TÉCNICO LISBOA

Outlier Detection in Functional Time Series

Catarina Padrela Loureiro

Thesis to obtain the Master of Science Degree in

Mathematics and Applications

Supervisor: Prof. Isabel Maria Alves Rodrigues

Examination Committee

Chairperson: Prof. António Manuel Pacheco Pires
Supervisor: Prof. Isabel Maria Alves Rodrigues
Member of the Committee: Prof. Maria da Conceição Esperança Amado

December 2019

To my sister, mom, and dad.

Acknowledgments

I would like to start by expressing my gratitude to my advisor Prof. Isabel Rodrigues for all the support, guidance and motivation provided throughout this thesis. Her availability and readiness to help was a major contributor to the conclusion of this work.

I would also like to acknowledge the financial support of Fundação para a Ciência e Tecnologia (FCT Portugal), through the research project PTDC/EGE-ECO/30535/2017 (Finance Analytics Using Robust Statistics).

Lastly, I would like to thank my family, in particular my parents and sister, for all their support and unrelenting encouragement all my life. For always believing in me. Without them this milestone would not have been possible.

Resumo

Recentemente, métodos para representar dados através de funções ou curvas têm vindo a ganhar atenção. Tais dados são conhecidos como dados funcionais. Conjuntos de dados funcionais típicos consistem em séries temporais e dados de corte transversal, como por exemplo séries temporais de preços de ações. Contudo, a presença de *outliers* tem efeitos adversos na modelação e previsão de dados funcionais. Neste contexto, deteção de *outliers* não é uma tarefa fácil considerando que nem sempre é possível visualizar todo o conjunto de dados funcionais (curvas, imagens ou funções). Todavia, processos para detetar *outliers* funcionais têm sido propostos nos últimos anos. Alguns destes processos são baseados na Análise de Componentes Principais Funcionais (ACPF), onde se assume que cada curva amostral provem de uma distribuição independente e idêntica. Esta suposição é inconsistente com dados financeiros, onde as amostras são frequentemente interligadas por um processo dinâmico temporal subjacente. Estas dinâmicas deveriam ser consideradas na deteção de *outliers* em séries temporais funcionais. Para lidar com esta situação, foi proposta uma extensão dinâmica de ACPF, que tem em consideração a estrutura de dependências dos dados funcionais. Posto isto, este trabalho introduz um método de deteção de *outliers* para séries temporais funcionais baseado nestas componentes principais dinâmicas. Esta técnica é, então, aplicada na deteção de *outliers* em dados financeiros.

Palavras-chave: Dados funcionais, séries temporais funcionais, deteção de *outliers*, componentes principais funcionais dinâmicas, dados financeiros

Abstract

In recent years methods for representing data through functions or curves have gained attention. Such data are known as functional data. Typical data sets consist of time series and cross-sectional data, such as time series of stock prices. However, the presence of outliers has adverse effects on the modelling and forecasting of functional data. In this context, outliers detection is not an easy task since the whole set of functional data (curves, images or functions) is not always possible to visualise. Nevertheless, procedures for detecting functional outliers have been proposed over recent years. Some of these procedures are based on Functional Principal Components Analysis (FPCA), which assumes each sample curve is drawn from an independent and identical distribution (i.i.d.). This assumption is inconsistent with financial data, where samples are often interlinked by an underlying temporal dynamic process. These dynamics should be taken into account to detect outliers in functional time series. To overcome this situation, a dynamic extension of FPCA that takes into account the dependence structure of the functional data has been proposed. Taking this into consideration, this work introduces an outlier detection method for functional time series based on these dynamic principal components. This technique is applied to anomaly detection in financial data.

Keywords: Functional data, functional time series, outlier detection, dynamic functional principal components, financial data

Contents

Acknowledgments v
Resumo vii
Abstract ix
List of Tables xiii
List of Figures xv

Acronyms **xvii**

1 Introduction **1**
1.1 Motivation 1
1.2 State Of The Art 2
1.3 Objectives 3
1.4 Thesis Outline 4

2 Background **5**
2.1 Functional Data 5
2.1.1 Representing Functional Data using Basis Functions 8
2.1.2 Smoothing by Least Squares 12
2.1.3 Smoothing with a Roughness Penalty 14
2.2 Functional Principal Components Analysis 16
2.2.1 Defining Functional Principal Components 17
2.2.2 Visualizing Principal Component Analysis 19
2.2.3 Computational Methods for Functional Principal Components Analysis 20
2.2.4 Regularised Principal Component Analysis 22
2.3 Functional Time Series 23
2.3.1 Functional Time Series Stationarity 23
2.4 Dynamic Functional Principal Components Analysis 25
2.5 Robust Functional Principal Components Analysis 26

3 Outlier Detection in Functional Data **29**
3.1 Methods Based on Distances 29
3.2 Methods Based on Graphical Tools 31

3.3 Method Based on Projections 33

4 Application to Financial Data Sets **35**

4.1 Banco Comercial Português 35

4.2 The Walt Disney Company 41

5 Conclusions **49**

5.1 Summary and Conclusions 49

5.2 Future Work 50

Bibliography **51**

A R code **55**

A.1 Banco Comercial Português 55

A.2 The Walt Disney Company 61

List of Tables

4.1 Outliers detected by the different graphical tools for the BCP data set. 37
4.2 Outliers detected using methods based on distances for the BCP data set. 38
4.3 Outliers detected using the method based on projections for the BCP data set. 40
4.4 Outliers detected by the different graphical tools for the DIS data set. 43
4.5 Outliers detected using methods based on distances for the DIS data set. 44
4.6 Outliers detected using the method based on projections for the DIS data set. 46

List of Figures

2.1 Examples of functional data, adapted from Ramsay et al. (2009). 6
2.2 Solid lines represent spline functions of orders 2, 3, and 4 fitting the sine (left) function and its derivative (right), the cosine function, both in dashed lines. The interior breakpoints are shown as the vertical dotted lines (Ramsay et al., 2009). 10
2.3 The eight B-spline basis functions of order four, with four equally spaced interior breaking points represented by the dashed lines, adapted from Ramsay et al. (2009). 11
2.4 Time series of the closing price of BCP stocks. The dashed lines indicate the place of the split, so that each coloured piece will correspond to a curve in the FTS. 24

4.1 Closing price of BCP stocks. 35
4.2 Variance-covariance and cross-correlation surfaces of the BCP data set. 36
4.3 Functional graphical tools for outlier detection obtained with R package `rainbow` for the BCP data set. 37
4.4 Functional graphical tools for outlier detection obtained with R package `roahd` for the BCP data set. 37
4.5 Smoothed BCP data obtained with two different basis systems. 38
4.6 Static principal components obtained with R package `fda` for the BCP data set. 39
4.7 Scores obtained from the dynamic functional principal components of the BCP data set. . 39
4.8 Outlying frequency of each curve using the three types of detection methods (graphical, distance and FPCA) for the BCP data set. 40
4.9 DIS stock prices. 41
4.10 CIDR transformation over the DIS data set. 42
4.11 Variance-covariance and cross-correlation surfaces of the DIS data set. 42
4.12 Functional graphical tools for outlier detection obtained with `rainbow` for the DIS data set. 43
4.13 Functional graphical tools for outlier detection obtained with `roahd` for the DIS data set. . 43
4.14 The DIS data smoothed through two different basis systems. 44
4.15 Static principal components obtained with R package `fda` for the DIS data set. 45
4.16 Scores obtained from the dynamic functional principal components of the DIS data set. . 45
4.17 Outlying frequency of each curve using the three types of detection methods (graphical, distance and FPCA) for the DIS data set (only outliers detected more than twice). 47

Acronyms

AIC Akaike Information Criterion.

BCP Banco Comercial Português.

BIC Bayesian Information Criterion.

CIDR Cumulative Intraday Returns.

DFPCA Dynamic Functional Principal Components Analysis.

DIS The Walt Disney Company Stock Price.

DWT Discrete Wavelet Transform.

FDA Functional Data Analysis.

FFT Fast Fourier Transform.

FPCA Functional Principal Components Analysis.

FTS Functional Time Series.

GCV Generalised Cross-Validation.

HDR Highest Density Region.

MBD Modified Band Depth.

MEI Modified Epigraph Index.

MSE Mean Squared Error.

MVE Minimum Volume Ellipsoid.

PCA Principal Components Analysis.

PSI-20 Portuguese Stock Index.

RFPCA Robust Functional Principal Components Analysis.

Chapter 1

Introduction

1.1 Motivation

In recent years data represented by functions or curves have been the receiving subject of a growing interest. More and more it is possible to obtain uninterrupted records of numerous types of data. This kind of data, which is observed over a continuous measure, is called functional data. The label “functional” arises from the fact that this data seem to have an underlying function or smooth curve that originated them. Therefore, time series data are a natural candidate to be treated as functional. In particular, this work will focus on financial time series consisting of stock prices.

As with any kind of data the presence of outlying observations has severe effects on forecasting and modelling of functional data. Thus, the first step on any descriptive analysis of the data set should be outliers detection, before any modelling or prediction method. In the functional context, however, this can be a very complex task due to the nature of the data and the difficulty in visualising the entire data set (curves, images or functions). Nonetheless, their analysis in functional data has been seldom addressed with several methods based on distances and FPCA. The problem is that FPCA assumes the sample curves are independent with each other. In financial data, samples are most likely interlinked by an underlying temporal dynamic process, leading to an inconsistency with the assumption of FPCA. Therefore, in order to properly detect outliers in Functional Time Series (FTS), the dynamics and dependence structure in the data must be taken into account. The Dynamic Functional Principal Components Analysis (DFPCA) was recently proposed and takes these dependences into consideration. However, although the DFPCA is a useful tool it has not been used for outliers detection yet.

Furthermore, intraday stock price curves are one of the most natural and obvious functional data. The functional perspective enables the analysis of the information present in the shapes of these curves, which reflects the reactions and expectations of intraday investors. Forecasting of the stock prices is also very important for the investors in the stock exchange markets. On that account, outlier detection becomes of the utmost importance in dealing with financial data.

1.2 State Of The Art

With the current technological evolution, it has become easier to record all kinds of data during a continuous time period or another continuous measure. This way, Functional Data Analysis (FDA) arises from the need to deal with data that are inherently described by functions. The term FDA was first introduced by Ramsay (1982) and Ramsay and Dalzell (1991), even though the history of this field started further back in the 1950s, *e.g.* Rao (1958). Ramsay and Dalzell (1991) also stated some of the advantages of turning to FDA like: a set of finite observations can be put through a smoothing process in order to obtain a functional approximation; some problems can be more naturally modelled if looked at from a functional perspective; derivatives of the observed functions could provide additional information, regarding for instance data visualisation.

Ramsay and Silverman (2005) comprise a detailed study on FDA, with the application of several multivariate data analysis methods adapted to the functional scenario. Moreover, Ramsay and Silverman (2002) show how FDA can be applied to several data sets from different fields of study. In fact, many applications have been put in practice like: growth curves (Rao, 1958), online auction prices (Wang et al., 2008), medicine (Erbas et al., 2007), climatology (Meiring, 2007), demography (Hyndman and Ullah, 2007; Hyndman and Shang, 2009), among many others.

One of the first multivariate data analysis techniques to be adjusted to FDA was Principal Components Analysis (PCA) by Dauxois et al. (1982). The idea, however, emerged earlier with Karhunen (1946) and Loève (1946) independently formulating what would be called the Functional Principal Components Analysis (FPCA) expansion or the Karhunen-Loève theorem. As expected, and in a similar fashion as with the FDA, FPCA has been applied in several fields of study, where medicine (Hasenstab et al., 2017), spatial data (Li et al., 2017), economics (Chong et al., 2015), stock market (Wang et al., 2014), electricity market (Vilar et al., 2016), and gene expression (Barra, 2004) are only a few examples.

With the evolving interest around FPCA, some refinements have been proposed. Rice and Silverman (1991), for example, added the smoothness to the estimation of FPCA using distinct roughness penalty strategies. Benko et al. (2009) presented common functional principal components estimated from discrete noisy data. Robust estimators for the principal components were proposed by Locantore et al. (1999), reducing the problem to a multivariate one. Hyndman and Ullah (2007) used a projection-pursuit approach to obtain a Robust Functional Principal Components Analysis (RFPCA), while Gervini (2008) proposed spherical principal components. Robust functional principal components estimators with a projection-pursuit approach were also proposed by Bali et al. (2011), and their asymptotic properties studied. In addition, Boente and Barrera (2015) suggested S-estimators for functional principal components.

Besides the proposals for RFPCA, there have also been some attempts at identifying outliers through the use of graphical tools (Hyndman and Shang, 2009; Arribas-Gil and Romo, 2014; Tarabelloni, 2017). Prior to these, the first efforts consisted of distance based methods, using depth measures (Febrero et al., 2007, 2008). Sawant et al. (2012), on the other hand, approaches the detection of outliers by applying robust PCA techniques to a coefficient matrix constructed from the coefficients in the expansion

2.14 introduced in Section 2.1. Moreover, Vilar et al. (2016) puts forward two outlier detection methods, one based on projections and another based on the residual of the original curve with an uncontaminated version of each curve, using RFPCA and then applying time series outliers detection methods.

Financial and economic data can usually be seen as a Functional Time Series (FTS), considering they comprise a set of curves registered continuously over time. In order to work with this type of data, it is necessary to test its stationarity. Horváth et al. (2014) proposes fully functional tests and tests based on projections that deal with this assumption of stationary Functional Time Series. More recently, Hörmann et al. (2015) proposed Dynamic Functional Principal Components Analysis (DFPCA), a functional version of dynamic principal components (Brillinger, 1981; Peña and Yohai, 2016), which takes into account information in the serial dependence of the functional data. Gao et al. (2018) also employs DFPCA in forecasting high-dimensional functional time series. Moreover, Kidziński et al. (2016) further extends this concept to periodically correlated FTS.

On the computational front, Ramsay et al. (2009) presents several practical applications using R and MATLAB, which is accompanied by the Functional Data Analysis website (Ramsay, 2013). Recently several R packages focusing on FDA have been made available, like the `fda` package (Ramsay et al., 2018), or the `fda.usc` one (Bande et al., 2019). The packages `rainbow` (Shang and Hyndman, 2019) and `refund.shiny` (Wrobel and Goldsmith, 2016) are more focused on graphical representations of functional data, while `roahd` (Tarabelloni et al., 2018) contains methods for the robust analysis of functional data. There are also some packages, `freqdom.fda`, `pcdpca` and `ftsa`, (Hormann and Kidziński, 2017; Kidziński et al., 2017; Hyndman and Shang, 2019) that deal with DFPCA. Package `ftsa` can also test the stationarity of FTS.

Wang et al. (2016) offers a thorough review of Functional Data Analysis, including some fundamental methods. Shang (2011), however, gives greater emphasis to surveying FPCA and functional principal component regression.

1.3 Objectives

This work aims to analyse two financial data sets from a functional perspective. The main objective is to compare several outliers detection methods among each other and with external social-economic events that could have caused the anomalies. Moreover, since most outliers detection methods are based on FPCA, an important aspect of the analysis is obtaining the principal components. As mentioned in Section 1.1, Dynamic Functional Principal Components Analysis is better suited to financial data sets. The idea is to transform the Functional Time Series into a vector time series of low dimension, where the individual component processes are mutually uncorrelated, and account for most of the dynamics and variability of the original process. Consequently, an adapted version of the outlier detection method based on projections introduced by Vilar et al. (2016) is proposed in this work. This new version is based on DFPCA, since the data in study is considered a FTS.

The first data set under study consists of yearly curves of the closing price of Banco Comercial Português (BCP)'s stocks. BCP is part of PSI-20 and Euronext Lisbon exchange market. This particular

data set is composed of 30 curves that range from 1989 to 2018 (366 points per curve). Thus, the objective is to determine which of these years can be considered as an anomaly, making use of several outlier detection methods available in R packages. In this case, it is possible to compare the years identified as having an abnormal behaviour to years of recession and financial crises or economic growth. A secondary objective is to compute static, dynamic and robust functional principal components, considering they are a fundamental part of some of the methods used.

The second data set is composed by intraday stock price curves regarding The Walt Disney Company Stock Price (DIS). In particular, it is comprised by 248 curves covering the year 2018. The objective is to identify the days over the year that had an anomalous behaviour, once again making use of different outliers detection methods. Consequently, it is necessary to obtain static, dynamic and robust functional principal components in order to apply some of these methods.

1.4 Thesis Outline

The objectives previously described are unfolded in this thesis throughout five chapters. After this introductory chapter, Chapter 2 provides the essential background of Functional Data Analysis. It introduces the main concepts and tools in FDA and a formal definition of functional data and its representation using basis functions. Moreover, the static FPCA is presented, which is the natural extension of Principal Components Analysis to the functional setting. What is more, the particular type of data in FDA covered in this thesis, Functional Time Series, will be introduced. Finally, the dynamic and robust FPCA will be exposed. Chapter 3 is devoted to presenting some tools for outliers detection in Functional Data. Methods based on distances, graphics and projections will be presented and a novel outlying method, also based on projections, will be proposed. In Chapter 4, the performance of the new outliers detection methodology together with some competitors was compared. In order to accomplish this, two real financial data sets were selected, the Banco Comercial Português and The Walt Disney Company. Both data sets were analysed as FTS. The R code used to conduct the analysis was relegated to the Appendix. Finally, Chapter 5 provides the concluding remarks of the dissertation, summarizing the main results and suggestions for future developments.

Chapter 2

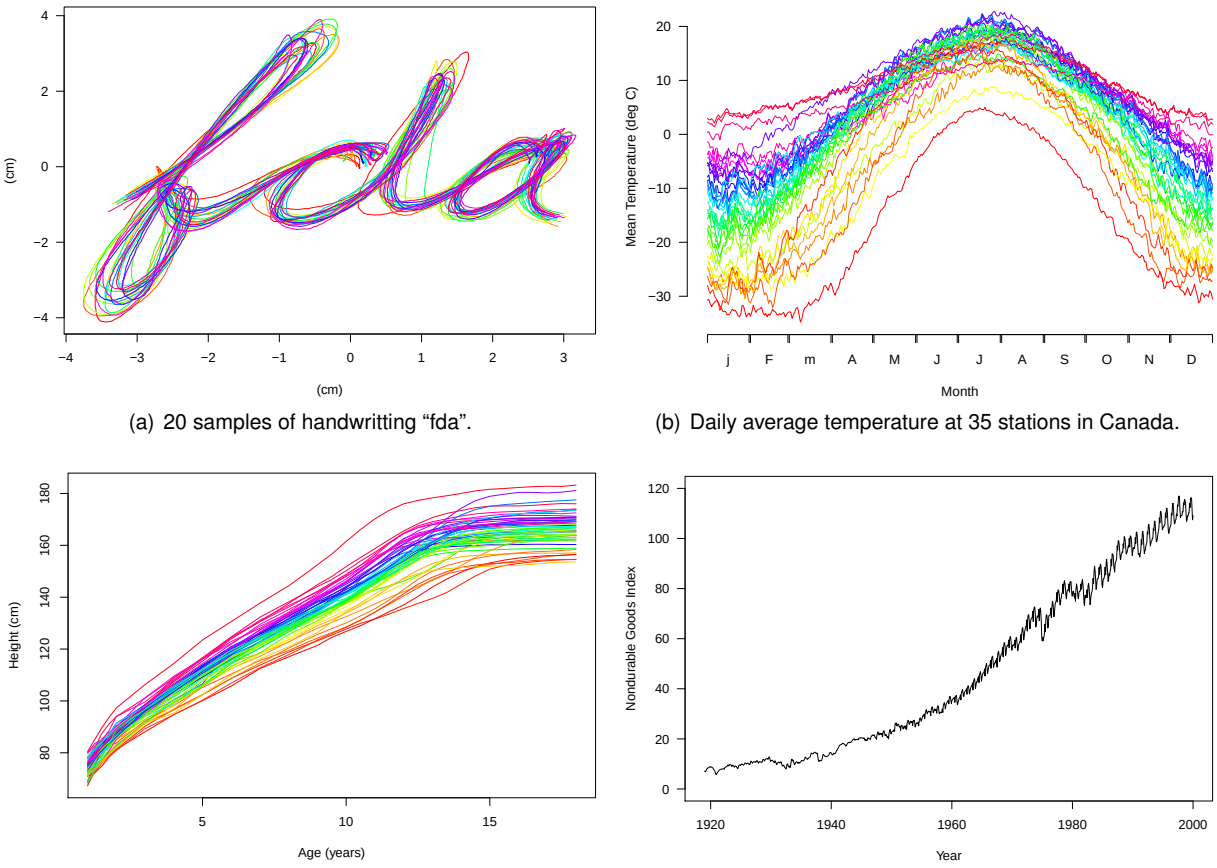
Background

2.1 Functional Data

Functional data can be obtained from the most various sources and may appear in several forms. Nevertheless, there is one constant characteristic underlying these functional data, they are comprised of functions or there is a function that originates the observed data. Thus, the observed data functions are considered as one entity. Moreover, functional data are continuously defined, regardless of in practice the observed data being typically discrete. These functions are usually defined over time, even though it could be any variable (*e.g.* spatial position, frequency, weight, ...). Some examples of functional data are represented in Figure 2.1.

The main goals of FDA are basically the same as most of other branches of statistics: representing the data in order to facilitate statistical reasoning and further analysis; displaying the data emphasizing important properties; studying variability and mean; among others. The FDA approach is also useful in a parsimonious representation of the data by taking advantage of their smoothness. Instead of looking at a function as a dense vector of values, it can often be represented in an linear combination of a handful of (well-chosen) basis functions. Much of the foundational description of FDA was recorded in Ramsay and Dalzell (1991), with a solid theoretical basis. In that sense, the first step is to define the space where FDA operates, which is not $\mathbb{R}^n$ as in multivariate analysis. The Hilbert space of square integrable absolutely continuous functions defined on a compact domain, $\mathcal{T} = [a, b]$, called $L^2_{[a,b]}$, is considered instead. This choice allows to take advantage of many properties of projection and distance inherent to a Hilbert space. From the construction of the space, it follows that a random variable X is a function such that $X : (\Omega, \Sigma, \mathbb{P}) \to L^2_{[a,b]}$, where $(\Omega, \Sigma, \mathbb{P})$ is a probability space. In more detail, Ω is the sample space, Σ is a sigma algebra on Ω and $\mathbb{P}$ is a probability measure on Σ. In this setting, each realization of X produces a function rather than a vector on a scalar. Thus, a functional random variable X is a real-valued function $X(t)$ considered as an element of the Hilbert space $L^2_{[a,b]}$, satisfying $\int_a^b X^2(t)dt < \infty$. From now on, it is considered that an integral sign without the limits of integration denotes the integral over the whole interval of the domain where the functions are defined.

Being able to measure distances between functional data is of most importance. Consequently, the



(a) 20 samples of handwritting "fda".

(b) Daily average temperature at 35 stations in Canada.

(c) Height of 54 different girls measured at 31 moments.

(d) US monthly nondurable goods index.

Figure 2.1: Examples of functional data, adapted from Ramsay et al. (2009).

definition of a norm that measures the distance between two elements is an essential tool in FDA.

In the L^2 Hilbert space, the following inner product $\langle \cdot, \cdot \rangle$ is defined for the functional variables X, Y, as $\langle X, Y \rangle = \int X(v)Y(v)dv$. The inner product generates the following L^2 norm $\|X\| = \langle X, X \rangle$. It is worth noting that a function e with $\|e\| = 1$ can be considered as a direction in the functional space and thus, $\langle X, e \rangle$ can be viewed as projecting the curve X over the direction given by e.

Additionally, an integral operator acting on $L^2_{[a,b]}$ called Φ has the form,

$$\Phi(x)(t) = \int \phi(t,x)x(s)ds, \quad t \in [a,b], \quad x \in L^2_{[a,b]}, \tag{2.1}$$

where $\phi(\cdot, \cdot)$ is called the integral kernel of Φ. Some properties of this kernel are listed in the following theorem known as Mercer's theorem (e.g. Kokoszka and Reimherr, 2017, Chapter 10).

Theorem 1. *Suppose $\phi(t,s)$ is a continuous, symmetric and non-negative definite integral kernel, and Φ is its corresponding integral operator defined in* (2.1)*. Then there is an orthonormal basis $(\varphi_i, i \in \mathbb{N})$ of $L^2_{[a,b]}$ consisting of eigenfunctions of Φ such that the corresponding eigenvalues $(\lambda_i, i \in \mathbb{N})$ are non-negative. $(\varphi_i, i \in \mathbb{N})$ and the corresponding $(\lambda_i, i \in \mathbb{N})$ are defined by,*

$$\lambda_i \varphi_i(t) = \Phi(\varphi_i)(t) = \int \phi(t,s)\varphi_i(s)ds, \quad \forall t \in [a,b], \quad \forall i \in \mathbb{N}. \tag{2.2}$$

Furthermore, ($\varphi_i, i \in \mathbb{N}$) are continuous on $[a, b]$ and $\phi(t, s)$ has the representation,

$$\phi(t,s) = \sum_{i=1}^{\infty} \lambda_i \varphi_i(t) \varphi_i(s), \tag{2.3}$$

where the convergence is absolute and uniform.

Moreover, the functional mean of X is defined as,

$$\mu(t) = \mathbb{E}[X(t)], \quad t \in [a, b], \tag{2.4}$$

and the covariance function as,

$$\text{Cov}(X(t), X(s)) = \gamma(t,s) = \mathbb{E}\left[(X(t) - \mu(t))(X(s) - \mu(s))\right]. \tag{2.5}$$

The corresponding positive and semi-defined covariance operator, $\Gamma : L^2_{[a,b]} \to L^2_{[a,b]}$, is such that,

$$\Gamma(x) = \mathbb{E}[\langle X - \mu, x \rangle (X - \mu)], \quad x \in L^2_{[a,b]}. \tag{2.6}$$

This leads to,

$$\Gamma(x)(t) = \int \gamma(t,s) x(s) ds, \quad t \in [a, b], \quad x \in L^2_{[a,b]}, \tag{2.7}$$

satisfying,

$$\int \int \gamma^2(t,s) dt ds < \infty. \tag{2.8}$$

Thus, Γ is an Hilbert–Schmidt integral operator and $\gamma(t, s)$ is its corresponding integral kernel. This operator is of key importance in Functional Principal Components Analysis (FPCA), as will be addressed in Section 2.2.

The random function X can be represented with the eigenfunctions of the covariance operator Γ, which is known as the Karhunen-Loéve representation given by the following theorem (e.g. Kokoszka and Reimherr, 2017, Chapter 11).

Theorem 2. *Suppose $X(t) \in L^2_{[a,b]}$ with $\mathbb{E}[X(t)] = \mu(t)$, then $X(t)$ can be represented by,*

$$X(t) = \mu(t) + \sum_{k=1}^{\infty} \langle X(t) - \mu(t), \varphi_k(t) \rangle \varphi_k(t), \tag{2.9}$$

where $\varphi_k(t)$ are the orthonormal eigenfunctions of the covariance operator Γ defined in (2.7)*.*

Observed Functional Data

Functional data are often considered smooth so that adjacent values are not too different from each other, which brings some advantages like the possibility of evaluation at any point. Smooth ordinarily means that the function has one or more derivatives. In practice the observed data is discrete, which is then used to estimate the function and some of its derivatives. If there is some observational error, this

transformation will likely use smoothing, while if there is no error, the data will go through interpolation.

Functions can be represented as a linear combination of basis functions. The use of these is a computational trick that allows to fit hundreds of thousands of data points, providing flexibility and the possibility of performing calculations resorting to the well-known matrix algebra. Accordingly, smoothing can be accomplished either using data-driven basis functions or known basis functions. The latter are, for example, the Fourier basis functions which are better suited for periodic data, or even B-splines for data without any strong cyclic variation.

Consider now the observed data vector $\boldsymbol{y} = (y_1, \ldots, y_n)$, where each y_j is a "snapshot" of the underlying function x. These can be expressed as,

$$y_j = x(t_j) + \epsilon_j, \tag{2.10}$$

where ϵ_j is the noise or error contributing with roughness to the data, and $t_j \in \mathcal{T}$ the interval over which data is collected. The standard model considers ϵ_j are assumed to be independently distributed with mean zero and constant variance σ^2. Rewriting (2.10) with vector notation results in,

$$\boldsymbol{y} = x(\boldsymbol{t}) + \boldsymbol{\epsilon}, \tag{2.11}$$

whose variance assuming the standard model is,

$$\text{Var}(\boldsymbol{y}) = \boldsymbol{\Sigma}_\epsilon = \sigma^2 \boldsymbol{I}_{n\times n}, \tag{2.12}$$

where $\boldsymbol{\Sigma}_\epsilon$ is the residual variance-covariance matrix. Instead of assuming the standard model, it could be created a model for $\boldsymbol{\Sigma}_\epsilon$, but this can slow down computation and waste degrees of freedom, while the result is usually very similar to what would be obtained using the standard model.

The sampling rate or resolution of the raw data is a local property indicating the density of the argument values t_j comparative to the amount of curvature in the data. This property is important to determine what is achievable in FDA. The curvature of a function x at argument t is calculated by the size of the second derivative, $|D^2x(t)|$ or $\left[D^2x(t)\right]^2$, where $D^m x$ represents the m-th derivative of a function x.

2.1.1 Representing Functional Data using Basis Functions

Each functional variable can be represented as a linear combination of known basis functions. A functional variable X belongs to an infinite dimensional space L^2 and, as a consequence, an infinite number of basis functions are needed in order to represent the functional variable with zero error. Hence, all the realizations $\{x_n\}_{n\in\mathbb{N}}$ of a functional random variable X can be expressed in relation to the same functional basis as,

$$x(t) = \sum_{k=1}^{\infty} c_k \phi_k(t), \tag{2.13}$$

where ϕ_k are known basis functions that are mathematically independent of each other and c_k are the coefficients or coordinates that represent each observation in the selected basis.

In practice, working with an infinite basis is not feasible and the usual approach is to truncate the number of basis functions to a finite K. So, a function x can be represented by a basis-expansion,

$$x(t) \approx \sum_{k=1}^{K} c_k \phi_k(t), \tag{2.14}$$

provided that K is sufficiently large. Therefore, each observation $x(t)$ is represented by a vector of finite scores $(c_1, \ldots, c_K)$. Thus, the curves dimensionality has been reduced from infinite to K. Considering $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_K)$ and $\boldsymbol{c} = (c_1, \ldots, c_K)$ one gets,

$$x = \boldsymbol{c}^\top \boldsymbol{\phi} = \boldsymbol{\phi}^\top \boldsymbol{c}, \tag{2.15}$$

where K is seen as a parameter to tune based on the data at hand. Optimally, there are more degrees of freedom to test hypothesis and obtain confidence intervals, if the K is small and the basis functions represent the data well. The computation also benefits from those conditions. However, there is not only one good basis. In fact, one criteria for distinguishing a good basis can be if one or more of the derivatives of the approximation are good estimates.

The Fourier Basis System

A very well known basis expansion used for periodic data is based on the Fourier series,

$$\hat{x}(t) = c_0 + c_1 \sin \omega t + c_2 \cos \omega t + c_3 \sin 2\omega t + c_4 \cos 2\omega t + \ldots, \tag{2.16}$$

resulting in the Fourier basis $\phi_0(t) = 1$, $\phi_{2r-1}(t) = \sin r\omega t$, and $\phi_{2r}(t) = \cos r\omega t$. This basis has period $\frac{2\pi}{\omega}$, meaning this periodicity should be present in the data to some degree. The basis functions are normalized, *i.e.*, their values are bounded, in this case in $[-1, 1]$.

The Fast Fourier Transform (FFT) is an algorithm that determines all the coefficients efficiently if n is a power of 2 and t_j are equally spaced (finding c_k and all n smooth values at $x(t_j)$ in $O(n \log n)$).

The Spline Basis System

In the event of non-periodic data the most popular approximation system is the spline basis one. Splines provide flexibility and fast computation of polynomials, with only a small number of basis functions.

Firstly, in order to define a spline it is necessary to split the interval $\mathcal{T}$ into L subintervals separated by values τ_l, $l = 1, \ldots, L-1$, called breakpoints or knots. Over each subinterval, the spline is a polynomial of order m (degree of the polynomial + 1). These polynomial segments must be smooth at the breakpoints, where its values must be the same. Their derivatives up to order $m - 2$ also need to coincide in these breakpoints.

Therefore, in order to define a spline function, one needs to establish the order of the polynomials and

the sequence τ_l of breakpoints. Hence, the number of degrees of freedom is the order of the polynomials plus the number of interior breakpoints: $m + L - 1$. Furthermore, increasing the number of breakpoints provides a greater flexibility to the spline, so that in regions where the function's variation is more complex there should be more breakpoints. On the other hand, increasing the order also produces a better fit. This is clearly exemplified in Figure 2.2, where spline functions of increasing order approximate $\sin(t)$ and its derivative over $[0, 2\pi]$.

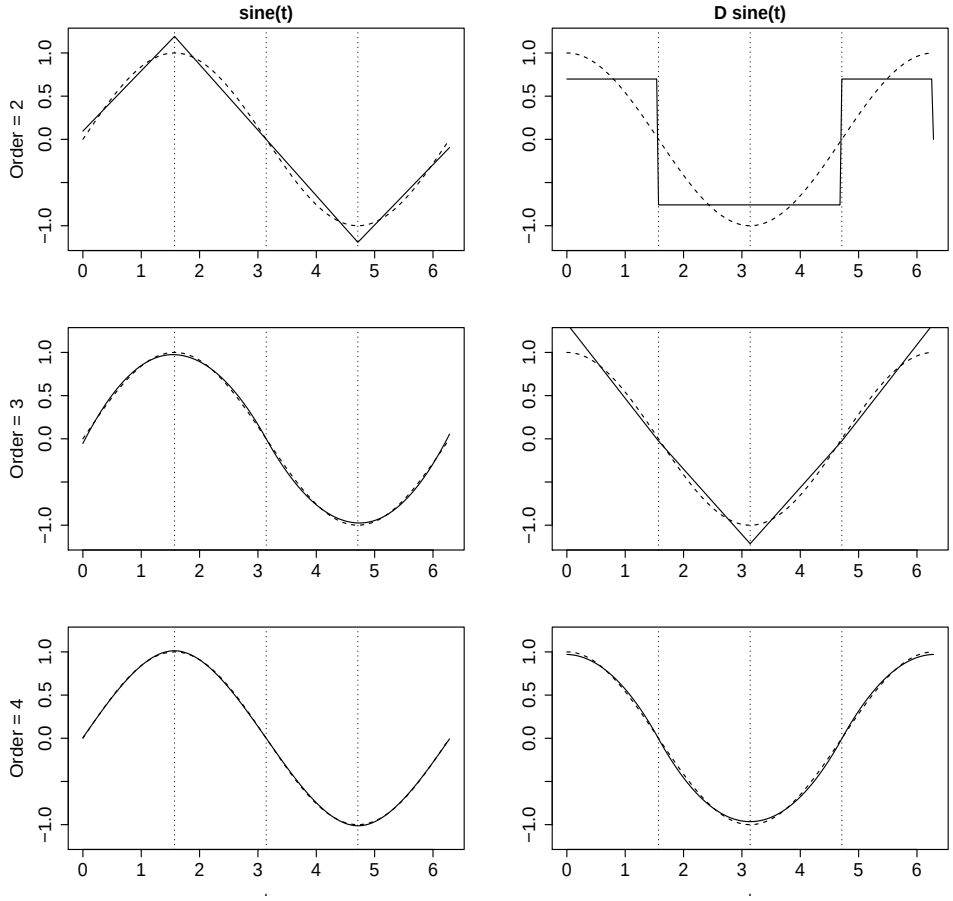


Figure 2.2: Solid lines represent spline functions of orders 2, 3, and 4 fitting the sine (left) function and its derivative (right), the cosine function, both in dashed lines. The interior breakpoints are shown as the vertical dotted lines (Ramsay et al., 2009).

B-spline Basis

In order to build a spline function, it is essential to specify a system of basis functions $\phi_k(t)$ with the subsequent characteristics: each basis function $\phi_k(t)$ is itself a spline function; a linear combination of $\phi_k(t)$ is a spline function; any spline function can be conveyed as a linear combination of $\phi_k(t)$.

The compact support property implies that a B-spline basis function of order m is positive over m or less adjacent intervals. This property is indispensable for efficient computation. It is also important to note that increasing the number K of B-splines, however, might not improve the approximation to the data. As with the Fourier system, the basis functions are bounded, in this case in $[0, 1]$.

In addition, $B_k(t, \tau)$ can be defined as the value at t of the B-spline basis function determined by the breakpoint sequence τ, where k is the number of the largest knot at or to the immediate left of t.

Thereby, a spline function $S(t)$ with discrete interior knots is defined as,

$$S(t) = \sum_{k=1}^{m+L-1} c_k B_k(t, \tau). \tag{2.17}$$

In Figure 2.3, it is possible to see the eight order four B-splines basis functions over the interval $[0, 10]$, with four equally spaced interior breakpoints. The six central basis functions begin at zero, then at a breakpoint increase to a peak, followed by a descent back to zero. These transitions are made smoothly because of the continuity of the cubic splines' two derivatives. The two central basis splines are also similar in shape due to the equally spaced knots. The first and last basis functions start growing from the first and last interior breakpoints until they reach the value one, respectively, on the left and right boundaries. Moreover, the two central basis functions are positive over four adjacent sub-intervals, the third and sixth basis functions are positive over three adjacent sub-intervals, while the second and seventh functions are positive over two. This illustrates the compact support property of the B-splines.

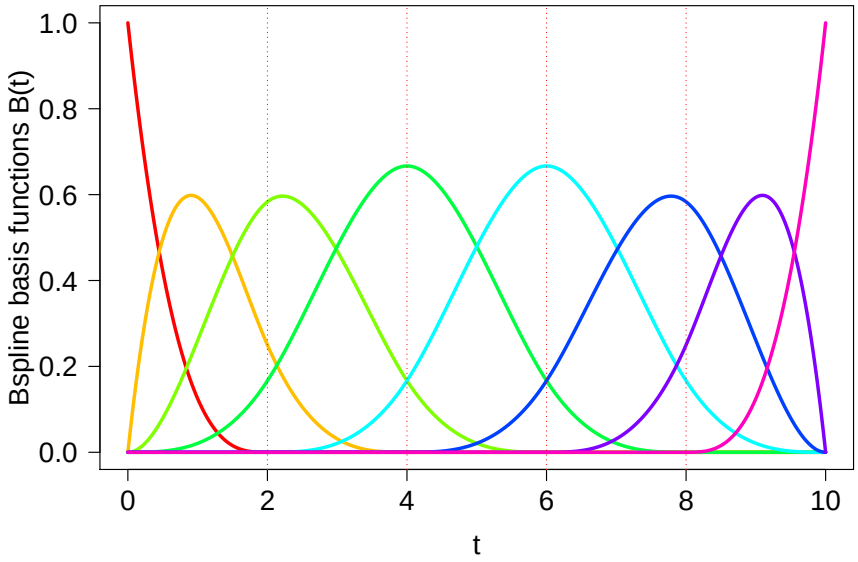


Figure 2.3: The eight B-spline basis functions of order four, with four equally spaced interior breaking points represented by the dashed lines, adapted from Ramsay et al. (2009).

Other Basis Systems

Besides the Fourier and Spline basis system, there are several other bases that can also have their own importance. Some of them consist in,

- **Wavelets**: Wavelets bring together some characteristics of the Fourier series and of splines. It is possible to build a basis for all square-integrable functions on $(-\infty, +\infty)$ using an appropriate mother wavelet function ψ and its dilations and translations of the form,

$$\psi_{jk}(t) = 2^{j/2}\psi(2^j t - k), \quad j, k \in \mathbb{Z}. \tag{2.18}$$

The integral of the product of two distinct basis function is zero, so the basis is orthogonal based on

the construction of the mother wavelet. In general, all the basis functions have compact support, since the mother wavelet does too. Wavelet expansions also handle discontinuities or drastic variations in behaviour, unlike Fourier series.

A Discrete Wavelet Transform (DWT) produces n coefficients related to the wavelet coefficients of a function x observed without error at $n = 2^M$ regularly spaced points on an interval $\mathcal{T}$. The DWT and its inverse can be obtained in $O(n)$.

- **Exponential Bases**: Exponential functions $e^{\lambda_1 t}, \ldots, e^{\lambda_k t}, \ldots$, where λ_k are the rate parameters and all different.
- **Power Bases**: $t^{\lambda_1}, \ldots, t^{\lambda_k}, \ldots$ might be important, specially if $t > 0$ so that the powers can be negative.
- **Polynomial Bases**: Monomial basis $\phi_k(t) = (t - \omega)^k$, $k = 0, \ldots, K$, where ω is a shift parameter, in general set to be in the centre of the interval of approximation. Only the use of a large K allows it to express local features. Additionally, polynomials have a good fit in the centre of the data unlike in the tails.
- **Empirical Bases**: It is possible to construct an empirical base recurring to Functional Principal Components Analysis (FPCA). The advantages of the functional principal components basis is that the coordinates are good descriptors of the data, thus it allows selecting a low number of basis functions while accounting for much of the variability of the original data. This subject is approached in Section 2.2.

2.1.2 Smoothing by Least Squares

Unweighted Least Squares

Based on (2.10) and (2.14), a linear smoother can be achieved by calculating c_k, through the minimization of the least squares criterion,

$$\text{SMSSE}(\boldsymbol{y}|\boldsymbol{c}) = \sum_{j=1}^{n} \left[y_j - \sum_{k=1}^{K} c_k \phi_k(t_j) \right]^2 = (\boldsymbol{y} - \boldsymbol{\Phi c})^\top (\boldsymbol{y} - \boldsymbol{\Phi c}), \tag{2.19}$$

where $\boldsymbol{\Phi} = [\phi_k(t_j)]_{jk}$ is a $n \times K$ matrix, and which results in the estimate,

$$\hat{\boldsymbol{c}} = (\boldsymbol{\Phi}^\top \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^\top \boldsymbol{y}. \tag{2.20}$$

This method is preferred when assuming the standard model for the error mentioned in Section 2.1.

Weighted Least Squares

Nonetheless, the standard model for the error might not always be suitable. So in cases where the errors are non-stationary or autocorrelated there is an extension of (2.19) using differential weighting of

residuals,

$$\text{SMSSE}(\boldsymbol{y}|\boldsymbol{c}) = (\boldsymbol{y} - \boldsymbol{\Phi c})^\top \boldsymbol{W}(\boldsymbol{y} - \boldsymbol{\Phi c}), \tag{2.21}$$

where $\boldsymbol{W}$ is a symmetric positive definite matrix, which makes it possible to have unequal weights on the squares and products of residuals. If $\boldsymbol{\Sigma}_\epsilon$ is known then $\boldsymbol{W} = \boldsymbol{\Sigma}_\epsilon^{-1}$. Again, this leads to the estimate,

$$\hat{\boldsymbol{c}} = (\boldsymbol{\Phi}^\top \boldsymbol{W} \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^\top \boldsymbol{W} \boldsymbol{y}. \tag{2.22}$$

Linear Transformations of the Data

In practice, most smoothing methods are linear, which is computationally simpler. So, $S_j(t_l)$ can weight the l-th discrete data value leading to the generation of the fit to y_j. On that account, a linear smoother estimates the function value $\hat{y}_j = \hat{x}(t_j)$ by a linear combination of the discrete observations,

$$\hat{x}(t_j) = \sum_{l=1}^{n} S_j(t_l) y_l, \tag{2.23}$$

writing now in matrix form,

$$\hat{x}(\boldsymbol{t}) = \boldsymbol{S}\boldsymbol{y}, \tag{2.24}$$

where $\boldsymbol{S}$ is called the smoothing matrix. In addition, $\boldsymbol{S}$ is a projection matrix, meaning the residual and fit vectors are orthogonal $(\boldsymbol{y} - \hat{\boldsymbol{y}})^\top \hat{\boldsymbol{y}} = 0$. In the case of the weight least squares, the relation turns into $(\boldsymbol{y} - \hat{\boldsymbol{y}})^\top \boldsymbol{W} \hat{\boldsymbol{y}} = 0$. This implies that $\boldsymbol{S}$ is also idempotent: $\boldsymbol{S}\boldsymbol{S} = \boldsymbol{S}$. In the case of the unweighted least squares, $\boldsymbol{S} = \boldsymbol{\Phi}(\boldsymbol{\Phi}^\top \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^\top$, while in the weighted least squares,

$$\boldsymbol{S} = \boldsymbol{\Phi}(\boldsymbol{\Phi}^\top \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^\top \boldsymbol{W}. \tag{2.25}$$

A linear smoother satisfies the linearity property,

$$\boldsymbol{S}(a\boldsymbol{y} + b\boldsymbol{z}) = a\boldsymbol{S}\boldsymbol{y} + b\boldsymbol{S}\boldsymbol{z}, \tag{2.26}$$

which is important to figure out some properties of the smooth representation. The linear smoother wins in simplicity and efficient computation. However, non-linear smoothers might adjust better to different behaviour in some regions of the data, or even be robust to outliers.

The number of unknown parameters in smoothing by least squares is K, hence the number of degrees of freedom (df) for the error is $n - K$. However, a more general expression for the degrees of freedom of the smooth fit is,

$$df = \text{trace}(\boldsymbol{S}). \tag{2.27}$$

Choosing K

As would be expected, the larger the K is, the better the approximation to the data will also be. However, this could lead to overfitting the data. In contrast, a smaller K can result in loosing important information.

On that account, it is possible to find an equilibrium through minimization of the Mean Squared Error (MSE) or the $\mathcal{L}^2$ loss function,

$$\text{MSE}[\hat{x}(t)] = \mathbb{E}\left[(\hat{x}(t) - x(t))^2\right] = \text{Bias}^2[\hat{x}(t)] + \text{Var}[\hat{x}(t)], \tag{2.28}$$

where $\text{Bias}[\hat{x}(t)] = x(t) - \mathbb{E}[\hat{x}(t)]$ and $\text{Var}[\hat{x}(t)] = \mathbb{E}[(\hat{x}(t) - \mathbb{E}[x(t)])^2]$.

Occasionally, other loss functions could be more efficient. For example, in the presence of outliers, it might be better to minimize the $\mathcal{L}^1$ norm, $\mathbb{E}[|\hat{x}(t) - x(t)|]$. Furthermore, there are several algorithms for selecting K, like stepwise variable selection or variable-pruning. Notwithstanding, none of these are full proof and should be used with caution.

2.1.3 Smoothing with a Roughness Penalty

The roughness penalty or regularization is a more efficient method for fitting a function to discrete data. It has the advantages of the basis function (and local expansion) smoothing, but evades some of its limitations, usually leading to better results particularly concerning the derivatives. Therefore, this method is generally preferred over the least squares. Roughness penalty methods are also founded on optimizing a fitting criterion, which in this case conveys explicitly the significance of smooth.

Spline Smoothing

The objective is to fit a non-periodic function x to discrete and noisy observations in a vector y. This is the simplest functional case over which it can be applied spline smoothing by using roughness penalties. This method produces its estimations by making explicit that the aim is assuring a good fit to the data, while avoiding a fit too good if it implies that x is too locally variable. This can be expressed through the previously stated formula: $\text{MSE} = \text{Bias}^2 + \text{Var}$. Hence, MSE can be minimized by allowing some bias so that the sampling variance will be smaller, which is of great importance to ensure smoothness on the fitted curve.

The curvature of a function x at t is $\left[D^2x(t)\right]^2$, given that a straight line (no curvature) has a zero second derivative. Therefore, a possible measure of the function's roughness is,

$$\text{PEN}_2(x) = \int \left[D^2x(s)\right]^2 ds, \tag{2.29}$$

which tends to high values if x is highly variable, since its second derivative is large over at least some of the interval in study. However, it might be necessary to generalize this roughness penalty as,

$$\text{PEN}_m(x) = \int \left[D^m x(s)\right]^2 ds. \tag{2.30}$$

Adapting (2.21), so that $\text{PEN}_2(x)$ is taken into consideration, leads to a new criterion to be minimized

that balances smoothing with data fitness. Thus, the penalized residual sum of squares is defined as,

$$\text{PENSSE}_\lambda(x|\boldsymbol{y}) = [\boldsymbol{y} - x(\boldsymbol{t})]^\top \boldsymbol{W}[\boldsymbol{y} - x(\boldsymbol{t})] + \lambda \text{PEN}_2(x), \tag{2.31}$$

where $x(\boldsymbol{t})$ is the vector obtained from evaluating x at the argument values of vector t. In addition, λ is a smoothing parameter that controls the trading ratio between the fit to the data (1st term) and the function's variability (2nd term). Accordingly, as $\lambda \to \infty$ there is an increasing emphasis on the smoothness. On the other hand, as $\lambda \to 0$, the roughness penalty decreases, and so the emphasis turns to obtaining a better fit to the data.

Assuming that x has a second derivative and that t_j, $j = 1, \ldots, n$, are distinct, the function x that minimizes $\text{PENSSE}_\lambda(x|\boldsymbol{y})$ is a cubic spline with a knot at each data point t_j. The knots' placement will lead to an exploitation of areas where there is a great amount of observations and to substantial smoothness where there are few. Hereupon, cubic spline smoothing is the most usual computational method for spline smoothing. It consist of using a B-spline basis function expansion of order four (piece-wise cubic fitting function) with a knot at each sampling point, minimizing (2.31) with respect to the coefficients of the expansion. Hence, there are $n + 2$ basis functions to fit the n data points.

The roughness penalty (2.30) can be rewritten in matrix form,

$$\begin{aligned}
\text{PEN}_m(x) &= \int \left[D^m x(s)\right]^2 ds \overset{(2.15)}{=} \int \left[D^m \boldsymbol{c}^\top \boldsymbol{\phi}(s)\right]^2 ds \\
&= \int \boldsymbol{c}^\top D^m \boldsymbol{\phi}(s) D^m \boldsymbol{\phi}^\top(s) \boldsymbol{c} ds = \boldsymbol{c}^\top \left[\int D^m \boldsymbol{\phi}(s) D^m \boldsymbol{\phi}^\top(s) ds\right] \boldsymbol{c} \\
&= \boldsymbol{c}^\top \boldsymbol{R} \boldsymbol{c},
\end{aligned} \tag{2.32}$$

by defining,

$$\boldsymbol{R} = \int D^m \boldsymbol{\phi}(s) D^m \boldsymbol{\phi}^\top(s) ds. \tag{2.33}$$

This leads to,

$$\text{PENSSE}_m(\boldsymbol{y}|\boldsymbol{c}) = (\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{c})^\top \boldsymbol{W}(\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{c}) + \lambda \boldsymbol{c}^\top \boldsymbol{R} \boldsymbol{c}, \tag{2.34}$$

and to the estimated coefficient vector,

$$\hat{\boldsymbol{c}} = (\boldsymbol{\Phi}^\top \boldsymbol{W} \boldsymbol{\Phi} + \lambda \boldsymbol{R})^{-1} \boldsymbol{\Phi}^\top \boldsymbol{W} \boldsymbol{y}. \tag{2.35}$$

As a result, the estimated vector $\hat{\boldsymbol{y}}$ is,

$$\hat{\boldsymbol{y}} = \boldsymbol{\Phi}(\boldsymbol{\Phi}^\top \boldsymbol{W} \boldsymbol{\Phi} + \lambda \boldsymbol{R})^{-1} \boldsymbol{\Phi}^\top \boldsymbol{W} \boldsymbol{y} = \boldsymbol{S}_{\phi,\lambda} \boldsymbol{y}, \tag{2.36}$$

where,

$$\boldsymbol{S}_{\phi,\lambda} = \boldsymbol{\Phi}(\boldsymbol{\Phi}^\top \boldsymbol{W} \boldsymbol{\Phi} + \lambda \boldsymbol{R})^{-1} \boldsymbol{\Phi}^\top \boldsymbol{W}. \tag{2.37}$$

If $\lambda = 0$, then the expression is reduced to (2.25). Moreover, unlike $\boldsymbol{S}$ defined in (2.25), $\boldsymbol{S}_{\phi,\lambda}$ is not idempotent, considering that $\boldsymbol{S}_{\phi,\lambda} \boldsymbol{S}_{\phi,\lambda} \neq \boldsymbol{S}_{\phi,\lambda}$. This fact indicates that the spline smooth of a spline

smooth is smoother. What is more, $\boldsymbol{S}_{\phi,\lambda}$ can be used in calculating a value for degrees of freedom for a spline smooth,

$$df(\lambda) = \text{trace}(\boldsymbol{S}_{\phi,\lambda}). \tag{2.38}$$

The estimation of derivatives might be of interest, however (2.29) only controls the curvature in x, so it does not demand anything from the second derivative. Consequently, if the highest derivative required is of order m, then the derivatives of order $m + 2$ should be penalized, so that the curvature of the derivative with highest order can be controlled.

This spline smoothing method can be extended in several ways. For example, if the knots are less than the sampling points the mathematics do not suffer any changes, and a roughness penalty can still be an effective method. Another example is using another measure of goodness-of-fit to the data, or even another measure of roughness.

Choosing the Smoothing Parameter

Taking into consideration computational limitations, $\lambda\boldsymbol{R}$ should not be greater than 10^{10} times the size of $\boldsymbol{\Phi}^\top\boldsymbol{W}\boldsymbol{\Phi}$. Moreover, λ should be large enough so that the size of $\lambda\boldsymbol{R}$ is at least with ten orders of magnitude of the size of $\boldsymbol{\Phi}^\top\boldsymbol{W}\boldsymbol{\Phi}$.

One possible method for choosing the smoothing parameter is the leave-one-out cross-validation. In this method one observation is left out of the estimation in order to be used as a validation sample. This is repeated leaving at each time a different observation out, resulting in the cross-validated error sum of squares. Additionally, it is computed over a range of values for λ, being chosen as the smoothing parameter the λ that leads to the minimum of the error. However, this method can be demanding computation-wise, and can often result in under-smoothing the data.

A more popular method is the Generalised Cross-Validation (GCV). Unlike cross-validation, it has no need to re-smooth n times and is less prone to under-smooth. This criterion can be expressed as,

$$\text{GCV}(\lambda) = \frac{n^{-1}\text{SSE}}{[n^{-1}\text{trace}(\boldsymbol{I} - \boldsymbol{S}_{\phi,\lambda})]^2}, \tag{2.39}$$

or even,

$$\text{GCV}(\lambda) = \left(\frac{n}{n - df(\lambda)}\right)\left(\frac{\text{SSE}}{n - df(\lambda)}\right), \tag{2.40}$$

where SSE is the sum of squared estimated errors. Nevertheless, minimizing GCV with respect to λ will require trying a great amount of values of λ. This could be done by grid-search or with a numerical optimization algorithm.

2.2 Functional Principal Components Analysis

Functional Principal Components Analysis (FPCA) is one of the most important techniques in FDA. It is used to reduce the dimension of infinitely dimensional functional data to a small finite dimension. The FPCA can be seen in two ways, as coordinates maximizing variability, or as an optimal orthonormal basis

(empirical bases). On the other hand, interpreting the variance-covariance and correlation functions can be a hard task. As such, FPCA can be a very helpful alternative at retrieving information from the covariance structure.

2.2.1 Defining Functional Principal Components

The main idea behind Principal Components Analysis (PCA) for multivariate data is to find sets of normalized weights, $\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_p$ that maximize variation in linear combinations of variable values, $f_{11}, \ldots, f_{Np}$. Considering x_{ij} as the observed values of the jth variable, this can be done in a stepwise manner:

1. Find $\boldsymbol{\xi}_1 = (\xi_{11}, \ldots, \xi_{p1})^\top$ for which the values $f_{i1} = \sum_j \xi_{j1} x_{ij} = \boldsymbol{\xi}_1^\top x_i$ have maximum mean square, $N^{-1} \sum_i f_{i1}^2$, subject to $\sum_j \xi_{j1}^2 = \|\boldsymbol{\xi}_1\|^2 = 1$.

2. The subsequent steps, up until a limit of the number of variables p, are the same. On the mth step calculate $\boldsymbol{\xi}_m$ in the same way as in the first step with the additional $m - 1$ constraint(s): $\sum_j \xi_{jk} \xi_{jm} = \boldsymbol{\xi} k^\top \boldsymbol{\xi}_m = 0$, $k < m$.

On the first step, one is maximizing the mean square, thus singling out the strongest variation in the variables. The constraint on the weights is needed so that the problem is well defined. On the subsequent steps, the additional constraint forces the weights to be orthogonal with the ones already computed. Furthermore, the weights $\boldsymbol{\xi}_m$ are not defined uniquely and the values f_{im} are called principal component scores. What is more, before performing PCA, it is usual to subtract each variable's mean from the corresponding variable values, meaning that, when maximizing the mean square of f_{im}, one is actually maximizing their sample variance.

Subsequently, extending this idea to functional data, the variable values x_{ij} are now function values $x_i(s)$. Since in this case $x_i(s)$, $i = 1, \ldots, N$, and $\xi(s)$ are functions, the sample principal component scores are,

$$f_i = \langle \xi, x_i \rangle = \int \xi(s) x_i(s) ds, \quad i = 1, \ldots, N, \tag{2.41}$$

and the functional PCA steps are:

1. Choose $\xi_1(s)$ that maximizes $N^{-1} \sum_i f_{i1}^2 = N^{-1} \sum_i \left(\int \xi_1 x_i \right)^2$, subject to $\int \xi_1(s) ds = \|\xi_1\|^2 = 1$.

2. Again, the subsequent steps are the same as step 1, but with the added orthogonality constraint: $\int \xi_k \xi_m = 0$, $k < m$.

Principal Component Analysis and Eigenanalysis

Another way of looking at PCA is through the eigenanalysis of the variance-covariance function or operator. In the multivariate data case, PCA can be accomplished by computing the eigenvalues and eigenvectors of the covariance or correlation matrix. In that sense, there is a functional equivalent to this process. In order to do this, it is assumed that the mean function values have been subtracted from the observed $x_i(t)$, $i = 1, \ldots, N$, meaning that their cross-sectional means, $N^{-1} \sum_i x_i(t)$, are zero.

Accordingly, the sample covariance function is defined by,

$$v(s,t) = N^{-1} \sum_{i=1}^{N} x_i(s) x_i(t). \tag{2.42}$$

The principal component weight functions $\xi(s)$, for a suitable eigenvalue ρ, satisfy the equation,

$$\int v(s,t)\xi(t)dt = \rho\xi(s). \tag{2.43}$$

The left side of (2.43) is an integral transform of ξ, called the covariance operator V defined by,

$$V\xi = \int v(\cdot,t)\xi(t)dt. \tag{2.44}$$

Hence, the equation (2.43) can also be expressed as,

$$V\xi = \rho\xi. \tag{2.45}$$

This way, functional PCA is also equivalent to performing an eigenanalysis of the covariance operator V. Therefore, the eigenanalysis of V results in the pairs of eigenvalues/eigenfunction, $(\rho_i, \xi_i(\cdot))$, by the Mercer's theorem. The eigenvalues are order descending, $\rho_i \geqslant \rho_{i+1}$, and the eigenfunctions, $\xi_i(\cdot)$s, correspond to the functional principal components. However, in this case, the maximum number of different eigenvalues/eigenfunctions is the number of function values, ergo infinity. Nevertheless, considering the functions x_i are linearly independent, V has rank $N-1$ and, as such, there are only $N-1$ non-zero eigenvalues.

Optimal Empirical Orthonormal Basis

As mentioned before in Section 2.1.1, it is possible to find a set of K orthonormal basis functions ξ_k, whose expansion of each curve fits it as best as possible. This expansion is a sample approximation of the Karhunen-Loève expansion given by the Karhunen-Loève theorem (2),

$$x_i(t) \approx \hat{x}_i(t) = \sum_{k=1}^{K} f_{ik}\xi_k(t). \tag{2.46}$$

In fact, if one is trying to minimize a global measure of approximation like,

$$\text{PCASSE} = \sum_{i=1}^{N} \|x_i - \hat{x}_i\|^2 = \sum_{i=1}^{N} \int \left[x_i(s) - \hat{x}_i(s)\right]^2 ds, \tag{2.47}$$

the basis of choice that minimizes (2.47) is precisely the set of principal component weight functions.

Selecting the Number K of Principal Components

Another important aspect is choosing how many principal components (K) to use in the approximation. This is not straightforward and there is not an optimal way of choosing. Some approaches (Li et al., 2013) are based on the Akaike Information Criterion (AIC) or on the Bayesian Information Criterion (BIC). On the other hand, some multivariate PCA strategies can also be applied to the functional PCA. For example, the scree plot or the percentage of explained variance by the first K principal components given by,

$$\frac{\sum_{k=1}^{K} \rho_k}{\sum_{m=1}^{\infty} \rho_m}. \tag{2.48}$$

2.2.2 Visualizing Principal Component Analysis

One of PCA's hardships is that interpreting the components can be a very complicated matter. A technique that might help with this problem is plotting the overall mean function, $\hat{\mu}$, and the resulting functions from adding and subtracting to $\hat{\mu}$ a multiple of the principal component function. In order to choose this multiple, a constant C is defined,

$$C^2 = T^{-1}\|\hat{\mu} - \bar{\mu}\|^2, \tag{2.49}$$

where,

$$\bar{\mu} = T^{-1} \int \hat{\mu}(t)dt, \tag{2.50}$$

and T is the length of time over which the integration takes place. Then, it is plotted, for example, $\hat{\mu}$ and $\hat{\mu} \pm 0.2C\xi_i$, where the constant 0.2 should be adjusted according to the behaviour of $\hat{\mu}$.

An additional way of looking at PCA is through plotting the principal component scores f_{im} of each curve on each component. Yet another approach can be to rotate the principal components into a new set of just as good orthonormal functions. Therefore, considering ξ as the vector-valued function, $(\xi_1, \ldots, \xi_K)^\top$, and $\boldsymbol{T}$ as an orthonormal matrix of order K ($\boldsymbol{T}^\top\boldsymbol{T} = \boldsymbol{T}\boldsymbol{T}^\top = \boldsymbol{I}$), another orthonormal set is defined by,

$$\psi = \boldsymbol{T}\xi. \tag{2.51}$$

On that account, the vector of functions ψ is a rigid rotation of ξ. However, ψ_1 is no longer the largest component of variation, but $\psi_1, \ldots, \psi_K$ are just as good at fitting the original curves.

In an attempt to find a rotation that will be easier to interpret, another multivariate analysis tool can be retrieved, the VARIMAX rotation. Defining $\boldsymbol{B}$ as a $K \times n$ matrix representing the first K principal component functions $\xi_1, \ldots, \xi_K$, its m-th row has values $\xi_m(t_1), \ldots, \xi_m(t_n)$, for n equally spaced argument values in $\mathcal{T}$. Then, the matrix $\boldsymbol{A}$ of values of the rotated functions corresponding to (2.51) is,

$$\boldsymbol{A} = \boldsymbol{T}\boldsymbol{B}. \tag{2.52}$$

VARIMAX chooses $\boldsymbol{T}$ by maximizing the variation in the values a_{mj}^2 of the matrix $\boldsymbol{A}$, arranged as a single vector. Seeing that $\boldsymbol{T}$ is a rotation matrix, $\sum_m \sum_j a_{mj}^2$ does not change with the rotation, *i.e.*, $\sum_m \sum_j a_{mj}^2 = \text{trace}(\boldsymbol{A}^\top\boldsymbol{A}) = \text{trace}(\boldsymbol{B}^\top\boldsymbol{T}^\top\boldsymbol{T}\boldsymbol{B}) = \text{trace}(\boldsymbol{B}^\top\boldsymbol{B})$. Maximizing the variance of the

a_{mj}^2 requires that these values are either relatively large or near zero. This fact is what makes these components easier to interpret, since the information tends to be condensed.

2.2.3 Computational Methods for Functional Principal Components Analysis

Some possible techniques for solving (2.43) involve converting the continuous functional eigenanalysis problem into a somewhat equivalent matrix eigenanalysis one. One way to accomplish this is by discretising the observed functions x_i, obtaining a grid of n equally spaced values s_j, spread over $\mathcal{T}$. This results in a $N \times n$ matrix $\boldsymbol{X}$ that can be used in a classic multivariate PCA. Considering

$$\boldsymbol{V}\boldsymbol{u} = \lambda \boldsymbol{u}, \tag{2.53}$$

where $\boldsymbol{u}$ is a n-dimensional vector, this equation is satisfied by the eigenvalues and eigenvectors provided by the PCA. Then, it is necessary to convert the principal components back into functional domain. Taking into account that the sample variance-covariance is $\boldsymbol{V} = N^{-1}\boldsymbol{X}^\top \boldsymbol{X}$ with elements $v(s_j, s_k)$, $\tilde{\boldsymbol{\xi}}$ is a n-vector with values $\xi(s_j)$, and $\omega = \frac{T}{n}$, then for each s_j,

$$V\xi(s_j) = \int v(s_j, s)\xi(s)ds \approx \omega \sum v(s_j, s_k)\tilde{\boldsymbol{\xi}}_k. \tag{2.54}$$

Therefore, an approximate discrete form of (2.45) is,

$$\omega \boldsymbol{V}\tilde{\boldsymbol{\xi}} = \rho\tilde{\boldsymbol{\xi}}, \tag{2.55}$$

whose solutions are the same as the ones of (2.53), where the eigenvalues are $\rho = \omega\lambda$. The normalization constraint here is $\omega\|\tilde{\boldsymbol{\xi}}\|^2 = 1$, leading to $\tilde{\boldsymbol{\xi}} = \omega^{-1/2}\boldsymbol{u}$, with $\boldsymbol{u}$ a normalized eigenvector of $\boldsymbol{V}$. Lastly, one can apply an interpolation method to transform the discrete values $\tilde{\boldsymbol{\xi}}$ into an approximate function ξ.

Another approach to dealing with (2.43) is to use known basis functions ϕ_k in the expansion of the function x_i. How many basis functions (K) are used will depend on several factors like the number of discrete sampling points (n) in the data, or the need for some smoothing ($K < n$), among others. Supposing that,

$$x_i(t) \approx \sum_{k=1}^{K} c_{ik}\phi_k(t), \tag{2.56}$$

plus that $\boldsymbol{x} = (x_1, \ldots, x_N)$, $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_K)$, and $\boldsymbol{C}_{N\times K}$ is the coefficient matrix, then,

$$\boldsymbol{x} = \boldsymbol{C}\boldsymbol{\phi}. \tag{2.57}$$

Hence, the variance-covariance function can be given by,

$$v(s,t) = N^{-1}\boldsymbol{\phi}(s)^\top \boldsymbol{C}^\top \boldsymbol{C}\boldsymbol{\phi}(t). \tag{2.58}$$

It is also defined a $K \times K$ symmetric matrix, $\boldsymbol{W}$, with entries $w_{k_1,k_2} = \int \phi_{k_1}\phi_{k_2}$, that is $\boldsymbol{W} = \int \boldsymbol{\phi}\boldsymbol{\phi}^\top$.

Moreover, ξ has an expansion $\xi(s) = \sum_{k=1}^{K} b_k \phi_k(s)$, or $\xi(s) = \boldsymbol{\phi}(s)^\top \boldsymbol{b}$. This results in,

$$\int v(s,t)\xi(t)dt = \int N^{-1}\boldsymbol{\phi}(s)^\top \boldsymbol{C}^\top \boldsymbol{C}\boldsymbol{\phi}(t)\boldsymbol{\phi}(t)^\top \boldsymbol{b}dt = \boldsymbol{\phi}(s)^\top N^{-1}\boldsymbol{C}^\top \boldsymbol{C}\boldsymbol{W}\boldsymbol{b}.$$

Thus, (2.43) can be written as $\boldsymbol{\phi}(s)^\top N^{-1}\boldsymbol{C}^\top \boldsymbol{C}\boldsymbol{W}\boldsymbol{b} = \rho\boldsymbol{\phi}(s)^\top \boldsymbol{b}$. Seeing that it must hold for all s, the matrix equation is,

$$N^{-1}\boldsymbol{C}^\top \boldsymbol{C}\boldsymbol{W}\boldsymbol{b} = \rho\boldsymbol{b}. \tag{2.59}$$

It is important to take note that $\|\xi\| = 1$ implies $\boldsymbol{b}^\top \boldsymbol{W}\boldsymbol{b} = 1$ and ξ_1 and ξ_2 are orthogonal if and only if $\boldsymbol{b}_1^\top \boldsymbol{W}\boldsymbol{b}_2 = 0$, where $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$ are the corresponding vectors of coefficients. Therefore, in order to obtain the principal components, it is necessary to define $\boldsymbol{u} = \boldsymbol{W}^{1/2}\boldsymbol{b}$, solve,

$$N^{-1}\boldsymbol{W}^{1/2}\boldsymbol{C}^\top \boldsymbol{C}\boldsymbol{W}^{1/2}\boldsymbol{u} = \rho\boldsymbol{u}, \tag{2.60}$$

and calculate $\boldsymbol{b} = \boldsymbol{W}^{-1/2}\boldsymbol{u}$ for each eigenvector. A special case is if the basis is orthonormal, which implies that $\boldsymbol{W} = \boldsymbol{I}$, the FPCA problem is reduced to the multivariate PCA of $\boldsymbol{C}$ and the eigenanalysis is done over $N^{-1}\boldsymbol{C}^\top \boldsymbol{C}$.

Numerical Quadrature

In order to discretise the integral $\int x_i(s)\xi(s)ds$ it is required to approximate it by a sum of discrete values. As such, nearly all numerical integration or quadrature strategies entail an approximation of the form,

$$\int f(s)ds \approx \sum_{j=1}^{n} w_j f(s_j). \tag{2.61}$$

Three parameters in this approximation can be adjusted in accordance to objectives: n (number of discrete argument values s_j), s_j (quadrature points) and w_j (quadrature weights).

Employing (2.61) to the operator V in (2.45), leads to,

$$V\xi \approx \boldsymbol{V}\boldsymbol{W}\tilde{\boldsymbol{\xi}}, \tag{2.62}$$

which results in the equivalent eigenanalysis problem,

$$\boldsymbol{V}\boldsymbol{W}\tilde{\boldsymbol{\xi}} = \rho\tilde{\boldsymbol{\xi}}, \tag{2.63}$$

with the orthonormality constraint $\tilde{\boldsymbol{\xi}}_m^\top \boldsymbol{W}\tilde{\boldsymbol{\xi}}_m = 1$ and $\tilde{\boldsymbol{\xi}}_{m_1}^\top \boldsymbol{W}\tilde{\boldsymbol{\xi}}_{m_2} = 0$, for $m_1 \neq m_2$.

Bearing in mind that the majority of quadrature schemes use positive weights and defining $\boldsymbol{u} = \boldsymbol{W}^{1/2}\tilde{\boldsymbol{\xi}}$ and $\boldsymbol{u}^\top \boldsymbol{u} = 1$, it is possible to approximate the eigenequation through,

$$\boldsymbol{W}^{1/2}\boldsymbol{V}\boldsymbol{W}^{1/2}\boldsymbol{u} = \rho\boldsymbol{u}. \tag{2.64}$$

Then the steps to take are,

1. Select n, the w_j's and the s_j's.

2. Calculate the eigenvalues ρ_m and eigenvectors $\boldsymbol{u}_m$ of $\boldsymbol{W}^{1/2}\boldsymbol{V}\boldsymbol{W}^{1/2}$.

3. Calculate $\tilde{\boldsymbol{\xi}}_m = \boldsymbol{W}^{-1/2}\boldsymbol{u}_m$.

4. Optional: convert each vector $\tilde{\boldsymbol{\xi}}$ into a function ξ_m, through an interpolation method.

The maximum number of recovered approximate eigenfunctions is n, if $n < N$. Nonetheless, most PCA applications only need a few of the main eigenfuctions, given that they will explain most of the variability in the data.

2.2.4 Regularised Principal Component Analysis

It would also be pertinent to consider the application of smoothing to FPCA. In order to do so, one can use the regularization method discussed in Section 2.1 by incorporating it into the PCA, beginning by taking into account the leading principal component. It is necessary to make the roughness penalty $\text{PEN}_2(\xi) = \int \xi''(t)^2 dt$ of the estimated principal component ξ as small as possible without jeopardizing too much the maximization of the sample variance. This trade is once again controlled by a smoothing parameter, $\lambda \geqslant 0$, that adjusts the roughness penalty.

Thus, the penalized sample variance is defined as,

$$\text{PCAPSV}(\xi) = \frac{\text{Var}\left(\int \xi x_i\right)}{\|\xi\|^2 + \lambda \text{PEN}_2(\xi)}. \tag{2.65}$$

Once again, if $\lambda \to 0$ this measure reduces to the sample variance, while if $\lambda \to \infty$ the component ξ is obliged to be the smoothest possible resulting in a linear component.

The subsequent components are estimated by maximizing PCAPSV(ξ), while satisfying the two constraints: $\|\xi\|^2 = 1$ and a modified orthogonality condition,

$$\int \xi_j(s)\xi_k(s)ds + \int D^2\xi_j(s)D^2\xi_k(s)ds = 0, \quad k = 1, \ldots, j-1. \tag{2.66}$$

This condition enables the estimation of all principal components through the solution of one eigenvalue problem.

Once again, a possible way of choosing the smoothing parameter λ is by cross-validation. It is supposed that x is an observation and that an expansion in terms of $\xi_1, \ldots, \xi_m$ is the one with m functions that best explains the variation in x (principal components property). Therefore, in order to measure this explained variation, it is necessary to define x^* as the projection of x onto the subspace spanned by $\xi_1, \ldots, \xi_m$, and ζ_m as the residual $x - x^*$ so that it is orthogonal to the functions $\xi_1, \ldots, \xi_m$. Hence, a possible candidate to measure the efficiency of these first m components is $\sum_m \mathbb{E}\left[\|\zeta_m\|^2\right]$. Therefore, the cross-validation procedure is:

1. Calculate the observed data x_i minus the overall mean.

2. $\xi_j^{[i]}(\lambda)$ is the estimate of ξ_j obtained from the data with the exception of x_i, for a given λ.

3. $\zeta_m^{[i]}(\lambda)$ is the x_i's component orthogonal to the subspace covered by $\{\xi_j^{[i]}(\lambda) : j = 1, \ldots, m\}$.

4. Compute the cross-validation scores,

$$\mathsf{CV}_m(\lambda) = \sum_{i=1}^{n} \|\zeta_m^{[i]}(\lambda)\|^2, \tag{2.67}$$

leading to,

$$\mathsf{CV}(\lambda) = \sum_{m=1}^{\infty} \mathsf{CV}_m(\lambda). \tag{2.68}$$

In practice, this sum is truncated at most at $n - 1$ (n is the number of data curves), given that it is the maximum of principal components that it is possible to estimate.

5. Choose the λ that minimizes CV(λ).

An alternative to the regularised principal components is to smooth the data and only afterwards perform a regular PCA. In this case, any method for smoothing can be applied to the data.

2.3 Functional Time Series

In many practical situations functions are naturally ordered in time. For example, when dealing with daily observations of the stock market. A Functional Time Series (FTS) is a sequence of curves $(X_t(u) : t \in \mathbb{Z}, u \in \mathcal{T})$, where t is a discrete parameter and u is a continuous one. In fact, t is the time index, which typically refers to day or year, and u is the time within that unit. For example, when dealing with daily observations, t refers to the day and u is the intra-day time parameter. Thus, the main idea behind Functional Time Series is that the time record can be split into natural intervals, treating the curve within each interval as a unit.

As a motivating example, Figure 2.4 shows the closing stock price of Banco Comercial Português (BCP). In this case the discrete parameter t is considered as the year, and u as the days within that year. Hence, the split here is made by year, so that each year corresponds to a curve. In this case, an assumption of independence between the curves would be too strong, considering values at the beginning of each year are correlated with values at the end of the previous year. This indicates significant temporal dependence.

2.3.1 Functional Time Series Stationarity

Many procedures assume the stationarity of FTS. In that sense, the null hypothesis that the series is stationary can be written as, for any h and any t,

$$(X_{1+h}, \ldots, X_{t+h}) \stackrel{d}{=} (X_1, \ldots, X_t), \tag{2.69}$$

where two random variables X and Y are equal in distribution ($X \stackrel{d}{=} Y$) if $P(X \leqslant x) = P(Y \leqslant x)$, $\forall x$.

Figure 2.4: Time series of the closing price of BCP stocks. The dashed lines indicate the place of the split, so that each coloured piece will correspond to a curve in the FTS.

Horváth et al. (2014) introduces tests which are nontrivial extensions of the KPSS family tests. Two classes of tests are considered, based on the curves and on the finite dimensional projections of the curves on the functional principal components. The fully functional tests are based on two test statistics,

$$T_N = \int \int Z_N^2(x,t)dtdx, \tag{2.70}$$

$$M_N = T_N - \int \left(\int Z_N(x,t)dx \right)^2 dt = \int \int \left(Z_N(x,t) - \int Z_N(y,t)dy \right)^2 dxdt, \tag{2.71}$$

where,

$$Z_N(x,t) = N^{-1/2} \sum_{t=1}^{\lfloor Nx \rfloor} X_t(u) - xN^{-1/2} \sum_{t=1}^{N} X_t(u), \quad 0 \leqslant x,u \leqslant 1, \tag{2.72}$$

and N is the number of observed curves in the data. The tests based on projections take advantage of the eigenvalues, λ_i ($\lambda_1 > \cdots > \lambda_{d+1} > 0$), and eigenfunctions, φ_i, of the covariance operator to define,

$$T_N^0(d) = \sum_{i=1}^{d} \frac{1}{\hat{\lambda}_i} \int \langle Z_N(x,\cdot), \hat{\varphi}_i \rangle^2 dx, \tag{2.73}$$

$$T_N^*(d) = \sum_{i=1}^{d} \int \langle Z_N(x,\cdot), \hat{\varphi}_i \rangle^2 dx, \tag{2.74}$$

$$M_N^0(d) = \sum_{i=1}^{d} \frac{1}{\hat{\lambda}_i} \int \left(\langle Z_N(x,\cdot), \hat{\varphi}_i \rangle - \int \langle Z_N(u,\cdot), \hat{\varphi}_i \rangle du \right)^2 dx, \tag{2.75}$$

$$M_N^*(d) = \sum_{i=1}^{d} \int \left(\langle Z_N(x,\cdot), \hat{\varphi}_i \rangle - \int \langle Z_N(u,\cdot), \hat{\varphi}_i \rangle du \right)^2 dx. \tag{2.76}$$

Critical values for these two statistics (under the stationarity null hypothesis) can be obtained based on Monte Carlo simulations (Horváth and Kokoszka, 2012) or on normal approximations (Shorack and Wellner, 1986).

A common example of FTS are intraday price curves. In that setting, the price of a financial asset at time t_j on day n is defined as $P_n(t_j)$, $n = 1, \ldots, N$, $j = 1, \ldots, m$. The tests when applied to this kind of data and to sufficiently long periods of time reject the stationarity. For shorter periods of time, the rejection of stationarity does not always happen. Therefore, a transformation should be employed to deal with the non-stationarity, resulting in the Cumulative Intraday Returns (CIDR), which were first introduced by Gabrys et al. (2010) and can be defined as,

$$R_n(t_j) = 100 \left[\ln P_n(t_j) - \ln P_n(t_1)\right], \quad n = 1, \ldots, N, j = 1, \ldots, m. \tag{2.77}$$

CIDR, however, are not directly comparable to daily returns, since they disregard the overnight price change. The CIDR do enable the statistical analysis of the intraday price curves' shapes.

2.4 Dynamic Functional Principal Components Analysis

The FPCA presented in Section 2.2 is considered "static", since it does not take into account possible dependencies between the several observations or repetitions of the function. In some applications the data are dependent, like, for example, daily curves of financial transactions or environmental data. Therefore, considering the observations as independent might conduct to misleading results. In order to deal with this shortcoming, Hörmann et al. (2015) proposed a "dynamic" alternative to FPCA. The data should be considered as a stationary Functional Time Series (FTS), $(X_t(u) : t \in \mathbb{Z})$, where t is a discrete parameter and u is a continuous one. The objective of Dynamic Functional Principal Components Analysis (DFPCA) is very similar to PCA's in general, obtain a transformation of FTS into a low dimension vector time series, whose components are uncorrelated to each other and explain most of the dynamics and variability of the original data. Gao et al. (2018) does just that by making use of DFPCA as part of a dimension reduction technique in order to forecast high-dimensional FTS.

Considering $\mu(u) = \mathbb{E}[X_t(u)]$, as the continuous mean function and the auto-covariance function at lag h as,

$$\gamma_h(u, s) = \mathsf{Cov}(X_t(u), X_{t+h}(s)), \tag{2.78}$$

the long-run covariance function is,

$$\gamma(u, s) = \sum_{h=-\infty}^{\infty} \gamma_h(u, s). \tag{2.79}$$

This leads to the definition of the operator,

$$\Gamma(x)(u) = \int \gamma(u, s)x(s)ds, \tag{2.80}$$

which admits an eigendecomposition,

$$\Gamma(x) = \sum_{k=1}^{\infty} \lambda_k \left(\int x(u)\varphi_k(u)du \right) \varphi_k, \tag{2.81}$$

where λ_k, $k \geqslant 1$, are the eigenvalues (in descending order) and φ_k, $k \geqslant 1$, its corresponding eigenfunctions.

Analogously to FPCA, $X_t(u)$ can be approximated using the Karhunen-Loève expansion making use of the first K principal components obtained,

$$X_t(u) \approx \sum_{k=1}^{K} \beta_{t,k}\varphi_k(u), \tag{2.82}$$

where the k-th principal component score at time t is given by,

$$\beta_{t,k} = \int X_t(u)\varphi_k(u)du. \tag{2.83}$$

Therefore, the observed FTS, $x_t(u)$, can be approximated using the empirical principal components. The DFPCA is implemented in the R package `freqdom.fda`.

2.5 Robust Functional Principal Components Analysis

The FPCA's main use is dimension reduction, but it can also serve as an instrument for outlier detection. Consequently, a robust version of FPCA should be considered to deal with outlying observations. To that end, Hyndman and Ullah (2007) introduce a two-step algorithm for Robust Functional Principal Components Analysis (RFPCA), which combines the weighted principal component method and the RAPCA projection pursuit algorithm (Hubert et al., 2002). The point is to find the functions $\varphi_k(u)$ which maximize the variance of the scores,

$$z_{t,k} = w_t \int \varphi_k(u)X_t(u)du, \tag{2.84}$$

subject to,

$$\int \varphi_k^2(u)du = 1 \quad \text{and} \quad \int \varphi_k(u)\varphi_{k-1}(u)du = 0 \text{ if } k \geqslant 2.$$

The weights w_t are given by,

$$w_t = \begin{cases} 1 & \text{if } \upsilon_t < s + \lambda\sqrt{s} \\ 0 & \text{otherwise} \end{cases} \tag{2.85}$$

where s is the median of $\{\upsilon_1, \ldots, \upsilon_t\}$ and $\lambda > 0$ is a tuning parameter that controls the level of robustness (the smaller λ is, the greater the number of curves deemed as outliers). Additionally, υ_t is the integrated

squared error given by,

$$\upsilon_t = \int e_t^2(u)du = \int \left(X_t(u) - \sum_{k=1}^{K} \beta_{t,k}\varphi_k(u) \right)^2 du, \tag{2.86}$$

where K is the number of components, φ_k are the principal component functions, and $\beta_{i,k}$ their scores.

Thus, the two-step algorithm follows,

1. Obtain initial values for $\beta_{t,k}$ and $\varphi_k(u)$, using the RAPCA algorithm (Hubert et al., 2002) and assuming equal weights w_t.

2. Update the values of w_t according to (2.85), and use them to update $\beta_{t,k}$ and $\varphi_k(u)$ through the weighted principal component method described in (Hyndman and Ullah, 2007, Section 3.1).

Assuming that $e_t(u)$ is normally distributed for large enough K, υ_t has a χ^2 distribution. Hence, using a normal approximation, the probability that $\upsilon_t < s + \lambda\sqrt{s}$ is approximately $\Phi(\lambda/\sqrt{2})$, where $\Phi(\cdot)$ is the distribution function of the standard normal distribution. This is also the efficiency of the algorithm compared to the classical approach when $w_t = 1$ for all t. For example, $\lambda = 3$ results in an efficiency of $\Phi(3/\sqrt{2}) = 98.3\%$.

This algorithm has been implemented in the R package `ftsa`.

Chapter 3

Outlier Detection in Functional Data

Data exploration which aims to reveal some aspects of the dynamics of the underlying process is an important task. In that sense, outlier detection plays a crucial part, considering outliers can have severe adverse effects on the modelling and forecasting of the data. There are two types of outlying curves: "magnitude outliers", curves that lie outside the range of the rest of the data; "shape outliers", curves that have a different shape of the rest of the data even though they are within their range; they can also be a combination of these two types. In this chapter, several methods to detect outliers in functional data are presented.

3.1 Methods Based on Distances

The earliest attempt at outlier detection was based on the idea of distance, with Febrero et al. (2007) using a likelihood ratio test statistic. The statistic is given by,

$$\Lambda = \max_{1 \leqslant t \leqslant N} O_\alpha(X_t), \tag{3.1}$$

where,

$$O_\alpha(X_t) = \left\| \frac{X_t - \hat{\mu}_{TM,\alpha}}{\hat{\sigma}_{TM,\alpha}} \right\|, \tag{3.2}$$

and where $\|\cdot\|$ is a norm in the functional space ($\|\cdot\|_1$, $\|\cdot\|_2$ or $\|\cdot\|_\infty$). Furthermore, $\hat{\mu}_{TM,\alpha}$ is the α-trimmed mean and $\hat{\sigma}_{TM,\alpha}$ is the α-trimmed standard deviation. Considering the curves $X_1, \ldots, X_N$ and given a value α, they are defined as,

$$\hat{\mu}_{TM,\alpha} = \frac{1}{N - [\alpha N]} \sum_{t=1}^{N-[\alpha N]} X_{(t)}, \tag{3.3}$$

$$\hat{\sigma}_{TM,\alpha} = \left(\frac{1}{N - [\alpha N]} \sum_{t=1}^{N-[\alpha N]} \left(X_{(t)} - \hat{\mu}_{TM,\alpha} \right)^2 \right)^{\frac{1}{2}}. \tag{3.4}$$

The ordered curves, $X_{(1)}, \ldots, X_{(N)}$, are ranked according to decreasing values of their functional

depths,

$$FD(X_t) = \int D(X_t(u))du, \quad t = 1, \ldots, N, \tag{3.5}$$

where D is a univariate depth defined on $\mathbb{R}$. Hereupon, the outlier detection method follows,

1. Given $X_1, \ldots, X_N$, compute the statistic (3.1).

2. Let X_M be the curve that attains the maximum value of the statistic (3.1). Then, X_M is an outlier if $\Lambda = O_\alpha(X_M) > C$ (threshold C is obtained with a bootstrap procedure).

3. If X_M is an outlier, remove it from the sample. Then repeat steps 1 and 2, until no more outliers are discovered.

Febrero et al. (2008) proposes another method based on functional depths. Functional depths measure the centrality of a curve within a sample of curves. In that sense, depths allow to order the sample curves from the centre outwards. Thus, depth and outlyingness are inverse notions, which means magnitude outliers are expected to have a significantly low depth. Therefore, the following functional outlier detection method, looks for the curves with lower depths,

1. Obtain the functional depths $D_N(X_1), \ldots, D_N(X_N)$.

2. Let $X_{t_1}, \ldots, X_{t_k}$ be the k curves such that $D_N(X_{t_k}) \leqslant C$, for a given cutoff C. Then, $X_{t_1}, \ldots, X_{t_k}$ are considered outliers. Delete them from the sample.

3. Repeat steps 1 and 2, until no more outlliers are found.

Step 3 deals with the masking problem, where some outliers might mask the presence of others. Moreover, the selection of C is the key feature of the method, so C is selected such that, in the absence of outliers,

$$\mathbb{P}(D_N(X_t) \leqslant C) = 0.01, \quad t = 1, \ldots, N. \tag{3.6}$$

The threshold C is found by robustly estimating this percentile, using the observed sample curves. That is possible through two different bootstrap procedures. One is based on trimming the sample of suspicious curves. The other is based on bootstrapping the curves of the original data set with probability proportional to their depth.

Hyndman and Ullah (2007) advance another alternative which is a product of the robust functional principal components algorithm introduced in Section 2.5. It is built on the integrated squared error given by (2.86), which is used to update the weights w_t (2.85). Consequently, a curve is considered an outlier if its associated weight w_t is zero. The higher the integrated squared error is, the most likely a curve can be considered an outlier.

Another distance to consider is the familiar robust Mahalanobis distance. If the functional data are recorded on equally spaced points $u_1, \ldots, u_p$, the squared robust Mahalanobis distance is given by,

$$r_t = (X_t(u_j) - \hat{\mu})^\top \hat{\Sigma}^{-1} (X_t(u_j) - \hat{\mu}), \quad t = 1, \ldots, N, \quad j = 1, \ldots, p, \tag{3.7}$$

where $\hat{\mu}$ is a robust estimate of the sample mean and $\hat{\Sigma}$ is a robust estimate of the covariance matrix of $X_t(u_j)$. Rousseeuw and Leroy (1987) proposes to compute the squared robust Mahalanobis distance relative to the Minimum Volume Ellipsoid (MVE) estimates for $\hat{\mu}$ and $\hat{\Sigma}$. Thus, $\hat{\mu}$ is the centre of the minimal volume ellipsoid covering at least $[N/2]+1$ points, while $\hat{\Sigma}$ is given by the ellipsoid itself (multiplied by a factor). Additionally, $\hat{\Sigma}$ is supposed to be positive definite, in order to assure that $\hat{\Sigma}^{-1}$ is nonsingular. These distances are compared to a threshold that follows a χ^2 distribution with p degrees of freedom. For example if $\alpha = 0.99$, an observation is considered an outlier if $r_t > \chi^2_{0.99,p}$.

All these distance based methods are implemented in the R package `rainbow`.

3.2 Methods Based on Graphical Tools

The difficulty in visualising the whole set of functional curves can complicate the detection of outliers. As such, Hyndman and Shang (2009) proposed two graphical methods that can identify outliers: the functional bagplot and the Highest Density Region (HDR) boxplot. These methods are also available in the R package `rainbow`.

The functional bagplot shows the median curve (curve with the greatest depth), inner region (region enclosed by all curves corresponding to points in the bivariate bag and containing 50% of the curves) and outer region (region confined by all curves corresponding to points within the bivariate fence region). Thus, the functional bagplot is obtained by mapping the bagplot of the first two robust principal component scores to the functional curves. Although this method can detect outliers when they are distant from the median, it can misidentify outliers near the median. The functional HDR is more suitable to deal with those.

The functional HDR is obtained by computing the bivariate kernel density estimate on the first two robust principal component scores, and then applying the bivariate HDR boxplot. A Highest Density Region is defined as,

$$R_\alpha = \{\boldsymbol{z} : \hat{f}(\boldsymbol{z}) \geqslant f_\alpha\}, \tag{3.8}$$

where $\hat{f}(\boldsymbol{z})$ is the bivariate kernel density estimate and f_α is such that $\int_{R_\alpha} \hat{f}(\boldsymbol{z})d\boldsymbol{z} = 1-\alpha$. Thus, it is the region with probability coverage $1-\alpha$ and where points have a higher density estimate. The functional HDR shows the modal curve (curve with the highest density), inner region (region limited by all curves corresponding to points inside the 50% bivariate HDR, containing 50% of the curves) and outer region (region limited by all curves corresponding to points within the outer bivariate HDR). Consequently, curves outside the outer HDR are considered outliers.

Still within the graphical domain, Arribas-Gil and Romo (2014) introduce the outliergram, which helps identify shape outliers by taking advantage of the relation between two measures of depth for functional data. These two depth measures consist of the Modified Band Depth (MBD) and the Modified Epigraph Index (MEI). They offer an idea of how central or deep a curve is with respect to a sample of curves. Let $X_1, \ldots, X_N$ be N continuous functions defined on a given closed real interval $\mathcal{T}$. For any $X \in$

$\{X_1, \ldots, X_N\}$ the MBD is defined as,

$$MBD_{\{X_1,\ldots,X_N\}}(X) = \binom{N}{2}^{-1} \sum_{i=1}^{N} \sum_{j=i+1}^{N} \frac{\lambda(\{u \in \mathcal{T} : \min(X_i(u), X_j(u)) \leqslant X(u) \leqslant \max(X_i(u), X_j(u))\})}{\lambda(\mathcal{T})}, \tag{3.9}$$

where $\lambda(\cdot)$ is the Lebesgue measure on $\mathbb{R}$. Moreover, each pair of curves X_i and X_j in the sample defines in $\mathcal{T} \times \mathbb{R}$ a band, $\{(u, y) : u \in \mathcal{T}, \min(X_i(u), X_j(u)) \leqslant y \leqslant \max(X_i(u), X_j(u))\}$. Hence, $MBD_{\{X_1,\ldots,X_N\}}(X)$ can be seen as the mean over all possible bands of the proportion of time that $X(u)$ spends inside a band.

Likewise, the MEI of $X \in \{X_1, \ldots, X_N\}$ is defined as,

$$MEI_{\{X_1,\ldots,X_N\}}(X) = \frac{1}{N} \sum_{i=1}^{N} \frac{\lambda(\{u \in \mathcal{T} : X_i(u) \geqslant X(u)\})}{\lambda(\mathcal{T})}, \tag{3.10}$$

and it represents the mean proportion of time that X lies below the curves of the sample. Considering MBD and MEI are closely related, the outliergram consists of mapping the (MEI, MBD) points. As such, in order to identify the outliers, it is necessary to define the distances, for $i = 1, \ldots, N$,

$$d_i = -\frac{2}{N(N-1)} + \frac{2(N+1)}{N-1} MEI_{\{X_1,\ldots,X_N\}}(X_i) - \frac{2N}{N-1} MEI^2_{\{X_1,\ldots,X_N\}}(X_i) - MBD_{\{X_1,\ldots,X_N\}}(X_i). \tag{3.11}$$

Therefore, shape outliers are defined as the curves with $d_i \geqslant Q_{d3} + 1.5IQR_d$, where Q_{d3} and $1.5IQR_d$ are the third quantile and inter-quartile range of $d_1, \ldots, d_N$, respectively. An additional step is required for extreme curves (MEI values close to 0 or 1). In this case, the extreme curves are vertically shifted towards the centre of the sample one by one. If this new (MEI, MBD) point lies in the previous outlying region, then the curve is considered a shape outlier.

Tarabelloni (2017) also proposes a robust adjusted functional boxplot, this one better suited to the detection of magnitude outliers. To construct the functional boxplot it is necessary to use a depth measure to rank the functions from the centre of the distribution outwards. Ranking a sample of N curves in decreasing order, $X_{(1)}, \ldots, X_{(N)}$, with respect to a given depth definition, leads to the definition of,

$$C_\alpha(X) = \left\{ (u, z(u)) : \min_{l=1,\ldots,\lceil \alpha N \rceil} X_{(l)}(u) \leqslant z(u) \leqslant \max_{r=1,\ldots,\lceil \alpha N \rceil} X_{(r)}(u) \right\}. \tag{3.12}$$

Here, C_α is the generic, sample, functional α-central region of the data, meaning it contains the $\alpha \times 100\%$ most central observations of the sample. The functional boxplot is obtained with the following steps:

1. Obtain region $C_{0.5}$, containing the 50% most central curves of the sample.
2. Inflate it by a factor $F \geqslant 1$ and construct the fences given by the envelope of the functions entirely contained inside the inflated region.
3. Curves crossing the fences or completely external are identified as outliers.

Here the greatest challenge is tuning the parameter F. This is done through a Gaussian population simulated using robust estimators.

Both Arribas-Gil and Romo (2014)'s outliergram and Tarabelloni (2017)'s functional boxplot are available in the R package `roahd`.

3.3 Method Based on Projections

More recently, Vilar et al. (2016) introduced two algorithms that detect outliers in Functional Time Series, which take advantage of FPCA. The following algorithm consists of an adapted version of the method based on projections,

1. Apply FPCA and obtain the time series of principal components scores $\{\beta_{t,1} \ldots, \beta_{t,K}\}_{t=1}^{n}$.
2. Apply a time series outlier detection method on the series obtained.
3. Define the set of outliers as $\mathcal{O} = \{X_t : t \in \mathcal{I}\}$, where $\mathcal{I} = \{t : (\beta_{t,1} \ldots, \beta_{t,K})$ was identified as outlier$\}$.

This algorithm allows to combine the power of FPCA with methods that detect outliers in time series. The projections $\beta_{t,k}$ display the most prominent characteristics of the data. As such, one of these projections is considered an outlier, only if its originating curve is also an outlier.

In the implementation of this algorithm it is necessary to specify the type of FPCA and a time series outliers detection method. In this work, it will be tested the use of static, dynamic and robust FPCA. It is also required to choose the threshold parameter K (number of principal components to retain). According to the sensitivity studies performed by Vilar et al. (2016), it is recommended to select K so that at least $98\%/99\%$ of the variability is explained. This value is so high because this method uses the scores and not the original curves. Moreover, the first scores are related to the presence of magnitude outliers, while the scores of higher order are related to the presence of shape outliers.

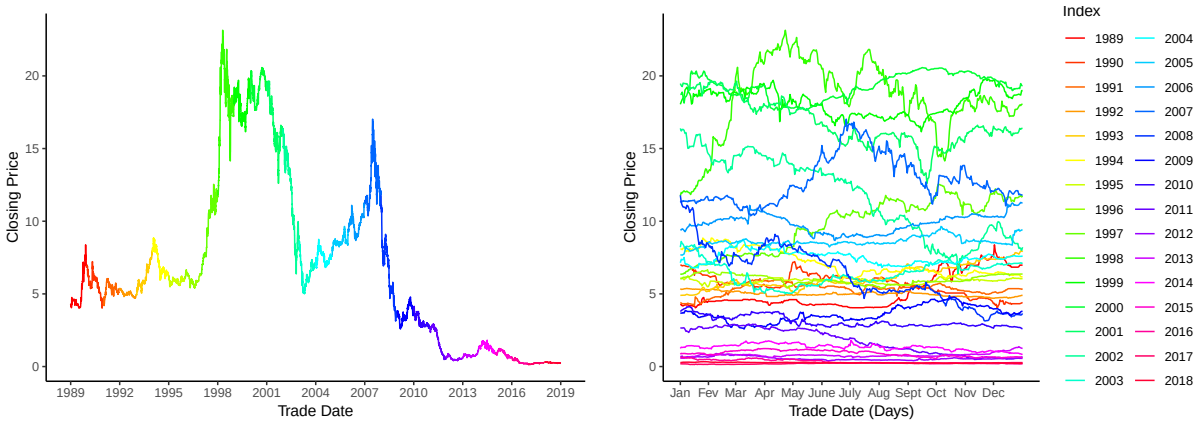
Chapter 4

Application to Financial Data Sets

In this chapter, two data sets are introduced and studied from a Functional Time Series perspective. Additionally, the results from the outlier detection methods introduced in Chapter 3 applied to these data sets are presented.

4.1 Banco Comercial Português

This data set is composed by the closing price of Banco Comercial Português (BCP) stocks, from January 1989 to December 2018, as represented in Figure 4.1(a). BCP trades on Euronext Lisbon, being the largest contributor to the PSI-20 index. The objective here is to detect outliers using the methods introduced in Chapter 3 for FTS. In order to accomplish that, it is necessary to split the time series and create a FTS, as Figure 4.1(b) shows. This way, the observed Functional Time Series, $\{x_t(u)\}_{t=1}^{30}$, is composed by thirty curves each corresponding to a year of daily measurements (366 points per curve). Additionally, the stationarity tests proposed by Horváth et al. (2014), available in R package `ftsa`, were performed on the time series. With a p-value of 0.23, the series can be considered as stationary for the usual significant levels (1%, 5%, 10%).

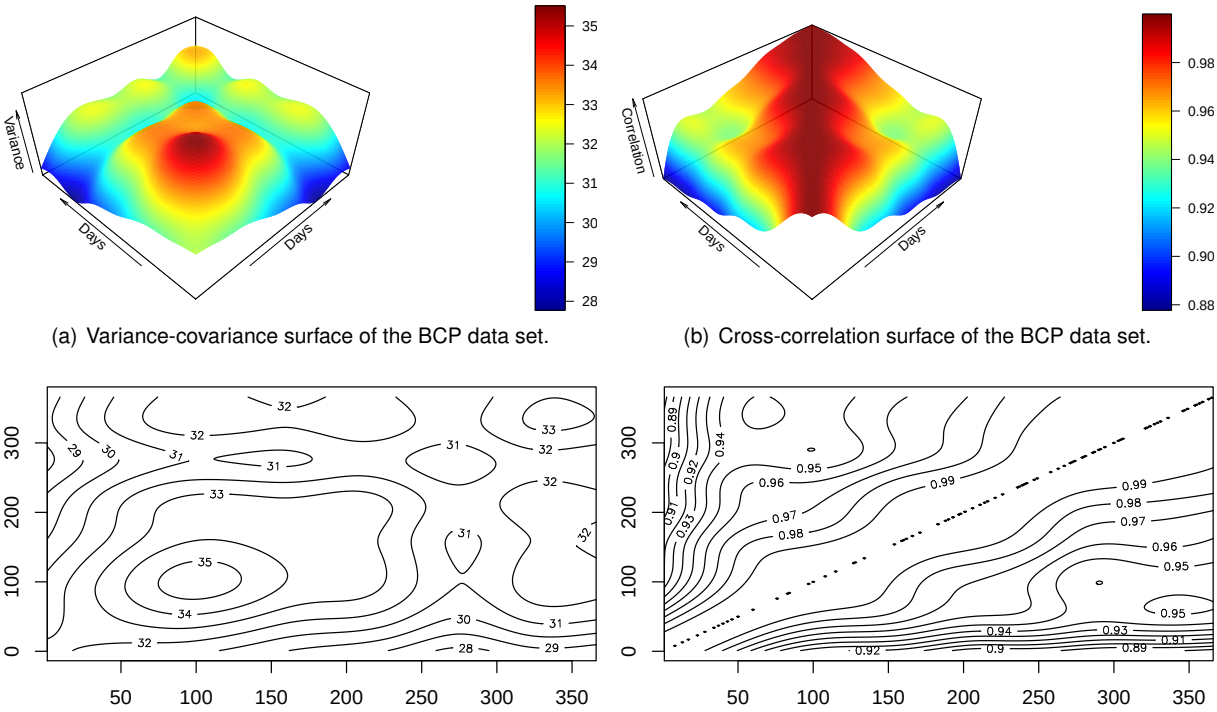


(a) Time series of the closing price of BCP stocks.

(b) Yearly curves of the closing price of BCP stocks.

Figure 4.1: Closing price of BCP stocks.

Figure 4.2 shows the variance-covariance and cross-correlation surfaces of the BCP data set and their contour plots. These were obtained using R package `fda`. It is possible to see that the highest variability occurs around day 100 (around April), while the lowest happens between days 250 and 300 (around October). From the cross-correlation plot it is possible to discern that correlation among closest days is higher, which makes sense considering these data are sequential in time.



(a) Variance-covariance surface of the BCP data set.

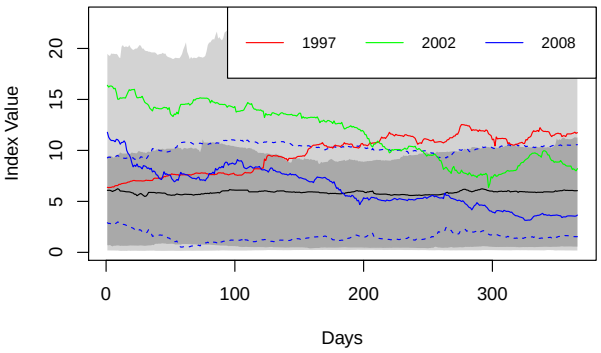
(b) Cross-correlation surface of the BCP data set.

(c) A contour plot of the bivariate variance-covariance surface for the BCP data set.

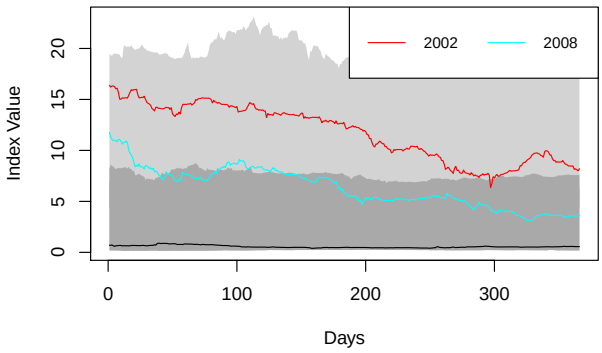
(d) A contour plot of the bivariate cross-correlation surface for the BCP data set.

Figure 4.2: Variance-covariance and cross-correlation surfaces of the BCP data set.

The graphical methods functional bagplot and HDR boxplot (Hyndman and Shang, 2009) were obtained as shown in Figure 4.3 (R package `rainbow`). Moreover, the outliergram and functional boxplot (Tarabelloni, 2017) were also computed and are represented in Figure 4.4 (R package `roahd`). The detected outliers by each of these methods are summarized in Table 4.1. The years 2002 and 2008 were identified as outliers by three of the methods. Economically speaking these years were marked by recession, so it is not unexpected that they are marked as outliers. The year 1997 was detected as an outlier by two of the methods. This year, on the other hand, saw some economic growth and development. It is also worth recalling that the functional boxplot is better suited to detect magnitude outliers. Additionally, the years $1998 - 2001$ identified by this method were precisely years marked by economic growth, so it makes sense that the stock price curves are above the rest of the curves. The outliergram, however, is more apt in identifying shape outliers, being that the years selected correspond to periods of recession.

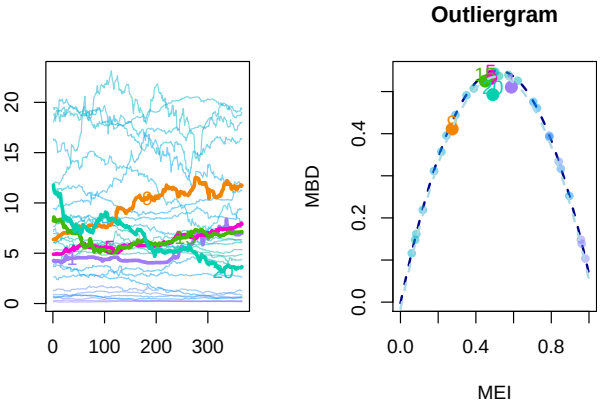


(a) Functional bagplot. The dark and light grey regions correspond to the bag and fence regions, respectively. The black line is the median curve, while the dotted blue lines correspond to 95% pointwise confidence intervals. The colored curves are identified as outliers.

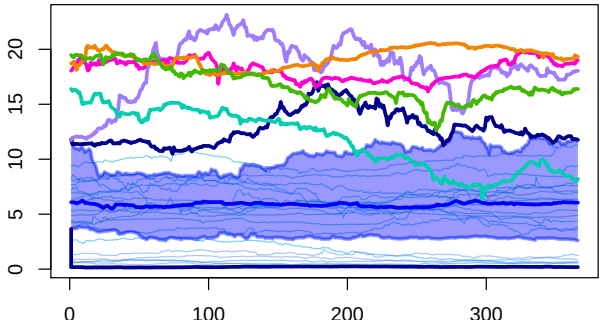


(b) Functional HDR boxplot. The dark and light grey regions represent the 50% HDR and outer HDR, respectively. The black line is the modal curve. The colored curves are outliers.

Figure 4.3: Functional graphical tools for outlier detection obtained with R package `rainbow` for the BCP data set.



(a) Outliergram. On the left: original data set with outliers in colored lines. On the right: outliergram with the outliers' IDs.



(b) Functional boxplot. The fences are represented in dark blue and the outliers in colored lines.

Figure 4.4: Functional graphical tools for outlier detection obtained with R package `roahd` for the BCP data set.

Method	**Detected Outliers**
Functional Bagplot	1997, 2002, 2008
Functional HDR Boxplot	2002, 2008
Outliergram	1989, 1993, 1997, 2003, 2008
Functional Boxplot	1998, 1999, 2000, 2001, 2002

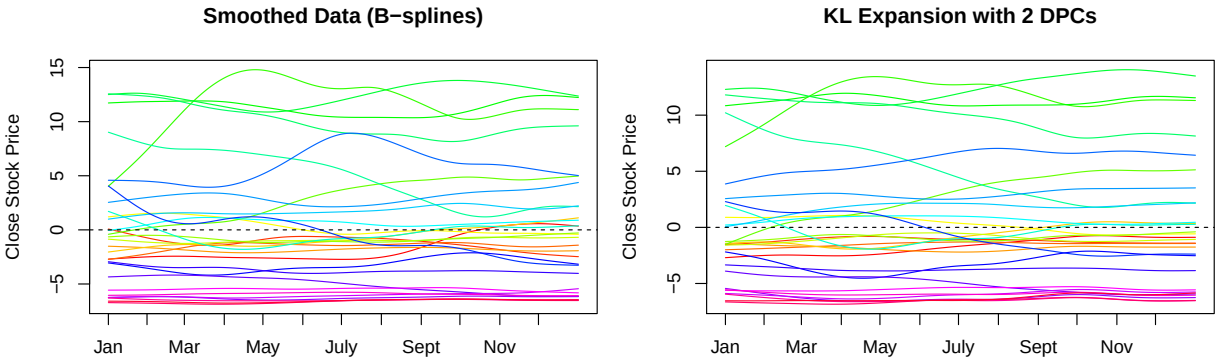
Table 4.1: Outliers detected by the different graphical tools for the BCP data set.

The methods introduced in Section 3.1 based on distances were also applied to the data set, leading to the results in Table 4.2 (R package `rainbow`). The trimming percentage used was $\alpha = 0.1$.

Method	**Detected Outliers**
Likelihood Ratio Test	None
Functional Depth	1998, 1999, 2000, 2002, 2004, 2006, 2007, 2010, 2013
Integrated Square Error	1989, 1990, 1993, 1994, 1997, 1998, 1999, 2001, 2002, 2003, 2006, 2007, 2008, 2011
Robust Mahalanobis Distance	2002

Table 4.2: Outliers detected using methods based on distances for the BCP data set.

Smoothing using a roughness penalty with cubic splines and knots at each day was performed on the data (R package `fda`). The weights were all considered equal to 1 (unweighted). The smoothing parameter mentioned in Section 2.1.3 was chosen by minimizing the GCV, resulting in $\lambda = 10^5$. The resulting smoothed data can be seen in Figure 4.5(a). The versions of static, dynamic and robust FPCA were applied on the smoothed data. In any version of FPCA, the number of principal components, K, was selected so that at least 99% of the variability was explained. The resulting first two static principal components ($K = 3$) are represented in Figure 4.6, which shows the mean function and the consequences of adding and subtracting small amounts of each component. The first principal component, which accounts for 96.9% of the variability, appears to represent a constant vertical shift in the mean.

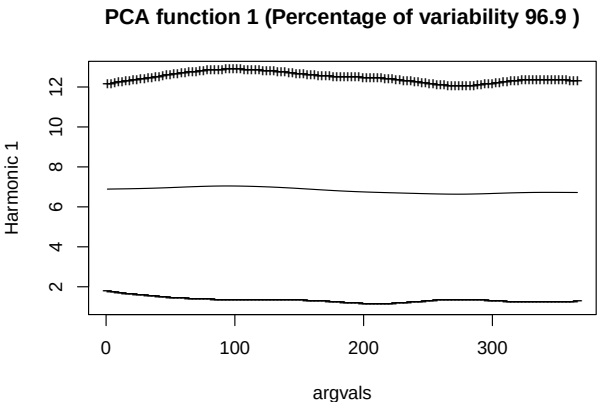


(a) Smoothed data obtained using B-spline basis and R package `fda`.

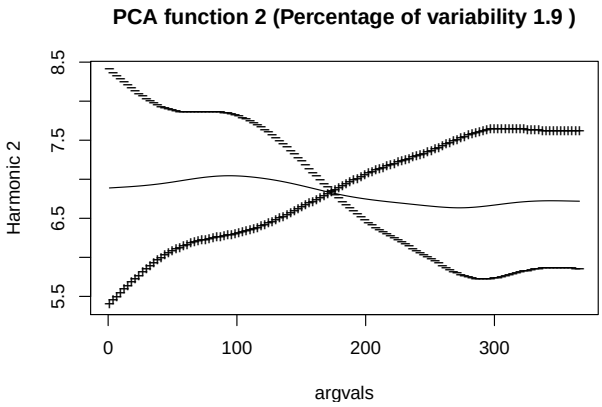
(b) Karhunen-Loève expansion with 2 dynamic principal components obtained with `freqdom.fda`. Smoothed data using an empirical base.

Figure 4.5: Smoothed BCP data obtained with two different basis systems.

Likewise, the dynamic functional principal components resulted on the Karhunen-Loève expansion represented in Figure 4.5(b) with $K = 2$ ($q = \lfloor \sqrt{N} \rfloor = 5$ (Hörmann et al., 2015) was taken as the window size for the kernel estimator). These two dynamic principal components explain 99.15% of the variability in the data. Moreover, the scores obtained with these components are represented in Figure 4.7. These are the scores used to construct the new time series in the algorithm introduced in Section 3.3 and over which are detected the outliers.

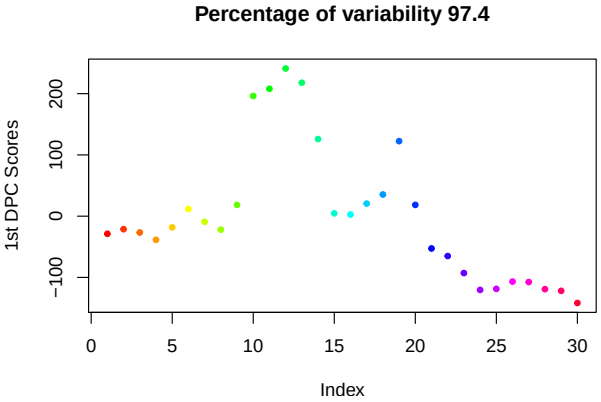


(a) First principal component, represented by adding (+) and subtracting (-) the component from the mean function (line).

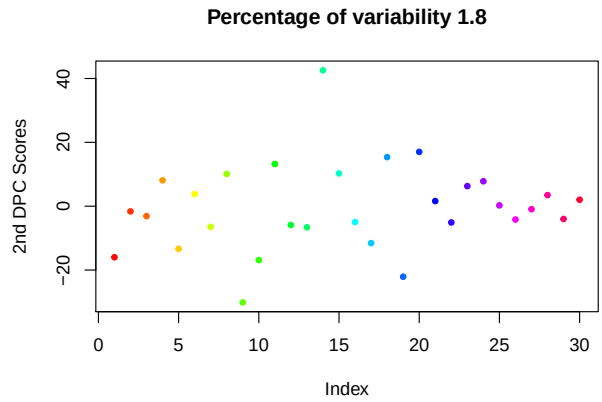


(b) Second principal component, represented by adding (+) and subtracting (-) the component from the mean function (line).

Figure 4.6: Static principal components obtained with R package `fda` for the BCP data set.



(a) Scores obtained from the first dynamic functional principal component.



(b) Scores obtained from the second dynamic functional principal component.

Figure 4.7: Scores obtained from the dynamic functional principal components of the BCP data set.

In the method based on projections introduced in Section 3.3, robust FPCA (Hyndman and Ullah, 2007) was computed, with a tuning parameter $\lambda = 3$ (efficiency of 98.3%). Therefore, the detected outliers using that method are summarized in Table 4.3. As mentioned before, the aim is to compare the results of using static, dynamic and robust FPCA as dimension reduction tools in FTS. In addition, two different methods of time series outlier detection methods were experimented on the principal components scores. These methods are implemented in the R packages `tsoutliers` (de Lacalle, 2019) and `anomalize` (Dancho and Vaughan, 2019). The `tsoutliers` uses Chen and Liu (1993)'s outlier detection method, while `anomalize` implements a method called generalized extreme studentized deviation.

A summary of the results is reported in Figure 4.8, plotting the outlying frequency of each curve using the three types of detection methods (graphical, distance and FPCA). Graphical Tools refers to Table 4.1, Distance Based Methods to Table 4.2, and FPCA Based Methods to Table 4.3. As the barplot suggests, the years $1997 - 2003$ and $2007 - 2008$ are the most frequently detected as outliers. In fact, $1997 - 2000$ is seen as a period of social-economic development, while $2001 - 2003$ and $2007 - 2008$ as periods of recession. Thus, this results are not surprising and somewhat correspond to what was expected based on social-economic events.

Type of FPCA	TS Outlier Method	PC	Detected Outliers
Static FPCA	`tsouliers`	1st	1997, 1998, 2000, 2002, 2003, 2007
		2nd	1997, 2002, 2008
		3rd	1998, 1999
	`anomalize`	1st	None
		2nd	1997, 1999, 2001, 2002, 2011
		3rd	1989, 1998, 1999, 2000, 2001, 2003
Dynamic FPCA	`tsouliers`	1st	1998, 2003
		2nd	1997, 2002
	`anomalize`	1st	1998, 1999, 2000, 2001
		2nd	1997, 2002
Robust FPCA	`tsouliers`	1st	1997, 1998, 2000, 2002, 2003, 2007
	`anomalize`	1st	None

Table 4.3: Outliers detected using the method based on projections for the BCP data set.

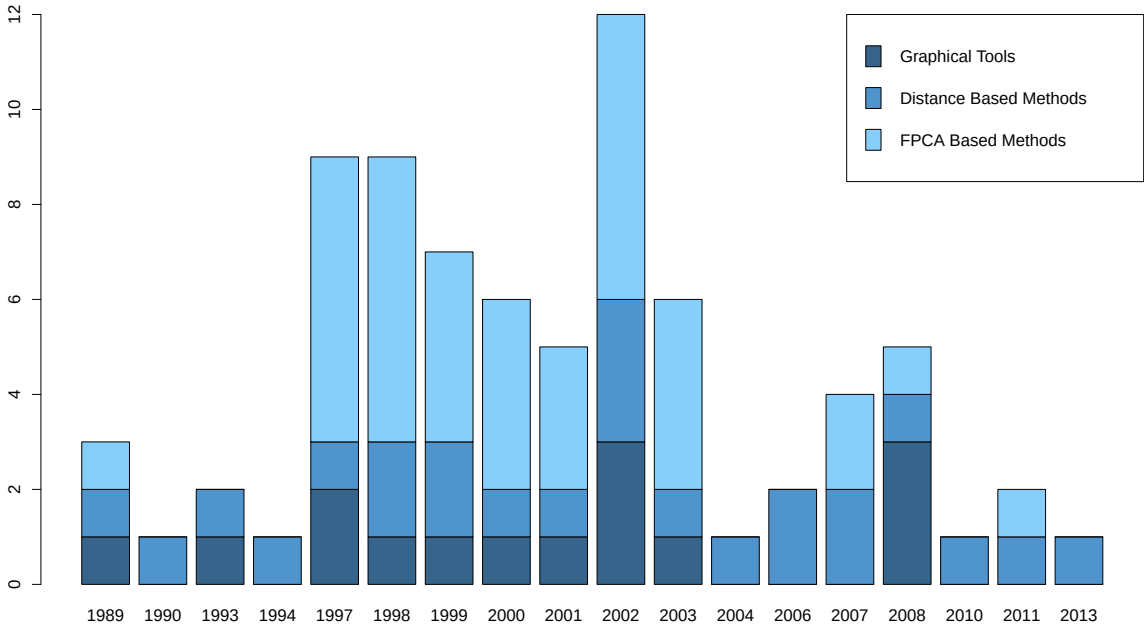
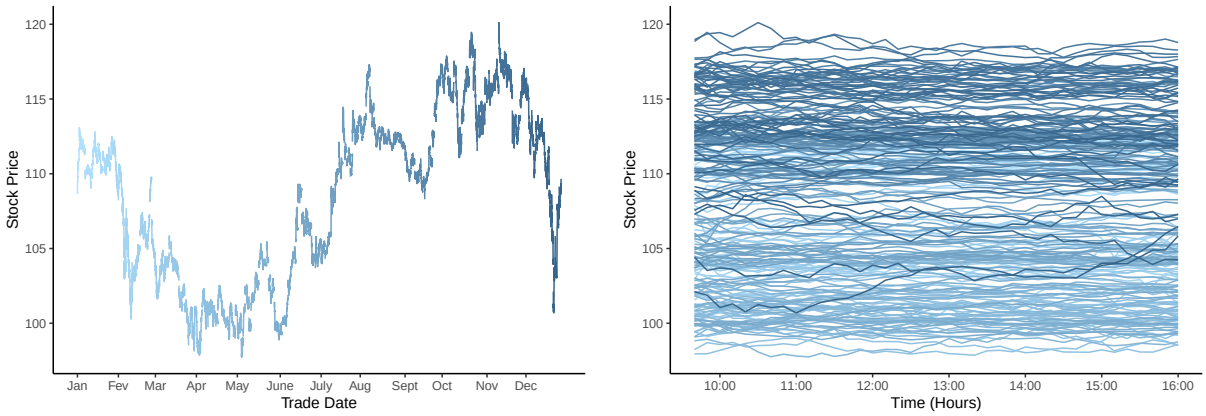


Figure 4.8: Outlying frequency of each curve using the three types of detection methods (graphical, distance and FPCA) for the BCP data set.

4.2 The Walt Disney Company

This second data set is composed by the stock prices of The Walt Disney Company (DIS) over the year 2018, represented in Figure 4.9(a). It comprises 148 curves of intraday stock prices, with measures taken in intervals of 10 minutes from $9:40$h to $16:00$h, as shown in Figure 4.9(b) (39 points per curve). These curves form a Functional Time Series, $\{x_t(u)\}_{t=1}^{148}$. Since these are stock market data, values are only available for work days (which explains the gaps visible in Figure 4.9(a)). Therefore, days related to major holidays (Independence Day, Thanksgiving and Christmas), $07/03$, $11/23$ and $12/24$, only had values for half the day, resulting in their elimination from the dataset. The objective is the same as with the BCP data set, apply the outlier detection methods introduced in Chapter 3 to this FTS.



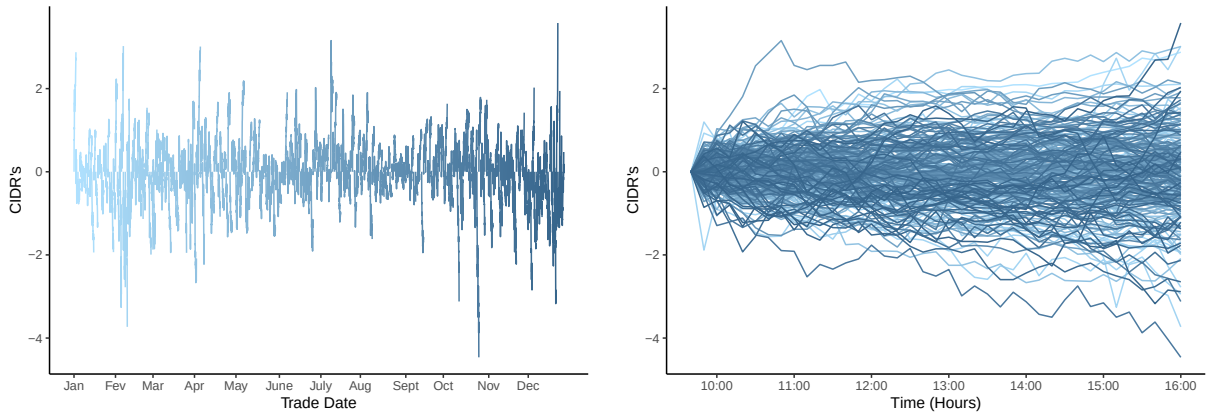
(a) Time series of Disney stock price.

(b) Intraday curves of Disney stock price.

Figure 4.9: DIS stock prices.

For starters, in order to perform DFPCA it is necessary to verify the stationarity condition. The stationarity tests (Horváth et al., 2014) performed on the functional time series resulted in a p-value of 0.02 (R package `ftsa`). Therefore, this time series rejects the null hypothesis of stationarity for the levels of significance 5% and 10%. To deal with this problem, the Cumulative Intraday Returns transformation presented in (2.77) was applied. The CIDR are represented in Figure 4.10. After this transformation, the stationarity tests were repeated which resulted in a p-value of 0.2, confirming that for the usual significant levels this new time series is stationary. Hence, the rest of the study is performed on these transformed data.

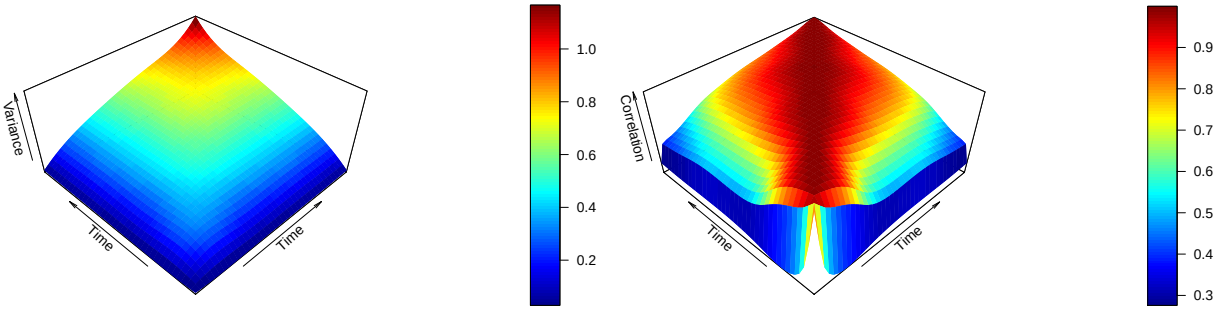
Figure 4.11 shows the variance-covariance and cross-correlation surfaces. The variance-covariance surface reflects the nature of the data after the transformation, considering every curve begins in 0 and then evolves from that. This leads to the very low variance near the starting point and then the steadily increasing values. The cross-correlation surface seems to indicate the temporal dependence between the points, considering the correlation is higher for closer times.



(a) CIDR of the Disney data set.

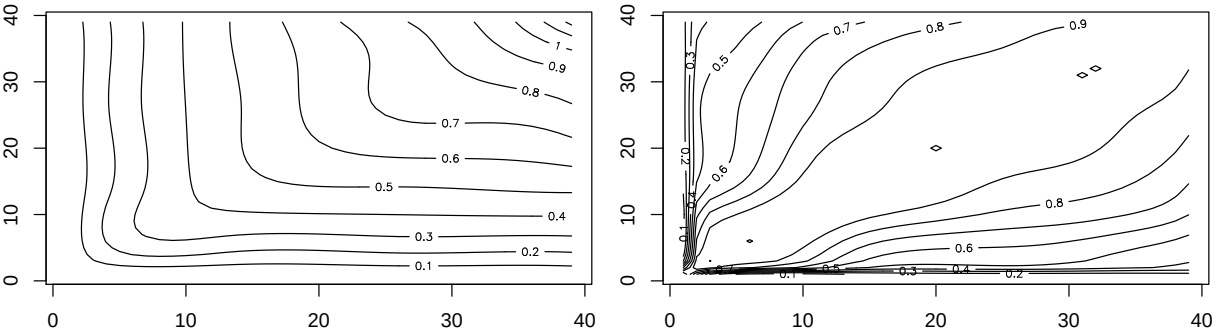
(b) Daily CIDR curves of the Disney data set.

Figure 4.10: CIDR transformation over the DIS data set.



(a) Variance-covariance surface of the Disney data set.

(b) Cross-correlation surface of the Disney data set.

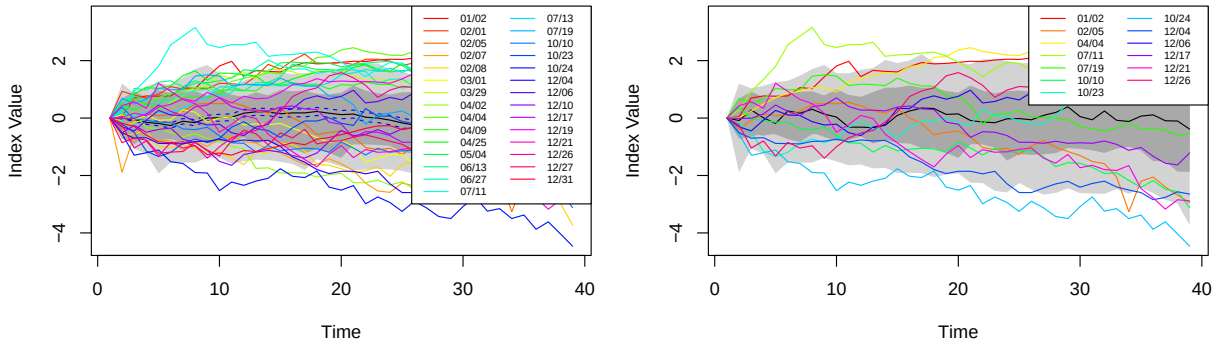


(c) A contour plot of the bivariate variance-covariance surface for the Disney data set.

(d) A contour plot of the bivariate cross-correlation surface for the Disney data set.

Figure 4.11: Variance-covariance and cross-correlation surfaces of the DIS data set.

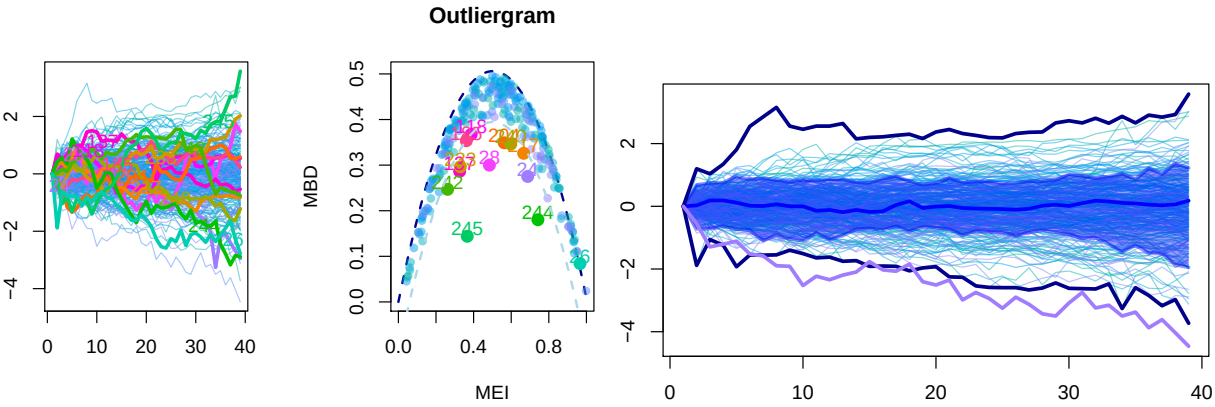
Once again, the graphical tools for outlier detection were employed. Figure 4.12 shows the functional bagplot and HDR boxplot (Hyndman and Shang, 2009), while Figure 4.13 shows the outliergram and functional boxplot (Tarabelloni, 2017). The outliers identified by these methods are gathered in Table 4.4. The functional boxplot only detected one outlier (magnitude outlier), which seems to indicate that the majority are shape outliers. It is possible to see that, for example, many of the identified curves correspond to the month of December, which is most likely due to the effect of Christmas.



(a) Functional bagplot. The dark and light grey regions correspond to the bag and fence regions, respectively. The black line is the median curve, while the dotted blue lines correspond to 95% pointwise confidence intervals. The colored curves are identified as outliers.

(b) Functional HDR boxplot. The dark and light grey regions represent the 50% HDR and outer HDR, respectively. The black line is the modal curve. The colored curves are outliers.

Figure 4.12: Functional graphical tools for outlier detection obtained with `rainbow` for the DIS data set.



(a) Outliergram. On the left: original data set with outliers in colored lines. On the right: outliergram with the outliers' IDs.

(b) Functional boxplot. The fences are represented in dark blue and the outliers in colored lines.

Figure 4.13: Functional graphical tools for outlier detection obtained with `roahd` for the DIS data set.

Method	**Detected Outliers**
Functional Bagplot	01/02, 02/01, 02/05, 02/07, 02/08, 03/01, 03/29, 04/02, 04/04, 04/09 04/25, 05/04, 06/13, 06/27, 07/11, 07/13, 07/19, 10/10, 10/23, 10/24 12/04, 12/06, 12/10, 12/17, 12/19, 12/21, 12/26, 12/27, 12/31
Functional HDR Boxplot	01/02, 02/05, 04/04, 07/11, 07/19, 10/10, 10/23, 10/24, 12/04, 12/06 12/17, 12/21, 12/26
Outliergram	02/05, 02/09, 06/20, 07/19, 09/26, 10/23, 11/09, 12/06, 12/17, 12/19 12/21, 12/26
Functional Boxplot	10/24

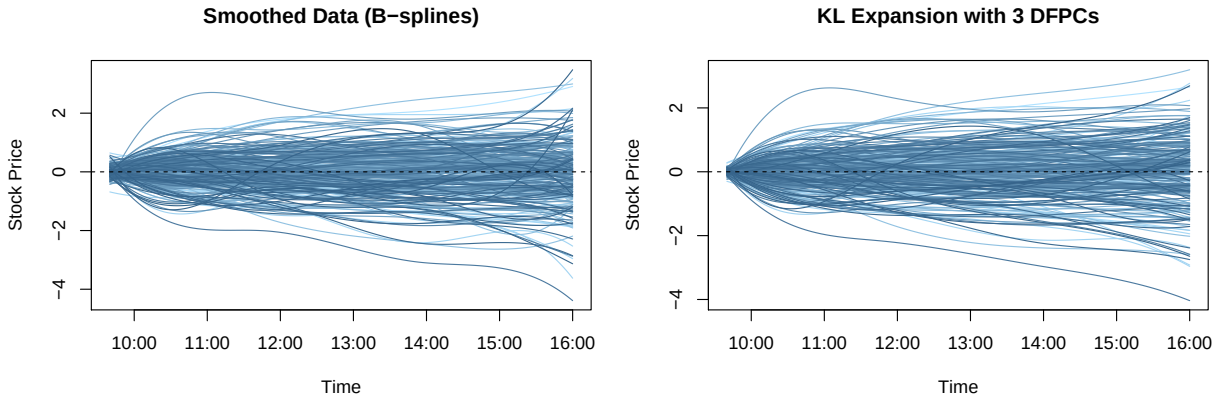
Table 4.4: Outliers detected by the different graphical tools for the DIS data set.

Furthermore, the methods based on distances described in Hyndman and Shang (2009) were applied to this data set, resulting in the outliers in Table 4.5 (R package `rainbow`). The trimming percentage used was $\alpha = 0.1$.

Method	Detected Outliers
Likelihood Ratio Test	None
Functional Depth	None
Integrated Square Error	02/05, 02/06, 02/07, 02/08, 02/09, 03/27, 04/03, 04/06, 06/13, 06/20 06/27, 07/11, 10/24, 10/30, 11/20, 12/19, 12/21, 12/26, 12/27, 12/28
Robust Mahalanobis Distance	None

Table 4.5: Outliers detected using methods based on distances for the DIS data set.

Smoothing using cubic splines and knots spaced by ten minutes was performed on the data (R package `fda`). The weights were also all considered equal to 1 (unweighted). In this case, it was opted to use $\lambda = 0$, meaning there was no roughness penalization. The resulting smoothed data is represented in Figure 4.14(a). Functional principal components were then computed over these data, resulting in the first two components represented in Figure 4.15, which shows the mean function and the consequences of adding and subtracting small amounts of each component. This first component accounts for 87.4% of the variability and seems to represent the general shape of the data. The number of principal components, K, was chosen so that at least 98% of the variability is explained. In this case, four static principal components were retained.

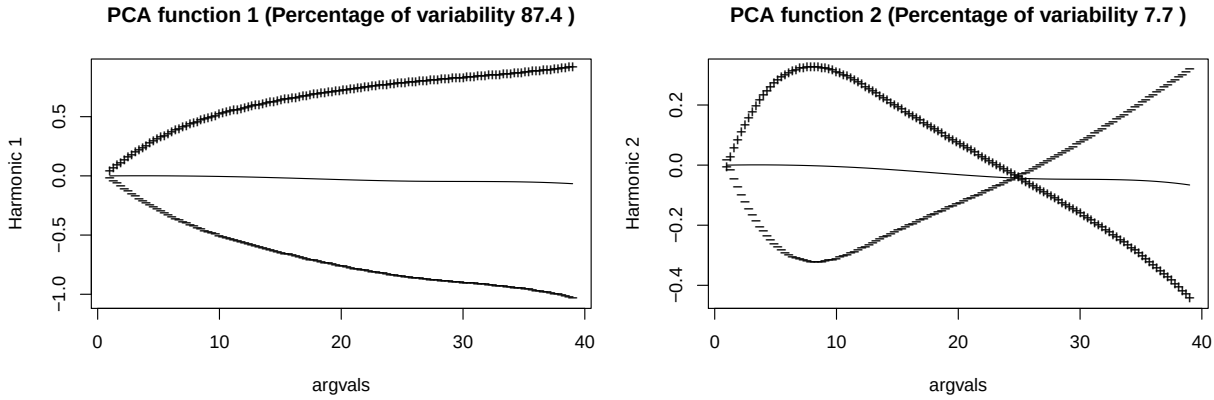


(a) Smoothed data obtained using B-spline basis and R package `fda`.

(b) Karhunen-Loève expansion with 3 dynamic principal components obtained with `freqdom.fda`. Smoothed data using an empirical base.

Figure 4.14: The DIS data smoothed through two different basis systems.

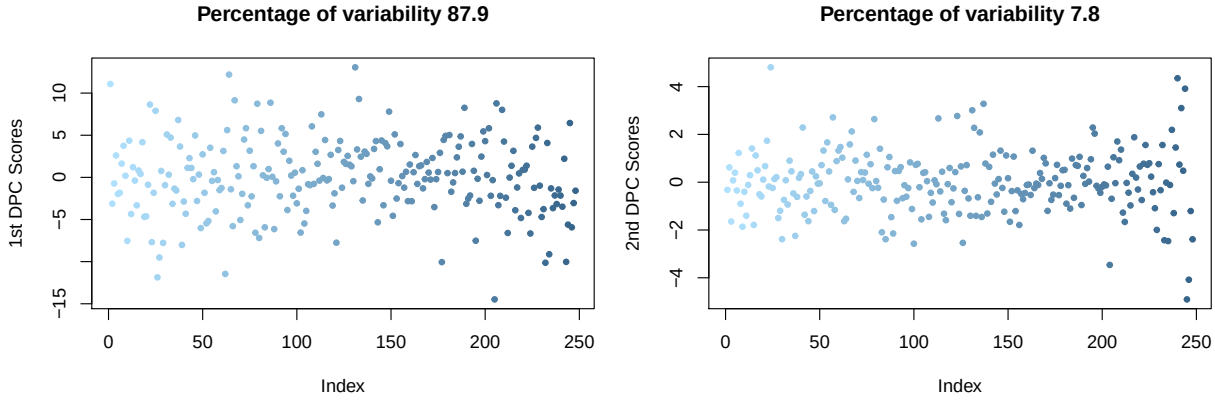
Moreover, the dynamic functional principal components were computed over the CIDR curves. The number of components selected was three, explaining 98.25% of the variability. In addition, $q = \lfloor \sqrt{N} \rfloor = 15$ (Hörmann et al., 2015) was taken as the window size for the kernel estimator. Figure 4.14(b) shows the intraday curves after smoothing with the dynamic functional principal components as basis instead of the cubic splines. The resulting scores of the first two dynamic principal components are represented in Figure 4.16. In the application of the outliers detection method based on projections (Section 3.3), the outliers are detected over a new time series composed by these scores.



(a) First principal component, represented by adding (+) and subtracting (-) the component from the mean function (line).

(b) Second principal component, represented by adding (+) and subtracting (-) the component from the mean function (line).

Figure 4.15: Static principal components obtained with R package `fda` for the DIS data set.



(a) Scores obtained from the first dynamic functional principal component.

(b) Scores obtained from the second dynamic functional principal component.

Figure 4.16: Scores obtained from the dynamic functional principal components of the DIS data set.

In order to apply the algorithm presented in Section 3.3, robust FPCA (Hyndman and Ullah, 2007) was also performed on the data, with a tuning parameter of $\lambda = 3$ (efficiency of 98.3%). The algorithm was then employed with static, dynamic and robust functional components. Two time series detection methods required by the algorithm were used, one provided by package `tsoutliers` (Chen and Liu (1993)'s method) and the other by package `anomalize` ("gesd" method). The results provided by these choices are summarized in Table 4.6.

A summary of the results is reported in Figure 4.17, plotting the outlying frequency of each curve using the three types of detection methods (graphical, distance and FPCA). In order to make the representation easier, it were only considered the outliers that were detected more than twice. Moreover, Graphical Tools refers to Table 4.4, Distance Based Methods to Table 4.5, and FPCA Based Methods to Table 4.6. Some of the curves most frequently detected as outliers correspond to days belonging to the month of December. As mentioned above, this is most likely due to the Christmas season. Moreover, some of this curves correspond to the month of February, which might be the outcome of the release of a very profitable movie. Consequently, to some extent these results correspond to what one might have been expecting from this data set.

Type of FPCA	TS Outlier Method	PC	Detected Outliers
Static FPCA	`tsouliers`	1st	10/24
		2nd	02/05, 07/19, 12/26
		3rd	02/09, 03/27, 12/19
		4th	12/27
	`anomalize`	1st	10/24
		2nd	02/05, 07/11, 07/19, 12/21, 12/26
		3rd	02/09, 03/27, 12/19
		4th	02/07, 03/01, 06/20, 12/21, 12/27
Dynamic FPCA	`tsouliers`	1st	07/11
		2nd	02/05, 12/26
		3rd	02/09
	`anomalize`	1st	None
		2nd	02/05, 12/17, 12/21, 12/26, 12/27
		3rd	02/09
Robust FPCA	`tsouliers`	1st	10/24
		2nd	02/05, 07/19, 12/26
		3rd	02/09, 03/27, 12/19
		4th	None
		5th	02/06, 10/30, 12/26
		6th	02/08, 04/06, 12/26, 12/28
		7th	02/07
	`anomalize`	1st	10/24
		2nd	02/05, 07/19, 12/26
		3rd	02/09, 03/27, 12/19
		4th	12/21, 12/27
		5th	02/06, 02/21, 03/23, 05/02, 10/30, 12/26, 12/27
		6th	04/06, 12/27, 12/28
		7th	02/06, 02/07

Table 4.6: Outliers detected using the method based on projections for the DIS data set.

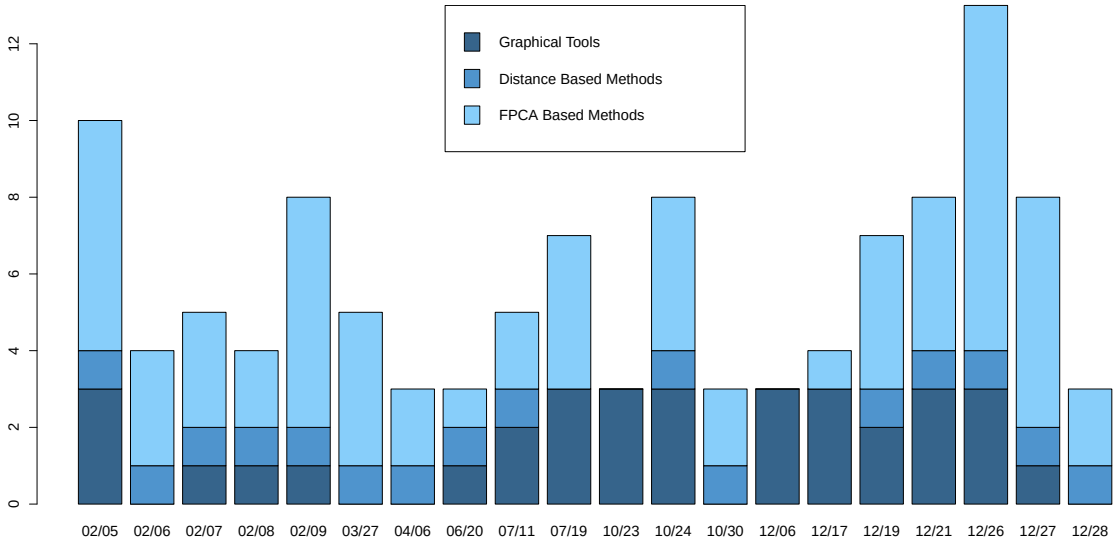


Figure 4.17: Outlying frequency of each curve using the three types of detection methods (graphical, distance and FPCA) for the DIS data set (only outliers detected more than twice).

Chapter 5

Conclusions

This last chapter summarizes the developments of this dissertation. The main conclusions from the outliers detection methods applied to two real data sets, from a functional time series perspective, are stated. Finally, the open issues that have not been tackled in the thesis, as well as possible future research lines, are discussed.

5.1 Summary and Conclusions

In this work, two financial data sets were analysed from a functional perspective. In fact, studying time series from a functional perspective is still not very common. Most existing literature treats time series as scalars or vectors, and functional data as i.i.d. observations. Thus, Functional Time Series represents mostly uncharted territory. However, this approach allowed to detect anomalous curves with several outlying procedures. In this work, the dynamic FPCA was employed in an attempt to deal with the temporal dependencies inherent to FTS, which is a novelty in the literature.

Starting with the Banco Comercial Português data set, modelling the closing stock prices as a functional time series allowed to search for curves whose behaviour differed from the rest. In this analysis, as each curve corresponded to a year, the aim was to search for years marked by recession or economic development. This pursuit begun with outliers detection methods based on graphical tools, on distances, and finally using a method based on projections. Also, taking into consideration the serial dependence structure was fundamental in order to avoid misleading conclusions. Consequently, when using dynamic FPCA two components were enough to explain 99% of the variability in the data, while with static FPCA three components were necessary. Therefore, DFPCA does seem to be better suited in representing this type of data. In addition, as reported in Table 4.3, there were less detected outliers over the dynamic functional principal components. This might be the result of these components accounting for the serial dependencies in the data, that otherwise would have been seen as outlying curves.

The second data set consisted of intraday stock price curves of The Walt Disney Company. Once again, the major advantage of using the framework of functional data is analysing information present in the shapes of these curves. Nonetheless, since this Functional Time Series is not stationary, it

was necessary to perform a transformation. The CIDR transformation can, however, make it harder to interpret some results, since it leads to a FTS where each curve always starts from zero. In order to identify the outlying curves in the data set, methods based on graphical tools, distances and projections were used. Furthermore, taking into consideration the serial dependence in the data is once again a key aspect of the analysis. The dynamic FPCA was used to deal with this problem. Similarly to the first data set, in order to explain at least 98% it were needed three dynamic principal components and four static principal components. So dynamic FPCA seems once more to be better at representing the data. Moreover, the tendency to detect less outliers with dynamic FPCA appears to repeat itself in this data set.

As for the methods based on distances they seem less realistic in terms of results. Considering these methods were the first attempt at outliers detection in functional data, this is not surprising. The methods based on graphical tools and on projections seem to have much more plausible results. This methods use principal component analysis in order to simplify the data representation and only then attempt at selecting outlying observations. Hence, performing FPCA seems to provide a clear advantage in the detection of outliers. What is more, in both data sets, the results obtained appear to correspond fairly well to what was expected based on external social-economic events.

5.2 Future Work

Regarding future work suggestions, there are some lines of research that were left unexplored in the scope of this work. The most pressing suggestion is performing a simulation study to confront the outlying efficiency of the different methods. Then it would also be easy to compare the simulation output with the results obtained for the BCP and The Walt Disney Company data sets. For example, check if the tendency for detecting less outliers with dynamic FPCA in the method based on projections holds.

Another suggestion is using the robust functional principal components estimators proposed by Bali et al. (2011) as another type of FPCA in Step 1 of the algorithm based on projections presented in Section 3.3. Some other time series outliers detection methods could also be experimented in Step 2 of this algorithm. Perhaps a robust time series outlying detector might be more suitable to the task at hand.

An additional topic would be to apply some changes to the methods based on graphical tools, in particular, to both the functional bagplot and the functional HDR (Hyndman and Shang, 2009). Considering that both of these methods first obtain the robust principal components scores, it would be interesting to obtain new plots based on the dynamic functional principal components scores instead.

Bibliography

Arribas-Gil, A. and Romo, J. (2014). Shape outlier detection and visualization for functional data: the outliergram. *Biostatistics*, 15(4):603–619.

Bali, J. L., Boente, G., Tyler, D. E., and Wang, J.-L. (2011). Robust functional principal components: a projection-pursuit approach. *The Annals of Statistics*, 39(6):2852–2882.

Bande, M. F., de la Fuente, M. O., Galeano, P., Nieto, A., and Garcia-Portugues, E. (2019). *fda.usc: Functional Data Analysis and Utilities for Statistical Computing*. `https://CRAN.R-project.org/package=fda.usc`. R package version 1.5.0.

Barra, V. (2004). Analysis of gene expression data using functional principal components. *Computer Methods and Programs in Biomedicine*, 75(1):1–9.

Benko, M., Härdle, W., and Kneip, A. (2009). Common functional principal components. *The Annals of Statistics*, 37(1):1–34.

Boente, G. and Barrera, M. S. (2015). S-estimators for functional principal component analysis. *Journal of the American Statistical Association*, 110(511):1100–1111.

Brillinger, D. R. (1981). *Time Series Data Analysis and Theory*. Holden Day, San Francisco.

Chen, C. and Liu, L.-M. (1993). Joint estimation of model parameters and outlier effects in time series. *Journal of the American Statistical Association*, 88(421):284–297.

Chong, C. O. C., García, J. E. S., and DelaCerda, J. (2015). Análisis de componentes principales funcionales en series de tiempo económicas. *Revista Internacional de Gestión del Conocimiento y la Tecnología*, 3(2):13–25.

Dancho, M. and Vaughan, D. (2019). *anomalize: Tidy Anomaly Detection*. `https://CRAN.R-project.org/package=anomalize`. R package version 0.2.0.

Dauxois, J., Pousse, A., and Romain, Y. (1982). Asymptotic theory for the principal component analysis of a vector random function: Some applications to statistical inference. *Journal of Multivariate Analysis*, 12:136–154.

de Lacalle, J. L. (2019). *tsoutliers: Detection of Outliers in Time Series*. `https://CRAN.R-project.org/package=tsoutliers`. R package version 0.6-8.

Erbas, B., Hyndman, R. J., and Gertig, D. M. (2007). Forecasting age-specific breast cancer mortality using functional data models. *Statistics in Medicine*, 26(2):458–470.

Febrero, M., Galeano, P., and González-Manteiga, W. (2007). A functional analysis of nox levels: location and scale estimation and outlier detection. *Computational Statistics*, 22(3):411–427.

Febrero, M., Galeano, P., and González-Manteiga, W. (2008). Outlier detection in functional data by depth measures, with application to identify abnormal nox levels. *Environmetrics*, 19(4):331–345.

Gabrys, R., Horváth, L., and Kokoszka, P. (2010). Tests for error correlation in the functional linear model. *Journal of the American Statistical Association*, 105(491):1113–1125.

Gao, Y., Shang, H. L., and Yang, Y. (2018). High-dimensional functional time series forecasting: An application to age-specific mortality rates. *Journal of Multivariate Analysis*, 170:232–243.

Gervini, D. (2008). Robust functional estimation using the median and spherical principal components. *Biometrika*, 95(3):587–600.

Hasenstab, K., Scheffler, A., Telesca, D., Sugar, C. A., Jeste, S., DiStefano, C., and Şentürk, D. (2017). A multi-dimensional functional principal components analysis of eeg data. *Biometrics*, 73:999–1009.

Hormann, S. and Kidziński, L. (2017). *freqdom.fda: Functional Time Series: Dynamic Functional Principal Components*. `https://CRAN.R-project.org/package=freqdom.fda`. R package version 0.9.1.

Horváth, L. and Kokoszka, P. (2012). *Inference for Functional Data with Applications*. Springer.

Horváth, L., Kokoszka, P., and Rice, G. (2014). Testing stationarity of functional time series. *Journal of Econometrics*, 179(1):66–82.

Hörmann, S., Kidziński, L., and Hallin, M. (2015). Dynamic functional principal components. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 77(2):319–348.

Hubert, M., Rousseeuw, P. J., and Verboven, S. (2002). A fast method for robust principal components with applications to chemometrics. *Chemometrics and Intelligent Laboratory Systems*, 60(1-2):101–111.

Hyndman, R. and Shang, H. L. (2019). *ftsa: Functional Time Series Analysis*. `https://CRAN.R-project.org/package=ftsa`. R package version 5.5.

Hyndman, R. and Ullah, M. (2007). Robust forecasting of mortality and fertility rates: a functional data approach. *Computational Statistics and Data Analysis*, 51(10):4942–4956.

Hyndman, R. J. and Shang, H. L. (2009). Rainbow plots, bagplots and boxplots for functional data. *Journal of Computational and Graphical Statistics*, 19(1):29–45.

Karhunen, K. (1946). Zur spektraltheorie stochastischer prozesse. *Annales Academiae Scientiarum Fennicae*, 37:1–37.

Kidziński, L., Jouzdani, N., and Kokoszka, P. (2017). *pcdpca: Dynamic Principal Components for Periodically Correlated Functional Time Series*. `https://CRAN.R-project.org/package=pcdpca`. R package version 0.4.

Kidziński, L., Kokoszka, P., and Jouzdani, N. M. (2016). Principal component analysis of periodically correlated functional time series. *Journal of Time Series Analysis*, 39(4).

Kokoszka, P. and Reimherr, M. (2017). *Introduction to Functional Data Analysis*. Taylor & Francis.

Li, Y., Huang, C., and Härdle, W. K. (2017). Spatial functional principal component analysis with applications to brain image data. *SFB 649 Discussion Paper 2017-024*.

Li, Y., Wang, N., and Carroll, R. J. (2013). Selecting the number of principal components in functional data. *Journal of the American Statistical Association*, 108(504).

Locantore, N., Marron, J. S., Simpson, D. G., Tripoli, N., Zhang, J. T., and Cohen, K. L. (1999). Robust principal component analysis for functional data. *Test*, 8:1–73.

Loève, M. (1946). Fonctions aléatoires a décomposition orthogonale exponentielle. *La Revue Scientifique*, 84:159–162.

Meiring, W. (2007). Oscillations and time trends in stratospheric ozone levels: A functional data analysis approach. *Journal of the American Statistical Association*, 102(479):788–802.

Peña, D. and Yohai, V. J. (2016). Generalized dynamic principal components. *Journal of the American Statistical Association*, 111(515):1121–1131.

Ramsay, J. O. (1982). When the data are functions. *Psychometrika*, 47(4):379–396.

Ramsay, J. O. (2013). Functional data analysis. McGill University, Canada. Viewed 18 April 2019,`http://www.psych.mcgill.ca/misc/fda`.

Ramsay, J. O. and Dalzell, C. J. (1991). Some tools for functional data analysis. *Journal of the Royal Statistical Society*, 53(3):539–572.

Ramsay, J. O., Hooker, G., and Graves, S. (2009). *Functional Data Analysis with R and MATLAB*. Use R! Springer.

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

Ramsay, J. O. and Silverman, B. W. (2005). *Functional Data Analysis*. Springer Series in Statistics. Springer, second edition.

Ramsay, J. O., Wickham, H., Graves, S., and Hooker, G. (2018). *fda: Functional Data Analysis*. `https://CRAN.R-project.org/package=fda`. R package version 2.4.8.

Rao, C. R. (1958). Some statistical methods for comparison of growth curves. *Biometrics*, 14(1):1–17.

Rice, J. A. and Silverman, B. W. (1991). Estimating the mean and covariance structure nonparametrically when the data are curves. *Journal of the Royal Statistical Society: Series B*, 53(1):233–243.

Rousseeuw, P. J. and Leroy, A. M. (1987). *Robust Regression and Outlier Detection*. Wiley Series in Probability and Statistics. John Wiley and Sons, New York.

Sawant, P., Billor, N., and Shin, H. (2012). Functional outlier detection with robust functional principal component analysis. *Computational Statistics*, 27(1):83–102.

Shang, H. L. (2011). A survey of functional principal component analysis. Department of Econometrics & Business Statistics, Monash University, Australia. Working Paper.

Shang, H. L. and Hyndman, R. (2019). *rainbow: Bagplots, Boxplots and Rainbow Plots for Functional Data*. `https://CRAN.R-project.org/package=rainbow`. R package version 3.6.

Shorack, G. and Wellner, J. (1986). *Empirical Processes with Applications to Statistics*. Wiley.

Tarabelloni, N. (2017). *Robust Statistical Methods in Functional Data Analysis*. PhD thesis, Politecnico di Milano.

Tarabelloni, N., Arribas-Gil, A., Ieva, F., Paganoni, A. M., Romo, J., and Palma, F. (2018). *roahd: Robust Analysis of High Dimensional Data*. `https://CRAN.R-project.org/package=roahd`. R package version 1.4.1.

Vilar, J. M., Raña, P., and Aneiros, G. (2016). Using robust fpca to identify outliers in functional time series, with applications to the electricity market. *SORT: Statistics and Operations Research Transactions*, 40(2):321–348.

Wang, J.-L., Chiou, J.-M., and Müller, H.-G. (2016). Review of functional data analysis. *Annual Review of Statistics and Its Application*, 3:257–295.

Wang, S., Jank, W., and Shmueli, G. (2008). Explaining and forecasting online auction prices and their dynamics using functional data analysis. *Journal of Business & Economic Statistics*, 26(2):144–160.

Wang, Z., Sun, Y., and Li, P. (2014). Functional principal components analysis of shanghai stock exchange 50 index. *Discrete Dynamics in Nature and Society*, 2014. Article ID 365204.

Wrobel, J. and Goldsmith, J. (2016). *refund.shiny: Interactive Plotting for Functional Data Analyses*. `https://CRAN.R-project.org/package=refund.shiny`. R package version 0.3.0.

Appendix A

R code

A.1 Banco Comercial Português

```
library(ggplot2)
library(reshape2)
library(fda)
library(ftsa)
library(rainbow)
library(roahd)
library(plot3D)
library(freqdom.fda)
library(tsoutliers)
library(anomalize)
library(tibble)

#dadosTS is a data.frame with 366 rows and 31 columns without missing values.
#The first column corresponds to the breakpoints (366 days).
#The remaining columns correspond each to a different curve (30 years).

meses=c("Jan", "Fev", "Mar", "Apr", "May", "June", "July", "Aug", "Sept",
        "Oct", "Nov", "Dec")

##Plot FTS
dadosTS[,1]=as.numeric(dadosTS[,1])
new=melt(dadosTS, id.vars = 'Date', variable.name = 'Index')
ggplot(new, aes(Date, value)) + geom_line(aes(colour = Index)) +
        scale_color_manual(values = rainbow(30)) +
        scale_x_continuous(name="Trade Date (Days)",
                breaks=c(1,31,60,91,121,152,182,213,244,274,305,335),
                labels=meses) +
```

```
        scale_y_continuous(name="Closing Price") + theme_classic()

##Plot TS
new[,4]=1:10980
colnames(new)[4]='ID'
ggplot(new, aes(ID, value)) + geom_line(aes(colour = Index)) +
        scale_color_manual(values = rainbow(30), guide=F) +
        scale_x_continuous(name="Trade Date",
                breaks=c(1,1099,2197,3295,4393,5491,6589,7687,8785,9883,10980),
                labels=c(1989,1992,1995,1998,2001,2004,2007,2010,2013,2016,2019))
        +  scale_y_continuous(name="Closing Price") + theme_classic()

##Plot example fts
new1=new[4027:7687,]
ggplot(new1, aes(ID, value)) + geom_line(aes(colour = Index)) +
        scale_color_manual(values = rainbow(11), guide=F) +
        scale_x_continuous(name="Trade Date",
                breaks=c(4027,4393,4759,5125,5491,5857,6223,6589,6955,7321,7687),
                labels=c(2000,2001,2002,2003,2004,2005,2006,2007,2008,2009,2010),
                minor_breaks = NULL)
         + scale_y_continuous(name="Closing Price", minor_breaks = NULL)
         + theme_light()
         + theme(panel.grid.major.y=element_blank(),
                panel.grid.major.x = element_line(colour = "black",
                linetype = "dashed"))

##Stationarity
T_stationary(dadosTS[,-1], pivotal=T) #stationary p-value=0.22

##OUTLIERS graphic tools
smoothedopen=fts(x=dadosTS[,1], y=dadosTS[,-1], xname = "Days",
        yname = "Index Value") #
fboxplot(smoothedopen, plot.type = "functional", type = "bag",
        projmethod = "PCAproj", ncol=3, cex=0.8) #1997 2002 2008
fboxplot(smoothedopen, plot.type = "functional", type = "hdr",
        projmethod="PCAproj", cex=0.8) #2002 2008

outliergram(fData(dadosTS[,1], t(dadosTS[,-1]))) #shape: 1989 1993 1997 2003 2008
par(mfrow=c(1,1))
fbplot(fData(dadosTS[,1], t(dadosTS[,-1]))) #amplitude: 1998 1999 2000 2001 2002

##OUTLIERS based on distances
```

```
foutliers(smoothedopen, method = "robMah") #2002
foutliers(smoothedopen, method = "depth.trim") #10 14 19 11 16 22 12 18 25
foutliers(smoothedopen, method = "depth.pond") #10 14 19 11 16 22 12 18 25
foutliers(smoothedopen, method = "HUoutliers") #"1989" "1990" "1993" "1994" "1997"
        #"1998" "1999" "2001" "2002" "2003" "2006" "2007" "2008" "2011"
foutliers(smoothedopen, method = "lrt")

##FDA: smoothing
basis=create.bspline.irregular(dadosTS[,1])

#  set up range of smoothing parameters in log_10 units
loglam <- 2:10
nlam   <- length(loglam)

dfsave  <- rep(0,nlam)
gcvsave <- rep(0,nlam)

#  loop through smoothing parameters
for (ilam in 1:nlam) {
        lambda     <- 10^loglam[ilam]
        cat(paste("lambda =",lambda,"\n"))
        fdParobj   <- fdPar(basis, lambda=lambda)
        smoothlist <- smooth.basis(dadosTS[,1], as.matrix(dadosTS[,-1]), fdParobj)
        fdobj      <- smoothlist[[1]]
        df         <- smoothlist[[2]]
        gcv        <- smoothlist[[3]]
        dfsave[ilam]  <- df
        gcvsave[ilam] <- sum(gcv)}

cbind(loglam, dfsave, gcvsave)

par(mfrow=c(1,2), pty="m")
plot(loglam, gcvsave, type="b", cex=1,
xlab="Log_10 lambda", ylab="GCV Criterion",
main="Index Value Smoothing")

plot(loglam, dfsave, type="b",  cex=1,
xlab="Log_10 lambda", ylab="Degrees of freedom",
main="Index Value Smoothing")

#  minimum GCV estimate
lambda   <- 10^5
```

```
fdParobj <- fdPar(basis, lambda=lambda)
smoothlist <- smooth.basis(dadosTS[,1], as.matrix(dadosTS[,-1]), fdParobj)
returnsfd  <- smoothlist$fd

##var-cov & cor
var=var.fd(returnsfd) #variance-covariance surface
var_eval=eval.bifd(dadosTS[,1], dadosTS[,1], var)
persp3D(x=as.numeric(dadosTS[,1]),y=as.numeric(dadosTS[,1]),z=var_eval,
        r=3, expand = 0.5, theta = -45, #phi=25,
        xlab='Days', ylab='Days', zlab='Variance')
contour(dadosTS[,1], dadosTS[,1], var_eval)

corr=cor.fd(dadosTS[,1],returnsfd) #cross-correlation surface
persp3D(x=as.numeric(dadosTS[,1]),y=as.numeric(dadosTS[,1]),z=corr,
        r=3, expand = 0.5, theta = -45, #phi=25,
        xlab='Days', ylab='Days', zlab='Correlation')
contour(dadosTS[,1], dadosTS[,1], corr)

##FPCA
pcalist=pca.fd(returnsfd, nharm = 3) #default subtracts the mean
cumsum(pcalist$varprop)
pcascores=pcalist$scores
par(mfrow=c(1,1))
plot(pcalist)
plot(pcascores[,1], pch=20, col=rainbow(30), ylab='1st PC Scores')
plot(pcascores[,2], pch=20, col=rainbow(30), ylab='2nd PC Scores')
plot(pcascores[,3], pch=20, col=rainbow(30), ylab='3rd PC Scores')

##DFPCA
dpca=fts.dpca(center.fd(returnsfd), q=5, Ndpc = 2) #q=floor(sqrt(n))
cumsum(fts.dpca.var(dpca$spec.density))
scores=dpca$scores
rownames(scores)=colnames(dadosTS)[-1]
plot(scores[,1], pch=20, col=rainbow(30), ylab='1st DPC Scores',
        main='Percentage of variability 97.4')
plot(scores[,2], pch=20, col=rainbow(30), ylab='2nd DPC Scores',
        main='Percentage of variability 1.8')

##Plot smoothed data
plot(center.fd(returnsfd), main='Smoothed Data (B-splines)', xaxt='n',
        xlab='Trade Date', ylab='Close Stock Price', col=rainbow(30), lty=1)
plot(dpca$Xhat, main='KL Expansion with 2 DPCs', xaxt='n', xlab='Trade Date',
```

```
         ylab='Close Stock Price', col=rainbow(30), lty=1)
axis(side = 1, at=c(1,31,60,91,121,152,182,213,244,274,305,335), labels = meses)

##RFPCA
rfpca=ftsm(smoothedopen, order=1, method = "M")
cumsum(rfpca$varprop)
coeffs=rfpca$coeff[,-1]
plot(as.vector(coeffs[,1]) , pch=20, col=rainbow(30), ylab='1st RPC Scores')
plot(as.vector(coeffs[,2]), pch=20, col=rainbow(30), ylab='2nd RPC Scores')

##OUTLIERS based on projections
##tsoutliers
#static FPCA
tso(ts(pcascores[,1], start = 1989, end = 2018))$outliers$time
        #1997 1998 2000 2002 2003 2007
tso(ts(pcascores[,2], start = 1989, end = 2018))$outliers$time #1997 2002 2008
tso(ts(pcascores[,3], start = 1989, end = 2018))$outliers$time #1998 1999

#dynamic FPCA
tso(ts(scores[,1], start = 1989, end = 2018))$outliers$time #1998 2003
tso(ts(scores[,2], start = 1989, end = 2018))$outliers$time #1997 2002

#robust FPCA
tso(ts(coeffs, start = 1989, end = 2018))$outliers$time
        #1997 1998 2000 2002 2003 2007

##anomalize
#static FPCA
anomalize(as_tibble(pcascores), target = 1, method = 'gesd',
        verbose = T)$anomaly_details$outlier_idx #
anomalize(as_tibble(pcascores), target = 2, method = 'gesd',
        verbose = T)$anomaly_details$outlier_idx #1999,1997,2001,2002,2011
anomalize(as_tibble(pcascores), target = 3, method = 'gesd',
        verbose = T)$anomaly_details$outlier_idx #1989,1998,1999,2000,2001,2003

#dynamic FPCA
anomalize(as_tibble(scores), target = 1, method = 'gesd',
        verbose = T)$anomaly_details$outlier_idx #1998-2001
anomalize(as_tibble(scores), target = 2, method = 'gesd',
        verbose = T)$anomaly_details$outlier_idx #1997,2002

#robust FPCA
```

```
anomalize(as_tibble(coeffs), target = 1, method = 'gesd',
        verbose = T)$anomaly_details$outlier_idx #

##outliers barplot
outliers=c("1989", "1990", "1993", "1994", "1997", "1998", "1999", "2000", "2001",
        "2002", "2003", "2004", "2006", "2007", "2008", "2010", "2011", "2013")
freq=cbind(c(1,1,1),c(0,1,0),c(1,1,0),c(0,1,0),c(2,1,6),c(1,2,6),c(1,2,4),
        c(1,1,4),c(1,1,3), c(3,3,6),c(1,1,4),c(0,1,0),c(0,2,0),c(0,2,2),c(3,1,1),
        c(0,1,0),c(0,1,1),c(0,1,0))
colnames(freq)=outliers
rownames(freq)=c("Graphical Tools", "Distance Based Methods", "FPCA Based Methods")
barplot(freq, col=c("steelblue4","steelblue3","lightskyblue"),
        cex.names=1.2, cex.axis = 1.2)
legend("topright", legend = rownames(freq),
        fil=c("steelblue4","steelblue3","lightskyblue"), cex=1.2)
```

A.2 The Walt Disney Company

```
library(ggplot2)
library(reshape2)
library(fda)
library(ftsa)
library(rainbow)
library(roahd)
library(plot3D)
library(freqdom.fda)
library(tsoutliers)
library(anomalize)
library(tibble)

#dadosTS is a data.frame with 39 rows and 249 columns without missing values.
#The first column corresponds to the breakpoints (39 intraday measures).
#The remaining columns correspond each to a different curve (248 days).

##Plot FTS
dadosTS[,1]=1:39
colnames(dadosTS)[1]='Timestamp'
new=melt(dadosTS, id.vars = 'Timestamp', variable.name = 'Date')
fc <- colorRampPalette(c("lightskyblue1", "steelblue4"))
ggplot(new, aes(Timestamp, value)) + geom_line(aes(colour = Date))
        + scale_color_manual(values = fc(248), guide=F)
        + scale_x_continuous(name="Time (Hours)", breaks=c(3,9,15,21,27,33,39),
                labels=c("10:00","11:00","12:00","13:00","14:00","15:00","16:00"))
        + scale_y_continuous(name="Stock Price") + theme_classic()

##Plot TS
new[,4]=1:9672
colnames(new)[4]='ID'
meses=c("Jan", "Fev", "Mar", "Apr", "May", "June", "July", "Aug", "Sept",
        "Oct", "Nov", "Dec")
ggplot(new, aes(ID, value)) + geom_line(aes(colour = Date))
        + scale_color_manual(values = fc(248), guide=F)
        + scale_x_continuous(name="Trade Date",
                breaks=c(1,820,1561,2380,3199,4057,4876,5656,6553,7294,8191,8971),
                labels=meses) + scale_y_continuous(name="Stock Price")
        + theme_classic()

##Stationarity
```

```
T_stationary(dadosTS[,-1], pivotal = T) #p-value=0.02

#CIDR transformation
dadosCIDR=dadosTS
for (n in 2:249){
        for (j in 1:39){
                dadosCIDR[j,n]=100*(log(dadosTS[j,n])-log(dadosTS[1,n]))
        }
}

T_stationary(dadosCIDR[,-1], pivotal = T) #stationary p-value=0.2

##Plot CIDR
new1=melt(dadosCIDR, id.vars = 'Timestamp', variable.name = 'Date')
new1[,4]=1:9672
colnames(new1)[4]='ID'
ggplot(new1, aes(Timestamp, value)) + geom_line(aes(colour = Date))
        + scale_color_manual(values = fc(248), guide=F)
        + scale_x_continuous(name="Time (Hours)", breaks=c(3,9,15,21,27,33,39),
                labels=c("10:00","11:00","12:00","13:00","14:00","15:00","16:00"))
        + scale_y_continuous(name="CIDR's") + theme_classic()

ggplot(new1, aes(ID, value)) + geom_line(aes(colour = Date))
        + scale_color_manual(values = fc(248), guide=F)
        + scale_x_continuous(name="Trade Date",
                breaks=c(1,820,1561,2380,3199,4057,4876,5656,6553,7294,8191,8971),
                labels=meses) + scale_y_continuous(name="CIDR's") + theme_classic()

##OUTLIERS graphic tools
smoothedopen=fts(x=dadosCIDR[,1], y=dadosCIDR[,-1],
        xname = "Time", yname = "Index Value")
fboxplot(smoothedopen, plot.type = "functional", type = "bag",
        projmethod = "PCAproj", cex=0.6) #CIDR: 01/02 02/01 02/05 02/07
        #02/08 03/01 03/29 04/02 04/04 04/09 04/25 05/04 06/13 06/27
        #07/11 07/13 07/19 10/10 10/23 10/24 12/04 12/06 12/10 12/17
        #12/19 12/21 12/26 12/27 12/31
fboxplot(smoothedopen, plot.type = "functional", type = "hdr",
        projmethod="PCAproj", cex=0.6) #CIDR: 01/02 02/05 04/04 07/11
        #07/19 10/10 10/23 10/24 12/04 12/06 12/17 12/21 12/26

outliergram(fData(dadosCIDR[,1], t(dadosCIDR[,-1])))
#shape: 02/05 02/09 06/20 07/19 09/26 10/23 11/09 12/06 12/17 12/19 12/21 12/26
```

```
par(mfrow=c(1,1))
fbplot(fData(dadosCIDR[,1], t(dadosCIDR[,-1]))) #amplitude: 10/24

##OUTLIERS distance based
foutliers(smoothedopen, method = "robMah") #
foutliers(smoothedopen, method = "lrt") #
foutliers(smoothedopen, method = "depth.trim") #
foutliers(smoothedopen, method = "depth.pond") #
foutliers(smoothedopen, method = "HUoutliers") #CIDR: "02/05" "02/06" "02/07"
        #"02/08" "02/09" "03/27" "04/03" "04/06" "06/13" "06/20" "06/27" "07/11"
        #"10/24" "10/30" "11/20" "12/19" "12/21" "12/26" "12/27" "12/28"

##FDA: smoothing
basis=create.bspline.irregular(dadosTS[,1])

#  set up range of smoothing parameters in log_10 units
loglam <- 2:10
nlam   <- length(loglam)

dfsave  <- rep(0,nlam)
gcvsave <- rep(0,nlam)

#  loop through smoothing parameters
for (ilam in 1:nlam) {
        lambda     <- 10^loglam[ilam]
        cat(paste("lambda =",lambda,"\n"))
        fdParobj   <- fdPar(basis, lambda=lambda)
        smoothlist <- smooth.basis(dadosCIDR[,1], as.matrix(dadosCIDR[,-1]), fdParobj)
        fdobj      <- smoothlist[[1]]
        df         <- smoothlist[[2]]
        gcv        <- smoothlist[[3]]
        dfsave[ilam]  <- df
        gcvsave[ilam] <- sum(gcv)}

cbind(loglam, dfsave, gcvsave)

par(mfrow=c(1,2), pty="m")
plot(loglam, gcvsave, type="b", cex=1,
        xlab="Log_10 lambda", ylab="GCV Criterion",
        main="Index Value Smoothing")

plot(loglam, dfsave, type="b",  cex=1,
```

```
        xlab="Log_10 lambda", ylab="Degrees of freedom",
        main="Index Value Smoothing")

#  minimum GCV estimate
lambda   <- 10^3
fdParobj <- fdPar(basis, lambda=0)

smoothlist3 <- smooth.basis(dadosCIDR[,1], as.matrix(dadosCIDR[,-1]), fdParobj)
returnsfd3 <- smoothlist3$fd

##var-cov & cor
var=var.fd(returnsfd3) #variance-covariance surface
var_eval=eval.bifd(dadosCIDR[,1], dadosCIDR[,1], var)
persp3D(x=as.numeric(dadosCIDR[,1]),y=as.numeric(dadosCIDR[,1]),z=var_eval,
        r=3, expand = 0.5, theta = -45, #phi=25,
        xlab='Time', ylab='Time', zlab='Variance')
contour(dadosCIDR[,1], dadosCIDR[,1], var_eval)

corr=cor.fd(dadosCIDR[,1],returnsfd3) #cross-correlation surface
persp3D(x=as.numeric(dadosCIDR[,1]),y=as.numeric(dadosCIDR[,1]),z=corr,
        r=3, expand = 0.5, theta = -45, #phi=25,
        xlab='Time', ylab='Time', zlab='Correlation')
contour(dadosCIDR[,1], dadosCIDR[,1], corr)

##FPCA
pcalist=pca.fd(returnsfd3, nharm = 4)
cumsum(pcalist$varprop)
plot(pcalist)
plot(pcalist$scores[,1], pch=20, col=fc(248), ylab='1st PC Scores')
plot(pcalist$scores[,2], pch=20, col=fc(248), ylab='2nd PC Scores')
pcascores=pcalist$scores

##DFPCA
dpca=fts.dpca(center.fd(returnsfd3), q=15, Ndpc = 3) #q=floor(sqrt(n))
cumsum(fts.dpca.var(dpca$spec.density))
scores=dpca$scores
rownames(scores)=colnames(dadosTS)[-1]
plot(dpca$scores[,1], pch=20, col=fc(248), ylab='1st DPC Scores',
        main='Percentage of variability 87.9')
plot(dpca$scores[,2], pch=20, col=fc(248), ylab='2nd DPC Scores',
        main='Percentage of variability 7.8')
```

```
#Plot smoothed data
plot(returnsfd3, main='Smoothed Data (B-splines)', xaxt='n', xlab='Time',
        ylab='Stock Price', col=fc(248), lty=1)
plot(dpca$Xhat, main='KL Expansion with 3 DFPCs', xaxt='n', xlab='Time',
        ylab='Stock Price', col=fc(248), lty=1)
axis(side = 1, at=c(3,9,15,21,27,33,39),
        labels=c("10:00","11:00","12:00","13:00","14:00","15:00","16:00"))

##RFPCA
colnames(dadosCIDR)=1:249
smoothedopen=fts(x=dadosCIDR[,1], y=dadosCIDR[,-1],
        xname = "Time", yname = "Index Value")
rfpca=ftsm(smoothedopen, order=7, method = "M")
cumsum(rfpca$varprop)
coeffs=rfpca$coeff[,-1]
plot(as.vector(coeffs[,1]) , pch=20, col=fc(248), ylab='1st RPC Scores')
plot(as.vector(coeffs[,2]), pch=20, col=fc(248), ylab='2nd RPC Scores')

##OUTLIERS based on projections
##tsoutliers
#static FPCA
tso(ts(pcascores[,1], start = 1, end = 248))$outliers$time #CIDR: "10/24"
tso(ts(pcascores[,2], start = 1, end = 248))$outliers$time
        #CIDR:"02/05" "07/19" "12/26"
tso(ts(pcascores[,3], start = 1, end = 248))$outliers$time
        #CIDR:"02/09" "03/27" "12/19"
tso(ts(pcascores[,4], start = 1, end = 248))$outliers$time #CIDR:"12/27"

#dynamic FPCA
tso(ts(scores[,1], start = 1, end = 248))$outliers$time #CIDR:"07/11"
tso(ts(scores[,2], start = 1, end = 248))$outliers$time #CIDR:"02/05" "12/26"
tso(ts(scores[,3], start = 1, end = 248))$outliers$time #CIDR:"02/09"

#robust FPCA
for (i in 1:7){print(tso(ts(coeffs[,i], start = 1, end = 248))$outliers$time)}
        #"10/24"
        #"02/05" "07/19" "12/26"
        #"02/09" "03/27" "12/19"
        #NULL
        #"02/06" "10/30" "12/26"
        #"02/08" "04/06" "12/26" "12/28"
        #"02/07"
```

```
##anomalize
#static FPCA
for (i in 1:4){print(anomalize(as_tibble(pcascores), target = i,
        method = 'gesd', verbose = T)$anomaly_details$outlier_idx)}
        #"10/24"
        #"12/26" "02/05" "07/19" "12/21" "07/11"
        #"02/09" "12/19" "03/27"
        #"12/27" "12/21" "06/20" "02/07" "03/01"

#dynamic FPCA
for (i in 1:3){print(anomalize(as_tibble(scores), target = i,
        method = 'gesd', verbose = T)$anomaly_details$outlier_idx)}
        #None
        #"02/05" "12/26" "12/17" "12/27" "12/21"
        #"02/09"

#robut FPCA
for (i in 1:7){print(anomalize(as_tibble(coeffs), target = i,
        method = 'gesd', verbose = T)$anomaly_details$outlier_idx)}
        #"10/24"
        #"12/26" "02/05" "07/19"
        #"02/09" "12/19" "03/27"
        #"12/27" "12/21"
        #"12/27" "12/26" "02/06" "10/30" "03/23" "05/02" "02/21"
        #"12/28" "12/27" "04/06"
        #"02/06" "02/07"

##Barplot outliers
outliers=c("02/05", "02/06", "02/07", "02/08", "02/09", "03/27", "04/06",
        "06/20", "07/11", "07/19", "10/23", "10/24", "10/30", "12/06", "12/17",
        "12/19", "12/21", "12/26", "12/27", "12/28")
freq=cbind(c(3,1,6),c(0,1,3),c(1,1,3),c(1,1,2),c(1,1,6),c(0,1,4),c(0,1,2),
        c(1,1,1),c(2,1,2),c(3,0,4),c(3,0,0),c(3,1,4),c(0,1,2),c(3,0,0),c(3,0,1),
        c(2,1,4),c(3,1,4),c(3,1,9),c(1,1,6),c(0,1,2))
colnames(freq)=outliers
rownames(freq)=c("Graphical Tools", "Distance Based Methods", "FPCA Based Methods")
barplot(freq, col=c("steelblue4","steelblue3","lightskyblue"),
        cex.names=1.2, cex.axis = 1.2)
legend("top", legend = rownames(freq),
        fil=c("steelblue4","steelblue3","lightskyblue"), cex=1.2)
```