated with the cases belonging to the given node. In both cases this assignment is probabilistic, in the sense that a measure of error is associated to it.

The main aim of the procedure is to define a classification/prediction

rule on the basis of a learning set (also called training set), for which the values of a response variable Y, and of a set of K explanatory variables $(X_1, \ldots, X_k, \ldots, X_K)$ (either numerical or/and categorical) have been recorded.
The recursive partitioning, following a *divide et impera* procedure, continues partitioning nodes until all leaves contain a single case or cases either belonging to the same class or presenting the same response value. This leads to overlarge trees with many rules which are hard to understand and over-fit the data.
In practice, when performing binary segmentation one has to look for a compromise that allow for the trade-off between the exploratory and the confirmatory purposes of the tree structures methodology. A distinction is made between the two problems involved in investigating the data sets: that is, whether to explore dependency, or to predict and decide about future responses on the basis of the selected predictors.
Explanation can be obtained by performing a segmentation of the objects until a given stopping rule defines the final partition of the objects to interpret.
Confirmation requires the definition of decision rules, usually obtained by performing a pruning procedure immediately soon after a segmentation procedure. Therefore, a further step, tree pruning, is usually carried out to avoid over-fitting and improve the understandability of the tree by retrospectively pruning some of the branches.
Summarizing, tree based methods involve the following steps:

- the definition of a splitting criterion;
- the definition of a stopping rule;
- the definition of the response classes/values to the terminal nodes;
- tree pruning, aimed at simplifying the tree structure, and tree selection, aimed at selecting the final decision tree for decisional purposes

2.1.1 *Splitting criteria*

Let (Y, X) be a multivariate random variable where X is a set of K categorical or numerical predictors $(X_1, \ldots, X_k, \ldots, X_K)$ and Y is the response variable. The first problem in tree building is how to determine the binary

splits of the data into smaller and smaller subgroups. Since the partitioning is just two branches, splitting variables need to be created from the original explanatory variables. Accordingly, data partitioning is based on a set of Q binary questions of the form:

$$\text{is } X_k \in A?$$

so that, if X_k is categorical, A includes subsets of levels, while if X_j is numeric, Q includes all questions of the form:

$$\text{is } X_k \leq \text{c}?$$

for all c ranging over the domain of X_k. For example, if $K = 3$, X_1, X_2 are numerical and $X_3 \in a_1, a_2, a_3$, Q includes all questions of the form:

$$X_1 \leq 3.5?$$

$$X_2 \leq 5?$$

$$X_3 \in a_1, a_3?$$

The set of possible splitting variables is finite and the number of splitting variables that can be created from a given explanatory variable depends on the type of variable, i.e., according to its measurement. Table 2.1 reports the number of splitting variables that can be generated by any type of explanatory variable according to its scale of measurement.

Explanatory variable	Categories	Number of splitting variables
Numeric	N	$N-1$
Binary	2	1
Ordered	M	$M-1$
Unordered	M	$2^{M-1}-1$

Table 2.1: Origin of the splitting variables

The algorithm generates all the possible splitting variables and searches through them one by one. Unordered variables are the most difficult to deal with because they can generate a very large number of splitting variables even for a small value of M. Once that the set of binary questions has been created, some criterion which guides the search in order to choose the best

one to split the node is needed. As said before, the key idea is to split each node so that each descendant is more homogeneous than the data in the parent node. To reach this aim, we need a measure of homogeneity to be evaluated by means of a splitting criterion.
In the CART methodology the idea of finding splits which generate more homogeneous descendant nodes has been implemented for classification trees by introducing the so-called impurity function.
Let $p(j|t) \geq 0$ be the proportions of cases in node t belonging to class j with $\sum_{j=1}^{J} p(j|t) = 1$.
An impurity function ϕ is a function of the set of all J-tuples of numbers $p(j|t)$ with the properties (Breiman et al., 1984, p. 24):

1. ϕ is maximum only at the point $\{1/J, 1/J, \ldots, 1/J\}$;
2. ϕ achieves its minimum only at the points $(1, 0, \ldots, 0), (0, 1, \ldots, 0), (0, 0, \ldots, 1)$;
3. ϕ is a symmetric function of $p(j|t)$.

There are several impurity functions satisfying these three properties. The most common are:

1. the error rate, or the misclassification ratio:

$$i(t) = 1 - max_j p(j|t)$$

2. the Gini diversity index

$$i(t) = 1 - sum_j p(j|t)^2$$

3. the entropy measure

$$i(t) = -sum_j (pj|t) log(j|t)$$

Talking about regression trees, the splitting criterion is based on the search of that split that generates the most different descendant nodes in terms of mean value of the response variable:

$$i\,(t) = \frac{1}{N} \sum\nolimits_{x_n \in t} (y_n - \bar{y}_t)^2 \tag{2.1}$$

(2.1) can be meant as the total sum of squares (TSS), divided by N where N is the sample size, $\bar{y}_t = \frac{1}{N_t} \sum_{x_n \in t} y_n$, N_t is the total number of cases in within node t where the sum is over all y_n such that $x_n \in t$. If s is a proposed split of a generic node t into two offspring t_l and t_r , and p_l and p_r are the proportions of objects in node t which the split s puts into nodes t_l and t_r respectively, then a measure of the change in impurity which would be produced by split s of node t is given by:

$$\Delta i(t, s) = i(t) - [i(t_l)p_{t_l} + i(t_r)p_{t_r}] \tag{2.2}$$

Δi, called decrease in impurity, can be used as splitting criterion: a high value of (2.2) means that a proposed split is a good one. At a given node t, a split s^* maximizing equation 2.2 is optimal and used for generate two descendants t_l and t_r . Let $\tilde{T}$ be the set of all terminal nodes of the tree T: the total impurity of any tree T is defined as

$$I\,(T) = \sum_{t \in \tilde{T}} i\,(t)\,p(t)$$

To proceed with tree growing, CART procedure must compute the decrease in impurity associated to each possible split generated by each variable. For example, suppose to have a binary response variable and a set of six predictors as defined in table 2.2. In the root node the number of decreases in

Variable	Nature	Categories	Number of split
X_1	*Nominal*	6	$2^5 - 1 = 31$
X_2	*Nominal*	7	$2^6 - 1 = 63$
X_3	*Ordinal*	3	$3 - 1 = 2$
X_4	*Binary*	2	1
X_5	*Ordinal*	5	$5 - 1 = 4$
X_6	*Ordinal*	4	$4 - 1 = 3$

Table 2.2: Example of generation of splits according to the nature of the predictors

impurity to be computed is $31 + 63 + 2 + 1 + 4 + 3 = 104$. Therefore, computational cost of CART is really high, because this procedure must be repeated until a stooping rule in tree-building occurs.

2.1.2 *Two Stage splitting criterion*

Mola and Siciliano (1992, 1994) have proposed a Two-Stage splitting criterion to choose the best split. This approach relies on the assumption that a predic-

tor X_k is not merely used as a generator of partitions but it plays also a global role in the analysis. In the first stage, a variable selection criterion is applied to find one or more predictors that are the most predictive for the response variable. On the basis of the set of partitions generated by the selected predictor(s), a partitioning criterion is considered in the second stage in order to find the best partition of the objects at a given node. The criteria to be used in the two stages depends on the nature of the variables, the tool of interpretation and the desired description in the final output. The partitioning algorithm takes account of the computational cost induced by the recursive nature of the procedure and the number of possible partitions at each node of the tree. Further developments of the Two Stage procedure face the computational efficiency problem. In fact, from a computational point of view, the growing procedure is crucial when dealing with very large data sets or when dealing with ensemble methods. At any node t the two stages can be defined as:

- **global selection**; one or more predictors are chosen as the most predictive for the response variable according to a given criterion; the selected predictors are used to generate the set of partitions or splits. In this stage an index needs to be defined to evaluate the Global Impurity Proportional Reduction (Global IPR) of the response variable Y at node t, due to the predictor X;

- **local selection**; the best partition is selected as the most predictive and discriminatory for the subgroups according to a given rule. In this stage one has to define an index as the Local Impurity Proportional Reduction (Local IPR) of the response Y due to the partition p generated by the predictor X

For classification trees the Global IPR is defined as τ index of Goodman and Kruskal

$$\tau_t(Y|X) = \frac{\sum_i \sum_j p_t^2(j|i)p_t(i) - \sum_j p_t^2(j)}{1 - \sum_j p_t^2(j)}, \tag{2.3}$$

where $p_t(i)$, for $i = 1, \ldots, I$, is the proportion of cases in node t that have category i of X, and $P_t(j|i)$, for $j = 1, \ldots, J$, is the proportion of cases in the

node t belonging to class j of Y given the i^{th} category of X. Note that the denominator in equation (2.3) is the Gini diversity index.
For regression trees, Global IPR can be defined as the Pearson's squared correlation η^2:

$$\eta^2_{Y|X}(t) = \frac{BSS_{Y|X}(t)}{TSS_Y(t)}, \tag{2.4}$$

where SST is the total sum of squares of the numerical response variable Y and BSS is the between group sum of squares due to the predictor X.
In a similar way, the Local IPR for both classification and regression trees are defined as in equation (2.3) and (2.4), with the difference that in these cases indexes are computed between the response variable Y and the set of split s generated by the global IPR functions.
More precisely, for classification trees, at each node t of the splitting procedure, a split s of the I categories of X into two sub-groups (e.g. $i \in l$ or $i \in r$), leads to the definition of a splitting variable X_s with two categories denoted by l and r. Local IPR is defined as

$$\tau_t(Y|s) = \frac{\sum_j p^2_l(j|tl)p_{tl} + \sum_j p^2_{tr}(j|r)p_{tr} - \sum_j p^2_t(j)}{1 - \sum_j p^2_t(j)} \tag{2.5}$$

whereas for regression trees it is the following:

$$\eta^2_{Y|s}(t) = \frac{BSS_{Y|s}(t)}{TSS_Y(t)}. \tag{2.6}$$

Two stage splitting criterion works as follow:

1. select the best predictor $X^*(t)$ at t node by maximizing equation (2.3) or 2.4 for classification or regression problems respectively:

2. select the best split $s^*(t)$ at node t by maximizing equation (2.5) or (2.6) for all splits of $X^*(t)$ for classification or regression trees respectively

2.1.3 *FAST splitting criterion*

Fast Splitting for Splitting Tree (FAST algorithm) proposed by Mola and Siciliano (1998) provides a faster method to find the best split at each node when using CART methodology. As discussed in above section, when applying the

two-stage criterion the best predictor could be found minimizing the Global Impurity Proportional Reduction factor due to any predictor X, then the Local Impurity Proportional Reduction factor determines the split with respect to all partitions derived from the best predictor.
Main issue of FAST is that the measure of Global IPR measure satisfies the following property:

$$\gamma(Y|X) \geq \gamma(Y|s), \tag{2.7}$$

in which γ is the generic Global IPR measure, and s is the set of split generated by X variable.

FAST algorithm consists in two step:

- computing Global IPR measure as in equation (2.3) or (2.4) for all variables belonging to the predictor matrix X and sort in decreasing order these measures;
- computing Local IPR measure as in equation (2.5) or (2.6) for the first previously ordered variable with maximum Global IPR. If Local IPR of this variable is higher than Global IPR of the second ordered X variable, stop the procedure, otherwise continue until inequality is satisfied.

The computational cost of FAST algorithm is really lower than the one of CART procedure, with the advantage that the final trees are exactly the same. In the example showed at the end of section 1.2.1 in the table 2.2, there is a set of six predictors, 2 nominal with 6 and 7 categories respectively, 3 ordinal with respectively 3, 5 and 4 categories and one binary variable. It was shown that CART procedure for each variable must examine each possible split to decide which one is the best. Considering the root node, CART technique has to compute $(25-1)+(26-1)+2+1+4+3=104$ splits.
FAST algorithm computes at the beginning only six Global IPR measure (in this case there are only six predictors) and then only the local impurity reduction factor until inequality of the second step of the procedure is satisfied. In this small example, the number of computations made is $6+(25-1)=30$ (it is assumed that Global IPR measure relative to the second-best predictor is lower than the local impurity reduction factor obtained by the second one).

The computational advantage of using FAST instead of CART is clear: one obtains the same tree-based structure with a great gain in terms of computational cost.

2.1.4 *Stopping rules and assignment of the response classes/values to the terminal nodes*

Once the rules for growing the tree has been defined, another set of rules to stop the building of the structure are needed. There is no unique rule to define the stopping of the procedure, but there are several rules used according the discretion of the researcher. Tree growing can be arrested considering a suitable combination of the following conditions:

- *Bound on the decrease in impurity*.
  A node is terminal if the reduction in impurity due to the further partition of the node is lower than a fixed threshold; a node should be splitted if their contribution to the total impurity reduction is significant;

- *Bound on the number of observations*.
  In general, can be useless to continue splitting nodes with a few number of individuals: sample size within-node should be *rational*;

- *Tree size*.
  A further condition could be based on either the total number of terminal nodes or the number of levels of the tree to limit its expansion.

Once the tree has been built, terminal nodes must be associated with a response.

In the case of classification trees the assignment of a response to each terminal node is based on a simple majority rule. Specifically, node t is assigned to class j^* if the highest proportions of objects in node t belong to class j^* so that:

$$p(j^*|t) = \max_{j \in C}[p(j|t)]$$

In the case the response variable is numeric the response values for the object falling into a given terminal node t can be summarized by means of a

synthetic measure; in general this is simply given by the mean, so that $\bar{Y}_t$ is assigned to node t where:

$$\bar{y}_t = \frac{1}{n(t)} \sum_{x_n \in t} y_{i_t}$$

2.1.5 *Pruning*

Exploratory trees can be used to investigate the structure of data but they cannot be used in a straightforward way for induction purposes. For inductive purposes the aim is to establish how large should be the tree. A very large tree might over-fit the data, while a small tree may not be able to capture the important structure. Tree size is a tuning parameter governing the complexity of the model, and the optimal tree size should be adaptively chosen from the data. To choose the *honest* tree in terms of its size, Breiman et al. (1984) defined the *minimal cost-complexity pruning*. Before proceeding with pruning description, the definition of an error measure of a tree structure is necessary.

- For classification trees, the error at the generic node t is defined as

$$r(t) = \frac{1}{n_t} \sum_{i=1}^{n_t} (\widehat{Y}_t \neq Y_i),$$

  where n_t is the size at t^{th} node, $\widehat{Y}_t$ is the classification returned by the tree in the same node. The error rate of the overall tree is defined as

$$R(T) = \sum_{h \in H_T} r(t)p(t),$$

  where H_T is the set of all terminal nodes of the tree T, and $p(t)$ is the proportion of cases falling into the t^{th} terminal node.

- For regression trees the error rate is defined exactly as in equation (2.1), that is as the sum of TSS in the t^{th} node divided by the total sample size, whereas the prediction error of overall tree is defined as:

$$RR(T) = \frac{R(T)}{R(t_1)},$$

  where $R(t_1)$ is the error in the root node.

Pruning procedure works as follow: Let T_{max} be the maximum tree, let $\left|\tilde{T}\right|$ denote the set of all terminal nodes of T_{max}, that is its complexity. The cost-complexity measure is defined as:

$$R_\alpha(T) = R(T) + \alpha|\tilde{T}|,$$

where α is a non negative complexity parameter which governs the trade-off between tree size and its goodness of fit to the data Hastie et al. (2009).

The idea is, for each α, find the subtree $T^*_\alpha \supseteq T_{max}$ to minimize $R_\alpha(T)$. When $\alpha = 0$ the solution is the full tree T_{max}, and the more α increases the more the size of the tree decreases.

The pruning procedure is the same for both classification and regression cases, so the attention can be focused on the classification problem without loss of generality. The cost complexity measure is defined for any internal node t and the branch T_t rooted at t as:

$$R_\alpha(t) = r(t)p(t) + \alpha,$$

$$R_\alpha(T_t) = \sum_{h \in H_t} r(h)p(h) + \alpha|\tilde{T}_t|,$$

where $R_{(t)}$ is the re-substitution error at node t, $p(t) = \frac{n(t)}{N}$ is the weight of node t given by the proportion of training cases falling in it and H_t is the set of terminal nodes of the branch having cardinality $|\tilde{T}|$. The branch T_t will be kept as long as:

$$R_\alpha(t) > R_\alpha(T_t).$$

It means that the branch T_t will be kept as the error complexity of node t being higher than the error complexity of its branch. As α increases the two measures tends to become equal, this occurs for a critical value of α that can be found solving the above inequality:

$$\alpha = \frac{R(t) - R(T_t)}{|\tilde{T}_t| - 1}.$$

Thus α represents for any internal node t the cost due to the removal of any terminal node of the branch.

The pruning process produces a finite sequences of subtrees $\Omega = T_1 \subset T_2 \subset \ldots \subset T_{max}$, where T_1 is a tree constituted only by the root node. It can be

proved (Breiman et al., 1984) that the minimal cost-complexity pruning procedure produces the subtrees with the minimum error rate given the number of its terminal nodes. In other words, if T_α has five terminal nodes, there is no other subtree $T_s \subseteq T_{max}$ having five terminal nodes with smaller error (Breiman et al., 1984, p. 71).
To validate a tree-based structure one has to consider its accuracy: the misclassification ratio or the prediction error. In both classification and regression cases an estimation of the error rate is needed. There are three possible ways to estimate it:

- *Re-substitution estimate*
  Resubstitution estimate is computed by using the same dataset used to build the tree. It is an optimistic estimate, therefore it is not used.

- *Test set estimate*
  If the sample size is sufficiently large, data can be randomly splitted into two sub-samples (training sample and test sample). Then training sample is used to grow the tree-based structure and the test set is used to validate it.

- *Cross validation estimate*
  When sample size is not sufficiently large to be splitted into two subsamples, one can use the cross-validation estimate. Data set is splitted into V sub-samples approximately of the same size, then V trees are built using the V^{th} sub-sample as test set and the other $V-1$ as training set. By averaging over the V test set estimates, finally the cross-validation estimate of the error rate is achieved.

A single final tree is then selected either as the one producing the smallest error estimate on an independent test set ($0-SErule$) or the one which error estimate is within one standard error of the minimum ($1-SErule$). Denoting by $R^{ts}(T)$ the test set error estimate associated with a generic tree T in the sequence Ω, according to the $0-SE$ rule the tree T^* will be selected if:

$$R^{ts}(T^*)) = \min_{T \in S} R^{ts}(T)$$

whereas, if $1-SE$ rule is employed tree T^{**} will be selected if:

$$R^{ts}(T^{**}) \leq [R^{ts}(T^* \pm SE(R^{ts}(T^*)))$$

2.2 *Unsupervised classification*

Cluster analysis has become an increasingly important topic in recent years, caused by its application in several fields of science as engineering, computer science, medical science, social science and economics.
Clustering is an unsupervised learning problem that groups objects based upon a distance or a similarity. Each group is known as a cluster.
The goal is to separate a finite, unlabeled data set into a finite and discrete set of natural hidden data structures, rather than to provide an accurate characterization of unobserved samples generated from the same probability distribution.
Many clustering algorithms have been introduced in the literature. Since clusters can be formally seen as subsets of the data set, one possible classification of clustering methods can be done according to whether the subsets are fuzzy (soft) or crisp (hard).
Let $\mathbf{Z}$ be a data matrix n-by-p to be clustered where n represents the objects and p the variables. Let i be the subscript for objects. Let C be an integer, with $2 \leq c < C$. Crisp clustering methods are based on classical set theory and restrict that each object of data set belong to exactly one cluster. In other words, this means partitioning the data $\mathbf{Z}$ into a specified number of mutually exclusive clusters $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_C$.
A hard partition of $\mathbf{Z}$ can be defined as a family of subsets $\mathcal{A}_c$ that satisfies the following properties (Bezdek, 1981):

$$\bigcup_{c=1}^{C} \mathcal{A}_c = \mathbf{Z},$$

$$\mathcal{A}_c \cap \mathcal{A}_t = \emptyset,\ c \neq t$$

$$\emptyset \subset \mathcal{A}_c \subset \mathbf{Z},\ 1 \leq c \leq C.$$

The first condition indicates that the union subsets $\mathcal{A}_c$ contains all the data. The latter conditions indicate that the subset are disjoint and none of them is empty nor contains data in $\mathbf{Z}$. Let μ_{ic} be the membership function and let $\mathbf{U} = [\mu_{ic}]$ be the $n \times C$ partition matrix. The elements of $\mathbf{U}$ must satisfy the following conditions:

$$\mu_{ic} \in \{0, 1\},\ 1 \leq c \leq C,\ 1 \leq i \leq n;$$

$$\sum_{c=1}^{C} \mu_{ic} = 1;$$

$$0 < \sum_{i=1}^{n} \mu_{ic} < n.$$

The c^{th} column of $\mathbf{U}$ contains the value of μ_{ic} of the c^{th} subset $\mathcal{A}_c$ of $\mathbf{Z}$. Following Bezdek (1981) the hard partionining space is thus defined by:

$$M_h = \{\mathbf{U} \in \mathbb{R}^{C \times n} | \mu_{ic} \in \{0,1\}, \forall c, i; \sum_{c=1}^{C} \mu_{ic} = 1, \forall i, 0 < \sum_{i=1}^{n} \mu_{ic} < n, \forall c\}.$$

M_h is the space of all possible hard partition matrices for $\mathbf{Z}$.
Generalizing the crisp partition, $\mathbf{U}$ is a fuzzy partition of $\mathbf{Z}$ with elements μ_{ic} of the partition matrix bearing real values in $[0,1]$ (Kaufman and Rousseeuw, 2009). The idea of fuzzy sets was conceived by Zadeh (1965). Fuzzy clustering methods allow the objects to belong to several clusters simultaneously, with different degrees of membership. In contrast to hard clustering, each object have a membership value in each cluster. The larger the value of the membership value for a given object with respect to a cluster, the larger the probability of that object to be assigned to that cluster.
Similarly to crisping conditions, the conditions for a fuzzy partitions (Ruspini, 1970) are:

$$\mu_{ic} \in [0,1],\ 1 \leq c \leq C,\ 1 \leq i \leq n;$$

$$\sum_{c=1}^{C} \mu_{ic} = 1,$$

$$0 < \sum_{i=1}^{n} \mu_{ic} < n.$$

Finally, the fuzzy partitioning space is the set:

$$M_s = \{\mathbf{U} \in \mathbb{R}^{C \times n} | \mu_{ic} \in [0,1], \forall c, i; \sum_{c=1}^{C} \mu_{ic} = 1, \forall i, 0 < \sum_{i=1}^{n} \mu_{ic} < n, \forall c\}$$

According to the classification proposed by Han and Kamber (2001, chapter 7), the major categories of clustering methods are partitioning methods, hierarchical methods, density-based methods, grid-based methods and model-based methods.

2.2.1 *Partitioning methods*

Let $\mathcal{Z}$ be a data set of n objects $\{z_1, z_2, \ldots, z_n\}$, and C, the number of clusters to form, a partitioning algorithm organizes the objects into C partitions $(C \leq n)$, where each partition represents a cluster. That is, it classifies the data into C groups, which satisfy the following requirements: (1) each group contain at least one object, and (2) each object belongs to exactly one group. A partitioning method uses an iterative relocation technique that attempts to improve the partitioning by moving objects from one group to another. The clusters are formed by optimizing an objective partitioning criterion, such as a dissimilarity function based on distance. Most of the well known partitioning clustering methods iteratively update the so-called centroids or cluster centers, and for this reason they are often referred as center-based clustering methods. Achieving global optimality in partitioning-based clustering would require the exhaustive enumeration of all of possible partitions. Instead, most applications adopt one of a few popular heuristic methods, such as (1) the k-means algorithm, where each cluster is represented by the mean value of the objects in the cluster, and (2) the k-medoids algorithm, where each cluster is represented by one of the objects located near the center of the cluster.

k-means algorithm

The k-means algorithm (MacQueen, 1967; Hartigan and Wong, 1979) takes the input parameter, C, and partitions a set of n objects into C clusters in such a way the resulting intra-cluster similarity is high but the inter-cluster similarity is low. Cluster similarity is measured in regard to the mean value of the objects in a cluster, which can be viewed as the centroid of cluster.
The k-means algorithm proceeds as follows. First, it randomly selects c of the objects, each of which initially represents a center. For each of the remaining objects, an object is assigned to the cluster to which it is the most similar, based on the distance between the object and the cluster mean. It then computes the new mean for each cluster. This process iterates until the criterion

function converges. Typically, the square-error criterion is used, defined as:

$$EC = \sum_{c=1}^{C} \sum_{i=1}^{n} |z_i - m_c|^2, \tag{2.8}$$

where EC is the sum of the square errors for all objects in the data set, z_i represents a given object, and m_c is the mean of cluster $\mathcal{A}_c$. Thus, for each object in each cluster, the distance from the object to its cluster center is squared, and the distances are summed. This criterion tries to make the resulting c clusters as compact and as separate as possible. The k-means algorithm has the following steps:

1. arbitrarily choose c objects from $\mathcal{Z}$ as initial cluster centers;
2. assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;
3. update the cluster means, i.e., calculate the mean value of the objects for each cluster.

The procedure is stopped when the cluster centers do not move any more. Otherwise, repeat Steps 2 and 3 until convergence.
The method often terminates at a local optimum and the necessity for users to specify the number of clusters in advance can be seen as a disadvantage. Moreover, it is sensitive to noise and outlier data points because a small number of such data can substantially influence the mean value. There are quite a few variants of the k-means method. These can differ in the selection of the initial means, the calculation of dissimilarity, and the strategies for calculating cluster means. An interesting strategy that often yields good results is to first apply a hierarchical agglomeration algorithm, which determines the number of clusters and finds an initial partition, and then use iterative relocation to improve the clustering.

Fuzzy *c*-means algorithm

The counterpart of k-means for fuzzy partitions is the fuzzy c-means algorithm, proposed by Dunn (1973) and developed by Bezdek (1981).
Fuzzy c-means considers each data point as a possible member of multiple

clusters with a membership value. This algorithm is based on minimization of the following objective function:

$$\sum_{i=1}^{n}\sum_{c=1}^{C}(\mu_{ic})^m\|z_i - m_c\|^2 \tag{2.9}$$

$s.t.$

$$\mu_{ic} \in [0,1],\ \forall i, c;$$

$$\sum_{c=1}^{C}\mu_{ic} = 1,\ ;$$

$$0 < \sum_{i=1}^{n}\mu_{ic} < n.$$

In the equation (2.9) , m is any real number greater than 1, μ_{ic} is the degree of membership of z_i in the cluster c, $\|\cdot\|$ is any norm expressing the similarity between any measured data point and the center. The parameter m is called *fuzzifier* or *weighting coefficient*. To perform fuzzy partitioning the number of cluster and the weighting coefficient have to be chosen. The procedure is carried out through an iterative optimization of the objective function shown above, with the update of membership value and the cluster centers.

Probabilistic Distance clustering

Ben-Israel and Iyigun (2008) proposed a new iterative approach for probabilistic clustering of data. It is a generalization, to several centers, of the Weizsfeld method (Weiszfeld, 1937) for solving the Fermat-Weber location problem (Kuhn, 1973). Given a vector of data points $\{z_1, \ldots, z_i, \ldots, z_n\} \in \mathbb{R}^n$, the assignment of points to clusters is "soft", in the sense that the membership of a data point in a cluster is given as a probability. The basis of the probabilistic distance algorithm is explained for any point and all clusters by the following relationship between the distance and the probabilities:

$$P_c * d_c = constant\ \forall c,$$

where P_c is the probability that z_i is a member of the cluster $\mathcal{A}_c$ and d_c is a distance function of the data point from the given cluster center c. The distance functions are, in general, different from one cluster to another.

According to the working principle proposed by Ben-Israel and Iyigun, given clusters, their centers and the distances of data points from these centers, the probability of cluster membership at any point is assumed inversely proportional to the distance from the cluster center. At each iteration, the distances (Euclidean or elliptic) from the cluster centers are computed for all data points, and the centers are updated as convex combinations of these points, with weights determined by the above principle. Computations stop when the centers stop moving. The progress of the algorithm is monitored by the Joint Distance Function (JDF), a distance function that captures the data in its low contours. It is equal to zero if and only if z_i coincides with one of the cluster centers, in such a case z_i belongs to that cluster with probability 1. If all the distances are equal, say equal to d, then the JDF is equal to d/c and all $P_c = 1/C$ showing indifference between the clusters. As the distances increase, so the JDF does, indicating greater uncertainty about the cluster where a data point belongs. The centers updated by the algorithm are stationary points of the JDF. The algorithm requires a small number of cheap iterations and it is insensitive to outliers

k-medoids algorithm

k-medoids method (Kaufman and Rousseeuw, 2009) overcomes the limit of k-means algorithm with respect to its sensitive to outliers. Instead of taking the mean value of the objects in a cluster as a reference point, the algorithm uses one representative object per cluster. Each remaining object is clustered with the representative object to which it is the most similar. The partitioning method is then performed by minimizing the sum of the dissimilarities between each object and its corresponding reference point. To group n objects into c clusters, an absolute-error criterion is used, defined as:

$$AEC = \sum_{c=1}^{C} \sum_{i=1}^{n} |z_i - o_c|^2, \tag{2.10}$$

which, AEC, is the sum of the absolute error for all objects in the data set and o_c is the representative object of $\mathcal{A}_c$.

2.2.2 *Hierarchical methods*

A hierarchical method creates a hierarchical decomposition of the given set of data objects by grouping them into a tree of clusters. Generally, there are two types of hierarchical clustering methods: the agglomerative and the divisive, depending on whether the hierarchical decomposition is formed in a bottom-up (merging) or top-down (splitting) fashion.
Agglomerative methods start by placing each object in its own cluster and then merging clusters into larger and larger clusters, until all objects are in a single cluster or until certain termination conditions, such as the desired number of clusters, are satisfied. Most hierarchical clustering methods belong to this category. They differ only in their definition of inter-cluster similarity.
Divisive methods, following a top-down strategy, do the reverse of agglomerative hierarchical clustering by starting with all objects in one cluster. They subdivide the cluster into smaller and smaller pieces, until each object forms a cluster on its own or until it satisfies certain termination conditions, such as a desired number of clusters is obtained or the diameter of each cluster is within a certain threshold.
In either agglomerative or divisive hierarchical clustering, the user can specify the desired number of clusters as a termination condition. A tree structure called dendrogram is commonly used to represent the process of hierarchical clustering. It shows how objects are grouped together step by step.
Different measures for distance between clusters can be used. The single (complete) linkage clustering measures the distance between two clusters as the shortest (largest) distance from any member of one cluster to any member of the other cluster. The average linkage clustering measures the distance between two cluster as the average distance from any member of one cluster to any member of the other cluster. The minimum variance criterion proposed by Ward Jr (1963) minimizes the total within-cluster variance. At each step, the pair of clusters with the smallest increase in the value of sum of squares variance are merged. The algorithm at each step finds the pair of clusters that leads to minimum increase in the total within-cluster variance after merging. The quality of a pure hierarchical clustering method suffers from its inability to perform any adjustment once a merge or split decision has been executed.

That is, if a particular merge or split decision later turns out to be a poor choice, the method cannot backtrack and correct it.

2.2.3 *Density-based methods*

To discover clusters with arbitrary shape, density-based clustering methods have been developed. These methods typically regard clusters as dense regions of objects in the data space that are separated by regions of low density (representing noise). Density-Based Spatial Clustering of Applications with Noise (DBSCAN) (Ester et al., 1996) grows clusters according to a density-based connectivity analysis. To produce a cluster ordering obtained from a wide range of parameter settings, Ankerst et al. (1999) proposed Ordering Points To Identify The Clustering Structure (OPTICS).

DBSCAN

The general idea of DBSCAN algorithm is to grow regions with sufficiently high density into clusters and to discover clusters of arbitrary shape in spatial databases with noise. It defines a cluster as a maximal set of density-connected points. The basic ideas of density-based clustering involve a number of new definitions.

- The neighborhood within a radius ε of a given object is called the $\varepsilon - neighborhood$ of the object.
- If the ε-neighborhood of an object contains at least a minimum number, *MinPts*, of objects, then the object is called a core object.
- Given a set of objects, $\mathbf{Z}$, an object $\mathbf{z}$ is directly density-reachable from object $\mathbf{q}$ if $\mathbf{z}$ is within the ε-neighborhood of $\mathbf{q}$, and $\mathbf{q}$ is a core object.
- An object $\mathbf{z}$ is density-reachable from object $\mathbf{q}$ with respect to ε and *MinPts* in a set of objects, $\mathbf{Z}$, if there is a chain of objects, $\mathbf{p}_1, \ldots, \mathbf{p}_n$, where $\mathbf{p}_1 = \mathbf{q}$ and $\mathbf{p}_n = \mathbf{p}$ such that $\mathbf{p}_{i+1}$ is directly density-reachable from $\mathbf{p}_i$ with respect to ε and *MinPts*, for $1 \leq i \leq n, \mathbf{p}_i \in \mathbf{Z}$.
- An object $\mathbf{p}$ is density-connected to object $\mathbf{q}$ with respect to ε and *MinPts* in a set of objects,$\mathbf{Z}$, if there is an object $\mathbf{o} \in \mathbf{Z}$ such that both $\mathbf{p}$ and $\mathbf{q}$ are density-reachable from $\mathbf{o}$ with respect to ε and *MinPts*.

Density reachability is the transitive closure of direct density reachability and this relationship is asymmetric. Only core objects are mutually density reachable. Density connectivity, however, is a symmetric relation. A density-based cluster is a set of density-connected objects that is maximal with respect to density-reachability. Every object not contained in any cluster is considered to be noise. DBSCAN searches for clusters by checking the ε-neighborhood of each point in the database. If the ε-neighborhood of a point $\mathbf{p}$ contains more than *MinPts*, a new cluster with $\mathbf{p}$ as a core object is created. DBSCAN iteratively collects directly density-reachable objects from these core objects, which may involve the merge of a few density-reachable clusters. The process terminates when no new point can be added to any cluster.

OPTICS

Although DBSCAN can cluster objects given input parameters such as ε and *MinPts* , it still leaves the user with the responsibility of selecting the parameter values that lead to the discovery of acceptable clusters. To overcome this difficulty, a cluster analysis method called OPTICS was proposed by Ankerst et al. (1999). Rather than producing a data set clustering explicitly, OPTICS computes an augmented cluster ordering for automatic and interactive cluster analysis. This ordering represents the density-based clustering structure of the data. It contains information that is equivalent to density-based clustering obtained from a wide range of parameter settings. The cluster ordering can be used to extract basic clustering information (such as cluster centers or arbitrary-shaped clusters) as well as provide the intrinsic clustering structure. This order selects an object that is density-reachable with respect to the lowest ε value so that clusters with higher density (lower ε) will be finished first. Based on this idea, two values need to be stored for each object, the core-distance and the reachability-distance:

- The core-distance of an object $\mathbf{p}$ is the smallest $\varepsilon^{'}$ value that makes $\mathbf{p}$ a core object. If $\mathbf{p}$ is not a core object, the core-distance of $\mathbf{p}$ is undefined.
- The reachability-distance of an object $\mathbf{q}$ with respect to another object $\mathbf{p}$ is the greater value of the core-distance of $\mathbf{p}$ and the Euclidean distance between $\mathbf{p}$ and $\mathbf{q}$. If $\mathbf{p}$ is not a core object, the reachability-distance between $\mathbf{p}$ and $\mathbf{q}$ is undefined

The OPTICS algorithm creates an ordering of the objects in a database, additionally storing the core-distance and a suitable reachability distance for each object. To extract clusters based on the ordering information produced by OPTICS an algorithm was proposed. Such information is sufficient for the extraction of all density-based clusterings with respect to any distance ε' that is smaller than the distance ε used in generating the order.

2.2.4 *Grid-based methods*

The grid-based clustering approach uses a multi-resolution grid data structure. It quantizes the object space into a finite number of cells that form a grid structure on which all of the operations for clustering are performed. The main advantage of the approach is its fast processing time, which is typically independent of the number of data objects, yet dependent on only the number of cells in each dimension in the quantized space. Some typical examples of the grid-based approach include STING, proposed by Wang et al. (1997), which explores statistical information stored in the grid cells.

STING

STING is a grid-based multi-resolution clustering technique in which the spatial area is divided into rectangular cells. There are usually several levels of such rectangular cells corresponding to different levels of resolution, and these cells form a hierarchical structure: each cell at a high level is partitioned to form a number of cells at the next lower level. Statistical information regarding the attributes in each grid cell (such as the mean, is precomputed and stored. Statistical parameters of higher-level cells can easily be computed from the parameters of the lower-level cells. These parameters are useful for query processing and they include the following: the attribute-independent parameter (count); the attribute-dependent parameters (mean, standard deviation, minimum, maximum); and the type of distribution that the attribute value in the cell follows (normal, uniform, exponential, or none -if the distribution is unknown). When the data are loaded into the database, the parameters count, mean, standard deviation, minimum, maximum of the bottom-level cells are calculated directly from the data. The value of distribution may either be assigned by the user if the distribution type is known beforehand

or obtained by hypothesis tests such as the χ^2 test. The type of distribution of a higher-level cell can be computed based on the majority of distribution types of its corresponding lower-level cells in conjunction with a threshold filtering process. If the distributions of the lower level cells disagree with each other and fail the threshold test, the distribution type of the high-level cell is set to none. The statistical parameters can be used in a top-down, grid-based method as follows. First, a layer within the hierarchical structure is determined from which the query-answering process is to start. This layer typically contains a small number of cells. For each cell in the current layer, it computes the confidence interval (or estimated range of probability) reflecting the cell's relevancy to the given query. The irrelevant cells are removed from further consideration. Processing of the next lower level examines only the remaining relevant cells. This process is repeated until the bottom layer is reached. At this time, if the query specification is met, the regions of relevant cells that satisfy the query are returned. Otherwise, the data that fall into the relevant cells are retrieved and further processed until they meet the requirements of the query. STING offers several advantages: (1) the grid-based computation is query-independent, because the statistical information stored in each cell represents the summary information of the data in the grid cell, independent of the query; (2) the grid structure facilitates parallel processing and incremental updating; (3) the method's efficiency, because after generating hierarchical structure, the query processing time depends on the total number of grid cells at the lowest level, which is usually much smaller than the total number of objects. Because STING uses a multi-resolution approach to cluster analysis, the quality of STING clustering depends on the granularity of the lowest level of the grid structure. If the granularity is very fine, the cost of processing will increase substantially; however, if the bottom level of the grid structure is too coarse, it may reduce the quality of cluster analysis. Moreover, STING does not consider the spatial relationship between the children and their neighboring cells for construction of a parent cell. As a result, all of the cluster boundaries are either horizontal and vertical and no diagonal boundary is detected. This may lower the quality and accuracy of the cluster despite the fast processing time of technique.

2.2.5 *Model-based methods*

Model-based clustering methods attempt to optimize the fit between the given data and some mathematical model. Such methods are often based on the assumption that the data are generated by a mixture of underlying probability distributions. In practice, each cluster can be represented mathematically by a parametric probability distribution. The entire data is a mixture of these distributions, where each individual distribution is typically referred to as a component distribution. We can therefore cluster the data using a finite mixture density model of c probability distributions, where each distribution represents a cluster. The problem is to estimate the parameters of the probability distributions so as to best fit the data.

Expectation-Maximization

The Expectation-Maximization (EM) algorithm is a popular iterative refinement algorithm that can be used for finding the parameter estimates. It can be viewed as an extension of the k-means paradigm, discussed in section 2.2.1. Instead of assigning each object to a dedicated cluster, EM assigns each object to a cluster according to a weight representing the probability of membership. In other words, there are no strict boundaries between clusters. Therefore, new means are computed based on weighted measures. EM starts with an initial estimate the parameters of the mixture model (collectively referred to as the parameter vector). It iteratively re-scores the objects against the mixture density produced by the parameter vector. The re-scored objects are then used to update the parameter estimates. Each object is assigned a probability that it would possess a certain set of attribute values given that it was a member of a given cluster. The algorithm is described as follows:

1. Make an initial guess of the parameter vector: This involves randomly selecting c objects to represent the cluster means or centers (as in k-means partitioning), as well as making guesses for the additional parameters.

2. Iteratively refine the parameters (or clusters) based on the following two steps:

a) Expectation Step: Assign each object $\mathbf{z}_i$ to cluster A_c with the probability

$$P(z_i \in A_c) = P(A_c|z_i) = \frac{P(C_c)P(z_i|A_c)}{P(z_i)} \tag{2.11}$$

where $P(z_i|A_c) \sim \mathcal{N}(m_c, E_c(z_i))$. In other words, this step calculates the probability of cluster membership of object z_i, for each of the clusters. These probabilities are the expected cluster memberships for object z_i.

b) Maximization Step: Use the probability estimates from above to re-estimate the model parameters.

$$m_c = \frac{1}{n}\sum_{i=1}^{n}\frac{z_i P(z_i \in \mathcal{A}_c)}{\sum_t P(z_i \in \mathcal{A}_t)} \tag{2.12}$$

This step is the maximization of the likelihood of the distributions given the data.

The EM algorithm is simple and easy to implement. In practice, it converges fast but may not reach the global optima. Convergence is guaranteed for certain forms of optimization functions. The computational complexity is linear in the number of input features, in the number of objects and in the number of iterations. Bayesian clustering methods focus on the computation of class-conditional probability density.

Self Organizing feature Maps

A neural network is a set of connected input/output units, where each connection has a weight associated with it. Neural networks have several properties that make them popular for clustering. First, neural networks are inherently parallel and distributed processing architectures. Second, neural networks learn by adjusting their interconnection weights so as to best fit the data. Third, neural networks process numerical vectors and require object patterns to be represented by quantitative features only. The neural network approach to clustering tends to represent each cluster as an exemplar. An exemplar acts as a prototype of the cluster and does not necessarily have to correspond to a particular data example or object. New objects can be distributed to the cluster whose exemplar is the most similar, based on some

distance measure. The attributes of an object assigned to a cluster can be predicted from the attributes of the cluster's exemplar.
Self-organizing feature maps (SOM) developed by Kohonen (1990) are one of the most popular neural network methods for cluster analysis. The aim is to represent all points in a high-dimensional source space by points in a low-dimensional (usually 2-D or 3-D) target space, such that the distance and proximity relationships (hence the topology) are preserved as much as possible. The method is particularly useful when a nonlinear mapping is inherent in the problem itself. SOMs can also be viewed as a constrained version of k-means clustering, in which the cluster centers tend to lie in a low-dimensional manifold in the feature or attribute space. With SOM, clustering is performed by having several units competing for the current object. The unit whose weight vector is closest to the current object becomes the winning or active unit. So as to move even closer to the input object, the weights of the winning unit are adjusted, as well as those of its nearest neighbors. SOMs assume that there is some topology or ordering among the input objects and that the units will eventually take on this structure in space. The organization of units is said to form a feature map.

2.2.6 *Dissimilarity measure*

An important problem in the application of cluster analysis is the decision regarding the determination of a suitable metric.
In the specific context of time series clustering, the concept of dissimilarity is particularly complex due to the dynamic character of the series. Thus, the choice of a suitable metric becomes fundamental. Dissimilarities usually considered in conventional clustering could not work well with time dependent data because they ignore the interdependence relationship between values. The choice of an adequate distance measure depends largely on the nature of the clustering. Different approaches to define dissimilarity between time series have been proposed in the literature.

Following Montero and Vilar (2014) a short survey is reported. The different distance measures between time series have been grouped into four categories: non parametric measures, model-based measures, complexity-based measures and prediction-based measures.

Non parametric approaches

Let y_t and $y_t^{'}$ be two time series with the same length T.
A simple model-free approach to measure the proximity between two time series is to consider conventional metrics based on the closeness of their values at specific points of time.

The Minkowski distance of order q, where q is a positive integer, is defined by:

$$d = \left[\sum_{t=1}^{T} (y_t - y_t^{'})^q \right]^{(1/q)} \tag{2.13}$$

The L_q-norm distance is typically used with q being 2 (Euclidean distance) or 1 (Manhattan distance).
The proximity notion relies on the closeness of the values observed at corresponding points of time so that the observations are treated as if they were independent. Thus, if the objective is to compare profiles of series, the Euclidean or Manhattan distances between raw data evaluating a one-to-one mapping of each pair of sequences, can produce satisfactory results. However this metric is sensitive to signal transformations or time scaling and it is invariant to permutations over time.

The distance introduced by Fréchet (1906) to measure the proximity between continuous curves has been extensively used on the discrete case by Eiter and Mannila (1994) and in the time series framework. A formal definition for the discrete case is the following. Let M be the set of all possible sequences of m pairs preserving the observations order in the form:

$$r = \left((y_{a_1}, y_{b_1}^{'}), \ldots, (y_{a_m}, y_{b_m}^{'}) \right)$$

where $a_i, b_j \in \{1, \ldots T\}$ such that $a_1 = b_1, = 1$, $a_m = b_m = T$, and $a_{i+1} = a_i$ or $a_i + 1$ and $b_{i+1} = b_i$ or $b_i + 1$, for $i \in \{1, \ldots, m-1\}$. The Fréchet distance is defined by:

$$d = \min_{r \in M} \left(\max_{i=1,\ldots m} |y_{a_i} - y_{b_i}^{'}| \right), \tag{2.14}$$

Unlike the Minkowski distance, this distance does not just treat the series as two points sets, but it takes into account the ordering of the observations. Moreover, it can be computed on series of different length.

The dynamic time warping (DTW) distance was studied in depth by Sankoff and Kruskal (1983) and proposed to find patterns in time series by Berndt and Clifford (1994). As Fréchet distance, DTW distance is aimed to find a mapping r between the series so that a specific distance measure between the coupled observations is minimized. The definition of the DTW distance is given by:

$$d = \min_{r \in M} \left(\sum_{i=1,\dots m} |y_{a_i} - y_{b_i}^{'}| \right). \tag{2.15}$$

Fréchet distance and DTW distance allow to recognize similar shapes, even in the presence of signal transformations such as shifting and/or scaling. However, as in the case of L_q-norm distance, both distances ignore the temporal structure of the values since the proximity is based on the differences $|y_{a_i} - y_{b_i}^{'}|$ regardless of the behavior around these values.

To cover both conventional measures for the proximity on observations and temporal correlation for the behavior proximity estimation, Chouakria and Nagabhushan (2007) introduced a dissimilarity measure. The proximity between the dynamic behaviors of the series is evaluated by means of the first order temporal correlation coefficient $\rho_{y_t, y_t^{'}}(t)$. The dissimilarity index proposed modulates the proximity between the raw-values of two series y_t and $y_t^{'}$ by $\rho_{y_t, y_t^{'}}(t)$. Specifically, the dissimilarity index proposed by Chouakria and Nagabhushan is defined as:

$$d = \phi_q \left[\rho_{y_t, y_t^{'}}(t) \right] d, \tag{2.16}$$

where $\phi_q(\cdot)$ is an adaptive tuning function and d is a distance measure between y and $y^{'}$. $\phi_q(\cdot)$ automatically modulate a conventional raw data distance $d(\cdot)$ (e.g. L_q-norm, Fréchet and DTW distances) according to the temporal correlation. The modulating function should work increasing (decreasing) the weight of the dissimilarity between observations as the temporal

correlation decreases from 0 to -1 (increases from 0 to $+1$). In addition, this distance should approach the raw-data discrepancy as the temporal correlation is zero. Instead of, for instance, a linear tuning function, Chouakria and Nagabhushan (2007) proposed to use an exponential adaptive function given by

$$\phi_q(u) = \frac{2}{1 + exp(qu)} \quad q \geq 0. \tag{2.17}$$

The following distance measures are all model-free dissimilarity measures but based on particular features of the time series. The aim of these distances is to replace the raw data by a reduced number of features characterizing the time series, which allows us to take into account the temporal structure of the series.

A first and simple dissimilarity criterion is to consider the Pearson's correlation factor between two time series $\rho_{(y_t, y'_t)}$. Golay et al. (1998) constructed a fuzzy k-means algorithm using the following two cross-correlation-based distances:

$$d = \sqrt{2(1 - \rho_{(y_t, y_t^i)})}, \tag{2.18}$$

$$d = \sqrt{\left(\frac{1 - \rho_{(y_t, y'_t)}}{1 + \rho_{(y_t, y'_t)}}\right)^{\beta}}, \quad \beta \geq 0, \tag{2.19}$$

where the parameter β allows regulation of the fast decreasing of the distance.

Several authors have considered measures based on the estimated autocorrelation functions. Galeano and Peña (2000) define a distance between y_t and y'_t as follows:

$$d = \sqrt{\left(\mathbf{A\hat{C}F}_{y_t} - \mathbf{A\hat{C}F}_{y'_t}\right)^{\top} \mathbf{\Omega} \left(\mathbf{A\hat{C}F}_{y_t} - \mathbf{A\hat{C}F}_{y'_t}\right)}, \tag{2.20}$$

where $\mathbf{A\hat{C}F}_{y_t}$ and $\mathbf{A\hat{C}F}_{y'_t}$ are the estimated autocorrelation vectors of y_t and y'_t respectively and $\mathbf{\Omega}$ is a matrix of weights. When $\mathbf{\Omega} = \mathbf{I}$, the

autocorrelation-based distance becomes the Euclidean distance between the estimated autocorrelation functions. When Ω is the inverse covariance matrix of the autocorrelations, the Mahalanobis distance between the autocorrelations is returned. It is also common to use weights that decrease with the autocorrelation lag.
Analogous distances can be constructed by considering the partial autocorrelation functions.

So far all metrics work in the time domain, but the frequency domain approach also offers an interesting alternative to measure the dissimilarity between time series. The frequency domain approach assesses the dissimilarity between the corresponding spectral representations of the series. Power spectrum analysis is concerned with the distribution of the signal power in the frequency domain.

Signal processing methods provide a variety of tools for modeling, analysis, coding, synthesis and recognition of signals. The purpose of a transformation is to express a signal or a system in terms of a combination of a set of elementary simple signals (such as sinusoidal signals, eigenvectors or wavelets) that lend themselves to relatively easy analysis, interpretation and manipulation. Transform-based signal processing methods include Fourier transform, Laplace transform, z-transform and wavelet transforms. The most widely applied signal transform is the Fourier transform. Its strength in signal analysis and pattern recognition is its ability to reveal spectral structures that may be used to describe a signal. The Fourier transform of the correlation function is the power spectrum. The power spectrum of a signal gives the distribution of the signal power among various frequencies and reveals the existence, or the absence, of repetitive patterns and correlation structures in a signal process. Correlation is a measure of self-similarity of a signal with its delayed version. Like power spectrum, correlation function reveals information on the periodic or random structure of the signal. In general, the more correlated or predictable a signal, the more concentrated its power spectrum, and conversely the more random or unpredictable a signal, the more spread its power spectrum. Therefore the power spectrum of a signal can be used to deduce the existence of repetitive structures or correlated patterns in the signal process (Vaseghi, 2008).

Let $f_j = 2\pi j/T$, $j = 1, \ldots, T/2$ in the range 0 to π be the frequencies of the series.

For stochastic signals, the power-spectral density is defined as the Fourier transform of the autocorrelation function. The classic method for estimation of the power spectral density of a sample record is the periodogram introduced by Schuster (1897).

The periodogram of series y_t is defined as:

$$PSD_y(f_j) = \frac{1}{T}\sum_{t=1}^{T} |y_t(f_j)\exp(-\iota t f_j)|^2 \tag{2.21}$$

The power-spectral density function defined in (2.21) is the basis of non-parametric methods of spectral estimation. Owing to the finite length and the random nature of most signals, the spectra obtained from different records of a signal vary randomly about an average spectrum.

Let $PSD_{y'}(f_j) = \frac{1}{T}\sum_{t=1}^{T} |y'_t(f_j)\exp(-\iota t f_j)|^2$ be the periodogram of series y'_t. Caiado et al. (2006) introduced the following three dissimilarity measure based on the periodograms. A distance between y and y' is thus defined by the Euclidean distance between :

- the periodogram ordinates:

$$d = \sqrt{\sum_{j=1}^{(n/2)} [PSD_y(f_j) - PSD_{y'}(f_j)]^2}; \tag{2.22}$$

- the normalized periodogram (or rescaled periodogram) ordinates:

$$d = \sqrt{\sum_{j=1}^{(n/2)} [NPSD_y(f_j) - NPSD_{y'}(f_j)]^2}; \tag{2.23}$$

  by replacing $PSD(f_j)$ in (2.22) by $NPSD(f_j) = PSD(f_j)/\hat{\gamma}_0$ where $\hat{\gamma}_0$ is the sample variance of the time series. It is useful when the interest is about the correlation structure.

- the logarithm of the normalized periodogram ordinates:

$$d = \sqrt{\sum_{j=1}^{(n/2)} [\log NPSD_y(f_j) - \log NPSD_{y'}(f_j)]^2}. \tag{2.24}$$

Model-based approaches

Model-based dissimilarity measures assume that the underlying models are generated from specific parametric structures.
Let $Y_t = (y_{1t}, \ldots, y_{nt})'$ be a vector of time series with components represented by autoregressive integrated moving average ARIMA(p, d, q) models:

$$\varphi_i(B)(1-B)^d y_{it} = \theta_i(B)\varepsilon_{it}, \quad i = 1, \ldots, n. \tag{2.25}$$

where $\varphi(B)$ is the autoregressive operator of order p and $\theta_i(B)$ is the moving average operator of order q; B is the back-shift operator and $(1-B)^d$ is the differencing operator of order d. The autoregressive and moving average polynomials in (2.25) are assumed to have all roots outside the unit circle, so that each process $Y_{it} = (1-B)^d y_{it}$, is causal and invertible.
The main approach in the literature is to assume that the generating processes of time series follow invertible ARIMA(p, d, q) models. In such a case, the idea is fitting an ARIMA model to each series and then measuring the dissimilarity between the fitted models. First step requires estimating the structure and the parameters of ARIMA models. The structure is either assumed to be given or automatically estimated using, for example, the Akaike's information criterion (AIC) or the Schawartz's Bayesian information criterion (BIC). The parameter values are commonly fitted using generalized least squares estimators.
Some of the most relevant dissimilarity measures derived in the literature under the assumption of underlying ARIMA models are surveyed in the following.
Piccolo (1990) defines a dissimilarity measure in the class of invertible ARIMA processes as the Euclidean distance between the AR(∞) operators approximating the corresponding ARIMA structures. Piccolo argues that the autoregressive expansions convey all the useful information about the stochastic structure of this kind of processes (except for initial values). If the series are non-stationary, differencing is carried out to make them stationary, and if the series possess seasonality, then it should be removed before further analysis.
Let y_t be a zero mean stochastic process following an invertible ARIMA($p, 0, q$) model: $\varphi(B)y_t = \theta(B)\varepsilon$. Then it can be represented by

the AR(∞) operator $\delta(B) = \theta^{-1}(B)\varphi(B) = 1 - \delta_1(B) - \delta_2(B^2) - \ldots$, and the δ coefficients contain all the information about the stochastic dependence structure of a time series. Then, the distance proposed by Piccolo is a metric by comparing the respective δ sequences, defined as:

$$d = \sqrt{\sum_{t=1}^{\infty} \left(\delta_{t,y} - \delta_{t,y'}\right)^2}. \tag{2.26}$$

For the class of invertible and stationary ARMA processes, Maharaj (1996, 2000) introduced two discrepancy measures based on hypotheses testing to determine whether, or not, two time series have significantly different generating processes. The first of these metrics is given by the test statistic:

$$d = \sqrt{T}(\hat{\Delta}_{y_t} - \hat{\Delta}_{y_t'})^\top \hat{\mathbf{V}}^{-1}(\hat{\Delta}_{y_t} - \hat{\Delta}_{y_t'}), \tag{2.27}$$

where $\hat{\Delta}_{y_t}$ and $\hat{\Delta}_{y_t'}$ are AR(p) parameter estimations of two time series y_t and y_t', respectively, with p selected as in Piccolo (1990), $\hat{\mathbf{V}}$ is an estimator of $\mathbf{V} = \sigma^2_{y_t}\mathbf{S}^{-1}_{y_t}(p) + \sigma^2_{y_t'}\mathbf{S}^{-1}_{y_t'}(p)$ with $\sigma^2_{y_t}$ and $\sigma^2_{y_t'}$ denoting the variances of the white noise processes associated with y_t and $y_t^{'}$, and $\mathbf{S}_{y_t}$ and $\mathbf{S}_{y_t'}$ the sample covariance matrices of both series. Maharaj (1996) demonstrated that the distance is asymptotically χ^2 distributed under the null hypothesis of equality of generating processes. Therefore, the dissimilarity between AR(p) parameter estimations of y_t and $y_t^{'}$, can also be measured through the associated p value. Both the test statistic and the associated p-value satisfy the properties of non-negativity and symmetry so that any of them can be used as dissimilarity measure between y_t and $y_t^{'}$.

Although Maharaj (1996) and Piccolo (1990) evaluate the dissimilarity between two series by comparing their autoregressive approximations, there is a substantial difference between them. The distance proposed by Piccolo does not take into account the variance of the white noise processes associated with the observed series, while the statistic proposed by Maharaj involves these variances in its definition. The measures proposed by Maharaj (1996) come from a hypothesis testing procedure designed to compare two independent time series. To overcome this limitation, Maharaj (2000) introduced a new testing procedure that can be applied to time series that are

not necessarily independent. In this case, a pooled model including collectively the models fitted to y_t and $y_t^{'}$ is considered, and the combined vector of $2p$ AR parameters $\Delta = \left(\Delta_{y_t}, \Delta_{y_t^{'}}\right)$ is estimated by using generalized least squares. Assuming that the two models are correlated at the same points in time but uncorrelated across observations, the proposed test statistic is also asymptotically distributed as χ^2 with p degrees of freedom. As before, a dissimilarity measure based on the p-values associated with this new test can be constructed.

Complexity-based approaches

Dissimilarity measures based on comparing levels of complexity of time series are presented in the following.

Here, similarity of two time series does not rely on specific serial features or the knowledge of underlying models, but on measuring the level of shared information by both time series. The mutual information between two series can be formally established using the Kolmogorov complexity concept, although this measure cannot be computed in practice and must be approximated.

The Kolmogorov complexity $K(y)$ of an object y is the length of the shortest program capable to produce y on a universal computer, such as a Turing machine (Li and Vitányi, 2009). Intuitively, $K(y)$ is the minimal quantity of information required to generate y by an algorithm, and therefore, the level of complexity of y is related to $K(y)$. Analogously, given two objects y and $y^{'}$, the conditional Kolmogorov complexity $K(y|y^{'})$ of y given $y^{'}$ is defined as the length of the shortest program producing y when $y^{'}$ is given as an auxiliary input on the program. Therefore, $K(y) - K(y|y^{'})$ measures the amount of information that $y^{'}$ produces about y.

Based on these concepts, Li et al. (2004) proposed a normalized information distance between two objects y and $y^{'}$given by:

$$d = \frac{\max\left\{K(y|y^{'}), K(y^{'}|y)\right\}}{\max\left\{K(y), K(y^{'})\right\}} \tag{2.28}$$

Li et al. show that (2.28) is a metric and it is universal in the following sense: it leads to the smallest values (up to constant precision) among a broad class of normalized distances. (2.28) can be applied to different collections of

objects such as time series, images, texts, etc.
We now assume that y and $y^{'}$ represent times series.
A approach to measure complexity differences between two time series is shortly described below.
As Kolmogorov complexity is non computable, (2.28) is approximated by replacing the quantities $K(\cdot)$ by the length of the compressed objects obtained from data compressors (such as gzip, bzip2). Consider a specific data compression algorithm and denote by $C(y)$ the compressed size of y . The denominator of (2.28) is easy to approximate by $\max\left\{C(y), C(y^{'})\right\}$, but the numerator involves conditional Kolmogorov complexities making more difficult to obtain the approximation. To overcome this drawback Li et al. taking into account that $K(y|y^{'})$ is roughly equal to $K(yy^{'}) - K(y)$, where $K(yy^{'})$ is the length of the shortest program to compute the concatenation of y and $y^{'}$. The resulting approximation by following this approach is called "normalized compression distance" and takes the form:

$$d = \frac{C(y, y^{'}) - \min\left\{C(y), C(y^{'})\right\}}{\max\left\{C(y), C(y^{'})\right\}} \tag{2.29}$$

The (2.29) takes nonnegative values ranging from 0 to $1 + \varepsilon$, where ε is due to flaws in the compression techniques. The smaller the (2.29) , the more closely related y and $y^{'}$ are.

Prediction-based approaches

Now we focus on a new dissimilarity notion governed by the performance of future forecasts, i.e., two time series are similar if their forecasts at a specific future time are close. Obviously, a clustering procedure based on this dissimilarity concept may produce results very different to the ones generated from model-based or feature-based clustering methods. For instance, two time series coming from the same generating process can produce different forecasts at a pre-specified horizon, and hence these series might not be clustered together by using this new dissimilarity criterion. Alonso et al. (2006) proposed a dissimilarity measure based on comparing the forecast densities for each series at a future horizon of interest. They argue that using

full forecast densities permits to take into account the variability of the predictions, which is completely ignored when comparisons are based on point wise forecasts. In practice, the forecast densities are unknown and must be approximated from the data. The authors construct this approximation using a smoothed sieve bootstrap procedure combined with kernel density estimation techniques. This procedure requires assuming that the time series admit an AR(1) representation because the sieve bootstrap is based on re-sampling residuals from autoregressive approximations. Vilar et al. (2010) extend this methodology to cover the case of nonparametric models of arbitrary autoregressions. In this new scenario, the sieve bootstrap is not valid, and the forecast densities are approximated considering a bootstrap procedure that mimics the generating processes without assuming any parametric model for the true autoregressive structure of the series.

Specifically, let y_t and $y_t^{'}$ be realizations of stationary processes that admit a general autoregressive representation of the form $R_t = \varphi(R_{t-1}) + \varepsilon_t$, with $\{\varepsilon\}$ an i.i.d. sequence and $\varepsilon(\cdot)$ a smooth function not restricted to any prespecified parametric model. Given a particular future time $T + h$, Vilar et al. introduce the following distance between y and $y^{'}$:

$$d = \int |\hat{f}_{y_{T+h}}(u) - \hat{f}_{y'_{T+h}}(u)|du \tag{2.30}$$

where $\hat{f}_{y_{T+h}}$ and $\hat{f}_{y'_{T+h}}$ denote estimates of the forecast densities at horizon $T+h$ for y_t and $y_t^{'}$ respectively. By construction, (2.30) measures the distance between y_t and $y_t^{'}$ in terms of the disparity between the behaviors of their predicted values at horizon $T + h$. The true forecast densities are replaced by kernel-type estimators based on bootstrap predictions. It takes advantage of being free of the linearity requirement, and hence, it can be applied to a wider class of nonparametric models. A more detailed description can be seen in Vilar et al. (2010).

2.3 *Penalized Spline*

Penalized splines (P-splines) have been introduced by Eilers and Marx (1996) as flexible smoothing procedures combining basis splines (B-spline) and dif-

ference penalties. We recall some essential background.

2.3.1 B-Spline

A B-spline is a piecewise polynomial function defined in a domain spanned by a set of points called "knots". The degree of the polynomial pieces defining the basis function determines its order. A B-spline of degree q is a piecewise polynomial function built using $q+1$ polynomial pieces of degree q joined at q inner knots. The knots can be equally spaced or not. In this work we always deal with B-splines defined on a set of equidistant knots. Taking equidistant knots does not affect our further results and, in our opinion, is convenient in many applications, as shown by Eilers and Marx (2010). At joining points, the derivatives up to the degree $q-1$ of each polynomial piece are continuous. Each basis function takes nonzero values in the domain spanned by $q+2$ internal knots and overlaps to $2q$ polynomial pieces of adjacent bases (except at the boundaries of the domain). At a given knot point $q+1$ B-splines assume nonzero values. In the upper side of figure 2.2 a B-spline of degree 2 is shown. It consists of three quadratic pieces, joined at two knots. In the bottom side of figure 2.2 several over-lappings of a B-spline of degree 3 is shown. Collecting these piecewise polynomial functions it is possible to define a basis matrix $\boldsymbol{B}$: each column of this matrix contains a B-spline. The localness of the polynomials defining the bases makes the $\boldsymbol{B}$ matrix really sparse so that each of its row contains only few nonzero elements. This sparseness is an important feature making B-splines computationally convenient for function interpolation and approximation.

$$\boldsymbol{B} = \begin{bmatrix} B_1(x_1) & B_2(x_1) & B_3(x_1) & ... & B_n(x_1) \\ B_1(x_2) & B_2(x_2) & B_3(x_2) & ... & B_n(x_2) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ B_1(x_n) & B_2(x_n) & B_3(x_n) & ... & B_n(x_n) \end{bmatrix}.$$

Readers interested in a deeper and more detailed reference about B-spline may refer to de Boor and Swartz (1973) and Dierckx (1995).

Several algorithms can be adopted to compute the basis spline functions forming the $\boldsymbol{B}$ matrix. A not convenient way is to evaluate the polynomial segments forming the basis functions analytically. A more practical approach

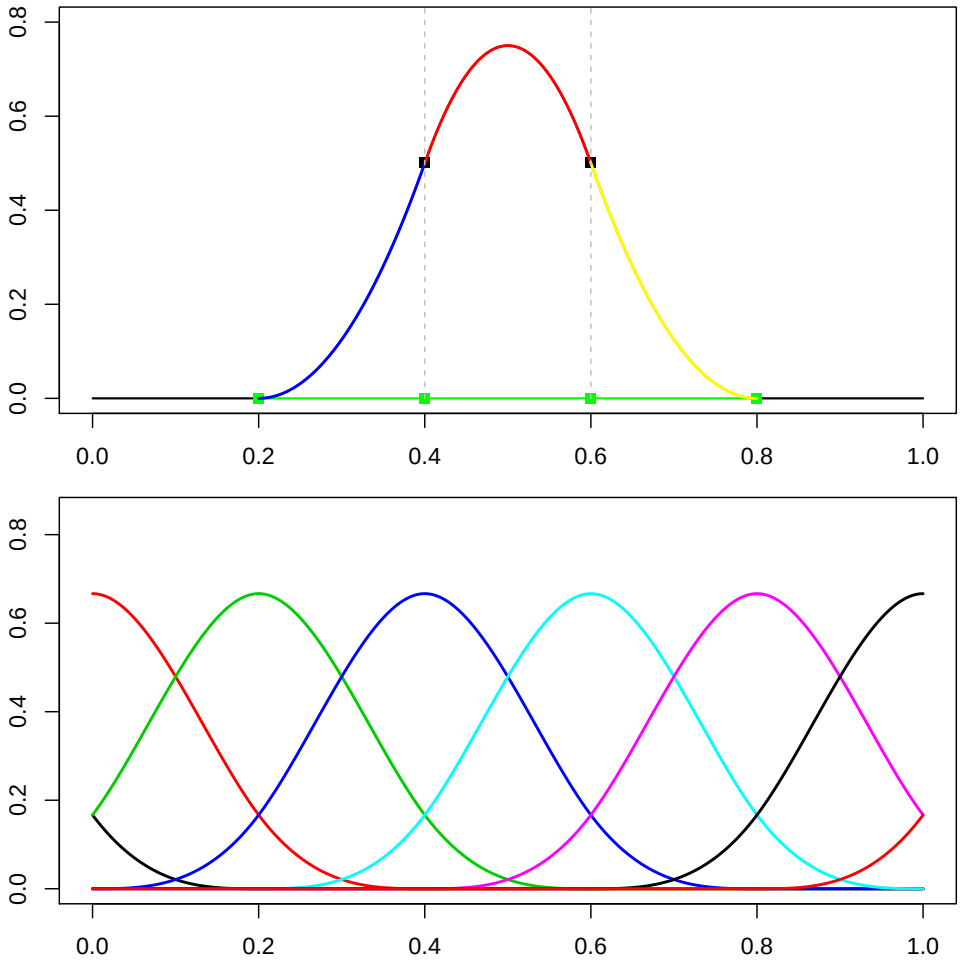

Figure 2.2: The first panel shows one 2^{th} order B-spline with its polynomial components. In the second panel more B-splines of 3^{th} order are shown.

is represented by the algorithm proposed by de Boor (1978). The last alternative that can be mentioned consists in computing the spline functions using differences between truncated power functions (Schumaker, 1981; Eilers and Marx, 2010).

The B-spline matrix $\boldsymbol{B}$ can be used to interpolate or approximate any unknown function. Suppose that we want to approximate the values y assumed by an unknown function $f(x)$ for some values of x. If we denote with $B_j(x,q)$ the value of the j^{th} B-spline at point x we can represent y using a linear combination of B-splines $\widehat{y}(x) = \sum_j \widehat{c}_j B_j(x,q)$ where $\widehat{c}_j$ is the j^{th} B-spline coefficient.

As shown by de Boor (1978), there is a convenient relationship between the d^{th} derivative of a B-spline of order p and a B-spline of reduced order $p-d$. Indeed, if h is the distance between two adjacent knots, the following relation holds:

$$\widehat{y}^{(d)} = \sum_j c_j B_j^{(d)}(x,q) = \frac{\sum_j \Delta^{(d)} c_j B_j(x, q-d)}{h^d}, \tag{2.31}$$

where $\Delta^{(d)} c_j$ is a d^{th} order difference operator applied to the B-spline coefficients. So, if we define the d^{th} order difference matrix $\boldsymbol{D}^{(d)}$, the d^{th} derivative basis matrix can be defined as:

$$\boldsymbol{B}^{(d)}(x,q) = \frac{\boldsymbol{B}(x, q-d)\, \boldsymbol{D}^{(d)}}{h^d}.$$

2.3.2 *Penalized regression*

Penalized regression has a prominent place in modern smoothing. It combines a rich set of basis functions with a roughness penalty, to tune smoothness of the estimated curve. The penalty can be derived from classical roughness measures, like the integrated squared second derivative (O'Sullivan, 1986), or it can be discrete, working directly on the regression coefficients. An extensive discussion is presented by Eilers and Marx (2010).

Consider that a set of data $\{x,y\}_{j=1}^n$, where the vector $\boldsymbol{x}$ represents the independent (explanatory) variable and the vector $\boldsymbol{y}$ the dependent variable, has been observed. We want to describe $\boldsymbol{y}$ through an appropriate smooth function. Denote $B_j(x;q)$ the value of the jth B-spline of degree q defined on a domain spanned by equidistant knots (in case of not equally spaced knots

our reasoning can be generalized using divided differences). A curve that fits the data is given by $\widehat{y}(x) = \sum_{j=1}^{n} c_j B_j(x;q)$ where c_j (with $j = 1, ..., n$) are the estimated B-splines coefficients. Unfortunately this curve, obtained minimizing $\|\boldsymbol{y} - \boldsymbol{Bc}\|^2$ w.r.t. $\boldsymbol{c}$, shows more variation than is justified by the data if the number of spline functions is too large. To avoid this over-fitting tendency it is possible to estimate $\boldsymbol{c}$ using a generous number of bases in a penalized regression framework:

$$\widehat{\boldsymbol{c}} = \underset{\boldsymbol{c}}{\operatorname{argmin}} \|\boldsymbol{y} - \boldsymbol{Bc}\|^2 + \lambda \|\boldsymbol{Dc}\|^2, \tag{2.32}$$

where $\boldsymbol{D}$ is a d^{th} order difference penalty matrix and λ is a smoothing parameter. Second or third order difference penalties are suitable in many applications. A second order difference matrix appears as follows:

$$\boldsymbol{D}_2 = \begin{bmatrix} 1 & -2 & 1 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 1 & -2 & 1 \end{bmatrix}.$$

The optimal spline coefficients follow from (2.32) as:

$$\widehat{\boldsymbol{c}} = (\boldsymbol{B}^\top \boldsymbol{B} + \lambda \boldsymbol{D}^\top \boldsymbol{D})^{-1} \boldsymbol{B}^\top \boldsymbol{y}. \tag{2.33}$$

The smoothing parameter λ controls the trade-off between smoothness and goodness of fit. For $\lambda \to \infty$ the final estimates tend to be constant while for $\lambda \to 0$ the smoother tends to interpolate the observations. Figure 2.3 shows how different values of the smoothing parameter influence the estimated smoother (for brevity only four λ values are shown). The data were simulated by adding a Gaussian noise to a sine wave trend (200 observations). The B-spline matrix has 30 equidistant knots. Cubic B-splines and second order difference penalties have been used to estimate the smoothing functions. A P-spline smoother has some desirable properties. One of them is the conservation of moments. If we define $v_{ip} = x_i^p$ with integer p, the inner product $y^\top v_p$ defines the p^{th} moment of y. A nice property of penalized splines built using a m^{th} order difference penalty is that, for $0 < p < m$, it is true that $y^\top v_p = \widehat{y}^\top v_p$ for each value of λ.

Eilers and Marx (1996) combine a B-spline matrix with a penalty on (higher order) differences of their coefficients. If we have data on equally

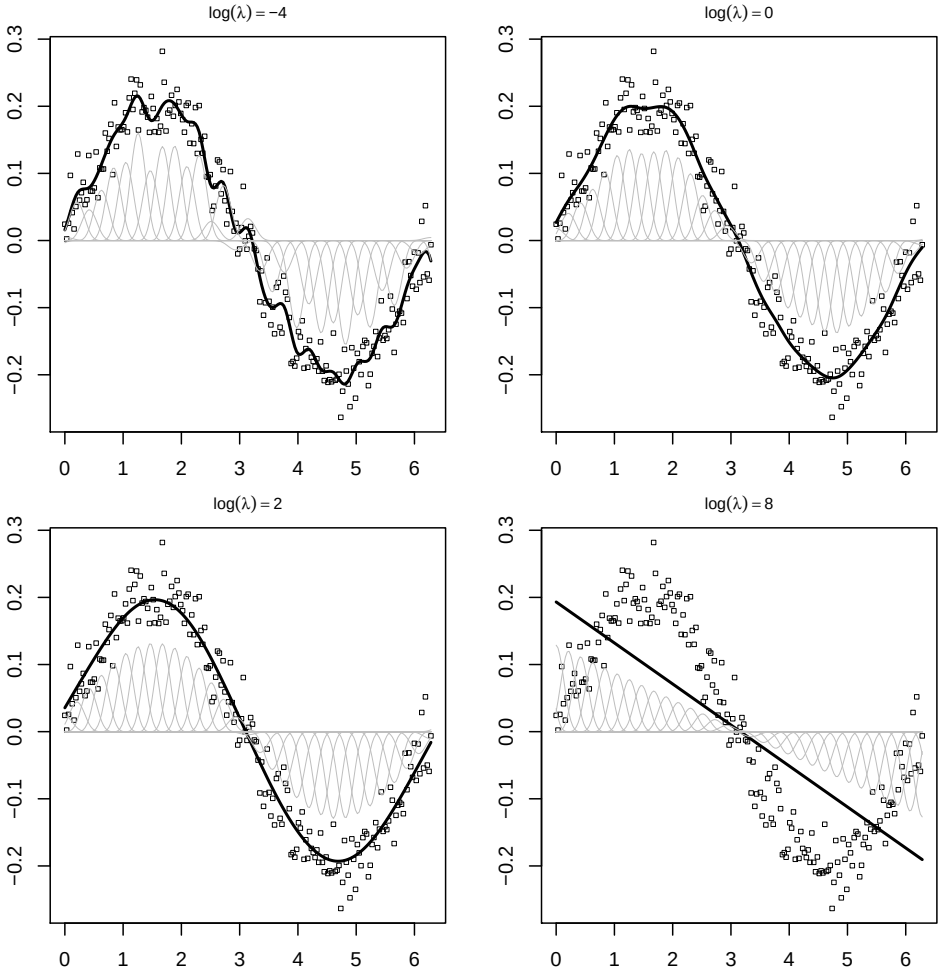

Figure 2.3: Influence of the smoothing parameter on the P-spline smoother.

spaced positions and go to the limit, we will have a basis function for each observation and the regression basis will be the identity matrix. This brings us back to the Whittaker's smoother (Whittaker, 1922; Eilers, 2003), which became popular in the econometric literature as the Hodrick-Prescott filter Hodrick and Prescott (1997). It is an attractive smoother, because effectively the basis functions disappear and with just one smoothing parameter one can move all the way from a straight line fit to essentially reproducing the data themselves. On the other hand sparse matrix algorithms allow fast computations.

The Whittaker smoother (Whittaker, 1922) can be viewed as a special case

of the P-spline smoother. It arises when the observations are located on an equally-spaced grid, a knot is placed at each abscissa point, and $\boldsymbol{B} = \boldsymbol{I}$, the identity matrix. It was proposed by Hodrick and Prescott as a tool to separate the cyclical component of a time series from raw data in order to obtain a smoothed version of the series. The smoothed time series has the advantage to be less influenced by short term fluctuations than by long term ones.
In their paper Hodrick and Prescott suggest a $\lambda = 1600$ as a good choice.
Eilers (2003) uses a leave-one-out cross-validation. Kauermann et al. (2011), work in the other direction, replacing the H-P filter with a penalized spline smoother.

2.3.3 *Smoothing parameter selection*

It is desirable to have an automatic procedure for selecting a value for the smoothing parameter. In principle many choices are available.
Popular methods for smoothing parameter selection are: the Akaike Information Criterion, Cross Validation. AIC estimates the predictive log likelihood, by correcting the log likelihood of the fitted model (Λ) by its effective dimension (ED): $AIC = 2ED - 2\Lambda$.
Following Hastie and Tibshirani (1990) we can compute the effective dimension as $ED = \text{tr}[(\boldsymbol{B}^\top\boldsymbol{B} + \lambda\boldsymbol{D}^\top\boldsymbol{D})^{-1}\boldsymbol{B}^\top\boldsymbol{B}]$ for the P-spline smoother while $ED = \text{tr}[(\boldsymbol{I} + \lambda\boldsymbol{D}^\top\boldsymbol{D})^{-1}]$ is the effective dimension of the Whittaker smoother, and

$$2\ell = -2n\ln\hat{\sigma}^2 \sum_{j=1}^{n} \frac{(y_j - \hat{y}_j)^2}{\hat{\sigma}_0^2},$$

where $\hat{\sigma}$ is the maximum likelihood estimate of σ. But $\hat{\sigma}^2 = \sum_j (y_j - \hat{y}_j^2)^2/n$, so the second term of ℓ is a constant. Hence the AIC can be written as:

$$AIC(\lambda) = 2ED + 2n\ln\hat{\sigma}. \tag{2.34}$$

The optimal parameter is the one that minimizes the value of $AIC(\lambda)$.
LOO-CV chooses the value of λ that minimizes:

$$CV(\lambda) = \sum_{j=1}^{n} \left[\frac{y_j - \widehat{y}_j}{1 - h_{jj}}\right]^2, \tag{2.35}$$

where h_{jj} is the j^{th} diagonal entry of $\boldsymbol{H} = \boldsymbol{B}(\boldsymbol{B}^\top\boldsymbol{B} + \lambda\boldsymbol{D}^T\boldsymbol{D})^{-1}\boldsymbol{B}^\top$ for P-splines or $\boldsymbol{H} = (\boldsymbol{I} + \lambda\boldsymbol{D}^\top\boldsymbol{D})^{-1}$ in case of the Whittaker smoother.

Analogous to CV is the generalized cross validation measure Wahba (1990):

$$GCV(\lambda) = \sum_{j=1}^{n} \left[\frac{y_j - \widehat{y}_j}{n - ED} \right]^2, \tag{2.36}$$

where $ED = \text{tr}(\boldsymbol{H})$. In analogy with cross validation we select the smoothing parameter that minimizes $GCV(\lambda)$.

Related to this last method is the Generalize Correlated Cross Validation procedure proposed by Carmack et al. (2012). This method exploits the possibility to modify the definition of the degrees of freedom to be taken into account in computing the GCV including the (estimated or a priori known) correlation structure in the noise component. If we denote with $\boldsymbol{C}$ this correlation matrix and with $\boldsymbol{S}$ the smoothing matrix this modified GCV approach can be computed as follows:

$$GCCV(\lambda) = \sum_{j=1}^{n} \frac{1}{n} \left[\frac{y_j - \widehat{y}_j}{1 - \text{tr}[2\boldsymbol{SC} - \boldsymbol{SCS}^\top]} \right]^2. \tag{2.37}$$

The established method for selection of the smoothing parameter have two things in common: 1) they require the computation of the effective model dimension, and 2) they are sensitive to serial correlation in the noise around the trend. The effective dimension is equal to the trace of the smoother matrix, and so inversion of a large matrix is required; for long data series this is prohibitive. Serial correlation generally leads to under-smoothing. At first sight this is surprising, but it is not hard to see why it happens. Indeed cross validation methods assume data with independent noise. If $\widehat{f}(x_e)$ is the estimated value taking into account the leave-out procedure y_e is an observation:

$$E\left[(\widehat{f}(x_e) - y_e)^2\right] = E\left[(\widehat{f}(x_e) - f(x_e))^2\right] + \sigma^2 - 2\text{Cov}[\widehat{f}(x_e), y_e],$$

which shows that the expected squared error for a cross validation term is equal to the true expected squared error plus the noise variance σ^2 minus two times the covariance between the observed data and the estimates. Even if this last term is not zero (as in the case of serial dependence in the noise component) the cross validation mis-specifies it to be equal to zero (see Carmack et al., 2012). This leads to a smoothing function that tends to consider the correlated errors as a part of the wanted signal.

To overcome these drawbacks, an alternative approach proposed by Frasso and Eilers (2015) can be considered. It based on the L-curve method for ill-posed inverse problems (Hansen, 1992; Hansen and O'Leary, 1993; Hansen, 2000).

The L-curve is a parameterized curve showing the two ingredients of every regularization or smoothing procedure: the goodness of fit and the roughness of the final estimate.

This approach was originally proposed by Hansen (1992) for the selection of the regularization parameter in ill-posed inverse problems. Regularization arises both in statistics and inverse problems applications even if the aim of the latter is slightly different. Indeed, in statistical modeling one posits a true data generating model and aims to estimate it. In inverse problems one simply look for a *good* approximate solution of a given equation without taking into account any data generating process. Also the errors are considered differently. An important parameter for the solution of inverse problems is the error upper bound level while in statistical applications the error it treated as an exogenous component (see Vogel, 1996).

We start considering ridge regression.

Ridge regression is a common regularization tool in statistics. It adds a penalty term to the standard regression problem to shrink the coefficients:

$$\underset{\beta}{\operatorname{argmin}} \|\boldsymbol{y} - \boldsymbol{X}\beta\|^2 + \lambda\|\beta\|^2,$$

$$\widehat{\beta} = (\boldsymbol{X}^\top \boldsymbol{X} + \lambda \boldsymbol{I})^{-1} \boldsymbol{X}^\top \boldsymbol{y}.$$

The strength of the shrinkage depends on λ. Define:

$$\{\omega(\lambda); \theta(\lambda)\} = \{\|\boldsymbol{y} - \boldsymbol{X}\beta\|^2; \|\beta\|^2\} \text{ and } \{\psi(\lambda); \phi(\lambda)\} = \{\log(\omega); \log(\theta)\}.$$

The L-curve is a plot of $\phi(\lambda)$ vs. $\psi(\lambda)$, parameterized by λ. Figure 2.4 shows a toy example. A sample of 200 realizations of 50 explanatory variables was drawn from a multivariate normal distribution with mean vector $\mu_i = 1$ with high correlation ($\rho \in \{0.7, 0.8, 0.9, 0.99\}$).
The dependent variable was obtained as a linear combination of the 50 independent variables plus a random noise $y_i = cX + N(0, 0.2)$ (where c_j is the regression parameter associated to each independent variable randomly drawn from a uniform distribution). The name of the curve is due to its

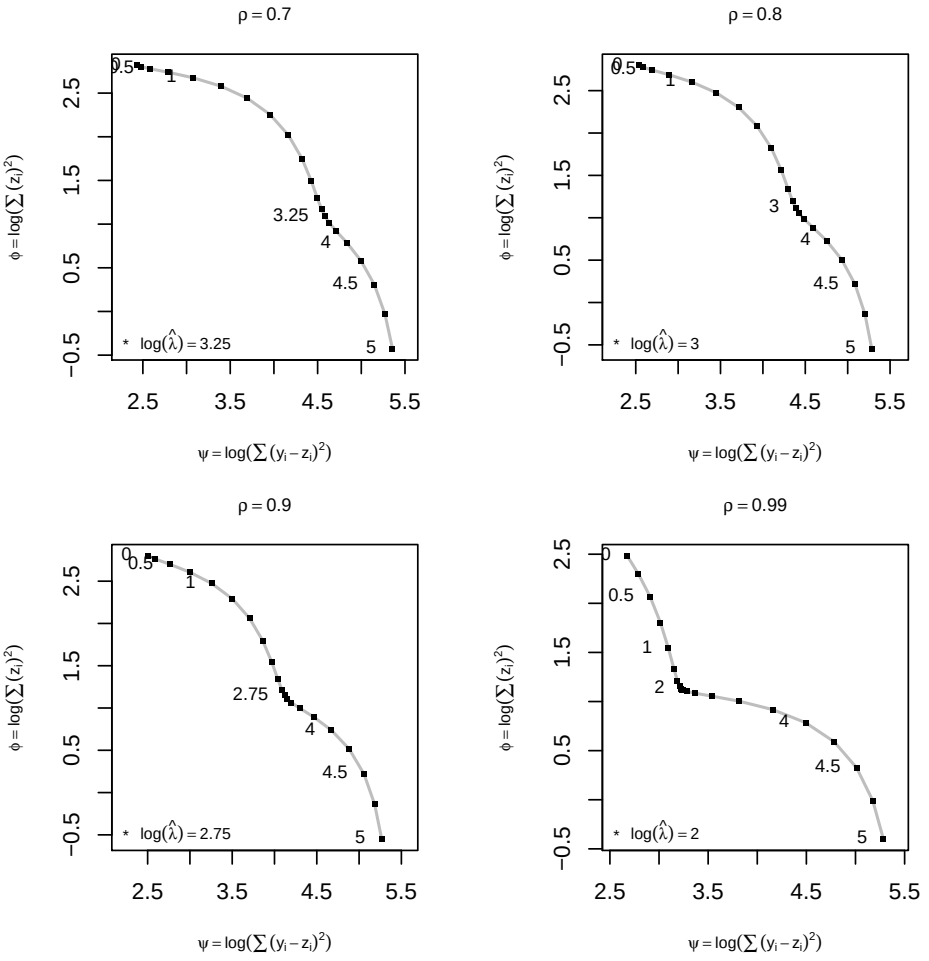

Figure 2.4: Four L-curves obtained for simulated ridge regression examples with different degrees of correlation between the explanatory variables. The corner point represents the point of maximum curvature while the other points represent the points used to draw the curve. For some points the associated logarithmic value of the parameter is shown ($\log_{10}(\lambda) \in [0, 5]$).

shape. It shows a corner in a region characterized by intermediate values of ψ, ϕ and λ. Hansen suggested to select the regularization parameter that corresponds to the corner, the point of maximum curvature. The curvature can be computed using:

$$k(\lambda) = \frac{\psi^{'}\phi^{''} - \psi^{''}\phi^{'}}{[(\psi^{'})^2 + (\phi^{'})^2]^{3/2}}. \tag{2.38}$$

The maximization of $k(\lambda)$ requires the computation of the first and second derivatives but the computations can be simplified in some cases as will be shown in what follows. It is important to notice that that while methods such as cross validation optimize the smoothing parameter minimizing a measure of the prevision accuracy, the L-curve suggests to look for the λ parameter that guarantees an optimal compromise between the residual sum of squares and the amount of penalty.
The L-curve is usually characterized by two components: a vertical and an horizontal part (consider the fourth panel of figure 2.4). The vertical part is drawn for small values of the regularization parameter. In this region the residual sum of squares is small while the amount of penalty tends to be high. The second component is the horizontal one: it is drawn for increasing values of the smoothing parameter leading to higher residual sum of squares. This flat part of the curve is characterized by a rather constant amount of penalty. If we think about an ideal L plot we can say that for a constant amount of penalty we could tune the residual sum of squares along the horizontal part of the plot. It is an ideal situation but it holds till we reach the corner of the L. At the corner we obtain the best goodness of fit for the smallest possible price (smallest penalty).

Now consider a P-spline smoother. Taking $\boldsymbol{z} = \boldsymbol{B}\alpha$, the following quantities can be defined:

$$\{\omega(\lambda); \theta(\lambda)\} = \{\|\boldsymbol{y} - \boldsymbol{B}\boldsymbol{z}\|^2; \|\boldsymbol{D}\boldsymbol{z}\|^2\},$$

and the L-curve is given by:

$$L = \{\psi(\lambda); \phi(\lambda)\} = \{\log(\omega); \log(\theta)\}. \tag{2.39}$$

The L-curve for a simple smoothing problem is depicted in figure 2.5. These

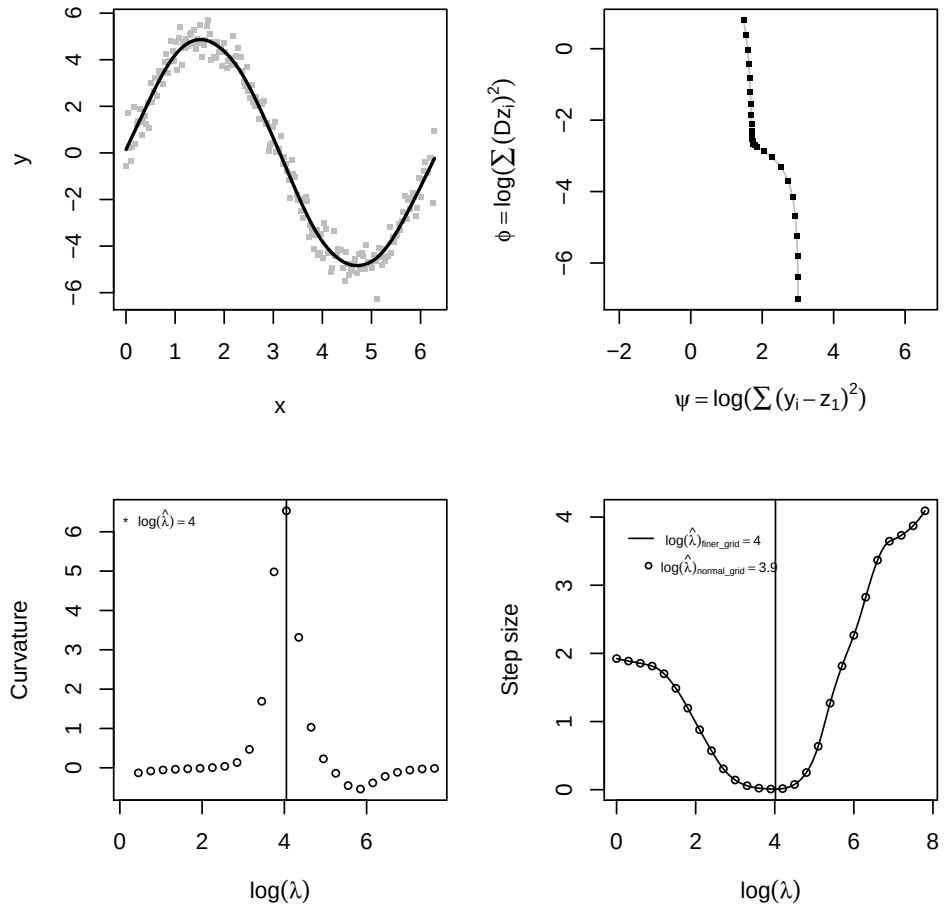

Figure 2.5: P-spline smoothing of simulated data using an L-curve approach. The first panel shows the obtained smoothing function (line) and the second the associated L-curve. The lower panels show the curvature function and Euclidean distance between adjacent points of L-curve. The values of these functions are plotted against different values of $\log_{10}(\lambda)$.

results were obtained from 200 observations simulated using the following scheme: $y = 5\sin(x) + N(0, 0.5)$ where $x \in [0, 2\pi]$.

The lower panel of figure 2.5 shows the point-wise curvature function and the Euclidean distance between adjacent points of the L-curve of the upper panel. As can be noted that the smoothing parameters selected maximizing the curvature and minimizing the Euclidean distance between adjacent points are the same. The density of the points defining the curve tends to increase moving from the tail to the corner of the L.
Frasso and Eilers (2015) exploited this characteristic to simplify the selection procedure.
The curvature function can be computed using (2.38) per each λ parameter on a given grid. By setting $\ell = \log(\lambda)$, the rate of change of the arc length distance between each point on the curve w.r.t. ℓ is given by:

$$\frac{ds}{d\ell} = \sqrt{\left(\frac{d\psi}{d\ell}\right)^2 + \left(\frac{d\phi}{d\ell}\right)^2}. \tag{2.40}$$

Minimizing it a good approximation of $\max\{k(\lambda)\}$ is obtained, when the L-curve shows a clear convex area (i.e. when there is a corner). Thus the selection procedure was simplified. The corner, if it exists, coincides (at least approximately) with the point satisfying:

$$\min\left\{\sqrt{(\Delta\psi)^2 + (\Delta\phi)^2}\right\}. \tag{2.41}$$

The criterion in (2.41) suggests that the best smoothing parameter can be selected minimizing the Euclidean distance between adjacent points on the L-curve. The Euclidean distance between these points describes a U-shaped as shown in the last panel of figure 2.5. This procedure proposed by Frasso and Eilers (2015) to select the smoothing parameter is named "V-curve".

3 VISUAL PRUNING FOR DECISION TREES

The most common approach to build a decision tree is based on a two step procedure: growing a full tree and then prune it back. The goal is to identify the tree with the lowest error rate. Alternative pruning criteria have been proposed in literature. Within the framework of recursive partitioning algorithms by tree-based methods, this paper provides a contribution on both the visual representation of the data partition in a geometrical space and the selection of the decision tree. In our visual approach the identification of the best tree and of the weakest links is immediately evaluable by the graphical analysis of the tree structure without considering the pruning sequence. The results in terms of error rate are really similar to the ones returned by the Classification And Regression Trees procedure, showing how this new way to select the best tree is a valid alternative to the well known cost-complexity pruning. [a]

***Keywords**: Classification and Regression trees, Model Selection, Pruning, Visual Representations.*

[a] This chapter has been accepted for publication as: Iorio, C., Aria, M., D'Ambrosio, A. (2014). *A new proposal for tree model selection and visualization.*

3.1 *Tree based recursive partitioning method and tree model selection: an overview*

Recursive partitioning tree procedures have been the subject of extensive research in the past. Specially tree-based methods have been proposed for both prediction and exploratory purposes. Hierarchical segmentation obtained by decision trees can be seen as a stepwise procedures performed according to some optimization criteria, that provides a progressive sequence of partitions of an initial set of objects, described by some explanatory variables (either numerical or/and categorical) and a response variable (Hastie et al., 2009), via a top down criteria. Several methods have been proposed over the years.
The oldest tree based method was Automatic Interaction Detector (AID) pro-

posed by Morgan and Sonquist (1963). Goal of AID is to grow regression trees through binary splitting rules that provide recursive reduction in unexplained sum of squares.
Messenger and Mandell (1972) and Morgan and Messenger (1973) extended AID for categorical outcome according to the so-called *theta criterion* (THAID, THeta Automatic Interaction Detector).
A descendant of AID and THAID is CHAID (CHi-square Automatic Interaction Detector), introduced by Kass (1980). CHAID uses Chi-square splitting criterion to classify a categorical response variable.
Quinlan (1979; 1987) developed an iterative algorithm, known as ID3. The input is a table of objects and each object induces a decision tree. Leaves of decision tree indicate the class to which the objects belong. ID3 uses the entropy criteria for splitting nodes.
An extension of ID3 is C4.5. It utilizes a normalized entropy measure, known as Gain Ratio, which expresses the proportion of information induced by any split (Apté and Weiss, 1997).
One of the most popular tree-based techniques is Classification And Regression Trees (CART) developed by Breiman et al. (1984). Induction of decision trees is typically performed in two steps.
In the first step, a training set (used to grow the tree) is recursively divided into subgroups according to splitting criteria expressed in terms of decrease in impurity. Often, the criterion used to split is the Gini diversity index. The tree-growing step continues until some stopping rule is reached, such as all samples for a node belong to the same class. In literature there are several proposal for tree growing step (Mola and Siciliano, 1997; Aria and Siciliano, 2003; Siciliano et al., 2008). In the second step, called pruning, the tree is reduced to prevent "overfitting". Pruning generates a decision tree by simplifying the tree structure by removing some of the branches of the fully expanded tree with the goal of improving the classification accuracy.
A generic internal node of a tree can be seen as a starting point for a sub-tree that will end with several leaves or terminal nodes. The data falling down in the leaves are evaluated via misclassification rate or expected value according to the nature of the response. As a consequence global badness of fit indices can be either the misclassification ratio or the mean squared error (Breiman et al., 1984).

Alternative pruning criteria have been proposed in the literature (Esposito et al., 1997).
The CART pruning procedure considers both the accuracy (evaluated by some error measure not necessarily coincident with the one used for the growing step) and the complexity (given by the number of terminal nodes) of the tree, introducing the so-called *cost-complexity measure* (Breiman et al., 1984). The algorithm works either by using a separate independent set of samples or by cross-validation. The goal is to produce the best sequence of pruned subtrees of the fully expanded tree. A complexity parameter needs to be defined. It represents both a penalty for any additional node and the cost associated to the removal of any terminal node belonging to a given branch. The optimal decision tree is based on the definition of a trade-off measure between the accuracy (*cost*) and the size (*complexity*) of the tree.
Quinlan suggested two methods of pruning. The first one (Quinlan, 1987) is known as *reduced error pruning* and it prunes the nodes according to a bottom-up approach. It generates a sequence of subtrees and uses the test set to evaluate the performance of the tree. Since the misclassification rate on the training set is optimistically biased, Quinlan introduced a continuity correction for the binomial distribution, that might provide a more realistic error rate. This pruning method is known as *pessimistic error pruning*. The second pruning method is implemented in C4.5 (Quinlan, 1993). It is known as *error based pruning* and produces a simplified tree-structure.
Cappelli et al. (1998) proposed an alternative pruning method based on a so called *impurity-complexity measure* which evaluates the accuracy of the classification directly through the impurity measure.
Siciliano et al. (2008) proposed a model-based tree growing that implicitly prunes the tree within the tree-growing step. This phase is based on the concept of retrospective split as well as on the recursive estimation of GLM.

3.2 *Decision tree visualization*

Fayyad et al. (2002) stated that without proper visualization techniques, data mining models may not give the desired insight to help humans to understand the phenomena.
Different visual representations of decision tree have been proposed. Hier-

archical and radial views are the two most popular graphs for decision tree (Liu and Salvendy, 2007).
Hierarchical view is the most natural way to display a decision tree.
A decision tree is defined as a directed connected acyclic graph. A graph is a set of nodes. Scheduling information is placed in the nodes. In the internal node, the information represents the splitting rule; in the terminal node, it consists on prediction. Radial view is applied manly in displaying object structures, such as organizations.
Node-link diagrams are the most familiar tree structures. This representation is poor in revealing the overall structure of a tree, such as its depth levels. In addition, it does not demonstrate node sizes.
A tree map (Shneiderman, 1992) is another way to display all the partitions using area based plot, in which each terminal node is represented by a rectangle. The rectangular area of tree-map corresponds to the full data set. This area is partitioned recursively with an alternating horizontal and vertical partitioning directions until the terminal node are reached. The size of each subrectangle is proportional to the number of cases in the corresponding node. Tree map does not allow a relative comparison of groups within nodes.
A tree ring maps the hierarchies into circles and it displays both tree topology and node size. The most inner circle represents the root node.
Tree maps and tree rings are space-filling visualization methods, since they make full use of the available space.
The icicle plot represents a tree node as a rectangle whose length is proportional to the number of records associated with it. This visualization is more space-efficient than node-link diagrams, since there are no links between nodes (Barlow and Neville, 2001).
The basic idea of circle segment visualization technique (Ankerst et al., 1996) is to display the data dimension as a segment of a circle. If the data consists on n dimension, the circle is partitioned into n segments, each segment represents one different attribute; each pixel inside a segment is a single value of the attribute. Values of each attribute are sorted independently and assigned to a different color based upon its class.
Another similar approach proposed by Ankerst et al. (2000) uses a stacked bar representation instead of circle segments. In other words, bars representing an attribute are displayed horizontally and are displayed stacked upon

each other. This technique is easily expandable to support many attributes. The circle segments method appears to start loosing display granularity as the number of segments increase.
Circle segments and similar techniques provide great visibility into multivariate classification techniques. They are a great aid in identifying obvious relationships between data values and classes and for identifying potentially weak relationships as well.

3.3 *Visual tree model selection*

In the following we introduce a visual tree model selection method. This is based on the definition of a new way to represent the tree structure by a node-link diagram where the edges of the tree have different length depending on the decrease in impurity of the response variable when passing from any node to its child. Two are the main advantages of this representation, namely adding meaningful statistical measures of the tree quality to the interpretation of the exploratory tree as well as providing a visual selection criterion to choose the best decision tree for prediction of new cases.
First, we introduce some definitions of the tree structure and then we define the proper measures to be used for the visual tree model selection.
A path of an oriented tree is a sequence of nodes and edges connecting a node with a descendant. A depth of path is the number of edges from the node to the tree's root node. A branch of a tree is any sub-tree hold by any node of the tree.
Let $\mathcal{P}_{t,z}$ be the path from the starting node t to any descendant z of the branch T_t hold by the node t. We define the *depth path* $\tilde{\mathcal{P}}_{t,z}$ as the number of edges from the descendant z to the node t of the path $\mathcal{P}_{t,z}$. Let $\mathbf{r}_{\mathcal{P}_{t,z}} = [r_k]$ be the *path index vector* where the entries r_k are the numbers denoting the nodes of the path $\mathcal{P}_{t,z}$ in the ordered list from node t to node z, for $k = 1, \ldots, K$ with $K = \tilde{\mathcal{P}}_{t,z} + 1$. For any path $\mathcal{P}_{t,z}$ we define the *path decrease in impurity* from the node t to the its descendant z within the branch T_t as:

$$V_z(T_t) = i(t)p(t) - i(z)p(z), \tag{3.1}$$

where $i(t)$ and $i(z)$ are the impurity measures of the response variable in the nodes t and z with the proportions of cases equal to $p(t)$ and $p(z)$ respectively. It is worth noting that the path decrease of impurity from the node t can be

evaluated considering any node z belonging to the branch T_t, either internal or terminal node.
It can be shown that

$$V_z(T_t) = \sum_k [i(k)p(k) - i(k+1)p(k+1)]. \tag{3.2}$$

Let t be the parent node with children nodes $2t$ and $(2t+1)$. The paths $\mathcal{P}_{t,2t}$ and $\mathcal{P}_{t,2t+1}$ have depth path equal to one by definition. These are called *one-step paths*. The edges connecting the parent node and its children nodes can be visualized with a length which can be proportional to the one-step path decrease in impurity as follows:

$$V_{2t}(T_t) = i(t)p(t) - i(2t)p(2t), \tag{3.3}$$

$$V_{2t+1}(T_t) = i(t)p(t) - i(2t+1)p(2t+1). \tag{3.4}$$

As it is known, in CART methodology the best split is selecting by maximizing the decrease in impurity when passing from node t to its children nodes. It can be shown that this choice is equivalent to maximizing the one-step paths decrease in impurity.
The proposed tree structure representation with edges of different lengths appears so asymmetric that it is possible to visualize levels of the tree corresponding to the distinct internal nodes. Specifically, for any node t of the tree T it is possible to define a measure of quality of the partitioning procedure until node t by considering the relative measure of the path decrease in impurity from the root node to its descendant t as follows:

$$V_t(T) = \frac{i(1) - i(t)}{i(1)}, \tag{3.5}$$

where $0 \leq V_t(T) \leq 1$ by definition. This can be called *node impurity proportional reduction*. An indexing measure can be associated to the tree structure describing the gain in the partitioning level by level.
Furthermore, an overall improvement of the tree structure with respect the root node can be evaluated by a weighted sum of the node impurity reductions in the terminal nodes of the tree T:

$$V(T) = \sum_{h \in H_T} V_h(T)p(h), \tag{3.6}$$

where H_T is the set of terminal nodes of the tree T.
In terms of prediction of new cases, we can evaluate the node impurity proportional reduction of any internal node of the tree on the basis of a test sample. In this way it is possible to associate another indexing measure such to identify the best cutting line to find the decision tree. It is also possible to evaluate the same measure considering a cross-validation approach.

3.4 An application of visual tree

In the framework of CART methodology we want to show how it is possible to use a visual model instead of the classic pruning procedure.
The main interpretative advantages of the visual tree are shown in an application on a real data set. We have developed the analysis of Credit data set (Decisia SPAD Repository). According to the numbering system developed by CISIA Software Informer, let k be the generic t^{th} node, it generates descendant nodes numbered as $2k$ (on the left) and $2k + 1$ (on the right). By convention, node number 1 indicates the root node.
Figure 3.1 displays the graphical representation of the CART approach. Both the maximum expanded tree and the decision tree are provided respectively on the left and the right side.
Decision tree was selected via test-set procedure, the size of test sample is about 30% of the entire data set.
As can be noted, no information is contained in length of paths and in levels of tree (Figure 3.1, left side). In the classical visualization there is a lack of information about the goodness of split, the purity of nodes and the goodness of the tree (Figure 3.1, right side).
Figure 3.2 shows the visualization of the tree structure using our approach on the Credit data set. The sub-figures provide respectively the maximum expanded tree (left side) and the decision tree (right side). In the sub-figure of the right side $\varphi_{.t}$ indicates the impurity proportional reduction related to node t as defined in equation 3.5. The pruned tree was obtained by using the same test-set used before.
In the sub-figure on the left side, the left axis measures the node impurity proportional reduction and the right axis measures the misclassification error linked to each cutting point calculated respectively on the training sample

				Visual		Cart	
Dataset	Continue	Categorical	Response	Error	Size	Error	Size
Credit	2	11	Binary	0,2489	4,516	0,2467	4,676

Table 3.1: Tree model comparison: Visual Pruning vs Classical Pruning (1000 Bootstrap replications)

(right) and test sample (left). At first sight, this plot points out the relative importance of splits and the best cut level to find an optimal decision tree.
By looking at the graph, as in a dendrogram of the hierarchical cluster analysis, we can decide an automatic cutting of the tree as a function of impurity proportional reduction. As can be noted in the right side of Figure 3.2, the tree structure highlights that the relative measure of the path decrease of impurity of terminal nodes $3, 4, 11$ is close to 0, while it is higher for node 10. Note that overall improvement of tree structure V_T, as defined in equation 3.6, is proportional to the misclassification error on the training sample $R(T)$. The visual decision tree shows the contribution of the nodes with a lower node impurity proportional reduction to the tree badness of fit. The shorter is the length path, the lower is the gain in the partitioning level by level. By visual tree model selection we can see that at the first split there is a significant decrease of impurity especially for node 3 that become immediately a terminal node (Figure 3.2, right side). The splits are the same for both trees obtained with the CART approach and the Visual Pruning for decision tree but in the visual tree model selection method the depth path explains the predictive strength of a split to decrease of impurity. This visual approach can be used to build trees both in supervised classification and in nonparametric regression.
We carried out a comparison by bootstrap and empirical evidence suggests how both procedures return similar outcomes. The results, reported in Table 3.1, show that the misclassification rate and the tree size are very similar. Both measures are referred to trees validated via test-set procedure.
Moreover in visual approach the identification of the best tree and the weakest links is immediately evaluable by the graphical analysis of the tree structure without considering the pruning sequence.

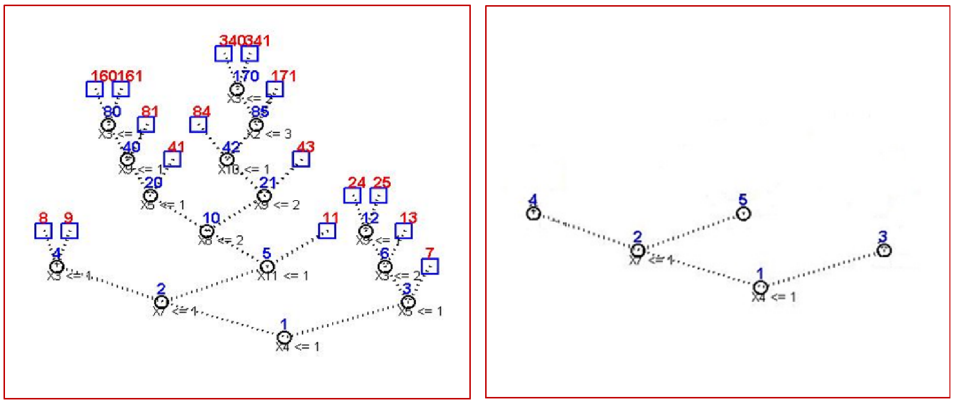

Figure 3.1: CART approach: Exploratory classification tree (on the left) and decision tree (on the right).

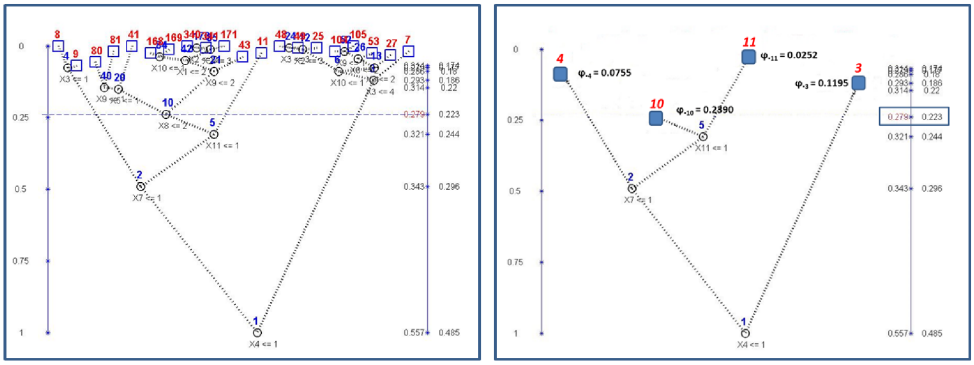

Figure 3.2: Visual tree model selection: Fully expanded tree and cutting sequence (on the left) and decision tree (on the right).

3.5 Concluding remarks

The proposed visual tree model selection seems to be a valid alternative to the cost-complexity strategy to select decision trees. We propose a new tree structure visualization that allows to identify more discriminant splits, weakest links and help the user to catch the optimal sub-structure as decision tree.

P-SPLINE BASED CLUSTERING OF CORRELATED SERIES

4

Time series arise in many scientific areas. Several clustering algorithms have been proposed. In many cases data must be preprocessed, i.e. by modeling each series with an appropriate model. In this paper, we propose a new approach exploiting a P-Spline framework. The basic idea is to perform the classification task on the reduced space spanned by optimal spline coefficients. This ensures a number of advantages, including excellent performance of resulting clustering procedure and a reduced computational time. [a]

***Keywords**: Clustering, P-splines, Time series.*

[a]This chapter has been submitted for publication as: Iorio, C., Frasso, G., D'Ambrosio, A., Siciliano, R. (2015). *P-Spline based Clustering of Correlated Series.*

4.1 Introduction

Time series can be found in different domains such as finance, economics, engineering, medicine and operations management. Most of the research on time series has been devoted to the reach of similarity measures which arises in many tasks as clustering and classification. In this paper, we focus on the clustering task. Clustering methods are popular tools in data analysis.
The goal of clustering is to identify a structure in unlabeled dataset by organizing data into homogeneous groups. Each group consists of observations that are similar among themselves and dissimilar with respect to objects of other groups. Cluster analysis has been performed mainly on static data. Due to recent interests in time series, clustering of this type of data has become an active research area with useful applications in several fields. For this reason, in the last years clustering techniques dealing with time series have been proposed. The data characteristics suggest the appropriate clustering procedure and the dissimilarity measure.
Time series can be distinguished according to their nature (real or discrete

valued, univariate or multivariate). Moreover, time series can be of equal length or unequal length due to different time domains and/or missing values.

Košmelj and Batagelj (1990) proposed a cross sectional approach to cluster multivariate time varying data, using the modified relocation clustering procedure. They used a modified relocation procedure consisting in two steps, involving first a cross-sectional approach to incorporate time dimension and then developing a specific model to determine time dependent linear weights.

Maharaj (2000) proposed an agglomerative hierarchical procedure based on the p-value of hypothesis testing applied to every pair of a given stationary time series. Maharaj assumed that each stationary time series can be fitted by a linear autoregressive model denoted by a vector of parameters. Then a chi-square distributed test statistic was derived to test the null hypothesis that there is no difference between the generating processes of two stationary time series. Two series belong to same group if the associated p-value is greater than a pre-specified significance level.

Baragona (2001) proposed some meta-heuristic methods to partition the set of time series into clusters in such a way that *(i)* the cross-correlation maximum absolute value between each pair of time series that belong to the same cluster is greater than some given threshold, and *(ii)* the k-min cluster criterion is minimized. The cross-correlation matrix function is computed from the residual of the original time series.

Fu et al. (2001) introduced an algorithm called perceptually important point (PIP) to reduce the dimensionality of the problem. Their clustering proposal belong to the framework of self-organizing map (SOM) methods.

Kumar et al. (2002) took data error into account. They presented a new scale-invariant distance function based on a Gaussian distribution of errors in data. The distance function is used in hierarchical clustering to discover meaningful clusters. In their model, time series sampled at T points are represented by a sequence of T distributions, assuming that each of these T samples are independent of each other and are distributed according to one-dimensional Gaussian distributions.

Ramoni et al. (2002) presented a Bayesian algorithm for clustering by dynamics (BCD). The method recasts the task of clustering time series as a Bayesian

model selection problem and searches for the most probable set of clusters given the observed time series. Given a set of time series, the algorithm transforms each series into a Markov chains (MC) and then clusters similar MCs to discover the most probable set of generating processes. The method applies an agglomerative clustering procedure to discover the most probable set of clusters capturing different dynamics, using an entropy-based heuristic search strategy. The task of their clustering algorithm is two-fold: finding the set of clusters that gives the best partition according to some measure, assigning each MC to one cluster. Current limitations of the BCD are that it is univariate and it can cluster discrete time series.

Möller-Levet et al. (2003) considered unevenly sampled data. Motivated by several experiments in molecular biology, the authors proposed a short time series (STS) distance to measure the similarity in shape formed by the relative change of amplitude and the corresponding temporal information of uneven sampling intervals.

Liao (2005) distinguishes among raw data based approaches, feature and models based approach. The former category works directly with raw time series replacing the distance measure for cross-sectional data with an appropriate one for time series. To compare the series, the raw data based approach consider two time series normally sampled at the same interval. However, clustering based on raw data deal with high dimensional space and very noise data. The feature and model based approach are similar. Both work with two steps. The difference between them is in the first step. The feature-based approach converts a raw time series into a feature vector of lower dimension. The model-based approach converts a raw time series into a number of model parameters. Then both apply a conventional clustering algorithm. In the framework of model-based clustering, time series are considered similar when the models characterizing individual series, or the remaining residuals after fitting the model, are similar.

In order to perform clustering of time-course gene expression data, Coffey et al. (2014) propose a new approach based on the linear mixed effects model representation of penalized spline (Ruppert, 2003). Their method exploits the connection between the linear mixed effects model and P-spline smoothing to simultaneously smooth the gene expression data to remove any measurement error/noise and cluster the expression profiles using finite mixtures of

Normal distribution, which variance depends on the estimated random effect variance.
Besides the similarity problem in time series, another issue concerns the high dimensionality that characterizes time series data in many application domains. When a large number of measurements (time points) are considered, the algorithms leads to computational problems. In the presence of data error, clustering method can be employed after that data used have been preprocessed.
Many of the algorithms mentioned above do not facilitate the removal of noise from data, have difficulties handling time series with unequal length, require operation of preprocessing of the series and do not consider series with correlated noise.
We propose to model time series by penalized spline (P-spline) smoother and to perform clustering on the estimated P-spline coefficients. The considered series can be of both equal and unequal length, unregardless the distance measure used to cluster. Our strategy can be combined with many clustering approaches.
This paper is organized as follows. Section 4.2 introduces the notation and definitions. In Section 4.3 we present our proposal. In Section 4.4 the performance of our clustering approach is evaluated through a large simulation study. In Section 4.5 an application on financial time series is presented. Section 4.6 concludes a paper with a discussion.

4.2 *Time series clustering*

Clustering approaches require two choice: the definition of a clustering algorithm and the selection of a suitable distance measures. In what follows, we introduce the notations, some clustering algorithms and the distance measure suitable for a P-Spline based clustering of time series.

4.2.1 *Notations and definitions*

Let y_i be the i^{th} series where $i = 1, \ldots, N$ is the number of series.
Let n_i be the length of i^{th}series. To avoid confusion we will use $n_i = n \; \forall i$, instead of n_i, when the series have the same length.
Let c_k be the k^{th} cluster center, where $k = 1, \ldots, K$ is the number of centers,

$1 < K < N$.
Let x_j, $j = 1, \ldots, n_i$, be the support of time points of the i^{th} series.

4.2.2 *Clustering algorithms*

Clustering techniques are traditionally divided in hierarchical and partitioning. The formers perform a hierarchical decomposition of the data set using some criterion. The latters construct various partitions of the data, then evaluate them by a clustering criterion.
A relocation clustering procedure (Košmelj and Batagelj, 1990) on time series has the following three steps:

1. Start with an initial partition, denoted by C, having the prescribed k number of clusters.

2. For each time point compute the dissimilarity matrix and store all resultant matrices for the calculation of trajectory similarity.

3. Find a partition $C^{'}$ such that $C^{'}$ is better than C in terms of the generalized Ward criterion.

The partition $C^{'}$ is obtained from C by relocating one member for C_p to C_q or by swapping two members between C_p and C_q , where $C_p, C_q \in C$ and $p, q = 1, 2, ..., l, p \neq q$. If a such partition does not exists, stop; else replace C by $C^{'}$ and repeat Step 3.
This procedure works only with time series with equal length because the distance between two time series at some cross sections (time points where one series does not have value) is ill defined (Liao, 2005).
k-means algorithm (MacQueen, 1967; Hartigan and Wong, 1979) is the most popular clustering method. It relies on iterative scheme that starts with arbitrarily chosen initial cluster centers. The procedure follows a simple way to classify a given data set through a certain number of k clusters fixed a priori. The main idea is to define k centroids, one for each cluster. As a result of this iterative procedure, the centroids change their location step by step until no more changes are done. The centroids should be placed in a running way. Since different location causes different result, the better choice is to place them as much as possible far away from each other. The next step is

to take each point belonging to a given data set and associate it to the nearest centroid. When no point is pending, the first step is completed and an early grouped is done. At this point we need to re-calculate k new centroids as centers of the clusters resulting from the previous step. After we have these k new centroids, a new binding has to be done between the same data set points and the nearest new centroid. The k-means method aims at minimizing an objective function. Given N patterns $(y_i|i = 1, ..., N)$, the k-means algorithms defines k centroid $(c_k|k = 1, ..., K)$ minimizing the following objective function:

$$J = \sum_{i=1}^{N} \sum_{k=1}^{K} \|y_i - c_k\|^2 \tag{4.1}$$

where $\|\cdot\|$ is the Euclidean distance measure between a data point and the cluster center. However, it is possible to choose other distance measures. The algorithm has the following steps:

1. Initialize K points into the space represented by the objects that are being clustered, $(2 \leq K < N)$,. These points represent initial group centroids.

2. Assign each pattern y_i to the cluster whose distance from the cluster center is minimum of all the cluster centers.

3. When all objects have been assigned, update the positions of the K cluster centers.

The procedure is stopped when the cluster centers do not move any more. Otherwise, repeat Steps 2 and 3 until convergence.
Among the model-based methods, self-organizing map (SOM) developed by Kohonen (1990) is a very popular unsupervised neural-network model. The principal goal of a SOM is to transform an incoming signal pattern of arbitrary dimension into a low dimensional discrete structure . During the self-organization process, the cluster unit whose weight matches most closely the input data is chosen as the winner. Euclidean distance is used. The weights are computed for all neurons in the network. The weights of the winning node t and its neighbors will then be updated proportionally. The formation of a clustering/map occurs in the following phases:

1. Assign random values to the initial weights w of the neuron in the network.

2. Choose input vector randomly from the input space.

3. Evaluate the network.

4. Update the weight vectors.

Depending on whether a neuron i is within a certain spatial neighborhood $N_t(l)$ around the neuron with the smallest distance, its weight is updated according to the following updating rule:

$$w_i(l+1) = \begin{cases} w_i(l) + \alpha(l)[x(l)w_i(l)] & \text{if } i \in N_t(l), \\ w_i(l) & \text{if } i \notin N_t(l). \end{cases} \tag{4.2}$$

Both the size of the neighborhood and the step size of weight adaptation α shrink monotonically with the iteration.
$k-$means and SOM are algorithms that work better with time series of equal length. In the first algorithm the concept of centroid becomes unclear when the same cluster contains series of unequal length; in the latter one difficulties arise to define the dimension of weight vectors (Liao, 2005).
Hierarchical algorithms build clusters gradually by grouping time series into a dendrogram. Hierarchical method further subdivided into agglomerative and divisive, depending upon whether it merges (bottom-up strategy) or split (top-down strategy) clusters iteratively. Most of hierarchical methods belong to agglomerative procedure. Given a set of N series to be clustered and an $(N \times N)$ distance matrix, the basic process of an agglomerative hierarchical clustering is summarized in four steps (Johnson, 1967):

1. Start by assigning each object to a cluster.

2. Find the most similar pair of clusters and merge them into a single cluster.

3. Compute distances between the new cluster and each of the old clusters.

4. Repeat Steps 2 and 3 until all items are clustered into a single cluster.

Step 3 can be done in different ways according the choice of the linkage process.
The single (complete) linkage clustering measures the distance between two clusters as the shortest (largest) distance from any member of one cluster to any member of the other cluster. The average linkage clustering measures the distance between two cluster as the average distance from any member of one cluster to any member of the other cluster. The minimum variance criterion proposed by Ward Jr (1963) minimizes the total within-cluster variance. At each step, the pair of clusters with the smallest increase in the value of sum of squares variance are merged. The algorithm at each step finds the pair of clusters that leads to minimum increase in total within-cluster variance after merging.
Hierarchical clustering is not restricted to cluster time series with equal length when appropriate distance measure (such as dynamic time warping) is used to compute the distance (Liao, 2005).
For a more detail about time series clustering one can refer Liao (2005).

4.2.3 Distance measure

Every cluster analysis require to choose a suitable distance measure In time series clustering, the concept of dissimilarity is particularly complex due to the dynamic character of the data. Dissimilarities usually considered in conventional clustering could not work well with time dependent data because they ignore the interdependence of the observations. The choice of an adequate distance measure depends largely on the nature of the clustering. Different dissimilarity measures for time series have been proposed.
Let $\mathcal{D}$ be a dataset consisting of N series $\{y_1, y_2, ..., y_N\} \subset \mathbb{R}^n$ and let K be an integer with $1 < K < N$, to partition $\mathcal{D}$ into cluster $\mathcal{C}_1, \mathcal{C}_2, ...\mathcal{C}_K$, data series are assigned to clusters using a metric. At each cluster $\mathcal{C}_K$ is associated a center c_k. Each data series is assigned to the cluster to whose center is the nearest.
The Euclidean distance between the i^{th} time series and the centroid time series of the k^{th} cluster is computed as:

$$d = \left[\sum_{j=1}^{n_i} (y_{ij} - c_{kj})^2 \right]^{1/2} . \tag{4.3}$$

Mikowski distance is a generalization of Euclidean distance, which is defined as:

$$d = \sqrt[q]{\sum_{j=1}^{n_i}(y_{ij} - c_{kj})^q}, \tag{4.4}$$

where q is a positive integer.
Pearson's correlation distance between y_i and c_k is defined as:

$$d = 1 - \rho_{(y_i,c_k)}, \tag{4.5}$$

where $\rho_{(y_i,c_k)}$ is the Pearson's correlation factor between y_i and c_k.
A common way to compare two time series is warping the time to achieve an alignment between the data points of the series. Unlike the Euclidean distance, the Dynamic Time Warping (DTW) (Berndt and Clifford, 1994) allows elastic shifting of a series to provide a better match with another series, thus it can handle time series with local time shifting and different lengths. Given any two time series $\{y_j\}_{j=1}^{n_i}$ and $\{y_s\}_{s=1}^{n_s}$, with $n_i \neq n_s$, DTW performs a non linear mapping of one sequence to the other one by minimizing the total distance between them. For doing this, a $n_i \times n_s$ matrix storing the distance $d(y_j, y_s)$ between the two point y_j and y_s is used to find an optimal warping path via a dynamic programming algorithm. Typically Euclidean distances is used. Each matrix element (j, s) corresponds to the alignment between the points y_j and y_s. A warping path, $W = w_1, w_2, w_p, ..., w_P$ is a sequence of matrix elements that defines a mapping between y_j and y_s. The p^{th} element of W is defined as $w_p = (j, s)_p$ where $max(n_i, n_s) \leq P < n_i + n_s - 1$. The warping path is typically subject to several constraints. The monotonicity forces the points in W to be monotonically spaced in time. The continuity restricts the allowable steps in W to adjacent cells (including diagonally adjacent cells). The boundary conditions requires the warping path to start and finish in diagonally opposite corner cells of the matrix. There are many warping paths that satisfy the above conditions. The interested warping path is the one which minimizes the distance between series:

$$d = \min\left(\frac{\sum_{p=1}^{P} w_p}{P}\right), \tag{4.6}$$

where P is used to compensate for the fact that warping paths may have different lengths. The dynamic programming formulation is based on the following recurrence relation which defines the cumulative distance $d_{cum}(i, j)$, for each point:

$$d_{cum}(j, s) = d(y_j, y_s) + \min(d_{cum}(j-1, s-1), \\ d_{cum}(j-1, s), d_{cum}(j, s-1)). \tag{4.7}$$

A weakness of DTW is that it aligns a single point of a series with multiple points of another series. To reduce the singularity phenomenon, Keogh and Pazzani (2001) proposed a variant of variant of DTW, named Derivative Dynamic Time Warping (DDTW). DDTW estimates the local derivatives of the data points to capture information on the trends in the sequences and to find a warping more robust to singularities.
The mentioned metrics work in the time domain, but the frequency domain approach also offers an interesting alternative to measure the dissimilarity between time series.
For a more detail about distance measures used in time series clustering one can refer Vilar et al. (2010).

4.3 The key idea

A series y can be thought as arising from a smooth underlying function $g(x)$. Time series are usually measured at a discrete number of time points $x_i, i = 1, ..., n$ and are subject to large amounts of noise. Thus it is assumed that the observed data can be modeled as:

$$y_i = g(x_i) + \varepsilon_i, \quad i = 1, ..., n \tag{4.8}$$

where $g(x_i)$ is the value of smooth series evaluated of time point x_i and ε_i is the measurement error. P-spline smoothers provide a flexible way to remove noisy measurement from the raw data and estimate the underlying smooth interesting part, that is the signal of a series.

4.3.1 Series parametrization

P-splines have been introduced by Eilers and Marx (1996) as flexible smoothing procedures combining B-splines (see e.g. de Boor, 1978) and difference

penalties.
Suppose to observe a set of data $\{x, y\}_{j=1}^{n_i}$, where the vector $\boldsymbol{x}$ indicates the independent variable (e.g. time) and the vector $\boldsymbol{y}$ the dependent one.
We want to describe the available measurements through an appropriate smooth function. Denote as $B_h(x; q)$ the value of the h^{th} B-spline of degree q defined on a domain spanned by equidistant knots (in case of not equally spaced knots our reasoning can be generalized using divided differences). Moreover different knots selection procedure has been proposed. A convenient rule is the one suggested by Ruppert (2002). A curve that fits the data is given by $\widehat{y}(x) = \sum_h a_h B_h(x; q)$ where a_h (with $h = 1, ..., \#$ of basis) are the estimated B-splines coefficients. Unfortunately the curve obtained by minimizing $\|\boldsymbol{y} - \boldsymbol{Ba}\|^2$ w.r.t. $\boldsymbol{a}$ shows more variation than is justified by the data if a dense set of spline functions is used. To avoid this over-fitting tendency it is possible to estimate $\boldsymbol{a}$ using a generous number of bases in a penalized regression framework

$$\widehat{\boldsymbol{a}} = \underset{\boldsymbol{a}}{\operatorname{argmin}} \|\boldsymbol{y} - \boldsymbol{Ba}\|^2 + \lambda \|\boldsymbol{Da}\|^2, \tag{4.9}$$

where $\boldsymbol{D}$ is a d^{th} order difference penalty matrix and λ is a smoothing parameter. Second or third order difference penalties are suitable in many applications. A second order difference matrix appears as follows:

$$\boldsymbol{D}_2 = \begin{bmatrix} 1 & -2 & 1 & \cdots & 0 & 0 \\ 0 & 1 & -2 & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & -2 & 1 \end{bmatrix}.$$

The optimal spline coefficients follow from (4.9) as:

$$\widehat{\boldsymbol{a}} = (\boldsymbol{B}^\top \boldsymbol{B} + \lambda \boldsymbol{D}^\top \boldsymbol{D})^{-1} \boldsymbol{B}^\top \boldsymbol{y}. \tag{4.10}$$

The smoothing parameter λ controls the trade-off between smoothness and goodness of fit. For $\lambda \to \infty$ the final estimates tend to be constant while for $\lambda \to 0$ the smoother tends to interpolate the observations.

The penalty approach relaxes the importance of the number and location of the knots. An extensive discussion of P-spline is presented by Eilers and Marx (2010).

Figure 4.1 shows how different values of the smoothing parameter influence the estimated smoother (for brevity only four λ values are shown). The data were simulated by adding a Gaussian noise to an exponential signal (50 observations). The B-spline matrix has 20 equidistant internal knots. Cubic B-splines and second order difference penalties have been used to estimate the smoothing functions. As noticed by Eilers and Marx (2010) the estimated P-spline coefficients in (4.10) have the desirable property to be close to the curve representing the final fit. As it appears from Figure 4.1 (encircled points) they represent the skeleton of the fitted smoother. This property plays a crucial rule in the methods described in this paper allowing for a convenient reduction of the dimensionality of our clustering task.

4.3.2 *Smoothing parameter selection*

Popular methods for smoothing parameter selection are: the Akaike Information Criterion, Cross Validation. AIC estimates the predictive log likelihood, by correcting the log likelihood of the fitted model (Λ) by its effective dimension (ED): AIC $= 2\text{ED} - 2\Lambda$. Following Hastie and Tibshirani (1990) we can compute the effective dimension as ED $= \text{tr}[(\boldsymbol{B}^\top\boldsymbol{B} + \lambda\boldsymbol{D}^\top\boldsymbol{D})^{-1}\boldsymbol{B}^\top\ \boldsymbol{B}]$ for the P-spline smoother and

$$2\ell = -2n\ln\hat{\sigma}^2\sum_{j=1}^{n_i}\frac{(y_j-\hat{y}_j)^2}{\hat{\sigma}_0^2},$$

where $\hat{\sigma}$ is the maximum likelihood estimate of σ. But $\hat{\sigma}^2 = \sum_j(y_j-\hat{y}_j^2)^2/n_i$, so the second term of ℓ is a constant. Hence the AIC can be written as

$$\text{AIC}(\lambda) = 2\text{ED} + 2n\ln\hat{\sigma}. \tag{4.11}$$

The optimal parameter is the one that minimizes the value of $AIC(\lambda)$.
LOO-CV chooses the value of λ that minimizes

$$\text{CV}(\lambda) = \sum_{j=1}^{n_i}\left[\frac{y_j-\widehat{y}_j}{1-h_{jj}}\right]^2, \tag{4.12}$$

where h_{jj} is the j^{th} diagonal entry of $\boldsymbol{H} = \boldsymbol{B}(\boldsymbol{B}^\top\boldsymbol{B} + \lambda\boldsymbol{D}^\top\boldsymbol{D})^{-1}\boldsymbol{B}^\top$.
Analogous to CV is the generalized cross validation measure (Wahba, 1990)

$$\text{GCV}(\lambda) = \sum_{j=1}^{n_i}\left[\frac{y_j-\widehat{y}_j}{n-\text{ED}}\right]^2, \tag{4.13}$$

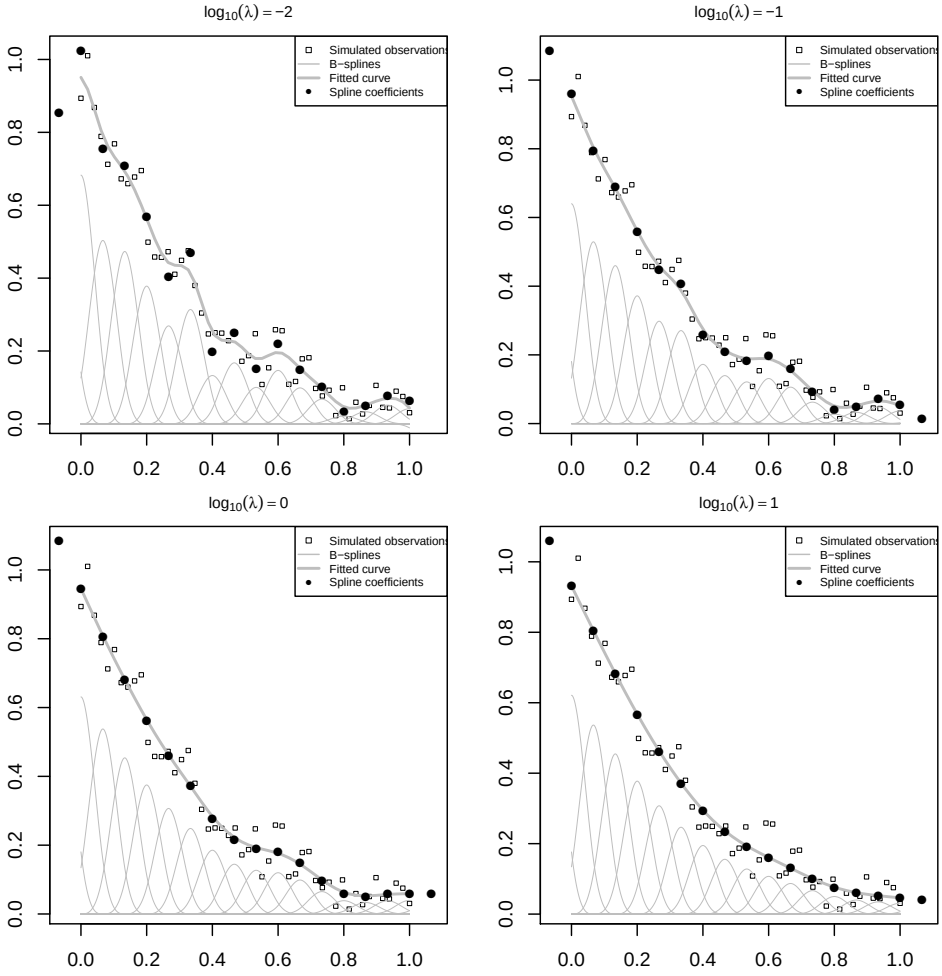

Figure 4.1: Influence of the smoothing parameter on the P-spline smoother and related spline coefficients.

where ED $= \text{tr}(\boldsymbol{H})$. In analogy with cross validation we select the smoothing parameter that minimizes GCV(λ).
All these selection procedures suffer of two drawbacks: 1) they require the computation of the effective model dimension which can become time consuming for long data series, and 2) they are sensitive to serial correlation in the noise around the trend. The L-curve (Hansen, 1992) and the derived V-curve criteria (Frasso and Eilers, 2015) overcome these hitches. The L-curve is a parameterized curve comparing the two ingredients of every regularization or smoothing procedure: badness of the fit and roughness of the final estimate. For a P-spline smoother, the following quantities can be defined

$$\{\omega(\lambda); \theta(\lambda)\} = \{\|\boldsymbol{y} - \boldsymbol{B}\widehat{\boldsymbol{a}}\|^2; \|\boldsymbol{D}\widehat{\boldsymbol{a}}\|^2\}.$$

The L-curve is obtained by plotting $\psi(\lambda) = \log(\omega)$ against $\phi(\lambda) = \log(\theta)$. This plot typically shows a L-shaped curve and the optimal amount of smoothing is located in the corner of the "L" by maximizing the local curvature measure

$$\kappa(\lambda) = \frac{\psi^{'}(\lambda)\phi^{''}(\lambda) - \psi^{''}(\lambda)\phi^{'}(\lambda)}{[\psi^{'}(\lambda)^2 + \phi^{'}(\lambda)^2]^{3/2}}. \tag{4.14}$$

The V-curve criterion offers a valuable simplification of the searching criterion by requiring the minimization of the Euclidean distance between the adjacent points on the L-curve.

4.3.3 Clustering procedure

We propose to model time series by penalized spline smoothers and performing clustering directly on the spline coefficients. As discussed in Eilers and Marx (2010), the P-spline coefficients are close to the fitted curve and present the skeleton of the fit. We suggest the following clustering steps. We fit each series by a P-splines. We obtain the matrix $\mathbf{A} = [\widehat{a}_{i,h}]$, whose elements represent the optimal P-spline coefficients. It is worth noting that the dimensionality of the problem reduces from $\mathbb{R}^n$ to $\mathbb{R}^{q+o}$, where q is the number of internal knots and o is the order of B-spline. From the matrix $\mathbf{A}$ we construct a suitable partition with homogeneous groups and center representative $\boldsymbol{c}_k = \{c_k\}_{k=1}^K$, where each c_k belong to $\mathbb{R}^{q+o}$. Any clustering algorithms and distance measure can be used to cluster the coefficients of P-spline. Series with different lengths (e.g. with missing values) can be handled by P-splines due to the extrapolation properties.

4.4 Simulation study

To examine the performance of our proposal two simulation studies were carried out. The first one consider series with equal length. The second one deal with series of different length.
We generated $K = 6$ clusters of numerical series at $n = 500$ equally spaced time points in $[0, 1]$ as described in Coffey et al. (2014). Distinct cluster specific models were used:

$$
\begin{aligned}
y^{(1)}_{i_{(j)}} &= \alpha_i + \sin(\beta_i * \pi * x_{j_{(i)}}) + \gamma_i + \varepsilon_{ij} \\
y^{(2)}_{i_{(j)}} &= x_{j_{(i)}} + (\delta_i)^{-3} + \iota_i + \gamma_i + \varepsilon_{ij} \\
y^{(3)}_{i_{(j)}} &= \nu_i + \gamma_i + \varepsilon_{ij} \\
y^{(4)}_{i_{(j)}} &= \zeta_i + \cos(\zeta_i * \pi * x_{j_{(i)}}) + \gamma_i + \varepsilon_{ij} \\
y^{(5)}_{i_{(j)}} &= \xi_i - \eta_i * \exp(-\theta_i * x_{j_{(i)}}) + \gamma_i + \varepsilon_{ij} \\
y^{(6)}_{i_{(j)}} &= -3(x_{j_{(i)}} - 0.5) + \gamma_i + \varepsilon_{ij}
\end{aligned}
$$

where:
$\alpha_i \sim N(\sqrt{2}; \sigma_e^2)$ with $\sigma_e^2 = 0.08$, $\beta_i \sim N(4 * \pi; \sigma_e^2)$, $\delta_i \sim N(0.75; \sigma_e^2)$,
$\iota_i \sim N(1; \sigma_e^2)$, $\nu_i \sim N(0; \sigma_e^2)$, $\zeta_i \sim N(2; \sigma_e^2)$, $\xi_i \sim N(2; \sigma_v^2)$ with $\sigma_v^2 = 0.85$,
$\eta_i \sim N(4; \sigma_v^2)$, $\theta_i \sim N(6; \sigma_e^2)$, $\gamma_i \sim N(0; \sigma_u^2)$ with σ_u^2 ranging from 0.3 to 1 and ε_{ij} is an autoregressive model of order 1.
Cluster means were chosen to reflect the situation where there are series that show little variation in value over time (as given by cluster 3) and series which have distinct signal over time. Cluster sizes were equal to 90, 50, 100, 25, 60 and 35, for cluster $1, 2, 3, 4, 5, 6$ respectively, giving a total number of 360 simulated series. One hundred data set were generated using the above scheme. An example of data set is plotted in Fig.4.2. Our proposal was implemented in R. To perform a cluster we use K-means algorithm. For this example we chose to use Pearson's correlation distance. We set a configuration with 50 random starting point and 50 replicates. For each cluster analysis the adjusted Rand Index (ARI) by Hubert and Arabie (1985) was computed in order to evaluate the degree of similarity between the estimated clusters and the known (true) cluster labels for each of the 100 data

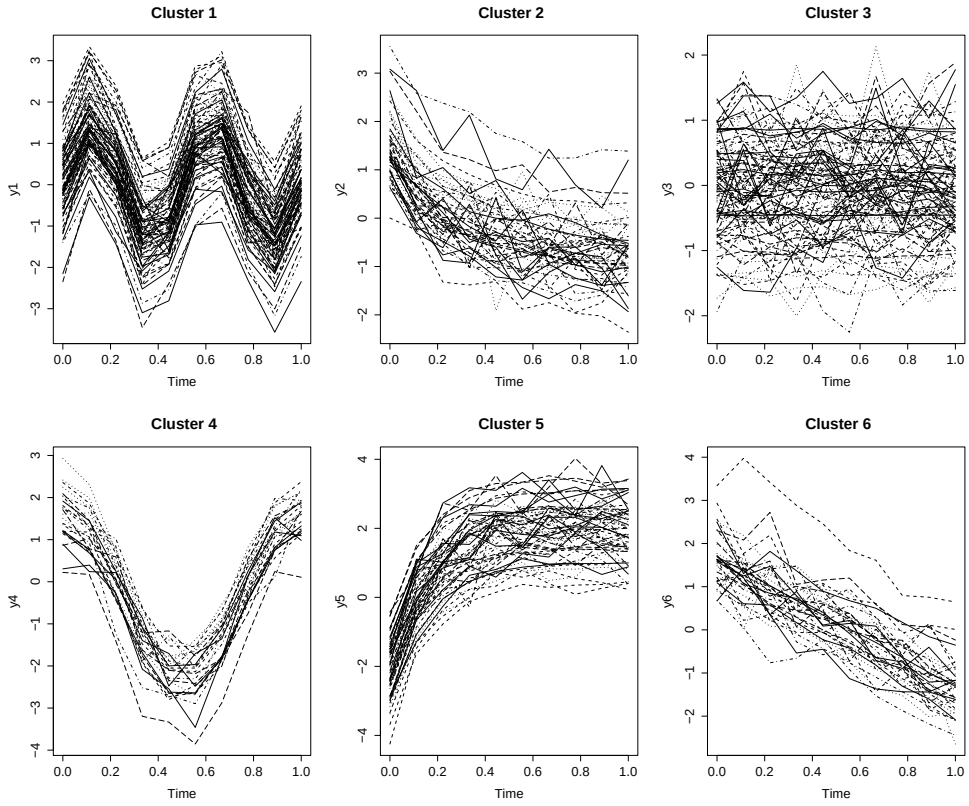

Figure 4.2: Example data set (one of 100) generated for simulation study.

set. In addition, the computation time was recorded. To check the sensitivity of our procedure with respect to a different number of basis spline, we performed four analyses for each simulated data set. We considered respectively $10\%, 20\%, 30\%, 40\%$ of the length of the simulated series. The degree of B-spline is the same for each analysis. Table 4.1 shows the result of our simulation study. We can notice that the clustering performance are not dramatically influenced by the choice of the number of internal knots. The average ARI remains above 90% even with a limited number of bases. As expected an increasing number of knots requires a large computational effort. The proposed method is compared with that of Coffey et al. (2014) which is implemented in R (package `MFDA`). The effect of our proposal were investigated. Figure 4.3 displays boxplots of computational time required by the clustering procedure to fit the true number of clusters $K = 6$, the corresponding ARI for the `MFDA` package and for our approach based on using k-means and

	Mean ARI	Sd ARI	Average Time
10%	0.9499769	0.07696571	5.3083
20%	0.9568504	0.07282104	12.4815
30%	0.9634295	0.06863666	16.2428
40%	0.9815861	0.05073459	25.7644

Table 4.1: Mean value and standard deviation of ARI and mean computational times related to clustering procedure used for simulated data set. The values are reported for each procedure using a number of inner knots equal to $10\%, 20\%, 30\%, 40\%$ of length of series.

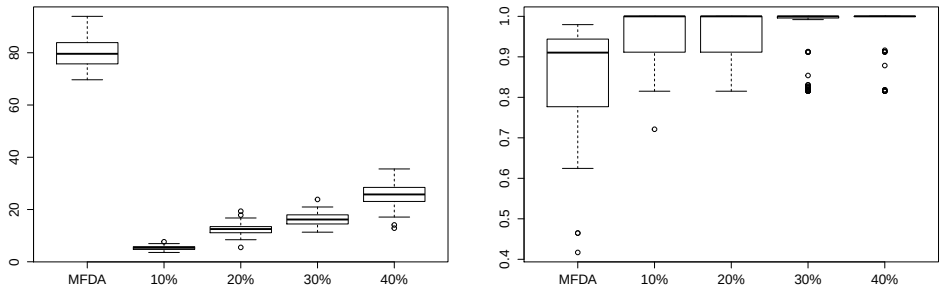

Figure 4.3: Boxplots of computational time required by the clustering procedure to fit $K = 6$ clusters (left side) and corresponding ARI values (right side).

considering a number of internal knots equal to $10\%, 20\%, 30\%, 40\%$ of the length of the simulated series. It can be seen that our proposed method has significantly lower computation time. There is also no evidence of any loss of performance associated with the decrease of computing burden since the average ARI values increases with time. These results show that the proposed method has significantly a time saving and a good performance compared to the proposal of Coffey et al. (2014). To examine the performance of our method to deal with series with different length we use the same class of function of previous simulation study. The number of simulated series and the number of simulated data set are also the same. In each dataset 15% of missing data were randomly taken from a Normal distribution with mean equal to 75 and variance equal to 5. Again, to perform a cluster we use K-means algorithm and Pearson's correlation distance. The configuration setting is equal to the one of the previous study. For each cluster analysis the adjusted Rand Index (ARI) by Hubert and Arabie (1985) was computed in order to evaluate the degree of similarity between the estimated clusters and the known (true) cluster. In addition, the computation time was also

	Mean ARI	Sd ARI	Average Time
10%	0.9315027	0.06047894	2.2199
20%	0.900624	0.04707609	4.7399
30%	0.9045141	0.05734862	7.0961
40%	0.9107914	0.072089451	7.2088

Table 4.2: Mean value and standard deviation of ARI and mean computational times related to clustering procedure used for simulated data set with different length. The values are reported for each procedure using a number of inner knots equal to $10\%, 20\%, 30\%, 40\%$ of length of series.

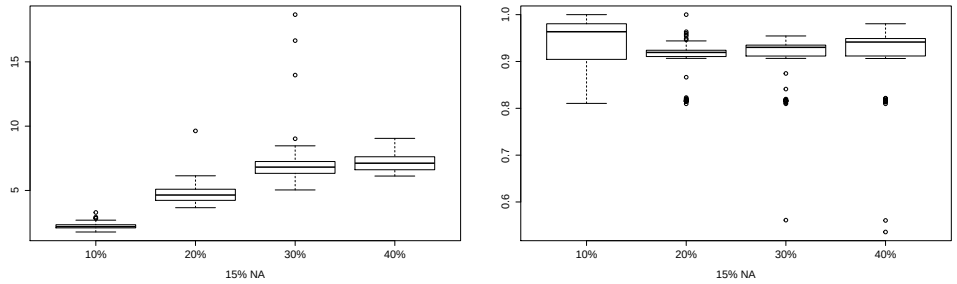

Figure 4.4: Boxplots of computational time required by the clustering procedure to fit $K = 6$ clusters (left side) and corresponding ARI values (right side) for series with different length.

recorded.

Table 4.2 shows the results of the simulation study. The average ARI index is lower than the one obtained in previous simulation study with equal length. However, the proposed method shows a good performance even in presence of series with different length.

Figure 4.4 displays the boxplots of the computation time taken and the corresponding ARI values using a number of basis spline equal to $10\%, 20\%, 30\%, 40\%$ of the length of the simulated series. As it can be noted as the number of basis increase, the computational time increases. Surprisingly, the maximum average ARI was recorded in the setting with 10% of interior knot. In his case, there is also the maximum variability. Nevertheless, the average ARI remains above 90% even with dealing with series of different length.

4.5 *Clustering of financial time series*

The proposed technique was then applied to a financial time series with the aim to build a portfolio. The data are provided by *yahoo.finance.com* and were

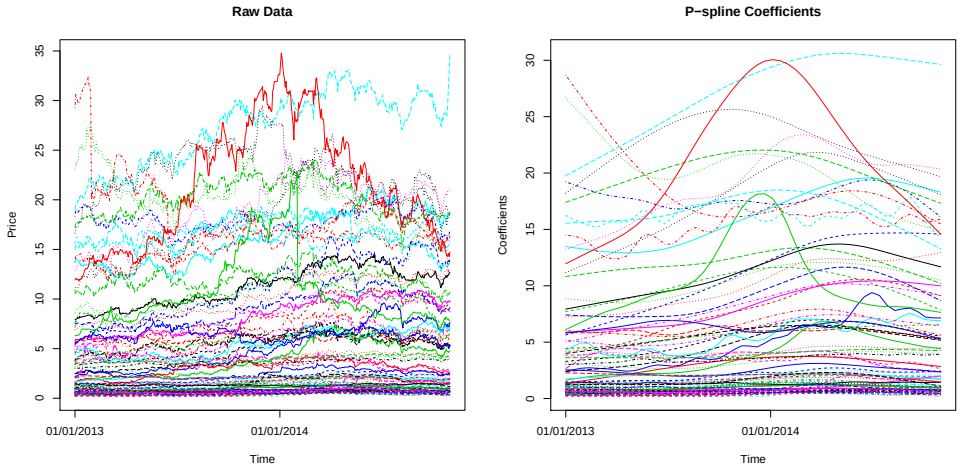

Figure 4.5: Financial time series containing closing price of 86 stocks belonging to FTSE Italia All-Share Index listed on Borsa Italiana during the period 01/01/2013 to 10/30/2014(left side) and their estimated P-spline coefficients (right side).

collected daily from 01/01/2013 to 10/30/2014. The financial time series contain the closing price of the FTSE Italia All-Share Index that consist of stocks listed on the MTA and MIV markets of Borsa Italiana (BIt) that meet minimum size and liquidity criteria. The index measures the performance of the major capital and industry segments of the Italian market. The FTSE Italia All-Share Index comprises all the constituents from the FTSE MIB, FTSE Italia Mid Cap and FTSE Italia Small Cap, and captures approximately 95% of the domestic market capitalization. The FTSE MIB Index is capped at 15% and the FTSE Italia Mid Cap and Small Cap Indices are capped at 10%. The FTSE Italia All-Share and All-Share Sector Indices are not capped. The indices use a transparent, rules-based construction process. Index Rules are freely available on the FTSE website. Index constituents are categorized in accordance with the Industry Classification Benchmark (ICB), the global standard for industry sector analysis. In our analysis we choose 86 financial time series belonging to 10 different industry sectors.

We model each financial time series by P-spline and the computed fit is described by its coefficients. The number of basis spline, with cubic degree, was chosen according to Ruppert (2002) and it is equal to 40.

The Figure 4.5 illustrates the financial time series used for the application and the estimated P-spline coefficients for each series of closing prices. For this

application we used the Euclidean distance. To select the number of clusters a Silhouette index validation method (Kaufman and Rousseeuw, 2009) is adopted. According to this validity index, the optimal number of clusters chosen for k-means algorithm on P-spline coefficient of each closing prices series is 3. The 3 groups classify the series according to the price dynamics. The cluster 1 contains the stocks with higher price than the other groups. In this cluster the minimum price is equal to 10.85 and the maximum price is equal to 34.75. The cluster 2 contains the stocks with smaller price than the other groups. In this cluster the minimum price is equal to 0.17 and the maximum price is equal to 12.93. The cluster 3 contains the stock with "moderate" price respect to the other groups. After the classification, the assets can be selected from these cluster to build a portfolio. To reach this aim, we computed the return for financial time series and chose the cluster containing the stocks minimizing risk for a given level of expected return. One of the most referenced risk/return measures, the Sharpe ratio 1966, is then computed. It is a risk-adjusted measure of the portfolio return performance. Figure 4.6 shows it for the 13 stocks belonging to cluster 1, Figure 4.7 for the 52 stocks belonging to cluster 2 and Figure 4.9 for 21 stocks belonging to cluster 3. The cluster 1 and 2 contains the stocks with smaller and higher expected return and risk compared to other clusters, respectively. The cluster 3 presents a risk-return trade-off of the stocks in between two other cluster. Moreover, cluster 3 contains 6 stocks (IF, BSS, REC, AVE, VAS, CAI) with Sharpe index higher.
Modern portfolio theory suggests to consider assets in a diversified portfolio that have correlations of returns less than one with each other because in order to decrease portfolio risk without sacrificing return. Such diversification produce an increase in the Sharpe ratio. The correlation coefficient is negative only for the for stocks of cluster 3. This suggest to select the assets belonging to cluster 3 for portfolio. Thus, to start with to perform a cluster analysis on P-spline coefficients of closing prices series is an efficient strategy to build a portfolio.

Within the stocks belonging to cluster 3 we choose the following. The selection was based on a higher Sharpe ratio (IF, BSS, REC, ACE, VAS, CAI) and lower correlation criteria (SOL, AGL, MARR, PCP). The 10 selected stocks also belong to different industrial sectors improving the risk exposure of our portfolio.

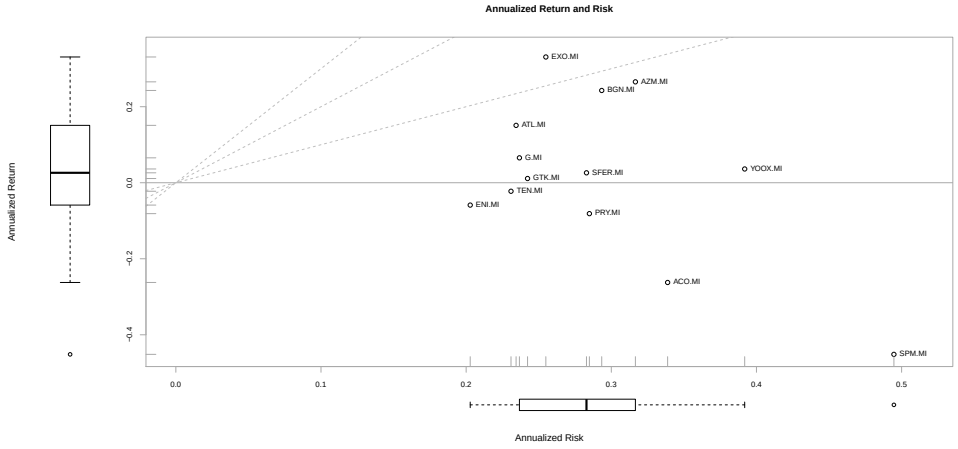

Figure 4.6: Scatter chart of annualized returns versus annualized risk (standard deviation) for comparing performance of stocks belonging to Cluster 1

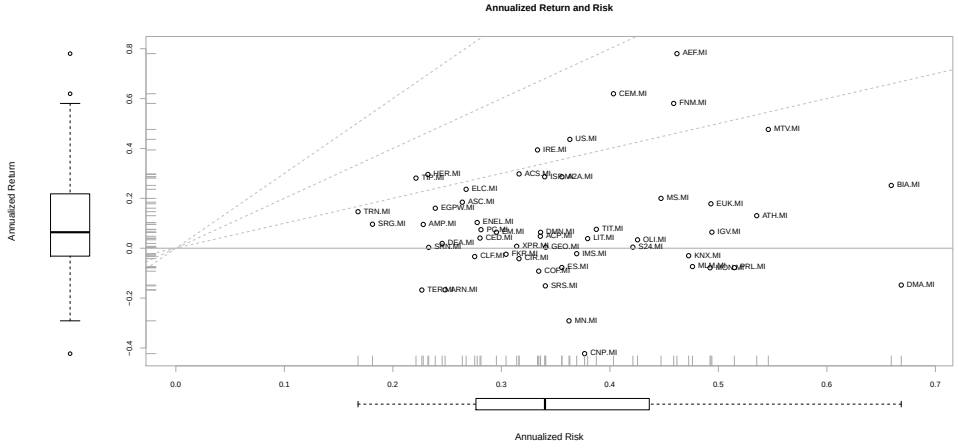

Figure 4.7: Scatter chart of annualized returns versus annualized risk (standard deviation) for comparing performance of stocks belonging to Cluster 2

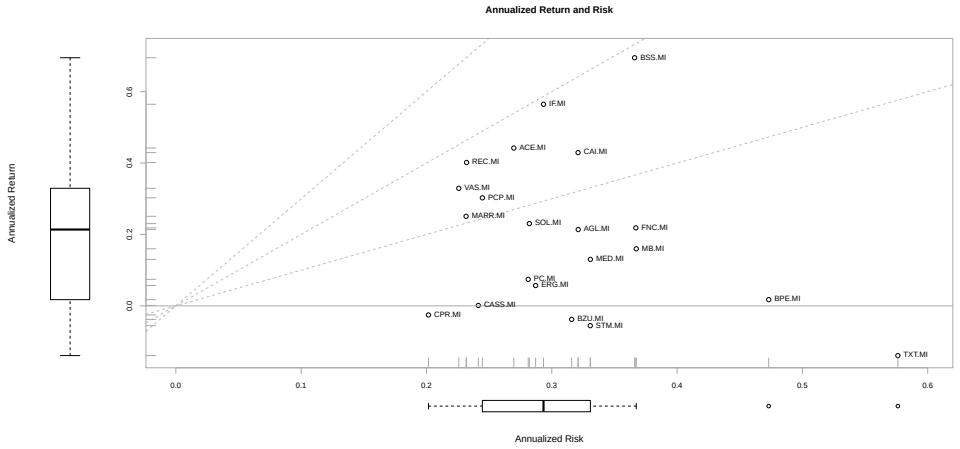

Figure 4.8: Scatter chart of annualized returns versus annualized risk (standard deviation) for comparing performance of stocks belonging to Cluster 3

The process of efficiently allocating wealth among stocks, has a longstanding history in the academic literature.

Markowitz (1952, 1956) formalized the problem in a mean-variance framework where one assumes that the rational investor seeks to maximize the expected return for a given volatility level. This solution has drawbacks in its practical implementation (Maillard et al., 2008). First, optimal portfolios tend to be excessively concentrated in a limited subset of the full set of assets or securities. Second, the mean-variance solution is overly sensitive to the input parameters. Small changes in those parameters, most notably in expected returns (Merton, 1980), can lead to significant variations in the composition of the portfolio.

Alternative methods to deal with these issues have been suggested in the literature.

The minimum variance portfolio selects a specific portfolio on the mean-variance efficient frontier. This portfolio is easy to compute since the solution is unique. As the only mean-variance effcient portfolio not incorporating information on the expected returns as a criterion, it is also recognized as robust.

Benartzi and Thaler (2001) attribute the same weight to all the assets considered for inclusion in the portfolio. If all assets have the same correlation coefficient as well as identical means and variances, the equally-weighted

Asset	MV	EW	ERC
IF	0.066696	0.1	0.090712
BSS	0	0.1	0.065481
REC	0.180253	0.1	0.123514
ACE	0.098225	0.1	0.097693
VAS	0.114629	0.1	0.107322
CAI	0.024891	0.1	0.081550
SOL	0.165993	0.1	0.123447
AGL	0.014189	0.1	0.078979
MARR	0.171747	0.1	0.117327
PCO	0.163379	0.1	0.113975

Table 4.3: Weights of our portfolio according to Mean-Variance (MV), Equally Weighted (EW) and Equally Weighted Risk Contributions (ERC) methods.

	MV	EW	ERC
$E(R_{pf})$	0.38	0.44	0.42
$\sigma(R_{pf})$	0.13	0.15	0.14

Table 4.4: Return and Volatility of our portfolio

portfolio is the unique portfolio on the efficient frontier.
Maillard et al. (2008) proposed another heuristic approach, which constitutes a middle-ground stemming between minimum variance and equally-weighted portfolios. The authors equalize risk contributions from the different stocks. The risk contribution of a asset is the share of total portfolio risk attributable to it. It is computed as the product of the allocation in component with its marginal risk contribution, the latter one being given by the change in the total risk of the portfolio induced by an infinitesimal increase in holdings of component.
Dealing with risk contributions has become standard practice for institutional investors, under the label of "risk budgeting": the analysis of portfolio in terms of risk contribution than asset weights. In Table 4.3 and in Table 4.4 are displayed the weights and the expected return and volatility for our portfolio, respectively. As one can notice the Equally Weighted Risk Contributions (ERC) method represents an interesting alternative compared to the other two allocation strategies. Unlike the Minimum Variance (MV) portfolio, the ERC portfolio is includes all the assets. To evaluate the performance of these asset allocation strategy during the considered period we compute the cumulative returns of the portfolios in according to the three methods.
Table 4.5 reports for each portfolio its performance indicators. One differ-

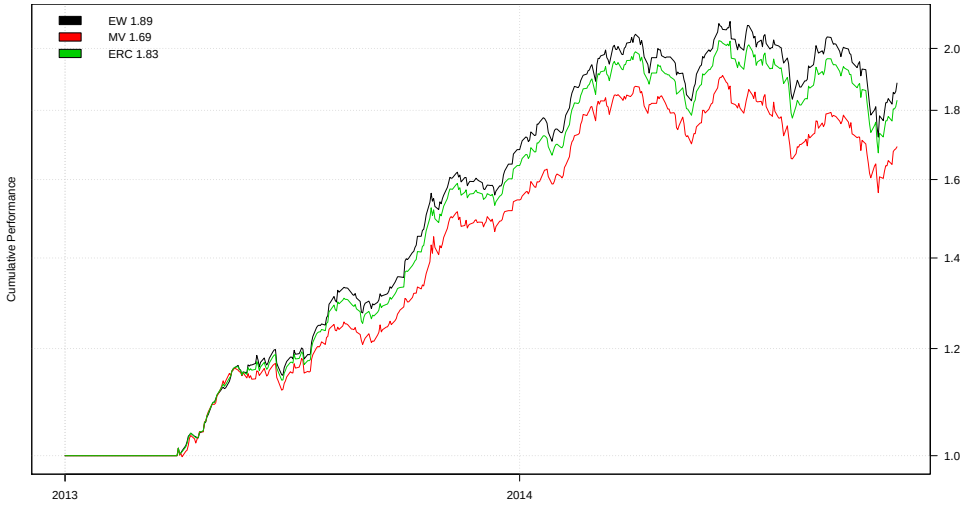

Figure 4.9: Cumulative Performance of our portfolio built according to Mean-Variance (MV), Equally Weighted (EW) and Equally Weighted Risk Contributions (ERC) allocation strategy to compare the performance of 10 selected stocks of Cluster 3 chosen for our portfolio.

	MV	EW	ERC
CAGR	33.28	41.41	39.14
Sharpe Index	2.07	2.43	2.40
Volatility	13.85	14.16	13.63
Max DD	-18.09	-17.88	-17.39
VaR	-1.41	-1.39	-1.35
Exposure	86.64	86.64	86.64

Table 4.5: Performance indicators of the three strategy for our portfolio (01/01/13 - 10/30/14).

ence among the strategies is in Compound Annual Growth Rate (CAGR). It is a *pro forma* investment yield on an annually compounded basis. However, the CAGR does not take the volatility into account. Indeed, we notice that the portfolio built by MV and ERC strategies have smaller volatility than the EW portfolio. ERC portfolio has more stable returns during the considered period. Its Maximum DrawDown (MaxDD), that measures the largest peak-to-trough decline in the value of a portfolio (before a new peak is achieved), is lower than the other strategies. The performance of the ERC portfolio are very similar to that obtained from its competitor EW. However the Value at Risk (VaR), that measures the potential loss in value of the portfolio over the defined period for a 99% confidence level, is lower for ERC portfolio.

According to Maillard et al. (2008) we can chose the ERC portfolio. In summary, this result suggested that the our proposed method can not only iden-

tify clusters of stock prices with different profiles over time, but can be also useful to build a financial portfolio.

4.6 Concluding remarks

This paper presented a new approach to cluster analysis of data (time) series. We model each series by penalized spline (P-spline) smoothers (Eilers and Marx, 1996) and we perform a cluster analysis on the estimated coefficients. When series are observed in different time domain as they are of different length, we perform cluster analysis on fitted values of a reduced time domain common to all series.

P-spline smoothers separate the signal of series from the noise, capturing the different shapes. P-spline coefficients are close to the fitted curve and present the skeleton of the fit (Eilers and Marx, 2010). Thus, summarizing each series by spline coefficients reduces the dimensionality of the problem, saving computational time without reduction in performance of clustering procedure. To select the smoothing parameter we adopt a V-curve procedure (Frasso and Eilers, 2015). This criterion does not require the computation of the effective model dimension and it is insensitive to serial correlation in the noise around the trend. Our propoal can be adopted within several clustering frameworks.

To evaluate the performance of our approach we conducted two analysis considering series with equal and unequal length, respectively. We simulated series as in Coffey et al. (2014). The results of both analysis evaluated in terms of adjusted Rand Index (ARI) (Hubert and Arabie, 1985) show excellent performance even with a limited number of bases. The average ARI remains above 90% even with 15% of the missing data. Furthermore, the proposed method ensures a decrease of the computational effort. Finally our method was applied to a financial time series with the aim to build a portfolio that can help financial practitioners to support their investment decisions.

BOOSTED-ORIENTED PROBABILISTIC CLUSTERING OF SERIES

5

Fuzzy clustering methods allow the objects to belong to several clusters simultaneously, with different degrees of membership. However, a factor that influences the performance of fuzzy algorithms is the value of fuzzifier parameter. In this paper, we propose a fuzzy clustering procedure for data (time) series that does not depend on the definition of a fuzzifier parameter. It comes from two approaches, theoretically motivated for unsupervised and supervised classification cases, respectively. The first is the Probabilistic Distance (PD) clustering procedure. The second is the well known Boosting philosophy. Our idea is to adapt the boosting approach to unsupervised learning problems, specially to non hierarchical cluster analysis. The aim is to assign each instance (i.e. a series) of a data set to a cluster. The representative instance of a given cluster (i.e. the cluster center) can be assumed as a target instance, a loss function can be assumed as a synthetic index of the global performance, the probability of each instance to belong to a given cluster can be assumed as the individual contribution of a given instance to the overall solution. The global performance of the proposed method is investigated by various experiments evaluated by using a fuzzy variant of the Rand Index. [a]

Keywords: *Boosting, Cluster validity, Dissimilarity measure, Fuzzy Clustering, P-Spline, Time series.*

[a] This chapter has been submitted for publication as: Iorio, C., Frasso, G., D'Ambrosio, A., Siciliano, R. (2015). *Boosted-Oriented Probabilistic Clustering of Series.*

5.1 Introduction

Our aim is to define a clustering technique for data (time) series belonging to the category of fuzzy (probabilistic) clustering. The goal of clustering is to discover groups so that objects within a cluster have high similarity among them, and at the same time they are dissimilar to objects in other clusters.

Many clustering algorithms for time series have been introduced in the literature. Since clusters can formally be seen as subsets of the data set, one possible classification of clustering methods can be according to whether the subsets are fuzzy (soft) or crisp (hard). Let $\mathcal{D}$ be a data set consisting of N series $\{y_1, y_2, ..., y_N\} \subset \mathbb{R}^n$ and let K be an integer, with $2 \leq K < N$, the goal is to partition $\mathcal{D}$ into $\mathcal{C}_K$ groups. Crisp clustering methods are based on classical set theory, and restrict that each object of data set belongs to exactly one cluster. It means partitioning the data $\mathcal{D}$ into a specified number of mutually exclusive clusters $\mathcal{C}_1, \mathcal{C}_2, ...\mathcal{C}_K$.
A hard partition of $\mathcal{D}$ can be defined as a family of subsets $\mathcal{C}_k$ that satisfies the following properties (Bezdek, 1981):

$$\bigcup_{k=1}^{K} \mathcal{C}_k = \mathcal{D},$$

$$\mathcal{C}_k \cap \mathcal{C}_h = \emptyset,\ k \neq h$$

$$\emptyset \subset \mathcal{C}_k \subset \mathcal{D},\ 1 \leq k \leq K.$$

Let μ_{ik} be the membership function and let $\mathbf{U} = [\mu_{ik}]$ be the $N \times K$ partition matrix. The elements of $\mathbf{U}$ must satisfy the following conditions:

$$\mu_{ik} \in \{0, 1\},\ 1 \leq k \leq K,\ 1 \leq i \leq N;$$

$$\sum_{k=1}^{K} \mu_{ik} = 1;$$

$$0 < \sum_{i=1}^{N} \mu_{ik} < N.$$

The k^{th} column of $\mathbf{U}$ contains value of μ_{ik} of the k^{th} subset $\mathcal{C}_k$ of $\mathcal{D}$.
In a hard partition, $\mu_k(y_i)$ is the indicator function:

$$\mu_k(y_i) = \begin{cases} 1, & \text{if } y_i \in \mathcal{C}_k; \\ 0, & \text{otherwise.} \end{cases}$$

Following Bezdek (1981) the hard partitioning space is thus defined by:

$$M_c = \{\mathbf{U} \in \mathbb{R}^{K \times n} | \mu_{ik} \in \{0, 1\}, \forall i, k; \sum_{k=1}^{K} \mu_{ik} = 1, \forall i, 0 < \sum_{i=1}^{N} \mu_{ik} < N, \forall k\},$$

M_c being the space of all possible hard partition matrices for $\mathcal{D}$.
Generalizing the crisp partition, $\mathbf{U}$ is a fuzzy partitions of $\mathcal{D}$ with elements μ_{ik} of the partition matrix bearing real values in $[0, 1]$ Kaufman and Rousseeuw (2009).
The idea of fuzzy set was conceived by Zadeh (1965). Fuzzy clustering methods allow the objects to belong to several clusters simultaneously, with different degrees of membership. In contrast to hard clustering, each object will have a membership value in every cluster. The larger is the value of the membership value for a given object with respect to a cluster, the larger is the probability of that object to be assigned to that cluster.
Similarly to crisping conditions, the conditions for a fuzzy partitions are given by Ruspini (1970):

$$\mu_{ik} \in [0, 1],\ 1 \leq k \leq K,\ 1 \leq i \leq N;$$

$$\sum_{k=1}^{K} \mu_{ik} = 1;$$

$$0 < \sum_{i=1}^{N} \mu_{ik} < N.$$

Finally, the fuzzy partitioning space is the set:

$$M_f = \{\mathbf{U} \in \mathbb{R}^{K \times n} | \mu_{ik} \in [0, 1], \forall i, k; \sum_{k=1}^{K} \mu_{ik} = 1, \forall i, 0 < \sum_{i=1}^{N} \mu_{ik} < N, \forall k.\}$$

Several clustering criteria have been proposed to identify fuzzy partition in $\mathcal{D}$. Among these proposals, the most popular method is fuzzy c-means.
Proposed by Dunn (1973) and developed by Bezdek (1981), fuzzy c-means considers each data point as a possible member of multiple clusters with a membership value. This algorithm is based on minimization of the following objective function:

$$J_m = \sum_{i=1}^{N} \sum_{k=1}^{K} (\mu_{ik})^m \|y_i - c_k\|^2 \tag{5.1}$$

s.t.

$$\mu_{ik} \in [0,1],\ \forall i,k;$$
$$\textstyle\sum_{k=1}^{K} \mu_{ik} = 1;$$
$$0 < \textstyle\sum_{i=1}^{N} \mu_{ik} < N.$$

In the equation (5.1), m is any real number greater than 1, μ_{ik} is the degree of membership of y_i in the cluster k. and $\|\cdot\|$ is any norm expressing the similarity between any measured data and the center. The parameter m is called *fuzzifier* or *weighting coefficient*. To perform fuzzy partitioning, the number of clusters and the weighting coefficient have to be chosen. The procedure is carried out through an iterative optimization of the objective function shown above, with the update of membership value μ_{ik} and the cluster centers c_k by solving:

$$c_k = \frac{\sum_{i=1}^{N} (\mu_{ik})^m y_i}{\sum_{i=1}^{N} (\mu_{ik})^m}, \quad k = 1, \ldots, K. \tag{5.2}$$

$$\mu_{ik} = \left(\sum_{h=1}^{K} \left(\frac{\sqrt[(m-1)]{\|y_i - c_k\|^2}}{\sqrt[(m-1)]{\|y_i - c_h\|^2}} \right) \right)^{-1} \quad i = 1, \ldots, N; \quad k = 1, \ldots, K. \tag{5.3}$$

The algorithm is synthesized in box 1.

One of limitations of fuzzy c-means clustering is the value of fuzzifier m. A large fuzzifier value suppresses outliers in data sets, i.e. the larger m, the more clusters share their objects and vice-versa. For $m \to \infty$ all data objects have identical membership to each cluster, for $m = 1$, the method becomes equivalent to k-means. The role of the weighting exponent has been well investigated in literature.

Pal and Bezdek (1995) suggested taking $m \in [1.5, 2.5]$.

Dembélé and Kastner (2003) obtain the fuzzifier with an empirical method calculating the coefficient of variation of a function of the distances between all objects of the entire data set.

Yu et al. (2004) proposed a theoretical upper bound for m that can prevent the sample mean from being the unique optimizer of a fuzzy c-means objective functions.

Box 1 Fuzzy c-means algorithm

Initialize: K = number of centers, $m, (1 < m < \infty)$, ε = a small threshold. Set the counter $l = 1$ and initialize the matrix of the fuzzy c-partitions $\mathbf{U} = [\mu_{ik}^{(l)}]$.
while $|\mathbf{U}^{(l+1)} - \mathbf{U}^{(l)}| > \varepsilon$ **do**
- Calculate the cluster center, $c_k^{(l)}$ by using equation (5.2).
- Update the membership matrix $\mathbf{U} = [\mu_{ik}]$ by using equation (5.3), if $y_i \neq c_k^{(l)}$,
otherwise set $\mu_{ik} = 1$ if $l = i$ or set $\mu_{ik} = 0$ if $l \neq i$.
- Compute $|\mathbf{U}^{(l+1)} - \mathbf{U}^{(l)}|$.
if $|\mathbf{U}^{(l+1)} - \mathbf{U}^{(l)}| > \varepsilon$ **then**
- Set $l = l + 1$
end if
end while
output: estimated centers $\widehat{c}_k$, membership matrix $\mathbf{U}$.

Futschik and Carlisle (2005) search for a minimal fuzzifier value for which the cluster analysis of the randomized data set produces no meaningful results, by comparing a modified partitions coefficient for different values of both parameters.

Schwämmle and Jensen (2010) showed that the optimal fuzzfier takes values far from the its frequently used value equal to 2. The authors introduced a method to determine the value of the fuzzifier without using the current working data set. Then for high dimensional ones, the fuzzifier value depends directly on the dimension of data set and its number of objects. For low dimensional data set with small number of objects, the authors reduce the search space to find the optimal value of the fuzzifier. According to the authors, this improvement helps choosing the right parameter and saving computational time when processing large data set.

On the basis of a robust selection analysis of the algorithm, Wu (2012) founds that a large value of m will make fuzzy c-means algorithm more robust to noise and outliers. The author suggested to use value of fuzzifier ranging between 1.5 and 4.

Since the weighting coefficient determines the fuzziness of the resulting classification, we propose a method that is independent from the choice of the fuzzifier. It comes from two approaches, theoretically motivated for un-

supervised and supervised classification cases respectively. The first is the Probabilistic Distance (PD) clustering procedure defined by Ben-Israel and Iyigun (2008). The second is the well known Boosting philosophy. From the PD approach we took the idea of determining the probabilities of each series to any of the k clusters. As this probability is unequivocally related to the distance of each series from the centers, there are no degrees of freedom in determine the membership matrix. From the Boosting approach (Freund and Schapire, 1997) we took the idea of weighting each series according some measure of badness of fit in order to define an unsupervised learning process based on a weighted re-sampling procedure. This paper is organized as follows: Section 5.2 contains our proposal, in Section 5.3 the results of some experimental evaluation studies are carried out and some concluding remarks are presented in Section 5.4.

5.2 *Boosted-oriented probabilistic clustering of time series*

5.2.1 *The key idea*

The boosting approach is based on the idea that a supervised learning algorithm (weak learner) improves its performance by learning from its errors (Freund and Schapire, 1997). It consists of an ensemble method that work with a re-sampling procedure (Dietterich, 2000). The general idea consists in running several times the supervised learning algorithm and assigning a weight to each instance of a data set that governs the re-sampling (with replacement) process during the iterations. The weights are set in such a way that the misclassified instances gets a weight larger than the weight assigned to well classified instances. In this way, the probability to be included in the sample during the iterations is higher for those instances for which the supervised learning algorithm returns a wrong classification. There exist boosting algorithms for both classification and regression problems (Freund and Schapire, 1997; Dietterich, 2000; Eibl and Pfeiffer, 2002; Gey and Poggi, 2006). In both cases the weighting system combine a synthetic index of the performance of the supervised learning algorithm with some index that represents the individual contribution of a given instance to the overall solution. Our idea is to adapt the boosting philosophy to unsupervised learning problems, specially to non hierarchical cluster analysis. In such a case there not exists

a target variable, but as the goal is to assign each instance (i.e. a series) to a cluster, we have a target instance. The representative instance of a given cluster (i.e. the cluster center) can be assumed as a target instance, a loss function to be minimized can be assumed as a synthetic index of the global performance, the probability of each instance to belong to a given cluster can be assumed as the individual contribution of a given instance to the overall solution. In contrast to the boosting approach, the larger is the probability of a given series to be member of a given cluster, the larger is the weight of that series in the re-sampling process. As a learner we use a P-spline smoother. To define the probabilities of each series to belong to a given cluster we use the PD clustering approach (Ben-Israel and Iyigun, 2008). This approach allows us to define a suitable loss function and, at the same time, to propose a fuzzy clustering procedure that does not depend on the definition of a fuzzifier parameter.

5.2.2 *A brief introduction to P-splines*

P-splines have been introduced by Eilers and Marx (1996) as flexible smoothing procedures combining B-splines (de Boor, 1978) and difference penalties. Suppose to observe a set of data $\{x, y\}_{j=1}^{n}$, where the vector $\boldsymbol{x}$ indicates the independent variable (e.g. time) and $\boldsymbol{y}$ the dependent one. We want to describe the available measurements through an appropriate smooth function. Denote $B_h(x; p)$ the value of the h^{th} B-spline of degree p defined on a domain spanned by equidistant knots (in case of not equally spaced knots our reasoning can be generalized using divided differences). A curve that fits the data is given by $\widehat{y}(x) = \sum_h a_h B_h(x; p)$ where a_h ()with $h = 1, ..., \# \, of \; basis$) are the estimated B-splines coefficients. Unfortunately the curve obtained by minimizing $\|\boldsymbol{y} - \boldsymbol{Ba}\|^2$ w.r.t. $\boldsymbol{a}$ shows more variation than is justified by the data if a dense set of spline functions is used. To avoid this over-fitting tendency it is possible to estimate a using a generous number of bases in a penalized regression framework:

$$\widehat{\boldsymbol{a}} = \underset{\boldsymbol{a}}{\operatorname{argmin}} \|\boldsymbol{y} - \boldsymbol{Ba}\|^2 + \lambda \|\boldsymbol{Da}\|^2, \tag{5.4}$$

where $\boldsymbol{D}$ is a d^{th} order difference penalty matrix and λ is a smoothing parameter. Second or third order difference penalties are suitable in many applica-

tions.
The optimal spline coefficients follow from (5.4) as:

$$\widehat{\boldsymbol{a}} = (\boldsymbol{B}^\top \boldsymbol{B} + \lambda \boldsymbol{D}^\top \boldsymbol{D})^{-1} \boldsymbol{B}^\top \boldsymbol{y}. \tag{5.5}$$

The smoothing parameter λ controls the trade-off between smoothness and goodness of fit. For $\lambda \to \infty$ the final estimates tend to be constant while for $\lambda \to 0$ the smoother tends to interpolate the observations.
Popular methods for smoothing parameter selection are: the Akaike Information Criterion, Cross Validation. AIC estimates the predictive log likelihood, by correcting the log likelihood of the fitted model (Λ) by its effective dimension (ED): AIC $= 2\text{ED} - 2\Lambda$. Following Hastie and Tibshirani (1990) we can compute the effective dimension as ED $= \text{tr}[(\boldsymbol{B}^\top \boldsymbol{B} + \lambda \boldsymbol{D}^\top \boldsymbol{D})^{-1} \boldsymbol{B}^\top \boldsymbol{B}]$ for the P-spline smoother and

$$2\ell = -2n \ln \hat{\sigma}^2 \sum_{j=1}^{n} \frac{(y_j - \hat{y}_j)^2}{\hat{\sigma}_0^2},$$

where $\hat{\sigma}$ is the maximum likelihood estimate of σ. But $\hat{\sigma}^2 = \sum_j (y_j - \hat{y}_j^2)^2/n$, so the second term of ℓ is a constant. Hence the AIC can be written as

$$\text{AIC}(\lambda) = 2\text{ED} + 2n \ln \hat{\sigma}.$$

The optimal parameter is the one that minimizes the value of $AIC(\lambda)$.
LOO-CV chooses the value of λ that minimizes

$$\text{CV}(\lambda) = \sum_{i=j}^{n} \left[\frac{y_j - \widehat{y}_j}{1 - h_{jj}} \right]^2,$$

where h_{jj} is the j^{th} diagonal entry of $\boldsymbol{H} = \boldsymbol{B}(\boldsymbol{B}^\top \boldsymbol{B} + \lambda \boldsymbol{D}^\top \boldsymbol{D})^{-1} \boldsymbol{B}^\top$.

Analogous to CV is the generalized cross validation measure Wahba (1990)

$$\text{GCV}(\lambda) = \sum_{j=1}^{n} \left[\frac{y_j - \widehat{y}_j}{n - \text{ED}} \right]^2, \tag{5.6}$$

where ED $= \text{tr}(\boldsymbol{H})$. In analogy with cross validation we select the smoothing parameter that minimizes GCV(λ).

All these selection procedures suffer of two drawbacks: 1) they require the computation of the effective model dimension which can become time

consuming for long data series, and 2) they are sensitive to serial correlation in the noise around the trend. The L-curve (Hansen, 1992) and the derived V-curve criteria (Frasso and Eilers, 2015) overcome these hitches. The L-curve is a parameterized curve comparing the two ingredients of every regularization or smoothing procedure: badness of the fit and roughness of the final estimate. For a P-spline smoother, the following quantities can be defined

$$\{\omega(\lambda); \theta(\lambda)\} = \{\|\boldsymbol{y} - \boldsymbol{B}\widehat{\boldsymbol{a}}\|^2; \|\boldsymbol{D}\widehat{\boldsymbol{a}}\|^2\}.$$

The L-curve is obtained by plotting $\psi(\lambda) = \log(\omega)$ against $\phi(\lambda) = \log(\theta)$. This plot typically shows a L-shaped curve and the optimal amount of smoothing is located in the corner of the "L" by maximizing the local curvature measure

$$\kappa(\lambda) = \frac{\psi^{'}(\lambda)\phi^{''}(\lambda) - \psi^{''}(\lambda)\phi^{'}(\lambda)}{[\psi^{'}(\lambda)^2 + \phi^{'}(\lambda)^2]^{3/2}}. \tag{5.7}$$

The V-curve criterion offers a valuable simplification of the searching criterion by requiring the minimization of the Euclidean distance between the adjacent points on the L-curve.

5.2.3 PD clustering approach

Let $\mathcal{D}$ be a dataset consisting of N series $\{y_1, y_2, ..., y_N\} \subset \mathbb{R}^n$ and let $\mathcal{C}_k$ be k^{th} cluster, with $k \in (1, K)$, partitioning $\mathcal{D}$. We suppose that each series has the same domain of length n.

At each cluster $\mathcal{C}_k$ is associated a cluster center c_k, with $k = 1, ..., K$.
Let $d_{i,k} = d(y_i, c_k)$ be a distance function of the ith series from the kth cluster center.
Let $P_{i,k} = P(y_i, \mathcal{C}_k)$ be the probability of the ith series belonging to the kth cluster.
For each series $y \in \mathcal{D}$ and each cluster $\mathcal{C}_k$, we assume the following relation between probabilities and distances (Ben-Israel and Iyigun, 2008):

$$P_{i,k} d_{i,k} = constant. \tag{5.8}$$

The constant in (5.8) only depends on series y and it is independent of the cluster k. Equation (5.8) allows to to define the membership probabilities as (Heiser, 2004; Ben-Israel and Iyigun, 2008) as:

$$P_{i,k} = \frac{\prod_{i \neq j} d_{j,k}}{\sum_{k=1}^{K} \prod_{i \neq j} d_{i,k}}. \tag{5.9}$$

In contrast to the proposal of Ben-Israel and Iyigun, any suitable distance measure can be used.

5.2.4 *The algorithm*

Since the probabilities as defined in equation (5.9) sum up to one among the clusters, we used the quantity $\prod_{k=1}^{K} P_{i,k}$ as a measure of badness representation of the i^{th} series with respect to the overall solution of the clustering procedure. It is easy to note that $\prod_{k=1}^{K} P_{i,k} = 0$ if the i^{th} series coincides with the k^{th} cluster center, as well as $\prod_{k=1}^{K} P_{i,k} = K^{-1}$ if there is maximum uncertainty in assigning the i^{th} series to any cluster center. For this reason we use as measure of badness of cluster solution the quantity:

$$BC = \frac{1}{N} \sum_{i=1}^{N} (\prod_{k=1}^{K} P_{i,k}) K^{K}. \tag{5.10}$$

Equation (5.10) is a synthetic measure of badness of clustering procedure: the lower is its value, the better is the solution. It equals zero when there is a perfect solution (i.e., each series has probability equal to one to belong to some cluster center). The maximum possible value of equation (5.10) is 1, when each series has probability equal to K^{-1} to belong to each of the K cluster. The BC index allows to compare the overall clustering solution when the number K of the clusters differs.

We define the loss function to be minimized during the iterative process as:

$$\beta = \sum_{i=1}^{N} (\prod_{k=1}^{K} P_{i,k}) K^{K}. \tag{5.11}$$

Let $\gamma_{i,k} = d_{i,k} / max_{k=1}^{K} d_{i,k}$ be the contribution of the i^{th} series to generate th k^{th} cluster.

Let $\mathbf{\Gamma}$ be a $N \times K$ indicator matrix whose entries are 1 if $P_{i,k} > P_{i,h}$ $(k, h =$

$1, \ldots, K$, $k \neq h$) and -1 otherwise.
We define the weight of the i^{th} series for the k^{th} cluster as

$$w_{i,k} = \beta^{\gamma_{i,k}\Gamma_{i,k}}.$$

For each cluster k, the weights are first normalized in this way:

$$w^{\bullet}_{i,k} = \frac{w_{i,k}}{\sum_{h=1}^{K} w_{i,h}},$$

then within each cluster we set

$$W_{i,k} = \frac{w^{\bullet}_{i,k}}{\sum_{i=1}^{N} w^{\bullet}_{i,k}}. \tag{5.12}$$

For each cluster k, a sample $\mathcal{L}^{(k)}$ is extracted with replacement from $\mathcal{D}$, taking in account the weights as defined in equation (5.12). Then the cluster centers $\widehat{\mathbf{c}}_k = \mathbf{B}\widehat{\mathbf{a}}_k$, $k = 1, \ldots, K$ are estimated by using a P-spline smoother. These centers are then used to compute the membership probabilities as in equation (5.9) for the next iteration. The cluster centers are re-estimated and adaptively updated with an optimal spline smoother.
The choice of the metric depends on the nature of the series, the optimal P-spline smoothing procedure frames our approach in the framework of model-based clustering but it frees the user from the choice of the optimal model to be chosen for each series. Box 2 shows the pseudo-code of our the Boosted-Oriented P-Spline Probabilistic Clustering algorithm.
The procedure described in box 2 is repeated a certain number of times due to the sensitivity of final solution to the random choice of cluster center.

5.3 *Experimental evaluation*

To evaluate the performance of the proposed algorithm, we conducted three experiments. In estimating the optimal P-splines smoother, we always used the L-curve criterion as in equation (5.7) to select the optimal λ parameter, and we used a number of interior knots equal to $\min(\frac{n}{4}; 40)$, in which n is the length of time domain, as suggested by Ruppert (2002). Moreover we need a measure of goodness of fuzzy partitions. To reach this aim, we decided to use a fuzzy variant of the Rand Index proposed by Hullermeier et al. (2012).

Box 2 Boosted-oriented P-spline probabilistic clustering of time series

input $\mathcal{D}$
initialize: maxiter = maximum number of iterations; K = the number of clusters; d = a suitable distance measure; c_k, $k = 1 : \ldots, K$ random cluster centers.
for iter=1:maxiter **do**
- compute the $N \times K$ distance matrix $\mathbf{D} = [d_{i,k}]\ \forall i, k$;
- compute the membership probabilities $\mathbf{P} = [P_{i,k}]\ \forall i, k$ as in equation (5.9);
- compute $\beta^{[iter]}$ as in equation (5.11);
- assign the weights to each series for each cluster and compute the $N \times K$ matrix W as in equation (5.12);

for $k = 1 : K$ **do**
- extract the sample $\mathcal{L}^k$ from $\mathcal{D}$
- compute center $\widehat{\mathbf{c}}_k^{[iter]} = \mathbf{B}\widehat{\mathbf{a}}_k$

end for
if iter $= 1$ **then**
- $\widehat{\mathbf{c}}_k^* = \mathbf{B}\widehat{\mathbf{a}}_k$

else
for $k = 1 : K$ **do**
- update cluster centers $\widehat{c}_k^* = B\widehat{a}_k^*$, with $\widehat{\mathbf{a}}_k^* = (\mathbf{B}^\top\mathbf{B} + \lambda\mathbf{D}^\top\mathbf{D})^{-1}\mathbf{B}^\top\widehat{\mathbf{c}}_k^{[1:iter]}$

end for
end if
end for
output: estimated cluster centers $\widehat{\mathbf{c}}_k^*$, membership probabilities matrix $\mathbf{P}$.

This index is defined by the complement to 1 of the normalized sum of degree of discordance. The Rand index developed by Rand (1971) is a *external evaluation measure* to compare the clustering partitions on a set of data. The problem of evaluating the solution of a fuzzy clustering algorithm with the Rand index is that it requires converting the soft partition into a hard one, losing information.

As shown in Campello (2007), different fuzzy partitions describing different structures in the data may lead to the same crisp partition and then in the same Rand index value. For this reason the Rand index is not appropriate for fuzzy clustering assessment.

To overcome this problem Hullermeier et al. (2012) proposed a generalization of the Rand index for fuzzy partitions. We recall some essential background.

Let $\mathcal{P} = \{\mathcal{P}_1, \ldots, \mathcal{P}_K\}$ be a fuzzy partition of the data set $\mathcal{D}$, each element $y_i \in \mathcal{D}$ is characterized by its membership vector:

$$\mathcal{P}_i = (\mathcal{P}_1(y_i), \mathcal{P}_2(y_i), \ldots, \mathcal{P}_k(y_i), \ldots, \mathcal{P}_K(y_i)) \quad \in [0,1]^K \tag{5.13}$$

where $\mathcal{P}_k(y_i)$ is the degree membership of the i^{th} series to the k^{th} cluster $\mathcal{P}_k$. Given any pair $(y_i, y_i^{'}) \in \mathcal{D}$, Hullermeier et al. (2012) defined a fuzzy equivalence relation on $\mathcal{D}$ in terms of similarity measure on the associated membership vectors (5.13). Generally, this relation is of the form:

$$E_{\mathcal{P}} = 1 - \|\mathcal{P}_i - \mathcal{P}_{i'}\|$$

where $\|\cdot\|$ represents the L_1norm divided by 2 that constitutes a proper metric on $[0,1]^K$ and yields value on $[0,1]$. $E_{\mathcal{P}}$ is equal to 1 if and only if y_i and $y_i^{'}$ have the same membership pattern and is equal to 0 otherwise. The basic idea of the authors to reach the fuzzy extension of the Rand index was to generalize the concept of concordance in the following way.
Given 2 fuzzy partition, $\mathcal{P}$ and $\mathcal{Q}$ and considering a pair $(y_i, y_i^{'})$ as being concordant as $\mathcal{P}$ and $\mathcal{Q}$ agree on its degree of equivalence, they defined the degree of concordance as

$$conc(y_i, y_i^{'}) = 1 - \parallel E_{\mathcal{P}}(y_i, y_i^{'}) - E_{\mathcal{Q}}(y_i, y_i{'}) \parallel \quad \in [0,1],$$

and degree of discordance as:

$$disc(y_i, y_i^{'}) = \parallel E_{\mathcal{P}}(y_i, y_i^{'}) - E_{\mathcal{Q}}(y_i, y_i^{'}) \parallel \quad \in [0,1].$$

Finally, the distance measure proposed by Hullermeier et al. (2012) is defined as the normalized sum of degrees of discordance:

$$d(\mathcal{P}, \mathcal{Q}) = \frac{\sum_{(y_i, y_i^{'}) \in \mathcal{D}} \|E_{\mathcal{P}}(y_i, y_i^{'}) - E_{\mathcal{Q}}(y_i, y_i^{'})\|}{N(N-1)/2}$$

The direct generalization of the Rand index corresponds to the normalized degree of concordance and it is equal to:

$$R_E(\mathcal{P}, \mathcal{Q}) = 1 - d(\mathcal{P}, \mathcal{Q})$$

and it reduces to the original Rand index when partitions $\mathcal{P}$ and $\mathcal{Q}$ are non-fuzzy.

As true fuzzy partition, we always computed the true cluster centers with an optimal P-spline smoother, and then we computed the true probabilities by applying equation (5.9).

5.3.1 Simulated data

As a first experiment, we generated $K = 6$ clusters of numerical series at $n = 10$ equally spaced time points in $[0, 1]$ as described in Coffey et al. (2014). Distinct cluster specific models were used (subscript i refers to the series, subscript$j$ refers to the time domain):

$$
\begin{aligned}
y_{ij}^{(1)} &= \alpha_i + \sin(\beta_i * \pi * x_{ij}) + \gamma_i + \varepsilon_{ij} \\
y_{ij}^{(2)} &= x_{ij} + (\delta_i)^{-3} + \iota_i + \gamma_i + \varepsilon_{ij} \\
y_{ij}^{(3)} &= \nu_i + \gamma_i + \varepsilon_{ij} \\
y_{ij}^{(4)} &= \zeta_i + \cos(\zeta_i * \pi * x_{ij}) + \gamma_i + \varepsilon_{ij} \\
y_{ij}^{(5)} &= \xi_i - \eta_i * \exp(-\theta_i * x_i) + \gamma_i + \varepsilon_{ij} \\
y_{ij}^{(6)} &= -3(x_{ij} - 0.5) + \gamma_i + \varepsilon_{ij}
\end{aligned}
$$

where:
$\alpha_i \sim N(\sqrt{2}; \sigma_e^2)$ with $\sigma_e^2 = 0.08$, $\beta_i \sim N(4 * \pi; \sigma_e^2)$, $\delta_i \sim N(0.75; \sigma_e^2)$, $\iota_i \sim N(1; \sigma_e^2)$, $\nu_i \sim N(0; \sigma_e^2)$, $\zeta_i \sim N(2; \sigma_e^2)$, $\xi_i \sim N(2; \sigma_v^2)$ with $\sigma_v^2 = 0.85$, $\eta_i \sim N(4; \sigma_v^2)$, $\theta_i \sim N(6; \sigma_e^2)$, $\gamma_i \sim N(0; \sigma_u^2)$ with σ_u^2 ranging from 0.3 to 1 and ε_{ij} is an autoregressive model of order 1.
Cluster means were chosen to reflect the situation where there are series that show little variation in value over time (as given by cluster 3) and series which have distinct signal over time. Cluster sizes were equal to 90, 50, 100, 25, 60 and 35, for cluster $1, 2, 3, 4, 5, 6$ respectively, giving a total number of 360 simulated series. Data set is plotted in Fig.5.1.

Given the nature of the simulated series, we are interested in the similarity of the shape of the series. For this reason the chosen metric was the Penrose shape distance (Penrose, 1952), defined as:

$$
d_{i,j} = \sqrt{\frac{n_i}{n_i - 1}(d_{i,j}^2 - q_{ij}^2)}, \tag{5.14}
$$

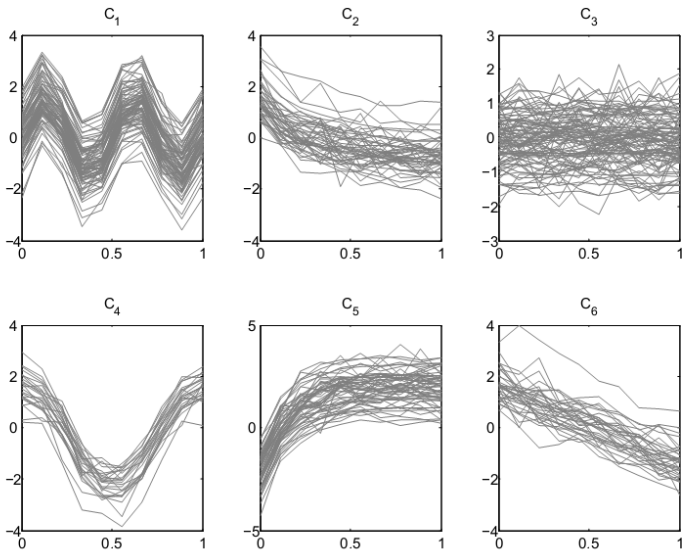

Figure 5.1: Data set generated for simulation study.

where $d^2_{i,j}$ is the squared average Euclidean distance coefficient and $q^2_{ij} = \frac{1}{n_i^2}\left(\sum_{j=i}^{n_i} y_{ji} - \sum_{j=1}^{n_i} c_{jk}\right)^2$.

We performed five analyses with $100, 500, 1000, 5000$ and 10000 boosting iterations. In all cases we set 10 random starting points. Figure 5.2 shows the behavior of the BC function as defined in equation (5.10) during the boosting iterations. In this case the BC values appear to be non-increasing as the number of iterations increases. The values of the BC function are equal to $0.3615, 0.2783, 0.2643, 0.2584, 0.2583$ for $100, 500, 1000, 5000$ and 10000 boosting iterations respectively.

All the solutions return in fact the same results in terms of estimated centers: in example, figure 5.3 shows the estimated cluster centers for each cluster as returned by the first analysis.

For this data set, by using the Penrose shape distance, the Fuzzy Rand Index is equal to $0.8599, 0.8954, 0.9059, 0.9178$ and 0.9194 for the solutions with respectively $100, 500, 1000, 5000$ and 10000 boosting iterations. Even if the solutions in terms of "hard" clustering are the same, the difference in terms of Fuzzy Rand Index indicates that the partitions returned by the algorithm are

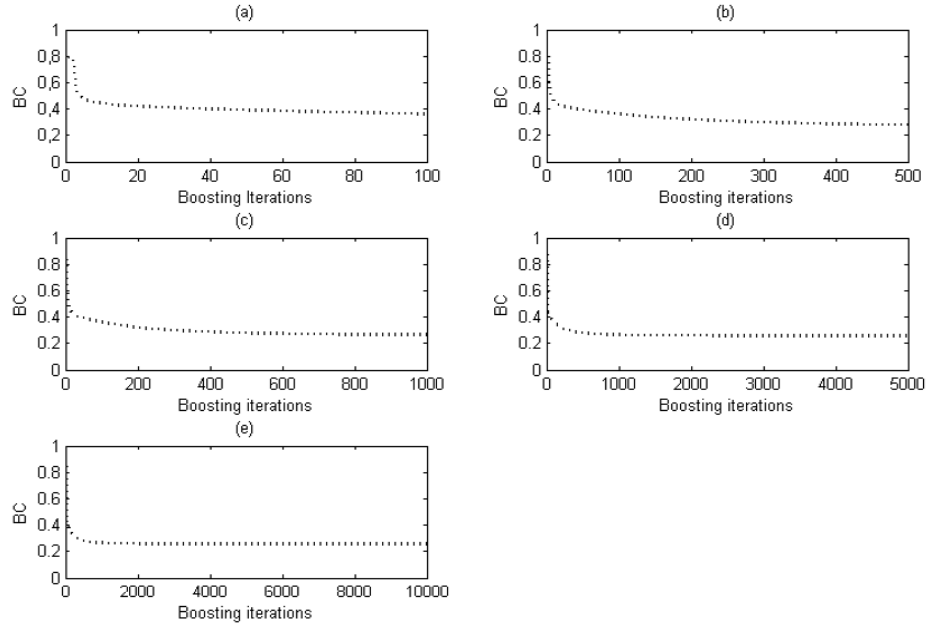

Figure 5.2: BC function progress through 100 boosting iterations: (a) = 100 boosting iterations; (b) = 500 boosting iterations; (c) = 1000 boosting iterations; (d) = 5000 boosting iterations; (e) = 10000 boosting iterations

really close to the true one. The true value of the BC index is 0.1977.

5.3.2 *Synthetic data set*

Synthetic.tseries data set is freely available from the `TSclust` R-package (Montero and Vilar, 2014). Synthetic.tseries data consist of three partial realizations of length $n = 200$ of six first order autoregressive models. Figure 5.4 shows separately the six groups of series. Subplot (a) shows an AR(1) process with moderate autocorrelation. Subplot (b) contains series from a bilinear process with approximately quadratic conditional mean. Subplot (c) is formed by an exponential autoregressive model with a more complex non-linear structure. Subplot (d) shows a self-exciting threshold autoregressive model with a relatively strong non-linearity. Subplot (e) contains series generated by a general non-linear autoregressive model and subplot (f) shows a smooth transition autoregressive model presenting a weak non-linear structure. As we did not generate these series we do not show completely the

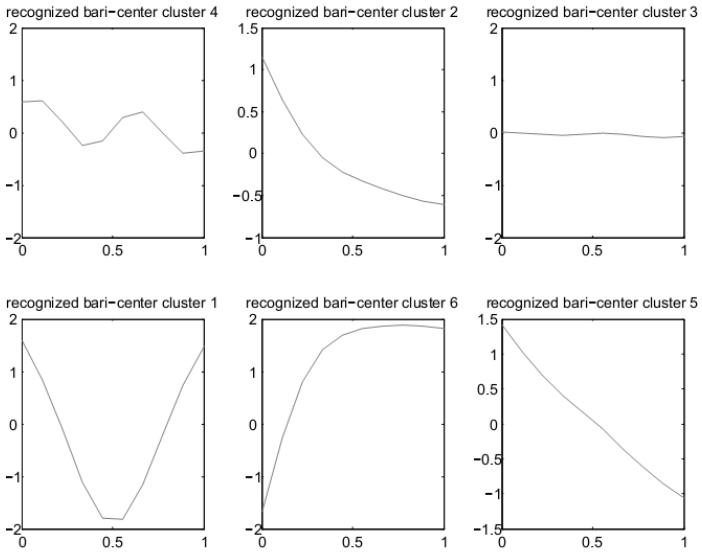

Figure 5.3: Estimated cluster centers

simulation setting. For more details about the generating models we refer to Montero and Vilar, 2014, p. 24.

Assuming that the aim of cluster analysis is to discover the similarity between underlying models, the "true" cluster solution is given by the six clusters involving the three series from the same generating model. Given the nature of the data set considered, we use a periodogram-based distance measure proposed by Caiado et al. (2006). It assesses the dissimilarity between the corresponding spectral representation of time series.

By following also the suggestion of to Montero and Vilar (2014), an interesting alternative to measure the dissimilarity between time series is the frequency domain approach. Power spectrum analysis is concerned with the distribution of the signal power in the frequency domain. The power-spectral density is defined as the Fourier transform of the autocorrelation function of i^{th} series. It is a measure of self-similarity of a signal with its delayed version. The classic method for estimation of the power spectral density of an n-sample record is the periodogram introduced by Schuster (1897). Let y and $y^{'}$ be two time series of length n.

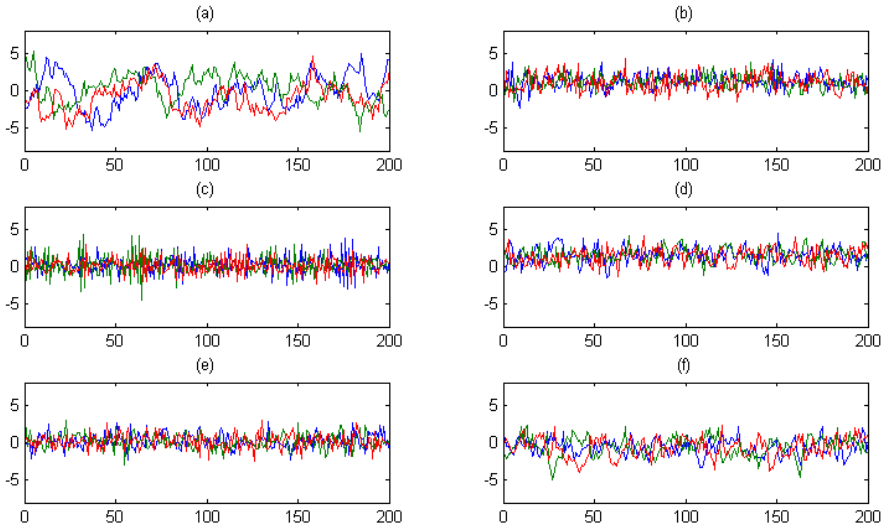

Figure 5.4: Synthetic.tseries data set

Let $f_j = 2\pi j/n$, $j = 1, \ldots, n/2$ in the range 0 to π, be the frequencies of the series.

Let $PSD_y(f_j) = \frac{1}{n}\sum_{t=1}^{n} |y_t(f_j)\exp(-\iota t f_j)|^2$.

Let $PSD_{y'}(f_j) = \frac{1}{n}\sum_{t=1}^{n} |y_t'(f_j)\exp(-\iota t f_j)|^2$ be the periodograms of series y and $y^{'}$, respectively.

Finally, the dissimilarity measure between y and $y^{'}$ proposed by Caiado et al. (2006) is defined as the Euclidean distance between periodogram ordinates :

$$d_{y,y'} = \sqrt{\sum_{j=1}^{(n/2)} [PSD_y(f_j) - PSD_{y'}(f_j)]^2}. \tag{5.15}$$

We performed our analysis by setting 800 boosting iterations and 10 random starting points.

Table 5.1 shows the results of applying our algorithm to the Synthetic.tseries data set. Each series is assigned to the estimated cluster according to the value of the membership probability matrix. In order to obtain the Fuzzy Rand Index, we computed the true cluster centers with a periodogram modeled by P-spline , and then we computed the true probabilities by applying

		Estimated Clusters					
		C1	C2	C3	C4	C5	C6
True Clusters	a	0	0	0	0	0	3
	b	0	1	0	2	0	0
	c	3	0	0	0	0	0
	d	0	3	0	0	0	0
	e	0	0	3	0	0	0
	f	0	0	0	0	3	0

Table 5.1: Confusion matrix from clustering on Synthetic.tseries data set

equation (5.9) by using the periodogram-based distance as in equation (5.15). The Fuzzy Rand Index is equal to 0.9698. Even if the solutions in terms of "hard" clustering seems to be excellent (since only series is misclassified), the difference in terms of Fuzzy Rand index indicates that the partitions returned by the algorithm are really close to the true one.

5.4 *A real data example*

The "'growth"' data set is freely available from the internal repository of the R-package `fda` (Ramsay et al., 2012). This data set comes from the Berkeley Growth Study (Tuddenham and Snyder, 1954). Left hand side of figure 5.5 shows the growth curves of 93 children, 39 boys and 54 girls, starting by the age of one year till the age of 18. The right hand side of the same figure displays the corresponding growth velocities.

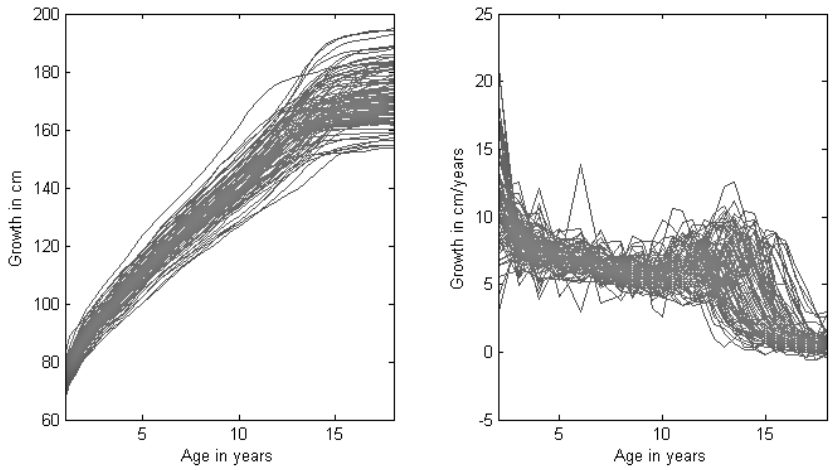

Figure 5.5: Growth curves (left hand side) and growth velocity curves (right hand side) of 93 children from Berkeley Growth Study data

In the framework of cluster analysis this data set was mainly used for problems of clustering of misaligned data (Sangalli et al., 2010a,b). We performed two analyses with 800 boosting iterations and with 10 random starting point with $K = 2$. In the first partitioning analysis we used the Euclidean distance. The estimated centers of both the growth curves and the growth velocity curves are displayed respectively in the left and right hand side of figure 5.6. As it can be noted, Euclidean distance discriminates between children growing more and children growing less. This can be appreciated by looking at left hand side of the same figure. On average, as expected, boys grow more than girls.

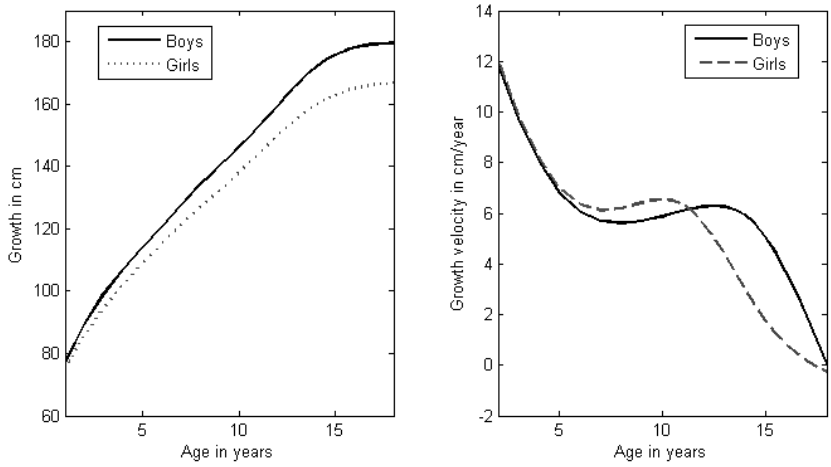

Figure 5.6: Estimated centers of growth curves (left hand side) and growth velocities (right hand side): Euclidean distance

Nevertheless, Euclidean distance does not seem the right measure to be used in such a case. Probably researchers are interested in the shape of both growth and growth velocity curves during the years. For this reason, we repeated the analysis by using the Penrose shape distance as defined in equation (5.14). Figure 5.7 shows the estimated centers for both the growth and the growth velocity curves. The recognized centers are really similar to the ones obtained by Sangalli *et al.* (2010a; 2010b): firstly, as confirmed by looking at tables 5.4 and 5.5 with respect to tables 5.2 and 5.3, there is a neat separation of boys and girls. Secondly, by looking at right hand side of figure 5.7, boys start to grow later but they seem to have a more pronounced growth, as

it can be noticed by looking at the higher peak in correspondence of 15 year.

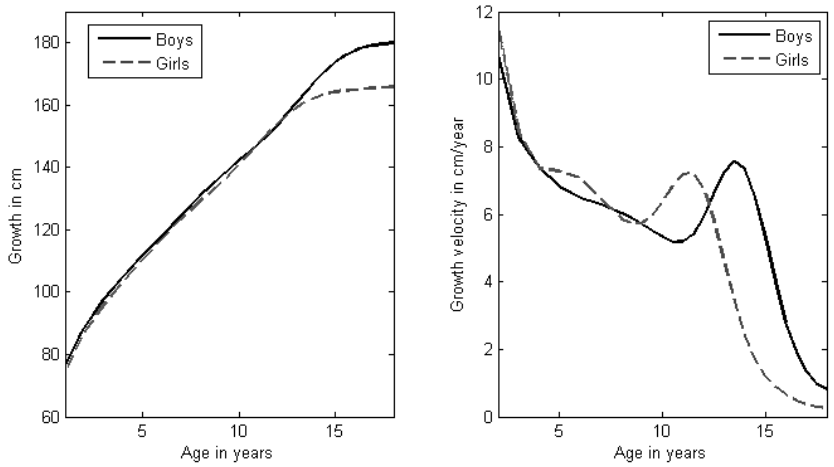

Figure 5.7: Estimated centers of growth curves (left hand side) and growth velocities (right hand side): Penrose shape distance.

	Cluster 1	Cluster 2
Boys	23	16
Girls	16	38

Table 5.2: Confusion matrix of growth curves with the Euclidean distance. Series have been assigned to the clusters according the values of membership probabilities computed as in equation (5.9).

	Cluster 1	Cluster 2
Boys	31	8
Girls	9	45

Table 5.3: Confusion matrix of growth velocity curves with the Euclidean distance. Series have been assigned to the clusters according the values of membership probabilities computed as in equation (5.9).

The Fuzzy Rand Index is equal to 0.8884 and 0.8240 by using the Euclidean distance for the partitions of growth and growth velocity curves respectively. The Fuzzy Rand Index is equal to 1.000 and 0.9246 by using the Penrose shape distance for the partitions of growth and growth velocity curves respectively.

	Cluster 1	Cluster 2
Boys	0	39
Girls	52	2

Table 5.4: Confusion matrix of growth curves with the Penrose shape distance. Series have been assigned to the clusters according the values of membership probabilities computed as in equation (5.9).

	Cluster 1	Cluster 2
Boys	36	3
Girls	4	50

Table 5.5: Confusion matrix of growth velocity curves with the Penrose shape distance. Series have been assigned to the clusters according the values of membership probabilities computed as in equation (5.9).

5.5 *Concluding remarks*

In this paper we merged two approaches, theoretically motivated for respectively unsupervised and supervised classification cases, to introduce a new non-hierarchical fuzzy clustering algorithm.

From the Probabilistic Distance (PD) clustering (Ben-Israel and Iyigun, 2008) approach we took the idea of determining the probabilities of each series to any of the k clusters. As this probability is unequivocally related to the distance of each series from the cluster centers, there are no degrees of freedom in determine the membership matrix. From the Boosting approach (Freund and Schapire, 1997) we took the idea of weighting each series according some measure of badness of fit in order to define an unsupervised learning process based on a weighted re-sampling procedure.

In contrast to the boosting approach, the higher is the probability of a given instance to be member of a given cluster, the higher is the weight of that instance in the re-sampling process. As a learner we used a P-spline smoother (Eilers and Marx, 1996). In this way we defined a suitable loss function and, at the same time, we proposed a fuzzy clustering procedure that does not depend on the definition of a fuzzifier parameter. To evaluate the performance of our proposal, we conducted three experiments, one of them on simulated data and the remaining two on data sets known in literature. The results show that our Boosted-oriented procedure help users to obtain good results in terms of data partitioning. Even if the final fuzzy partition is sensitive

to the choice of a distance measure, it is independent on any other input parameters. This consideration allows to define a suitable true fuzzy partition with which evaluate the final solution in terms of Fuzzy Rand Index (Hullermeier et al., 2012). The weighted re sampling process allows each series to contribute to the composition of each cluster as well as the adaptive estimation of cluster centers allow the algorithm to learn by its progresses.

6 CONCLUSIONS AND PERSPECTIVE

The framework of this thesis is supervised and unsupervised classification. The general framework has been presented in chapter 2. Three methodological contributions have been provided and widely discussed in chapters 3, 4 and 5. In the following we give a chapter-wise summary of the main results.

Chapter 3 introduced *Visual Pruning for Decision Trees*. Within the framework of Classification And Regression Trees (CART) methodology (Breiman et al., 1984), we proposed a strategy to select decision trees, which is alternative to the classical cost-complexity pruning. The contribution is on both the visual representation of the data partition in a geometrical space and the selection of the decision tree. Main idea is to define a new way to represent the tree structure by a node-link diagram. The tree structure appears with edges of different length depending on the impurity decrease when passing from one node to another. The lower is the impurity in the descendant node, the longest is the length of path. Indeed, the edges connecting the parent node and its child node can be visualized with a length proportional to the decrease of impurity. In this way we have two advantages from the proposed graphical representation. The first one is to add meaningful statistical measures to the interpretation of the exploratory tree. The second one is to provide a visual selection criterion to choose the best decision tree for predicting new cases. As a consequence, it is possible to define for any branch of the oriented tree the path decrease in impurity from the starting node to its descendants. In this way we show the best splits and the purest nodes of the tree. The proposed method points out the relative importance of each split, identifying which of them are the most discriminant and the best cut level to obtain an optimal decision tree. The advantage of the proposed visual approach is to identify the weakest link for pruning as well as the best tree by a graphical analysis. With respect to CART methodology there is not need to

identify the sequence of nested trees by a pruning procedure on which basis selecting the best decision tree. Since the badness of fit of a tree is measured by misclassification rate, the pruning is immediately apparent in accordance with it. The main interpretative advantages of the proposed visual tree are shown in an application on a real data set. The results in terms of error rate were really similar to the ones returned by the CART procedure, showing how this new way to select the best tree is a valid alternative to the well known cost-complexity pruning.

Chapter 4 introduced *P-spline based Clustering of Correlated series*. Within the framework of cluster analysis of time series we proposed a new way to partition the data based on Penalized spline (P-spline) proposed by Eilers and Marx (1996). Time series arise in many scientific areas. Several clustering algorithms have been proposed. Most of the time, these procedures do not facilitate the removal of noise from data, have difficulties handling time series with unequal length and require a preprocessing step of the considered data, i.e. by modeling each series with an appropriate model for time series. We tried to overcome these hitches. In this work we proposed a new approach exploiting a P-spline framework. We model each series by penalized spline (P-spline) smoothers (Eilers and Marx, 1996) and we perform a cluster analysis on the estimated coefficients. P-spline smoothers separate the signal of series from the noise, capturing the different shapes. P-spline coefficients are close to the fitted curve and present the skeleton of the fit (Eilers and Marx, 2010). Thus, summarizing each series by spline coefficients reduces the dimensionality of the problem, saving computational time without reduction in performance of clustering procedure. To select the smoothing parameter in a P-spline framework, we adopted the V-curve criterion proposed by Frasso and Eilers (2015). This choice was due to its advantage to be insensitive to serial correlation in the noise around the trend of series. P-Spline smoothing requires a relatively large number of basis function, but still less than the number of observations. The basic idea was to perform the classification task on the reduced space spanned by optimal spline coefficients. Thus to partition the data (time) series into k cluster, we need to partition their estimated spline coefficients. Any clustering algorithms and distance measure can be used to cluster the coefficients of P-spline. Series with different lengths (e.g. with missing values) can be handled by P-spline due to extrapolation prop-

erties (Eilers and Marx, 2010). When series are observed in different time domain as they are of different length, we perform cluster analysis on fitted values of a reduced time domain common to all series. To evaluate the performance of our proposal we conducted two analysis considering series with equal and unequal length, respectively. We simulated series as in Coffey et al. (2014). We compared our result with their proposal. We showed that the proposed approach is significantly faster than other proposal while still retaining good performance in terms of ARI (Hubert and Arabie, 1985) values. We also applied our proposal on financial time series showing as it can help financial practitioners to support their investment decision.

Chapter 5 introduced *Boosted-Oriented Probabilistic Clustering of Series*. Within the framework of fuzzy (probabilistic) clustering, we proposed a fuzzy clustering procedure for data (time) series that is independent from the choice of a fuzzifier parameter. Main idea is to adapt the boosting approach to unsupervised learning problems, specially to non hierarchical cluster analysis. The aim is to assign each series of a data set to a cluster. Our proposal comes from two approaches, theoretically motivated for unsupervised and supervised classification cases, respectively. From the Probabilistic Distance (PD) clustering approach (Ben-Israel and Iyigun, 2008) we took the idea of determining the probabilities of each series to belong to any clusters. As this probability is unequivocally related to the distance of each series from the cluster centers, there are no degrees of freedom in determine the membership matrix. From the Boosting approach (Freund and Schapire, 1997) we took the idea of weighting each series according some measure of badness of fit in order to define an unsupervised learning process based on a weighted re-sampling procedure. In contrast to the boosting approach, the higher is the probability of a given series to be member of a given cluster, the higher is the weight of that series in the re-sampling process. In this way we also defined a suitable loss function. As a learner we used a P-spline smoother (Eilers and Marx, 1996). The weighted re-sampling process allows each series to contribute to the composition of each cluster as well as the adaptive estimation of cluster centers allow the algorithm to learn by its progresses. The global performance of the proposed method was investigated by three experiments (one of them on simulated data and the remaining two on data sets known in literature) evaluated by using a fuzzy variant of the Rand In-

dex (Hullermeier et al., 2012). The results showed that our Boosted-oriented procedure can help users to obtain good results in terms of data partitioning. The final fuzzy partition is independent on any other input parameters. However, it is worth nothing that the choice of the distance measure influences the goodness of partition.

BIBLIOGRAPHY

Alonso, A. M., Berrendero, J. R., Hernández, A., and Justel, A. (2006). Time series clustering based on forecast densities. *Computational Statistics & Data Analysis*, 51(2):762–776.

Ankerst, M., Breunig, M. M., Kriegel, H.-P., and Sander, J. (1999). Optics: ordering points to identify the clustering structure. In *ACM Sigmod Record*, volume 28, pages 49–60. ACM.

Ankerst, M., Ester, M., and Kriegel, H.-P. (2000). Towards an effective cooperation of the user and the computer for classification. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 179–188. ACM.

Ankerst, M., Keim, D. A., and Kriegel, H.-P. (1996). Circle segments: A technique for visually exploring large multidimensional data sets.

Apté, C. and Weiss, S. (1997). Data mining with decision trees and decision rules. *Future generation computer systems*, 13(2):197–210.

Aria, M. and Siciliano, R. (2003). Learning from trees: Two-stage enhancements. *Proceedings of Classification and Data Analysis Group (CLADAG 2003)*, pages 22–24.

Baragona, R. (2001). A simulation study on clustering time series with metaheuristic methods. *Quaderni di Statistica*, 3:1–26.

Barlow, T. and Neville, P. (2001). A comparison of 2-d visualizations of hierarchies. In *Information Visualization, IEEE Symposium on*, pages 131–131. IEEE Computer Society.

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

Benartzi, S. and Thaler, R. H. (2001). Naive diversification strategies in defined contribution saving plans. *American economic review*, pages 79–98.

Berndt, D. J. and Clifford, J. (1994). Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA.

Bezdek, J. C. (1981). *Pattern Recognition with Fuzzy Objective Function Algorithms*. Kluwer Academic Publishers, Norwell, MA, USA.

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. (1984). *Classification and regression trees*. CRC press.

Caiado, J., Crato, N., and Peña, D. (2006). A periodogram-based metric for time series classification. *Computational Statistics & Data Analysis*, 50(10):2668–2684.

Campello, R. J. (2007). A fuzzy extension of the rand index and other related indexes for clustering and classification assessment. *Pattern Recognition Letters*, 28(7):833–841.

Cappelli, C., Mola, F., and Siciliano, R. (1998). An alternative pruning method based on the impurity-complexity measure. In *COMPSTAT*, pages 221–226. Springer.

Carmack, P. S., Spence, J. S., and Schucany, W. R. (2012). Generalised correlated cross-validation. *Journal of Nonparametric Statistics*, 24(2):269–282.

Chouakria, A. D. and Nagabhushan, P. N. (2007). Adaptive dissimilarity index for measuring time series proximity. *Advances in Data Analysis and Classification*, 1(1):5–21.

Coffey, N., Hinde, J., and Holian, E. (2014). Clustering longitudinal profiles using p-splines and mixed effects models applied to time-course gene expression data. *Computational Statistics & Data Analysis*, 71:14–29.

de Boor, C. (1978). *A Practical Guide to Splines*. Springer-Verlag.

de Boor, C. and Swartz, B. (1973). Collocation at Gaussian Points. *SIAM Journal on Numerical Analysis*, 10(4).

Dembélé, D. and Kastner, P. (2003). Fuzzy c-means method for clustering microarray data. *Bioinformatics*, 19(8):973–980.

Dierckx, P. (1995). *Curve and Surface Fitting with Splines*. Oxford University Press.

Dietterich, T. G. (2000). Ensemble methods in machine learning. In *Multiple classifier systems*, pages 1–15. Springer.

Dunn, J. C. (1973). A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters.

Eibl, G. and Pfeiffer, K. P. (2002). How to make adaboost. m1 work for weak base classifiers by changing only one line of the code. In *Machine Learning: ECML 2002*, pages 72–83. Springer.

Eilers, P. H. (2003). A perfect smoother. *Analytical chemistry*, 75(14):3631–3636.

Eilers, P. H. C. and Marx, B. D. (1996). Flexible smoothing with B-splines and penalties. *Statistical Science*, 11(2):pp. 115–121.

Eilers, P. H. C. and Marx, B. D. (2010). Splines, knots, and penalties. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(6):637–653.

Eiter, T. and Mannila, H. (1994). Computing discrete fréchet distance.

Esposito, F., Malerba, D., Semeraro, G., and Kay, J. (1997). A comparative analysis of methods for pruning decision trees. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(5):476–491.

Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231.

Fayyad, U. M., Wierse, A., and Grinstein, G. G. (2002). *Information visualization in data mining and knowledge discovery*. Morgan Kaufmann.

Frasso, G. and Eilers, P. H. (2015). L- and v-curves for optimal smoothing. *Statistical Modelling*, 15(1):91–111.

Fréchet, M. M. (1906). Sur quelques points du calcul fonctionnel. *Rendiconti del Circolo Matematico di Palermo (1884-1940)*, 22(1):1–72.

Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139.

Fu, T.-c., Chung, F.-l., Ng, V., and Luk, R. (2001). Pattern discovery from stock time series using self-organizing maps. In *Workshop Notes of KDD2001 Workshop on Temporal Data Mining*, pages 26–29. Citeseer.

Futschik, M. E. and Carlisle, B. (2005). Noise-robust soft clustering of gene expression time-course data. *Journal of bioinformatics and computational biology*, 3(04):965–988.

Galeano, P. and Peña, D. P. (2000). Multivariate analysis in vector time series. *Resenhas do Instituto de Matemática e Estatística da Universidade de São Paulo*, 4(4):383–403.

Gey, S. and Poggi, J.-M. (2006). Boosting and instability for regression trees. *Computational statistics & data analysis*, 50(2):533–550.

Golay, X., Kollias, S., Stoll, G., Meier, D., Valavanis, A., and Boesiger, P. (1998). A new correlation-based fuzzy logic clustering algorithm for fmri. *Magnetic Resonance in Medicine*, 40(2):249–260.

Han, J. and Kamber, M. (2001). Data mining: concepts and technologies.

Hansen, P. C. (1992). Analysis of Discrete Ill-Posed Problems by Means of the L-curve. *SIAM Review*, 34(4):pp. 561–580.

Hansen, P. C. (2000). The L-curve and its use in the numerical treatment of inverse problems. In *Computational Inverse Problems in Electrocardiology, ed. P. Johnston, Advances in Computational Bioengineering*, pages 119–142. WIT Press.

Hansen, P. C. and O'Leary, D. P. (1993). The use of the L-Curve in the regularization of discrete ill-posed problems. *SIAM J. SCI. COMPUT.*, 14(6):pp. 1487–1503.

Hartigan, J. A. and Wong, M. A. (1979). Algorithm as 136: A k-means clustering algorithm. *Applied statistics*, pages 100–108.

Hastie, T., Tibshirani, R., Friedman, J., Hastie, T., Friedman, J., and Tibshirani, R. (2009). *The elements of statistical learning*, volume 2. Springer.

Hastie, T. J. and Tibshirani, R. J. (1990). *Generalized additive models*. London: Chapman & Hall.

Heiser, W. J. (2004). Geometric representation of association between categories. *Psychometrika*, 69(4):513–545.

Hodrick, R. J. and Prescott, E. C. (1997). Postwar U.S. Business Cycles: An Empirical Investigation. *Journal of Money, Credit and Banking*, 29(1):pp. 1–16.

Hubert, L. and Arabie, P. (1985). Comparing partitions. *Journal of classification*, 2(1):193–218.

Hullermeier, E., Rifqi, M., Henzgen, S., and Senge, R. (2012). Comparing fuzzy partitions: A generalization of the rand index and related measures. *Fuzzy Systems, IEEE Transactions on*, 20(3):546–556.

Johnson, S. C. (1967). Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254.

Kass, G. V. (1980). An exploratory technique for investigating large quantities of categorical data. *Applied statistics*, pages 119–127.

Kauermann, G., Krivobokova, T., and Semmler, W. (2011). Filtering time series with penalized splines. *Studies in Nonlinear Dynamics & Econometrics*, 15(2):2.

Kaufman, L. and Rousseeuw, P. J. (2009). *Finding groups in data: an introduction to cluster analysis*, volume 344. John Wiley & Sons.

Keogh, E. J. and Pazzani, M. J. (2001). Derivative dynamic time warping. In *SDM*, volume 1, pages 5–7. SIAM.

Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480.

Košmelj, K. and Batagelj, V. (1990). Cross-sectional approach for clustering time varying data. *Journal of Classification*, 7(1):99–109.

Kuhn, H. W. (1973). A note on fermat's problem. *Mathematical programming*, 4(1):98–107.

Kumar, M., Patel, N. R., and Woo, J. (2002). Clustering seasonality patterns in the presence of errors. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 557–563. ACM.

Li, M., Chen, X., Li, X., Ma, B., and Vitányi, P. M. (2004). The similarity metric. *Information Theory, IEEE Transactions on*, 50(12):3250–3264.

Li, M. and Vitányi, P. M. (2009). *An introduction to Kolmogorov complexity and its applications*. Springer Science & Business Media.

Liao, T. W. (2005). Clustering of time series data a survey. *Pattern recognition*, 38(11):1857–1874.

Liu, Y. and Salvendy, G. (2007). Design and evaluation of visualization support to facilitate decision trees classification. *International journal of human-computer studies*, 65(2):95–110.

Ma, P., Castillo-Davis, C. I., Zhong, W., and Liu, J. S. (2006). A data-driven clustering method for time course gene expression data. *Nucleic Acids Research*, 34(4):1261–1269.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations.

Maharaj, E. A. (1996). A significance test for classifying arma models. *Journal of Statistical Computation and Simulation*, 54(4):305–331.

Maharaj, E. A. (2000). Cluster of time series. *Journal of Classification*, 17(2):297–314.

Maillard, S., Roncalli, T., and Teïletche, J. (2008). On the properties of equally-weighted risk contributions portfolios. *Available at SSRN 1271972*.

Markowitz, H. (1952). Portfolio selection*. *The journal of finance*, 7(1):77–91.

Markowitz, H. (1956). The optimization of a quadratic function subject to linear constraints. *Naval research logistics Quarterly*, 3(1-2):111–133.

Merton, R. C. (1980). On estimating the expected return on the market: An exploratory investigation. *Journal of financial economics*, 8(4):323–361.

Messenger, R. and Mandell, L. (1972). A modal search technique for predictive nominal scale multivariate analysis. *Journal of the American statistical association*, 67(340):768–772.

Mola, F. and Siciliano, R. (1992). A two-stage predictive splitting algorithm in binary segmentation. In *Computational statistics*, pages 179–184. Springer.

Mola, F. and Siciliano, R. (1994). Alternative strategies and catanova testing in two-stage binary segmentation. In *New Approaches in Classification and Data Analysis*, pages 316–323. Springer.

Mola, F. and Siciliano, R. (1997). A fast splitting procedure for classification trees. *Statistics and Computing*, 7(3):209–216.

Mola, F. and Siciliano, R. (1998). A general splitting criterion for classification trees. *Metron*, 56:3–4.

Möller-Levet, C. S., Klawonn, F., Cho, K.-H., and Wolkenhauer, O. (2003). Fuzzy clustering of short time-series and unevenly distributed sampling points. In *Advances in Intelligent Data Analysis V*, pages 330–340. Springer.

Montero, P. and Vilar, J. A. (2014). TSclust: An R package for time series clustering. *Journal of Statistical Software*, 62(1):1–43.

Morgan, J. N. and Messenger, R. C. (1973). Thaid: A sequential analysis program for the analysis of nominal scale dependent variables.

Morgan, J. N. and Sonquist, J. A. (1963). Problems in the analysis of survey data, and a proposal. *Journal of the American statistical association*, 58(302):415–434.

O'Sullivan, F. (1986). A statistical perspective on ill-posed inverse problems. *Statistical science*, pages 502–518.

Pal, N. R. and Bezdek, J. C. (1995). On cluster validity for the fuzzy c-means model. *Fuzzy Systems, IEEE Transactions on*, 3(3):370–379.

Penrose, L. S. (1952). Distance, size and shape. *Annals of Eugenics*, 17(1):337–343.

Piccolo, D. (1990). A distance measure for classifying arima models. *Journal of Time Series Analysis*, 11(2):153–164.

Quinlan, J. R. (1987). Simplifying decision trees. *International journal of man-machine studies*, 27(3):221–234.

Quinlan, J. R. (1993). *C4. 5: Programs for Machine Learning*. Morgan Kaufmann.

Quinlan, J. R. et al. (1979). *Discovering rules by induction from large collections of examples*. Expert systems in the micro electronic age. Edinburgh University Press.

Ramoni, M., Sebastiani, P., and Cohen, P. (2002). Bayesian clustering by dynamics. *Machine learning*, 47(1):91–121.

Ramsay, J., Wickham, H., Graves, S., and Hooker, G. (2012). Functional data analysis,.

Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850.

Ruppert, D. (2002). Selecting the number of knots for penalized splines. *Journal of computational and graphical statistics*, 11(4).

Ruppert, D., W. M. C. R. (2003). *Semiparametric Regression*. Cambridge Series in Statistical and Probabilistic Mathematics (No. 12).

Ruspini, E. H. (1970). Numerical methods for fuzzy clustering. *Information Sciences*, 2(3):319–350.

Sangalli, L. M., Secchi, P., Vantini, S., and Vitelli, V. (2010a). Functional clustering and alignment methods with applications. *Communications in Applied and Industrial Mathematics*, 1(1):205–224.

Sangalli, L. M., Secchi, P., Vantini, S., and Vitelli, V. (2010b). K-mean alignment for curve clustering. *Computational Statistics & Data Analysis*, 54(5):1219–1233.

Sankoff, D. and Kruskal, J. B. (1983). Time warps, string edits, and macromolecules: the theory and practice of sequence comparison. *Reading: Addison-Wesley Publication, 1983, edited by Sankoff, David; Kruskal, Joseph B.*, 1.

Schumaker, L. L. (1981). *Spline Functions: Basic Theory*. John Wiley and Sons, New York.

Schuster, A. (1897). On lunar and solar periodicities of earthquakes. *Proceedings of the Royal Society of London*, 61(369-377):455–465.

Schwämmle, V. and Jensen, O. N. (2010). A simple and fast method to determine the parameters for fuzzy c–means cluster analysis. *Bioinformatics*, 26(22):2841–2848.

Sharpe, W. F. (1966). Mutual fund performance. *Journal of business*, pages 119–138.

Shneiderman, B. (1992). Tree visualization with tree-maps: 2-d space-filling approach. *ACM Transactions on graphics (TOG)*, 11(1):92–99.

Siciliano, R., Aria, M., and DAmbrosio, A. (2008). Posterior prediction modelling of optimal trees. In *COMPSTAT 2008*, pages 323–334. Springer.

Tuddenham, R. D. and Snyder, M. M. (1954). Physical growth of california boys and girls from birth to eighteen years. *Publications in child development. University of California, Berkeley*, 1(2):183.

Vaseghi, S. V. (2008). *Advanced digital signal processing and noise reduction*. John Wiley & Sons.

Vilar, J. A., Alonso, A. M., and Vilar, J. M. (2010). Non-linear time series clustering based on non-parametric forecast densities. *Computational Statistics & Data Analysis*, 54(11):2850–2865.

Vogel, C. R. (1996). Non-convergence of the l-curve regularization parameter selection method. *Inverse Problems*, 12(4):535.

Wahba, G. (1990). *Spline models for observational data*, volume 59 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA.

Wang, W., Yang, J., Muntz, R., et al. (1997). Sting: A statistical information grid approach to spatial data mining. In *VLDB*, volume 97, pages 186–195.

Ward Jr, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244.

Weiszfeld, E. (1937). Sur le point pour lequel la somme des distances de n points donnés est minimum. *Tohoku Math. J*, 43(355-386):2.

Whittaker, E. T. (1922). On a new method of graduation. *Proceedings of the Edinburgh Mathematical Society*, 41:63–75.

Wu, K.-L. (2012). Analysis of parameter selections for fuzzy c-means. *Pattern Recognition*, 45(1):407–415.

Yu, J., Cheng, Q., and Huang, H. (2004). Analysis of the weighting exponent in the fcm. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 34(1):634–639.

Zadeh, L. A. (1965). Fuzzy sets. *Information and control*, 8(3):338–353.

ACKNOWLEDGMENTS

Ultimo ”capitolo” di questo lavoro e ultima tappa di questo indimenticabile percorso.
Non é facile riassumere in poche parole le emozioni, i traguardi, le vittorie e le sconfitte che hanno scandito questi 3 anni. E soprattutto non saranno mai tanti i ”GRAZIE” alle persone che hanno reso possibile tutto ció.
Di questi anni non ricordo solo ”la conclusione” ma anche tutto quello che c'é stato in mezzo. Perché tra la partenza e questo traguardo (non solo accademico ma anche personale) ho avuto la paura del ”debutto” e il tremore dell'”esordio”. Ho avuto la possibilitá di imparare (anche se so di non sapere). Ho avuto la possibilitá di essere ancora ”sorpresa” dall'odore (e dalle altre cose) di un *piccolo* (e non solo) libro nuovo. Ho avuto la possibilitá di vivere per un pó anche un'altra Universitá, quella dell'incantevole Leiden. E soprattutto ho avuto la possibilitá di conoscermi meglio.
Ma la mia fortuna piú grande é stata quella di incontrare, e dunque essere circondata da, persone meravigliose. Alcune delle quali, AMICI. Di quelli VERI, peró!
Scusatemi se non saró brava a ringraziarVi cercando al contempo di trasmetterVi quello che giorno dopo giorno ho cercato, sileziosamente, di imparare da ognuno di Voi e di costruire col Vostro aiuto. Tuttavia, ci provo.

Ringrazio la Prof. Roberta Siciliano per i suoi preziosi consigli, per le oppurtunitá che mi ha concesso, per tutto quello che mi ha insegnato nella ricerca (facendomi capire che essa é condivisione) e nel lavoro, che svolge con ammirevole passione e dedizione. La ringrazio inoltre per i suoi incoraggiamenti e, soprattutto, per il suo essere sempre stata anche *mamma* (tra le tante cose non potró mai dimenticare i suoi ”stai serena” dell'ultimo anno).

Ringrazio il Dott. Antonio D'Ambrosio, mio mentore. É stata la prima persona a credere nelle mie capacitá e in veritá non saró mai abbastanza grata con lui per tutto quello che ha fatto durante il mio percorso. Lo ringrazio per

tutto il tempo che mi ha dedicato, per la sua costante e quotidiana presenza (anche quando ero a Leiden) e per tutti i nostri pomeriggi di studio, durante i quali mi ha insegnato che fare ricerca significa confrontarsi (e che *se ascolti dimentichi, se guardi ricordi, ma se fai capisci*).

Ringrazio il Prof. Massimo Aria per tutto il tempo speso per me nelle spiegazioni, con la semplicitá e la chiarezza che contraddistinguono il suo essere insegnante. Lo ringrazio per la sua capacitá di ascolto, per i suoi consigli, per il suo pragmatismo, per avermi tutelato e difeso da studenti "bizzarri", per la comprensione e l'affetto che ha sempre avuto nei miei confronti.

Special thanks to my foreign supervisor prof. Mark de Rooij (Leiden University). These few rows will never be able to convey my thankfulness but I'll try to express what I feel. I'll never forget his inestimable help in every aspect of my studies and for my first symposium colloquium. He did his best in helping me and I tried to do my best to make him proud of me. I had a really great time in Leiden University and I felt part of the department. I had the opportunity to meet a lot of colleagues and each of them gave something special to me. I would like to thank them for their kindness in considering me a member of their group. I hope to meet them all in the future.

Ringrazio il Prof. Natale Lauro, i cui saggi consigli si sono rivelati sempre dei preziosi spunti di riflessione.

Ringrazio il Prof. Stefano Ecchia, che mi ha accolto prima di cominciare questa esperienza e, dunque, senza il quale non sarei qui, ora.

Ringrazio il dott. Gianluca Frasso grazie al quale mi sono avvicinata al meraviglioso mondo delle P-spline. Lo ringrazio per i consigli, per tutta la disponibilitá che ha avuto per me e per i suoi affettuosi incoraggiamenti.

Ringrazio Sonia per il suo essermi stata vicina in quel di Leiden, per avere ascoltato non solo il discorso del mio primo Invitation Colloquium, ma anche le paure che lo accompagnavano e per il suo aiuto durante il mio primo studio di simulazione.

Ringrazio Marco, per il nostro studiare insieme (prima e dopo il concorso) e per la tranquillitá che é riuscito ad infondermi in qualsiasi momento.

Ringrazio Pasquale, mio compagno di avventura. Mai dimenticheró tutte le volte in cui ci siamo arrabbiati per poi sorridere immediatamente dopo. Lo ringrazio per aver condiviso con me parte dello studio, dei corsi seguiti, delle collaborazioni didattiche svolte, ma anche dello stress e delle ansie (fino

all'ultimo). Lo ringrazio, inoltre, per tutta la buona musica che mi ha fatto ascoltare durante lo studio, anche quando migliaia di km ci separavano.

Ringrazio Maddalena e Salvatore per le tante risate e gli indimenticabili momenti di studio, e non, condivisi.

Ringrazio Mena con la quale ho condiviso lunghe chiaccherate "scientifiche" e non solo. La ringrazio, inoltre, per tutte le volte in cui mi ha pensato quando c'era da fare qualcosa.

Ringrazio Roberto, il mio amico italiano a Leiden. L'avventura é cominciata per entrambi nello stesso periodo. Obiettivi diversi, studio diverso. Ma stesse paure. Lo ringrazio per tutte le volte in cui ci siamo spronati l'un l'altro e per tutte le cene condivise quando ad entrambi capitava di sentirci soli.

Ringrazio Debora per avermi ricordato l'importanza delle alghe, Momi per tutto il buonumore che mi metteva nelle giornate olandesi mandandomi bacini e cuoricini, Brunella e Antonio per il supporto, per l'infinita disponibilitá e per la pazienza nell'avermi dovuto sopportare, Silvia e Alberto per la loro amicizia.

Ringrazio Gius e Domenico per la serenitá, la comprensione e l'affetto che mi hanno sempre donato.

Ringrazio Assunta e Angelo, zia Carmela e zio Salvatore, nonna Giovanna, Arianna, Luigi Francesco, Maria Antonietta e Alfonso con Aurora, Giovanna e Costantino, zia Anna e zio Antonio, zia Cira, per la loro costante presenza, per l'aiuto che mi hanno donato, per gli incoraggiamenti e per tutto l'affetto che mi hanno sempre mostrato, condividendo, in modo speciale, con me i momenti importanti che si sono susseguiti durante questi anni.

Ringrazio Alessandro, per il suo essere sempre stato a mia disposizione per le mie stampe (all'improvviso).

Ringrazio Cristina, my little sister, per aver compreso le mie assenze, per avermi rimesso in sesto quando ne necessitavo e per il suo essere sempre presente.

Ringrazio Biagio, mio fratello e mio orgoglio, per aver cercato sempre di farmi sorridere (e di averlo fatto in ogni occasione della nostra vita), per avermi fatto capire che in due non si é mai soli, per tutto il sostegno che mi ha sempre dato, per il coraggio e la forza che mi ha donato ed insegnato ad avere. Da Te ho tanto da imparare!

Ringrazio i miei genitori, che mi hanno sempre spronato ad andare avanti e a non arrendermi. Mai. E nonostante tutto. Il loro supporto e il loro aiuto é stato per fondamentale. Punto di riferimento della mia vita, non me ne potevano capitare di migliori.

Ringrazio Mimmo, mio marito , per avermi capita e ascoltata e aspettata. Sempre. Grazie per tutto l'amore che solo chi ha creduto e crede in un concreto futuro insieme puó riuscire a dimostrare. Senza di Te molte cose non sarebbero state possibili.