Analysis of Spatially Correlated Functional Data Objects

A thesis submitted to the University of Glasgow in fulfilment of the requirements for the award of the degree of Doctor of Philosophy in the College of Science and Engineering

Salihah Safar Alghamdi

School of Mathematics and Statistics

University of Glasgow

April 2019

Abstract

Space-time data are of great interest in many fields of research, but they are inherently complex in nature which leads to practical issues when formulating statistical models to analyse them. In classical analysis of space-time data the temporal variation is modelled using traditional time-series analysis. This thesis focuses on building a comprehensive framework for analysing space-time data, where the temporal component is considered to be a continuous function and modelled using functional data analytic tools. There are several approaches for analysis spatially correlated functional data, but most of them are designed for specific applications and there is no easy way of comparing these methods. In summary, the challenge in modelling space-time data using functional data analytic techniques is that there is no clear rule regarding which method is most appropriate for analysing a new dataset. Existing methods have been developed for specific applications without giving a clear indication for a practitioner regarding their appropriateness. This motivates us to propose a clear flow chart of the analysis of space-time data using functional data analysis methods and develop a framework under which different existing methods can be compared.

In this research, we provide a clear comparison between two widely different methods of modelling spatial dependence one using parametric and the other using non-parametric spatial dependence. These techniques were developed for datasets with different complexities. First, we had to generalize the methodologies and codes of both of these methods to analyse data with features they were not originally designed for. We then compared the performance of these two methods on two real life datasets, the enhanced vegetation index (EVI) data and the electroencephalography (EEG) data. Further we have generalized our framework to accommodate

replicated data and used it to build classification tools that outperforms all existing approaches.

One major contribution of this thesis is the development of the methodological framework and computational tool for the analysis of spatially correlated functional data. We have also clearly demonstrated, theoretically, and through simulations that our approach outperforms exitsing methods. Finally, for the EEG data we have demonstrated that classification tools built on representations from our models can outperform classification tools using the raw data.

Declaration

I declare that, all the work presents in this thesis has been done by myself under the supervision of Dr. Surajit Ray, except where otherwise stated. This thesis represent work completed, between 2015 and 2019 in Statistics in the School of Mathematics and Statistics at the University of Glasgow. None of the work described has been submitted to other university or other institute.

Acknowledgements

First of all, I would like to express my gratitude to my supervisor Dr. Surajit Ray, for his guidance over the duration of my PhD. I am extremely grateful for his encouragement and support, Without him this work would never have been completed. I would also like to acknowledge the Saudi Arabian Cultural Bureau (SACB), for sponsoring my PhD.

A big "thank you" goes to all of my family and friends for their support and encouragement. I wish to express my appreciation to my honourable parents, loving brother and sisters for their constant love and support. Last and most importantly, I wish to thank my husband, Farraj who stands beside me and encourages me constantly for every step of my PhD studies. My thanks are also to my kids, Ryan and Rand for giving me happiness, joy and enduring love.

Contents

List of Figures

List of Tables

Abbreviations

FDA Functional Data Analysis

SPACE Spatial Principal Analysis of Conditional Expectation

ST-PDE Spatio-temporal Regression Model with Partial Differential Equations Regularization

RST-PDE Replicated Spatio-temporal Regression Model with Partial Differential Equations Regularization

EVI Enhanced Vegetation Index

EEG Electroencephalography

PACE Principal Analysis of Conditional Expectation

FPCA Functional Principal Component Analysis

GCV Generalised Cross Validation

FEA Finite Element Analysis

RMSE Root Mean Square Errors

SVM Support Vector Mechanism

RF Random Forest

KNN K-Nearest Neighbors

Chapter 1

Introduction

1.1 Introduction and Thesis Statement

Space-time data are one of the most common types of datasets owing the fact many applications involve data observed over time and space. Space-time data occur in many real world applications such as Meteorology, Biology, Medicine and Ecology. These datasets typically include both spatial and temporal features. Often, these data sets can be very large for example, data that consist of daily temperature measurements of many different sites of a region over many years. The observations are usually highly correlated either in time or space or both; for example, neighbouring observations tend to have similar values similarly, on the time domain one observation can depend on the previous one. In order to model this type of data, it is important to consider the spatial dependence as well as the temporal dependence.

Traditionally, space-time data are modelled using spatio-temporal methods where the temporal axis is treated as discrete time points. As an example of spatio-temporal methods, Sadeghi et al. (2010) analysed brain development data set where the data consist of temporal and spatial aspects. The temporal part was modelled using Gompertz function while the spatial part was modelled using three different spatial localisation strategies. Another spatio-temporal approach was proposed by Smith et al. (2003) to analyse $PM_{2.5}$ data (particulate matter of aerodynamic diameter 2.5 μm or less) from three locations. The variation was decomposed into four

parts, the first part was time effect represented by non-parametric approaches that model the mean of each week separately and time trend (weekly trend) for the year. The second part was the spatial effect which was estimated using thin plate splines. The last two parts are land use component modelled as dummy variable and random errors which are spatially correlated. In these two approaches the temporal effect and spatial effect were estimated separately.

However, the spatial and temporal effects can be modelled in a functional form in the framework of functional data analysis where each observation is a function in time over space. Before describing the full modelling framework we start with a gentle introduction to functional data analysis.

1.2 Functional Data Analysis

Functional data analysis (FDA) is a field of statistics that models functional observations observed over some continuum. The functions contain repeated measurements of the same process and can be viewed as smooth curves. Functional data analysis (FDA) has many applications in different areas such as medicine, public health, biological sciences and environmental science. Ramsay and Silverman introduced functional data analysis, providing many statistical techniques for analysing functional data (Silverman and Ramsay, 2005) and they also provide practical applications of FDA through several case studies (Ramsay and Silverman, 2002). Furthermore, Ramsay et al. (2014) provide the R package `"fda"`, to implement functional data analysis methods.

Functional data are usually defined on one dimension usually time, however, it can be extended to multi-dimensional spaces such as space-time data, image-time data and can be observed on manifolds or other complex domains. "The basic philosophy of functional data analysis is to think of observed data functions as single entities, rather than merely as a sequence of individual observations" (Silverman and Ramsay, 2005). This important feature can simplify the analysis of the complex structure of the data and allow one to use derivatives or other properties of curves

to analyse data. In the next chapter we provide a summary of data representation and exploratory analysis techniques using FDA.

According to Silverman and Ramsay (2005) the primary aims of functional data analysis (FDA) is to; display the data in a way that helps to present the important features, explore the variation among the data, and compare between different sets of the data.

FDA and multivariate analysis have many standard techniques in common. One distinction is that FDA can successfully analyse infinite dimensional data. Griswold et al. (2008), who studied the differences between multivariate and functional methods, suggested that functional approaches provide better estimate than multivariate methods. One important advantage of FDA is continuity between data points, which provides further information of the variation in the data. In particular, FDA is more accurate for change point detection. Horvath and Kokoszka (2012), pointed out that functional methods can detect more change points than the multivariate methods. Another advantage is that FDA doesn't assume equally spaced time points and can present the time interval as a smooth continuous function. A key feature of FDA is that it does not assume independent measurement error and can efficiently accommodate these measurements. We will provide a more detailed discussion of FDA in chapter 2. In the next section we introduce spatially correlated functional data which is our main focus in this thesis.

1.3 Spatially Correlated Functional Data

Many recent researches has focused on modelling spatially correlated functional data, which consist of curves observed in different locations of a region or over a manifold where the neighbouring observations behave similarly. Modelling the spatial dependence in functional data can be an important step in analysing these datasets.

There are several challenges for modelling spatially correlated functional data. The primary challenge is estimating the covariance function which is high dimen-

sional and computationally intensive. Most of the studies assume separable covariance to simplify the estimation. In separable covariance structure the covariance is written as Kronecker product of the spatial and temporal covariance. However, using a separable modelling approach might not always be appropriate. Another approach for introducing a simpler covariance structure is using a diagonal covariance matrix instead of a full matrix.

On the other hand, if the spatial dimension is two dimensional, one might need to model spatial correlation along several directions. For example, the isotropic assumption assumes the same covariance for all directions and provides explicit estimation of the covariance function.

There is an extensive literature on modelling spatially correlated functional data. We review some existing techniques, which we implemented in this thesis, in chapter 3.

1.4 Research Problem

To analyse any dataset, it is important to know the best technique that can describe and model the data. We have already seen that data observed over both space and time can be modelled either as spatiotemporal data (when the domain time is discrete) or functional data (when time is continuous). Currently, there is no clear pathway for a practitioner when analysing spatio-temporal data. First, the researcher needs to choose whether the dataset is in a discrete or continuous time domain. Then, based on the researcher's choice and knowledge of the existing methods, a parametric or non-parametric approach is used. Furthermore, when modelling the dependence in space-time data most of the current methods assume separability of the covariance operator. This simplifies the covariance estimation as the temporal and spatial covariances are estimated separately but this assumption might not always be correct. Some researchers apply fully parametric or non-parametric methods but in general these methods are used based on subjective choices. Similarly with respect to isotropy, for simplicity most of the method assume isotropy while many physical phenomenon are strongly anisotropic i.e. the covariance changes based on

the direction.

We start by providing a diagram in figure 1.1 that illustrates the available methods for analysing spatio-temporal data that a researcher can use to determine the appropriate approach for the analysis.

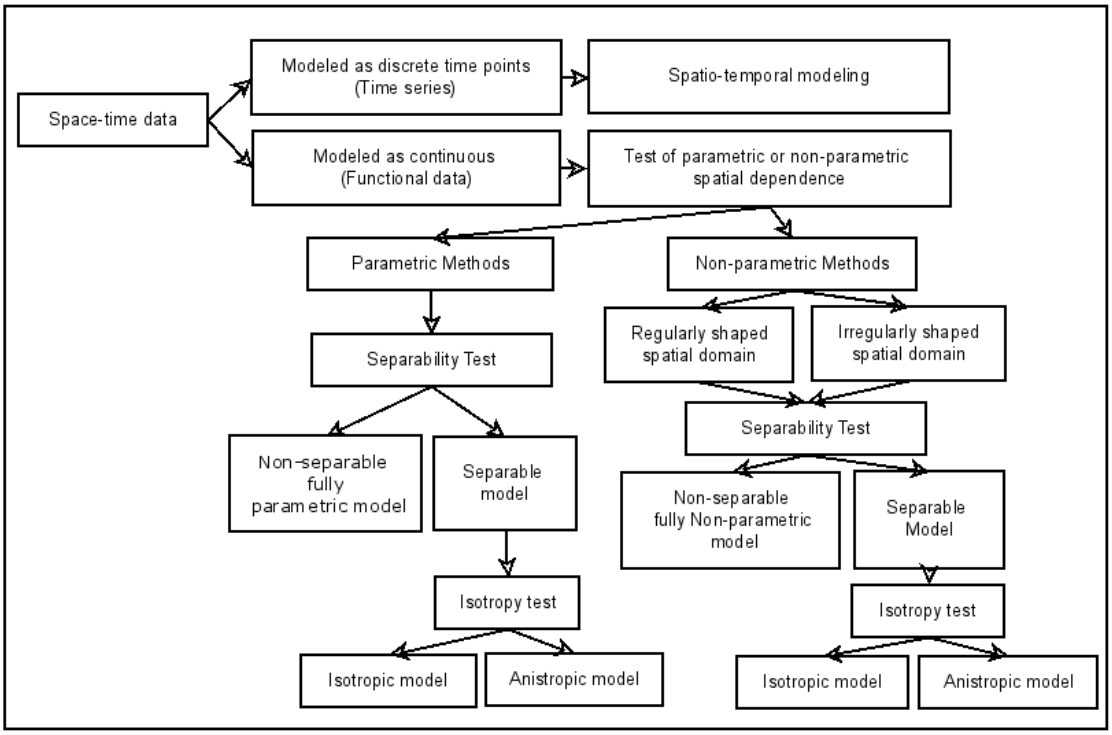

Figure 1.1: Flow chart of the analysis of space-time data using functional data analysis methods

The flow chart shows the process one should follow while analysing space-time data. First the researcher determines if the time domain is discrete or continuous. Then a test for parametric or non-parametric modelling is needed instead of making arbitrary choices. Moreover, in the non-parametric framework the spatial domain can be specified to be either regular or irregular. In both parametric and non-parametric frameworks we need to explicitly examine the separability assumption. When the assumption of separability is not valid a fully parametric or non-parametric model can be used. But such models have not been explored extensively, as the methods are computationally intensive. Similarly, with an isotropy test one should test the assumption and if it is not satisfied one should use an anisotropic

model.

In this thesis we use two datasets to develop our methodology and to test it and compare it with existing methods. The two data sets are;

- The enhanced vegetation index data set which consists of measurements reflecting the level of greenness of a 25 by 25 pixels area. The data consists of 625 observations over 276 time points.
- The electroencephalography (EEG) data which is a new dataset consist of EEG measurements for 18 subjects. Each subject was shown a stimulus, which is series of 250 pictures, 125 cars images and 125 faces images. The EEG data were recorded with 57 scalp electrodes and over 454 time points.

We use these datasets to go through this flow chart. We also generate our framework to include modelling functional data with replications. Some real life applications include replicated data where the data consists of replicated curves of the same process such as the Electroencephalography (EEG) data.

The research focuses on the analysis of spatially correlated functional data and aims to:

- Build a robust framework to determine which existing method is appropriate for spatially correlated data, generating those methods to accommodate datasets that do not fit into existing techniques.
- Develop a method to analyse replicated functional data, and apply it to the EEG data

1.5 Outline of the Thesis

This thesis is divided into 7 chapters. A brief review of each chapter and a description of the general structure of this thesis is now provided.

In **Chapter 2** we provide the reader with an overview of functional data analysis, covering its main techniques. Readers familiar with functional data analysis

can skip this chapter. In the first part of Chapter 2, we illustrate how functional data can be represented using basis expansions and discuss commonly used basis systems. Then, we provide a description of exploratory analysis methods in FDA and a functional principal components analysis approach which will be extensively used to build our framework.

In **Chapter 3** we discuss some existing techniques that are used to model correlated functional data. We review two approaches of modelling correlated functional data in detail. The first approach is spatial principal analysis of conditional expectation (SPACE), proposed by Liu et al. (2017), which models spatially correlated functional data. The second approach is called a spatio-temporal regression model with partial differential equations regularisation (ST-PDE), which is proposed by Bernardi et al. (2017), and concerns of data observed over an irregular manifold.

In **Chapter 4** we introduce our main dataset in this thesis, the EEG data and the problem of analysing replicated functional data. We provide a new framework that generalise the ST-PDE method to accommodate replicated functional data. We present a simulation study which investigates the performance of the replicated ST-PDE approach and compares ST-PDE and RST-PDE.

Chapter 5, extends the existing framework of SPACE and ST-PDE approaches to accommodates more general data structures and provides an application of the two approaches to the enhanced vegetation index data. In this chapter we compare the results obtained by applying these approaches to the EVI data.

Chapter 6, illustrates the results obtained of applying SPACE and RST-PDE approach to the EEG data and the comparison of the two approach. We review three popular classification methods and show the results of applying them to the EEG data.

In **Chapter** 7, we review and discuss the results obtained from the experimental work. Possible future work is then discussed.

Chapter 2

Statistical Background of Functional Data Analysis

In Chapter 1 we introduce FDA and show some examples of functional data. This chapter provides a background of the field of functional data analysis and includes an extensive literature review of existing techniques. Readers familiar with the literature on FDA can skip this chapter. The first section shows how functional data can be represented using smoothing techniques such as basis expansion and spline smoothing. Section 2 provides a description of the methods used for exploratory functional data analysis. Section 3 introduces functional principal component analysis. We have used Silverman and Ramsay (2005) as the main reference in this chapter. The plots and figures in this chapter are produced using the `"fda"` package in R (Ramsay et al., 2014).

2.1 Functional Data Representation

The first step in applying FDA is to convert the raw discretely observed data to functional data. Let y_j be the raw data vector, corresponding to a single replication, observed over some time points $t_j \in [T_1, T_2]$. Then the observations y_j can be written as follows

$$y_j = x(t_j) + \epsilon_j, \tag{2.1}$$

where $x(t_i)$ are smooth functional data and ϵ_j are the errors which are independent and normally distributed $\epsilon_j \sim N(0, \sigma^2)$. Most of the current methods assume that the errors ϵ_j are independent and normally distributed. However, this assumption is not valid in the case of correlated observations and FDA can handle the correlated errors.

The functional data $x(t)$ are then represented as a linear combination of basis functions.

$$x(t) = \sum_{q=1}^{Q} c_q \phi_q(t), \tag{2.2}$$

where c_q are the coefficients vector and $\phi_q(t)$ are a number Q of basis functions that are independent of each other. The number of basis function Q determines the level of smoothness. If the number of basis Q is small we might over-smooth the curve resulting in losing the important features of the data. However, as the number of basis functions increases the curves become more wiggly and might over-fit the data.

The use of an basis expansion approach allows the data to be presented with reduced errors. Furthermore, the basis system can represent data with a large number of time points t by a smaller number of coefficients. Another big advantage is that FDA can naturally deal with irregular time points. The basis expansion sets all curves to the same domain. There are many different kinds of basis functions such as Fourier basis, B-spline, polynomial basis and wavelets, etc. The choice of the basis is an important step and depends on the characteristics of the data. A brief description of some of the most common basis functions and smoothing techniques is given the following sections.

2.1.1 Fourier basis

Fourier basis is one of the most widely used basis functions. Typically, Fourier basis functions are used to represent periodic curves, where these functions repeat themselves over a period of time. The Fourier basis of size $(2q + 1)$ is given by the set

$$\begin{aligned}
\phi_1(t) &= 1, \\
\phi_2(t) &= \sin(\omega t), \\
\phi_3(t) &= \cos(\omega t), \\
\phi_4(t) &= \sin(2\omega t), \\
\phi_5(t) &= \cos(2\omega t), \\
&\vdots \\
\phi_{2q} &= \sin(q\omega t), \\
\phi_{2q+1} &= \cos(q\omega t),
\end{aligned}$$

where ω determines the period $2\pi/\omega$, the shortest time taken for x to repeat itself. Then the smooth functions can be approximated by the sum of sine and cosine functions given below.

$$\hat{x}(t) = c_0 + c_1 \sin \omega t + c_2 \cos \omega t + c_3 \sin 2\omega t + c_4 \cos 2\omega t + ... + c_{2q} \sin(q\omega t) + c_{2q+1} \cos(q\omega t),$$

where $\mathbf{c} = (c_1, \cdots, c_{2q+1})$ is the coefficient vector of the basis function. Figure 2.1 shows the first seven Fourier basis functions defined over the interval [1,20]. The first Fourier function is the constant function and represented by the black horizontal line. The rest of the functions are three pairs of sines and cosines with different periods for each pair.

The Fourier basis has the advantage that the calculation of the coefficients is done in a fast and efficient way, by using the Fast Fourier transform (FFT) Algorithm. Another advantage of Fourier basis is that the calculation of its derivatives is straightforward. For example the first derivative denoted by $D(.)$ of any pairs of sine and cosine can be calculated using the following rule

$$\begin{aligned}
D \sin(r\omega t) &= \cos(r\omega t), \\
D \cos(r\omega t) &= -\sin(r\omega t).
\end{aligned}$$

The same approach can be used to calculate higher order derivatives. Fourier basis is a very popular basis function, however, it is primarly used to fit periodic functions with no extreme changes or abrupt features (Silverman and Ramsay, 2005).

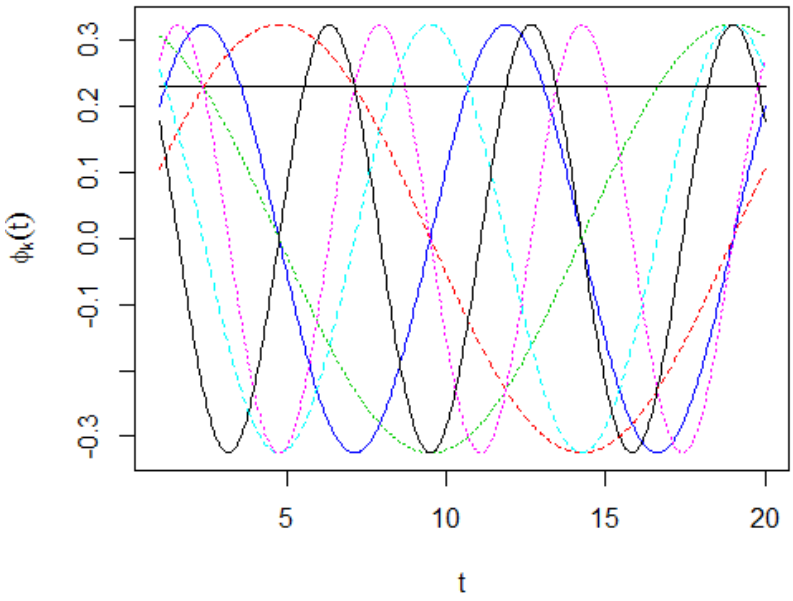

Figure 2.1: The first seven Fourier basis functions

2.1.2 Spline basis

Instead of the Fourier basis one can use a spline basis to represent noisy measurements. The spline method works by fitting piecewise polynomials to the data. A spline function of order m is defined by a piecewise polynomial of degree $m-1$. Spline functions are usually defined over the interval of the approximated function. The idea of the spline functions is to first decompose this interval into sub-intervals separated by breaking points or knots. Then for each subinterval, the spline function simply fit a polynomial which are joined together at the knots.

In contrast with Fourier basis, spline functions are commonly used for non-periodic data and are computationally fast. For more information on spline basis system see Hastie and Tibshirani (1990) and Green and Silverman (1993a).

Suppose $x(t)$ is the approximated function over some interval $[a, b]$. In order to usea spline basis system, the interval $[a, b]$ is divided into sub-intervals

$[(a, s_1), (s_1, s_2), \cdots, (s_m, b)]$ such that

$$x(t) = \begin{cases} x_0(t) & \text{for} \quad a \leq t \leq s_1 \\ x_1(t) & \text{for} \quad s_1 \leq t \leq s_2 \\ \vdots & \vdots \qquad \vdots \\ x_m(t) & \text{for} \quad s_m \leq t \leq b, \end{cases}$$

where s_i are the interior breakpoints (knots). The spline system is determined by the order of the polynomial and the location of the knots. Spline functions of order four, known as cubic spline functions, are one of the most frequently used orders in spline smoothing technique. In cubic splines, the first derivative $x'(t)$ and second derivative $x''(t)$ are equals at the knots s_i, which confirm the smoothness of the corrsponding derivatives at the knots.

One of the most popular smoothing spline approach developed by De Boor et al. (1978) is called the B-spline basis system. B-spline consists of a polynomials on specific sub-intervals and zero elsewhere, which produce a sparse design matrix. Due to this feature, the computation of the function is flexible and efficient. B-spline functions are specified by the order of the spline and the number and position of individual knots.

Suppose there are L subintervals which are connected by $\tau_l, (l = 1, \cdots, L-1)$ knots. Then the number of basis functions is equal to the order plus the number of knots $Q = m + L - 1$. A spline function $\phi(t)$ is defined as

$$\phi_q(t) = \textstyle\sum_{q=1}^{Q} c_q B_q(t),$$

where $B_q(t)$ is B-spline function and can be defined using Cox-De Boor formula (De Boor et al., 1978) as follows

$$B_q^m(t) = \tfrac{t - t_q}{t_{q+m+1} - t_q} B_q^{m-1}(t) + \tfrac{t_{q+1} - t}{t_{q+m+2} - t_{q+1}} B_{q+1}^{m-1}(t),$$

where

$$B_q^{-1}(t) = \begin{cases} 1 & t_q \leq t \leq t_{q+1}, \\ 0 & \text{otherwise.} \end{cases}$$

The most widely used B-spline functions are the cubic B-splines with order 4. Figure 2.2 illustrates the seven B-spline functions of order 4. and 3 interior knots.

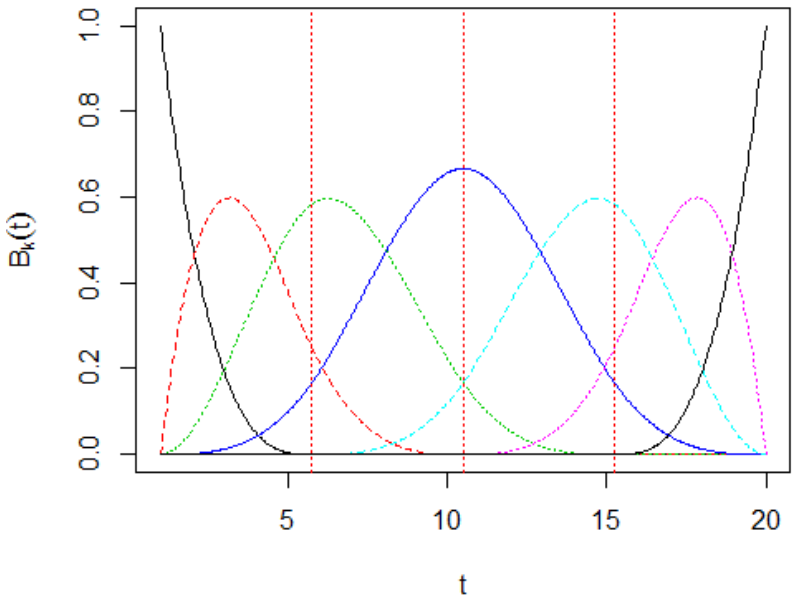

Figure 2.2: The seven B-spline basis functions for a cubic B-spline with 3 interior knots.

Fourier basis and B-spline basis are the most popular basis systems, However there are other basis functions which we describe below briefly.

2.1.3 Other Basis Systems

There are many other important basis functions that have received much attention such as Wavelets bases. Wavelets are multi-resolution basis functions that are mostly used for signal processing. They are generated from a single mother wavelet function as follows.

$$\psi_{jq}(t) = 2^{j/2}\psi(2^j t - q),$$

where j represents the scale (dilation) and q represents the shift (location). Wavelets can be useful for data that might have some discontinuities and irregular functions

with strong changes, as they have the frequency and time localization property. In the wavelets approach, the basis coefficients are calculated and then a threshold is applied to them to remove the small coefficients. The resulting estimator is frequency and time localized, which should accommodate different smoothing degrees. For more information on wavelets see (Chui and Quak, 1992).

Other basis functions such as the exponential basis functions considers a series of exponential functions $e^{\lambda_1 t}, e^{\lambda_2 t}, \cdots, e^{\lambda_q t}$ for the approximation. In contrast the power basis uses basis functions $t^{\lambda_1}, t^{\lambda_2}, \cdots, t^{\lambda_q}$, which are easy to interpret, however, the power basis functions grow quickly, which leads to inaccurate calculations.

After choosing the convenient basis function, the next step is to fit the data using smoothing techniques. In the next two section we show a smoothing procedure for a single curve using two approaches; smoothing by least squares and smoothing with roughness penalty.

2.1.4 Smoothing by Least Square

In this approach the model parameters are estimated by minimizing the sum of square of error between the observed data and the expected values, i.e. the smoothed curve. The linear smoother of equation 2.1 using the expansion of x in model 2.2 can be written as

$$\begin{aligned} SSE &= \sum_{j=1}^{m} [y_j - \sum_{q=1}^{Q} c_q \phi_q(t_j)]^2 \\ &= (\mathbf{y} - \Phi\mathbf{c})^T (\mathbf{y} - \Phi\mathbf{c}). \end{aligned} \tag{2.3}$$

where $\mathbf{y} = [y_1, \cdots, y_m]^T$ is a vector of length m representing the observation in the curve, $\mathbf{c} = [c_1, \cdots, c_q]^T$ indicates the vector of length Q of the basis coeifficents and Φ is an m by Q matrix which contains the values of the Q basis functions at the different time points t. Taking the derivatives of the equation (2.3) gives

$$2\Phi\Phi^T\mathbf{c} - 2\Phi^T\mathbf{y} = 0. \tag{2.4}$$

Then, the estimate of $\hat{\mathbf{c}}$ can be gained by solving (2.4) for $\mathbf{c}$,

$$\hat{\mathbf{c}} = (\Phi^T\Phi)^{-1}\Phi^T\mathbf{y}. \tag{2.5}$$

Finally, the data estimate (fitted values) $\hat{\mathbf{y}}$ can be derived by setting the value of $\hat{\mathbf{c}}$ in equation (2.2)

$$\hat{\mathbf{y}} = \Phi\hat{\mathbf{c}} = \Phi(\Phi\Phi^T)^{-1}\Phi^T\mathbf{y} = Sy \tag{2.6}$$

This simple linear smoother is the best estimator when the errors are independently and normally distributed. However, this might not be true in all cases. To tackle this problem we can add a weight matrix to the approximation, which will give each observation its appropriate amount of influence over the estimation. As a result, observations with small errors will have large weights while observations with big errors will have small weight. The approximation can then be written as

$$SSE = (\mathbf{y} - \Phi\mathbf{c})^T W(\mathbf{y} - \Phi\mathbf{c}), \tag{2.7}$$

where W is a symmetric positive weight matrix and can be defined by the variance-covariance matrix of the errors $W = \Sigma^{-1}$. Consequently, the vector $\hat{\mathbf{c}}$ is estimated as follows

$$\hat{\mathbf{c}} = (\Phi^T W\Phi)^{-1}\Phi^T W\mathbf{y}. \tag{2.8}$$

Thereafter, the estimated data values is written as

$$\hat{\mathbf{y}} = \Phi(\Phi^T W\Phi)^{-1}\Phi^T W\mathbf{y} = S\mathbf{y}, \tag{2.9}$$

where $S = \Phi(\Phi^T W\Phi)^{-1}\Phi^T W$ is the smoothing matrix known also as the hat matrix. Furthermore, the effective degree of freedom can be defined as $df = trace(S)$, where the trace of a square matrix is the sum of its diagonal elements.

Furthermore, in this approach it is important to specify the appropriate order of the expansion Q where a big Q can overestimate the data including the noise of the data and small Q might lose important information from the data. Choosing the number of basis is a trade-off between variance and bias, when Q is large the bias would be close to zero while the sample variance would be very high. Conversely, small Q reduces the sample variance and results in high bias. One way to obtain a better estimate is to minimize the mean squared error which is estimate as follows

$$MSE[\hat{x}(t)] = E[\{\hat{x}(t) - x(t)\}^2] = Bias^2[\hat{x}(t)] + Var[\hat{x}(t)].$$

This implies that allowing a small bias is acceptable in order to reduce the variance which in turn might reduce the over all MSE.

In practice, smoothing by least squares has some limitations such that it controls the smoothness by the number of basis function which leads to discontinuous control. However, smoothing with a roughness penalty approach allows continuous control of the smoothness. The next section briefly describes smoothness with roughness penalties.

2.1.5 Smoothing with Roughness Penalty

Unlike least square methods, roughness penalty methods uses differential equations to fit the data. The roughness penalty approach control the smoothing using a smoothing parameter λ and a penalty term. The penalized least square error is then given by

$$PENSSE = \sum_{j=1}^{m} \Big\{ (y_j - \sum_{q=1}^{Q} c_q \phi_q(t) dt \Big\}^2 + \lambda PEN_m(x), \tag{2.10}$$

The penalty term is defined by the squared derivative

$$\begin{aligned} PEN_m(x) &= \int [D^m x(s)]^2 ds \\ &= \int [D^m \mathbf{c}^T \phi(s)]^2 ds \\ &= \int \mathbf{c}^T D^m \phi(s) D^m \phi^T(s) \mathbf{c} ds \\ &= \mathbf{c}^T \Big[\int D^m \phi(s) D^m \phi^T(s) ds \Big] \mathbf{c} \\ &= \mathbf{c}^T R \mathbf{c}, \end{aligned} \tag{2.11}$$

where $R = \int D^m \phi(s) D^m \phi^T(s) ds$, Then the penalized least square estimate can be written as follows

$$PENSSE = (\mathbf{y} - \Phi\mathbf{c})^T W (\mathbf{y} - \Phi\mathbf{c}) + \lambda \mathbf{c}^T R \mathbf{c}, \tag{2.12}$$

where λ is the smoothing parameter that controls the amount of roughness. Taking the derivative and solving the equation for $\mathbf{c}$ gives

$$\hat{\mathbf{c}} = (\Phi^T W \Phi + \lambda R)^{-1} \Phi^T W \mathbf{y}, \tag{2.13}$$

Consequently, the data estimate $\hat{\mathbf{y}}$ is given by

$$\hat{\mathbf{y}} = \Phi(\Phi^T W \Phi + \lambda R)^{-1} \Phi^T W \mathbf{y} = S_\lambda \mathbf{y}. \tag{2.14}$$

In the roughness penalty approach we have λR added to the model which controls the smoothness. Green and Silverman (1993b) provide a good description of the roughness penalty approach and investigate situations that can be tackled by this approach. Furthermore, choosing the smoothing parameter plays an important role in the smoothing. Choosing the smoothing parameter is a trade-ff between bias and variance; a small smoothing parameter uses less information which results in small bias and large variance. In contrast, big smoothing parameter interpolates the data which increases bias and decreases variance. Practically, as the smoothing parameter increases the curves become more smooth while, as the smoothing decreases the curve become more wiggly. There exist many approaches for selecting the smoothing parameter such as cross validation (CV), Akaike's Information Criterion (AIC) (Akaike, 1998) and Bayesian Information Criterion (BIC) (Schwarz et al., 1978).

One of the most popular methods in selecting the smoothing parameter is cross-validation (CV). The basic idea of CV is to fit the smooth function to the data except for one data point, which is used as a validation sample. Then the fitted smooth function is used to predict the fitted value for the omitted data point. This is repeated for each data observation and the performance of these predictions is measured. This procedure is computed for a range of different smoothing parameter values, and we choose the λ values with the best performance.

Cross Validation can be applied to various cases, However, it has some limitations. First, the method is clearly computationally intensive, especially for big datasets. Second, the method may fit the noise of the data which can affect the smoothing. To overcome these problem, Wahba and Craven (1978) introduced a generalized cross validation (GCV) method which is basically a wighted version of the cross-validation approach.

Generalized cross validation (GCV) is a very popular approach for choosing smoothing parameters. GCV is defined as

$$GCV(\lambda) = \Big(\frac{n}{n - df(\lambda)}\Big)\Big(\frac{SSE}{n - df(\lambda)}\Big), \tag{2.15}$$

where $df(\lambda) = trace(S)$ is the degree of freedom of the smoothing parameter. For

more information of cross validation (CV) and generalized cross validation (GCV) see (Gu, 2013).

Now, we will use the smoothed functions as the new random variables and perform exploratory analysis on them.

2.2 Exploratory Functional Data Analysis

Exploratory analysis can be carried out with functional data analysis to summarise the general structure of the data and explore the main features. This includes estimating the mean, variance, covariance and correlation. Assuming a set of n curves observed at different time points $\{x_i(t), i = 1, \cdots, n\}$, the point-wise mean function is given by

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

where the mean is calculated from the curves at each time point and can be represented by one curve. Figure 2.3 illustrates the point-wise mean for an example data consist of 50 curves vary over 20 time points. The data are simulated from a multivariate distribution and then are converted to functional data using the techniques described previously.

In a similar way the point-wise variance function is given by:

$$Var_x(t) = \frac{1}{n-1}\sum_{i=1}^{n}[x_i(t) - \bar{x}(t)]^2.$$

The variance function is computed by the sample variance function which explain the variation between the curves at one time point say t. To investigate the dependence of the curves between different time points such as s and t, it is important to estimate the covariance function and the associated correlation function. The covariance function across time points is given by:

$$cov_x(s,t) = \frac{1}{n-1}\sum_{i=1}^{n}\{x_i(s) - \bar{x}(s)\}\{x_i(t) - \bar{x}(t)\}.$$

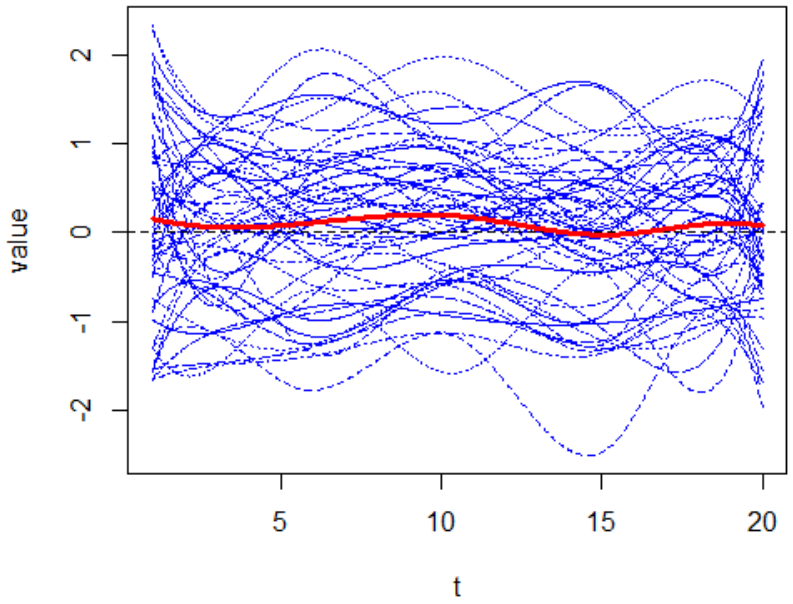

Figure 2.3: Smooth functional data with the point-wise mean (red line).

Figure 2.4 shows the variance covariance surface of the example data. The corresponding correlation function is given by:

$$corr_x(s,t) = \frac{cov_x(s,t)}{\sqrt{var_x(s)var_x(t)}}.$$

In some cases it is required to explore the variability between pairs of functions (x, y) and that can be done by calculating cross-covariance function which is given by

$$cov_{x,y}(s,t) = \frac{1}{n-1}\sum_{i=1}^{n}\{x_i(s) - \bar{x}(s)\}\{y_i(t) - \bar{y}(t)\},$$

and the corresponding cross-correlation function is given by:

$$corr_{x,y}(s,t) = \frac{cov_{x,y}(s,t)}{\sqrt{var_x(s)var_y(t)}}.$$

These functions are comparable to the classical multivariate measurements and can be computed to explain the general characteristics of the data.

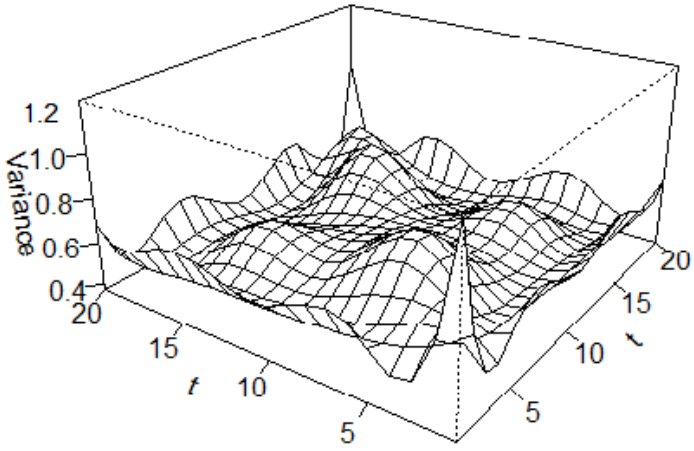

Figure 2.4: The variance-covariance function of the functional data.

One of the most useful and commonly used approaches to explore the variability in functional data is functional principal component analysis (FPCA). Functional principal components is one of the methods that was considered as an exploratory method by Silverman and Ramsay (2005). Functional principal component analysis (FPCA) is commonly used to determine the amount of variation and illustrate the trend in functional data. In the next section we describe the theory behind functional principal component analysis and illustrate how these components can be estimated. We also show how fPCA has been developed over time. We will make use of FPCA in developing many methodologies in Chapter 5.

2.3 Functional Principal Component Analysis

Functional principal component analysis (FPCA) is one of the most popular dimension reduction and modelling techniques in functional data analysis. As functional data can be interpreted as infinite dimensional multivariate data, FPCA can performed by generalising multivariate principal component techniques to infinite dimensions (Dauxois et al., 1982). FPCA is commonly used as dimension reduction approach where it computes a small number of components that represents most of the variation of the full functional data. Furthermore, FPCA can be used to inves-

tigate the modes of variation in the functional data where each component explains some amount of variation. The data then can be written as a linear combination of these components and can thus be used as new basis system. It is also considered as a rotation of the axes coordinates where the new axes coincide with the maximum variation of the data and are orthogonal to each other. We will make extensive use of this approach to analyse spatially correlated functional data in Chapter 5.

Many features of classical principal component analysis extend from vector space to the square integrable functional space. Computationally, FPCA differs from PCA as it replaces the vectors by functions, matrices by linear operators and summations by integrations. Note that the standard PCA can not be directly applied to high dimensional data where the number of observation is less than the number of variables $n < p$. A comparison between classical PCA and the functional PCA by Viviani et al. (2005), in the context of modeling fMRI data, clearly showed that FPCA was more effective in recovering the signals generated from different experimental conditions compared to the multivariate version.

2.3.1 The FPCA Methodology

In this section we show the details of how FPCA is calculated. Let $x_i(t)$ be a set of continuous functions defined over a bounded continuous time interval τ with mean $\mu = E(x(t))$. The covariance operator of $x(t)$ can be written as

$$G(s,t) = cov((x(s), x(t)).$$

The basic idea of FPCA is to find a weight function ξ_k that maximises the variation so that the majority of the variation in the data can be attributed to the linear combination given by the weight. The covariance function has a spectral decomposition to eigenvalues and eigenfunctions which is given by

$$G(s,t) = \sum_{k=1}^{\infty} \lambda_k \xi_k(s)\xi_k(t), \quad s,t \in \tau. \tag{2.16}$$

Where $\{\xi_k(t)\}_{k=1}^{\infty}$ are the eigenfunctions of FPC and $\lambda_{k=1}^{\infty}$ are non-increasing eigenvalues that indicate the proportion of variation explained by the components.

The eigenfunctions can be determined by solving the following eigen-equation

$$\int G(s,t)\xi(t)dt = \lambda\xi(s), \tag{2.17}$$

subject to the constraints $\int \xi_1(t)^2 dt = 1$ which is a normalisation constraint and $\int \xi_k(t)\xi_l(t)dt = 0$ to ensure that the components are orthogonal.

According to the Karhunen-Loeve decomposition each realisation $x_i(t)$ has the following expansion,

$$x_i(t) = \mu(t) + \sum_{k=1}^{\infty} \alpha_{ik}\xi_i(t), \quad i = 1, 2, ..., n, \tag{2.18}$$

where α_{ik} are independent functional principal component scores with expectation $E(\alpha_{ik})$ and variance λ_k. Usually, a finite number of components K is chosen to provide a good approximation of $x_i(t)$,

$$x_i(t) \approx \mu(t) + \sum_{k=1}^{K} \alpha_{ik}\xi_i(t), \quad i = 1, 2, ..., n \tag{2.19}$$

The number of principal component K is determined by the amount of variation explained by these components. The optimal K is not too large but provide a good approximation that are very close to the original data. The number of FPCs can be specified empirically based on the data by plotting the number of principal components with their corresponding eigenvalues known as (scree plot). The number of components is chosen to be the number where the curve starts to be flat line.

The functional principal component scores can now be defined as the integration of function values $x_i(t)$ with weight functions $\xi_k(t)$. For example, the first FPC scores are given by:

$$\alpha_{i1} = \int \xi_1(t)x_i(t), \qquad \text{for} \quad i = 1, \cdots, n.$$

2.3.2 The FPCA Estimation

The equation to obtain the eigen-values and eigen-functions given in 2.17 is difficult to be solved. A standard approach to change the equation similar to the multivariate

PCA can be done by discretizing the observed functions into a fine grid (Rao, 1958). Another method is to approximate FPCA using quadrature formula (Castro et al., 1986). The most widely used technique is to use basis expansion to represents both the observed functions $x_i(t)$ and the eigenfunctions $\xi(t)$ which we describe below.

Suppose that the observed functions x_i has the following basis expansion

$$x_i(t) = \sum_{k=1}^{K} c_{ik}\phi_k(t).$$

This expansion can be written in a matrices form by defining a vector of the observed functions $\mathbf{x}$ and and a vector of basis functions $\mathbf{\Phi}$. Then the expansion can be written as follows

$$\mathbf{x} = C\mathbf{\Phi} \ ,$$

where C is a matrix that contain the coefficients of the basis functions with number of rows equal to the number of observations and the number of columns equal to the number of basis functions and $\mathbf{\Phi}$ is a vector of basis functions $\phi_q(t), q = 1, \cdots, Q$. Then the covariance can be written in the matrix form as follows

$$N^{-1}\mathbf{\Phi}^T(s)C^TC\mathbf{\Phi}(t).$$

Furthermore, suppose that the eigenfunction $\xi_k(t)$ has the following basis expansion

$$\xi(t) = \sum_{k=1}^{K} \phi_k(t)^T b_k = \mathbf{\Phi}(t)^T\mathbf{b},$$

where $\mathbf{b}$ are the basis coefficients of the eigenfunction $\xi(t)$. Then, the eigen-equation (2.17) can be written as

$$\int N^{-1}\mathbf{\Phi}(t)^TC^TC\mathbf{\Phi}(t)\mathbf{\Phi}(t)^T\mathbf{b}dt = \lambda\mathbf{\Phi}(t)^T\mathbf{b}. \tag{2.20}$$

Let W be a $Q \times Q$ matrix such that $W = \int \phi(t)\phi(t)^T$ Then (2.20) can be written as

$$N^{-1}\phi(s)^TC^TCW\mathbf{b} = \lambda\phi(t)^T\mathbf{b}. \tag{2.21}$$

Since the equation is true for all arguments t, then it can be written as

$$N^{-1}C^TCW\mathbf{b} = \lambda\mathbf{b}, \tag{2.22}$$

subject to the normalization constraint $\|\xi\|^2 = \mathbf{b}^TW\mathbf{b} = 1$. Define $\mathbf{u} = W^{1/2}\mathbf{b}$ then, we have the symmetric equation

$$N^{-1}W^{1/2}C^TCW^{1/2}\mathbf{u} = \lambda\mathbf{u}. \tag{2.23}$$

The coefficient vector of the eigenfunctions C can be estimated as $\mathbf{b} = W^{-1/2}\mathbf{u}$ and thus the principal component scores α_{ik} can be obtained by

$$\alpha_{ik} = \int \xi_k(t)x_i(t) = CW\mathbf{b}. \tag{2.24}$$

These K scores represent the variation in the functional data and can be used to approximate the functions $x_i(t)$.

2.3.3 FPCA Extension

Standard functional principal analysis approach is designed to analyse functional data with non-missing values, dense and uncorrelated functional data. There exist some modifications which have been applied to FPCA regarding to different situations with many practical problems such as outliers, sparsity and correlated functional data.

When the FPCA's obtained from functional objects are very rough they become difficult to interpret. In those cases one may choose to smooth the FPCA's. Rice and Silverman (1991) proposed a method that incorporate the smoothness in the estimation of functional principal component by using a roughness penalty. This is done by applying a different smoothing parameter for each component. Their approach is computationally intensive as the eigen-equation is solved for each component separately. Silverman et al. (1996) introduced a method the overcome this limitation by estimating the smoothed principal components using a single smoothing parameter.

Another challenge is the presence of outliers. The functional principal component approach is mainly based on the covariance function which can be affected by outliers. Locantore et al. (1999) presented a robust method that deals with the problem. To generalize the functional principal component to data with multiple groups, Benko et al. (2009) introduced common functional principal components and presented a bootstrap test to test wheather the eigenvalues, eigenfunctions, and mean functions of two functional data samples are the same, while Boente et al. (2010) provide estimators of the common functional principal components and studied inference of these estimators.

The modifications also considered functional data with sparse observations, where FPC scores are not well approximated by the integration. Yao and Lee (2006) proposed a non-parametric method that deals with sparse data, by estimating the functional principal component via the conditional expectations (PACE) of the data. However, PACE assumed that the observations are uncorrelated. Liu et al. (2017) proposed a technique to model the spatial correlation in functional data by correlating functional principal component scores using their conditional expectations. The method is designed to accommodate data with missing values. More details of SPACE is provided in chapter 3. Another approach for the analysis of dependent functional data was proposed by Hörmann et al. (2015). This approach takes into account the correlation between the observations and the correlation within the the observations. It extended the dynamic PCA approach developed by Brillinger (1981) which is designed for vector time series, to the dynamic functional principal component analysis. The components obtained by this method accounts for the majority of the dynamics and variability in the data.

Another approach for obtaining FPCA is modeling functional data with multiple levels. Di et al. (2009) provided a multilevel FPCA which is designed for multilevel functional data by using multilevel mixed model. However, The multilevel FPCA is not applicable for high dimensional data where the covariance can not be calculated and stored. Zipunnikov et al. (2011) developed a high dimensional multilevel functional principal component analysis method which accommodates the high-dimensional data. THe method provide an algorithm that calculates the eigen-

values and eigen-function without the need to calculate the covariance operators.

In summary, FPCA has been considered as an important tool in functional data analysis and can be applied to various applications.

Chapter 3

Modelling Dependent Functional Data

3.1 Introduction

In this chapter, we review some existing techniques of modelling dependent functional data. Then we discuss two approaches that we apply to model spatially dependent functional data. The first approach, by Liu et al. (2017), focuses on modelling the spatial dependence of gridded data parametrically. The spatial correlation is modelled by correlating the functional principal component scores which are estimated using conditional expectation. The approach is called spatial principal analysis of conditional expectation and is described in details in section 3.3. The second approach, by Bernardi et al. (2017), called spatio-temporal regression model with partial differential equations regularisation (ST-PDE), models data sampled over complex boundaries non-parametrically. ST-PDE handles the complex boundary problem by using partial differential equations and the finite element method. The ST-PDE approach is described in detail in Section 3.4. For simplicity, we have used the same notation as in Liu et al. (2017) in Section 3.3 and the notation from Bernardi et al. (2017) in section 3.4.

3.2 Existing techniques for modelling spatially correlated functional data

Though dependence can arise from sources other than space, we will focus our discussion on spatially correlated functional data. In this section we will present some existing techniques that have been proposed in the last few years for analysing spatially correlated functional data. In the parametric framework, Liu et al. (2017) proposed the first comprehensive study to analyse spatially correlated data. They developed a new approach called the spatial principal analysis of conditional expectation (SPACE) which calculates spatial correlation assuming separable spatial and temporal covariance. Using the anisotropic Matérn family a parametric model was fitted to empirical spatial correlations at a sequence of spatial separations. Though their approach works for data that do not strictly follow separability, Liu et al. (2017) showed that the estimates are better for the separable covariance than non-separable covariance with finite samples. Moreover, this approach can calculate the spatial correlation for each spatial separation vector which can be used to reconstruct sparsely sampled curves. We review the SPACE approach in detail in 3.3. Liu et al. (2017) also provide a bootstrap test to test the separability of the covariance.

Around the same time, using a strictly non-parametric approach, Aston et al. (2015) presented an alternative test to investigate the separability assumption for the covariance. In their approach the difference between the sample covariance operator and its separable approximation is projected onto the first eigen-function of the covariance of the data. Furthermore, the distribution of the test statistic is approximated using bootstrap methods.

Considering functional data that are observed on a spatially irregularly shaped manifold, Sangalli et al. (2013) provided a spatial spline regression model that deals with data observed over a complex spatial domain. The proposed model is designed to accommodate complex boundary conditions and gaps and holes in regions. The method uses penalised bivariate spline smoothing with a roughness penalty that consists of Laplace operators. The spatial domain is modelled by a finite element method. While their method is designed only for univariate spatial data and thus

can not be applied to data observed over different time points. Bernardi et al. (2017) extended their method to accommodate space-time data, where two roughness penalties are included in the model, one for time and another for space. This approach in reviewed in 3.4. Later in Chapter 4, we extend this approach to accommodate replicated functional data, one of the major contribution of this thesis.

Marra et al. (2012) proposed a generalised additive model which includes a smoothing approach for spatio-temporal data. The smoothing procedure combines a cubic spline basis functions and a soap film basis function for time and space, respectively. Alternatively, Ignaccolo et al. (2014) developed a kriging approach for functional data varying over time. The approach models the spatial and temporal trends and can also include covariate estimation.

3.3 Spatial Principal Analysis of Conditional Expectation

This section describes in detail the method proposed by Liu et al. (2017) called spatial principal analysis of conditional expectation (SPACE). We demonstrate this approach by analysing a remotely sensed vegetation index dataset in Chapter 5. Their approach relies on two crucial steps. First, the spatial correlation of the functional data is modelled by building correlated functional principal component scores. Then, the empirical correlation is estimated using Matérn model. Let us now discuss the method starting with the real life data example that was used to motivate their approach.

3.3.1 Correlated Gridded Functional Data

Functional data observed over units distributed within a physical domain are likely to be spatially correlated in many applications. An example of correlated functional data is the enhanced vegetation index (EVI) data used in Liu et al. (2017). The EVI data was obtained from surface spectral reflectance satellite measurements over a

period of time. This dataset covers an area of 25x25 pixel located in Harvard Forest Long Term Experimental Research site in Petersham, Massachusetts, USA. The EVI measurements take values between -1 and 1 reflecting the level of greenness. The EVI data used in the paper are obtained at 8-day intervals (46 data points per year) for the period from January 1, 2001 to December 31, 2006 (Liu et al., 2012). The complete data consist of 625 replicated curves over 276 time series, where each curve corresponds to one pixel. Figure 3.1 gives an overview of the data structure where the bottom panel of the graph shows the data for all 6 years, while the top panel zooms in on the data for the first year year.

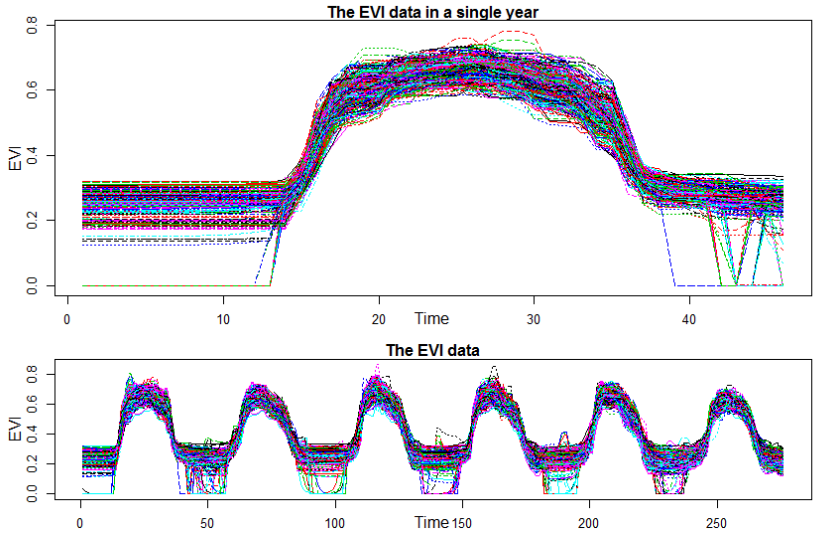

Figure 3.1: EVI data for 625 pixels over time. Each curve represents the data for one pixel. The bottom panel of the graph shows the whole data while the top panel shows the data for a single year.

The individual pixels seem to be high correlated and a seasonal effect is one of the dominant source of variation. The plot also reveals some variation from year to year and a close inspection shows gaps or missing observations for some locations.

3.3.2 Dependent Functional Data Model

In chapter 2 we reviewed the classical functional data model designed to analyse observations that are independent i.e. with no spatial correlation. We showed how the functional data can be represented by functional principal components.

For location i we have noisy measurements $y_i(t_j)$ at time t_j sampled from smooth functions $x_i(t)$ which have mean function $\mu = EX(t)$ and covariance function $G(s,t) = cov(x(s), x(t))$. For location i, these smooth functions can be represented by the functional principal components as follows

$$\begin{aligned} Y_i(t) &= x_i(t) + \epsilon_i(t) \\ &= \mu(t) + \sum_{k=1}^{\infty} \alpha_{ik}\xi_k(t) + \epsilon_i(t), \end{aligned} \tag{3.1}$$

where $\{\xi_k(t)\}_{k=1}^{\infty}$ are the functional principal component FPC functions and α_{ik} are the FPC scores with corresponding variance λ_k. In the uncorrelated case, the FPC scores are assumed to be independent, and a previous method by Yao and Lee (2006) for the principal components analysis can even be used for the analysis of sparsely and irregularly spaced observations. The approach called principal components analysis through conditional expectation (PACE), computes the principal component scores by their expectation conditioning on all observations, which allows one to analyse sparsely observed data. However, PACE is not designed to handle correlated functional data. The SPACE approach builds on the PACE approach to model dependent functional data.

In particular, in SPACE the FPC scores α_{ik} are assumed to be correlated across each location i for each component k. The covariance function between two eigenfunctions is given by

$$cov(\alpha_{ip}, \alpha_{jq}) = \begin{cases} \rho_{ij}(k)\lambda_k & \text{if } p = q = k, \\ 0, & \text{otherwise.} \end{cases} \tag{3.2}$$

Here $\rho_{ij}(k)$ estimates the correlation between the kth FPC scores for the curve i and j. We assume that when $p = q = k$ we have a correlation otherwise there is no correlation. This assumption gives the separability wich makes the model simpler and the computation easier. When we choose to retain only the first K eigenfunctions, the covariance between two realisation $x_i(s)$ and $x_j(t)$ can be written as

$$\begin{aligned} cov(x_i(s), x_j(t)) &= \xi(s)^T cov(\alpha_{\mathbf{i}}\alpha_{\mathbf{j}}^{\mathbf{T}})\xi(t) \\ &= \xi(s)^T \text{diag}(\rho_{ij}(1)\lambda_1, \rho_{ij}(2)\lambda_2, ..., \rho_{ij}(K)\lambda_K)\xi(t). \end{aligned} \tag{3.3}$$

In the above model the covariance is still not separable. Once we impose the condition that the between-curve correlation does not depend on k so $\rho_{ij}(k) = \rho_{ij}$ then the covariance between the FPC scores can be written as $Cov(\alpha_{\mathbf{i}}, \alpha_{\mathbf{j}}) = \rho_{ij}\text{diag}(\lambda_1, ..., \lambda_k)$ and the covariance between $x_i(s)$ and $x_j(t)$ can be simplified as

$$cov(x_i(s), x_j(t)) = \rho_{ij}\xi(s)^T\text{diag}(\lambda_1, ..., \lambda_k)\xi(t) = \rho_{ij}Cov(x(s), x(t)). \tag{3.4}$$

The covariance in this equation is separable, which assumes that the correlation across curves is independent of the correlation across time. We now focus on the spatial correlation. One of the popular methods to estimate the spatial correlation is the Matérn covariance. More details of the Matérn model are given in the next section.

3.3.3 Matérn covariance

Matérn covariance is a parametric model that is used to model the spatial correlation between two measurements observed at two locations. Denoting the distance between observations i and j by d the Matérn correlation is given by

$$\rho_{ij} = \rho(d; \zeta, \nu) = \frac{1}{2^{\nu-1}\Gamma(\nu)}\left(\frac{d}{\zeta}\right)^{\nu} K_{\nu}\left(\frac{d}{\zeta}\right), \tag{3.5}$$

where d is the distance between the locations of the two observations, K_ν is the modified Bessel function of the third kind of order $\nu > 0$ (description of Bessel function can be found in (Abramowitz and Stegun, 1965)). This model is indexed by two parameters, a range parameter ζ which rescales the distance and a smoothing parameter ν which controls the degree of smoothness. The range parameter ζ is also known as a decay parameter because it controls how fast the correlation drops with the distance d. Figure 3.2 shows how these parameters can affect the correlation function where higher ζ produces higher correlation over longer distances while higher ν leads to higher correlation at shorter distances.

The exponential class is a special case of the Matérn class when the smoothing parameter ν is equal to 0.5. It is known as the autoregressive model of order one AR(1) model in time series literature. On the other hand, when the smoothing parameter ν goes to ∞ the model converges to the squared exponential covariance

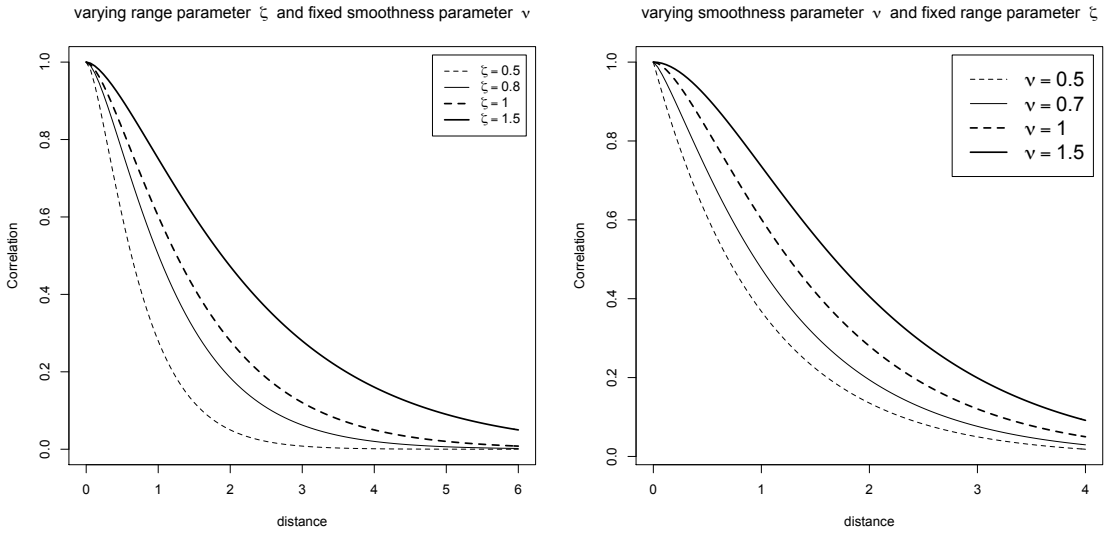

Figure 3.2: The correlation estimation using the Matérn function. The left panel shows the correlation estimation with fixed smoothing parameter ν and varying range parameter ζ. The right panel shows the correlation estimation with fixed range parameter ζ and varying smoothing parameter ν.

function in the Gaussian process. The Matérn function is more flexible in modelling the spatial correlation than other functions due to the smoothing parameter in the model.

The Matérn covariance function by default assumes isotropic covariance function which indicates that the covariance is the same for all directions. This is a strong assumption that should be checked before using the model. If the assumption is not valid for the application then, a transformation of the spatial coordinates is needed. This can be achieved by adding two parameters, θ which is an anisotropy angle specifies how much the axes are rotated and δ, which is the anisotropy ratio and defines how much the axes are stretched or shrunk. Furthermore, the distance d in the isotropic Matérn is estimated by the Euclidean distance $\rho(d;\zeta,\nu) = \rho(\sqrt{\Delta^T\Delta};\zeta,\nu)$. While, in the anisotropic case we need to implement a transformation to new coordinates. Let Δ^* be the new separation vector between two locations defined by

$$\Delta^* = \begin{bmatrix} \Delta_x^* \\ \Delta_y^* \end{bmatrix} = \begin{bmatrix} \sqrt{\delta} & 0 \\ 0 & 1/\sqrt{\delta} \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} \Delta_x \\ \Delta_y \end{bmatrix} = SR\Delta,$$

where S is the scaling matrix, R is the rotation matrix and The spatial separation

vector Δ between two locations is defined by Δ_x which is the difference between the two locations along the x-axis and Δ_y which is the difference between the two locations along the y-axis. Subsequently, the new distance function is defined as $d^*(\Delta, \theta, \delta) = \sqrt{\Delta^{*T}\Delta} = \sqrt{\Delta^T R^T S^2 R \Delta}$. Thereafter, the anisotropy correlation function is given by

$$\rho^*(\Delta; \theta, \delta, \zeta, \nu) = \rho(d^*(\Delta, \theta, \delta); \zeta, \nu) = \rho(\sqrt{\Delta^T R^T S^2 R \Delta}; \zeta, \nu). \tag{3.6}$$

Then, the covariance function in equation (3.2) can be written as;

$$cov(\alpha_{ip}, \alpha_{jq}) = \begin{cases} \rho_k^*(\Delta_{ij})\lambda_k & \text{if } p = q = k, \\ 0, & \text{otherwise.} \end{cases} \tag{3.7}$$

this equation represent the estimation of the covariance between functional principal component scores. Sequentially, the cross-covariance between the curves in (3.3) can be written as

$$G_\Delta(s,t) = \xi(s)^T \begin{bmatrix} \rho^*(\Delta)\lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \rho_k^*(\Delta)\lambda_k \end{bmatrix} \xi(t). \tag{3.8}$$

It is assumed that $\rho_1^*(\Delta)\lambda_1 > \rho_2^*(\Delta)\lambda_2 > \cdots > \rho_k^*(\Delta)\lambda_k > 0$ at that point $\{\rho_k^*(\Delta)\lambda_k\}_{k=1}^K$ are the eigenvalues of the cross-covariance $G_\Delta(s,t)$. However, $\rho_k^*(\Delta)$ is evaluated as the ratio of the eigenvalues of the cross-covariance $G_\Delta(s,t)$ and the cross-covariance $G_{(0,0)}(s,t)$ when $\Delta = (0,0)$. Then we can write

$$\rho_k^*(\Delta) = \frac{\hat{\lambda}_k(\Delta)}{\hat{\lambda}_k(0,0)}. \tag{3.9}$$

Once we obtain $\rho_k^*(\Delta)$ for all Δ values then the Matérn model parameters can be estimated by using these values to fit (3.6).

3.3.4 Mean and covariance estimation

In functional data analysis the mean and covariance are assumed to be smooth functions and can be estimated using a local linear smoother. The weight of the smoother is defined by a kernel density function and the bandwidth is defined using cross-validation. In this section we illustrate the estimation of mean and covariance functions and the variance of measurement errors.

Mean estimation

First, we estimate the mean function over pooled observations, as it can easily accommodate missing values. Following Yao and Lee (2006), we estimate the mean function $\hat{\mu}(t)$ by a local linear smoother. The mean function can be estimated by minimising the following with respect to β_0 and β_1

$$\sum_{i=1}^{m_i}\sum_{j=1}^{m_j} k_1\Big(\frac{t_{ij}-t}{h_\mu}\Big)(y_{ij}-\beta_0-\beta_1(t-t_{ij}))^2, \tag{3.10}$$

where k_1 is one dimensional kernel function and h is the bandwidth that controls the smoothing by specifying the size of the neighbourhood around t_{ij}. Then $\hat{\mu}(t)=\hat{\beta}_0$.

Cross-covariance surface

The cross-covariance surface is the covariance between any two curves over all different time points. Suppose we have $x_i(s), x_j(t)$ observations for two curves i and j at time points s and t. Then, the cross covariance function between the two locations is given by $G_{ij}(s,t) = cov(x_i(s), x_j(t))$. Varying over s and t this represents a surface. However, the cross-covariance can be estimated by smoothing the raw cross-covariance $D_{ij}(t_{ik}, t_{jl}) = (Y_i(t_{ik} - \hat{\mu}(t_{ik}))(Y_j(t_{jl} - \hat{\mu}(t_{jl}))$ using local linear smoothing.

In modelling the correlation, second order spatial stationarity of the fPC score process is assumed. This indicates that the covariance of the underlying process depends only on the separation vector between two points. However, this applies the stationarity to the observation space as well.

Consider a collection of location pairs $n(\Delta) = \{(i,j), \Delta_{ij} = (\Delta_x, \Delta_y) \quad \text{or} \quad \Delta_{ij} = (-\Delta_x, -\Delta_y)\}$ with the same covariance function $G_\Delta(s,t)$. However, when $\Delta = -\Delta$ then all raw covariances constructed based on locations in $n(\Delta)$ could be used to estimate $G_\Delta(s,t)$

$$E(D_{ij}(t_{ik}, t_{jl})) = G_{ij}(t_{ik}, t_{jl}) + \delta(i=j, s=t)\sigma^2, \tag{3.11}$$

where $\delta(i=j, s=t)$ is equal to 1 if $i=j$ and $s=t$, and 0 otherwise while σ is the variance of measurements errors which is estimated in the next section.

The covariance estimation fall under two situations, one when $i = j$ the covariance of the curve with itself and $i \neq j$ when we consider two different curves. First in the case of $i \neq j$ the cross covariance surface $G_\Delta(s,t)$ is estimated by minimizing the following

$$\sum_{(i,j)\in n(\Delta)} \sum_{i=1}^{m_i} \sum_{j=1}^{m_j} k_2\Big(\frac{t_{ik}-s}{h_G}, \frac{t_{jl}-t}{h_G}\Big)\Big(D_{ij}(t_{ik},t_{jl}) - \beta_0 - \beta_1(s-t_{ik}) - \beta_2(t-t_{jl})\Big)^2. \tag{3.12}$$

k_2 is the two-dimensional Gaussian kernel, h is the step size between two time points and $\hat{\beta}_0$, $\hat{\beta}_1$, $\hat{\beta}_2$ are the minimizer of (3.12), while $\hat{G}_\Delta(s,t) = \hat{\beta}_0$.

Secondly, in the case of $i = j$, we only need to estimate the covariance surface and this can be computed directly by the PACE method (Yao and Lee, 2006).

The $G_{(0,0)}(s,t)$ is estimated by minimising the following

$$\sum_{i=1}^{n} \sum_{1\leq j\neq k\leq m} k_2\Big(\frac{t_{ik}-s}{h_G}, \frac{t_{il}-t}{h_G}\Big)\Big(D_{ii}(t_{ik},t_{il}) - \beta_0 - \beta_1(s-t_{ik}) - \beta_2(t-t_{il})\Big)^2, \tag{3.13}$$

with respect to $\hat{\beta}_0, \hat{\beta}_1$ and $\hat{\beta}_2$. Then $\widehat{G}_{(0,0)}(s,t) = \hat{\beta}_0$.

Then the eigenfunctions $\xi_k(t)$ and eigenvalues λ_k of the cross-covariance can be given by solving the following

$$\begin{aligned} &\int_\tau \hat{G}_\Delta(s,t)\hat{\xi}_k(t)dt = \lambda_k\hat{\xi}_k(s) \\ &\int_\tau \hat{\xi}_j^2(t) = 1 \quad and \quad \int_\tau \hat{\xi}_j(t)\hat{\xi}_k(t) = 0. \end{aligned} \tag{3.14}$$

Variance estimation

The measurement error variance is estimated from (3.11) as the difference between $E(D_{ii}(t_{ik},t_{ik}))$ and the cross covariance when $(i = j)$, $G_{(0,0)}(t_{ik},t_{ik})$. First, we need to smooth the empirical covariance $D_{ii}(t_{ik},t_{ik})$ to be in the same smooth form as $G_{(0,0)}(t_{ik},t_{ik})$. The smooth estimate of the empirical covariance can be calculated by minimising the following

$$\sum_{i=1}^{n} \sum_{k=1}^{m} k_1\Big(\frac{t_{ik}-t}{h_v}\Big)\Big(D_{ii}(t_{ik},t_{ik}) - \beta_0 - \beta_1(t-t_{ik})\Big)^2, \tag{3.15}$$

with respect to $\hat{\beta}_0$ and $\hat{\beta}_1$. Then the smoother of the empirical covariance is $\hat{V}(\tilde{t}) = \beta_0$.

Then σ^2 is given by

$$\sigma^2 = max\Big(0, \frac{1}{|\tau_e|}\int_{\tau_e}\big(\hat{V}(t) - \hat{G}_{(0,0)}(t,t)\big)dt\Big), \tag{3.16}$$

where $\hat{V}(t)$ is the smoother of $D_{ii}(t_{ik}, t_{ik})$ and τ_e is the effective range which takes only a middle part of the closed interval τ in order to lower the boundary effect.

Bandwidth smoothing parameter

The choice of the bandwidth is critical for some approaches such as the local linear regression smoother. There are multiple approaches that can be used to select optimal bandwidth such as plug-in methods which are based on minimising mean integrated squared error (see (Woodroofe, 1970) and (Sheather and Jones, 1991)). A common approach of parameters selections is cross validation (CV) (see (Rudemo, 1982) and (Bowman, 1984)). In addition, there are many papers which discuss the choice of bandwidth and compare existing techniques see Jones et al. (1996), Sheather (2004) and Scott (2015).

In this chapter the bandwidth h is chosen by cross validation method specifically leave one point out (LOPO) cross validation with data binning. This is done using (sm) package (Bowman and Azzalini, 2014) in R (R Core Team, 2013) .

For a large dataset, binned data are used to increase the computational speed. The binning procedure constructs a frequency table associated with an appropriate interval covering the range of independent variables. Then, the binned data replace the independent variable by the midpoints of the bins and each observation of the dependent variables by the mean of its values across the the corresponding bin. The binned data then are used to implement cross validation which leaves one point (bin) out in turn.

3.3.5 FPC scores estimation and curve reconstruction

SPACE has the advantages that it can be used to reconstruct the curves and that it can also work when the data include missing values. Equation (3.1) is used to reconstruct the curve $x_i(t)$, where

$$x_i(t) = \mu(t) + \sum_{k=1}^{K} \alpha_{ik}\xi_i(t).$$

$\{\hat{\xi}_k\}_{k=1}^{K}$ estimation is given in (3.14) and $\hat{\mu}(t)$ is estimated in (3.10). However, we need to find a way to estimate the functional principal component scores α_{ik}. The best linear unbiased predictors (BLUP) (Henderson, 1950) of the FPC scores are give by

$$\alpha_{ik} = E[\alpha_{ik}|y_{ij}] \quad \text{where} \quad i = (1, \cdots, n) \quad \text{and} \quad j = (1, \cdots, m), \tag{3.17}$$

which is the conditional expectation under Gaussian assumptions. Suppose $\boldsymbol{y}_i = (y_i(t_{i1}), \cdots, y_i(t_{im}))^T$, $\widetilde{\boldsymbol{y}}_i = (\boldsymbol{y}_1, \cdots, \boldsymbol{y}_n)^T$, $\boldsymbol{\mu}_i = (\mu_i(t_{i1}), \cdots, \mu_i(t_{im})^T$, $\widetilde{\boldsymbol{\mu}} = (\boldsymbol{\mu}_i, \cdots, \boldsymbol{\mu}_n)^T$, $\boldsymbol{\alpha}_i = (\xi_{i1}, \cdots, \alpha_{iK})^T$, $\widetilde{\boldsymbol{\alpha}} = (\boldsymbol{\alpha}_1, \cdots, \boldsymbol{\alpha}_n)^T$, $\Lambda = diag(\lambda_1, \cdots, \lambda_K)$, $\boldsymbol{\xi}_{ik} = (\xi_k(t_{i1}), \cdots, \xi_k(t_{im}))$, $\boldsymbol{\xi}_i = (\boldsymbol{\xi}_{i1}, \cdots, \boldsymbol{\xi}_{iK})$ and $\widetilde{\boldsymbol{\xi}} = diag(\boldsymbol{\xi}_1, \cdots, \boldsymbol{\xi}_n)$. Then, the functional principal scores can be given as follows

$$\begin{aligned} \check{\widetilde{\boldsymbol{\alpha}}} &= \Sigma(\widetilde{\boldsymbol{\alpha}}, \widetilde{\boldsymbol{\alpha}})\widetilde{\xi}^T\left(\widetilde{\boldsymbol{\xi}}\Sigma(\widetilde{\boldsymbol{\alpha}}, \widetilde{\boldsymbol{\alpha}})\widetilde{\boldsymbol{\xi}}^T + \sigma^2 1\right)^{-1}(\widetilde{\boldsymbol{y}} - \widetilde{\boldsymbol{\mu}}), \\ &= \left(\sigma^2\Sigma(\widetilde{\boldsymbol{\alpha}}, \widetilde{\boldsymbol{\alpha}})^{-1} + \widetilde{\boldsymbol{\xi}}^T\widetilde{\boldsymbol{\xi}}\right)^{-1}\widetilde{\boldsymbol{\xi}}^T(\widetilde{\boldsymbol{y}} - \widetilde{\boldsymbol{\mu}}), \end{aligned} \tag{3.18}$$

where $\Sigma(\widetilde{\boldsymbol{\alpha}}, \widetilde{\boldsymbol{\alpha}})$ is the covariance of FPC scores. Using (3.2) this is given by,

$$\Sigma(\widetilde{\boldsymbol{\alpha}}, \widetilde{\boldsymbol{\alpha}}) = \begin{cases} \widetilde{\boldsymbol{\rho}}(1_{n\times n} \otimes \Lambda), & \text{non-separable,} \\ \boldsymbol{\rho} \otimes \Lambda & \text{separable,} \end{cases} \tag{3.19}$$

where $\widetilde{\boldsymbol{\rho}} = [\boldsymbol{\rho}_{ij}]$ and $\boldsymbol{\rho_{ij}} = diag(\rho^*_{ij1}, \cdots, \rho^* ijK)$. Then using (3.18) and (3.19) the FPC scores can be estimated as follows

$$\widehat{\widetilde{\alpha}} = \begin{bmatrix} \hat{\alpha}_1 \\ \vdots \\ \hat{\alpha}_n \end{bmatrix} = \begin{cases} \left(\hat{\sigma}^2\widehat{\widetilde{\boldsymbol{\rho}}}.(1_{n\times n} \otimes \widehat{\Lambda}) + \widehat{\widetilde{\boldsymbol{\xi}}}^T\widehat{\widetilde{\boldsymbol{\xi}}}\right)^{-1}\widehat{\widetilde{\boldsymbol{\xi}}}^T(\widetilde{\boldsymbol{y}} - \widehat{\widetilde{\boldsymbol{\mu}}}) & \text{non-separable} \\ \left(\hat{\sigma}^2\widehat{\boldsymbol{\rho}} \otimes \widehat{\Lambda} + \widehat{\widetilde{\boldsymbol{\xi}}}^T\widehat{\widetilde{\boldsymbol{\xi}}}\right)^{-1}\widehat{\widetilde{\boldsymbol{\xi}}}^T(\widetilde{\boldsymbol{y}} - \widehat{\widetilde{\boldsymbol{\mu}}}) & \text{separable} \end{cases} \tag{3.20}$$

Finally, it is possible to reconstruct the curves as

$$\widehat{\boldsymbol{x}}_i(t^{\text{eval}}) = \widehat{\boldsymbol{\mu}}_i(t^{\text{eval}}) + \widehat{\boldsymbol{\xi}}_i(t^{\text{eval}})\widehat{\boldsymbol{\alpha}}_i. \tag{3.21}$$

3.3.6 Consistency of estimates

SPACE approach is an extension of the method PACE proposed by Yao and Lee (2006), However, PACE assume that there is no spatial correlation in the data which is not the case in SPACE. To overcome this limitation, two conditions were introduced in SPACE and then theorem 3.1 is extended to the spatial correlation case.

Theorem 3.1. The uniform convergence rate of the cross covariance estimator is stated as

$$sup_{t,s\in\tau}|\hat{G}_{\Delta}(s,t) - G_{\Delta}(s,t)| = O_p\Big(\frac{1}{\sqrt{(|n(\Delta)h_G^2}}\Big),$$

where $\hat{G}_{\Delta}(s,t)$ is the smooth cross covariance estimates of $G(s,t) = Cov(x_i(s), x_j(t))$, $n(\Delta)$ represents the collection of the location pairs of the observations and h is the bandwidth. For more details regrading the prof and other theorems see the appendix in (Liu et al., 2017).

3.4 Spatio-temporal regression model with partial differential equations regularisation

This section describes the spatio-temporal regression model with partial differential equations regularisation (ST-PDE) approach, which was introduced by Bernardi et al. (2017). ST-PDE is a non-parametric method that deals with space-time data observed over non-planer spatial domains. The method focuses on surface estimation, considering the shape of the spatial domain combined with time evaluation. One of the major contributions of this thesis (see Chapter 4) is to develop a new framework, generalising the ST-PDE method, which will be capable of analysing replicated functional data obtained over non-planer spatial domains.

3.4.1 Functional data over complex domains

Functional data can be observed on irregularly shaped manifolds. These datasets might have complex boundaries and or interior gaps like the Montreal census data given in figure 3.3. The Montreal census data consist of 493 data points where each data point represents the average income for one area (Ramsay, 2002).

Figure 3.3 shows Montreal island with the data points, where the two internal gaps are the airport and factories and not included in the domain, as no people live there. When modelling this type of dataset, along with the complex external boundary shape of the island, we should also take into account the internal gaps where no data are collected for the variable of interest. It is quite challenging to model irregularly shaped data accommodating the complex domain.

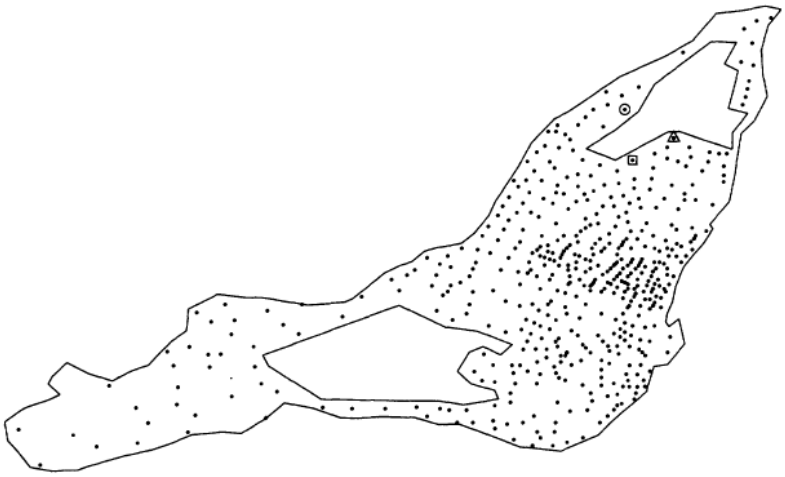

Figure 3.3: Montreal island with the data points (Ramsay, 2002)

Most of the existing classical approaches such as thin-plate splines, kernel smoothing, wavelet-based smoothing and kriging do not consider the shape of the spatial domain. For example, thin plate splines use roughness penalties that are based on integrated squared partial derivatives over the whole plane $\mathbb{R}^2$ and are not restricted to the domain of interest. One the other hand, kernel smoothing mostly uses Euclidean distance to measure the distance between data points, and it is well-known that Euclidean distance treats domains as connected, ignoring the holes and concave boundaries.

However, some recent methods for the analysis of space-time data have been designed to include the information of the domain of interest. For instance, Finite Element L-splines proposed by Ramsay (2002) present a penalised bivariate spline smoother. In this smoother, the roughness penalty consists of a partial differential operator and is integrated only over the region of interest using finite element analysis. Later, Wood et al. (2008) introduced a group of smoothers that consist of a low rank basis and a quadratic penalty. This approach can also accommodate irregularly shaped domain, as it does not smooth across boundaries.

The more recent approach of spatial spline regression (SSR) model, proposed by Sangalli et al. (2013), extends the finite element L-splines methods (Ramsay, 2002). The same smoother and penalty are used, however; the computational and modelling aspects were improved by Sangalli et al. (2013) method. Furthermore, this approach includes covariate estimation and more flexible boundary conditions. The SSR approach only models the spatial domain and do not include any temporal aspects in the model. Bernardi et al. (2017) extended the SSR to include the time component in the model by including two roughness penalties, one for space and another for time. The approach is named a spatio-temporal regression model with partial differential equations regularisation (ST-PDE) approach and is described in details in the next section.

3.4.2 The penalized model with partial differential regularisation

The smoothing function L-spline is a technique that was designed to smooth data observed in one dimensional space. Suppose y_i are set of observations represented by the smooth function $f(x_i)$ as follows;

$$y_i = f(x_i) + \epsilon_i,$$

Then, the L-spline function is the real-valued f that minimises the penalised sum of squares functional

$$\sum_{i=1}^{n}[y_i - f(x_i)]^2 + \lambda \int_b^a [Lf]^2 dx, \tag{3.22}$$

Where λ is the smoothing parameter, L is the linear differential operator and the integral is evaluated over the interval $[a, b]$ that includes all of x_i's. For more information of L-splines see (Wahba, 1990) and (Heckman and Ramsay, 2000).

Definition 3.4.1. The linear differential operator is a polynomial constructed from the differential operators $D^1, D^2, \cdots$. A differential operator of degree m can be written as:

$$L = D^m + \omega_{m-1}D^{m-1} + ... + \omega_1 D + \omega_0 I,$$

where I is the identity operator and ω_i are the coefficients.

The L-spline smoothing requires finding a solution that minimise (3.22). This problem can be solved by using Green's function (Green and Silverman, 1993b). A Green's function is the kernel of the integral operator inverse to the linear differential operator. In other words, the inverse of the linear differential operator L is an integral operator whose kernel function is the Green's function. We show later in this chapter how Green's theorem is used to find a function that minimises the penalised sum of square.

The L-spline smoothing function was generalised to the two dimensions case by Ramsay (2002) by introducing a penalised bivariate spline smoother . Let $\boldsymbol{p}_i = (x_i, y_i), i = 1, ..., n$ be a set of points on some domain $\Omega \in R^2$ and z_i be the data points that are observed at location $\boldsymbol{p}_i$.

$$z_i = f(p_i) + \epsilon_i,$$

The smooth estimate f of R^2 will be in this case a surface rather than a curve, and have to be estimated over a domain $\Omega \in R^2$ that consists of all the data location p_i. The bivariate L-spline is approximated by the function f that minimises the quantity

$$\sum_{i=1}^{n}[z_i - f(p_i)]^2 + \lambda \int_{\Omega} (L_p f)^2 d\Omega, \tag{3.23}$$

where L_p is a linear partial differential operator of order m.

Unlike the univariate L-spline, the bivariate L-spline is difficult to implement. The Green's functions that are used to solve the differential equations can not be defined easily in the two-dimensional case. Furthermore, the bivariate smoothing should not depend on the choice of the coordinate system. The roughness penalty in (3.23) should be invariant to rotation and translation. In this case, L_p should be defined by the Laplacian operator Δ. The Laplace operator is a second order differential operator that measure the curvature of some field. The Laplacian of function $f \in R^2$ is defined by the sum of the partial derivatives of the function f

$$\Delta f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}.$$

Thus, the linear differential operator L_p can be written as:

$$L_p = \Delta^p + c_{p-1}\Delta^{p-1} + ... + c_1\Delta + c_0 I,$$

for non-negative integer p and constant c. Then, the bivariate L-spline smoothing is approximated by the function f that minimises

$$\sum_{i=1}^{n}[z_i - f(p_i)]^2 + \lambda \int_{\Omega} (\Delta f)^2 d\Omega. \tag{3.24}$$

The minimisation problem in (3.24) includes only spatial aspects.

Bernardi et al. (2017) extended this smoother to space-time dependent data where both space and time components are included in the model. Suppose z_{ij} are data observed at a set of n spatial locations $\{p_i = (x_i, y_i); i = 1, ..., n\}$ on a bounded domain Ω, and over a set of m time points $\{t_j; j = 1, ..., m\}$ in time interval $[T_1, T_2] \subset R$. Bernardi et al. (2017) assumed that z_{ij} are noisy measurements of an underlying spatio-temporal smooth function $f(p, t)$ and thus can be written as,

$$z_{ij} = f(p_i, t_j) + \epsilon_{ij} \qquad i = 1, \cdots, n, \quad j = 1, \cdots, m,$$

where ϵ_{ij} are the error and independently distributed with mean zero and constant variance σ^2. Consequently, the minimiser considers two roughness penalties that allows one to impose regularity conditions on f separately in space and time. The temporal penalty is the classical penalty, the integral of the square of derivative d^r (Silverman and Ramsay, 2005). For any arbitrary function $h(t)$ The penalty is calculated as

$$J_T(h(t)) = \int_{T_1}^{T_2} \Big(\frac{d^r h(t)}{dt^r}\Big)^2 dt,$$

whereas the spatial penalty follows the penalty term in (3.24). Each penalty is applied to the function f and then integrated over the complementary domain. Then the minimiser can be written as;

$$\begin{aligned} J(f) = \sum_{i=1}^{n}\sum_{j=1}^{m}[z_{ij} - f(p_i, t_j)]^2 + \lambda_S \int_{T_1}^{T_2}\int_{\Omega} (\Delta f(p,t))^2 dpdt \\ +\lambda_T \int_{\Omega}\int_{T_1}^{T_2} (\frac{\partial^r f(p,t)}{\partial t^r})^2 dtdp, \end{aligned} \tag{3.25}$$

where λ_S and λ_T are the smoothing parameters that control the roughness in space and time respectively. The model (3.25) is the final model that is used to describe the data. However, in the next section we will show how the parameters of this model are estimated from the data.

3.4.3 Representing the spatio-temporal field

The model consists of three parts; the least square estimate, the spatial penalty and the temporal penalty. First, the least square part includes the data points z_{ij} and the spatio-temporal field f which is represented by the space and time basis functions. Suppose $\{\phi_k(t); k = 1, \cdots, M\}$ be a set of M basis functions defined over the time interval $[T_1, T_2]$ and $\{\Psi_l(p); l = 1, \cdots, N\}$ be a set of N basis functions defined on the space domain Ω. Then, under the assumption of separability the spatio-temporal field f can be written as;

$$f(p,t) = \sum_{l=1}^{N}\sum_{k=1}^{M} c_{lk}\Psi_l(p)\phi_k(t),$$

where c_{lk} are the coefficients of the spatio-temporal basis functions. The separability assumption can be implemented using a separable basis system, which in turn simplifies the estimation steps. Previous work on separating the spatial and temporal variation using the separability assumption can be found in Liu et al. (2017) and Aston et al. (2015). The previous authors have also provided tests of the separability assumption using parametric and non-parametric methods, respectively.

In the ST-PDE approach, a cubic B-spline basis is used as the temporal basis and the penalty is represented by the second derivative of the basis functions. For the spatial part, the most appropriate basis for the irregular domain is the finite element basis used in (Sangalli et al., 2013). The idea of finite element analysis is to choose a number of piece-wise polynomials defined over sub-regions and the sum of the solutions of these sub-regions provides an approximate solution to the entire domain. More information of finite element methods will be provided in the next section.

Finite element analysis

Finite elements analysis is a numerical method that appeared first in the later part of 1950 where the goal was to solve complex equations that were difficult to solve analytically, such as partial differential equations. The idea of finite element analysis (FEA) is to divide the given domain into small sub-domains referred to as the finite elements. Then each sub-domain (finite element) is modelled by a polynomial and the sum of the solutions of these sub-domains provides an approximate solution to the entire problem. The domain can be divided using triangular or quadrilateral mesh. Sangalli et al. (2013) and Bernardi et al. (2017) used the triangular mesh τ, to represent the spatial domain Ω. The approximated domain is denoted by Ω_τ. The process of dividing the domain into triangles is called triangulation.

Definition 3.4.2. The triangulation τ of the domain Ω is a partition $\overline{\Omega}$ into a finite number of non-overlapping triangles K_i such that

- $K_i \cap K_j = \phi \quad if \quad i \neq j.$
- $\cup \overline{K_i} = \overline{\Omega}.$

where K_i's are called finite elements.

There are different methods of triangulations, one of the most popular methods is called Delaunay triangulation. In this methods, the data points are used as vertices of the triangle. In other words, the circumcircle of every triangle is empty, that is, there is no point from the data in its interior. This can be achieved by maximising the smallest angle over all triangulations of a given point set. Then, the triangulation will be finer in the area where there are more data points and coarser in the area with sparse data points. The Delanuay triangulation will be used in this approach, however, the triangulation can be done using existing triangulation software. Figure 3.4 shows a triangular mesh for the Montreal dataset where the triangle's vertices are the data points.

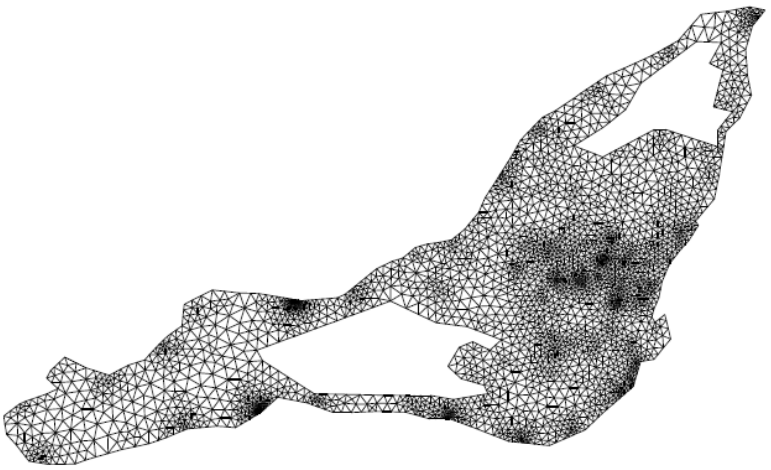

Figure 3.4: Example of triangulation mesh of the Montreal island (Ramsay, 2002)

Once the triangulation is done the domain will be divided into sub-domains (finite elements). Each finite element consists of a triangular domain, a set of nodes and an associated set of nodal basis functions. The basis functions are chosen to be polynomials of low degree. However, the polynomial can be either linear or quadratic. In the linear case, only the triangle vertices are used as nodes, and the polynomial is defined by three basis functions. On the other hand, the quadratic polynomial uses six basis functions which are associated with six nodes, the vertices and the midpoints on the edges of the triangle. In both cases, the basis function take the value 1 at a single node and zero on the others. Let ψ_k be the basis function

for a triangle at node k then the basis functions are defined as:

$$\psi_k(n_l) = \begin{cases} 1 & k = l \\ 0 & k \neq l. \end{cases}$$

Then any function f in the domain Ω can be defined as follow:

$$f(x,y) = \sum_{k=1}^{K} c_k \psi_k(x,y) = \sum_{k=1}^{K} f(n_k)\psi_k(x,y) = \boldsymbol{f}^T \psi(x,y),$$

where $f = (f(n_1), \cdots, f(n_k))^T$ which indicates that f is defined by its value at the K nodes in the finite element space. For more information of finite element methods see (Brenner and Scott, 2007) and (Braess, 2007).

The estimation in variational form

In order to use finite element analysis, it is required to define a variational formulation of the partial differential equation.

Suppose $H^m(\Omega)$ consists of all continuous functions of the domain Ω in $L^2(\Omega)$ having mth order partial derivatives. The normal derivatives of $H^m(\Omega)$ are equal to zero on the boundary of the domain and indicated by $H_0^m(\Omega)$. The spatial penalty function in (3.25) is uniquely defined in $H^2(\Omega)$.

Sangalli et al. (2013) proved that the minimiser has a unique solution which satisfies the boundary condition $f \in H_{n0}^2(\Omega)$; which assumes zero flow on the boundaries of the domain. Then, the minimiser problem can be defined for $f \in H_{n0}^2(\Omega)$ and the estimator f that minimise the model is given by:

$$u_n^T Q \hat{f}_n + \lambda \int_\Omega \Delta u \Delta \hat{f} = u_n^T Q z, \tag{3.26}$$

for every $u \in H_{n0}^2(\Omega)$.

The formulation (3.26) can only be defined in $H^2(\Omega)$. We need to transform this equation to be well defined in $H^1(\Omega)$ and thus can be solved using the finite element method.

3.4.4 Finite Element Solution

The problem of finding $\hat{f} \in H^2_{n0}(\Omega)$ in equation (3.26) can be solved by introducing an auxiliary function $g = \Delta f$. Then the model can be written as the problem of finding $(f, g) \in H^2_{n0}(\Omega) \times L^2(\Omega)$ that satisfies;

$$\begin{aligned} u_n^T Q \hat{f}_n + \lambda \int_\Omega g(\Delta u) &= u_n^T Q z \\ \int_\Omega gv - \int_\Omega (\Delta \hat{f}) v &= 0, \end{aligned} \tag{3.27}$$

for all $(u, v) \in H^2_{n0}(\Omega) \times L^2(\Omega)$. Over a region Ω in the plane with boundary $\partial\Omega$, Green's theorem states

$$\int_\Omega u \partial_i v = \int_{\partial\Omega} u \nu v_i - \int_\Omega \nu \partial_i u.$$

Based on this definition one can write

$$\begin{aligned} \int_\Omega g(\Delta u) &= -\int_\Omega (\nabla g . \nabla u) + \int_{\partial\Omega} g(\partial_\nu u) \\ \int_\Omega (\Delta \hat{f}) v &= -\int_\Omega (\nabla v . \nabla \hat{f}) + \int_{\partial\Omega} v(\partial_\nu \hat{f}), \end{aligned}$$

where $\int_{\partial\Omega} g(\partial_\nu u) = 0$ and $\int_{\partial\Omega} v(\partial_\nu \hat{f}) = 0$ due to the boundary conditions, i.e. the normal derivatives of f and u equal to zero.

Then equation (3.27) can be written as a problem of finding $(\hat{f}, g) \in \{H^1_{n0}(\Omega) \times C^0(\Omega)\} \times H^1(\Omega)$ that satisfies

$$\begin{aligned} u_n^T Q \hat{f}_n - \lambda \int_\Omega (\nabla u . \nabla g) &= u_n^T Q z, \\ \int_\Omega vg + \int_\Omega (\nabla v . \nabla \hat{f}) &= 0. \end{aligned} \tag{3.28}$$

The system in (3.28) is the finite element solution to the estimation problem for the model with the spatial penalty only. Since the equations are represented in $H^1(\Omega)$ and the function can be estimated for each triangle by the polynomial on the nodes. Then, the entire problem can be solved as a linear system of equations and be represented in a simple matrix equation.

As we mentioned in Section 3.4.3 every function in the finite element space is defined by its value at the nodes for example $f(x, y) = f^T \psi(x, y)$. Then the system

of equations in (3.28) can be written as follows;

$$u_n^T Q\hat{f}_n - \lambda \int_\Omega u^T(\psi_x\psi_x^T + \psi_y\psi_y^T)g = u_n^T Qz,$$
$$\int_\Omega v^T(\psi\psi^T)g + \int_\Omega v^T(\psi_x\psi_x^T + \psi_y\psi_y^T)\hat{f} = 0,$$

where ψ is a vector of space basis function and ψ_x and ψ_y are the vectors of first order partial derivatives of ψ. The problem now is reformulated to be a problem in $R^n \times R^n$.

By linearity, the coefficient vectors u and v can be taken out of the integral;

$$u_n^T Q\hat{f}_n - \lambda u^T \int_\Omega (\psi_x\psi_x^T + \psi_y\psi_y^T)g = u_n^T Qz,$$
$$v^T \int_\Omega (\psi\psi^T)g + v^T \int_\Omega (\psi_x\psi_x^T + \psi_y\psi_y^T)\hat{f} = 0.$$

Moreover, let L be a block matrix and D be a $K \times n$ block matrix defined by;

$$L := \left[\begin{array}{c|c} Q & O_{n\times(K-n)} \\ \hline O_{n\times(K-n)} & O_{(K-n)\times(K-n)} \end{array}\right],$$

$$D := \begin{bmatrix} I_n \\ O_{(K-n)\times n} \end{bmatrix},$$

where O is a $m_1 \times m_2$ with all entries equal to zero. Let set two matrices R_0 and R_1 which are given by

$$R_0 = \int_\Omega \psi\psi^T,$$
$$R_1 = \int_\Omega (\psi_x\psi_x^T + \psi_y\psi_y^T).$$

By plugging these matrices to the linear system it can be written as follows;

$$\begin{aligned} u_n^T L\hat{f}_n - \lambda u^T R_1 g &= u_n^T LDz, \\ v^T R_0 g + v^T R_1 \hat{f} &= 0. \end{aligned} \tag{3.29}$$

from the system of equations in 3.29 we can obtain $g = -R_0^{-1}R_1 f$ and $\hat{f} = (L + \lambda R_1 R_0^{-1} R_1)^{-1} LDz$. However, the two quantities R_0 and $L+\lambda R_1 R_0^{-1} R_1$ are invertible and positive definite

$$\begin{aligned} -Lf + \lambda R_1 g &= -LDz, \\ R_0 g + R_1 f &= 0, \end{aligned} \tag{3.30}$$

and then we can write

$$L(Dz - f) + \lambda R_1 R_0^{-1} R_1 f = 0.$$

Denoting $P_S = R_1 R_0^{-1} R_1$ we can see that the penalty term is equivalent to the spatial penalty matrix (Azzimonti et al., 2015).

3.4.5 Defining the penalised sum of squares

The penalised sum of square error functional in model (3.25) can be described numerically by the data and the basis functions with their penalties. Let $\mathbf{z}$ be the observations represented as vector of length nm with n spatial locations and m time points, $\mathbf{f}$ the evaluation of the spatio-temporal function $f(p_n, t_m)$, and $\mathbf{c}$ the basis coefficients vector of length NM.

$$\mathbf{z} = \begin{bmatrix} z_{11} \\ \vdots \\ z_{1m} \\ z_{21} \\ \vdots \\ z_{2m} \\ \vdots \\ z_{nm} \end{bmatrix}, \quad \mathbf{f} = \begin{bmatrix} f(p_1, t_1) \\ \vdots \\ f(p_1, t_m) \\ f(p_2, t_1) \\ \vdots \\ f(p_2, t_m) \\ \vdots \\ f(p_n, t_m) \end{bmatrix}, \quad \mathbf{c} = \begin{bmatrix} c_{11} \\ \vdots \\ c_{1M} \\ c_{21} \\ \vdots \\ c_{2M} \\ \vdots \\ c_{NM} \end{bmatrix}.$$

Let $\boldsymbol{\psi}$ be vector of spatial basis functions with length N and their first partial derivatives ψ_x and ψ_y are given by

$$\boldsymbol{\psi} = \begin{bmatrix} \psi_1(p) \\ \psi_2(p) \\ \vdots \\ \psi_N(p) \end{bmatrix}, \quad \boldsymbol{\psi_x} = \begin{bmatrix} \partial\psi_1(p)/\partial x \\ \partial\psi_2(p)/\partial x \\ \vdots \\ \partial\psi_N(p)/\partial x \end{bmatrix}, \quad \boldsymbol{\psi_y} = \begin{bmatrix} \partial\psi_1(p)/\partial y \\ \partial\psi_2(p)/\partial y \\ \vdots \\ \partial\psi_N(p)/\partial y \end{bmatrix}.$$

Then, the evaluation of the N basis functions at the n spatial points can be organised into the matrix

$$\Psi_{n\times N} = \begin{bmatrix} \psi_1(p_1) & \psi_2(p_1) & \cdots & \psi_N(p_1) \\ \psi_1(p_2) & \psi_2(p_2) & \cdots & \psi_N(p_2) \\ \vdots & \vdots & \cdots & \vdots \\ \psi_1(p_n) & \psi_2(p_n) & \cdots & \psi_N(p_n) \end{bmatrix}.$$

For temporal dimension, let $\boldsymbol{\phi}$ be vector of temporal basis functions of length M and their second derivatives φ_{tt}, which are given by

$$\boldsymbol{\varphi} = \begin{bmatrix} \varphi_1(t) \\ \varphi_2(t) \\ \vdots \\ \varphi_M(t) \end{bmatrix}, \boldsymbol{\varphi}_{tt} = \begin{bmatrix} d^2\varphi_1(t)/dt^2 \\ d^2\varphi_2(t)/dt^2 \\ \vdots \\ d^2\varphi_M(t)/dt^2 \end{bmatrix}.$$

Then the evaluation of the M basis functions at the m time points can be organised as

$$\Phi_{m\times M} = \begin{bmatrix} \varphi_1(t_1) & \varphi_2(t_1) & \cdots & \varphi_M(t_1) \\ \varphi_1(t_2) & \varphi_2(t_2) & \cdots & \varphi_M(t_2) \\ \vdots & \vdots & \cdots & \vdots \\ \varphi_1(t_m) & \varphi_2(t_m) & \cdots & \varphi_M(t_m) \end{bmatrix}.$$

Then, K_0 is $M \times M$ matrix defined by the integral of the cross products of the temporal basis.

$$K_0 = \int_{T_1}^{T_2} \varphi\varphi^T$$

The penalised sum of squares can now be denoted using the matrices defined in the previous sub-sections. let $B = \Psi \otimes \Phi$ where $\otimes$ is the Kronecker product which is the direct product of Ψ and Φ resulting in an $nm \times NM$ matrix B. Then $\mathbf{f}$ can be written as $\mathbf{f} = B\mathbf{c}$ and the sum of square error functional can be defined as follows,

$$(\mathbf{z} - B\mathbf{c})^T(\mathbf{z} - B\mathbf{c}).$$

The spatial penalty term is given by

$$\lambda_S \mathbf{c}^T(P_S \otimes K_0)\mathbf{c},$$

where P_S is the spatial penalty and is defined by the discretisation $P_S = R_1 R_0^{-1} R_1$. While the temporal penalty term is given by

$$\lambda_T \mathbf{c}^T (R_0 \otimes P_T)\mathbf{c},$$

where P_T is the time penalty which is the second derivative of the temporal basis functions $P_T = \int_{T_1}^{T_2} \phi_{tt}\phi_{tt}^T$. Putting these terms together we get

$$\begin{aligned} J = (\mathbf{z} - B\mathbf{c})^T(\mathbf{z} - B\mathbf{c}) + \lambda_s \mathbf{c}^T(P_S \otimes K_0)\mathbf{c} + \lambda_T \mathbf{c}^T(R_0 \otimes P_T)\mathbf{c} \\ = (\mathbf{z} - B\mathbf{c})^T(\mathbf{z} - B\mathbf{c}) + \mathbf{c}^T P\mathbf{c}, \end{aligned} \tag{3.31}$$

where P represent the overall penalty $P = \lambda_S(P_S \otimes K_0) + \lambda_T(R_0 \otimes P_T)$.

Once we have the matrix representation of the optimisation problem, the coefficients vector $\mathbf{c}$ can be obtained from model (3.31) as a solution of the penalised least square and is given by

$$\hat{\mathbf{c}} = (B^T B + P)^{-1} B^T \mathbf{z}.$$

3.4.6 Properties of the estimator

The mean and the variance of the coefficients vector are given by

$$E[\hat{\mathbf{c}}] = (B^T B + P)^{-1} B^T f$$

$$Var[\hat{\mathbf{c}}] = \sigma^2 (B^T B + P)^{-1} B^T B (B^T B + P)^{-1}$$

where σ^2 is the constant variance of the error term. The evaluation of the separable basis function at the spatio-temporal points (p, t) is given by the vector $B(p, t) = \psi(p)^T \otimes \phi(t)^T$. Furthermore, the estimated value of the spatio-temporal field $\mathbf{f}$ at any spatio-temporal location is given by

$$\hat{\mathbf{f}}(p, t) = B(p, t)\hat{\mathbf{c}} = B(p, t)(B^T B + P)^{-1} B^T \mathbf{z}.$$

Then, we can obtain the mean and the variance as follows

$$E[\hat{\mathbf{f}}(p, t)] = B(p, t)(B^T B + P)^{-1} B^T \mathbf{f}$$

$$Var[\hat{\mathbf{f}}(p, t)] = \sigma^2 B(p, t)(B^T B + P)^{-1} B^T B (B^T B + P)^{-1} B(p, t)^T$$

Furthermore, the covariance between two spatio-temporal locations is given by

$$Cov[\hat{\mathbf{f}}(p_1,t_1),\hat{\mathbf{f}}(p_2,t_2)] = \sigma^2 B(p_1,t_1)(B^TB+P)^{-1}B^TB(B^TB+P)^{-1}B(p_2,t_2)^T$$

.

Note that σ^2 is unknown and can be estimated from the data. Consider the vector $\hat{\mathbf{z}}$ of the fitted values at the space-time points $\hat{\mathbf{z}} = S\mathbf{z}$ where S is the smoothing matrix $S = B(B^TB+P)^{-1}B^T$.

σ^2 can be estimated by

$$\hat{\sigma^2} = \frac{1}{nm - tr(S)}(\mathbf{z}-\hat{\mathbf{z}})^T(\mathbf{z}-\hat{\mathbf{z}})$$

3.5 Summary

In this chapter we have reviewed two existing methods, SPACE and ST-PDE which can model specific types of spatio-temporal data. We have also clearly identified the limitations of each of these methods and pointed out that none of these methods are designed to analyse replicated spatio-temporal data such as the EEG data described in Chapter 6. So, in the next chapter we propose to extend the ST-PDE approach to analyse replicated functional data, and provide the corresponding theoretical results and computational tools for the new framework. We will then use the new method to analyse the EEG data in Chapter 6.

Chapter 4

Modeling Replicated Functional Data

4.1 Introduction

The functional data analysis approaches introduced in Chapter 3 are primarily designed to model space-time data, where we have only one curve for each location. However, in many applications we need to analyse replicated space-time data, which involve repeated measurements of the same process at the same location. In this chapter, we extend the ST-PDE approach introduced in Chapter 3 to accommodate replicated functional data. The first part of the chapter presents a motivational application which deal with the analysis of replicated brain EEG measurements on 18 subjects. The rest of the chapter focuses on developing the new framework of analysing replicated spatio-temporal data by extending the ST-PDE methodology. **Note:** This chapter is adopted from: Alghamdi,S. and S.Ray. Analysis of replicated spatially correlated functional data (2019). (Under preparation)

4.2 Motivating Application

Our motivating application concerns brain data that measure and record the electrical activity of the brain, over time and across several electrodes. There are different noninvasive techniques that are used to record brain activities such as functional Magnetic Resonance Imaging (fMRI), magnetoencephalography (MEG) and elec-

troencephalography (EEG). Our data records the EEG. First we provide a brief discussion of the structure of the human brain and the function of each part of the brain. Then we provide more information of electroencephalography (EEG) and introduce our data set.

4.2.1 Human Brain

The brain is the command centre of the human body and the main organ in the nervous system. It controls most of the body tasks and activities by sending the instructions to the body and receiving the information from the sense organs. The brain consists of three parts: cerebrum, cerebellum and brain-stem. However, the cerebrum is the largest and most important part of the brain. It consists of two hemispheres, right and left and each of them is divided into four lobes: the frontal, temporal, occipital and parietal. Figure 4.1 shows how the these lobes are distributed in the brain.

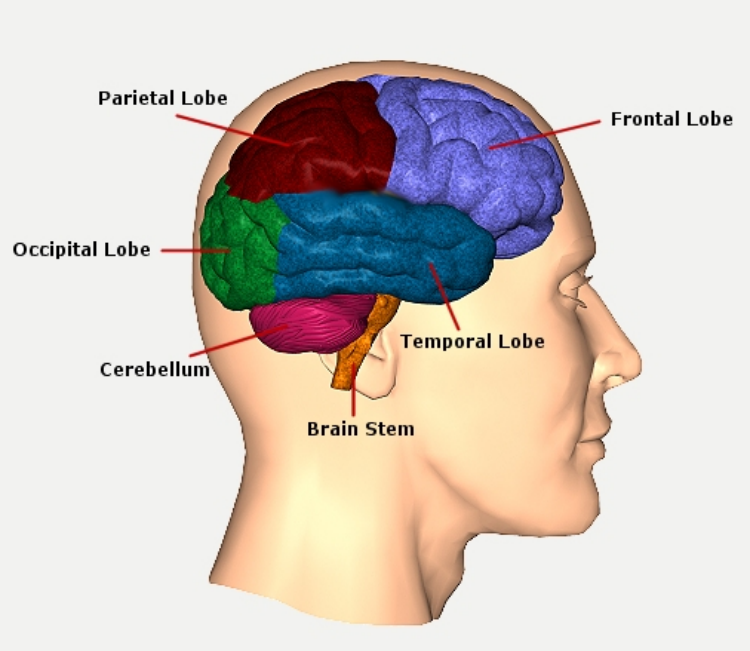

Figure 4.1: Lobes locations in the brain

- Frontal Lobe: It is positioned at the front of the brain and is associated with functions such as self-controlling, problem solving, planning, and social behaviour.

- Temporal Lobe: is located at the side of the head above the ears and it is responsible for functions include auditory perception, long-term memory and speech understanding.
- Occipital Lobe: is located at the back of the brain and responsible for visual perception system.
- Parietal Lobe: is located at the top and the back of the head. It is involved in sensation such as touch, pain, etc.

The brain is composed of billions of nerves cell which receive, process and send information via electrical signals. These electrical activities can be measured using different noninvasive techniques such as functional Magnetic Resonance Imaging (fMRI), magnetoencephalography (MEG) and electroencephalography (EEG). EEG is a monitoring approach that records brain waves. As our dataset consists of EEG measurements we provide more information on EEG in the next section.

4.2.2 Electroencephalography (EEG)

Electroencephalography is a monitoring technique that is used to measure and record the electrical activity of the brain. EEG measures voltage variation, which arises from ionic flow within the neural activity in the brain (Niedermeyer and da Silva, 2005). The technique of EEG was first used in 1875, when Richard Carton succeed in recording the electrical signals from the brains of monkeys and rabbits. In 1924, Hans Berger succeed in applying the technique to recording signals from human brains using a device called an electroencephalograph (Haas, 2003). Berger gathered hundreds of EEG measurements from different people and he suggested that these measurements changed with the psychological state of the subject. Apart from monitoring brain activities, EEG is used in the diagnosis of several types of neurological disorders such as epilepsy, Parkinson's disease and brain tumours. EEG has many advantages compared to other techniques as it is fast and can record the brain signals in milliseconds. Also it is very safe as it is only records the activities that the brain already produces. But one clear drawback is that EEG provides poor spatial

resolution which indicates that it might be less informative about active areas of the brain if the nodes are not placed in those regions.

EEG can be measured directly from the head surface by applying electrodes (small discs) on the surface of the scalp. These electrodes are connected to amplifiers and the amplified signals are converted to digital via an anti-aliasing filter. These signals represent the EEG readings. The locations of the electrodes are determined by the international 10-20 or 10-10 placement system, which contain 21 and 64 electrodes placement, respectively (Klem et al., 1999) . The "10" and "20" indicates the distances between the neighbouring electrodes which are either 10 % or 20 % of the total front-back or right-left distance of the skull.

This system is used internationally and was adopted by the International Federation in Electroencephalography and Clinical Neurophysiology. Figure 4.2, which is reproduced from Smolka et al. (2015), shows the electrodes locations specified by the 10-10 placement system.

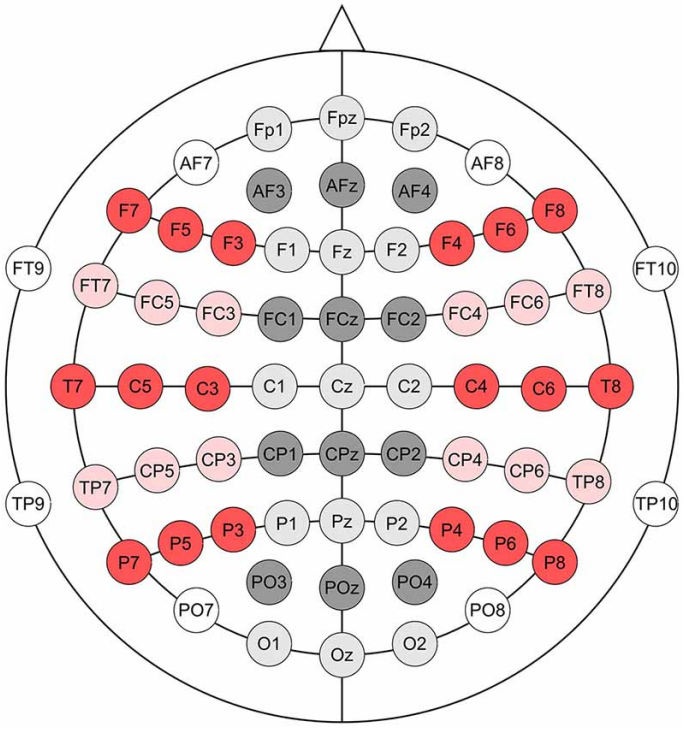

Figure 4.2: Electrodes locations on the surface of the brain with the 10-10 International Electrode Placement System

The head is divided into five main areas F, T, C, P and O which refer to frontal,

temporal, central, partial and occipital, respectively. The even numbers indicate the electrodes distributed over the right side of the head and the odd numbers denote the left side.

4.2.3 Data Description

Now we describe the data we will analyse later in Chapter 6. The data consist of EEG measurements from 18 subjects. Each subject was shown a stimulus, which is a series of 250 pictures, 125 of them being image of cars and the other being image of faces. Subjects performed a categorisation task, which is classifying the presented image. The EEG data were recorded with 57 scalp electrodes located according to the international 10-10 placement system and each record consists of 454 time points. The EEG signals were recorded 200 milliseconds before the subject was shown the stimulus and 500 milliseconds after the stimulus was presented. Figure 4.3 shows the data from one subject observed at one electrode for both stimuli images of car and face where the individual curves represent the replications. The data set is described in more details in Chapter 6. The data is time synchronised so we do not perform any additional egistration steps.

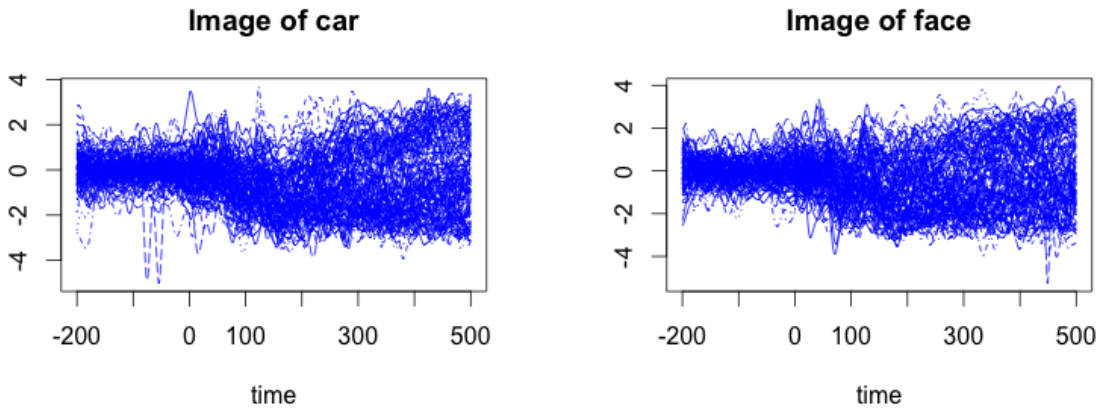

Figure 4.3: EEG measurements of one electrode for one subject seeing images of car and face

This data can't be analysed using the existing techniques of SPACE or ST-PDE due to the structure of the data. This motivates us to extend the ST-PDE

to the new framework replicated ST-PDE. Our approach naturally accommodates replicated data observed over space and time. In the next section we describe the details and mathematical derivations of the replicated ST-PDE model.

4.3 Replicated ST-PDE Model

Suppose z_{ijk} are data observed at a set of n spatial locations $\{p_i = (x_1, y_1), \cdots, (x_n, y_n)\}$ on a bounded domain Ω, and over a set of m time points $\{t_j; j = 1, ..., m\}$ in the time interval $[T_1, T_2] \subset R$ where each of these observations is repeated $\{k = 1, \cdots, l\}$ times. Then, these data are assumed to be noisy measurements that are sampled from the smooth function $f(p, t)$ and can be written as,

$$z_{ijk} = f(p_i, t_j) + \epsilon_{ijk} \quad i = 1, \cdots, n, j = 1, \cdots, m, k = 1, \cdots, l \tag{4.1}$$

where ϵ_{ijk} are the errors which are independently distributed with mean zero and constant variance.

Similar to the ST-PDE framework, the spatio-temporal function $f(s, t)$ is estimated by minimising the sum of squared errors. Recall that, the minimiser is controlled using two separate roughness penalties that allow the regularity of f in space and time. The penalties used in this approach are the same as the ones in (Bernardi et al., 2017). The main change from (Bernardi et al., 2017) is in estimating the penalised sum of square error function that was presented in (3.31). The difficulties and challenges of extending this method is discussed later in Section 4.5.

Remark. The replications in this extension are included as independent replications which might not be true as they correspond to the same subject. However, if we ignore within subject dependence, the expected value of the estimated regression coefficients will remain same. But ignoring the dependence might give us wrong estimation for the standard errors. But as we are mainly dealing with the means of the processes rather than the variances we continue with the independence assumption. If one wishes to account for the within-subject dependence then a mixed model might be more appropriate as it can build the correlation over different layers, but a mixed model on functional data is very complex and computationally intensive.

In the next section we will introduce the estimation steps for the RST-PDE model.

4.3.1 Notational details

We will first represent the model in (4.1) in vector matrix notation using Kronecker products.

$$\mathbf{z^*}_{(nml)} = \mathbf{f^*}_{(nml)} + \boldsymbol{\epsilon^*}_{(nml)} \tag{4.2}$$

$$\mathbf{z^*} = \begin{bmatrix} z_{111} \\ \vdots \\ z_{1m1} \\ z_{211} \\ \vdots \\ z_{nm1} \end{bmatrix} \text{replcation 1} \quad \begin{bmatrix} z_{112} \\ \vdots \\ z_{nm2} \end{bmatrix} \text{replcation 2} \quad \begin{bmatrix} \vdots \end{bmatrix} \quad \begin{bmatrix} z_{11l} \\ \vdots \\ z_{nml} \end{bmatrix} \text{replcation } l$$

$$\mathbf{f^*} = \begin{bmatrix} f(p_1,t_1) \\ \vdots \\ f(p_1,t_m) \\ f(p_2,t_1) \\ \vdots \\ f(p_n,t_m) \end{bmatrix} \text{replcation 1} \quad . \begin{bmatrix} f(p_1,t_1) \\ \vdots \\ f(p_n,t_m) \end{bmatrix} \text{replcation 2} \quad \begin{bmatrix} \vdots \end{bmatrix} \quad \begin{bmatrix} f(p_1,t_1) \\ \vdots \\ f(p_n,t_m) \end{bmatrix} \text{replcation } l$$

$$\boldsymbol{\epsilon^*} = \begin{bmatrix} \epsilon_{111} \\ \vdots \\ \epsilon_{1m1} \\ \epsilon_{211} \\ \vdots \\ \epsilon_{nm1} \end{bmatrix} \text{replcation 1} \quad \begin{bmatrix} \epsilon_{112} \\ \vdots \\ \epsilon_{nm2} \end{bmatrix} \text{replcation 2} \quad \begin{bmatrix} \vdots \end{bmatrix} \quad \begin{bmatrix} \epsilon_{11l} \\ \vdots \\ \epsilon_{nml} \end{bmatrix} \text{replcation } l$$

Where $\boldsymbol{\epsilon^*}$ is the error vector and $\mathbf{f^*}$ has the entries of $\mathbf{f}$ repeated l times.

Let us now define $r = \{1, 1, \cdots, 1\}$ to be the vector of ones with length equal to the number of replications. We can use the Kronecker product of $\boldsymbol{r}$ and $\mathbf{f}_{(nm)}$ to represent the vector $\mathbf{f}^*$.

Corollary 4.1. Using the vector of l ones denoted by r_l, equation (4.2) can be re-stated as

$$\mathbf{z^*}_{(nml)} = \boldsymbol{r}_{(l)} \otimes \mathbf{f}_{(nm)} + \boldsymbol{\epsilon^*}_{(nml)} \tag{4.3}$$

Proof. It is easy to show that by the Kroncker product rules

$$\boldsymbol{r}_l \otimes \mathbf{f}_{(nm)} = \begin{bmatrix} \mathbf{f}_{(nm)} \\ \mathbf{f}_{(nm)} \\ \vdots \\ \mathbf{f}_{(nm)} \end{bmatrix} = \mathbf{f^*}_{(nml)}$$

□

Corollary 4.2. Re-introducing the basis expansion $\mathbf{f}$ as $\mathbf{f} = B \otimes \mathbf{c}$ we can rewrite the model in (4.3) as

$$\mathbf{z^*}_{(nml)} = \boldsymbol{r}_{(l)} \otimes B_{(nm \times NM)} \mathbf{c}_{(NM)} + \boldsymbol{\epsilon^*}_{(nml)}, \tag{4.4}$$

where $B = \Psi \otimes \Phi$ and $\mathbf{c}$ is the vector of coefficients and with basis function expansion following Section 3.4.5. The evaluation of the N basis functions at the n spatial points is given by

$$\Psi_{(n \times N)} = \begin{bmatrix} \psi_1(p_1) & \psi_2(p_1) & \cdots & \psi_N(p_1) \\ \psi_1(p_2) & \psi_2(p_2) & \cdots & \psi_N(p_2) \\ \vdots & \vdots & \cdots & \vdots \\ \psi_1(p_n) & \psi_2(p_n) & \cdots & \psi_N(p_n) \end{bmatrix}.$$

For the temporal dimension, the evaluation of the M basis functions at the m time points is given by

$$\Phi_{(m\times M)} = \begin{bmatrix} \varphi_1(t_1) & \varphi_2(t_1) & \cdots & \varphi_M(t_1) \\ \varphi_1(t_2) & \varphi_2(t_2) & \cdots & \varphi_M(t_2) \\ \vdots & \vdots & \cdots & \vdots \\ \varphi_1(t_m) & \varphi_2(t_m) & \cdots & \varphi_M(t_m) \end{bmatrix}.$$

Proof. As $\mathbf{f}_{(nm)}$ can be written as $B\mathbf{c}$, replacing $\mathbf{f}$ in corollary 4.1 gives us

$$\mathbf{z^*}_{(nml)} = \boldsymbol{r}_{(l)} \otimes B_{(nm\times NM)}\mathbf{c}_{(NM)} + \boldsymbol{\epsilon}_{(nml)}$$

we can also check that the dimension of the left hand side and right hand side both equal nml. □

Before solving the penalised estimator, we will first work out the least square estimator (unpenalised) for the coefficient vector $\mathbf{c}_u$

Theorem 4.1. The sum of square for the estimation of $\mathbf{c}_u$ in

$$\mathbf{z^*}_{(nml)} = \boldsymbol{r}_{(l)} \otimes B_{(nm\times NM)}\mathbf{c}_{u_{(NM)}} + \boldsymbol{\epsilon}^*_{(nml)}$$

can be written as

$$(\mathbf{z}^* - \boldsymbol{r} \otimes B\mathbf{c})^T(\mathbf{z}^* - \boldsymbol{r} \otimes B\mathbf{c})$$

and the least squares solution of the model can be compactly written as

$$\hat{\mathbf{c}}_{u_{(NM)}} = \frac{1}{l}(B^TB)^{-1}(\boldsymbol{r}^T \otimes B^T)\mathbf{z}^*$$

Proof. Define a new matrix A that contains evaluation of the basis functions for all replications which is given by

$$A_{(nml\times NM)} = \boldsymbol{r}_{(l)} \otimes B_{(nm\times NM)}$$

Equation (4.4) can be restricted as $\mathbf{z}^* = A\mathbf{c} + \boldsymbol{\epsilon}^*$. Then the sum of squares is now given by

$$(\mathbf{z}^* - A\mathbf{c}_u)^T(\mathbf{z}^* - A\mathbf{c}_u)$$

Now re-introducing $A = \boldsymbol{r} \otimes B$ in terms of B matrix the least square can be written as,

$$(\mathbf{z}^* - \boldsymbol{r} \otimes B\mathbf{c}_u)^T(\mathbf{z}^* - \boldsymbol{r} \otimes B\mathbf{c}_u)$$

Solving the least square problem in A we get

$$(\mathbf{z}^* - A\mathbf{c}_u)^T(\mathbf{z}^* - A\mathbf{c}_u) = \mathbf{z}^{*T}\mathbf{z}^* - \mathbf{z}^{*T}(A\mathbf{c}_u) - (A\mathbf{c}_u)^T\mathbf{z}^* + (A\mathbf{c}_u)^T A\mathbf{c}_u$$

$$= \mathbf{z}^{*T}\mathbf{z}^* - 2\mathbf{z}^*\mathbf{c}_u^T A^T + \mathbf{c}_u^T A^T A\mathbf{c}_u$$

To determine the vector, $\hat{\mathbf{c}}_u$, we minimize the sum of squared with respect to the $\mathbf{c}_u$ and set it equal to zero we get

$$-2A^T\mathbf{z}^* + 2A^T A\hat{\mathbf{c}}_u = 0$$

$$\mathbf{z}^* A^T = A^T A\hat{\mathbf{c}}_u$$

$$\hat{\mathbf{c}}_u = (A^T A)^{-1}A^T\mathbf{z}^*$$

We wish to express the estimate in terms of B as follows

$$\hat{\mathbf{c}}_u = ((\boldsymbol{r} \otimes B)^T(\boldsymbol{r} \otimes B))^{-1}(\boldsymbol{r} \otimes B)^T\mathbf{z}^*$$

$$\hat{\mathbf{c}}_u = (\boldsymbol{r}^T\boldsymbol{r} \otimes B^T B)^{-1}(\boldsymbol{r}^T \otimes B^T)\mathbf{z}^*,$$

$$\hat{\mathbf{c}}_{u_{(NM)}} = \frac{1}{l}(B^T B)^{-1}(\boldsymbol{r}^T \otimes B^T)\mathbf{z}^*$$

where $\boldsymbol{r}^T\boldsymbol{r}$ is a scalar and B is a matrix of dimension nm which give us a huge computational advantage. □

Note that, if we work directly with the A matrix we need to calculate $A^T A$ which has a huge computational cost multiplying two matrices of dimension (nml). For example, in our brain data $nml = 454 \times 57 \times 125 = (3,234,750)$ for a single individual seeing one type of images. So we express the estimate in terms of B, which is of dimension nm, to overcome this issue.

Corollary 4.3. The sum of squares for the estimation of $\mathbf{c}_u$ which is given by

$$\hat{\mathbf{c}}_u = (\boldsymbol{r}^T\boldsymbol{r} \otimes B^T B)^{-1}\boldsymbol{r}^T \otimes B^T\mathbf{z}^*, \tag{4.5}$$

can written as

$$\hat{\mathbf{c}}_u = (B^T B)^{-1}B^T\bar{\mathbf{z}}, \tag{4.6}$$

where $\bar{\mathbf{z}}$ is given by

$$\bar{\mathbf{z}} = \begin{bmatrix} \bar{z}_{11} \\ \vdots \\ \bar{z}_{1m} \\ \bar{z}_{21} \\ \vdots \\ \bar{z}_{2m} \\ \vdots \\ \bar{z}_{nm} \end{bmatrix}$$

where $\bar{z}_{ij} = \frac{1}{l}\sum_{k=1}^{l} z_{ijk}$ are the average over the replications and z_{ij} are the observations at the ith location and the jth time point.

Proof. Using Kronecker product rules used in Seshadri (2017), one can write $(\boldsymbol{r}^T \otimes B^T)\mathbf{z}^* = B^T mat(\mathbf{z}^*)\boldsymbol{r}$. The matrix $mat(\mathbf{z}^*)$ can be written as

$$mat(\mathbf{z}^*)_{(nm\times l)} = \begin{bmatrix} z_{111} & z_{112} & \cdots & z_{11l} \\ \vdots & \vdots & \cdots & \vdots \\ z_{1m1} & z_{1m2} & \cdots & z_{1ml} \\ z_{211} & z_{212} & \cdots & z_{21l} \\ \vdots & \vdots & \cdots & \vdots \\ z_{2m1} & z_{2m2} & \cdots & z_{2ml} \\ \vdots & \vdots & \cdots & \vdots \\ z_{nm1} & z_{nm2} & \cdots & z_{nml} \end{bmatrix}.$$

Now, $mat(\mathbf{z}^*) \cdot \boldsymbol{r}$ can be simplified as

$$mat(\mathbf{z}^*)_{(nm\times l)} \cdot \boldsymbol{r}_{(l)} = \begin{bmatrix} z_{111} & z_{112} & \cdots & z_{11l} \\ \vdots & \vdots & \cdots & \vdots \\ z_{1m1} & z_{1m2} & \cdots & z_{1ml} \\ z_{211} & z_{212} & \cdots & z_{21l} \\ \vdots & \vdots & \cdots & \vdots \\ z_{2m1} & z_{2m2} & \cdots & z_{2ml} \\ \vdots & \vdots & \cdots & \vdots \\ z_{nm1} & z_{nm2} & \cdots & z_{nml} \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ \vdots \\ 1_l \end{bmatrix} = \begin{bmatrix} z_{111} + z_{112} + \cdots + z_{11l} \\ \vdots \\ z_{1m1} + z_{1m2} + \cdots + z_{1ml} \\ z_{211} + z_{212} + \cdots + z_{21l} \\ \vdots \\ z_{2m1} + z_{2m2} + \cdots + z_{2ml} \\ \vdots \\ z_{nm1} + z_{nm2} + \cdots + z_{nml} \end{bmatrix} = l \cdot \begin{bmatrix} \bar{z}_{11} \\ \vdots \\ \bar{z}_{1m} \\ \bar{z}_{21} \\ \vdots \\ \bar{z}_{2m} \\ \vdots \\ \bar{z}_{nm} \end{bmatrix} = l \cdot \bar{\mathbf{z}}$$

Then $\hat{\mathbf{c}}_u$ can be written as,

$$\hat{\mathbf{c}}_u = (B^T B)^{-1} B^T \bar{\mathbf{z}}_{(nm)}$$

□

Now we move on to the penalised estimator and we will use similar analytical results to benefit our computation.

4.3.2 RST-PDE with penalty

Using the same basis functions and the temporal and spatial penalties as in (Bernardi et al., 2017). We now show the steps for obtaining the penalised estimator for the penalised sum of square error.

Corollary 4.4. The penalised sum of square errors of the replicated model in (4.1) can be stated as

$$\begin{aligned} J &= (\mathbf{z}^* - A\mathbf{c})^T(\mathbf{z}^* - A\mathbf{c}) + \lambda_s \mathbf{c}^T(P_S \otimes K_0)\mathbf{c} + \lambda_T \mathbf{c}^T(R_0 \otimes P_T)\mathbf{c}, \\ &= (\mathbf{z}^* - A\mathbf{c})^T(\mathbf{z}^* - A\mathbf{c}) + \mathbf{c}^T P\mathbf{c}, \end{aligned} \tag{4.7}$$

and the coefficients vector $\hat{\mathbf{c}}$ can be obtained by least square approximation as follows

$$\hat{\mathbf{c}} = \left(B^T B + \frac{P}{l}\right)^{-1} B^T \bar{\mathbf{z}}. \tag{4.8}$$

Proof. using the fact that $A = \boldsymbol{r} \otimes B$, the penalised least square in (4.7) is given by

$$(\mathbf{z}^* - A\mathbf{c})^T(\mathbf{z}^* - A\mathbf{c}) + \mathbf{c}^T P\mathbf{c}$$

Expanding the sum of squares in A we get

$$(\mathbf{z}^* - A\mathbf{c})^T(\mathbf{z}^* - A\mathbf{c}) + \mathbf{c}^T P\mathbf{c} = \mathbf{z}^{*T}\mathbf{z}^* - \mathbf{z}^{*T}(A\mathbf{c}) - (A\mathbf{c})^T\mathbf{z}^* + (A\mathbf{c})^T A\mathbf{c} + \mathbf{c}^T P\mathbf{c}$$

$$= \mathbf{z}^{*T}\mathbf{z}^* - 2\mathbf{z}^*\mathbf{c}^T A^T + \mathbf{c}^T A^T A\mathbf{c} + \mathbf{c}^T P\mathbf{c}$$

To determine the vector, $\hat{\mathbf{c}}$, we differentiate the sum of squared with respect to the $\mathbf{c}$ and set it equal to zero we get

$$-2A^T\mathbf{z}^* + 2A^T A\hat{\mathbf{c}} + 2P\hat{\mathbf{c}} = 0$$

$$\mathbf{z}^* A^T = A^T A\hat{\mathbf{c}} + P\hat{\mathbf{c}}$$

$$\mathbf{z}^* A^T = \hat{\mathbf{c}}(A^T A + P)$$

$$\hat{\mathbf{c}} = (A^T A + P)^{-1} A^T \mathbf{z}^*$$

using $A = r \otimes B$, we can write $\hat{\mathbf{c}}$ as follows

$$\hat{\mathbf{c}} = ((r \otimes B)^T r \otimes B + P)^{-1}(r \otimes B)^T \mathbf{z}^*$$

$$\hat{\mathbf{c}} = (r^T r \otimes B^T B + P)^{-1}(\boldsymbol{r}^T \otimes B^T)\mathbf{z}^*$$

$$\hat{\mathbf{c}} = (r^T r \otimes B^T B + P)^{-1} B^T mat(\mathbf{z}^*)\boldsymbol{r}$$

Set $\bar{z} = mat(\mathbf{z}^*)\boldsymbol{r}^T$ then we get

$$\hat{\mathbf{c}} = \left(B^T B + \frac{P}{l}\right)^{-1} B^T \bar{z}$$

□

4.4 Properties of the estimator

First we introduce the properties of the standard least square estimator $\hat{\mathbf{c}}_u$. The coefficients vector $\hat{\mathbf{c}}_u$ can be obtained by least square approximation and then the mean and variance of $\hat{\mathbf{c}}_u$ can be defined.

Theorem 4.2.

$$E[\hat{\mathbf{c}}_u] = (B^T B)^{-1} B^T \mathbf{f},$$
$$Var[\hat{\mathbf{c}}_u] = \frac{\sigma^2}{l}(B^T B)^{-1}.$$

Proof. The mean of $\hat{\mathbf{c}}_u$ in term of A matrix is given by

$$E[\hat{\mathbf{c}}_u] = (A^T A)^{-1} A^T E(\mathbf{z}^*),$$

Using the fact that the mean of $\mathbf{z}^*$ is given by $E(\mathbf{z}^*) = \mathbf{f}^*$, Then we can write

$$E[\hat{\mathbf{c}}_u] = (A^T A)^{-1} A^T \mathbf{f}^*.$$

Using $A = \boldsymbol{r} \otimes B$ we get

$$E[\hat{\mathbf{c}}_u] = ((\boldsymbol{r} \otimes B)^T (\boldsymbol{r} \otimes B))^{-1} (\boldsymbol{r} \otimes B)^T \mathbf{f}^*.$$

Using corollary (4.1), we write

$$E[\hat{\mathbf{c}}_u] = (\boldsymbol{r}^T \boldsymbol{r} \otimes B^T B)^{-1} (\boldsymbol{r}^T \otimes B^T)(\boldsymbol{r} \otimes \mathbf{f}).$$

$$E[\hat{\mathbf{c}}_u] = (\boldsymbol{r}^T \boldsymbol{r} \otimes B^T B)^{-1} \boldsymbol{r}^T \boldsymbol{r} \otimes B^T \mathbf{f}.$$

Then, the mean of the estimator is written as

$$E[\hat{\mathbf{c}}_u] = (B^T B)^{-1} B^T \mathbf{f},$$

Similar to the mean, the variance of $\hat{\mathbf{c}}_u$ in terms of A matrix is given by

$$Var[\hat{\mathbf{c}}_u] = (A^T A)^{-1} A^T Var(\mathbf{z}^*) A (A^T A)^{-1},$$

Using the fact that $Var[\mathbf{z}^*] = \sigma^2 I$, then variance of the coefficient vector is given by

$$Var[\hat{\mathbf{c}}_u] = \sigma^2(A^TA)^{-1}.$$

We write it in terms of B matrix as follows

$$Var[\hat{\mathbf{c}}_u] = \sigma^2(\boldsymbol{r}^T\boldsymbol{r} \otimes B^TB)^{-1} = \frac{\sigma^2}{l}(B^TB)^{-1}.$$

□

Note that the mean of the replicated estimator is same as the estimator of the individual replicates, but the variance is $\frac{1}{l}$ times the variance of the estimator of the individual.

We defined the estimation of $\hat{\mathbf{c}}_u$ for the standard least square now similarly we define them for the penalised least square estimator $\hat{\mathbf{c}}$.

Theorem 4.3.

$$E[\hat{\mathbf{c}}] = \Big(B^TB + \frac{P}{l}\Big)^{-1}B^T\mathbf{f},$$
$$Var[\hat{\mathbf{c}}] = \frac{\sigma^2}{l}\Big(B^TB + \frac{P}{l}\Big)^{-1}B^TB\Big(B^TB + \frac{P}{l}\Big)^{-1}$$

Proof. The mean of $\hat{\mathbf{c}}$ is given by

$$E[\hat{\mathbf{c}}] = (A^TA + P)^{-1}A^TE(\mathbf{z}^*),$$

Using the fact that the mean of $\mathbf{z}^*$ is given by $E(\mathbf{z}^*) = \mathbf{f}^*$, Then we can write

$$E[\hat{\mathbf{c}}] = (A^TA + P)^{-1}A^T\mathbf{f}^*,$$

Using $A = \boldsymbol{r} \otimes B$ and the simplification of $\mathbf{f}^*$ we get

$$E[\hat{\mathbf{c}}] = ((\boldsymbol{r} \otimes B)^T\boldsymbol{r} \otimes B + P)^{-1}(\boldsymbol{r} \otimes B)^T(\boldsymbol{r} \otimes \mathbf{f}),$$
$$E[\hat{\mathbf{c}}] = \Big(B^TB + \frac{P}{l}\Big)^{-1}B^T\mathbf{f},$$

The variance of $\hat{\mathbf{c}}$ is given by

$$Var[\hat{\mathbf{c}}] = (A^TA + P)^{-1}A^TVar(\mathbf{z}^*)A(A^TA + P)^{-1},$$

Using the fact that the variance of $\mathbf{z}^*$ is given by $Var[\mathbf{z}^*] = \sigma^2 I$, then variance of $\hat{\mathbf{c}}$ is given by

$$Var[\hat{\mathbf{c}}] = \sigma^2(A^TA + P)^{-1}A^TA(A^TA + P)^{-1}.$$

we write it in terms of B matrix as follows

$$Var[\hat{\mathbf{c}}] = \sigma^2(\boldsymbol{r}^T\boldsymbol{r} \otimes B^TB + P)^{-1}(\boldsymbol{r}^T\boldsymbol{r} \otimes B^TB)(\boldsymbol{r}^T\boldsymbol{r} \otimes B^TB + P)^{-1}$$

$$Var[\hat{\mathbf{c}}] = \sigma^2\Big(l(B^TB + \frac{P}{l})\Big)^{-1}\Big(l(B^TB)\Big)\Big(l(B^TB + \frac{P}{l})\Big)^{-1}$$

$$Var[\hat{\mathbf{c}}] = \frac{\sigma^2}{l}\Big(B^TB + \frac{P}{l}\Big)^{-1}B^TB\Big(B^TB + \frac{P}{l}\Big)^{-1}$$

□

Now we need to define the estimation of the spatio-temporal surface $\hat{f}$.

Theorem 4.4. suppose $\boldsymbol{B}(p,t) = \boldsymbol{\psi}(p)^T \otimes \boldsymbol{\phi}(t)^T$ is a vector of the evaluation of the basis functions at the spatio-temporal points (p,t). Then, the estimated spatio-temporal field $\hat{\mathbf{f}}$ at (p,t) is given by

$$\hat{\mathbf{f}}(p,t) = \boldsymbol{B}(p,t)\Big(B^TB + \frac{P}{l}\Big)^{-1}B^T\bar{\mathbf{z}}.$$

Proof. We know that $\mathbf{f}$ is defined by the basis system as $\mathbf{f} = B\mathbf{c}$. Then the estimation of the surface $\hat{\mathbf{f}}$ at (p,t) can be given as

$$\hat{\mathbf{f}}(p,t) = \boldsymbol{B}(p,t)\hat{\mathbf{c}} = \boldsymbol{B}(p,t)(A^TA + P)^{-1}A^T\mathbf{z}^*.$$

$$\hat{\mathbf{f}}(p,t) = \boldsymbol{B}(p,t)\hat{\mathbf{c}} = \boldsymbol{B}(p,t)\Big(B^TB + \frac{P}{l}\Big)^{-1}B^T\bar{\mathbf{z}}.$$

□

Then, it is possible to define the mean and the variance of the spatio-temporal field.

Theorem 4.5. The mean and the variance of the spatio-temporal surface $\hat{\mathbf{f}}$ are given by

$$E[\hat{\mathbf{f}}(p,t)] = \boldsymbol{B}(p,t)\Big(B^TB + \frac{P}{l}\Big)^{-1}B^T\mathbf{f},$$
$$Var[\hat{\mathbf{f}}(p,t)] = \frac{\sigma^2}{l}\boldsymbol{B}(p,t)\Big(B^TB + \frac{P}{l}\Big)^{-1}B^TB\Big(B^TB + \frac{P}{l}\Big)^{-1}\boldsymbol{B}(p,t)^T.$$

Proof. The mean of $\hat{\mathbf{f}}$ is given by

$$E[\hat{\mathbf{f}}(p,t)] = \boldsymbol{B}(p,t)(A^TA + P)^{-1}A^TE(\mathbf{z}^*),$$

Using the fact that the mean of Z is giben by $E(\mathbf{z}^*) = \mathbf{f}^*$, Then we can write

$$E[\hat{\mathbf{f}}(p,t)] = \boldsymbol{B}(p,t)(A^TA + P)^{-1}A^T\mathbf{f}^*,$$

Using $A = \boldsymbol{r} \otimes B$ and $\mathbf{f}^* = \boldsymbol{r} \otimes \mathbf{f}$ we get,

$$E[\hat{\mathbf{f}}(p,t)] = \boldsymbol{B}(p,t)\Big(B^TB + \frac{P}{l}\Big)^{-1}B^T\mathbf{f},$$

Similar to the variance, the variance of $\hat{\mathbf{f}}$ is given by

$$Var[\hat{\mathbf{f}}(p,t)] = \boldsymbol{B}(p,t)(A^TA + P)^{-1}A^TVar(\mathbf{z}^*)A(A^TA + P)^{-1}\boldsymbol{B}(p,t)^T,$$

We know that the variance of $\mathbf{z}^*$ is given by $Var[\mathbf{z}^*] = \sigma^2 I$, then variance of $\hat{\mathbf{f}}$ is given by

$$Var[\hat{\mathbf{f}}(p,t)] = \sigma^2\boldsymbol{B}(p,t)(A^TA + P)^{-1}A^TA(A^TA + P)^{-1}\boldsymbol{B}(p,t)^T.$$
$$Var[\hat{\mathbf{f}}(p,t)] = \frac{\sigma^2}{l}\boldsymbol{B}(p,t)\Big(B^TB + \frac{P}{l}\Big)^{-1}B^TB\Big(B^TB + \frac{P}{l}\Big)^{-1}\boldsymbol{B}(p,t)^T.$$

□

Furthermore, the covariance between two spatio-temporal locations (p_1, t_1) and (p_2, t_2) can be given by

$$cov[\hat{\mathbf{f}}(p_1,t_1), \hat{\mathbf{f}}(p_2,t_2)] = \sigma^2\boldsymbol{B}(p_1,t_1)(A^TA + P)^{-1}A^TA(A^TA + P)^{-1}\boldsymbol{B}(p_2,t_2)^T.$$
$$cov[\hat{\mathbf{f}}(p_1,t_1), \hat{\mathbf{f}}(p_2,t_2)] = \frac{\sigma^2}{l}\boldsymbol{B}(p_1,t_1)\Big(B^TB + \frac{P}{l}\Big)^{-1}B^TB\Big(B^TB + \frac{P}{l}\Big)^{-1}\boldsymbol{B}(p_2,t_2)^T.$$

Consider the vector $\widehat{\mathbf{z}^*}$ of the fitted values at the space-time points over l replications $\widehat{\mathbf{z}^*} = S\mathbf{z}^*$ where S is the smoothing matrix $S = A(A^TA+P)^{-1}A^T$. The smoothing matrix can be written as

$$S = \frac{\boldsymbol{r} \otimes B}{l}\Big(B^TB + \frac{P}{l}\Big)^{-1}(\boldsymbol{r} \otimes B)^T.$$

Theorem 4.6. The vector of the fitted values $\widehat{\mathbf{z}^*}$ can be obtained as,

$$\widehat{\mathbf{z}^*} = \boldsymbol{r} \otimes \hat{\mathbf{z}},$$

where $\hat{\mathbf{z}}$ is the vector of length nm represents the fitted values at space-time points avaerging over replications.

Proof. We know that

$$\widehat{\mathbf{z}^*} = S\mathbf{z}^*,$$
$$\widehat{\mathbf{z}^*} = \boldsymbol{r} \otimes B\Big(B^TB + \frac{P}{l}\Big)^{-1}B^Tmat(\mathbf{z}^*)\boldsymbol{r},$$
$$\widehat{\mathbf{z}^*} = \boldsymbol{r} \otimes B\Big(B^TB + \frac{P}{l}\Big)^{-1}B^T\bar{\mathbf{z}}.$$

Set $B(B^TB + \frac{P}{l})^{-1}B^T\bar{\mathbf{z}} = \hat{\mathbf{z}}$ Then, $\widehat{\mathbf{z}^*}$ can be written as,

$$\widehat{\mathbf{z}^*} = \boldsymbol{r} \otimes \hat{\mathbf{z}}$$

□

$\widehat{\mathbf{z}^*}$ though is a vector of dimension nml. It is the same as $\hat{\mathbf{z}}$ repeated l times, which implies we have the same fitted surface for all replications.

We can also estimate σ^2 by

$$\hat{\sigma^2} = \frac{1}{nml - tr(S)}(\mathbf{z}^* - \widehat{\mathbf{z}^*})^T(\mathbf{z}^* - \widehat{\mathbf{z}^*})$$

The smoothing parameters for both the spatial penalty and temporal penalty are chosen using generalised cross validation (GCV) which is defined as follows,

$$GCV(\lambda_S, \lambda_T) = \frac{nml}{(nml - tr(S))^2}(\mathbf{z}^* - \widehat{\mathbf{z}^*})^T(\mathbf{z}^* - \widehat{\mathbf{z}^*}).$$

The smoothing parameters are chosen to minimise GCV.

4.5 Simplification of the estimator for increasing computation speed

The estimation of the model motivates a huge computational cost due to the large dimension of the A matrix. The challenges in estimating the model components include

- Storage; as the calculation of the matrix multiplication (A^TA) is very large we had to use external server to perform the computation.
- Computational speed.

The estimation of $(A^TA + P)^{-1}$ is complex as it require the multiplication of large matrices A^TA. We simplify the estimation using transpose and Kronecker product properties as follows,

Remark.

$$\begin{aligned}(A^TA + P)^{-1} &= [(\boldsymbol{r} \otimes B)^T(\boldsymbol{r} \otimes B) + P]^{-1} \\ &= [(\boldsymbol{r}^T \otimes B^T)(\boldsymbol{r} \otimes B) + P]^{-1} \\ &= [(\boldsymbol{r}^T\boldsymbol{r} \otimes B^TB) + P]^{-1} \\ &= [(l \otimes B^TB) + P]^{-1} \\ &= \frac{1}{l}\Big(B^TB + \frac{P}{l}\Big)^{-1}.\end{aligned}$$

The inverse now become simpler as we avoid large matrices in the estimation. In the next section, we provide a simulation study to investigate the performance of our new framework RST-PDE and compare it with existing techniques.

4.6 Simulation study

We perform a simulation study to compare the ST-PDE approach proposed by Bernardi et al. (2017) with our extend approach replicated ST-PDE which accommodates data with replications. First we should compare ST-PDE with other spatio-temoral estimation methods to examine ST-PDE performance.

The first approach is spatio-temporal kriging with a separable variogram and the other two models are similar to (3.25), with the penalty term consisting of a spatial penalty applied to a spatially varying coefficient on the temporal basis and a temporal penalty applied to a temporally varying coefficient on the space basis. The first approach is presented by Augustin et al. (2013) and uses cubic splines as temporal basis and thin-plate splines as space basis and the spatial penalty is also represented by thin-plate splines. The other approach presented by (Marra et al., 2012) uses cubic splines as temporal basis and soap film smoothing in space. Bernardi et al. (2017) shows that the ST-PDE approach outperforms the other three spatio-temporal approaches. We will illustrate the RMSE of these approaches later in this section.

In this simulation study, we sample 200 spatial points randomly generated on a c-shaped manifold using the `"spsample"` function which is provided by `"sp"` package in R (Pebesma and Bivand, 2005). The time component is defined to be equally spaced in the interval $[0, \pi]$. We simulate the data from model 4.1, where the spatio-temporal function is estimated from two functions in a combined way. The spatial part is estimated by a spatial test function which is proposed by (Ramsay, 2002), while the temporal part is added by multiplying the test function by a cosine function of time, $cos(t)$. The errors are generated from a normal distribution with mean 0 and standard deviation 0.5. Figure 4.4 shows the simulated data of one replication at one time point and the line illustrates the borders of the spatial domain.

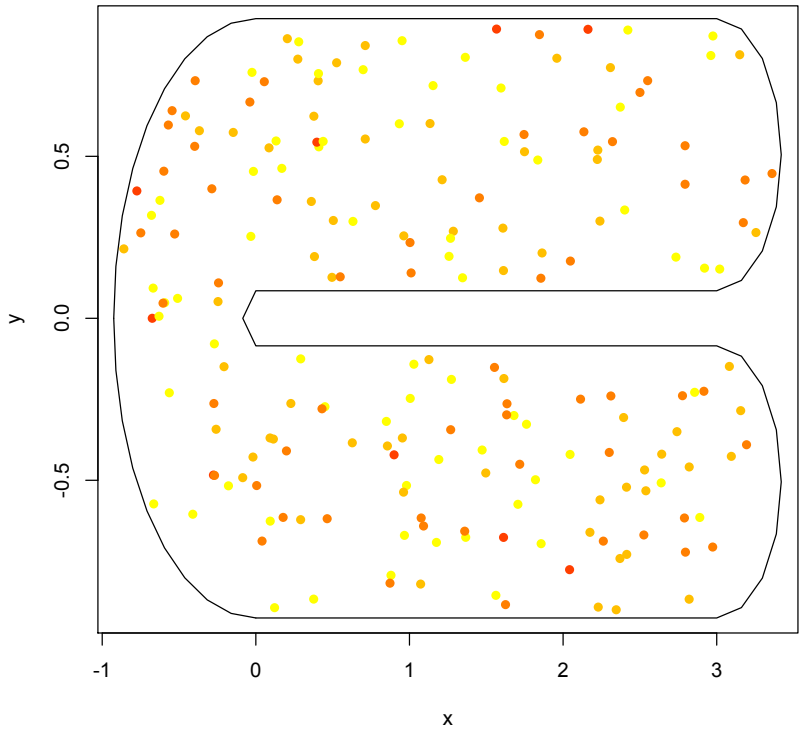

Figure 4.4: A plot of simulated data

We apply both standard ST-PDE and newly developed replicated ST-PDE approach to simulated data with 200 spatial points, 9 time points and 10 replications. Replicated ST-PDE is applied to the data directly. However, as there is no research on the application of the ST-PDE on replicates data, one way of pooling the results obtained from 10 replicates is to average over the results obtained from the individuals replications. Figure 4.5 shows the results of applying the two methods. The first column shows the spatio-temporal true function over different time points. while the second column shows the average of 10 replicated spatio-temporal estimates using ST-PDE and the third column shows the spatio-temporal estimates using replicated ST-PDE. Both ST-PDE and replicated ST-PDE provide good spatio-temporal estimation of the data.

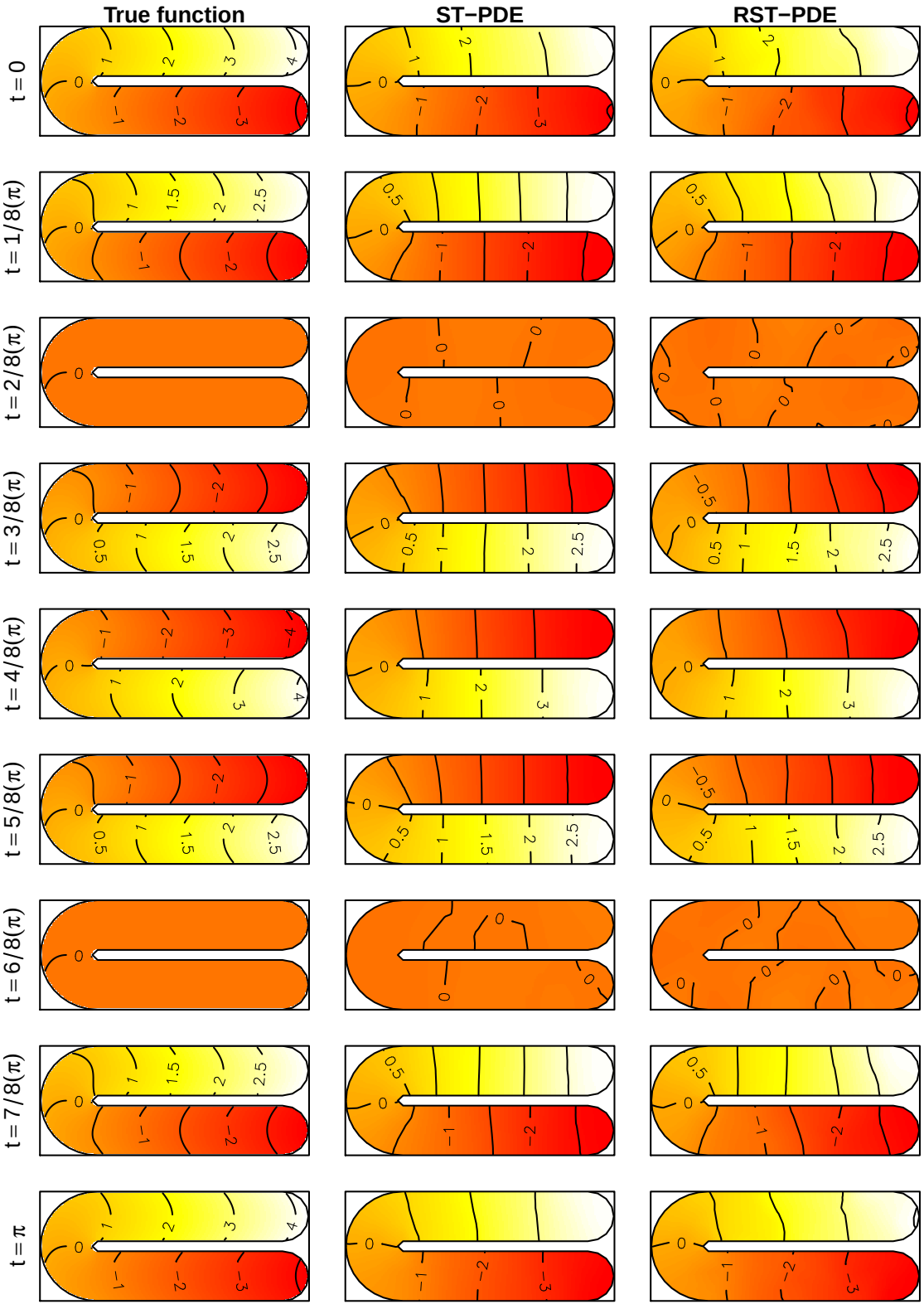

Figure 4.5: Spatio-temporal surface of true function (test function) in the first column, Spatio-temporal surface estimates using ST-PDE and RST-PDE models in the second and third column, respectively. Each row represent the spatio-temporal surface at fixed time point

To illustrate the approaches performances numerically, we calculate the RSME of both approaches applied to 50 iterations. Figure 4.6 shows box plots of the RMSE of the estimates of the spatio-temporal field produced by different methods. The right panel shows box plots of RMSE for ST-PDE and the other three approaches produced by (Bernardi et al., 2017), while the left panel shows the RMSE values of applying ST-PDE and RST-PDE to the simulated data.

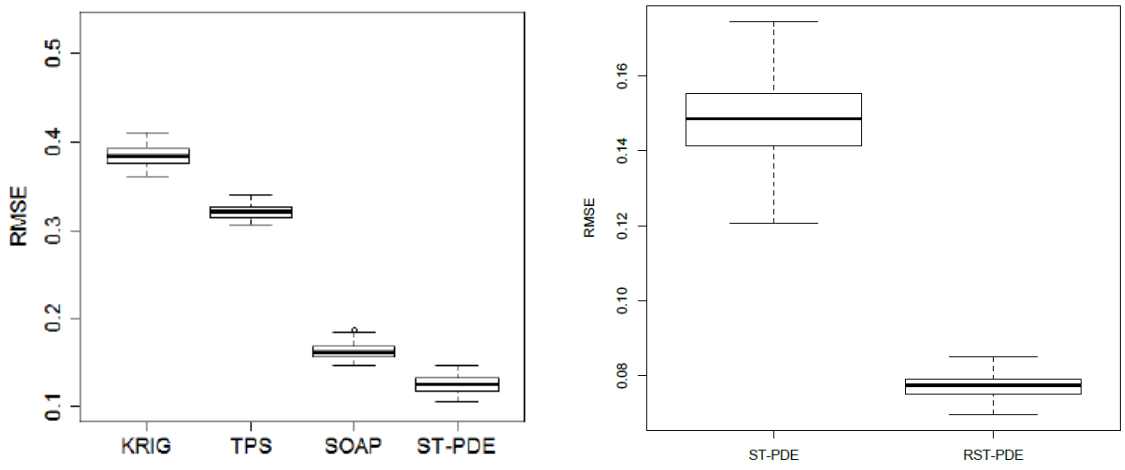

Figure 4.6: Left panel:Box plots of the RMSE of the estimates of the spatio-temporal field obtained by the four methods: spatio-temporal kriging (KRIG), space-time model using thin plate spline (TPS), space-time model using soap film smoothing (SOAP) and ST-PDE. (Bernardi et al., 2017). Right panel: Box plots of RMSE of both ST-PDE and RST-PDE approaches. Note that the ranges of y axis in the two plots are different

The left panel shows that ST-PDE outperforms the other three approaches and provides a lower RMSE. The right panel of Figure 4.6 indicates that the RST-PDE approach outperforms the ST-PDE. The median of RMSE of RST-PDE is lower than the RMSE median of all other approaches in the left panel. The box plots do not overlap indicating a significant difference. The maximum RMSE value in the RST-PDE is lower than the minimum values of RMSE of the ST-PDE. We believe that RST-PDE can effectively pool information from all replicates and thus performs better than 50 individuals ST-PDE estimates of the surface. We also compute the

computational time of ST-PDE and RST-PDE and found that ST-PDE takes more than two times the computational time of RST-PDE (see appendix A)

4.7 Summary

In this chapter we have developed a new framework by extending the ST-PDE approach which is designed to model space-time data to the RST-PDE approach which accommodates the replicated space-time data. We use the properties of Kronecker products to simplify high dimensional computations. Additionally we have investigated the performance of our new framework through a full-scale simulation study. The simulation shows that the the new approach perform well compared to existing approaches that are designed to analyse nonreplicated data.

Chapter 5

Harvard forest vegetation index data

In this chapter we will analyse the Harvard forest vegetation index data, introduced earlier in Chapter 3, using the two main approaches SPACE and ST-DPE. The main goal of this chapter is to extend the existing framework of SPACE and ST-DPE to accommodate more general data-structures than they were originally developed for. Note that SPACE was designed to analyse data which were obtained from a regular spatial grid, although it allows for temporal irregularity or sparseness. On the other hand, ST-DPE was built to analyse data with temporal regularity, but could analyse spatially sampled data over a pre-defined region. In particular, this chapter will investigate whether the SPACE approach can be applied to non-gridded data such as sampled data. Note that the original data on a 25x25 grid was analysed in Liu et al. (2017). In this chapter we will use a spatially sparse sample on the spatial domain and create a non-gridded sparse version on the 25x25 grid to test how SPACE perform on spatially non-gridded data. We develop a new framework by extending the distance measure used to calculate the spatial correlation and extend SPACE to handle non-gridded data. Coming back to ST-PDE approach which was originally designed for spatially sparse sampled data, in this chapter we extend it to accommodated spatially dense and gridded data.

The first part of the chapter provides exploratory analysis of the EVI dataset to highlight the main features of the data. Section 4 illustrates the new framework of implementing SPACE to non-gridded samples of the EVI data and compares

the reconstruction output of non-gridded EVI data using two different distance measures. Section 5 demonstrates the use of an ST-PDE approach to EVI data with a different spatial structure. Finally, we provide a detailed comparison of the performance of SPACE and ST-PDE approaches when analysing EVI data.

5.1 Data description

The EVI data consists of satellites images of 25x25 pixel area located in Harvard Forest. The data represent the surrogate measure of greenness, which takes values between -1 and 1, and is temporally observed over six years . However for simplicity, in this chapter we only use the data observed over one year and thresholded the data at 0.1 which corresponds to the historical minimum EVI values under snow-free conditions at this site. Any pixels with any EVI values below this threshold are treated as missing values. Then, the data observations are given by $\{y_{ij}, 1 \leq i \leq 625, 1 \leq j \leq 46\}$, 625 replicated curves over 46 time points where each curve corresponds to one pixel.

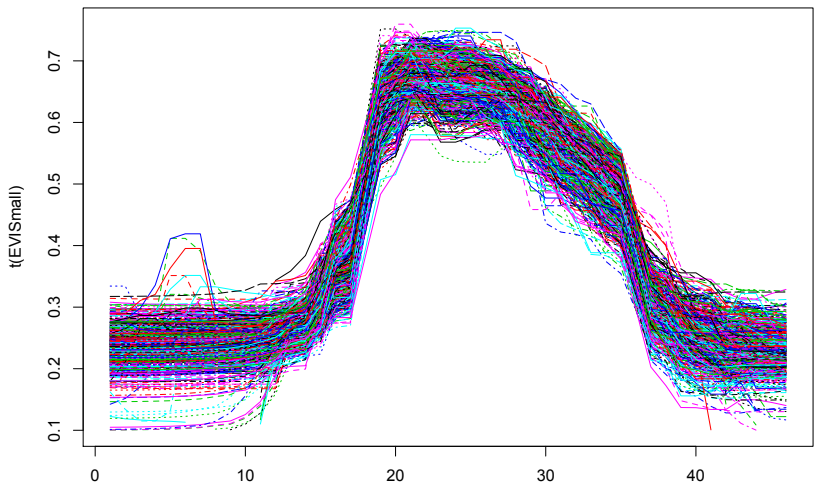

Figure 5.1: EVI data over one year time where each curves represents the data for one location.

Figure 5.1 shows the EVI data over a single year where the level of greenness

is low at the beginning and the end of the year and reaches the highest level in the middle of the year during summer months. Due to the nature of the dataset, it can be considered as functional data analysis. In the next section we represent the data in functional form.

5.2 Functional EVI data

The first step in functional data techniques is to convert the data from discrete points to continuous functions. Suppose y_{ij} is the observation at location $i(i = 1, \cdots, 625)$ and time point $j(j = 1, \cdots, 46)$, then following the notations in Chapter 2 the model can be written as functions in time and space as follows,

$$y_{ij} = x_i(t_j) + \epsilon_{ij},$$

where x_i are continuous smooth functions and ϵ_{ij} represent the error term. The continuous curves $x_i(t)$ are estimated using a regularisation approach based on basis functions. There are several types of basis functions; the most common ones are Fourier basis functions and b-spline basis functions. A Fourier basis would be more appropriate when we use the whole data set because it is periodic data. However, we are using data for a single year with no repeating cycle, so for this reason we choose to use b-spline basis functions. We represent our functional data using 46 b-spline basis functions of order 4. The smoothing level is controlled by a roughness penalty which is defined by the second derivative, while the smoothing parameter is chosen by generalised cross validation (GCV). Figure 5.2 illustrates that GCV has a minimum value at a smoothing parameter equal to 1 (log10 $\lambda = 0$). Using larger or smaller smoothing parameters lead to larger GCV.

To verify smoothing parameter selection, we also examine the effect of using different smoothing parameters on the estimated smoothing curves by visual inspection. This suggests that a smoothing parameter with value 1 provides the best fit of the data. Therefore, it has been decided to smooth the EVI data using penalized cubic b-splines basis with smoothing parameter equal to 1, where the penalty is defined by the integrated squared second derivative.

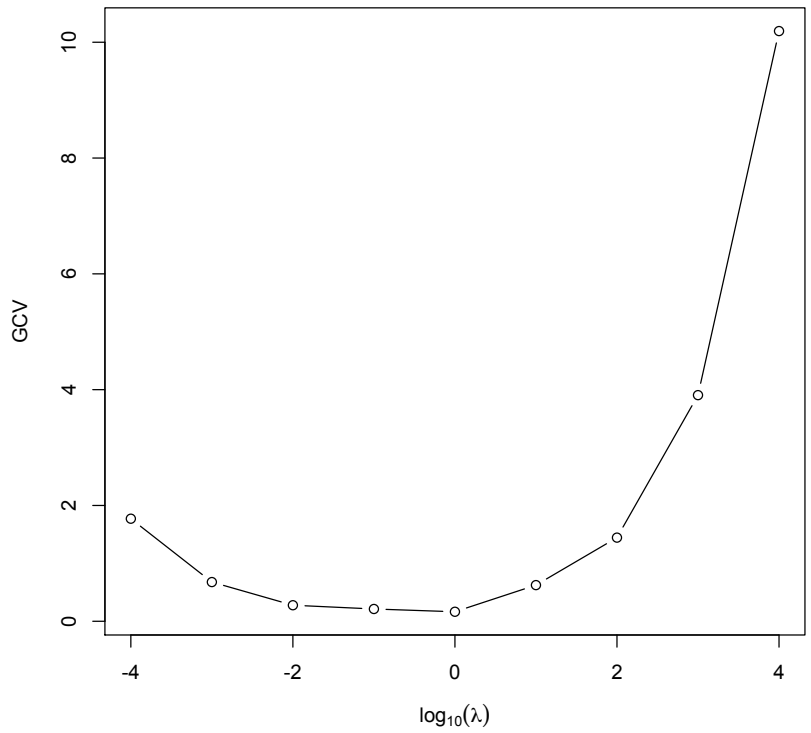

Figure 5.2: Plot of the GCV criterion against the corresponding smoothing parameter λ (in log10 scale) used to fit the EVI smooth curves

The smoothed EVI data are shown in Figure 5.3, where the red lines represent the mean of the data. The mean here is the sample mean which is defined in this case by

$$\overline{x}(t) = \frac{1}{625}\sum_{i=1}^{625} x_i(t).$$

The curves in 5.3, which represent the locations, behave similarly which indicate high spatial correlation among the data. In order to gain better understanding of the data structure we estimate the variance and correlation of the data using functional data techniques.

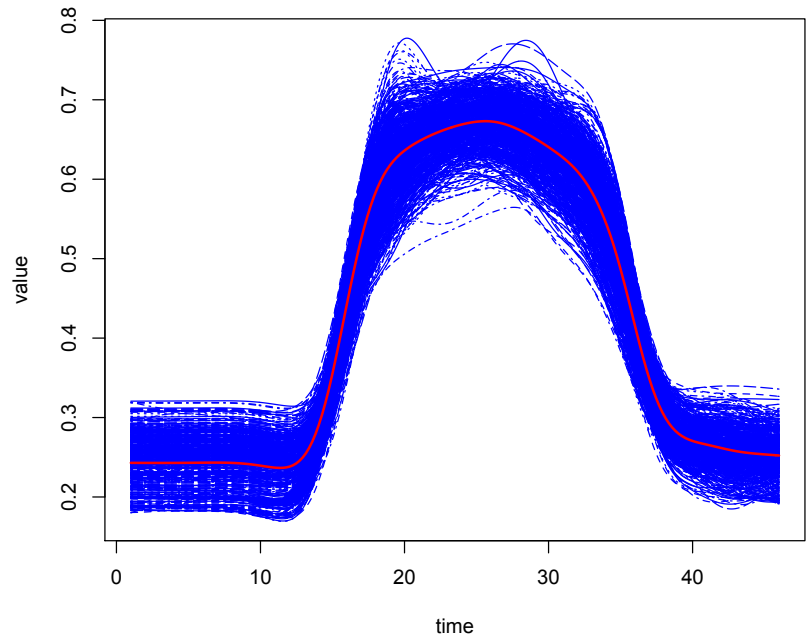

Figure 5.3: Smoothed EVI data over time using b-spline basis system. The red line represent the mean of the data.

The correlation function is calculated to investigate the dependence in the EVI dataset. Figure 5.4 shows the correlation function of the EVI data, both as a surface over the plane of all possible pairs of time points (t_1, t_2) and also as a contour plot. The correlation function calculates the correlation of the EVI values at every pair of time points along the curves. The diagonal running from the lower left corner to the upper right corner equals one. Naturally, the diagonal values in the two plots represents the correlation of the time points with themselves. The perspective plot in the left panel of 5.4 shows the correlation surface of the EVI values where red colour indicates high correlation values, while blue colour refer to low correlation values. As expected, the EVI values of winter and summer months have very low correlation of about 0.2, which is also clear from the mean plot where the level of greenness in summer months is higher than the level of greenness in winter months.

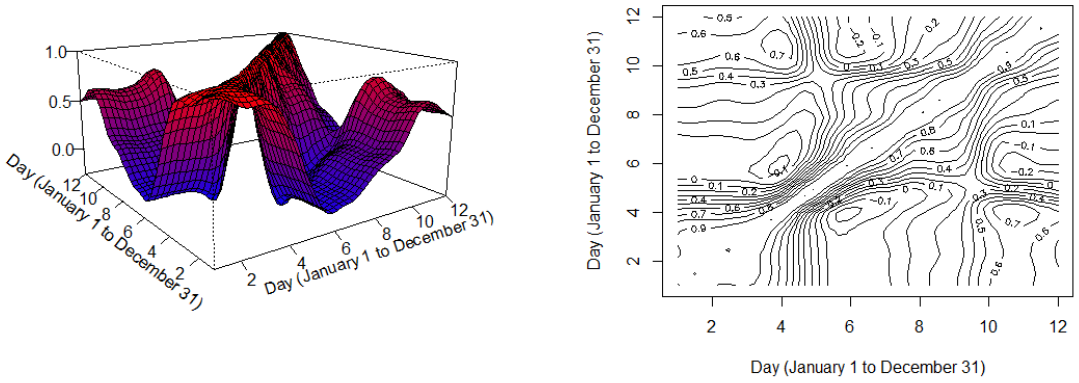

Figure 5.4: Correlation estimation of smoothed EVI data. The left panel is a perspective plot of the correlation function, while the right panel shows the same surface by contour plotting.

The correlation plots 5.4 only show the correlation of the EVI values in the temporal aspect. Thus, in the next section we apply principal component analysis to the EVI data to investigate the spatial correlation among the data and extract the major mode of variation among curves.

5.3 Functional principal component analysis

Functional principal components analysis (FPCA) is a common approach to explore variability in functional data. FPCA approach aims to reduce the dimensionality of functional data by determining uncorrelated components that capture the main modes of variation of the data. In this section, we apply functional principal component analysis, described previously in Chapter 2, to the EVI data. In particular, we determine a number of functional principal components $\xi_j(t)$ that provide a satisfactory approximation of the EVI data. Let $x_i(t)$ be the EVI observations with mean $\hat{\mu}(t) = \frac{1}{N}\sum_{i=1}^{N} x_i(t)$. The data is first centred by subtracting the mean for each variable, and the centred curves are obtained as $\tilde{x}_i(t) = x_i(t) - \hat{\mu}(t)$. This step is done to ensure that the principal components describe the direction of the maximum variation.

As described in Section 2.3, given the estimated covariance $G(t_k, t_l)$, the eigenfunctions $\xi_j(t)$ are estimated by solving the equation $G(t_k, t_l)\xi_j(t_l)dt = \lambda_j\xi_j(t_k)$, where λ_j represent the eigenvalues. The FPCA approach is carried out on the EVI data `"fda"` package in R. The estimated principal components of the EVI data are presented in figure 5.5. The first four principal component curves are displayed in the left panel of figure 5.5, where each component accounts for some amount of variation in the data. The components describe 40.8%, 28%, 12.3% and 7.6% for the first four components, respectively. The right panel represents the total variation accounted for by 46 principal components and it is clear that the first 7 principal components explain nearly 96% of the total variation. The eigenvalues of the first seven functional principal components are summarised in Table 5.1.

	var. explained	cum. var. explained
λ_1	0.408	0.408
λ_2	0.280	0.688
λ_3	0.123	0.811
λ_4	0.076	0.887
λ_5	0.033	0.920
λ_6	0.023	0.943
λ_7	0.017	0.960

Table 5.1: Variance explained by the eigenfunctions of EVI data. First column are the variances explained by each eigenfunction and the second column the cumulative sum of explained variances

The first functional principal component of the EVI data is positive throughout the year, while the values of this component in the winter months are about two times the values in summer months. This indicates that more variability between observations (pixel locations) is found to be during winter months.

The second principal component of the data accounts for 28% of the variation. It consists of positive values for the summer period and negative values for winter period. The second functional principal component can be interpreted as the

difference in the enhanced vegetation index between winter and summer months.

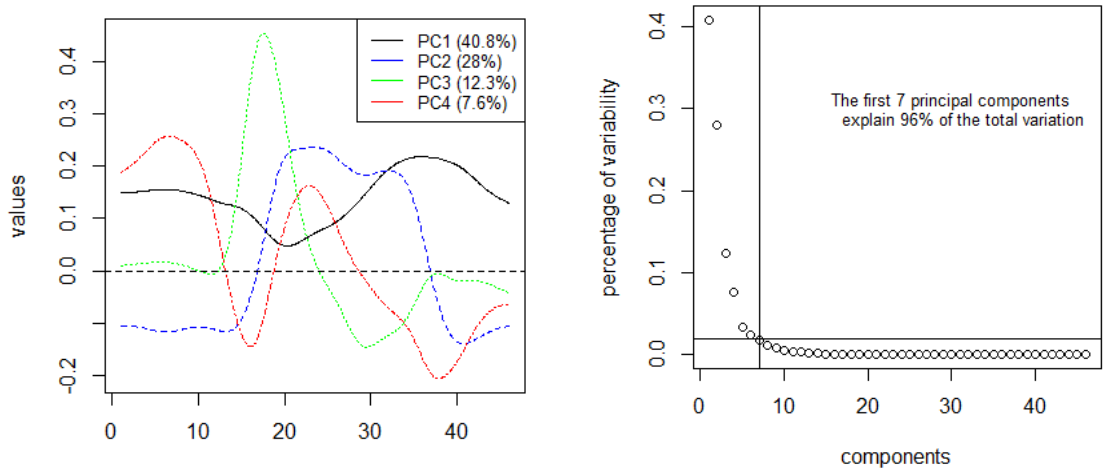

Figure 5.5: Left panel: the first four principal component curves of the EVI data. The percentages denote the variation explained by each component. Right panel: the scree plot of the functional principal components represent the total variation accounted by 46 principal components. The first 7 principal components explain 96% of the total variation.

The third and fourth components are hard to interpret. These components account for small proportions of the variation in the data.

As the first principal component represents the largest amount of variation, we plot the first component scores to explore the spatial variation, which provides a good indicator of whether the EVI data curves themselves are spatially similar to their neighbours. Figure 5.6 shows the first principal component scores for 25x25 pixels where different colours represent different vegetation index. White pixels are pixels with missing values and similar colours represent high correlation among neighbouring. The graph illustrates some amount of spatial correlation in the EVI data.

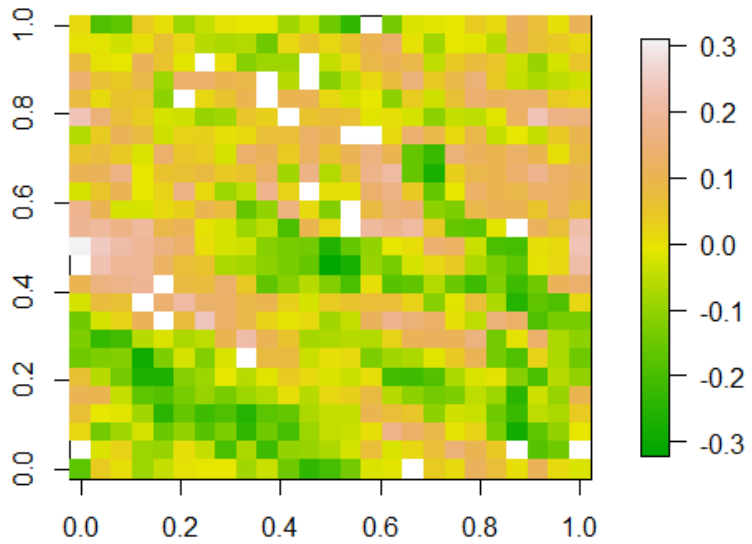

Figure 5.6: The first principal component scores (white boxes indicates pixels with missing EVI values)

The rest of the chapter shows the results obtained by applying SPACE and ST-PDE approaches to the EVI data and compare their performances in analysing and predicting the EVI data set.

5.4 Application of SPACE on EVI data

In this section, we apply the newly developed SPACE method using radius distance on a sparsely sampled set of locations in the EVI data and the full EVI data. The full EVI data was previously analysed using the SPACE method with neighbourhood selection method described in (Liu et al., 2017). We show that the new approach is comparable to SPACE method with neighbourhood selection method when applied on the full dataset, where both methods work. Additionally, we demonstrate that the new framework can incorporate non-gridded data sets, which the previous method was unable to work.

In the original SPACE method, the first four neighbours groups of an original point were counted as 1-unit distance from this point. The first neighbours group consists of 1-unit distance in the vertical line which are the points in the left and right of the original point. The second neighbours group consists of the 1-unit distance points in the right diagonal and this continues in an anticlockwise direction until the fourth group. The fifth neighbours group start from the next layer which is 2-unit distance and so on. The right panel of Figure 5.7 shows how the neighbours are selected in SPACE where each number represents a neighbours group. We generalised the distance concept by defining a radius around the original observation. This radius can be chosen by the user. Suppose we choose the radius to be 2, then the neighbours included for some locations are the ones that are located within 2-units of this location. The radius concept is illustrated in the left panel of Figure 5.7, when the radius equals 1, where the numbers indicates the group of neighbours.

Suppose we want to select the neighbourhood observation for point x (shown in red in the both plots). Figure 5.7 represents the neighbourhood selection of point x using the two selection methods.

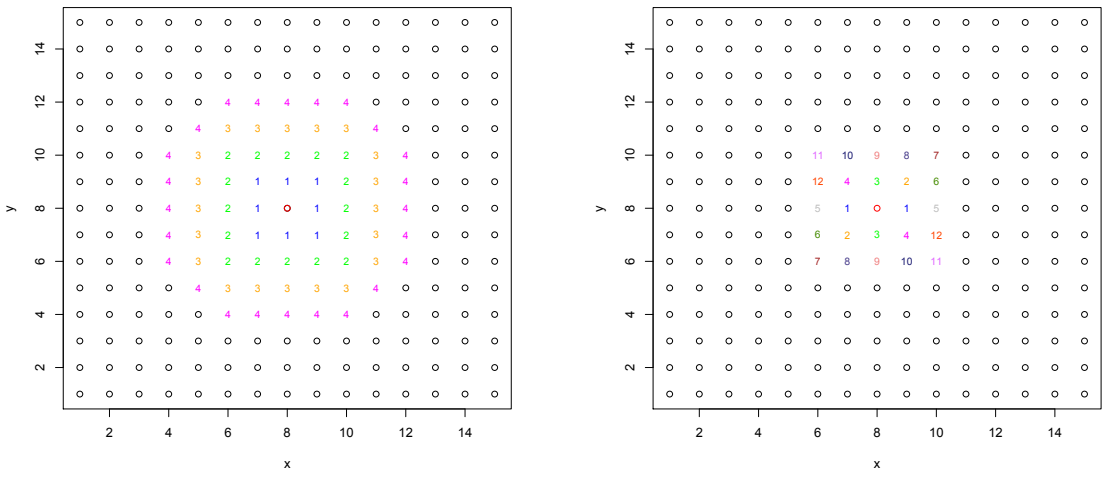

Figure 5.7: Neighbourhood selection for the two selection methods where radius methods is illustrated in the left panel and neighbouring method is shown in the right panel.

The left panel of figure 5.7 shows the neighbours using radius of one to four units and the right panel shows the neighbours using the method described in Liu et al. (2017). The selection using radius is more intuitive than the neighbour idea and is mathematically simpler to define. The other advantage is that it can accommodate non-gridded data.

The estimated curve of applying SPACE to the data is displayed in Figure 5.8, where points represent the raw data of a single observation and the red curve is the model fitted line for this observation. It is clear that the model successfully captures the trend of the data.

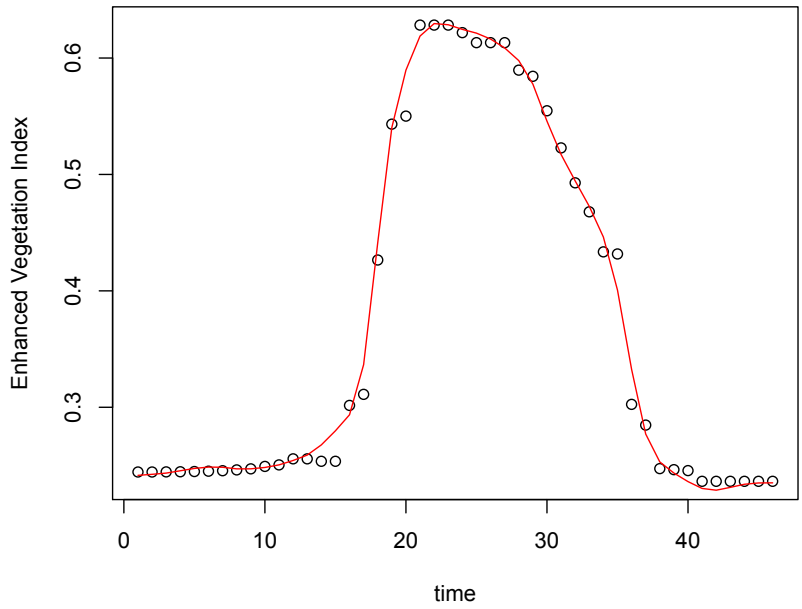

Figure 5.8: The points represent the raw EVI data while the red curves represent SPACE model estimates using radius distance

The root mean squared error (RMSE) of SPACE using radius is equal to 0.0160 while the (RSME) of the estimated values of the EVI data using SPACE with neighbouring distance is equal to 0.0161. There is no difference in errors of the estimated values between the two distance calculation methods.

Now we focus on the spatial correlation. The spatial correlation is estimated using the SPACE method and Figure 5.9 shows the spatial correlation values for

different radius values. As expected the correlation is higher when the radius is small, when the locations are very close to each either, and decreases as we go further a way.

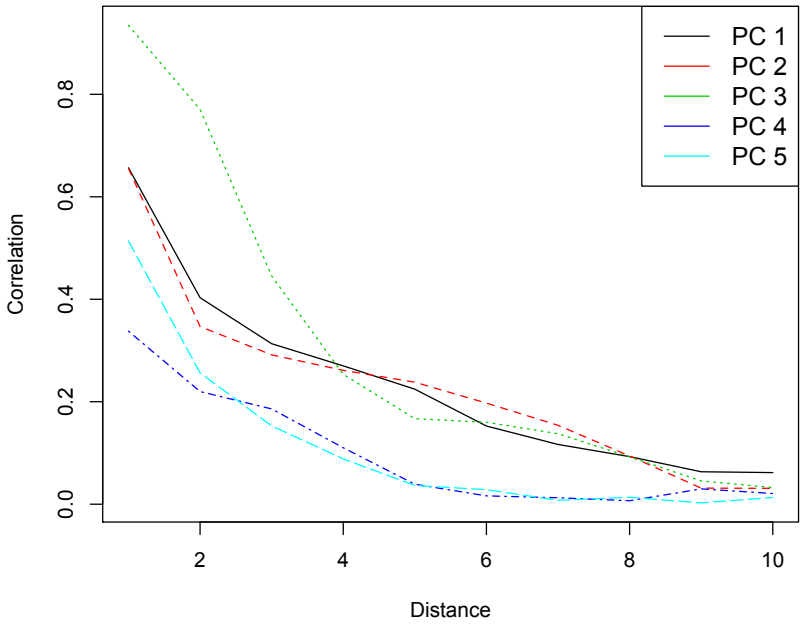

Figure 5.9: Correlation Estimation of the first 5 principal components as a function of distance from the original observations.

5.4.1 Comparison of reconstruction between two neighbourhood selection methods

In this section, we compare the curve reconstruction for the two neighbourhood selection radius and neighbouring locations. We previously showed how the neighbours of an observation are chosen using the two selection methods.

In order to investigate the difference in their performance, we apply SPACE to temporally sparse data, where the input consists of 10 points per curve instead of 46 points. SPACE performs a gap filling task on the missing points and returns the curves with all 46 points by using both the specified neighbour pixels values at these missing points and the values of the other points from the same curve. Figure 5.10

shows the fitted line for a single observation when using different neighbourhood selections, where the red and green fitted lines illustrate the reconstructed curves from only 10 points using radius and neighbouring measures, respectively.

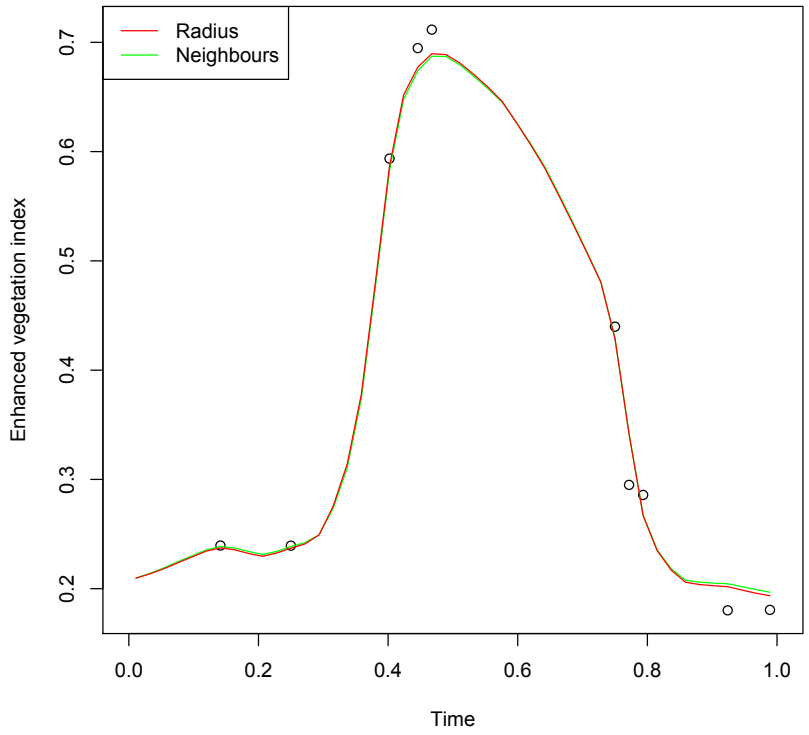

Figure 5.10: Reconstructed curve using the two neighbourhood selection

We also examine our method in the case of small size samples, where the number of the neighbours for each pixel is small. Table 5.2 shows the RSME values of different samples size using the two neighbourhood selections.

Data	RSME
All data observations (radius)	0.0207
All data observations (neighbours)	0.0213
Sample of 200 observations (radius)	0.0215
Sample of 200 observations (neighbours)	0.0216
Sample of 50 observations (radius)	0.0311
Sample of 50 observations (neighbours)	0.0277

Table 5.2: RSME values of different reconstructed data sets

RSME values are very similar even when the data size is small, which indicates that our new method to select the locations used in the calculations gives very similar curve reconstruction. However, the radius idea is simpler to use as an algorithm and can be easily generalised to other applications, even to datasets with non-gridded spatial observations.

5.5 Application of ST-PDE on EVI Data

ST-PDE approach has already been applied to sampled data with few spatial points over a complex boundary. However it is not well understood how ST-PDE will perform on gridded data set with high number of spatial points. Additionally, we examine the computation challenges and the method's ability to accommodate irregular boundaries domain on gridded data sets.

We apply ST-PDE to two different spatial domain sets of the EVI data. One using the data observed over a grid (all locations) and the other one using an irregular spatial domain of the data. First, we build a triangular mesh from the observation coordinates for the two cases. Figure 5.11 shows the mesh for data observed over regular domain in the left panel and data observed over irregular shaped domain in the right panel.

The mesh consists of triangles, where each vertex of these triangles is an original data point location, and some segments that define the domain boundary. In our case the domain boundaries are contained in the triangular mesh.

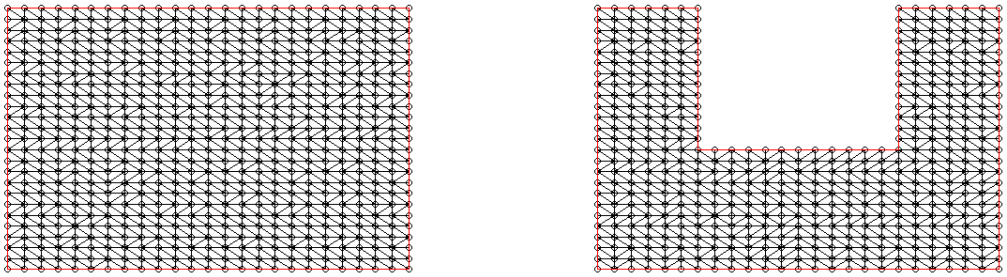

Figure 5.11: Right panel: a triangular mesh of the EVI data observed over regular spatial domain. Left panel: a triangular mesh of the EVI data observed over irregular spatial domain. Each vertex of the triangles represent an original data point location.

Then we estimate the space and time basis functions. Finite element basis, which is described in Section 3.4, represents the space basis. We use an order 1 finite element basis which indicates that each triangle is modelled using linear polynomial function. On the other hand, cubic b-spline basis functions are used as temporal basis functions. The ST-PDE approach is applied to the data using the package `"fdaPDE"` in R (Lila et al., 2016).

Figure 5.12 and 5.13 show the estimated spatio-temporal surface of the EVI data at fixed time points for the two cases regular and irregular spatial domain. As expected, the level of the EVI is higher in the middle of the year and less in other time points. Furthermore, the estimated surfaces show that the data are strongly correlated in space. Visually there is no clear difference in the spatio-temporal surface estimation between the regular and irregular spatial domain, which might indicate the ST-PDE approach succeed to accommodate the irregular boundaries of the domain and the relative density of the grid.

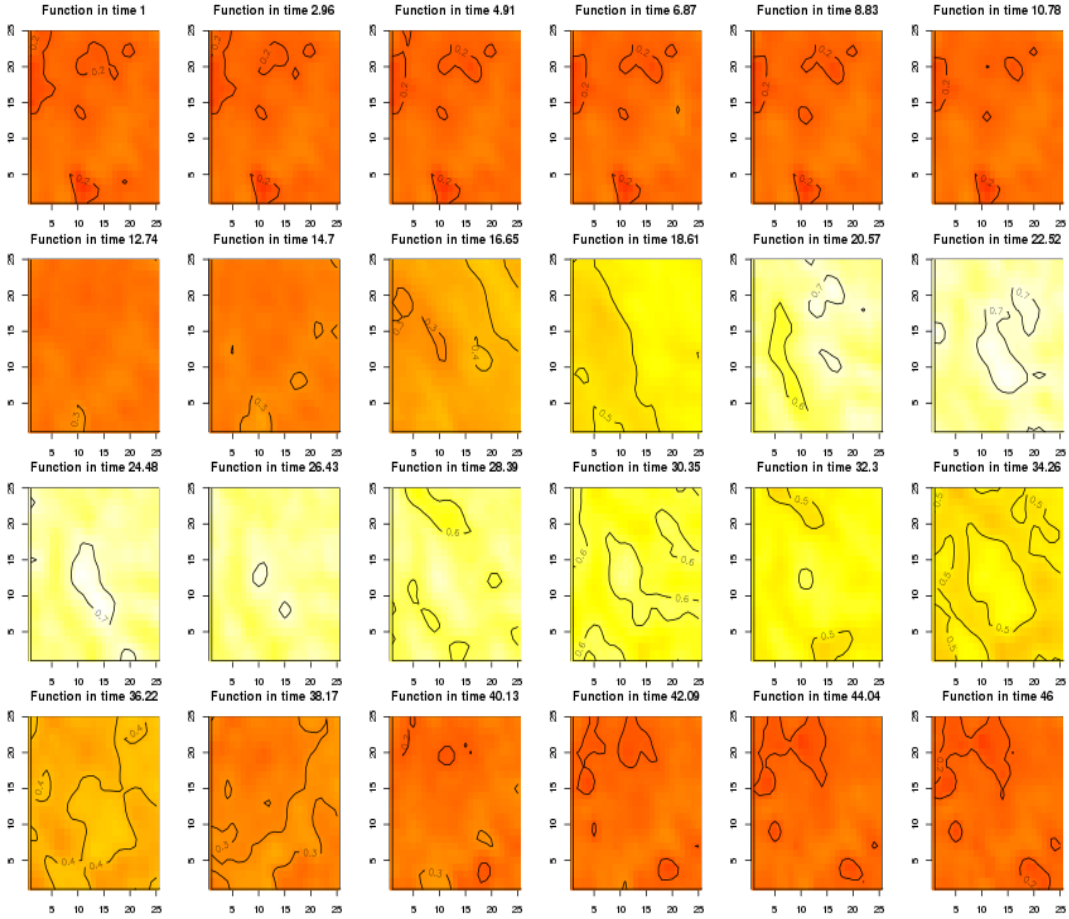

Figure 5.12: Spatio-Temporal surface of regular spatial domain EVI data giving different time points

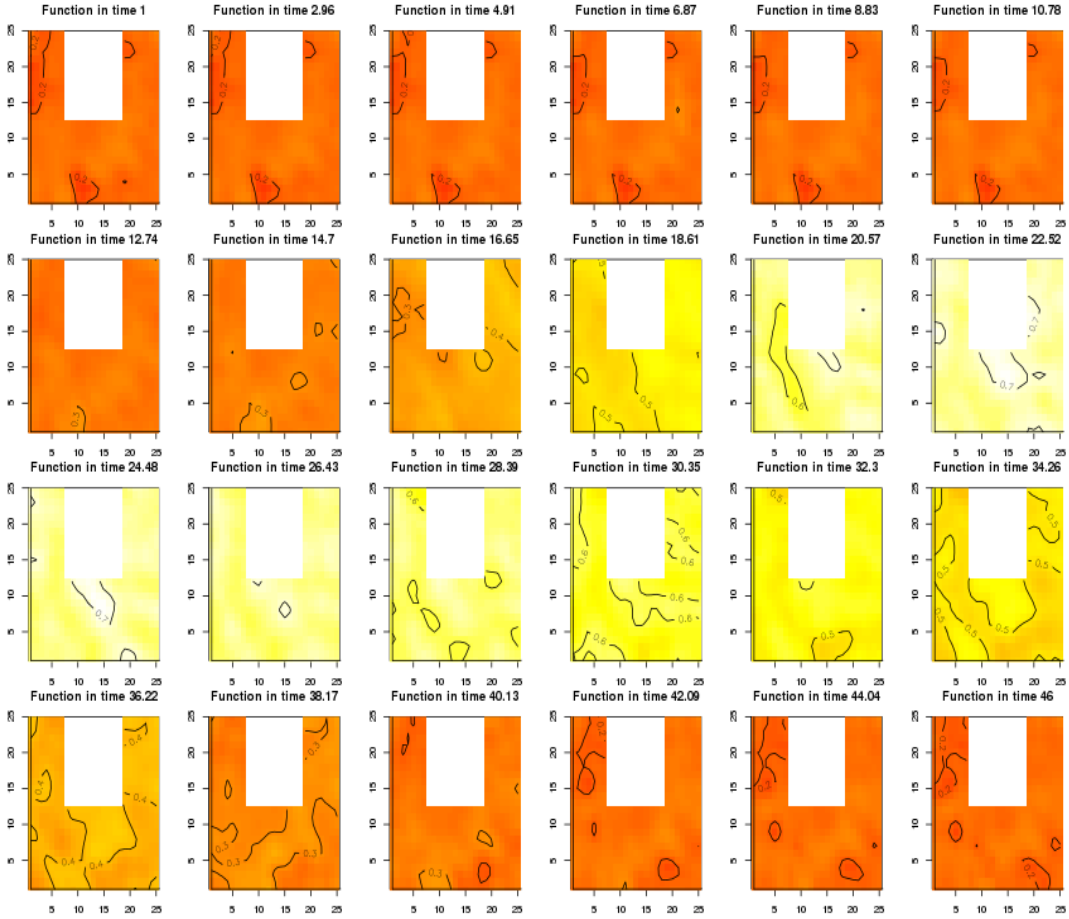

Figure 5.13: Spatio-Temporal surface of irregular spatial domain of EVI data giving different time points

Figure 5.14 displays the time evaluation of the EVI data at different locations where the red points represents the raw data while the line represents the time estimation. The approach succeeded to capture the temporal trend in the data.

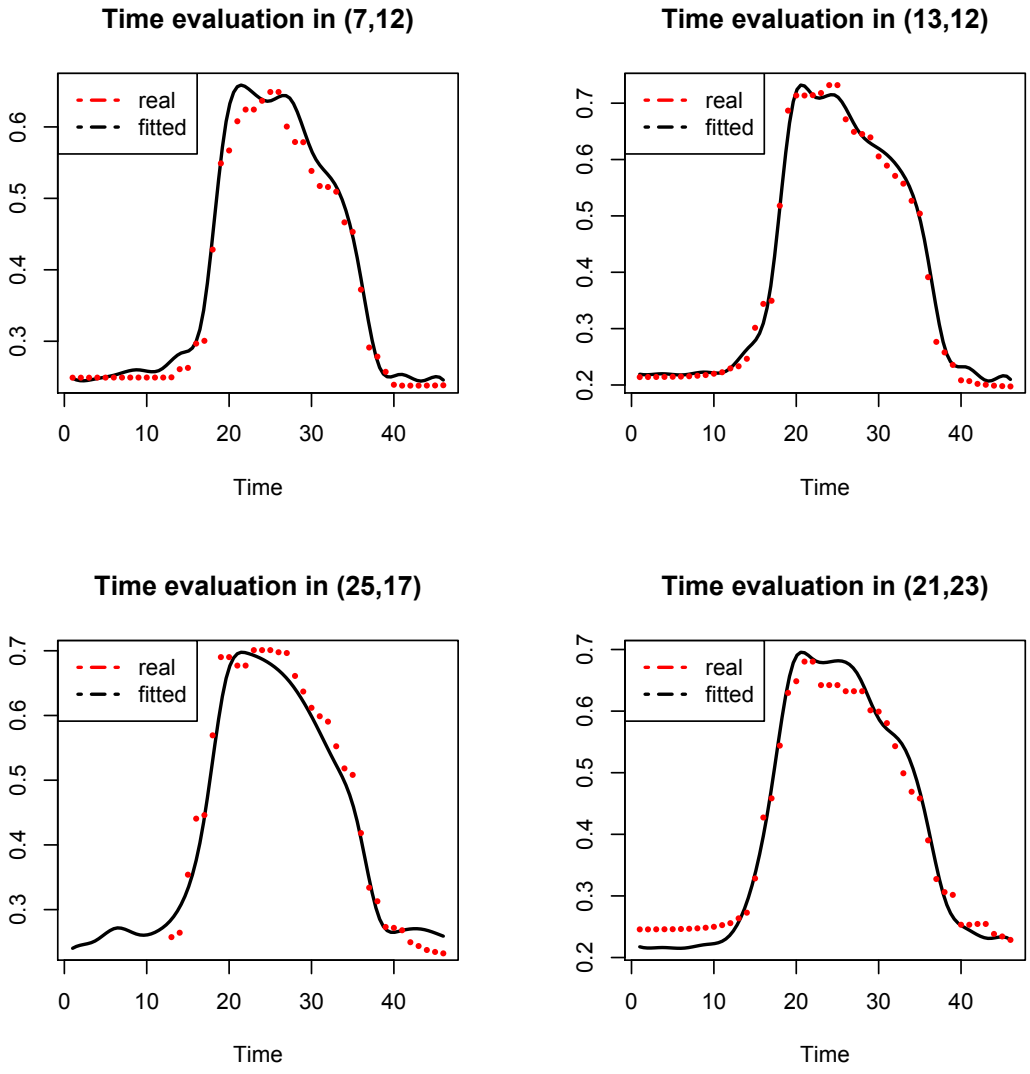

Figure 5.14: Time evaluation curves where the red points represents the raw data while the line represents the time estimation

5.6 Comparison of SPACE and ST-PDE

In this section, we compare the SPACE and ST-PDE performance to analysis the EVI data. We compare the two approaches in term of curves fitted values and root squared mean errors (RSME). Figure 5.15 compares between the fitted lines of one

observation using the two approaches.

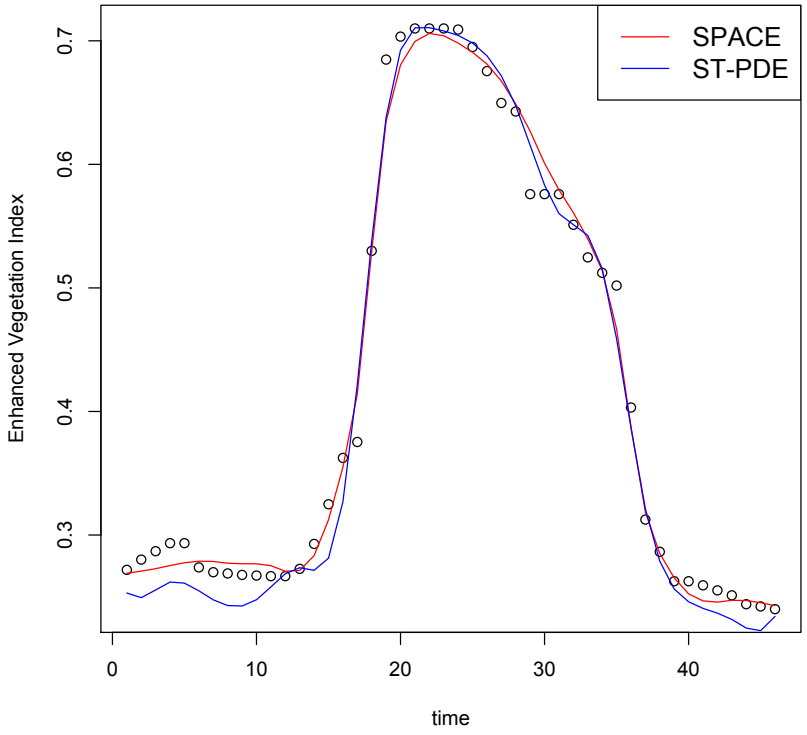

Figure 5.15: SPACE and ST-PDE fitted values for a single curve.

Both of the methods succeeded in fitting the EVI data. However, the Figure shows that the ST-PDE approach underestimate the values in both right and left tails. The root mean squares error (RSME) for SPACE is 0.0207 while RSME for ST-PDE is equal to 0.0224 which is not a large difference between the two approaches. We found that SPACE can be applied to data observed at irregular time points. However, the current codes of ST-PDE can only work for equally spaced time points. In addition, we apply SPACE using R program while, ST-PDE was applied to the data using cluster computing due to issues of computer memory.

5.7 Summary

In this chapter, we have extended the SPACE methodology to accommodate irregular and sparse spatial points through the introduction of the radius based based

distance. We have also applied the ST-PDE approach to dense and regularly gridded data, which posed the challenge of need of large amount of computer memory. These two extensions will now allow us to analyse any spatially correlated functional data by the two competing methods and provide an relative comparison of which method work well in the particular application. We have readily available codes for both extensions which we will provide as an R-package.

Chapter 6

Application to EEG data

6.1 Introduction

One of the most popular applications of functional data analysis is brain data which study the brain activity and provide better understanding of the brain functions. In this chapter, we analyse electroencephalography (EEG) data, introduced in Chapter 4, using our new frameworks of SPACE and RST-PDE approaches and use them to classify images. First we provide some exploratory analysis to explore the data set. Then we show the result of applying our developed SPACE approach in section 3. Section 4 illustrates how RST-PDE can be applied to analyse and classify the data. The final part of this chapter reviews some classification techniques and illustrate the results of applying these techniques to the EEG data set. **Note:** This chapter is adopted from: Alghamdi,S. and S.Ray. Classifying replicated spatially correlated functional data (2019). (Under preparation)

6.2 Exploratory analysis

First, we provide an exploratory analysis to highlight the main features of the dataset and to obtain primary knowledge of the data structure. The data consist of EEG measurements for 18 subjects recorded from 57 scalp electrodes (location) over 454 time points. The measurements for each subject are recorded 250 times, while the

overall data dimension is given by $(18 \times 57 \times 454 \times 250)$. As the data set is very large and complex, it is difficult to explore the data visually. We provide Figure 6.1 to show the complexity of the data. The plot shows the EEG measurements of a single subject viewing a set of 125 car images recorded from 4 locations of the brain. Each colour corresponds to one location (electrode), while the four dark lines represent the means of the replications of each location. The data are difficult to interpret visually from the plot due to the large number of observations.

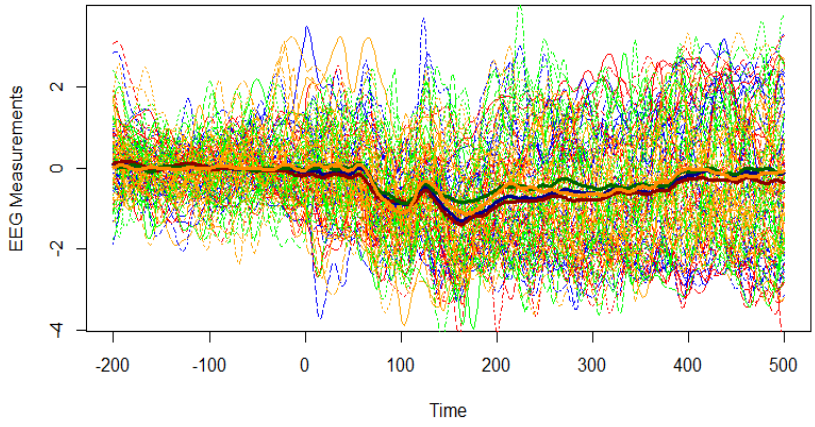

Figure 6.1: EEG measurements of one subject viewing a set of 125 car images recorded from 4 locations. Each colour corresponds to one location while the four dark line represent the mean of the replication of each location.

Another way to summarise the data is the following. We calculate the means over the replications of each location for one subject, once when the subject is seeing car images and the other time when the subject is seeing face images. Figure 6.2 shows the means of the EEG measurements for each location, the variation looks higher when the subject is seeing face images. However, taking the average over the replication might lead to ignoring some variability in the data and provide inaccurate information for future analysis.

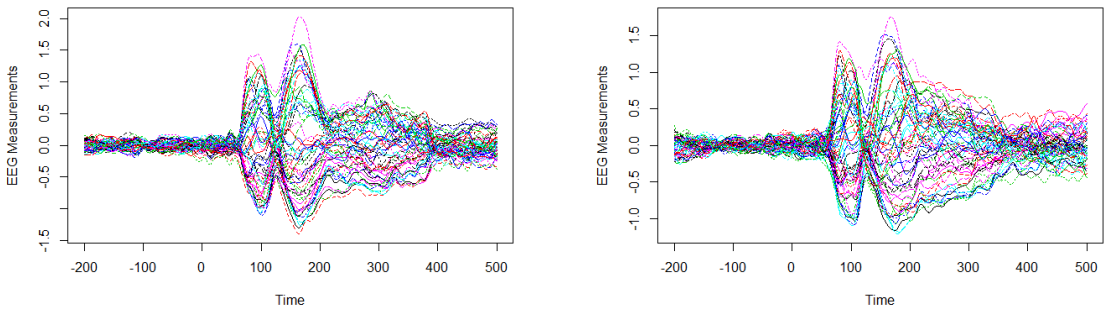

Figure 6.2: EEG measurements of one subject where each curves represents the mean over the 125 replications for all 57 locations. Left panel is for subject seeing car images while right panel is for the subject seeing face images

6.3 Application of SPACE on EEG Data

In this section, we show the results of applying SPACE approach to the EEG, which is non-gridded data, so we use our newly developed radius neighbourhood selection approach introduced in Chapter 5. As the SPACE method is not designed for replicated data, we only use one replication for each location. Then, we apply SPACE to two sets of the EEG data separately, one set for subject seeing one image of car and the other set for the same subject seeing one image of face. Each of the two data set consists of 57 spatial location measurements observed over 454 time points $\{x_{ij}, 1 \leq i \leq 57, 1 \leq j \leq 454\}$.

Figure 6.3 shows the smooth fit after applying SPACE approach to the EEG data on one location from one subject when seeing car image (left panel) and face image (right panel). The plot indicates that the fitted line captures the data pattern.

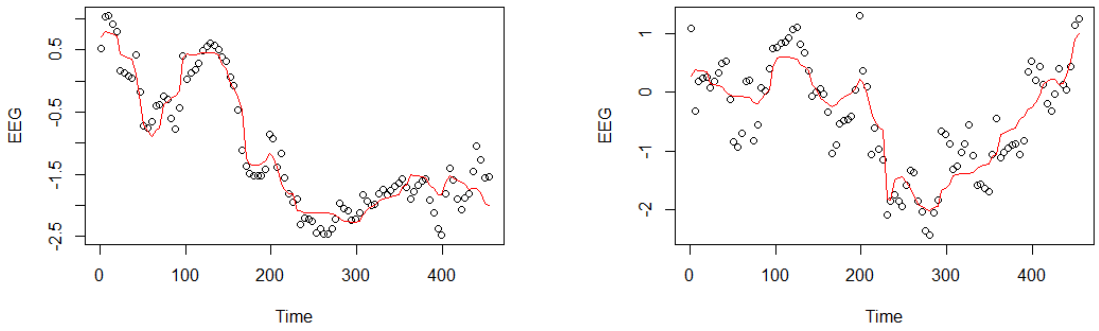

Figure 6.3: EEG measurements and its functional smooth fit from one location from one subject when seeing car image in the left panel and when seeing face image in the right panel. The red line represent the fitted line using SPACE.

For comparison purpose, we apply the ST-PDE approach to the same data set. Figure 6.4 shows the difference between SPACE and ST-PDE in fitting the EEG for different locations

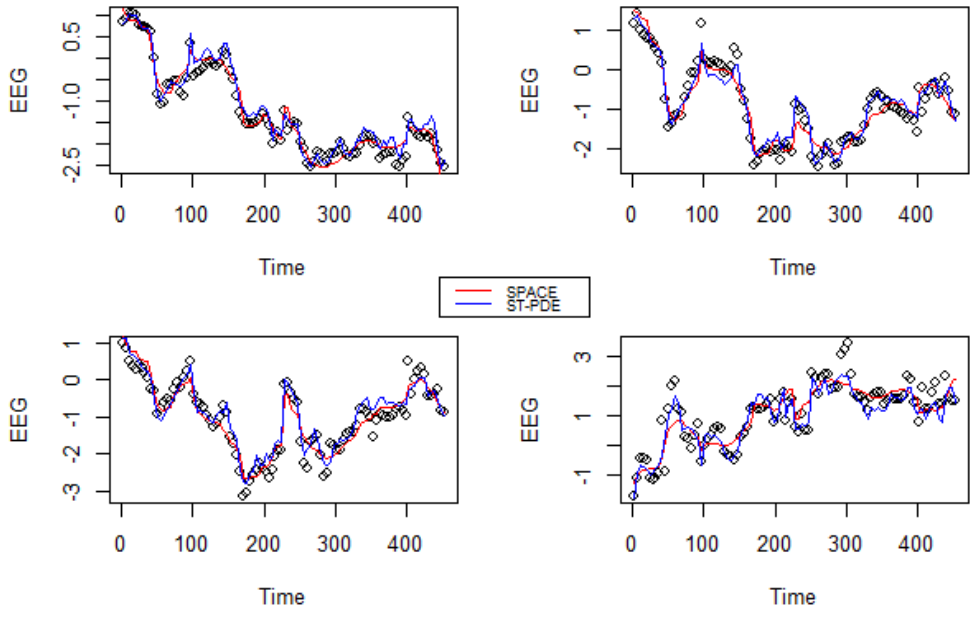

Figure 6.4: EEG measurements from 4 different locations from one subject. Red line represents the fitted line for these location using SPACE approach while the blue line represents the fitted line using ST-PDE approach

In general, both of the estimates are consistent and seem to capture the pattern of the data closely. SPACE over-smooths the data at some time points. However, ST-PDE seem to be more wiggly in general. Note that, one can use cross-validation to choose the smoothing parameter in SPACE and ST-PDE to choose optimal smoothing parameter. We also compared the mean absolute error of the two approaches and they are comparable. The mean absolute error of SPACE equals to 0.367 and the mean absolute error of ST-PDE equals to 0.413.

6.4 Application of replicated ST-PDE on EEG Data

EEG data consist of space-time data for multiple replications for each subject. In RST-PDE, the observations are represented as a vector of length nml where $n = 57$ represents the number of locations, $m = 454$ represents the number of time points and $k = 125$ represents the number of replications. First, we build a triangular mesh using the electrodes locations, where each triangle vertex is a data point location. The triangular mesh is represented in Figure 6.5, where the red line illustrates the spatial domain boundaries.

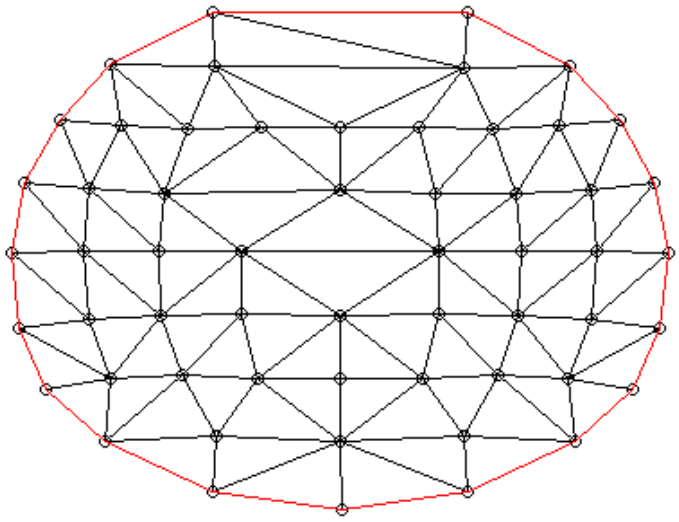

Figure 6.5: A triangulation mesh of the brain electrodes locations

We then estimated the spatial and temporal basis system using a cubic b-splines basis as the time basis function. In contrast, for the spatial part, we use a finite element basis with order one (linear polynomial).

Remark. We apply the RST-PDE codes to the EEG data for one subject seeing car and face images, separately. The process includes big matrices multiplications, Where we faced some problems related to the size and speed to run the program. As a result of that, we modified some codes to reduce the size of the process and increase the speed as it is explained in Chapter 4.

We also use the cluster computing to run the codes. The results for one subject seeing car and face images are summarised separately in Figures 6.6 and 6.7, respectively. The two figures show the spatio-temporal surface of the EEG data across different time points. It can be noted that variability is observed when the subject see face image.

Our next task will be to use the RST-PDE representation of the images for classification. We will compare them with the classification using the raw EEG data in Section 6.6. First we summarise a few classification techniques.

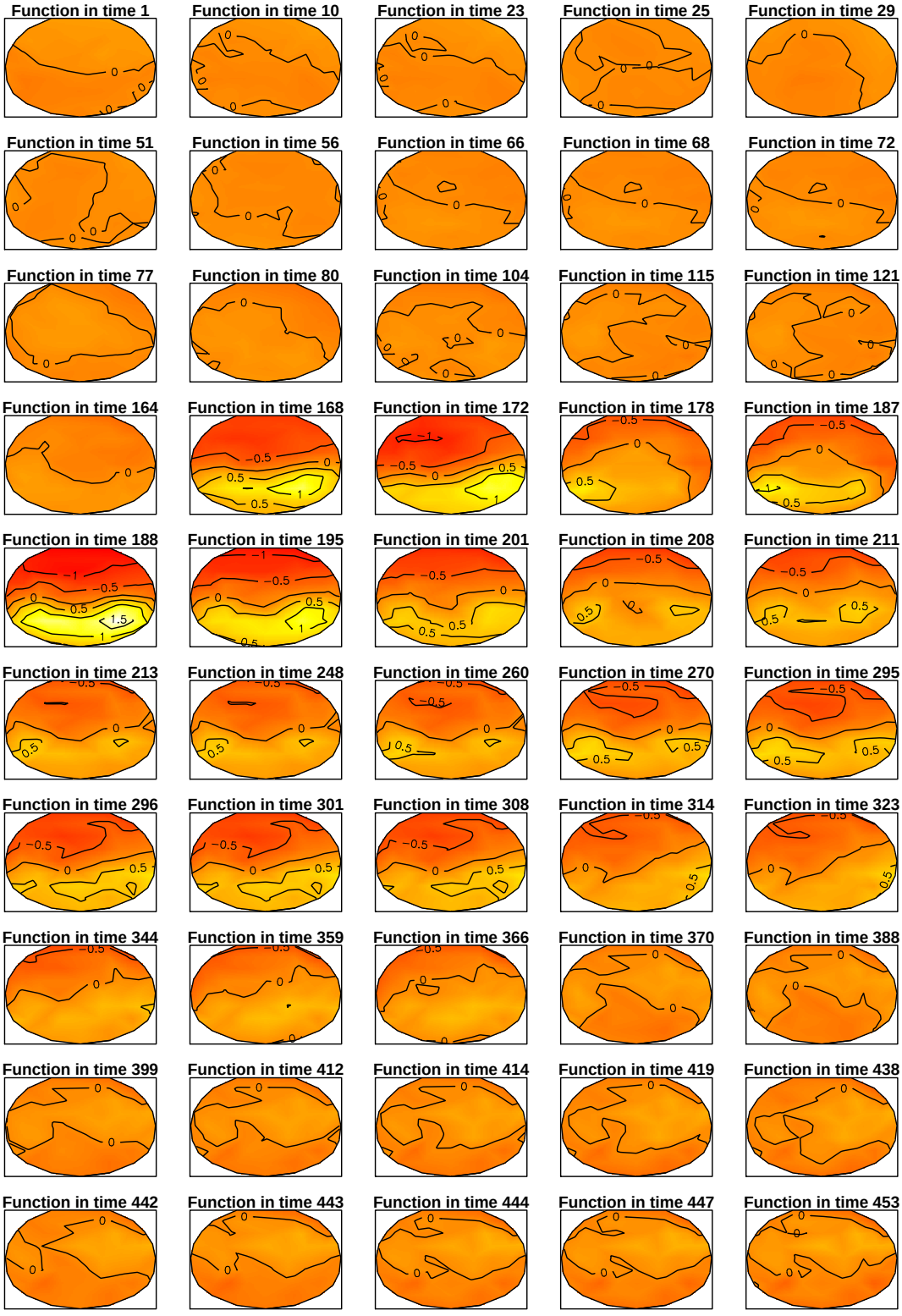

Figure 6.6: Spatio-temporal surface for one subject summarising the 125 replicates of the subject seeing car images

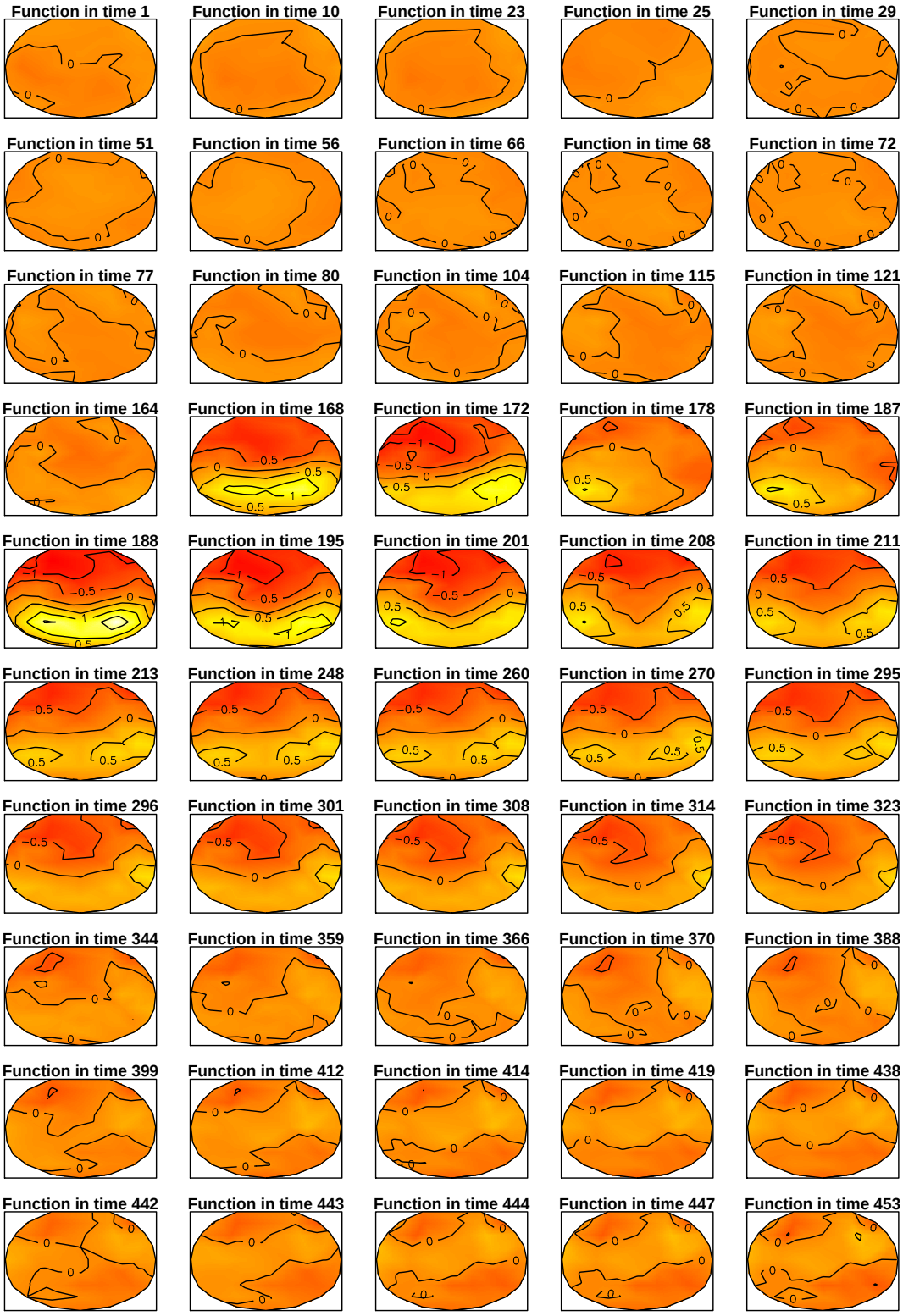

Figure 6.7: Spatio-Temporal surface for one subject summarising the 125 replicates of the subject seeing face images

6.5 Classifications

Statistical classification is an approach that builds predictive models to categorise new objects. First, the data is divided into a training set, which is used to train the models, and a test set which is used to test the accuracy of the models. The training set is used to extract the main features of the observations associated with known groups. Then, the models that include these features are tested in the test set. Once we have models with high accuracy rate then it is possible to predict class labels for new data. This process is also known as supervised learning, where the data teaches the algorithm to categorise future observations. Good references related to classification can be found in Friedman et al. (2001) and Duda et al. (2012).

One of the aims of analysing brain data is to predict from the data if the subject is seeing a car or a face image, which is an example of binary classification. Using the output of the replicated ST-PDE, we apply different classification approaches to the coefficients $\hat{\mathbf{c}}$ vector obtained from RST-PDE. Since we are using the coefficients as our new representation of the data, we can use multivariate classification methods rather than functional classification methods to classify the two classes. In particular, our input vector for each subject observing a specific image is of length 2850 (the number of coefficients from RST-PDE approach) and we have 36 instances, 2 from each of the 18 subjects. The response vector y, taking values 0 or 1, is a vector of length 36, represents the two categories car and face, respectively. In contrast, the raw data for individual replicates has 36×125 observations with a feature vector of 57×454. Instead of summarising over the replicates using RST-PDE one can also consider the ST-PDE representation in which case we will have 36×125 observations of length 2850. In this research, we use some popular classification methods to build a classification model for the EEG data and we use the `"caret"` package in R to implement these methods (Kuhn, 2008).

Many classification approaches are proposed to study the relationship between the observations features and the given classes. However, this section covers three of these methods; support vector mechanism (SVM), K-nearest neighbours (KNN) and random forest (RF).

6.5.1 Support Vector Mechanism (SVM)

Support vector machine is a supervised learning method that is widely used for classification purpose. SVM builds a hyperplane to separate the training data into two classes with maximum margin, which means that the hyperplane should have the largest distance to the nearest input point of each class. In order to understand the idea of SVM, we consider a simple example where there are two classes with a small number of covariates.

Figure 6.8 shows how SVM separate the data into two classes red and green, the classification will then assigns any new point to class blue when the point located above the hyperplane and to the class red when the point fall below the hyperplane.

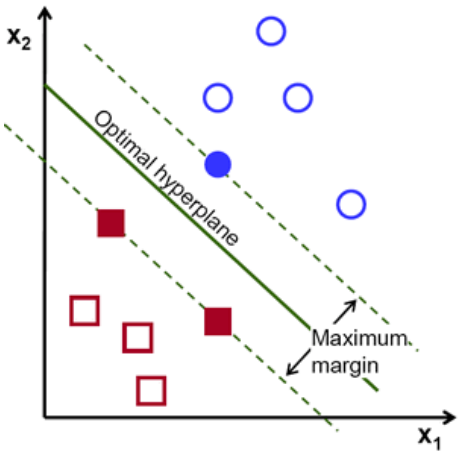

Figure 6.8: An example of SVM for two classes of linear separable data

Although choosing the hyperplane is a key element in performing SVM approach, the problem becomes more difficult in higher dimensions. SVM can be linear or non-linear. However, in our analysis we use linear SVM. The Linear SVM for binary classification seeks a hyperplane, also known as a decision function, to divide the data to two groups. Suppose x_i is the training input and y_i is the class vector, then, the decision function $f(x)$ is given by

$$f(x) = w^T x_i + b \tag{6.1}$$

where w is a weight vector and b is a constant bias term, such that

$$y_i = \begin{cases} 1 \quad \text{if} \quad f(x) > 0, \\ -1 \quad \text{if} \quad f(x) < 0. \end{cases} \tag{6.2}$$

Then the optimal hyperplane is chosen by maximising the marginal distance which is the distance between the hyperplane and the closest points from each class. In order to maximise the distance between the margins we need to minimise $||w||^2/2$. However, a constraint is needed to ensure that the input points do not fall into the margins. Using (6.2) The constraint can be written as follows

$$1 - y_i(w^T x_i + b) \geq 0. \tag{6.3}$$

In some situations, the data are not linearly separable in which case the above approach fails. A simple solution to tackle this problem is allowing a small number of points which are close to the boundary to be misclassified. A cost function is added for each misclassified point, depending on how far it is from meeting the margin restriction in (6.3). In order to apply this cost function, first we need to introduce positive slack variables ξ_i. The constraint with slack variables is written as follows

$$1 - y_i(w^T x_i + b) - \xi_i \leq 0, \tag{6.4}$$

when $\xi_i = 0$ the point is classified. Then we need a penalty term that controls the trade-off of misclassification, where the problem turns into a problem of minimising the following:

$$\frac{||w||^2}{2} + \lambda \sum \xi_i,$$

where λ is a regularisation parameter that controls the trade-off between data goodness of fit and over-fitting. In this case the assumption of linearly separable data points is no longer as strict, which is known as a soft-margin support vector machine.

To solve the minimisation problem, the Lagrange multipliers technique is used. The Lagrange multipliers is a way to find the maximum or minimum of a function when there are some constraints. Let $\alpha_i > 0$ to be the Lagrange multiplier, then the Lagrange is given by

$$L = \frac{||w||^2}{2} + \lambda \sum_i \xi_i + \alpha_i(1 - y_i(w^T x_i + b) - \xi_i). \tag{6.5}$$

To solve (6.5), we need to find the gradient of the Lagrange, we differentiate the Lagrange with respect to w, b and ξ.

$$\begin{aligned} w + \sum_i \alpha_i(-y_i)x_i = 0 \qquad &\Rightarrow \qquad w = \sum \alpha_i y_i x_i \\ &\sum \alpha_i y_i = 0 \\ \lambda - \sum \alpha_i = 0 \qquad &\Rightarrow \qquad \lambda = \sum \alpha_i \end{aligned} \tag{6.6}$$

by substituting 6.6 into the Lagrange in 6.7 we have

$$max \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i^T x_j \tag{6.7}$$

subject to $\lambda \geq \alpha_i \geq 0$ and $\sum_i \alpha_i y_i = 0$. points x_i with non-zero α_i are known as support vectors that are used to determine the hyperplane. The decision function $f(x)$ is given by

$$f(x) = w^T x_i + b = \sum_i \alpha_i y_i x_i^T x + b \tag{6.8}$$

Let t_j be the indices of the support vectors the n we have $w = \sum_j \alpha_{t_j} y_{t_j} x_{t_j}$. Then, for testing a future point z we compute the following

$$f(x) = w^T z + b = \sum_i \alpha_{t_j} y_{t_j} x_{t_j}^T z + b \tag{6.9}$$

we classify the new point z to class 1 if the (6.9) gives positive value and class -1 if the value is negative. More information on support vector machines (SVM) can be found in (Gunn et al., 1998).

6.5.2 K-Nearest Neighbours

K-Nearest neighbours (KNN) is a non-parametric method which is widely used in classifications. KKN technique is considered as one of the simplest classification algorithms as there is no training and it does not compute decision boundaries. It is an instance-based learning technique, it chooses to memorise the training instances which is used as a prior information for the predication instead teaching the algorithm the model. The data point in KNN is classified depending on the majority vote of its k nearest neighbours points.

In the KNN approach, two main factors need to be chosen before performing the algorithm. First, the number k of neighbours to be used, to controls the volume of the neighbourhood has to be determined. However, k can be determined by using cross-validation (CV), which calculates the misclassification rate for different k and chooses the one with the lowest misclassification rate. The second factor is the distance measure that determine the distance between the observations. There are multiple measures of the distances between the points such as Euclidean distance, Manhattan distance and Minkowski distance. Among all these distance measures Euclidean distance is the mostly common choice to measure the distance between the points. The Euclidean distance between two points (x_1, y_1) and (x_2, y_2) is given by

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

The Euclidean distance between these two point equals to the length of the line between them.

Basically, given a positive value k and a new observation x to be classified, the k-nearest neighbor algorithm works as follows

- Compute the distance between x and each training point.
- Locate the k training points which are close to the observation x let us call the set with these points A.
- obtain the most frequent class of A.
- Assign the new observation x to the class we obtain. However, in the case of tie, x is assigned randomly to one of the classes

The k-nearest neighbour approach has many advantages such as; no assumption of the data characteristics is required and it is simple to implement and understand. However, some drawbacks do exist, it is computationally intensive especially when the dataset is very large. Furthermore, in the case of high dimension data it can be less effective because it relies on the closeness between the points.

6.5.3 Random Forest

Random forest approach is another important supervised learning method that is used for data classification (Breiman, 2001). The idea of a random forest is to build multiple decision trees and combine them to make the forest and obtain the predictions. First, we begin by introducing decision trees which the random forest consists of. Decision trees technique is a classifier that breaks down the data into groups based on some features. It provides a graphical representation of the decision, where each node in the tree represents a feature and each branch represents a one of the possible values of the group. Figure 6.9 shows an example of a decision tree taken from (Mitchell, 1997).

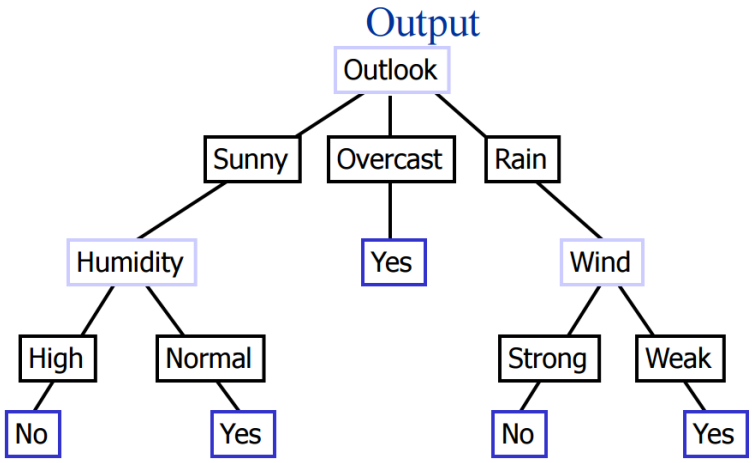

Figure 6.9: An example of decision tree

The example in Figure 6.9, classifies the weather expectation for Saturdays to determine whether these days can be suitable for playing tennis. A data point is classified by starting from the root node of the tree, testing the feature specified by this node, then moving to the branch associated with the value of the feature and this is then repeated with the next node. However, decision trees have some drawbacks. It tends to over-fit the training data which can lead to inaccurate prediction of the outcomes of the unseen data. The over-fitting happen when the model memorises its training data. Random forest overcome this limitation by choosing the root node and the features nodes randomly.

Basically, a random forest consists of a collection of these trees. However it

differs from a decision tree in that it does not include all the dataset in one tree. A subset of both observations features is chosen randomly and then is used to train the model and build the tree. Similarly, a number of different decision trees are grown and each tree will vote for a particular class. Then, these trees are merged together to get the best predication of the class, where the class with maximum number of votes is the predicted class.

The random forest algorithm works as follows;

- Randomly select a subset m variables (features) from the total M variables (features). such that $m < M$.
- Among the m chosen features, calculate the node d using the best split point.
- Split the node d into child nodes using the best split.
- Repeat the previous three steps until a number of nodes has been reached.
- Repeat steps 1 to 4 for n number times to build a forest by creating n number of trees.

While, to classify a new observation we pass the relevant feature of the observation through the rules of each randomly created decision tree to predict the outcome. Then, we calculate the votes for each predicted outcome, where, the high voted predicted class is the final prediction.

Random forests have many advantages. One advantage is that it can handle missing values, and large datasets with high dimensions. another advantage is the classifier can avoid over fitting problem that appear in decision tree method. However, random forests have a major disadvantage that it is time consuming as it takes time to create the decision trees and also to predict the class.

6.6 Comparison of classification results among three representations

In this section we discuss the results obtained by demonstrating the three described classification approaches to three representation sets of the EEG data. The three representation sets are the raw EEG data, the $\hat{\mathbf{c}}$ coefficients of the ST-PDE applied to each replication separately and the $\hat{\mathbf{c}}$ coefficient vector of the RST-PDE approach. We assigned the class of subject seeing a car image by 0 and the subject seeing a face image by 1. We use the `"caret"` package in r to implement the three classification approaches, support vector machine (SVM), random forest (RF) and k-nearest neighbours (KNN) described in Section 6.5.

The first step is to train an SVM model using the whole data set and calculate the model performance. In this step we just want to determine the most important variables in the data that will be used in the final model instead of using all variables. We use a k-fold cross validation approach which involves splitting the data to k-subset where the model is trained using all the $k-1$ subsets and the trained model is then applied to the remaining subset to test its performance. The process is performed for all subsets and for each time the accuracy of the predication is calculated and an overall accuracy vector is determined. We use variable selection to determine the most important features that should be included in the model. The `"caret"` package includes a feature selection method which evaluates the contribution of each feature to the model. We applied the variable selection to the SVM train model, where a loess smoother is fitted between the observations and the variables. Then the R-squared statistic is estimated for each model with the variables against the model with only intercept. Figure 6.10 shows the top 30 variables with the highest values.

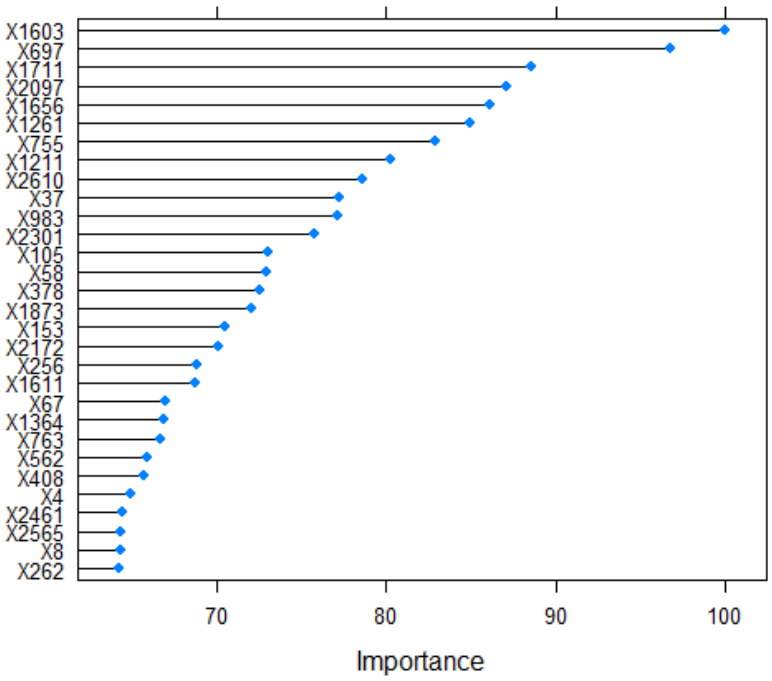

Figure 6.10: Variable importance plot

We use the most important variables in the predictive model. In the previous step we show how we determine the important variables then we apply the three classification methods to the data after variable selection. However, this time we split the data into training and test sets to make sure that the model is tested on data that have never been seen. The models are controlled using two different approaches; repeated k-fold cross-validation and bootstrap. Repeated k-fold cross-validation is performed with the number of folds equal to 5 and repeated 3 times. The process of dividing the data k-fold is repeated 3 times where the model to be used in the predication is the model with highest accuracy. The bootstrap approach is carried out with the number of iteration set to 100 and then we use the model with the highest accuracy. The bootstrap method selects samples randomly to fit the data for each iteration. The modelling process is then repeated 100 times where the test and train sets differ each time. Now we introduce the three data representations that we used in the classification and then show the results of the these representation later on this section.

6.6.1 Three different representations of the original data

We will be using three classification methods support vector machine, random forest and k-nearest neighbours but our main focus is to compare three data representations which are,

- **Raw data:** we use the raw data of all subjects with all replicates. The observations in this set consist of the 18 subjects for the two classes with all replications ($18 \times 2 \times 125 = 4500$). The raw data is very large and might include some noise.
- **ST-PDE:** we use the coefficient vector $\hat{\mathbf{c}}$ of the applying ST-PDE approach to the EEG data to each replication separately. The observations consist of the $\hat{\mathbf{c}}$ vector for 18 subject for the two classes with all 125 replications ($18 \times 2 \times 125 = 4500$). The observations of this set are very large.
- **RST-PDE:** we use $\hat{\mathbf{c}}$ coefficients as our data after applying RST-PDE approach to all subjects. The RST-PDE approach summarises the data over replications so the observations consist of the data for 18 subjects with two classes for each, so the observations length is ($18 \times 2 = 36$). However, we believe that RST-PDE method provides a good summary of the data and retains the important features of the data.

In the next section we show the results of applying classification methods to the three data representations. We also provide classification results of randomly chosen replications for each subject from both raw data and ST-PDE output. This is done to compare the raw data and the ST-PDE output with the RST-PDE output where the sample size is 36.

6.6.2 Classification results of raw data

We apply the classification methods on the raw EEG data and the results are illustrated in Table 6.1. Bold numbers indicate the best performing classification tool.

Control method	classification method	Accuracy mean	Accuracy SD
K-fold cross validation	SVM	**0.575**	0.016
	RF	0.567	0.013
	Knn	0.541	0.009
Bootstrap	SVM	**0.567**	0.011
	RF	0.558	0.008
	Knn	0.530	0.010

Table 6.1: Classification results using raw data all replications for 18 subjects

The accuracy mean is around 0.55 which indicates that the number of correct predications is quiet the same as the number of incorrect predications.

6.6.3 Classification results of ST-PDE output

In this section we show the result of applying classification methods using $\hat{\mathbf{c}}$ obtaine dfrom ST-PDE. Table 6.2 shows the result of using $\hat{\mathbf{c}}$ vector from the ST-PDE approach, the results are similar to raw data results. All three approaches gives accuracy mean around 0.55. Some shows a slight higher accuracy mean than the one in table 6.1, while overall the accuracy mean is low.

Control method	classification method	Accuracy mean	Accuracy SD
K-fold cross validation	SVM	**0.588**	0.020
	RF	0.567	0.141
	Knn	0.545	0.017
Bootstrap	SVM	**0.581**	0.010
	RF	0.563	0.009
	Knn	0.536	0.011

Table 6.2: Classification results using $\hat{\boldsymbol{c}}$ of individual replications for 18 subjects

Using the replications as our observations, we ignore the fact that each 250 replications set comes from one subject and we build the train model from all replications which include large noise.

6.6.4 Classification results of RST-PDE output

The results of applying the three classification approaches to $\hat{\boldsymbol{c}}$ obtained of RST-PDE approach is given in table 6.3

Control method	classification method	Accuracy mean	Accuracy SD
K-fold cross validation	SVM	0.667	0.158
	RF	**0.888**	0.065
	Knn	0.727	0.177
Bootstrap	SVM	0.642	0.108
	RF	**0.834**	0.119
	Knn	0.605	0.131

Table 6.3: Classification results using $\hat{\boldsymbol{c}}$ sumarising all replications for 18 subjects

Table 6.3 shows that all three approach gives high accuracy mean, while, random forest gives the highest accuracy mean at 0.89.

6.6.5 Classification results of randomly chosen samples

In order to comapre between RST-PDE data and the other data represntations raw data and ST-PDE data, we choose random replication from each subject. Then, the data will be same size as RST-PDE data. Table 6.4 shows that results of one randomly chosen replicate raw data, where the accuracy mean are high where all above 0.65. Random forest gives the highest accuracy mean at 0.87 comparing to support vector machine and k-nearest neighbour.

Control method	classification method	Accuracy mean	Accuracy SD
K-fold cross validation	SVM	0.664	0.084
	RF	**0.864**	0.032
	Knn	0.727	0.090
Bootstrap	SVM	0.654	0.079
	RF	**0.817**	0.030
	Knn	0.672	0.078

Table 6.4: Classification results using raw data of one randomly chosen replication for each subject

Table 6.5 indicate that using $\hat{\boldsymbol{c}}$ provide similar results , where all three approaches gives accuracy mean above 0.65. Random forest also provides the higher accuracy mean.

Control method	classification method	Accuracy mean	Accuracy SD
K-fold cross validation	SVM	0.657	0.067
	RF	**0.853**	0.045
	Knn	0.699	0.083
Bootstrap	SVM	0.639	0.067
	RF	**0.828**	0.028
	Knn	0.645	0.075

Table 6.5: Classification results using $\hat{\boldsymbol{c}}$ of one randomly chosen replication for each subject

Generally, using one replication provides better results than using the whole data with large noise. However, it is inefficient to through all data and use just one replication from each class across all subject as this can lose some important information in the data.

6.6.6 Comparison of classification results

Using all replications provides low accuracy rate as the data include large noise. Using one replication and ignoring other replications provides better accuracy rate. However, it cannot be a good representation of the data as we lose some information. As a result, we introduced the replicated ST-PDE approach which summarises and pools the information in the data.

Table 6.6 compares the accuracy rate for all three data represenations using different classification methods. Classification using $\hat{\boldsymbol{c}}$, sumarising all replications, provides the best accuracy rate among the five cases and includes all the data observation in the classification process. Furthermore, the random forest approach performs better than the support vector machine and k-nearest neighbours methos, which can be due the nature of the data. We also compare the compuational time

of the three data representations for each of the classification methods and found that RST-PDE data are much faster to train the model and classify new data (see appendix A).

Control method	classification method	Raw data	ST-PDE	RST-PDE	Random raw	Random ST-PDE
K-fold cross validation	SVM	0.575	0.588	0.667	0.664	0.657
	RF	0.567	0.567	**0.888**	0.864	0.853
	Knn	0.541	0.545	0.727	0.727	0.699
Bootstrap	SVM	0.567	0.581	0.642	0.654	0.639
	RF	0.558	0.563	**0.834**	0.817	0.828
	Knn	0.530	0.536	0.605	0.672	0.645

Table 6.6: Classification results using $\hat{\boldsymbol{c}}$ of one randomly chosen replication for each subject

6.7 Summary

In this chapter we show the results of applying the SPACE and ST-PDE approaches to one replication of the EEG data set. The two approaches perform very similarly on the data and provide a comprehensive picture of the brain functions. Additionally, we apply the RST-PDE approach to the EEG data which pools informational across several replications. Furthermore, we compare classification based on the RST-PDE approach with standard multivariate approaches and found the RST-PDE based classification outperforms existing approaches.

Chapter 7

Conclusion

In this thesis, we have developed a comprehensive framework for analysing spatially correlated functional data. We started by providing a flow chart that can be readily used by a researcher faced with the decision of choosing the most appropriate method for analysing a dataset which can be generally modelled as spatially correlated functional data. Furthermore, we provide case-studies of two datasets: one previously analysed datasets on modelling remote sensing observation on vegetation index (EVI data) and another new dataset on brain imaging (EEG data) and show how the flow chart can be used to decide on the most appropriate method for analysing each of these datasets.

The first approach is spatial principal analysis of conditional expectation (SPACE) which was designed to analyse spatially correlated functional datasets that are observed over a rectangular spatial grid, e.g. a rectangular region on the surface of the earth with observations on regular interval taken by remote sensing satellites. Often points are distributed at irregular intervals over the region of interest and the sampled spatial points are often opportunistic. To accommodate spatial points which do not fall under a regular grid, we have extended the SPACE methodology by generalizing the concept of neighbours by the radius distance. The new neighbour selection method provides comparable results to the original one for the EVI dataset. This modification also makes it possible to apply SPACE to the EEG data which are sampled data and do not fall under a regular grid. The extended SPACE

succeeded in capturing the important features of the brain imaging data.

The second approach is the spatio-temporal regression model with partial differential equations (ST-PDE), which primarily designed to model spatially non-gridded functional data observed over irregular spatial domain. First, we apply the ST-PDE approach to gridded data sets (EVI data), for both regular and irregular spatial domain. ST-PDE provides good results and is very comparable to SPACE even when the spatial domain is irregular.

Though we could compare ST-PDE and SPACE for the EVI data and for each replicate of the braing imaging data, we were unable to combine the information over the replicates using either of these existing approaches. So we developed the new framework of modelling replicated spatially correlated functional data which allowed us to accommodate the 125 replicates of each person for the EEG data set. The main change from ST-PDE is that replicated ST-PDE consists of replicated basis functions that allow us to accommodate the replicates of each sample. We apply replicated ST-PDE to the EEG data and the approach provides a good estimation of both the spatio-temporal surface and the time evaluation curves for each location.

The primary goal of the analysis of the EEG data was to design a classifier that will enable us to predict if a subject is seeing the image of a car or face. However, the original EEG data is very high dimensional and thereby it is difficult for any classifiers to extract the appropriate information needed to build a good classifier. The replicated ST-PDE approach provides an excellent representation of an individuals EEG observations summarized over available replications, which can then be used to build good classifiers. We applied three popular multivariate classification methods to the coefficient vector of replicated ST-PDE model. Among the three classification methods based on the coefficient vector. Random forests provide the highest accuracy rate in predicting a new data set. Moreover, all three classification methods performed better when using the coefficient vector from the RST-PDE fit compared to using the high-dimensional raw EEG data.

7.1 Future work

Though we have provided a broad framework for analysing spatially correlated functional data, not all datasets can be analysed using the SPACE, ST-PDE or RST-PDE framework.

Recall that one of the initial goals of the thesis was to provide a flow chart that determines the processes of analysing an arbitrary spatially correlated functional data. In this thesis, we have mostly generalized methodologies for analysing data that have a separable spatial and temporal components. However, there are some other nodes in the flow chart, focusing on non-separable covariance matrix which we could not explore in this thesis. We conjecture that the methodology used to accommodate replicates to extend the ST-PDE model can be used to accommodate replicates in other separable and non-separable models.

For the classification task, in this thesis we have used a two step classification approach, first computing the coefficient vector of the spatio-temporal model and then using these coefficients to build the classifier. One alternative approach is to build a model based classification tool extending the mixed model framework for functional data analysis similar to Antoniadis and Sapatinas (2007), who provide a functional mixed effect model by modelling both fixed effect and random effect using wavelet decomposition approach.

We will provide the codes used in the thesis as a github repository in the near future.

Bibliography

Abramowitz, M. and I. A. Stegun (1965). *Handbook of mathematical functions with formulas, graphs, and mathematical table*, Volume 2172. Dover New York.

Akaike, H. (1998). Information theory and an extension of the maximum likelihood principle. In *Selected Papers of Hirotugu Akaike*, pp. 199–213. Springer.

Antoniadis, A. and T. Sapatinas (2007). Estimation and inference in functional mixed-effects models. *Computational statistics & data analysis 51*(10), 4793–4813.

Aston, J. A., D. Pigoli, and S. Tavakoli (2015). Tests for separability in nonparametric covariance operators of random surfaces. *arXiv preprint arXiv:1505.02023*.

Augustin, N. H., V. M. Trenkel, S. N. Wood, and P. Lorance (2013). Space-time modelling of blue ling for fisheries stock management. *Environmetrics 24*(2), 109–119.

Azzimonti, L., L. M. Sangalli, P. Secchi, M. Domanin, and F. Nobile (2015). Blood flow velocity field estimation via spatial regression with pde penalization. *Journal of the American Statistical Association 110*(511), 1057–1071.

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

Bernardi, M. S., L. M. Sangalli, G. Mazza, and J. O. Ramsay (2017). A penalized regression model for spatial functional data with application to the analysis of the production of waste in venice province. *Stochastic Environmental Research and Risk Assessment 31*(1), 23–38.

Boente, G., D. Rodriguez, and M. Sued (2010). Inference under functional proportional and common principal component models. *Journal of Multivariate Analysis 101*(2), 464–475.

Bowman, A. W. (1984). An alternative method of cross-validation for the smoothing of density estimates. *Biometrika 71*(2), 353–360.

Bowman, A. W. and A. Azzalini (2014). *R package* `sm`*: nonparametric smoothing methods (version 2.2-5.4)*. University of Glasgow, UK and Università di Padova, Italia.

Braess, D. (2007). *Finite elements: Theory, fast solvers, and applications in solid mechanics*. Cambridge University Press.

Breiman, L. (2001). Random forests. *Machine learning 45*(1), 5–32.

Brenner, S. and R. Scott (2007). *The mathematical theory of finite element methods*, Volume 15. Springer Science & Business Media.

Brillinger, D. R. (1981). *Time series: data analysis and theory*, Volume 36. Siam.

Castro, P., W. Lawton, and E. Sylvestre (1986). Principal modes of variation for processes with continuous sample curves. *Technometrics 28*(4), 329–337.

Chui, C. K. and E. Quak (1992). Wavelets on a bounded interval. In *Numerical methods in approximation theory, Vol. 9*, pp. 53–75. Springer.

Dauxois, J., A. Pousse, and Y. Romain (1982). Asymptotic theory for the principal component analysis of a vector random function: some applications to statistical inference. *Journal of multivariate analysis 12*(1), 136–154.

De Boor, C., C. De Boor, E. U. Mathematicien, C. De Boor, and C. De Boor (1978). *A practical guide to splines*, Volume 27. Springer Verlag New York.

Di, C.-Z., C. M. Crainiceanu, B. S. Caffo, and N. M. Punjabi (2009). Multilevel functional principal component analysis. *The annals of applied statistics 3*(1), 458.

Duda, R. O., P. E. Hart, and D. G. Stork (2012). *Pattern classification*. John Wiley & Sons.

Friedman, J., T. Hastie, and R. Tibshirani (2001). *The elements of statistical learning*, Volume 1. Springer series in statistics New York, NY, USA:.

Green, P. J. and B. W. Silverman (1993a). *Nonparametric regression and generalized linear models: a roughness penalty approach*. CRC Press.

Green, P. J. and B. W. Silverman (1993b). *Nonparametric regression and generalized linear models: a roughness penalty approach*. CRC Press.

Griswold, C. K., R. Gomulkiewicz, and N. Heckman (2008). Hypothesis testing in comparative and experimental studies of function valued traits. *Evolution 62*(5), 1229–1242.

Gu, C. (2013). *Smoothing spline ANOVA models*, Volume 297. Springer Science & Business Media.

Gunn, S. R. et al. (1998). Support vector machines for classification and regression. *ISIS technical report 14*(1), 5–16.

Haas, L. F. (2003). Hans berger (1873–1941), richard caton (1842–1926), and electroencephalography. *Journal of Neurology, Neurosurgery & Psychiatry 74*(1), 9–9.

Hastie, T. and R. Tibshirani (1990). *Generalized additive models*. Wiley Online Library.

Heckman, N. E. and J. O. Ramsay (2000). Penalized regression with model-based penalties. *Canadian Journal of Statistics 28*(2), 241–258.

Henderson, C. R. (1950). Estimation of genetic parameters. In *Biometrics*, Volume 6, pp. 186–187. INTERNATIONAL BIOMETRIC SOC 1441 I ST, NW, SUITE 700, WASHINGTON, DC 20005-2210.

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

Horvath, L. and P. Kokoszka (2012). *Inference for functional data with applications*, Volume 200. Springer Science and Business Media.

Ignaccolo, R., J. Mateu, and R. Giraldo (2014). Kriging with external drift for functional data for air quality monitoring. *Stochastic environmental research and risk assessment 28*(5), 1171–1186.

Jones, M. C., J. S. Marron, and S. J. Sheather (1996). A brief survey of bandwidth selection for density estimation. *Journal of the American Statistical Association 91*(433), 401–407.

Klem, G. H., H. O. LÃijders, H. Jasper, and C. Elger (1999). The ten-twenty electrode system of the international federation. *Electroencephalogr Clin Neurophysiol 52*(3).

Kuhn, M. (2008). Building predictive models in r using the caret package. *Journal of Statistical Software, Articles 28*(5), 1–26.

Lila, E., L. M. Sangalli, J. Ramsay, and L. Formaggia (2016). *fdaPDE: Functional Data Analysis and Partial Differential Equations; Statistical Analysis of Functional and Spatial Data, Based on Regression with Partial Differential Regularizations*. R package version 0.1-4.

Liu, C., S. Ray, and G. Hooker (2017). Functional principal component analysis of spatially correlated data. *Statistics and Computing 27*(6), 1639–1654.

Liu, C., S. Ray, G. Hooker, and M. Friedl (2012). Functional factor analysis for periodic remote sensing data. *The Annals of Applied Statistics*, 601–624.

Locantore, N., J. Marron, D. Simpson, N. Tripoli, J. Zhang, K. Cohen, G. Boente, R. Fraiman, B. Brumback, C. Croux, et al. (1999). Robust principal component analysis for functional data. *Test 8*(1), 1–73.

Marra, G., D. L. Miller, and L. Zanin (2012). Modelling the spatiotemporal distribution of the incidence of resident foreign population. *Statistica Neerlandica 66*(2), 133–160.

Mitchell, T. M. (1997). *Machine Learning* (1 ed.). New York, NY, USA: McGraw-Hill, Inc.

Niedermeyer, E. and F. L. da Silva (2005). *Electroencephalography: basic principles, clinical applications, and related fields*. Lippincott Williams and Wilkins.

Pebesma, E. J. and R. S. Bivand (2005, November). Classes and methods for spatial data in R. *R News 5*(2), 9–13.

R Core Team (2013). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing.

Ramsay, J. O. and B. W. Silverman (2002). *Applied functional data analysis: methods and case studies*, Volume 77. Citeseer.

Ramsay, J. O., H. Wickham, S. Graves, and G. Hooker (2014). *fda: Functional Data Analysis*. R package version 2.4.4.

Ramsay, T. (2002). Spline smoothing over difficult regions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 64*(2), 307–319.

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

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

Rudemo, M. (1982). Empirical choice of histograms and kernel density estimators. *Scandinavian Journal of Statistics*, 65–78.

Sadeghi, N., M. Prastawa, J. H. Gilmore, W. Lin, and G. Gerig (2010). Spatio-temporal analysis of early brain development. In *2010 Conference Record of the Forty Fourth Asilomar Conference on Signals, Systems and Computers*, pp. 777–781. IEEE.

Sangalli, L. M., J. O. Ramsay, and T. O. Ramsay (2013). Spatial spline regression models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 75*(4), 681–703.

Schwarz, G. et al. (1978). Estimating the dimension of a model. *The annals of statistics 6*(2), 461–464.

Scott, D. W. (2015). *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons.

Seshadri, P. (2017). Kronecker product least squares. *arXiv preprint arXiv:1705.08731*.

Sheather, S. J. (2004). Density estimation. *Statistical science*, 588–597.

Sheather, S. J. and M. C. Jones (1991). A reliable data-based bandwidth selection method for kernel density estimation. *Journal of the Royal Statistical Society. Series B (Methodological)*, 683–690.

Silverman, B. and J. Ramsay (2005). *Functional Data Analysis*. Springer.

Silverman, B. W. et al. (1996). Smoothed functional principal components analysis by choice of norm. *The Annals of Statistics 24*(1), 1–24.

Smith, R. L., S. Kolenikov, and L. H. Cox (2003). Spatiotemporal modeling of pm2. 5 data with missing values. *Journal of Geophysical Research: Atmospheres (1984âĂŞ2012) 108*(D24).

Smolka, E., M. Gondan, and F. RÃűsler (2015). Take a stand on understanding: electrophysiological evidence for stem access in german complex verbs. *Frontiers in human neuroscience 9*, 62.

Viviani, R., G. Grön, and M. Spitzer (2005). Functional principal component analysis of fmri data. *Human brain mapping 24*(2), 109–129.

Wahba, G. (1990). *Spline models for observational data*. SIAM.

Wahba, G. and P. Craven (1978). Smoothing noisy data with spline functions. estimating the correct degree of smoothing by the method of generalized cross-validation. *Numerische Mathematik 31*, 377–404.

Wood, S. N., M. V. Bravington, and S. L. Hedley (2008). Soap film smoothing. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 70*(5), 931–955.

Woodroofe, M. (1970). On choosing a delta-sequence. *The Annals of Mathematical Statistics 41*(5), 1665–1671.

Yao, F. and T. Lee (2006). Penalized spline models for functional principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 68*(1), 3–25.

Zipunnikov, V., B. Caffo, D. M. Yousem, C. Davatzikos, B. S. Schwartz, and C. Crainiceanu (2011). Multilevel functional principal component analysis for high-dimensional data. *Journal of Computational and Graphical Statistics 20*(4), 852–873.

Appendix A

A.1 Computational times of simulation study

The analysis of our simulation study was carried on using cluster computing. The data consists of 20 replications. 200 spatial points and 9 time points, and the process repeated 50 times. The computational times of the simulated data analysis are given in table A.1

Method	Elapsed time
RST-PDE	1478
ST-PDE	5227

Table A.1: Computational times of modelling simulated data using RST-PDE and ST-PDE

The modelling process in RST-PDE approach is faster than the ST-PDE .

A.2 Computational times of classification methods given three data representations

We estimated the computational time of applying classification methods to three data representations, raw data, coefficient vector of ST-PDE approach and coefficient vector of RST-PDE approach. Table A.2 shows the elapsed time for each case.

Control method	classification method	Raw data	ST-PDE	RST-PDE
K-fold cross validation	SVM	16.58	15.53	**1.122**
	RF	131.5	164.1	**1.575**
	Knn	4.068	3.598	**0.957**
Bootstrap	SVM	152.8	143.6	**2.092**
	RF	1340	1321	**6.445**
	Knn	33.43	33.28	**2.278**

Table A.2: Computational times of classification methods using three data representations

From the table it is clear that RST-PDE data is faster to classify, which is due to the size of the data. Conversely, raw data and ST-PDE data are very large which result in high computational time