# Development of functional principal components analysis and estimating the time-varying gene regulation network

by

**Yunlong Nie**

M.Sc., University of British Columbia, 2013
B.Sc., Naikai University, 2011

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy

in the
Department of Statistics and Actuarial Science
Faculty of Science

**SIMON FRASER UNIVERSITY**
**Fall 2018**

# Approval

| | |
|---|---|
| **Name:** | **Yunlong Nie** |
| **Degree:** | **Doctor of Philosophy (Statistics)** |
| **Title:** | **Development of functional principal components analysis and estimating the time-varying gene regulation network** |
| **Examining Committee:** | **Chair:** Jinko Graham<br>Professor |
| | **Dr. Jiguo Cao**<br>Senior Supervisor<br>Associate Professor |
| | **Dr. Liangliang Wang**<br>Supervisor<br>Assistant Professor |
| | **Dr. Zhaosong Lu**<br>Internal Examiner<br>Professor<br>Department of Mathematics<br>Simon Fraser University |
| | **Dr. Pang Du**<br>External Examiner<br>Associate Professor<br>Department of Statistics<br>Virginia Polytechnic Institute and State University |
| **Date Defended:** | **September 27, 2018** |

# Abstract

Functional data analysis (FDA) addresses the analysis of information on curves or functions. Examples of such curves or functions include time-course gene expression measurements, the Electroencephalography (EEG) data motoring the brain activity, the emission rate of automobiles after acceleration and the growth curve of children on body fat percentage made over a growth time period. The primary interests for the underlying curves or functions varies in different fields. In this thesis, new methodology for constructing time-varying network based on functional observations is proposed. Several variations of Functional Principal Component Analysis (FPCA) are developed in the context of functional regression model. Lastly, the new use of FPCA are explored in terms of recovering trajectory functions and estimating derivatives.

**Keywords:** Functional Data Analysis; Functional Principal Component Analysis; Functional Regression Model; Time-varying network; Sparse Functional Data; Derivative Estimation

# Table of Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

Functional data analysis (FDA) addresses the analysis of information on curves or functions. Examples of such curves or functions include time-course gene expression measurements, the Electroencephalography (EEG) data motoring the brain activity, the emission rate of automobiles after acceleration and the growth curve of children on body fat percentage made over a growth time period. The primary interests for the underlying curves or functions vary in different fields.

Functional regression model is widely used to study the relationship between a scalar response variable and the functional predictor. One conventional approach under the functional linear regression framework is to first perform functional principal component analysis (FPCA) on the functional predictor and then use the first few leading functional principal component (FPC) scores to predict the response variable. The leading FPCs estimated by the conventional FPCA stand for the major source of variation of the functional predictor, but these leading FPCs may not be mostly correlated with the response variable, so the prediction accuracy of the functional linear regression model may not be optimal. In Chapter 2, we propose a supervised version of FPCA by considering the correlation of the functional predictor and response variable. It can automatically estimate leading FPCs, which represent the major source of variation of the functional predictor and are simultaneously correlated with the response variable. Our supervised FPCA method is demonstrated to have a better prediction accuracy than the conventional FPCA method by using one real application on electroencephalography (EEG) data and three carefully-designed simulation studies. This work has been published in Nie et al. (2018).

Chapter 3 considers a variation of the conventional FPCA method. The main goal of FPCA is to explore major sources of variation in a sample of random curves. These major sources of variation are represented by FPCs. The FPCs from the conventional FPCA method are often nonzero in the whole domain, and are hard to interpret in practice. In this paper, we consider the problem of estimating FPCs, which are only nonzero in subregions. The resulting sparse FPCs not only represent the major variance resources but also can be used to identify the subregions where those major variations exist. The current methods obtain sparse FPCs by adding a penalty term on the length of nonzero regions of FPCs in the conventional eigendecomposition framework. However, these methods become an NP-hard optimization problem. To overcome this issue, we propose a novel regression framework to estimate FPCs and the corresponding optimization is not NP-hard. We also show that the FPCs estimated with our proposed sparse FPCA method is equivalent to the FPCs using the conventional FPCA method when the sparsity parameter is zero. Simulation studies illustrate that the proposed sparse FPCA method can provide more accurate estimates for FPCs than other available methods when those FPCs are only nonzero in subregions. The proposed method is demonstrated by exploring the major variations among the acceleration rate curves of 107 diesel trucks, where the nonzero regions of the estimated sparse FPCs are found well separated.

Chapter 4 considers the problem of recovering the underlying trajectory when the longitudinal data are sparsely and irregularly observed and noise-contaminated. More specially, such data are popularly analyzed with functional principal component analysis via the Principal Analysis by Conditional Estimation (PACE) method. The PACE method may sometimes be numerically unstable because it involves the inverse of the covariance matrix. We propose a sparse orthonormal approximation (SOAP) method as an alternative. It estimates the optimal empirical basis functions in the best approximation framework rather than eigen-decomposing the covariance function. The SOAP method avoids estimating the mean and covariance function, which is challenging when the assembled time points with observations for all subjects are not sufficiently dense. The SOAP method avoids the inverse of the covariance matrix, hence the computation is more stable. It does not require the functional principal component scores to follow the Gaussian distribution. We show that the SOAP estimate for the optimal empirical basis function is asymptotically consistent. The finite sample performance of the SOAP method is investigated in simulation studies in comparison with the PACE method. Our method is demonstrated by recovering the CD4 percentage curves from sparse and irregular data in the Multi-center AIDS Cohort Study.

Motived by the MIT Growth and Development study, we consider the problem of estimating the individual growth rate when the measurements are sparse in Chapter 5. A new method is proposed to directly estimate the optimal empirical basis functions for the derivatives of individual trajectories in a best approximation framework. This method is refereed to as the derivative functional empirical component analysis (DeFECA). The novelty of our method is three-fold. First, the estimated empirical basis functions forms the most parsimonious or optimal representation of the derivative functions. Second, our method does not requires the coefficients of the empirical basis functions to be Gaussian distributed. Our method is still applicable when this assumption is invalid or difficult to verify. Third, our method does not require estimating the mean and covariance functions, which may be challenging when the assembled time points with observations for all subjects are not sufficiently dense. Last but not the least, simulation studies shows that our method is numerically stable in comparison to existing methods because our method avoids the inverse of the covariance matrix.

Chapter 6 considers estimating the functional single index with compact support. Functional single index are widely used to describe the nonlinear relationship between a scalar response and a functional predictor. The conventional functional single index model assumes that the coefficient function is nonzero in the entire time domain. In other words, the functional predictor always has a nonzero effect on the response all the time. We propose a new compact functional single index model, in which the coefficient function is only nonzero in a subregion. We also propose an efficient method which can simultaneously estimate the nonlinear link function, the coefficient function and also the nonzero region of the coefficient function. Hence, our method can identify the region in which the functional predictor is related to the response. Our method is applied to predict the total number of daily bike rentals based on the hourly temperature data. The finite sample performance of the proposed method is investigated with a simulation study in comparison with the conventional functional single index model.

Lastly, we consider the problem of of modeling the dynamical regulation process within a gene network in Chapter 7. We propose to model this dynamical system with a large number of nonlinear ordinary differential equations (ODEs), in which the regulation function is estimated directly from data without any parametric assumption. Most current research assumes the gene regulation network is static, but in reality, the connection and regulation function of the network may change with time or environment. This change is reflected in our dynamical model by allowing the regulation function varying with the gene expression and forcing this regulation function to be zero if no regulation happens. We introduce a statistical method called *functional SCAD* to estimate a time-varying sparse and directed gene regulation network, and, simultaneously, to provide a smooth estimation of the regulation function and identify the interval in which no regulation effect exists. The finite sample performance of the proposed method is investigated in a Monte Carlo simulation study. Our method is demonstrated by estimating a time-varying directed gene regulation network of 20 genes involved in muscle development during the embryonic stage of Drosophila melanogaster. This work has been published in Nie et al. (2017).

# Chapter 2

# Supervised Functional Principal Components Analysis

## 2.1 Introduction

In this chapter, we study the problem of predicting a scalar response $Y$ using the following functional linear model

$$E(Y|X(t)) = g\Big(\beta_0 + \int_{\mathcal{T}} \beta(t)\{X(t) - \mu(t)\}dt\Big) \tag{2.1}$$

where $\beta_0 \in \mathbb{R}$ is the intercept, $X(t)$ is the functional predictor process with the mean function $\mu(t)$, $\beta(t)$ is the slope function, and both $\beta(t)$ and $X(t)$ are assumed to be smooth and square integrable on the domain $\mathcal{T}$. The link function $g$ is assumed to be monotonic and invertible. The form of $g$ is chosen based on the distribution assumption on $Y$. For instance, $g$ is usually chosen as the inverse logit function if $Y$ is a binary variable.

The above functional linear model has been widely used to link a scalar response with an integral form of a functional predictor. Compared with the classic regression problem in which only scalar predictors are considered, the main challenge in this functional linear model is that even a single functional predictor can lead to a saturated model due to its high flexibility. A common strategy to address this problem is through the functional principal component analysis (FPCA). The FPCA method estimates the functional linear model (2.1) in two steps: estimating the functional principal components (FPCs) for the functional predictor; and then using several leading FPCs in the functional linear model. This topic has been extensively studied in the literature such as Ramsay and Silverman (2002); Yao et al. (2005a) and Huang et al. (2009). Furthermore, functional linear models have been naturally extended to generalized functional linear regression when the response variable is binary or multinomial. For example, Ratcliffe et al. (2002) applied a functional logistic regression to predict the high-risk birth rate based on periodically stimulated foetal heart rate tracings. Müller and Stadtmüller (2005) related the response with the integral form of a functional predictor through a smooth function. Cardot et al. (2003) used a multinomial functional regression model to predict the land usage based on the temporal evolution of coarse resolution remote sensing data. A more general approach is proposed by Du and Wang (2014) in which the response is assumed to follow an exponential distribution. A penalized lilkelihood method is used to estimate the unknown intercept and coefficient function and the resulting penalized likelihood estimator is shown to attain the optimal rate of convergence. The above functional linear model could also be extended to include both parametric and nonparametric components (Du et al., 2012).

However, a common limitation of the above methods is that the estimation of FPCs in the first step is totally separated from the regression model used to predict the response variable $Y$ in the second step. In the first step, the leading FPCs mainly focus on explaining the maximum variation of the functional predictor. Thus, the estimated FPCs may not have the maximum prediction power for $Y$. Therefore, practitioners usually have to include as many FPCs as possible to fit the functional regression, which introduces excessive variability into the model, especially when the sample size is relatively small. Our goal is to borrow the information from the response variable $Y$ to estimate FPCs in the first step such that the resulting FPCs have a better performance in terms of predicting $Y$. This strategy is called supervised FPCA in this manuscript.

Bair et al. (2006) introduced a supervised principal component analysis (PCA) method in the context of classic multivariate regression problem, especially when the number of predictors was much larger than the sample size. They proposed a latent variable framework in which the response variable is only associated with a subset of predictors through a latent variable. More specifically, their method consists of three steps: first a pre-screening procedure is employed to select those important predictors; then PCA is performed on those selected predictors to estimate the latent variable; finally a regression model is fitted with those estimated PC scores. Li et al. (2016) proposed another version of supervised PCA, namely, a supervised singular value decomposition (SupSVD) model. Unlike Bair et al. (2006) focusing on predicting the response variable $Y$, the primary interest of the SupSVD model is to recover the underlying low-rank structure of the predictor matrix with the supervision information from $Y$. In addition, Li et al. (2016) could incorporate a multi-dimensional response variable whereas Bair et al. (2006) only considered a single scalar response variable.

However, neither of the above work can accommodate functional predictors. The extension from supervised PCA to functional data is nontrivial. Recently, Li et al. (2016) extended the SupSVD model to functional principal component analysis (FPCA) and proposed a method called supervised sparse functional principal component (SupSFPC). They assume that the supervision data drive low-rank structures of the functional data of primary interest. The estimation procedure is based on the penalized likelihood function that imposes a smooth and sparsity penalty on PC loadings. The difference between our work and theirs is that we mainly focus on improving the prediction performance of FPCs, while Li et al. (2016) focused on recovering the true FPCs.

The novelty of this chapter is three-fold. Firstly, we propose a framework to utilize the scalar response variable, either continuous or categorical, to boost the prediction performance of the estimated FPCs. Our method is particularly useful in dealing with 'Large $p$, Small $n$' problem when multiple functional predictors exist. Secondly, unlike Bair et al. (2006) which employs three steps, our approach does not require a pre-screening procedure. Thirdly, our estimation algorithm is based on eigenvalue decomposition which is much easier to implement in comparison with the revised EM algorithm used by SupSFPC. An R package "sFPCA" is developed to implement our proposed supervised FPCA method and is available at `https://github.com/YunlongNie/sFPCA`.

The rest of this chapter is organized as follows. A review of conventional FPCA analysis is given in Section 2.2. Details of our method is described in Section 2.3. Then we show one real data application on electroencephalography (EEG) data in Section 2.4. Three carefully-designed simulation studies are used to evaluate the finite sample performance of our proposed method in Section 2.5. Section 2.6 provides concluding remarks.

## 2.2 Estimating Functional Linear Models using FPCA

We first introduce the conventional FPCA method for estimating the functional linear model (2.1), which is also called unsupervised FPCA method in this chapter. Consider a stochastic process $X(t)$ on the domain $\mathcal{T}$ with the mean function $E(X(t)) = \mu(t)$. Using the Karhunen-Loève expansion Fukunaga and Koontz (1970), the stochastic process $X(t)$ can be expressed as

$$X(t) = \mu(t) + \sum_{j=1}^{\infty} \alpha_j \xi_j(t), \quad i = 1, \ldots, n, \tag{2.2}$$

where $\xi_j(t), j = 1, \ldots, \infty$, are standard orthogonal to each other and are also called functional principal components (FPCs), and $\alpha_j$ is called the $j$th FPC score. The FPC score $\alpha_j$ are uncorrelated random variables with mean 0 and variance $\lambda_j$. It can also be calculated as $\alpha_j = \int_{\mathcal{T}} (X(t) - \mu(t))\xi_j(t)dt$. For the rest of this chapter, we assume $\mu(t) \equiv 0$ without loss of generality.

In practice, we usually select the first several leading FPCs to approximate each random curve $X(t)$. Here we denote the number of FPCs chosen as $p$ and we will discuss how to determine $p$ later in this manuscript. Then the representation in (2.2) reduces to

$$X(t) = \sum_{j=1}^{p} \alpha_j \xi_j(t) = \boldsymbol{\alpha}^T \boldsymbol{\xi}(t), \tag{2.3}$$

in which $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_p)^T$ and $\boldsymbol{\xi}(t) = (\xi_1(t), \xi_2(t), \ldots, \xi_p(t))^T$. Substituting (2.3) into (2.1) gives

$$\begin{aligned} E(Y|X(t)) &= g\Big(\beta_0 + \int_{\mathcal{T}} \beta(t)\Big(\sum_{j=1}^{p} \alpha_j \xi_j(t)\Big)dt\Big) \\ &= g\Big(\beta_0 + \sum_{j=1}^{p} \alpha_j \int_{\mathcal{T}} \beta(t)\xi_j(t)dt\Big). \end{aligned} \tag{2.4}$$

In the meanwhile, we can also express $\beta(t)$ as a linear combination of the $p$ leading FPCs as

$$\beta(t) = \sum_{j=1}^{p} \gamma_j \xi_j(t) = \boldsymbol{\gamma}^T \boldsymbol{\xi}(t), \tag{2.5}$$

in which $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, \ldots, \gamma_p)^T$ is an unknown coefficient vector to be estimated from the data. Plugging (2.5) back into (2.4), we have

$$E(Y|X(t)) = g\Big(\beta_0 + \boldsymbol{\alpha}^T \boldsymbol{\gamma}\Big). \tag{2.6}$$

We can estimate the unknown coefficient vector $\boldsymbol{\gamma}$ by regressing $Y$ on the FPC scores $\boldsymbol{\alpha}$.

Determining an appropriate value of $p$ is a difficult task in practice. One common strategy is first choosing a large value of $p$ such that the leading $p$ FPCs in (2.2) together explain more than 99% of the total variation. More formally,

$$p = \inf\{k : \frac{\sum_{j=1}^{k} \lambda_j}{\sum_{j=1}^{\infty} \lambda_j} \geq 99\%\}.$$

If the resulting $p$ is too large compared to the sample size, one of the popular shrinkage techniques such as LASSO and SCAD can be employed to do the variable selection. However, this procedure's prediction performance is still not satisfying in many complex problems due to three reasons. First, the prediction power of those FPCs might not coincide with the amount of variation they account for. For instance, the response variable might only depend on the 10th FPC instead of any of the first 9 FPCs; Second, given a small sample size, a large value of $p$ introduces excessive variability into the model, making the model selection a very difficult task. Particularly, in practice when multiple functional predictors exist in the model, even with only a small number of FPCs selected for each functional predictors, this large-$p$-small-$n$ problem is still quite common. Thus, there is necessity to boost the prediction power of the estimated FPCs for each functional predictor.

## 2.3 Method

We first consider the scenario when the response variable is continuous and then extend to the case in which the response variable is binary.

### 2.3.1 Supervised FPCA

Without loss of generality, we assume $E(X(t)) = 0$ and $E(Y) = 0$ in the following discussion. One can always centralize $X(t)$ and $Y$ to satisfy these two assumptions. We propose to estimate FPCs: $\xi_1(t), \xi_2(t), \ldots$, such that the estimate $\hat{\xi}_k(t)$ maximizes

$$Q(\xi) = \frac{\theta\langle \xi, \hat{\mathcal{C}}\xi\rangle + (1-\theta)\texttt{cov}^2(Y, \langle X, \xi\rangle)}{||\xi||^2}, \tag{2.7}$$

subject to $||\xi|| = 1$, $\langle \xi, \hat{\xi}_j\rangle = 0,$ for every $j < k$, and $0 \leq \theta \leq 1$. Here the norm $||\xi|| = \sqrt{\langle \xi, \xi\rangle}$ and $\langle f, g\rangle$ denotes the usual $\mathcal{L}^2$ inner product $\langle f, g\rangle = \int_{\mathcal{T}} f(t)g(t)dt$. In addition, $\hat{\mathcal{C}}$ is denoted as the empirical covariance operator:

$$\hat{\mathcal{C}}\xi = \int_{\mathcal{T}} \hat{C}(\cdot, t)\xi(t)dt,$$

where the empirical covariance function $\hat{C}(s,t) =$
$\frac{1}{n}\sum_{i=1}^{n} X_i(s)X_i(t)$, and $X_i(t)$ is an independent realization of the stochastic process $X(t)$.

We take a closer look at the formalization of $Q(\xi)$ shown in (2.7). The first term in the numerator, $\langle \xi, \hat{\mathcal{C}}\xi\rangle$, represents the variation within the functional predictor $X(t)$ that can be explained by $\xi(t)$; the second part in the numerator, $\texttt{cov}^2(Y, \langle X, \xi\rangle)$, represents the squared covariance between the corresponding FPC score $\langle X, \xi\rangle$ and the response variable $Y$. The balance between these two terms is governed by the weight parameter $\theta$. Apparently, specifying $\theta = 1$ will give rise to unsupervised FPCA. On the other hand, specifying $\theta$ less than 1 will lead to supervised FPCA. The weight parameter $\theta$ can be treated as a tuning parameter and can be determined using cross-validation.

It is also worth mentioning the main rationale behind the 'squared' covariance, the second term on the numerator in (2.7), is two-fold. First, we wish to keep this term, which describes the association between the estimated FPC score and the response variable, of the numerator in equation (8) positive, since the variance of the FPC scores in the first term is always positive. Second, the 'squared covariance' also helps to convert the estimation process into an eigenvalue decomposition problem, which will be illustrated in more details in Section 2.3.3

### 2.3.2 Smooth Supervised FPCA

The FPCs obtained using (2.7) might need to be further smoothed or regularized. We define another type of norm as $||f||_\lambda = \sqrt{||f||^2 + \lambda||\mathcal{D}^2 f||^2}$, in which $\mathcal{D}^2 f = \int_{\mathcal{T}} f''(t)dt$. The smooth estimate for the $k$-th supervised FPC is obtained by maximizing

$$Q(\xi) = \frac{\theta\langle \xi, \hat{\mathcal{C}}\xi\rangle + (1-\theta)\mathtt{cov}^2(Y, \langle X, \xi\rangle)}{||\xi||_\lambda^2}, \tag{2.8}$$

subject to $||\xi||_\lambda = 1$, $\langle \xi, \hat{\xi}_j\rangle = 0$, for every $j < k$, and $0 \leq \theta \leq 1$. The smoothing parameter $\lambda$ controls the degree of smoothness. For instance, when $\lambda = 0$, there is no penalty on the roughness of the estimated component $\hat{\xi}(t)$ and the smooth supervised FPCs will reduce to the regular supervised FPCs discussed in Section 2.3.1. On the other hand, a very large value of $\lambda$ will force the estimated component $\hat{\xi}(t)$ taking a linear form. Moreover, this method is very easy to implement once the smoothing parameter $\lambda$ is determined. In addition, Silverman (1996) showed that under appropriate conditions the estimated FPCs were consistent. In the rest of this section, we will focus on the details of estimating the smooth supervised FPCs, which can be easily applied to unsmooth supervised FPCs by setting $\lambda = 0$.

### 2.3.3 Computational Details

In this subsection, we give the computational details on how to estimate the smooth supervised FPC $\xi(t)$ given a set of value for $(\theta, \lambda)$. To distinguish them, we call $\theta$ and $\lambda$ as the weight and smoothing parameters, respectively. To ease the computation, we use the same B-spline basis functions $\phi_1(t), \phi_2(t), ..., \phi_M(t)$ to represent both the smooth supervised FPC $\xi_j(t)$ and the functional predictor $X_i(t)$, in which $M$ denotes the total number of basis functions. Note that our method is not restricted to B-spline basis system and can be extended to other basis systems as well. Let $\mathbf{\Phi}(t)$ denote the column vector $(\phi_1(t), \phi_2(t), ..., \phi_M(t))^T$, and rewrite $(X_1(t), X_2(t), ..., X_n(t))^T = \mathbf{S\Phi}(t)$, where $\mathbf{S}$ is an $n \times M$ coefficient matrix. In addition, we represent $\xi(t) = \sum_{m=1}^M \beta_m \phi_m(t) = \boldsymbol{\beta}^T \mathbf{\Phi}(t)$, in which $\boldsymbol{\beta}$ denotes the coefficient vector $(\beta_1, \beta_2, ..., \beta_M)^T$. Then the empirical covariance function can be expressed as

$$\hat{C}(s,t) = \frac{1}{n}\mathbf{\Phi}(t)^T(s)\mathbf{S}^T\mathbf{S\Phi}(t).$$

Thus the first term in the numerator of (2.7) is given by

$$\langle \xi, \mathcal{C}\xi\rangle = \frac{1}{n}\boldsymbol{\beta}^T\mathbf{W}\mathbf{S}^T\mathbf{S}\mathbf{W}\boldsymbol{\beta}, \tag{2.9}$$

where $\mathbf{W}$ is an $M \times M$ matrix with elements $w_{ij} = \langle \phi_i(t), \phi_j(t)\rangle$.

As for the second term in the numerator in (2.7), we first derive the form of the FPC score $\langle X_i, \xi\rangle$. For each $X_i(t)$, the FPC score $\langle X_i, \xi\rangle$ is written as

$$\langle X_i, \xi\rangle = \boldsymbol{\beta}^T\mathbf{W}\mathbf{S}_i = \boldsymbol{\beta}^T\mathbf{W}_i,$$

where $\mathbf{S}_i$ is the $i$-th row of the coefficient matrix $\mathbf{S}$, and $\mathbf{W}_i = \mathbf{W}\mathbf{S}_i$. Thus, combining all the scores for each $X_i(t)$

$$(\langle X_1, \xi\rangle, \langle X_2, \xi\rangle, \ldots, \langle X_n, \xi\rangle)^T = \mathbf{S}\mathbf{W}\boldsymbol{\beta}.$$

Finally the covariance term between $Y$ and the FPC score is written as

$$\texttt{cov}(Y, \langle X, \xi\rangle) = \frac{1}{n}\boldsymbol{\beta}^T \sum_{i=1}^{n} Y_i W_i = \frac{1}{n}\boldsymbol{\beta}^T \mathbf{W}\mathbf{S}^T\mathbf{Y},$$

in which $\mathbf{Y} = (Y_1, Y_2, \ldots, Y_n)^T$.

The squared covariance between $Y$ and the FPC score is given as

$$\texttt{cov}^2(Y, \langle X, \xi\rangle) = \frac{\boldsymbol{\beta}^T \mathbf{M}\mathbf{M}^T\boldsymbol{\beta}}{n^2}, \tag{2.10}$$

in which $\mathbf{M} = \mathbf{W}\mathbf{S}^T\mathbf{Y}$.

For the denominator part in (2.7), the norm of $\xi(t)$ is given by

$$||\xi||_\lambda^2 = \boldsymbol{\beta}^T\mathbf{W}\boldsymbol{\beta} + \lambda\boldsymbol{\beta}^T\mathbf{D}\boldsymbol{\beta} = \boldsymbol{\beta}^T\mathbf{G}\boldsymbol{\beta}, \tag{2.11}$$

where $\mathbf{D}$ denotes a $M \times M$ matrix with element
$d_{ij} = \langle \mathcal{D}^2\phi_i(t), \mathcal{D}^2\phi_j(t)\rangle$ and $\mathbf{G} = \mathbf{W} + \lambda\mathbf{D}$.

Putting (2.9), (2.10) and (2.11) together, $Q(\xi)$ in (2.8) is recast into

$$Q(\xi) = \frac{\boldsymbol{\beta}^T\mathbf{U}\boldsymbol{\beta}}{\boldsymbol{\beta}^T\mathbf{G}\boldsymbol{\beta}},$$

where

$$\mathbf{U} = \frac{\theta}{n}\mathbf{W}\mathbf{S}^T\mathbf{S}\mathbf{W} + \frac{1-\theta}{n^2}\mathbf{M}\mathbf{M}^T.$$

Let $\boldsymbol{\delta} = \mathbf{G}^{\frac{1}{2}}\boldsymbol{\beta}$, maximizing $Q(\xi)$ is equivalent to maximizing $\boldsymbol{\delta}^T(\mathbf{G}^{-1/2})^T\mathbf{U}G^{-1/2}\boldsymbol{\delta}$ subject to $\boldsymbol{\delta}^T\boldsymbol{\delta} = 1$. Then $\boldsymbol{\delta}_1, \ldots, \boldsymbol{\delta}_J$ will be the leading $J$ eigenvector of the matrix

$$(\mathbf{G}^{-1/2})^T\mathbf{U}\mathbf{G}^{-1/2}.$$

Consequently, one can derive $\widehat{\boldsymbol{\beta}}_j = (\mathbf{G}^{1/2})^{-1}\boldsymbol{\delta}_j$. The corresponding smooth supervised FPC is $\hat{\xi}_j(t) = \widehat{\boldsymbol{\beta}}_j^T\boldsymbol{\Phi}(t)$ for $j = 1, \ldots, J$.

### 2.3.4 Binary Response Variable

When the response variable $Y$ is binary, we suggest to replace $\texttt{cov}^2(Y, \langle X, \xi\rangle)$ in $Q(\xi)$ defined in (2.8) with the between-group variation of the FPC scores. Formally, let $\mathbf{Y} = (Y_1, Y_2, \ldots, Y_n)^T$, in which $Y_i \in \{0, 1\}, i = 1, \ldots, n$, and $n_j$ is the number of $Y_i$ satisfying $Y_i = j$

for $j = 0, 1$. Let $\boldsymbol{\alpha} = (\alpha_1, \alpha_2 \ldots, \alpha_n)^T$ denote the vector of FPC scores for one FPC $\xi(t)$, in which $\alpha_i = \langle X_i, \xi \rangle$, and $\bar{\alpha}_j = \frac{1}{n_j} \sum_{\{i:Y_i=j\}} \alpha_i$. Since we assume the mean function of the functional predictor, $E(X(t)) = \mu(t) = 0$, the expectation of the FPC score $E(\alpha) = \langle \mu, \xi \rangle = 0$. The between-group variation of the FPC scores is

$$\begin{aligned} R(\xi) &= \sum_{j=0}^{1} n_j (\bar{\alpha}_j - E(\alpha))^2 = n_1 \bar{\alpha}_1^2 + n_0 \bar{\alpha}_0^2 \\ &= \frac{1}{n_1} (\sum_{i=1}^{n} Y_i \alpha_i)^2 + \frac{1}{n_0} (\sum_{i=1}^{n} ((1 - Y_i) \alpha_i))^2. \end{aligned}$$

Note that $\sum_{i=1}^{n} Y_i \alpha_i = \boldsymbol{\alpha}^T \mathbf{Y} = \boldsymbol{\beta}^T \mathbf{W} \mathbf{S}^T \mathbf{Y}$, thus

$$\frac{1}{n_1} (\sum_{i=1}^{n} Y_i \alpha_i)^2 = \frac{1}{n_1} \boldsymbol{\beta}^T \mathbf{M}_1 \mathbf{M}_1^T \boldsymbol{\beta},$$

in which $\mathbf{M}_1 = \mathbf{W} \mathbf{S}^T \mathbf{Y}$. Similarly,

$$\frac{1}{n_0} (\sum_{i=1}^{n} ((1 - Y_i) \alpha_i))^2 = \frac{1}{n_0} \boldsymbol{\beta}^T \mathbf{M}_2 \mathbf{M}_2^T \boldsymbol{\beta},$$

in which $\mathbf{M}_2 = \mathbf{W} \mathbf{S}^T (\mathbf{I}_n - \mathbf{Y})$. Eventually, the between-group variation $R(\xi)$ can be expressed as a quadratic form of $\boldsymbol{\beta}$:

$$\begin{aligned} R(\xi) &= \frac{1}{n_1} \boldsymbol{\beta}^T \mathbf{M}_1 \mathbf{M}_1^T \boldsymbol{\beta} + \frac{1}{n_0} \boldsymbol{\beta}^T \mathbf{M}_2 \mathbf{M}_2^T \boldsymbol{\beta} \\ &= \boldsymbol{\beta}^T \Big( \frac{1}{n_1} \mathbf{M}_1 \mathbf{M}_1^T + \frac{1}{n_0} \mathbf{M}_2 \mathbf{M}_2^T \Big) \boldsymbol{\beta} \end{aligned}$$

Then the smooth estimate for the $k$-th supervised FPC is obtained by maximizing

$$Q_b(\xi) = \frac{\theta \langle \xi, \hat{\mathcal{C}} \xi \rangle + (1 - \theta) R(\xi)}{||\xi||_\lambda^2} = \frac{\boldsymbol{\beta}^T \mathbf{U}_b \boldsymbol{\beta}}{\boldsymbol{\beta}^T \mathbf{G} \boldsymbol{\beta}}, \ 0 \le \theta \le 1,$$

subject to $||\xi||_\lambda = 1$, $\langle \xi, \hat{\xi}_j \rangle = 0$, for every $j < k$, where

$$\mathbf{U}_b = \frac{\theta}{n} \mathbf{W} \mathbf{S}^T \mathbf{S} \mathbf{W} + (1 - \theta) \Big( \frac{1}{n_1} \mathbf{M}_1 \mathbf{M}_1^T + \frac{1}{n_0} \mathbf{M}_2 \mathbf{M}_2^T \Big).$$

Let $\boldsymbol{\delta} = \mathbf{G}^{\frac{1}{2}} \boldsymbol{\beta}$. It is equivalent to maximize
$\boldsymbol{\delta}^T (\mathbf{G}^{-1/2})^T \mathbf{U}_b \mathbf{G}^{-1/2} \boldsymbol{\delta}$, subject to $\boldsymbol{\delta}^T \boldsymbol{\delta} = 1$. Then $\boldsymbol{\delta}_1, \ldots, \boldsymbol{\delta}_J$ will be the the leading $J$ eigenvector of the matrix

$$(\mathbf{G}^{-1/2})^T \mathbf{U}_b \mathbf{G}^{-1/2}.$$

Consequently, one can derive the estimate for the vector of basis coefficients $\widehat{\boldsymbol{\beta}}_j = (\mathbf{G}^{1/2})^{-1}\boldsymbol{\delta}_j$. The corresponding estimate for the $j$-th smooth supervised FPC is $\hat{\xi}_j(t) = \widehat{\boldsymbol{\beta}}_j^T \boldsymbol{\Phi}(t)$ for $j = 1, \ldots, J$.

### 2.3.5 Functional Regression

With the estimated first leading $p$ FPCs, i.e., $\hat{\xi}_1(t), \hat{\xi}_2(t), \ldots,$
$\hat{\xi}_p(t)$, one can fit a functional regression model between the functional predictor $X(t)$ and the response $Y$ as discussed in Section 2.1. More specifically,

$$E(Y|X(t))) = g\Big(\beta_0 + \int_{\mathcal{T}} \beta(t)X(t)dt\Big), \tag{2.12}$$

in which $g(\cdot)$ is the link function. It is usually chosen as the inverse logit function if $Y$ is binary and the identify function if $Y$ is continuous. One can follow the same strategy described in Section 2.1 to express the unknown coefficient function

$$\beta(t) = \boldsymbol{\gamma}^T \hat{\boldsymbol{\xi}}(t),$$

in which the unknown coefficient vector $\boldsymbol{\gamma}$ can be estimated by maximizing the likelihood function of $Y$ with the mean expressed in terms of FPCs and FPC scores

$$E(Y|X(t)) = g\Big(\beta_0 + \sum_{j=1}^{p} \gamma_j \int_{\mathcal{T}} \hat{\xi}_j(t)\mu(t)dt + \boldsymbol{\alpha}^T\boldsymbol{\gamma}\Big).$$

The number of FPCs, denoted by $p$, used in the functional regression can be considered as a tuning parameter. We recommend to determine the value of $p$ in the following way. We start with the number of FPCs $p = 1$ and obtain the cross-validation error as $p$ increases. Our experience suggests choosing the value of $p$ when the cross-validation error stops decreasing significantly. For example, one can conduct a paired t-test between the cross-validation errors for $p$ and $p + 1$. If no significant improvement is observed, we choose $p$ as the optimal value. This rule is valid because the estimated first supervised FPC always has larger prediction ability than the second supervised FPC, and so forth.

Our method can also be extended to accommodate multiple function predictors. Suppose there are Q functional predictors: $X^{(1)}(t), \ldots, X^{(Q)}(t)$, then the multiple functional regression model can be expressed as

$$E(Y) = g\Big(\beta_0 + \sum_{q=1}^{Q} \int_{\mathcal{T}} \beta^{(q)}(t)X^{(q)}(t)dt\Big).$$

We can conduct the smooth supervised FPCA for each functional predictor $X^{(q)}(t)$, $q = 1, \ldots, Q$, and estimate the FPCs for $X^{(q)}(t)$. We denote the first $q_p$ estimated FPCs for the functional predictor $X^{(q)}(t)$ as $\hat{\boldsymbol{\xi}}^{(q)}(t) = (\hat{\xi}_1^{(q)}(t), \ldots, \hat{\xi}_{q_p}^{(q)}(t))$ with the corresponding score vector $\boldsymbol{\alpha}^{(q)}$ and the coefficient function $\beta^{(q)}(t) = (\boldsymbol{\gamma}^{(q)})^T \hat{\boldsymbol{\xi}}^{(q)}(t)$. Then the unkown coefficient vector $\boldsymbol{\gamma}^{(q)}, q = 1, \ldots, Q$, can also be estimated by maximizing the likelihood function of $Y$ with the mean expressed in terms of FPCs and FPC scores

$$E(Y) = g\Big(\beta_0 + \sum_{q=1}^{Q}\sum_{j=1}^{q_p} \gamma_j^{(q)} \int_{\mathcal{T}} \hat{\xi}_j^{(q)}(t)\mu^{(q)}(t)dt + \sum_{q=1}^{Q}(\boldsymbol{\alpha}^{(q)})^T\boldsymbol{\gamma}^{(q)}\Big), \tag{2.13}$$

in which $\mu^{(q)}(t)$ represents the mean trajectory for $X^q(t)$.

In practice, when multiple functional predictors exist, the number of total FPCs is sometimes close to or larger than the sample size. In this case, we recommend to employ one of those popular variable selection tools such as LASSO or SCAD to estimate the model. We will demonstrate this procedure in our real data application with a binary response variable.

## 2.4 Application

We apply our method to analyze an electroencephalography (EEG) dataset. The EEG dataset, collected by Zhang et al. (1995), is used to study the genetic predisposition to alcoholism. The original dataset is available in UCI machine learning repository (`https://archive.ics.uci.edu/ml/datasets/EEG+Database`). In total 122 subjects are separated into two groups: alcoholic and control. Each subject is exposed to two non-matching stimuli, i.e., two different pictures. In addition, 64 electrodes are placed on each subject's scalp to record the brain activities. Each electrode is sampled at 256Hz for 1 second. Our goal here is to classify alcoholic and control subjects based on their brain activities.

The number of trials with two non-matching stimuli ranges from 10 to 30 from subject to subject. Figure 2.1 displays the measurements averaged over all trials for 15 randomly selected subjects in both alcoholic and control groups at 256 time points at one sensor called the AF1 channel.

For each subject, we randomly select 2/3 of the total trials as the training trials and the rest 1/3 of the trials as the test trials. Then, the training and test observations are computed as the average of all training and test trials, respectively. We then apply our smooth supervised FPCA method to estimate the first leading $p$ FPCs for each sensor.

Next, we fit a multiple functional logistic regression

$$\text{logit}\{P(Y=1)\} = \beta_0 + \sum_{q=1}^{Q} \int_{\mathcal{T}} \beta^{(q)}(t) X^{(q)}(t)dt, \tag{2.14}$$

where $Y = 0, 1$ correspond to the control and alcoholic subject, respectively, and $X^{(q)}(t)$ is the brain activity for the $q$-th sensor. Following the method outlined in Subsection 2.3.5,

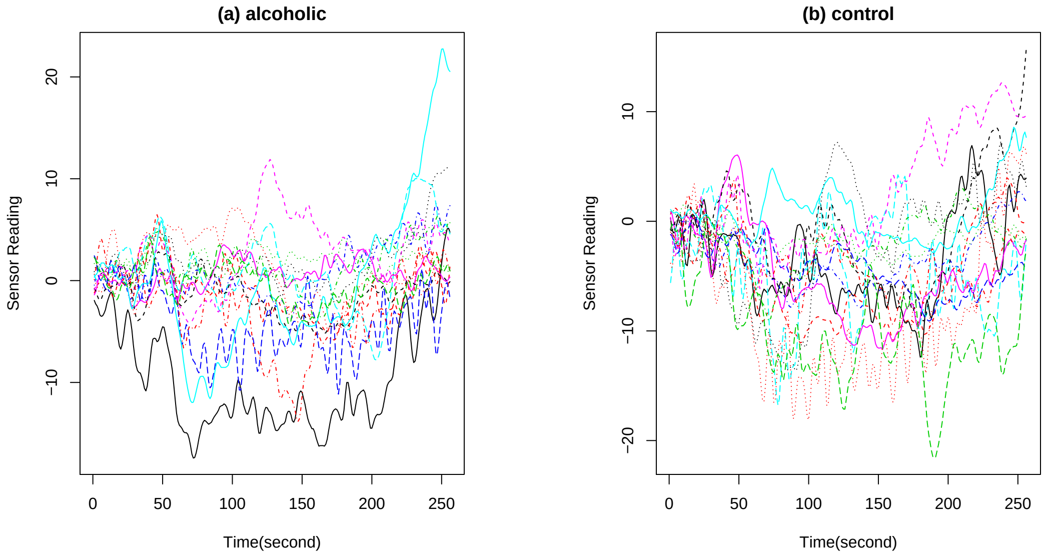

Figure 2.1: The readings of the brain activities at the AF1 channel for 15 randomly selected alcoholic subjects (panel a) and 15 randomly selected control subjects (panel b). All of them are exposed to two non-matching stimuli in an EEG case study on genetic predisposition to alcoholism.

we add an $L_1$ penalty on the coefficients for the slope function $\beta^{(q)}(t)$. In the context of a binary response, equation (2.13) becomes

$$\text{logit}\{P(Y=1)\} = \beta_0' + \sum_{q=1}^{Q} (\boldsymbol{\alpha}^{(q)})^T \boldsymbol{\gamma}^{(q)},$$

where $\beta_0' = \beta_0 + \sum_{q=1}^{Q}\sum_{j=1}^{p}\gamma_j^{(q)}\int_{\mathcal{T}}\hat{\xi}_j^{(q)}(t)\mu^{(q)}(t)dt$. Then the penalized log likelihood function is written as

$$\begin{aligned} l(\beta_0', \boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_q) = \frac{1}{n}\Big(n_1\log(p_i) + n_0\log(1-p_i)\Big) \\ + \lambda_L\Big(|\beta_0'| + \sum_{q=1}^{Q}\sum_{j=1}^{p}|\boldsymbol{\gamma}_j^{(q)}|\Big) \end{aligned}$$

where $p_i = \Pr(Y_i = 1|X_i^{(1)}(t), \ldots, X_i^{(Q)}(t)) = \text{inv-logit}(\beta_0' + \sum_{q=1}^{Q}(\boldsymbol{\alpha}^{(q)})^T\boldsymbol{\gamma}^{(q)})$ and $n_1 = \sum Y_i$. The tuning parameter, $\lambda_L$, in the LASSO penalty is chosen using a five-fold cross validation. For supervised FPCA, the weight parameter $\theta$ and the smoothing parameter $\lambda$ are selected from a 9-by-6 meshgrid, i.e., $[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9] \times [1, 10, 10^2, 10^3, 10^4, 10^5]$. Both of them are determined by a five-fold cross validation using the training data set simultaneously with the sparsity parameter, $\lambda_L$, in the LASSO penalty. We also apply the unsupervised FPCA method, in which the weight parameter $\theta$ is always set to be 1, and the smoothing parameter $\lambda$ is selected from $[1, 10, 10^2, 10^3, 10^4, 10^5]$ by a five-fold cross validation simultaneously with the sparsity parameter, $\lambda_L$, in the LASSO penalty. After obtaining the estimate $\hat{\beta}_0$ and $\hat{\beta}^{(q)}(t)$ for the multiple functional logistic regression (2.14), we classify the subjects on the test data, and obtain the corresponding classification error.

Table 2.1: The means and standard deviations of the classification error on testing set in 100 random data splitting using both supervised FPCA and unsupervised FPCA in the EEG data application. Here sFPCA and FPCA stands for supervised FPCA and unsupervised FPCA respectively.

| | | # of FPCs | | | |
|---|---|---|---|---|---|
| Method | classification error | 1 | 2 | 3 | 4 |
| sFPCA | mean | 0.208 | 0.200 | 0.180 | 0.166 |
| | sd | 0.018 | 0.020 | 0.031 | 0.024 |
| FPCA | mean | 0.266 | 0.217 | 0.207 | 0.212 |
| | sd | 0.024 | 0.024 | 0.032 | 0.026 |

We repeat the above process for 100 replicates of random data splittings and summarize the test classification errors. Table 2.1 shows the mean and standard deviation of the classification errors when the number of FPCs is selected for each sensor varies from 1 to 4 for supervised FPCA and unsupervised FPCA. It shows that supervised FPCA has a higher classification accuracy than unsupervised FPCA. For instance, supervised FPCA improves the classification accuracy by about 20%, when just using one FPC, in comparison with unsupervised FPCA. As one reviewer points out, it is not clear whether the difference of the misclassification rate between FPCA and sFPCA is statistically significant.

## 2.5 Simulation Studies

Three different simulations are conducted to evaluate the proposed method. We first briefly introduce the generation mechanism for the functional predictor $X(t)$ in the beginning of this section, since this generation mechanism stays the same across different simulations. Then we discuss each simulation in details. We also conduct two more simulation studies to compare our proposed supervised FPCA method with three alternative methods including supervised PCA proposed by Bair et al. (2006), SupSVD (Li et al., 2016) and SupSFPC (Li et al., 2016).

In order to make the simulation setting similar to real data, we use four FPCs, shown in Figure 2.2, to generate sample functional predictors. They are the first four leading FPCs estimated from the Canadian weather data (Ramsay and Silverman, 2005), which consist of daily temperature measurements at 35 weather stations across Canada. Each functional predictor $X_i(t), i = 1, \ldots, n$, is simulated as: $X_i(t_k) = \alpha_{1i}\xi_1(t_k) + \alpha_{2i}\xi_2(t_k) +$

$\alpha_{3i}\xi_3(t_k) + \alpha_{4i}\xi_4(t_k), k = 1, 2, \dots, 365$, where $\xi_j(t_k)$ is the $j$-th true FPCs, $j = 1, \dots, 4$. The simulated FPC score is simulated as: $\boldsymbol{\alpha}_i^T = (\alpha_{1i}, \alpha_{2i}, \alpha_{3i}, \alpha_{4i})^T \overset{i.i.d}{\sim} MVN(\mathbf{0}, \Sigma)$, in which $\Sigma = \text{diag}(100,$
$80, 50, 30)$. Figure 2.3 below displays 50 random curves simulated under these settings.

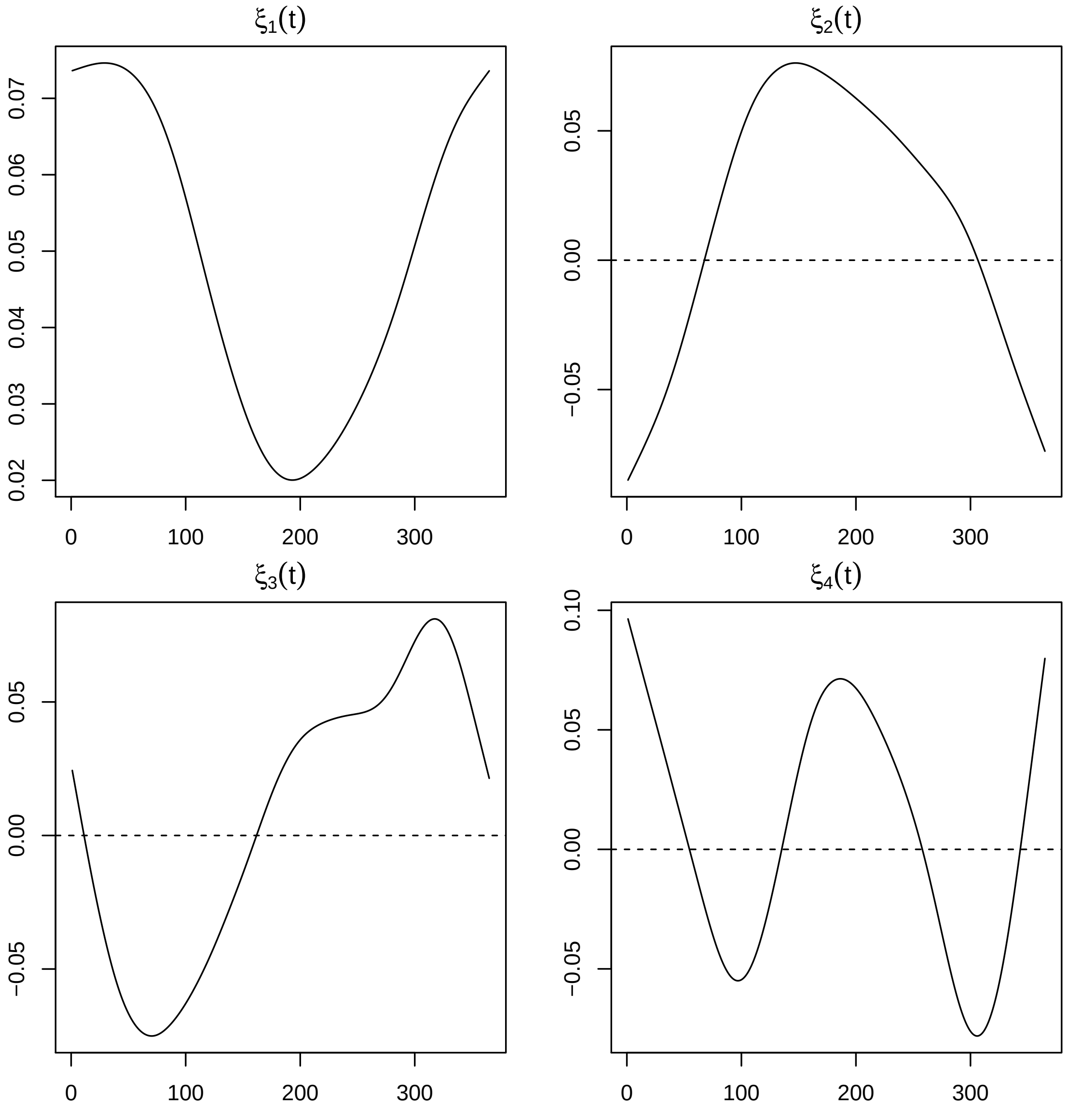

Figure 2.2: The first four leading FPC estimated from Canadian Weather data.

### 2.5.1 The First Simulation Study

The first simulation study is designed to evaluate the proposed method when the response variable is binary. Here we generate 1000 sample curves, $X_i(t), i = 1, \dots, 1000$. The response

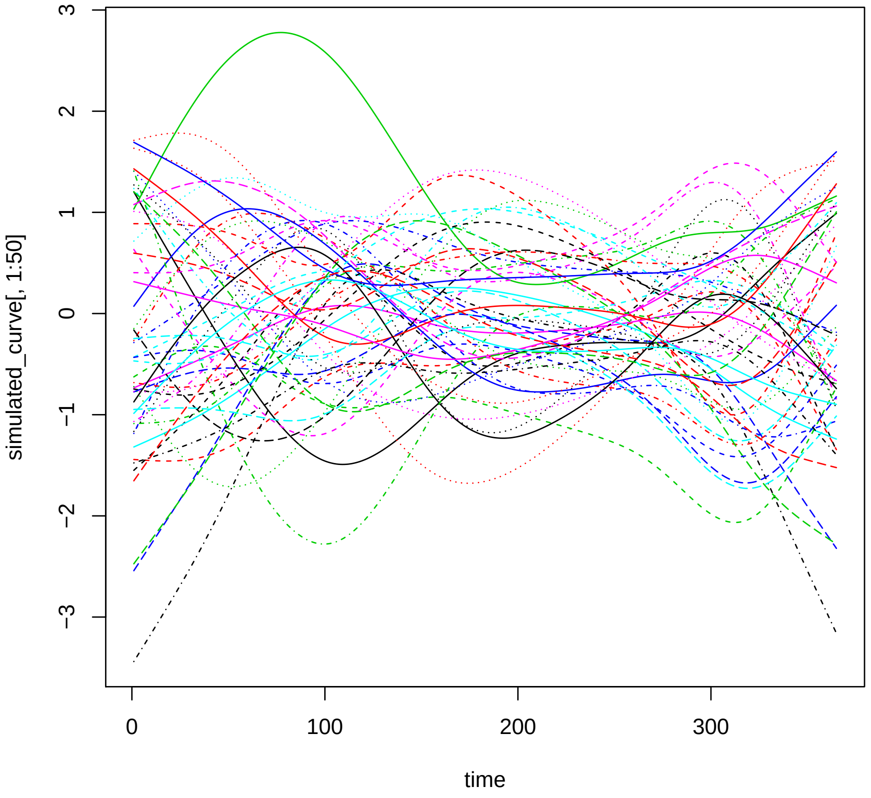

Figure 2.3: Fifty randomly simulated curves in the simulation studies

variable $Y$ is generated as:

$$Y_i \sim \text{Bernoulli}(p_i),$$
$$\text{logit}(p_i) = \int_{\mathcal{T}} \beta(t) X_i(t) dt, i = 1, \ldots, 1000,$$

in which $\beta(t) = \xi_4(t)$. In other words, the binary response $Y$ is only related to the fourth FPC $\xi_4(t)$. We randomly select 200 samples as the test set and used the other 800 samples as the training set. For supervised FPCA method, the weight parameter $\theta$ and the smoothing parameter $\lambda$ are selected on a 5-by-3 meshgrid, i.e., $[0.1, 0.3, 0.5, 0.7, 0.9] \times [10, 10^3, 10^5]$, through a five-fold cross validation using those 800 training samples only. For unsupervised FPCA method, the weight parameter $\theta$ is fixed to be 1 under different values of $\lambda$ and the smoothing parameter was selected from 10, $10^3$ and $10^5$ using a five-fold cross validation as well.

We compare the prediction performance of supervised FPCA with unsupervised FPCA in terms of classification errors on the test data in 100 simulation. Figure 2.4 summarizes the classification errors. Supervised FPCA yields a much lower classification error than unsupervised FPCA when the number of FPCs used, $p$, is less than 3. More specifically, the mean test classification error of unsupervised FPCA is slightly less than 50% unless choosing four FPCs, whereas the mean classification error of supervised FPCA is constantly less than 14% even when the number of FPCs is less than 3. This shows that supervised FPCA is able to detect the FPCs that are most related with the response variable in advance and our method can well accommodate the binary response.

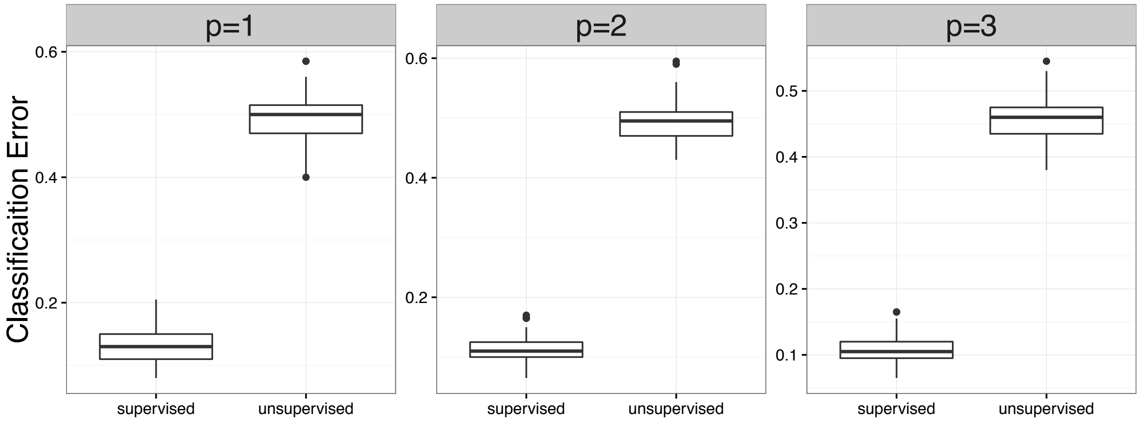

Figure 2.4: Boxplots of the classificaition errors for 100 simulation runs when using the first p FPCs estimated by supervised and unsupervised FPCA in the first simulation study when the response variable is binary.

To gain some insight, in Figure 2.5, we compare the first FPC $\hat{\xi}_1(t)$ estimated by supervised and unsupervised FPCA in one simulation run, along with the true FPC used to simulate the response variable. We can see that the first FPC estimated by supervised FPCA method is much closer to the true FPC, in comparison with the first FPC estimated by unsupervised FPCA method.

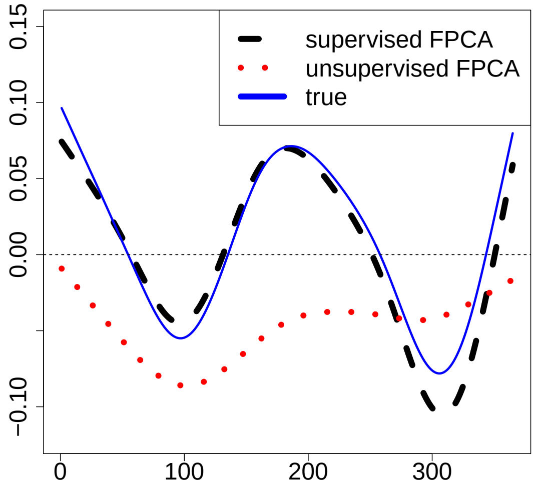

Figure 2.5: The first FPC estimated with supervised and unsupervised FPCA in one simulation run of the first simulation study when the response variable is binary.

### 2.5.2 The Second Simulation Study

We conduct three simulation scenarios to evaluate the proposed method in different settings when the response variable is continuous. Here we generate 100 sample curves, $X_i(t), i = 1, \ldots, 100$, in the same way as discussed in the beginning of this section. The response variable $Y$ is generated using the functional linear regression model (2.1) with $\beta(t)$ being specified as $\beta(t) = \gamma_1\xi_1(t) + \gamma_2\xi_2(t)\xi_j(t) + \gamma_3\xi_3(t) + \gamma_4\xi_4(t) = \boldsymbol{\gamma}^T\boldsymbol{\xi}(t)$, where $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, \gamma_3, \gamma_4)^T$ and $\boldsymbol{\xi}(t) = (\xi_1(t), \xi_2(t), \xi_3(t), \xi_4(t))^T$. In addition, the link function $g(\cdot)$ is the identity function. Without loss of generality we set $\beta_0 = 0$.

#### Scenario 1

In the first scenario, we set the true $\boldsymbol{\gamma} = (0, 0, 0, 1)^T$ such that the true slope function $\beta(t) = \xi_4(t)$. In other words, the response variable $Y$ is only related to the fourth leading FPC $\xi_4(t)$. In addition, the noise term $\epsilon$ follows a normal distribution $N(0, 30\rho)$, in which $\rho$ denotes the signal-to-noise ratio. We set $\rho = 5\%$ and $50\%$. We randomly select 20 samples as the test set and treat the other 80 samples as the training set. Both the smoothing parameter and the weight parameter are chosen via five-fold cross validation using the training samples only on the same meshgrid used in the previous simulation. As for the unsupervised FPCs, the weight parameter $\theta$ is set to be 1. The smoothing parameter $\lambda$ is selected from $\{10,\ 10^3,\ 10^5\}$ using a five-fold cross validation. For unsupervised FPCA method, the weight parameter $\theta$ is set to be 1. We compare the prediction performance of the supervised FPCs with that of the unsupervised FPCs using 500 simulation runs. The prediction error is evaluated using relative mean squre error (`RAMSE`) defined as

$$\texttt{RAMSE} = \frac{\sum_{\ell=1}^{n}(\hat{y}_\ell - y_\ell)^2}{\sum_{\ell=1}^{n}(\bar{y} - y_\ell)^2}. \tag{2.15}$$

Here $y_\ell$ and $\hat{y}_\ell$ denote the observed $\ell$th response in the test set, respectively, and $\bar{y}$ represents the average of theose oberved responses the training set.

Figure 2.6 summarizes the prediction `RAMSE`s for 100 repeated runs when the noise-to-signal ratio $\rho = 5\%$. As we can see, supervised FPCA method consistently give lower `RAMSE` compared with unsupervised FPCA when $p$ is less than 3. More specifically, when $p < 4$, the unsupervised FPCs perform no better than simply using the sample mean of the training set as the average prediction error is constantly around 100%. In contrast, the supervised FPCs is able to capture the information of the response variable and improve its prediction performance accordingly. For example, even restricting only one FPC in the functional linear regression, the average `RAMSE` is less than 45%, only half of the average `RAMSE` of the unsupervised FPCs.

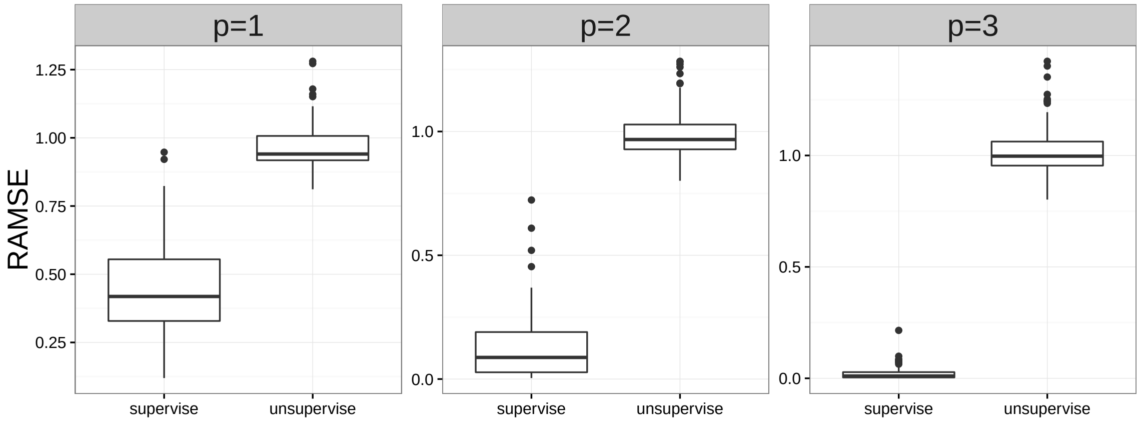

Figure 2.6: Boxplots of the prediction `RAMSE`s for 100 simulation runs using the first `p` FPCs estimated by supervised and unsupervised FPCA in Scenario 1 of the second simulation study when the response variable is continuous and the noise-to-signal ratio $\rho = 5\%$.

To gain some insight, Figure 2.7 displays the first FPC $\hat{\xi}_1(t)$ estimated by supervised and unsupervised FPCA along with the true FPC related to the response variable when the noise-to-signal ratio of the data is $\rho = 5\%$. We can see the first FPC estimated by supervised FPCA is much more closer to the true FPC compared with the first FPC estimated by unsupervised FPCA. This indicates that supervised FPCA is able to detect the FPC that is truly related to the continuous response variable.

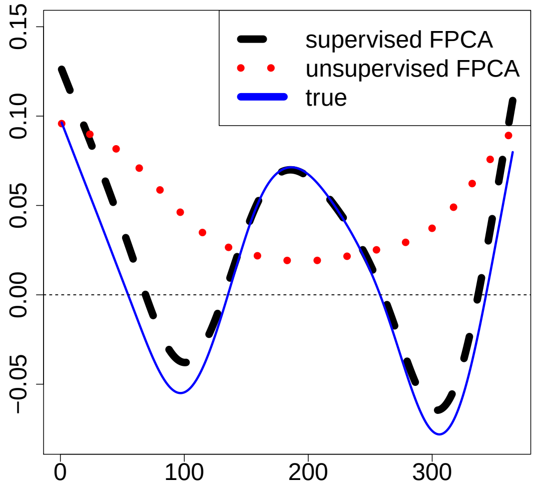

Figure 2.7: The first FPC estimated with supervised and unsupervised FPCA at one simulation run in Scenario 1 of the second simulation study when the response variable is continuous.

Figure 2.8 displays the boxplots of the prediction `RAMSE`s when the simulation data have the noise-to-signal ratio as $\rho = 50\%$. It shows that supervised FPCA yielded a more robust estimator since the mean `RAMSE` is only increased about 20% when the noise-to-signal ratio of the simulated data is increased from 5% to 50%.

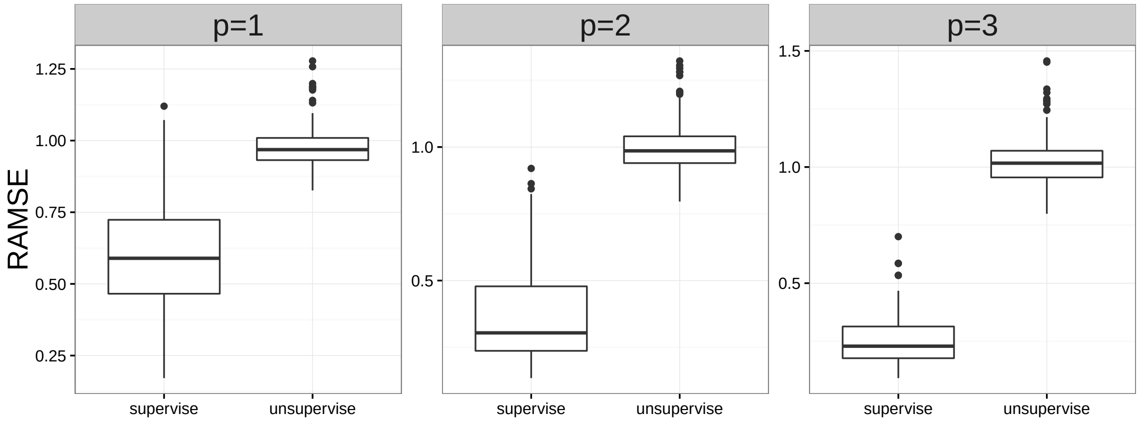

Figure 2.8: Boxplots of the prediction RAMSEs for 100 simulation runs using the first p FPCs estimated by supervised and unsupervised FPCA in Scenario 1 of the second simulation study when the response variable is continuous and the noise-to-signal ratio $\rho = 50\%$.

### Scenario 2

The only difference between this scenario and the previous one in section 2.5.2 is that we specify $\boldsymbol{\gamma} = (0.25, 0.73, 0.29,$
$0.56)^T$, such that the response variable $Y$ is related to a linear combination of all $\xi_i(t), i = 1, 2, 3, 4$. In practice, this case might be more realistic compared to the scenario when the response is only related to a single FPC.

Figure 2.9 summarizes the prediction errors for 100 simulation runs when the noise-to-signal ratio of the data is $\rho = 5\%$. It shows that supervised FPCA still outperforms unsupervised FPCA when using up to 3 FPCs. More specifically, when just using one FPC, i.e. $p = 1$, unsupervised FPCA only performs slightly better than simply using the sample mean of the training set as the average RAMSE is about 91%, because unsupervised FPCA only successfully recovers the first FPC, while the response variable is correlated with all four FPCs. In contrast, the average RAMSE using supervised FPCA is only 14.6% when just using one FPC, which is quite satisfying.

These two scenarios above show that the prediction performance of supervised FPCA seems quite satisfactory no matter whether the response variable is related to a single FPC or a linear combination of several FPCs.

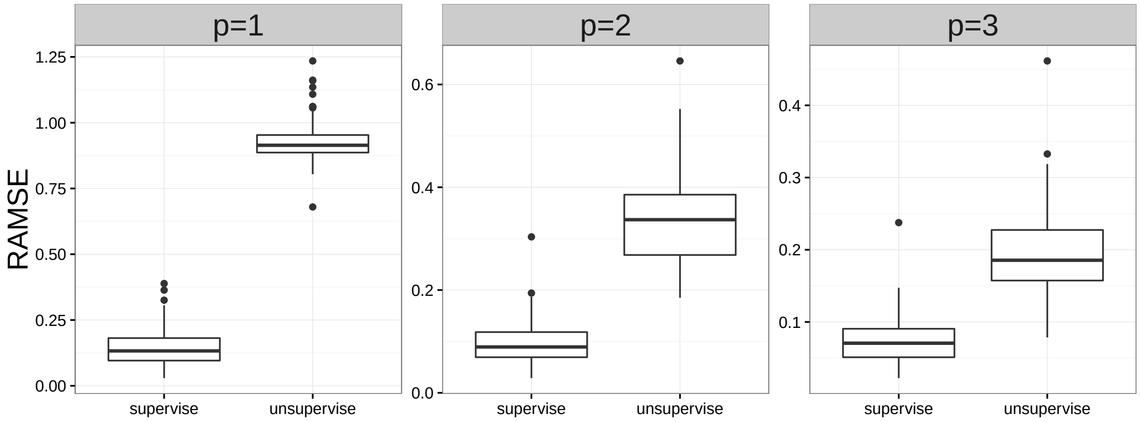

Figure 2.9: Boxplots of the prediction `RAMSE`s for 100 simulation runs using the first `p` FPCs estimated by supervised and unsupervised FPCA in Scenario 2 of the second simulation study when the response variable is continuous.

**Scenario 3**

In the third scenario, we set the true $\beta(t)$ as an arbitrary function generated using cubic b-spline basis functions. More specifically,

$$\beta(t) = \sum_{l=1}^{L} c_l \psi_l(t),$$

in which $\psi_l(t)$ denotes the $l$th B-spline basis function and the basis coefficient $c_l$ is independently sampled from the standard normal distribution. We set $L$ to be 10. The functional predictor $X_i(t), i = 1, \ldots, 40$ are generated in the same way as described in the beginning of Section 2.5. In each simulation run, we first randomly generate a $\beta(t)$ by sampling $c_l$ independently from the standard normal distribution, then generate the response $Y$ using the functional linear regression model with the corresponding $\beta(t)$ and the noise term $\epsilon$, where the noise-to-signal ratio $\rho$ is 50%. Then we randomly choose half of the samples as the training data set and treat the remaining half as the test data set.

We compare the prediction performance of our proposed supervised FPCA with the traditional unsupervised FPCA along with three other methods including supervised PCA (Bair et al., 2006), SupSVD (Li et al., 2016) and SupSFPC (Li et al., 2016). To be more specific, supervised PCA treats the functional predictor $X(t)$ at each observed time point as one scalar predictor. The R package *superpc* is used to implement this method. The optimal feature threshold is determined using the default 5-fold cross-validation proposed in Bair et al. (2006), which is also provided in the R package. For SupSVD and SupSFPC, we first estimate the FPCs and then fit a regression model afterwards. These two methods are implemented using the matlab code provided by the authors. For supervised FPCA, the weight parameter $\theta$ and the smoothing parameter $\lambda$ are selected simultaneously on a 9-by-6 meshgrid, i.e., $[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9] \times [1, 10, 10^2, 10^3, 10^4, 10^5]$, through a five-fold cross validation. For unsupervised FPCA, the weight parameter $\theta$ is fixed to be 1 and the smoothing parameter is selected from $[1, 10, 10^2, 10^3, 10^4, 10^5]$ using a five-fold cross validation as well. In each simulation replicate, we apply supervised PCA, SupPCA, SupFPCA, unsupervised FPCA and supervised FPCA on the same training set and obtain the `RAMSE` on the same test set. The above procedure is repeated for 100 times.

Table 2.2 shows the mean `RAMSE`s and the corresponding standard deviations for each method. As is shown in this table, supervised FPCA outperforms all the alternative methods, as it always produces the lowest mean `RAMSE`s.

Table 2.2: The mean and standard deviation of the `RAMSE` in 100 simulation replicates when the true $\beta(t)$ is an arbitrary function. In each simulation run, the true coefficient function $\beta(t) = \sum_{l=1}^{10} c_l \psi_l(t)$, where $c_i$ is independently drawn from the standard normal distribution. For instance, the mean `RAMSE` of sFPCA is 69.4% when the number of FPCs $p = 1$. Here FPCA, sFPCA and sPCA denote unsupervised FPCA, supervised FPCA and supervised PCA, respectively.

| | | # of FPCs | | |
|---|---|---|---|---|
| Method | RAMSE | 1 | 2 | 3 |
| FPCA | mean | 0.694 | 0.467 | 0.280 |
| | sd | 0.241 | 0.234 | 0.107 |
| sFPCA | mean | 0.306 | 0.283 | 0.260 |
| | sd | 0.102 | 0.100 | 0.090 |
| sPCA | mean | 0.477 | 0.429 | 0.410 |
| | sd | 0.147 | 0.121 | 0.117 |
| SupSFPC | mean | 1.018 | 1.044 | 1.086 |
| | sd | 0.057 | 0.073 | 0.096 |
| SupSVD | mean | 1.016 | 1.041 | 1.059 |
| | sd | 0.050 | 0.068 | 0.097 |

We further conduct a paired t-test to compare supervised FPCA with each of the alternative methods and confirm that the `RAMSE` yielded by supervised FPCA is significantly lower than that yielded by any other methods. The details of the paired t-tests are shown in Table 2.3.

Table 2.3: The mean of paired differences of the `RAMSE`s in 100 simulation replications between supervised FPCA and four alternative methods. In each simulation run, the true coefficient function $\beta(t) = \sum_{l=1}^{10} c_l \psi_l(t)$, where $c_i$ is independently drawn from the standard normal distribution. For instance, the mean `RAMSE` using the unsupervised FPCA method is about 17.3% higher than that using supervised FPCA, with a paired t-test p-value being 1.2e-31 when the number of FPCs $p = 1$. Here FPCA and sPCA denote unsupervised FPCA and supervised PCA, respectively.

| | # of FPCs | | |
|---:|---:|---:|---:|
| Method | 1 | 2 | 3 |
| FPCA | -17.3% (1.2e-31) | -8.8% (4.8e-14) | -3.5% (6.3e-04) |
| sPCA | -9.5% (1.2e-15) | -9.4% (1.9e-15) | -9.7% (4.8e-16) |
| SupSFPC | -61.5% (1.1e-80) | -61% (2.4e-80) | -64.5% (4.5e-82) |
| SupSVD | -62.7% (1.7e-81) | -62.3% (3.1e-81) | -62.2% (3.9e-81) |

**Scenario 4**

Previous simulations only assume the outcome is related to 4 FPCs at most. In this scenario, we first generate the functional predictor $X(t)$ using the first 10 leading FPCs estimated from the Canadian weather data (Ramsay and Silverman, 2005). More specifically, the functional predictor $X_i(t), i = 1, \ldots, n$, is simulated as: $X_i(t_k) = \sum_{j=1}^{10} \alpha_{ji}\xi_j(t_k), k = 1, 2, \ldots, 365$, where $\xi_j(\cdot)$ is the $j$th leading FPC estimated from the Canadian Weather data. The FPC score is simulated as: $\boldsymbol{\alpha}_i = (\alpha_{1i}, \ldots, \alpha_{10i})^T \overset{i.i.d}{\sim} MVN(\mathbf{0}, \Sigma)$, in which $\Sigma = \text{diag}(120, 108.8, 97.5, 86.2, 75,$
$63.8, 52.5, 41.2, 30, 15)$. Following this way, the 10th FPC, $\xi_{10}(t)$, only explains about 2% of the total variation. To relate the outcome to a larger number of FPC's, we set the true $\beta(t)$ as

$$\beta(t) = \sum_{j=1}^{10} c_j \xi_j(t),$$

In which the coefficient vector $\mathbf{c}^T = (c_1, \ldots, c_{10})^T = (0.31, 0.21, 0.61, 0.61, 0.07, 0.22,$
$0.95, 0.37, 0.59, 0.83)$, which is obtained by drawing random numbers from the uniform distribution.

In each simulation run, we first generate the response $Y$ using the functional linear regression model with $\beta(t)$ and the noise term $\epsilon \sim N(0, 15\rho)$. The noise-to-signal ratio $\rho$ is set as 50% and $n = 1000$. We then randomly choose 20% samples as the training data set and treat the remaining 80% samples as the test data set. For supervised FPCA, the weight parameter $\theta$ and the smoothing parameter $\lambda$ are selected simultaneously on a 9-by-6 meshgrid, i.e., $[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9] \times [1, 10, 10^2, 10^3, 10^4, 10^5]$, through a five-fold cross validation. For unsupervised FPCA, the weight parameter $\theta$ is fixed to be 1 and the smoothing parameter is selected from $[1, 10, 10^2, 10^3, 10^4, 10^5]$ using a five-fold cross validation as well. The above procedure is repeated 100 times. As shown in Table 2.4, supervised FPCA outperforms unsupervised FPCA. Even with only one FPC, supervised FPCA seems to be able to predict the response variable well.

Table 2.4: The mean and standard deviation of the `RAMSE` in 100 simulation replicates when the true $\beta(t)$ is related to 10 FPCs. Here FPCA and sFPCA represent unsupervised FPCA and supervised FPCA, respectively.

| | | # of FPCs | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Method | RAMSE | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| FPCA | mean | 0.921 | 0.889 | 0.558 | 0.379 | 0.208 | 0.205 | 0.190 |
| | sd | 0.022 | 0.031 | 0.072 | 0.079 | 0.022 | 0.022 | 0.018 |
| sFPCA | mean | 0.154 | 0.148 | 0.145 | 0.143 | 0.140 | 0.140 | 0.137 |
| | sd | 0.018 | 0.016 | 0.016 | 0.016 | 0.015 | 0.015 | 0.014 |

## 2.6 Summary

In this chapter, we consider the problem of predicting a scalar response variable by using one or several functional predictors. The conventional FPCA method focuses on finding FPCs that maximize the variation of FPC scores and ignores the response variable. We have proposed a one-step supervised FPCA to detect those FPCs whose scores are correlated with the response variable. The resulting FPCs have a better prediction performance compared to the conventional FPCA method.

Through our real data application and simulations, we demonstrate that our method can accommodate both continuous and binary response variable. Even through we only show examples with binary response variable, we believe that our method can be easily extended to predicting multinomial response variable. Lastly, our method is also quite user-friendly.

# Chapter 3

# Sparse Functional Principal Components Analysis

## 3.1 Introduction

Functional principal component analysis (FPCA) is a key dimension reduction tool in functional analysis. FPCA explores major sources of variability in a sample of random curves by finding functional principal components (FPCs) that maximize the curve variation. Consequently, the top few FPCs explain most of the variability in the random curve. In addition, each random curve can be approximated by a linear combination of the top few FPCs. Therefore, the infinite-dimensional curves are projected to a low-dimensional space defined by the top FPCs. This powerful dimensional reduction feature also promotes the popularity of FPCA.

The theoretical properties of FPCA have been carefully studied at length. For example, Dauxois et al. (1982) first studied the asymptotic properties of PCA estimators for the infinite dimensional data from a linear operator viewpoint. Following this point of view, Mas (2002); Bosq (2000) utilized functional analysis to study FPCA theoretically. On the other hand, Hall and Horowitz (2007); Hall et al. (2006); Yao et al. (2005a) studied FPCA from the kernel perspective. In addition, FPCA has been widely and successfully applied in many applications such as functional linear regression (Yao et al., 2005b), classification and clustering of functional data (Ramsay and Silverman (2005); Yao et al. (2005b); Müller (2005); Müller and Stadtmüller (2005); Peng and Müller (2008)). All these applications assume the functional data are densely and regularly observed. When it comes to sparse and irregularly observed data, Yao et al. (2005a) proposed to estimate the FPC score using conditional expectation, which allows recovering the individual trajectory by borrowing information across all the subjects. The smooth version of functional principal component analysis is carefully studied by Rice and Silverman (1991); Pezzulli (1993); Silverman (1996) and Yao et al. (2005a). There are mainly three methods to achieve smoothness. The first method smooths the functional data in the first step and conducts the regular FPCA on the sample covariance function. The second method smooths the covariance function first and then eigendecomposes the resulting smoothed covariance function to estimate the smoothed FPCs. The last method directly adds a roughness penalty in the optimization criterion for estimating the FPCs.

The conventional functional principal component analysis aims to estimate FPCs which maximize the curve variation. These FPCs represent the source or direction of maximum variations among curves, and the curves are projected to the low-dimensional space defined by these FPCs. Therefore, it is important to interpret them. However, these FPCs are usually nonzero in the entire domain, and users often find it hard to interpret these FPCs. On the other hand, if the estimated FPC is only nonzero in a subregion of the entire domain, we can easily use them to identify the subregions from which the major variation of the curves exhibits. In this chapter, our goal is to propose a method to estimate the sparse functional principal components, which are only nonzero in a subregion and at the same time account for an almost maximum amount of variation within the curves.

Two methods have been proposed to enhance the interpretability of functional principal components. The first method is the interpretable functional principal components analysis (iFPCA) proposed by Lin et al. (2016). This method adds an $\ell_0$-penalty on the length of the nonzero region of FPCs and obtains FPCs which are only nonzero in subregions. However, the optimization in their framework is an NP-hard problem because of the use of the $\ell_0$-penalty. A greedy backward elimination algorithm is proposed to solve this optimization problem approximately. The second method is called a localized functional principal components analysis (LFPCA) method proposed by Chen and Lei (2015). This method adds an $\ell_1$ penalty to the original eigendecomposition problem of smoothed FPCs, which is also not a convex optimization problem. They approximate this non-convex problem through a Deflated Fantope Localization method and propose a novel estimation procedure in a sequential manner.

This chapter has three major contributions. Firstly, we propose a new regression-type framework for the sparse functional principal component analysis. The estimated sparse FPCs can not only account for a reasonable variation within the functional data but also be sparse on the whole domain. We also show that the FPCs estimated with our proposed sparse FPCA method is equivalent to the FPCs with the conventional FPCA method when the sparsity parameter is zero. Secondly, our method is not an NP-hard optimization problem and the computation is very efficient. Lastly, our method estimates the top sparse FPCs simultaneously rather than estimating each FPC in a sequential manner. An R package “sparseFPCA" is developed to implement our proposed SFPC method and is available at `https://github.com/YunlongNie/sparseFPCA`.

The rest of the chapter is organized as follows. In Section 3.2, we introduce our SFPCA method and show its connection with the conventional FPCA. Details of our method and the computation algorithm are described in Section 3.3, followed by theoretical results in Section 3.4. In Section 3.5, we apply our proposed method in a real-data application to explore major sources of variation among the acceleration rates of 107 diesel trucks. In Section 3.6, a carefully-designed simulation is conducted to evaluate the finite sample performance of our proposed method in comparison with other alternative methods. Section 3.7 provides concluding remarks.

## 3.2 Sparse Functional Principal Component Analysis

Consider a stochastic process $X(t)$, which is square integrable on the compact domain $\mathcal{T}$. In other words, $X(t) \in L^2(\mathcal{T})$ almost surely, where $L^2(\mathcal{T})$ is the Hilbert space of square integrable functions on $\mathcal{T}$. We denote the inner product between two functions $f, g \in L^2(\mathcal{T})$ as $\langle f, g\rangle = \int_{\mathcal{T}} f(t)g(t)dt$ with the corresponding norm as $||f|| = \sqrt{\langle f, f\rangle}$. If $\langle f, g\rangle = 0$, we also use the notation $f \perp g$. Let $x_i(t), i = 1, \dots, n$, be the observed functional data for the stochastic process $X(t)$. Without loss of generality, we assume that $E(x(t)) = 0$ in the rest of this chapter. In practice, one can always center the observed functional data first to remove this assumption.

We propose to estimate the first $J$ leading unnormalized sparse FPCs $\boldsymbol{\beta}(t) = (\beta_1(t), \dots, \beta_J(t))^T$ by minimizing the following criterion:

$$\frac{1}{n}\sum_{i=1}^{n}\left|\left|x_i(t) - \sum_{j=1}^{J}\alpha_j(t)\langle\beta_j, x_i\rangle\right|\right|^2 + \tau\sum_{j=1}^{J}\int\beta_j^2(t)dt + \text{PEN}(\boldsymbol{\beta}(t)) \tag{3.1}$$

with respect to $\beta_j(t)$ and the ancillary parameter $\alpha_j(t)$, $j = 1, \dots, J$, with the constraints $||\alpha_j||^2 = 1$ and $\langle\alpha_\ell, \alpha_j\rangle = 0$ when $\ell \neq j$. Here, $\text{PEN}(\boldsymbol{\beta}(t))$ in (3.1) is a sparsity penalty term for $\boldsymbol{\beta}(t)$, which penalizes the length of nonzero regions of each $\beta_j(t)$. When the sparsity penalty term becomes zero, we will show in Section 3.4 that minimizing the first two terms in (3.1) will give rise to the conventional FPCs. In other words, the resulting FPC will be the optimal basis functions in explaining or recovering observed functional data $\{x_i(t), i = 1, \dots, n\}$. In fact, when the sparsity penalty term become zero, $\alpha_j(t)$ is equal to the conventional $j$-th FPC, and $\alpha_j(t) = \beta_j(t)/||\beta_j||$. Hence, $\langle\beta_j, x_i\rangle/||\beta_j||$ will be the corresponding conventional FPC score for $x_i(t)$. On the other hand, after adding the sparsity penalty, our criterion (3.1) will not only consider the resulting FPC's ability to explain the maximum variation among the functional data, but also take the sparsity of the FPCs into account. After obtaining the estimate for the first $J$ leading unnormalized sparse FPCs $\{\hat{\beta}_j(t)\}$, we normalize each $\{\hat{\beta}_j(t)\}$ to obtain the normalized sparse FPCs $\hat{\xi}_j(t) = \widehat{\beta}_j(t)/||\widehat{\beta}_j(t)||$.

### 3.2.1 Sparsity Penalty

The sparse penalty term in (3.1) penalizes the length of nonzero regions of $\beta_j(t)$. The functional SCAD method proposed by Lin et al. (2016) is a functional generalization of the

SCAD method (Fan and Li, 2001). The functional SCAD method is used in Lin et al. (2016) to find a locally sparse estimator for the coefficient function in functional linear regression models. The nice shrinkage property of functional SCAD allows the proposed estimator to locate null subregions of the coefficient function without over shrinking nonzero values of the coefficient functions.

We employ the functional SCAD penalty to achieve a locally sparse estimator of FPCs by defining:

$$\mathrm{PEN}(\boldsymbol{\beta}(t)) = \sum_{j=1}^{J} \int p_{\lambda_j}(|\beta_j(t)|)dt,$$

in which $p_\lambda(\cdot)$ is the SCAD function defined in Fan and Li (2001):

$$p_\lambda(u) = \begin{cases} \lambda u & \text{if } 0 \le u \le \lambda, \\ -\frac{u^2 - 2a\lambda u + \lambda^2}{2(a-1)} & \text{if } \lambda < u < a\lambda, \\ \frac{(a+1)\lambda^2}{2} & \text{if } u \ge a\lambda, \end{cases}$$

where $a$ is 3.7, as suggested by Fan and Li (2001), and $\lambda$ is the tuning parameter. A large value of the tuning parameter $\lambda_j$ will penalize the nonzero region of the corresponding $\beta_j(t)$, hence leading to a sparse estimation. On the other hand, when the sparse parameter, $\lambda_j$, is zero, the resulting $\beta_j(t)$ reduces to the conventional functional principal components. Before showing the details of estimating the sparse FPCs given the tuning parameters, we first show that the FPCs estimated with our proposed sparse FPCA method is equivalent to the FPCs with the conventional FPCA method when the sparsity parameter is zero.

### 3.2.2 Connection to the Conventional FPCA

The conventional FPCA methods estimate the top FPCs with the eigendecomposition method. We can show that the $j$-th functional principal components (FPCs) $\phi_j(t)$ is the $j$-th eigenfunction of the covariance function $G(s,t) = \mathrm{E}(X(s)X(t))$, and satisfies the following eigenequation:

$$\int G(s,t)\phi_j(s)ds = \lambda_j \phi_j(t), \tag{3.2}$$

where $\lambda_j$ is the corresponding eigenvalue and $\lambda_1 \ge \lambda_2 \ge \ldots \ge 0$. The conventional methods estimate the FPCs by solving the above eigenequation (3.2) by replacing the covariance function $G(s,t)$ with the empirical covariance function $g(s,t) = \frac{1}{n}\sum_{i=1}^{n} x_i(s)x_i(t)$. The FPC score $s_{ij}$ can be calculated as $s_{ij} = \langle x_i, \phi_j \rangle$. The FPC score $s_{ij}$ has mean 0 and variance $\lambda_j$. One widely-used strategy to determine the number of FPCs is to choose a value such that the first $J$ leading FPCs account for more than 90% of the total variation:

$$J = \inf\left\{k : \frac{\sum_{j=1}^{k}\lambda_j}{\sum_{j=1}^{\infty}\lambda_j} \geq 90\%\right\}.$$

Another conventional way to understand FPCs is through the Karhunen-Loève(KL) expansion (Fukunaga and Koontz, 1970). More specifically, according to the KL expansion, $x_i(t)$ can be expressed as

$$x_i(t) = \sum_{j=1}^{\infty} s_{ij}\phi_j(t), \quad i = 1, \ldots, n, \tag{3.3}$$

in which $\langle \phi_i, \phi_j \rangle = \delta_{ij}$, and $\delta_{ij}$ is the Kronecker's delta. A major advantage of FPCA is that by projecting each $x_i(t)$ onto orthogonal FPCs with uncorrelated scores, it allows us to approximate each $x_i(t)$ using the first $J$ leading FPCs:

$$x_i(t) \approx \sum_{j=1}^{J} s_{ij}\phi_j(t), \quad i = 1, \ldots, n. \tag{3.4}$$

In fact, there are many other basis functions on which $x_i(t)$ can be projected. However, the FPCs obtained from eigendecomposing the empirical covariance function are the optimal basis functions in the sense that they minimize the squared approximation errors (see Tran (2008)). Formally speaking, for any fixed $K \in \mathbb{N}$, the first J FPCs, $\phi_j, j = 1, \ldots, J$, satisfy

$$\{\phi_j, j = 1, \ldots, J\} = \underset{\langle \phi_\ell, \phi_j \rangle = \delta_{\ell j}}{\arg\min} \sum_{i=1}^{n} \left|\left| x_i(t) - \sum_{j=1}^{J} \langle x_i, \phi_j \rangle \phi_j \right|\right|^2.$$

This 'best-approximation' point of view inspires us to estimate the FPCs by searching for the optimal basis functions to approximate $x_i(t), i = 1, \ldots, n$.

To the best of our knowledge, our proposed method is the first attempt to numerically estimate FPCs from this 'best-approximation' point of view. We will show that the first empirical leading FPC is the solution of a least square optimization with some constraints. More specifically, let $\hat{\beta}(t)$ be the solution of minimizing

$$\frac{1}{n}\sum_{i=1}^{n} \left|\left| x_i(t) - \alpha(t)\langle \beta, x_i \rangle \right|\right|^2 + \tau \int \beta^2(t)dt$$

with respect to $\beta(t)$ and the ancillary parameter $\alpha(t)$, with the constaint $||\alpha||^2 = 1$, then $\hat{\beta}(t) = c\hat{\phi}_1(t)$, where $\hat{\phi}_1(t)$ is the first FPC, and $c$ is a constant scale factor. The detailed proof will be provided for Theorem 3.4.1 in Section 3.4.

Similarly, for the first $J$ leading FPCs, let $\hat{\beta}_1(t), \ldots, \hat{\beta}_J(t)$ be the solution of minimizing

$$\frac{1}{n}\sum_{i=1}^{n}\left|\left|x_i(t) - \sum_{j=1}^{J}\alpha_j(t)\langle\beta_j, x_i\rangle\right|\right|^2 + \tau\sum_{j=1}^{J}\int\beta_j^2(t)dt$$

with respect to $\beta_j(t)$ and the ancillary parameter $\alpha_j(t)$, $j = 1, \ldots, J$, with the constaints $\langle\alpha_i, \alpha_j\rangle = \delta_{ij}$. Then $\hat{\beta}_j(t) = c_j\hat{\phi}_j(t), j = 1 \ldots, J$, where $\hat{\phi}_j(t)$ is the $j$th FPC and $c_j$ is a scale factor. The detailed proof will be provided for Theorem 3.4.2 in Section 3.4. Theorem 3.4.2 shows that when the sparsity parameter $\lambda_j = 0$, the corresponding estimated sparse FPC, $\hat{\xi}_j(t)$, is equivalent to the conventional FPC, $\hat{\phi}_j(t)$. Therefore, the FPCs estimated with our proposed sparse FPCA method is equivalent to the FPCs with the conventional FPCA method when the sparsity parameter is zero.

## 3.3 Estimation Method

We propose to estimate the first $J$ unnormalized sparse FPCs, $\beta_1(t), \ldots, \beta_J(t)$, in a iterative optimization method. More specifically, within each iteration, the first step is to find the optimal $\beta_j(t)$ that minimizes the criterion (3.1) given the current estimate of $\alpha_j(t)$ and the second step is to search for a new $\alpha_j(t)$ which further minimizes the proposed criterion conditional on the optimal $\beta_j(t)$ from the first step. This procedure is repeated until it converges. In the rest of this section, we first give the details of these two steps. Then we discuss the tuning parameter selection and the adjusted variance explained in the end.

### 3.3.1 Estimate $\beta_j(t)$ for Given $\alpha_j(t)$

Given the $j$th $\alpha_j(t)$, the corresponding $\beta_j(t)$ is obtained by minimizing

$$\begin{aligned} Q(\beta_j(t)) = &\frac{1}{n}\sum_{i=1}^{n}\left|\left|\langle x_i, \alpha_j\rangle - \langle\beta_j, x_i\rangle\right|\right|^2 \\ &+ \tau\int\beta_j^2(t)dt + \int p_\lambda(|\beta_j(t)|)dt + \gamma\int\left[\frac{d^2\beta_j(t)}{dt^2}\right]^2 dt. \end{aligned}$$

Note that besides the sparsity penalty, we also add a roughness penalty with a smoothing parameter $\gamma$ to achieve smoothness for the resulting SFPC. A larger value of $\gamma$ will prevent the estimated SFPC from being too 'wiggly'. Without any parametric assumption on $\beta_j(t)$, we first represent $\beta_j(t)$ as a linear combination of basis functions

$$\beta_j(t) = \sum_{m=1}^{M} b_{jm}\psi_m(t) = \boldsymbol{\psi}(t)^T\mathbf{b}_j, \tag{3.5}$$

where $\boldsymbol{\psi}(t) = (\psi_1(t), \psi_2(t), \ldots, \psi_M(t))^T$ denotes the vector of B-spline basis functions, $\mathbf{b}_j$ is the corresponding vector of basis coefficients, and $M$ denotes the number of basis functions.

For simplicity, we recast each part in $Q(\beta_j(t))$ into a matrix form. Let $\mathbf{a}_j = (a_{1j}, a_{2j}, \ldots, a_{nj})^T$ with $a_{ij} = \int x_i(t)\alpha_j(t)dt$, then the first term in the loss function can be expressed as

$$\sum_{i=1}^{n} ||\langle x_i, \alpha_j\rangle - \langle \beta_j, x_i\rangle dt||^2 = (\mathbf{a}_j - \mathbf{Z}\mathbf{b}_j)^T(\mathbf{a}_j - \mathbf{Z}\mathbf{b}_j). \tag{3.6}$$

Here $\mathbf{Z}$ is a $n \times M$ matrix with entries $z_{ij} = \int x_i(t)\psi_j(t)dt$ for $1 \leq i \leq n$ and $1 \leq j \leq M$. The second term in the loss function can be expressed as

$$\tau \int \beta^2(t)dt = \tau \mathbf{b}_j^T \mathbf{\Psi} \mathbf{b}_j, \tag{3.7}$$

in which $\mathbf{\Psi}$ denotes a $M \times M$ matrix with entries $\mathbf{\Psi}_{ij} = \int \psi_i(t)\psi_j(t)dt$ for $1 \leq i, j \leq M$. The roughness penalty term in the loss function can be expressed as

$$\gamma \int \left[\frac{d^2\beta_j(t)}{dt^2}\right]^2 dt = \gamma \mathbf{b}_j^T \mathbf{R} \mathbf{b}_j \tag{3.8}$$

in which $\mathbf{R}$ denotes a $M \times M$ matrix with the (i,j)-th entry $\int \psi_i''(t)\psi_j''(t)dt$ for $1 \leq i, j \leq M$.

The sparsity penalty term in the loss function, as shown in Lin et al. (2016), can be approximated as

$$\int p_\lambda(|\beta_j(t)|)dt \approx \frac{T}{M-d} \sum_{\ell=1}^{M-d} p_\lambda\Big(\sqrt{\int_{t_{\ell-1}}^{t_\ell} \beta_j^2(t)dt}\Big),$$

in which $t_0, t_1, \ldots, t_{M-d}$ denote the sequence of the knots of B-spline basis functions $\boldsymbol{\psi}(t)$, and $d$ denotes the order of the basis functions. We further define

$$||\beta_{[\ell]}(t)||_2^2 \overset{def}{=} \int_{t_{\ell-1}}^{t_\ell} \beta_j^2(t)dt = \mathbf{b}_j^T \mathbf{\Psi}_j \mathbf{b}_j,$$

in which $\mathbf{\Psi}_j$ denotes a $M \times M$ matrix with the (p,q)-entry as $\int_{t_{j-1}}^{t_j} \psi_p(t)\psi_q(t)dt$ when $j \leq p, q \leq j + d$ and zero elsewhere. Using the local quadratic approximation (LQA) method proposed in Fan and Li (2001), given some initial estimate $\mathbf{b}_j^{(0)}$, we can derive that

$$\int p_\lambda(|\beta_j(t)|)dt \approx \frac{T}{M-d}\Big[\mathbf{b}_j^T \boldsymbol{W}^{(0)} \mathbf{b}_j + G(\mathbf{b}_j^{(0)})\Big], \tag{3.9}$$

where

$$\boldsymbol{W}^{(0)} = \frac{1}{2} \sum_{\ell=1}^{M-d} \Big(\frac{p'_\lambda(||\beta_{[\ell]}(t)||_2\sqrt{M-d/T})}{||\beta_{[\ell]}(t)||_2\sqrt{T/M-d}} \mathbf{\Psi}_j\Big),$$

and

$$G(\boldsymbol{\beta}^{(0)}) \equiv \sum_{\ell=1}^{M} p_\lambda\Big(\frac{||\beta_{[\ell]}^{(0)}||_2}{\sqrt{T/M-d}}\Big) - \frac{1}{2}\sum_{\ell=1}^{M} p'_\lambda\Big(\frac{||\beta_{[\ell]}^{(0)}||_2}{\sqrt{T/M-d}}\Big)\frac{||\beta_{[\ell]}^{(0)}||_2}{\sqrt{T/M-d}}.$$

Putting (3.6),(3.7),(3.8) and (3.9) together, we obtain

$$Q(\beta_j(t)) = \frac{1}{n}(\mathbf{a}-\mathbf{Z}\boldsymbol{\beta})^T(\mathbf{a}-\mathbf{Z}\boldsymbol{\beta}) + \tau\boldsymbol{\beta}^T\boldsymbol{\Psi}\boldsymbol{\beta} + \gamma\boldsymbol{\beta}^T\mathbf{R}\boldsymbol{\beta} + \frac{T}{M-d}\boldsymbol{\beta}^T\boldsymbol{W}^{(0)}\boldsymbol{\beta} + \frac{T}{M-d}G(\boldsymbol{\beta}^{(0)}).$$

By minimizing $Q(\beta_j(t))$, we obtain the estimate for the basis coefficients

$$\widehat{\boldsymbol{\beta}} = \Big(\frac{1}{n}\mathbf{Z}^T\mathbf{Z} + \tau\boldsymbol{\Psi} + \gamma\mathbf{R} + \frac{T}{M-d}\mathbf{W}^{(0)}\Big)^{-1}\mathbf{Z}^T\mathbf{a}.$$

Then we plug the estimate $\widehat{\boldsymbol{\beta}}$, into (3.5) to obtain the estimates for $\beta_j(t)$:

$$\widehat{\beta}(t) = \boldsymbol{\phi}(t)^T\widehat{\boldsymbol{\beta}}.$$

### 3.3.2 Estimate $\alpha_j(t)$ for Given $\beta_j(t)$

Let $\boldsymbol{\alpha}(t) = (\alpha_1(t),\dots,\alpha_J(t))^T$, $\boldsymbol{\beta}(t) = (\beta_1(t),\dots,\beta_J(t))^T$, and $u_{ij} = \langle\beta_j, x_i\rangle$. We obtain the estimate for $\boldsymbol{\alpha}(t)$ by minimizing

$$\begin{aligned} Q(\boldsymbol{\alpha}) &= \sum_{i=1}^{n}\Big|\Big|x_i(t) - \sum_{j=1}^{J}\alpha_j(t)u_{ij}\Big|\Big|^2 \\ &= \sum_{i=1}^{n}||x_i||^2 - \int 2\sum_{i=1}^{n}x_i(t)\Big(\sum_{j=1}^{J}\alpha_j(t)u_{ij}\Big)dt + \int\sum_{i=1}^{n}\Big(\sum_{j=1}^{J}\alpha_j(t)u_{ij}\Big)^2 dt. \end{aligned}$$

First, we can see that the first term is equivalent to the sum of norm of each observed $x_i(t)$, which does not depend on the value of $\boldsymbol{\alpha}(t)$. Second, the last term can be recast into:

$$\int\sum_{i=1}^{n}\Big(\sum_{j=1}^{J}\alpha_j(t)u_{ij}\Big)^2 dt = \int\sum_{i=1}^{n}\sum_{j=1}^{J}\alpha_j^2(t)u_{ij}^2 + \sum_{i=1}^{n}\sum_{l<k}2\alpha_l(t)u_{il}\alpha_k(t)u_{ik}dt = \sum_{i=1}^{n}u_{ij}^2$$

due to the fact that $\langle\alpha_i,\alpha_j\rangle = \delta_{ij}$. Thus, the last term does not depend on the value of $\boldsymbol{\alpha}(t)$, either. Therefore, minimizing $Q(\boldsymbol{\alpha})$ is equivalent to minimizing the second term. We can further recast the second term into the following form:

$$\begin{aligned}\int \sum_{i=1}^{n} x_i(t)\Big(\sum_{j=1}^{J}\alpha_j(t)u_{ij}\Big)dt &= \int\Big(\sum_{j=1}^{J}\alpha_j(t)\int\sum_{i=1}^{n}(x_i(t)x_i(s))\beta_j(s)ds\Big)dt \\ &= n\int\Big(\sum_{j=1}^{J}\alpha_j(t)\int g(t,s)\beta_j(s)ds\Big)dt \\ &= n\int\Big(\sum_{j=1}^{J}\alpha_j(t)\sum_{k=1}^{K}\lambda_k\langle\beta_j,\phi_k\rangle\phi_k(t)\Big)dt,\end{aligned}$$

where $\phi_k, k = 1, \ldots, K$, denotes the empirical eigenfunctions obtained from decomposing the sample covariance function $g(t,s)$ as discussed in Equation (3.2) and $\lambda_k$ is the corresponding eigenvalues. The last step in the above equation uses the fact that $g(t,s) = \sum_{k=1}^{K}\{\lambda_k\phi_k(t)\phi_k(s)\}$, where $K$ is the number of nonzero eigenvalues of the sample covariance function, and $K \le n$. To simplify the notation, let $\xi_j(t) = \sum_{k=1}^{K}\lambda_k\langle\beta_j,\phi_k\rangle\phi_k(t)$, so that the above equation becomes

$$\int \sum_{i=1}^{n} x_i(t)\Big(\sum_{j=1}^{J}\alpha_j(t)u_{ij}\Big)dt = n\int\Big(\sum_{j=1}^{J}\alpha_j(t)\xi_j(t)\Big)dt. \tag{3.10}$$

To maximize this term with respect to $\alpha_j(t)$, we first express $\{\xi_j(t), j = 1, \ldots, J\}$ by its eigenfunctions. That is,

$$\xi_j(t) = \sum_{l=1}^{J}\sqrt{\kappa_l}\frac{\rho_{jl}}{\sqrt{\kappa_l}}g_l(t), \tag{3.11}$$

in which $g_l(t), l = 1, \ldots, J$, denotes the $l$-th FPC for $\{\xi_j(t), j = 1, \ldots, J\}$, $\kappa_l$ is the corresponding eigenvalue, and $\rho_{jl} = \langle\xi_j, g_l\rangle$ is the corresponding FPC scores. Then we plug equation (3.11) back into equation (3.10):

$$\begin{aligned}\int \sum_{i=1}^{n} x_i(t)\Big(\sum_{j=1}^{J}\alpha_j(t)u_{ij}\Big)dt &= n\int\Big(\sum_{j=1}^{J}\alpha_j(t)\sum_{l=1}^{J}\sqrt{\kappa_l}\frac{\rho_{jl}}{\sqrt{\kappa_l}}g_l(t)\Big)dt \\ &= n\sum_{j=1}^{J}\sqrt{\kappa_l}\sum_{l=1}^{J}\frac{\rho_{jl}}{\sqrt{\kappa_l}}\int\Big(\alpha_j(t)g_l(t)\Big)dt.\end{aligned}$$

By the Cauchy-Swartz inequality, the above equation is maximized when

$$\int \alpha_j(t)g_l(t)dt \propto \frac{\rho_{jl}}{\sqrt{\kappa_l}}.$$

Note that $g_l(t), l = 1, \ldots, J$, can be viewed as the orthogonal basis functions that $\alpha_j(t)$ is projected onto. Therefore, the solution should be given as

$$\hat{\alpha}_j(t) = \sum_{l=1}^{J} \frac{\rho_{jl}}{\sqrt{\kappa_l}} g_l(t), j = 1, \ldots, J.$$

Putting into a matrix form, we have

$$\widehat{\boldsymbol{\alpha}}(t) = \mathbf{P}\mathbf{g}(t),$$

in which $\mathbf{P}$ is a $J \times J$ matrix with (j,l)-th element being $\frac{\rho_{jl}}{\sqrt{\kappa_l}}$. To check whether the resulting $\widehat{\boldsymbol{\alpha}}(t)$ satisfies the orthonormal condition, we can see that the coefficients matrix

$$\mathbf{P} = \begin{bmatrix} \frac{\rho_{11}}{\sqrt{\kappa_1}} & \cdots & \frac{\rho_{1J}}{\sqrt{\kappa_J}} \\ \vdots & \cdots & \vdots \\ \frac{\rho_{J1}}{\sqrt{\kappa_1}} & \cdots & \frac{\rho_{JJ}}{\sqrt{\kappa_J}} \end{bmatrix}$$

is an orthogonal matrix because $\mathbf{P}^T\mathbf{P} = \mathbf{I}$. Therefore, the resulting $\hat{\alpha}_j(t)$ satisfy $\langle \hat{\alpha}_i, \hat{\alpha}_j \rangle = \delta_{ij}$.

### 3.3.3 Detailed Algorithms

Below we summarize the proposed estimation method step by step:

Step I: Initialize $\alpha_j^{(0)}(t) = \hat{\phi}_j(t), j = 1, \ldots, J$, where $\hat{\phi}_j(t)$ is the estimated FPC using the conventional FPCA method, which satisfy $||\hat{\phi}_j||^2 = 1$ and $\langle \hat{\phi}_i, \hat{\phi}_j \rangle = 0, i \neq j$;

Step II: Given $\alpha_j^{(i)}(t)$, obtain the corresponding $\beta_j^{(i)}(t)$ by minimizing

$$Q_{\tau,\lambda,\gamma}(\beta(t)) = \frac{1}{n}(\sum_{i=1}^{n} || \int x_i(t)\alpha^{(i)}(t)dt - \langle \beta, x_i \rangle||^2 + \tau \int \beta^2(t)dt + \gamma \int \beta''^2(t)dt + \int p_\lambda(|\beta(t)|)dt.$$

Due to the fact that $\langle \alpha_k^{(i)}, \alpha_l^{(j)} \rangle = 0$ for $k \neq l$, we can obtain each $\beta_j^{(i)}(t)$ separately. The details of this step is discussed in Section 3.3.1.

Step III: Given $\beta_1^{(i)}(t), \ldots, \beta_J^{(i)}(t)$, obtain the corresponding $\alpha_1^{(i+1)}(t), \ldots, \alpha_J^{(i+1)}(t)$ by minimizing

$$Q(\boldsymbol{\alpha}(t)) = \frac{1}{n}\sum_{i=1}^{n} ||x_i(t) - \sum_{j=1}^{J} \alpha_j(t) \int \beta_j^{(i)}(t)x_i(t)dt||^2 + \text{Constant}.$$

The ‘Constant’ term represents $\sum_{j=1}^{J} \tau \int \{\beta_j^{(i)}\}^2(t)dt$ and the remaining constant terms when $\beta^{(i)}(t)$ is given. Unlike Step II, we can obtain $\alpha_1^{(i+1)}(t), \ldots, \alpha_J^{(i+1)}(t)$ simultaneously. The details of this step is provided in Section 3.3.2.

Step IV: Repeat Step II to Step III until they converge.

### 3.3.4 Choosing Tuning Parameters

There are three tuning parameters: the ridge-type parameter $\tau$, the sparsity parameter $\lambda$, and the smoothing parameter $\gamma$. The ridge-type parameter $\tau$ is only required to be positive by Theorem 3.4.2. A larger value of $\tau$ will force the norm of the estimated $\widehat{\beta}(t)$ to be smaller. Our numerical experience suggests the largest empirical eigenvalue $\lambda_i$ is a proper value to set for $\tau$. The sparsity parameter $\lambda$ controls the sparsity of the estimated sparse FPCs $\widehat{\boldsymbol{\beta}}(t)$. Note that the more compact the resulting sparse FPCs are, the less variation they can explain for the original functional data in comparison to the conventional FPCs or equivalently the errors of approximating the original functional data become larger. We recommend choosing a $\lambda$ that balances between sparsity and the errors of approximating the original functional data. In addition, the smoothing parameter $\gamma$ prevents the resulting $\beta(t)$ from being too ‘wiggly’. Again, a smoother $\boldsymbol{\beta}(t)$ explains less variation of the functional data and we recommend to choose a value that balances these two aspects. In practice, one can choose both the smoothing parameter and the sparsity parameter by cross-validation. More specifically, one can split the data into the training and test datasets. Then we can use the proposed sparse FPCA method to obtain the estimated sparse FPCs. Next, we can use the resulting sparse FPCs to recover the trajectory in the test set and compare it with what is observed to obtain the cross-validation errors. However, this two-dimensional cross-validation method might be computationally intensive in practice. We alternatively propose a two-step procedure to choose the tuning parameter. Our computational experience suggests this two-step procedure yields reasonable results.

In the first step, the smoothing parameter $\gamma$ is chosen using cross-validation as described in Ramsay and Silverman (2005). In the second step, we introduce the following AIC criterion to select the sparsity parameter $\lambda$:

$$\text{AIC} = n \log\left(\frac{\sum_{i=1}^{n} ||x_i(t) - \hat{x}_i(t)||^2}{nT}\right) + 2df(\lambda), \tag{3.12}$$

where $x_i(t)$ and $\hat{x}_i(t) = \sum_{k=1}^{K} \langle x_i, \hat{\phi}_k \rangle \hat{\phi}_k$ represent the $i$th sample curve and its corresponding estimate using the estimated FPCs, respectively. The degree of freedom is the number of nonzero bspline coefficients in the estimated FPCs under different values of the sparsity parameter $\lambda$. The goal is to balance the errors of approximating functional data and the length of the support regions of the estimated FPCs. We will demonstrate this procedure in both real data application and simulation studies.

### 3.3.5 Adjusted Total Variance Explained

Due to the fact that the sparse functional principal component scores are not necessarily uncorrelated, the variance explained by the $j$th SFPC is not simply the variance of the corresponding SFPC scores and need to take the correlation between SFPCs into account. Here we propose a new approach to compute the total variance explained by the $j$th SFPC. Let $\phi_j(t)$ and $\mathbf{s}_j$ denote the $j$th SFPC and the corresponding score vector. We regress $\mathbf{s}_j$ on $\mathbf{s}_1, \ldots, \mathbf{s}_{j-1}$ and denote the resulting residuals as $\mathbf{r}_j$. Then the adjusted variance explained by $\phi_j(t)$ is $||\mathbf{r}_j||^2$.

## 3.4 Theoretical Results

In this section we show that the empirical FPCs can be obtained by minimizing the mean $L^2$ errors to the observed function data $x_i(t)$. We start with only the first leading FPC in Theorem 3.4.1 and extend to the first $J$ leading FPCs in Theorem 3.4.2. The detailed proofs for these two theorems are in the Appendix A.

**Theorem 3.4.1.** For any $\tau > 0$, let

$$\hat{\beta}(t) = \arg\min \frac{1}{n}\sum_{i=1}^{n} \left|\left| x_i(t) - \alpha(t)\langle \beta, x_i\rangle \right|\right|^2 + \tau \int \beta^2(t)dt \tag{3.13}$$

subject to $||\alpha||^2 = 1$, then $\hat{\beta}(t) = c\hat{\phi}_1(t)$, where $\hat{\phi}_1(t)$ is the first empirical eigenfunctions of the sample covariance function $g(s,t) = \frac{1}{n}\sum_{i=1}^{n} x_i(s)x_i(t)$ and $c$ is a constant scale factor.

The next theorem extends Theorem 3.4.1 into the first $J$ leading FPCs.

**Theorem 3.4.2.** Let $\boldsymbol{\alpha}(t) = (\alpha_1(t), \ldots, \alpha_J(t))$ and $\boldsymbol{\beta}(t) = (\beta_1(t), \ldots, \beta_J(t))$. For any $\tau > 0$, let

$$(\widehat{\boldsymbol{\alpha}}(t), \widehat{\boldsymbol{\beta}}(t)) = \arg\min \frac{1}{n}\sum_{i=1}^{n} ||x_i(t) - \sum_{j=1}^{J} \alpha_j(t)\langle \beta_j, x_i\rangle dt||^2 + \tau \sum_{j=1}^{J} \int \beta_j^2(t)dt$$

subject to $\langle \alpha_i, \alpha_j\rangle = \delta_{ij}$ and $\delta_{ij}$ is the Kronecker delta, then $\hat{\beta}_j(t) = c_j\hat{\phi}_j(t), j = 1, \ldots, J$, where $\hat{\phi}_j(t)$ is the $j$-th empirical eigenfunctions of the sample covariance function $g(s,t) = \frac{1}{n}\sum_{i=1}^{n} x_i(s)x_i(t)$ and $c_j$ is a scale factor.

## 3.5 Application

Our proposed method is demonstrated by analyzing a real dataset relating to particulate matter (PM) emissions from diesel trucks (Clark et al., 2007). In the experiment, trucks are driven through a pre-determined driving cycle and PM at the exhaust pipe is measured every second via a particulate matter counter. Hall and Hooker (2015) analyzed this dataset to predict PM using the acceleration rate with a functional linear model. Figure 3.1 displays the acceleration rate curves for 107 diesel trucks. We will demonstrate our proposed sparse FPCA method by analyzing the major variations among these acceleration curves.

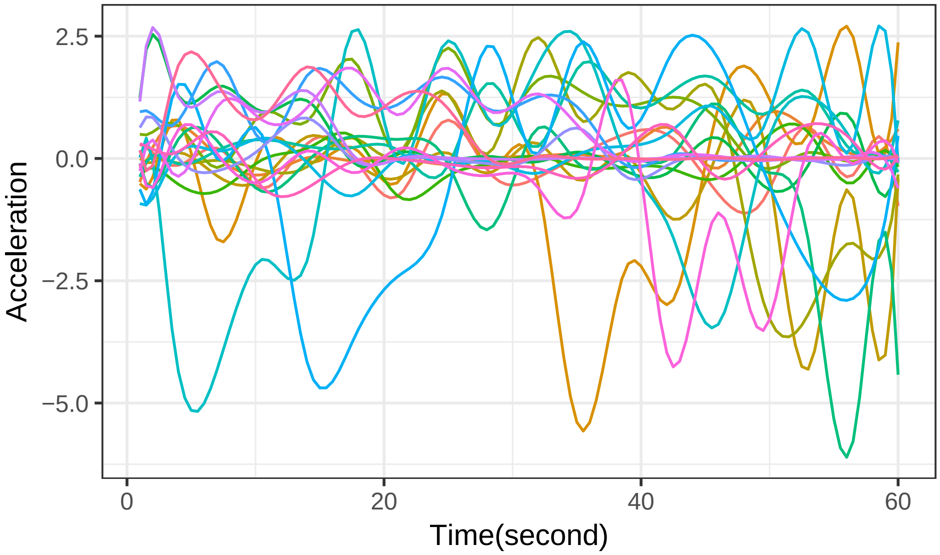

Figure 3.1: The observed acceleration rates for 20 randomly selected diesel trucks out of all 107 diesel trucks. Each curve respects one truck's observations.

We first applied the conventional FPCA method (Ramsay and Silverman, 2005) to analyze the major variations among these acceleration curves. The top four estimated FPCs are shown in Figure 3.2. They account for 25.7%, 24.6%, 17.4% and 15.3% of the total variation among the acceleration curves, respectively. In total, the first four FPCs explain 83.0% of the total variation. As expected, these estimated FPCs are nonzero on the entire time domain.

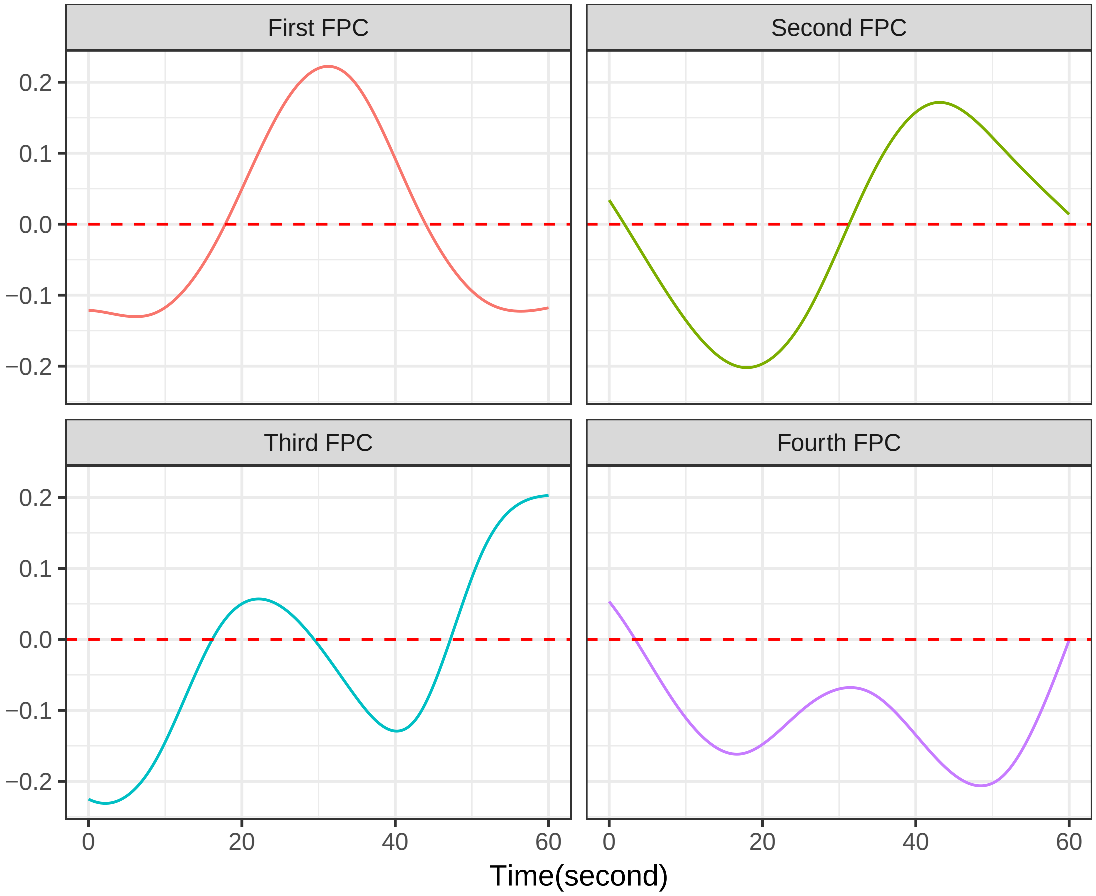

Figure 3.2: The estimated first four leading functional principal components using conventional FPCA for analyzing the acceleration curves. They account for 25.7%, 24.6%, 17.4% and 15.3% of the total variation among the acceleration curves, respectively.

We then apply our proposed sparse FPCA method to analyze the major variations among these acceleration curves. For the purpose of comparison, we also estimate the first four sparse FPCs. We select the sparsity parameter $\lambda = 40$ based on the AIC values given in Table 3.1. The corresponding estimated sparse FPCs are shown in Figure 3.3. They account for 20%, 19%, 17% and 14% of the total variation among the acceleration curves, respectively. The total variation of the first four sparse FPCs is 70%.

Figure 3.3 shows that the estimated sparse FPCs tend to be nonzero at different intervals along the whole time domain. For instance, the first sparse FPC is nonzero roughly in [25,45], which indicates that the major variation among the acceleration curves comes from this interval. Compared to the conventional FPCs, which are nonzero everywhere, the estimated SFPCs are nonzero at different intervals. More specifically, the second sparse FPC is nonzero between $[10, 25]$, the third sparse FPC is nonzero between $[40, 60]$ and the last sparse FPC is nonzero between $[0, 10]$. This observation suggests the major variation within the acceleration curves can be separated into different subintervals rather than mixing with each other in the entire domain.

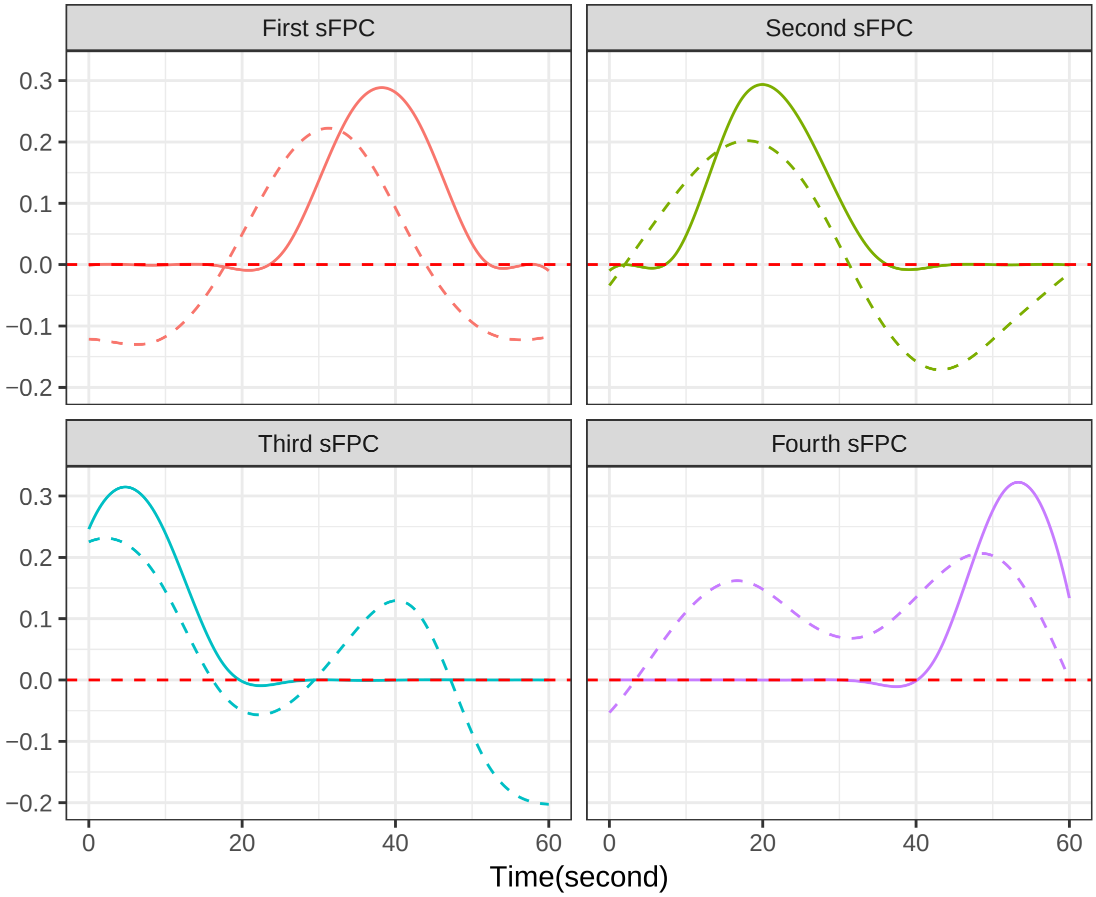

Figure 3.3: Estimated sparse functional principal components (solid line) compared to the conventional smoothing functional principal components (dashed line). They account for 20%, 19%, 17% and 14% of the total variation, respectively

Table 3.1: The AIC value defined in (3.12) when the sparsity parameter $\lambda$ varies.

| $\lambda$ | 0 | 1 | 20 | 40 | 80 | 100 |
|---|---|---|---|---|---|---|
| AIC | 14.68 | 14.63 | 5.83 | -12.19 | 5.74 | 0.18 |

## 3.6 Simulation Study

We conduct a simulation study to evaluate our proposed sparse FPCA method by comparing it with three available methods, including the conventional FPCA method (Ramsay and Silverman, 2005), the interpretable functional principal component analysis (iFPCA) proposed by Lin et al. (2016) and the localized functional principal component analysis (LFPCA) proposed by Chen and Lei (2015).

More specifically, the true underlying functional curves are generated using

$$X_i(t) = s_{i1}\xi_1(t) + s_{i2}\xi_2(t) + s_{i3}\xi_3(t) + s_{i4}\xi_4(t),$$

$t \in [1, 60]$, where $\xi_k(t), k = 1, 2, 3, 4$, are obtained from the real data application as shown in Figure 3.3, and $(\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3, \mathbf{s}_4)$ are generated from multivariate normal distribution with mean zero and the variance-covariance matrix $\Sigma = \text{diag}(30, 20, 10, 3)$. The observed trajectories are generated by $Y_{ij} = X_i(t_j) + \epsilon_{ij}$ for $j = 1, \dots, 60$, where $t_j$ is the $j$-th observed point equally spaced in $[1, 60]$ and $\epsilon_{ij} \overset{i.i.d}{\sim} N(0, 1)$. For the iFPCA method, following the authors' recommendation, the smoothing parameter is chosen by CV and the sparsity parameter is also selected using CV after the smoothing parameter is determined. For the LFPCA method, we also use the recommended CV method to determine the tuning parameters. For the proposed sparse FPCA method, we use AIC as our criterion to choose the tuning parameters as described in Section 3.3.4.

We compare the performance of the four methods using the integrated error (IE) defined as follows:

$$\text{IE}(\hat{\boldsymbol{\xi}}) = \sum_{k=1}^{4} \int (\xi_k(t) - \hat{\xi}_k(t))^2 dt, \tag{3.14}$$

in which $\xi_k(t)$ and $\hat{\xi}_k(t)$ represent the $k$-th true FPC and the corresponding estimated FPC, respectively. The results in Figure 3.4 show that the sparse FPCA method yields the lowest integrated errors in comparison with all three alternative methods. We plot those estimated FPCs from one simulation replicate in Figure 3.5 using sparse FPCA, LFPCA and iFPCA. We can see that the estimated FPCs from the sparse FPCA method are closest to the true FPCs. More specifically, iFPCA and LFPCA perform similarly as the sparse FPCA method in estimating the first two FPCs, but they become worse than the sparse FPCA method when estimating the 3rd and 4th FPCs.

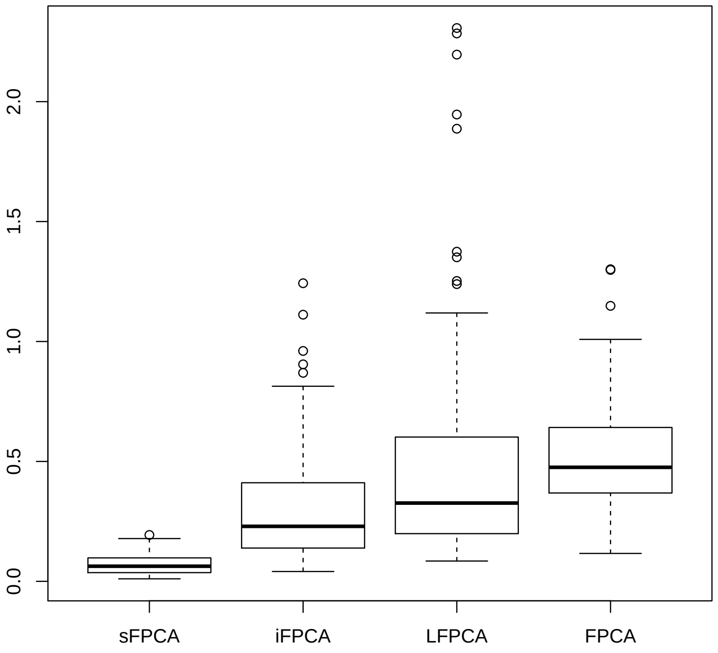

Figure 3.4: Boxplot of the integrated errors (3.14) for four methods including the sparse FPCA method, the conventional FPCA method, iFPCA and LFPCA in 100 simulation repetitions.

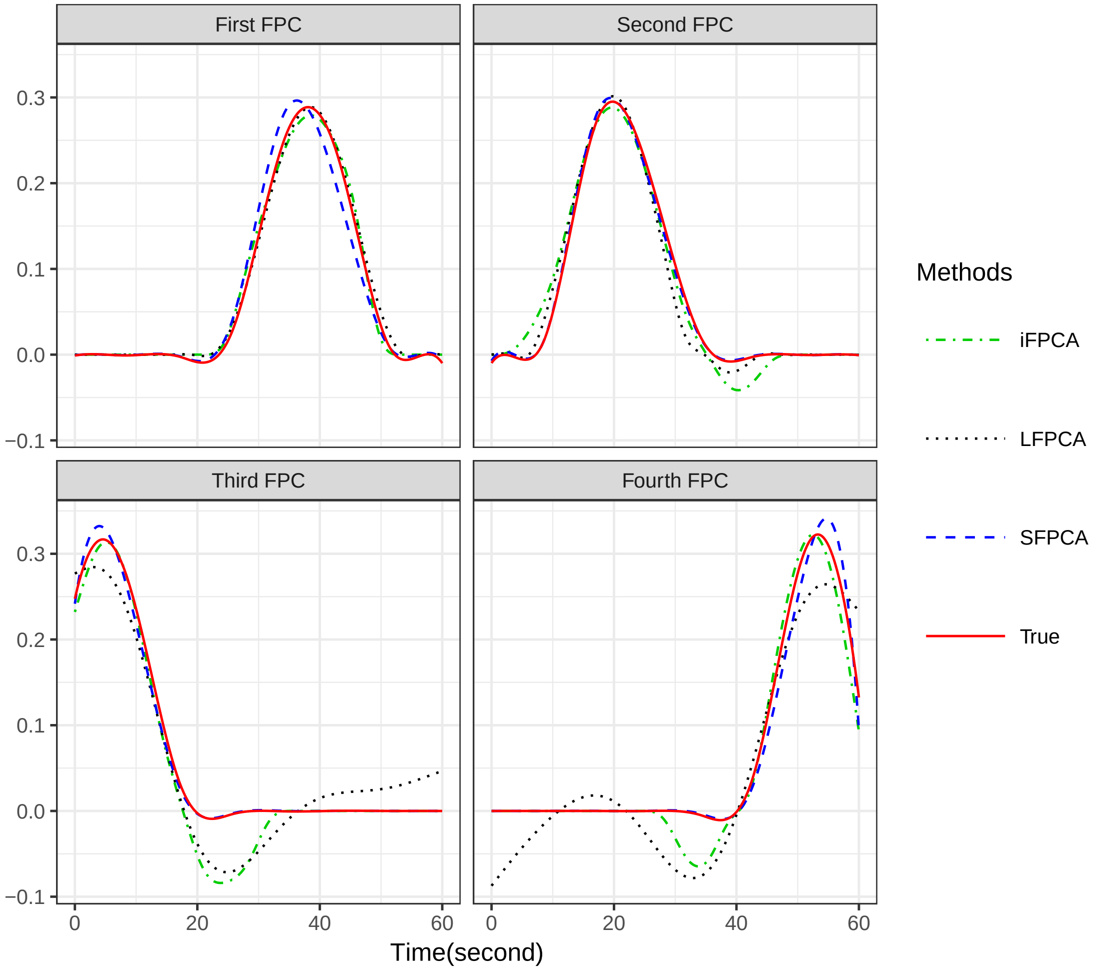

Figure 3.5: The estimated FPC using different methods including iFPCA (dot dashed line) LFPCA (dot line) and sparse FPCA (dashed line) in comparison with the true FPCs (solid line) in one simulation replicate.

## 3.7 Summary

In this chapter, we focused on the problem of estimating FPCs with the compact support. The conventional FPCA method estimate FPCs by maximizing the variation among the functional data. But these estimated FPCs are nonzero in the entire domain, hence they are often hard to interpret.

Conventional FPCA methods estimate FPCs by eigendecomposing the sample covariance function. However, when we need to add the regulation penalty to the FPCs, this eigendecomposition method always leads to an NP-hard problem. We proposed a new regression framework to estimate the sparse FPCs by minimizing the errors of approximating functional data. One major advantage of our framework is that the optimization problem is not NP-hard when adding a penalty term to regulate the FPCs. We also showed that the FPCs estimated with our proposed sparse FPCA method is equivalent to the FPCs with the conventional FPCA method when the sparsity parameter is zero.

Our sparse FPCA method was applied to explore the major variations among the acceleration rate curves of 107 diesel trucks. We found that the nonzero regions of the estimated sparse FPCs are well separated, which shows that the major variation within the acceleration curves can be separated into different subintervals rather than mixing with each other in the entire domain. We also compare our proposed sparse FPCA method with the conventional FPCA method (Ramsay and Silverman, 2005), the interpretable functional principal component analysis (Lin et al., 2016) and the localized functional principal component analysis (Chen and Lei, 2015) using a simulation study. The simulation study shows that the sparse FPCA method obtains more accurate estimates of FPCs in comparison with the alternative three methods when the true FPCs has compact support regions.

# Chapter 4

# Recovering the Underlying Trajectory from Sparse and Irregular Longitudinal Data

## 4.1 Introduction

Functional principal component analysis (FPCA) is a key dimension reduction tool in functional data analysis. FPCA explores major sources of variability in a sample of random curves by finding functional principal components (FPCs) that maximize the curve variation. Consequently, the top few FPCs explain most of the variability in the random curves. In addition, each random curve can be approximated by a linear combination of the top FPCs. Therefore, the infinite-dimensional curves are projected to a low-dimensional space defined by the top FPCs. This powerful dimensional reduction feature also contributes to the popularity of FPCA.

The theoretical properties of FPCA have been carefully studied at length. For example, Dauxois et al. (1982) first studied the asymptotic properties of PCA estimators for the infinite dimensional data from a linear operator perspective. Following this point of view, Mas (2002) and Bosq (2000) utilized functional analysis to study FPCA theoretically. On the other hand, Hall and Horowitz (2007); Hall et al. (2006); Yao et al. (2005a) studied FPCA from the kernel perspective. The smooth version of FPCA is carefully studied by Rice and Silverman (1991); Pezzulli (1993); Silverman (1996); Yao et al. (2005a). There are mainly three methods to achieve smoothness. The first method smooths the functional data in the first step and conducts the regular FPCA on the sample covariance function. The second method smooths the covariance function first and then eigen-decomposes the resulting smoothed covariance function to estimate the smoothed FPCs. The last method directly adds a roughness penalty in the optimization criterion for estimating FPCs.

FPCA has been widely and successfully applied in many applications such as functional linear regression (Cardot et al., 1999), classification and clustering of functional data (Ramsay and Silverman (2005); Müller (2005); Müller and Stadtmüller (2005); Peng and

Müller (2008)). All these applications assume the functional data are densely and regularly observed.

When the functional data are sparsely and irregularly observed, it is challenging to obtain a good estimate for FPCs and the corresponding FPC scores. Yao et al. (2005a) proposed the Principal Analysis by Conditional Estimation (PACE) method to analyze the sparse functional data. The PACE method estimates the covariance function by the local polynomial regression method and then eigen-decomposes the estimated covariance function to obtain the eigenfunctions as the estimates of FPCs. The corresponding FPC score is estimated using conditional expectation, which requires that FPC scores follow a Gaussian distribution. The asymptotic properties are established in Hall et al. (2006).

The PACE method is very successful. It is now popularly used to analyze sparse functional data. On the other hand, the PACE method also has two major assumptions, which may limit its applications. The first assumption of PACE is that the assembled time points with observations for all subjects are sufficiently dense. Otherwise, PACE cannot estimate the mean and covariance function by pooling data for all subjects together. The second assumption of PACE is that the FPC scores follow a Gaussian distribution. Otherwise, the conditional expectation formula is invalid. In addition, the PACE method involves the inverse of the estimated covariance matrix when estimating individual trajectories, which may be unstable. This problem will be demonstrated in our simulation studies. Peng and Paul (2009) proposed a restricted maximum likelihood approach to estimate FPCs and apply a Newton-Raphson procedure on the Stiefel manifold to guarantee that the resulting FPCs satisfies the orthonormality constraints. They also used conditional expectation to obtain FPC scores in order to recover individual trajectories. Therefore, their method also involves the inverse of the estimated covariance matrix and requires the FPC scores to be Gaussian distributed.

The main objective of this chapter is to recover the underlying trajectory given sparse and irregular longitudinal observations. Note that this objective is different from exploring the major variation patterns of the functional data, which is the central goal for the conventional FPCA.

We propose a new sparse orthonormal approximation (SOAP) method to recover the underlying trajectory. This method directly estimates the optimal empirical basis functions and the corresponding coefficients in the best approximation framework. The SOAP method has three main advantages. First, our method avoids the inverse of the covariance matrix, and the computation is stable and efficient. Second, it does not require that the scores follow the Gaussian distribution. Therefore, it can be applied in non-Gaussian cases. Lastly, our method does not need to estimate the mean and covariance function, which might be challenging when the assembled time points with observations for all subjects are not sufficiently dense.

The rest of this chapter is organized as follows. Section 4.2 introduces the best approximation framework for recovering the underlying trajectory given sparse and irregular longitudinal observations. Section 4.3 describes the SOAP method for estimating the optimal empirical basis functions and the corresponding coefficients. The asymptotic consistency results for the estimated functional empirical components(FECs) are provided in Section 4.4. Our proposed method is demonstrated in Section 4.5 by recovering the longitudinal CD4 percentage trajectories. In Section 4.6, we compare the finite sample performance of our method with the PACE method using simulation studies. Section 4.7 provides concluding remarks. An R package 'fSOAP' has been to implement the proposed method and is available at `https://github.com/YunlongNie/fSOAP`.

## 4.2 Functional Empirical Component Analysis

Consider $n$ independent realizations, $x_1(t), \dots, x_n(t)$, of an $L^2$ stochastic process $X(t) : t \in [0, T]$ at a sequence of random points on $[0, T]$ with measurement errors. That is, the observed data $y_{ij}, i = 1 \dots, n, j = 1 \dots, n_i$, is

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

where $\{\epsilon_{ij}\}$ are independent and identically distributed random errors with mean zero and variance $\sigma^2$. The number of measurements $n_i$ for each curve is random and small. The observed time points $t_{ij}$ can also be different for each curve. Using the Karhunen-Loève expansion (Fukunaga and Koontz, 1970), each $x_i(t)$ can be expressed as

$$x_i(t) = \mu(t) + \sum_{k=1}^{\infty} \alpha_{ik}\phi_k(t),$$

where $\mu(t) = E(X(t))$ is the mean function, and $\phi_k(t), k = 1, 2, \dots,$ are the eigenfunctions of the covariance function $C(s,t) = E[(X(s)-\mu(s))(X(t)-\mu(t))], t, s \in [0, T]$. We call $\phi_k(t)$ the functional principal components (FPCs) and $\alpha_{ik}$ is the corresponding FPC score. The above estimation procedure is called the functional principal component analysis (FPCA).

A main advantage of FPCA is that $x_i(t)$ is projected to orthogonal basis functions, which allows us to approximate $x_i(t)$ using the first $K$ leading FPCs:

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

There are many other basis functions on which $x_i(t)$ can be projected. However, the eigenfunctions of the covariance functions have been proved to be the optimal basis functions in the sense that they minimize the mean $L^2$ errors (see Tran (2008)). Formally speaking, for any fixed $K \in \{1, 2, \ldots\}$, the first $K$ leading FPCs minimize

$$\frac{1}{n}\Big(\sum_{i=1}^{n}\int\Big[x_i(t)-\mu(t)-\sum_{k=1}^{K}\langle x_i-\mu,\phi_k\rangle\phi_k(t)\Big]^2 dt\Big),$$

subject to $\langle \phi_k, \phi_l\rangle = \delta_{kl}$, where $\delta_{kl}$ is the Kronecker's delta. From the above criterion, we can see that the eigenfunctions $\phi_k(t), k = 1, \ldots, K$, are essentially the optimal empirical basis functions to the centered stochastic process $X(t) - \mu(t)$.

For the original stochastic process $X(t)$ without subtracting the mean function, the optimal empirical basis functions are the eigenfunctions of $K(s,t) = E[X(s)X(t)]$, as shown in Theorem 4.2.1 below. Note that although $K(s,t)$ is not a covariance function, it is a still Mercer kernel. By Mercer's theorem, there exists an orthonormal basis $\psi_m(t)$ such that $K(s,t)$ has the following representation:

$$K(s,t) = \sum_{m=1}^{\infty}\lambda_m\psi_m(s)\psi_m(t),$$

in which the eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq 0$ and the eigenfunctions satisfy $\langle \psi_m, \psi_\ell\rangle = \delta_{m\ell}$. Correspondingly, $x_i(t)$ can be represented as

$$x_i(t) = \sum_{m=1}^{\infty}\alpha_{im}\psi_m(t).$$

Now we will show that the empirical basis functions, $\psi_m(t), m = 1, \ldots, M$, optimal in the sense of minimizing the approximation error (4.1), are the eigenfunctions of the estimated $\widehat{K}(s,t) = \frac{1}{n}\sum_{i=1}^{n}[x_i(s)x_i(t)]$.

**Theorem 4.2.1.** For any given value of $M$, the optimal empirical basis functions $\psi_m(t), m = 1, \ldots, M$, which minimize

$$\frac{1}{n}\sum_{i=1}^{n}\Big(\int\Big[x_i(t)-\sum_{m=1}^{M}\alpha_{im}\psi_m(t)\Big]^2 dt\Big), \tag{4.1}$$

subject to $\langle \psi_m, \psi_\ell\rangle = \delta_{m\ell}$, are the first $M$ eigenfunctions of $\widehat{K}(s,t) = \frac{1}{n}\sum_{i=1}^{n}[x_i(s)x_i(t)]$ and $\alpha_{im} = \langle x_i, \psi_m\rangle$.

The detailed proof for Theorem 4.2.1 is given in the Appendix B. Theorem 4.2.1 not only shows that those eigenfunctions of $\widehat{K}(s,t)$ are the optimal empirical basis functions to approximate the original functional data, but also provides an alternative way to estimate these optimal empirical basis functions in the best approximation framework other than eigen-decomposing the uncentered sample covariance function $\widehat{K}(s,t)$. Note that estimating the sample covariance function may become challenging when the data are sparsely observed and the assembled time points with observations for all subjects are not sufficiently dense.

Moreover, this best approximation framework also allows for estimating the coefficients to the optimal empirical basis functions without inverting the sample covariance matrix. Furthermore, Theorem 4.2.1 shows that estimating the mean function $\mu(t)$ is not necessary if the goal is recovering or approximating the original trajectory. In practice, when the observed data are very sparsely observed and the assembled time points for all subjects are not sufficiently dense, it may be challenging to estimate the mean function $\mu(t)$. Alternatively, we can simply estimate those optimal empirical basis functions and represent each trajectory using the estimated optimal empirical basis functions.

In this chapter, the optimal empirical basis functions $\psi_m(t), m = 1, 2, \ldots,$ are called the functional empirical components (FECs), and $\alpha_{im}$ is the corresponding FEC score. Note that when the mean function of the stochastic process $X(t)$, $\mu(t) = E(X(t)) = 0$, the functional empirical components are equivalent to the functional principal components.

We propose the sparse orthonormal approximation (SOAP) method to estimate the first $M$ FECs $\psi_m(t), m = 1, \ldots, M$, by minimizing the observed loss function:

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}\left[y_{ij} - \sum_{m=1}^{M}\alpha_{im}\psi_m(t_{ij})\right]^2, \tag{4.2}$$

subject to $\langle\psi_m, \psi_\ell\rangle = \delta_{m\ell}$, where $m, \ell = 1, \ldots, M$. We solve the optimization problem (4.2) in a sequential manner. That is, we first obtain the first FEC. Then conditional on the estimated first FEC, we estimate the second FEC, and so on. When estimating each FEC, we estimate the $m$-th component $\psi_m$ and the corresponding FEC score $\boldsymbol{\alpha}_m = (\alpha_{1m}, \ldots, \alpha_{nm})^T$ in an iterative fashion. We first estimate $\boldsymbol{\alpha}_m$ based on the given FEC $\psi_m(t)$ and the observations $y_{ij}, i = 1, \ldots, n, j = 1, \ldots, n_i$. Then, given the estimated $\widehat{\boldsymbol{\alpha}}_m$, we obtain the corresponding FEC $\psi_m(t)$ by minimizing (4.2). In each iteration, the loss function (4.2) is guaranteed to decrease.

## 4.3 Sparse Orthonormal Approximation Method

We first describe our sparse orthonormal approximation (SOAP) method to estimate the first FEC in Section 4.3.1. Then our method is expanded to estimate the first $M$ FECs in Section 4.3.2.

### 4.3.1 Estimating the First FEC

Based on (4.2), the first FEC $\psi_1(t)$ is obtained by minimizing

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}\left[y_{ij} - \alpha_{i1}\psi_1(t_{ij})\right]^2, \tag{4.3}$$

subject to $||\psi_1(t)||^2 = 1$. We first express $\psi_1(t)$ as a linear combination of basis functions: $\psi_1(t) = \boldsymbol{\beta}_1^T \mathbf{b}(t)$, where $\mathbf{b}(t) = (b_1(t), \ldots, b_L(t))^T$ is a vector of basis functions, and $\boldsymbol{\beta}_1 = (\beta_{11}, \ldots, \beta_{1L})^T$ is the corresponding vector of coefficients. We propose to minimize (4.3) in an iterative fashion. That is, for a given $\psi_1(t)$, we find the corresponding $\alpha_{i1}$ which minimizes (4.3). Then given the value of $\alpha_{i1}$, we minimize (4.3) with respect to $\psi_1(t)$. In every iteration step, the value of the lost function (4.3) decreases. The detailed algorithm is outlined as follows:

Step I Set the initial value of $\psi_1(t)$ as $\psi_1^{(0)}(t)$, which satisfies $||\psi_1^{(0)}||^2 = 1$;

Step II Given the current value of $\psi_1^{(\ell)}(t), j = 0, 1, 2, \ldots$, we can obtain the value of $\boldsymbol{\alpha}_1^{(\ell)} = (\alpha_{11}, \ldots, \alpha_{n1})^T$ by minimizing

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}\left[y_{ij} - \alpha_{i1}\psi_1^{(\ell)}(t_{ij})\right]^2.$$

In fact, this is simply a least squares problem. The $i$th element of $\boldsymbol{\alpha}_1^{(\ell)}$ can be expressed as

$$\alpha_{i1} = (\boldsymbol{\psi}_{1i}^T \boldsymbol{\psi}_{1i})^{-1} \boldsymbol{\psi}_{1i}^T \mathbf{y}_i.$$

where $\boldsymbol{\psi}_{1i} = (\psi_1(t_{i1}), \ldots, \psi_1(t_{in_i}))^T$ is a $n_i \times 1$ vector and $\mathbf{y}_i = (y_{i1}, \ldots, y_{in_i})^T$.

Step III Given the current value of $\boldsymbol{\alpha}_1^{(\ell)}$, we update $\psi_1^{(\ell)}(t)$ to $\psi_1^{(\ell+1)}(t)$ by minimizing

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}[y_{ij} - \alpha_{i1}^{(\ell)}\psi_1(t_{ij})]^2,$$

subject to $||\psi_1||^2 = 1$.

We recast the above criterion into:

$$\begin{aligned}
&\sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}[y_{ij} - \alpha_{i1}^{(\ell)}\psi_1(t_{ij})]^2 \\
=&\sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}[y_{ij} - \alpha_{i1}^{(\ell)}\boldsymbol{\beta}_1^T\mathbf{b}(t_{ij})]^2 \\
=&\sum_{i=1}^{n}\sum_{j=1}^{n_i}[\frac{1}{\sqrt{n_i}}y_{ij} - \boldsymbol{\beta}_1^T\frac{1}{\sqrt{n_i}}\alpha_{i1}^{(\ell)}\mathbf{b}(t_{ij})]^2,
\end{aligned}$$

subject to $\boldsymbol{\beta}_1^T \mathbf{G} \boldsymbol{\beta}_1 = 1$, in which $\mathbf{G}$ is a $L \times L$ matrix with the $(i, j)$-th element $\langle b_i, b_j \rangle$. This is a constrained least squares problem. Fortunately, we can ignore the norm constrain and obtain the unconstrained least squares minimizer first and then scale it such that its norm is 1. More specifically, the solution can be written as $\boldsymbol{\beta}_1^{(\ell+1)} = \tilde{\boldsymbol{\beta}}_1^{(j+1)} / \sqrt{\{\tilde{\boldsymbol{\beta}}_1^{(j+1)}\}^T \mathbf{G} \tilde{\boldsymbol{\beta}}_1^{(j+1)}}$, in which $\tilde{\boldsymbol{\beta}}_1^{(j+1)} = (\mathbf{a}^{(\ell)T}\mathbf{a}^{(\ell)})^{-1}(\mathbf{a}^{(\ell)})^T\mathbf{y}_w$,

$\mathbf{y}_w = (\mathbf{y}_1^T/\sqrt{n_1}, \ldots, \mathbf{y}_n^T/\sqrt{n_n})^T$ and $\mathbf{a}^{(\ell)} = (\mathbf{a}_1^{(\ell)^T}, \ldots, \mathbf{a}_n^{(\ell)^T})^T$ is a $(\sum_{i=1}^n n_i) \times L$ matrix, in which $\mathbf{a}_i^{(\ell)}$ is a $n_i \times L$ matrix with $(p, q)$ elements being $\alpha_{i1}^{(\ell)}\psi_q(t_{ip})/\sqrt{n_i}$. It can be checked that the minimizer obtained from the least squares will satisfy the Karush-Kuhn-Tucker condition, thus it is the global minimizer of the loss function (4.3)

Step IV Repeat Step II and III until the algorithm converges.

### 4.3.2 Estimating the First and Second FECs

The first and second FECs are estimated by minimizing

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}[y_{ij} - \alpha_{i1}\psi_1(t_{ij}) - \alpha_{i2}\psi_2(t_{ij})]^2\,,$$

subject to $\langle\psi_m, \psi_\ell\rangle = \delta_{m\ell}, m, \ell \in \{1, 2\}$. We propose to use the following algorithm to simultaneously estimate $\psi_1(t)$ and $\psi_2(t)$.

Step I: Set an initial value of $\psi_1^{(0)}(t)$, which can be obtained using the algorithm described in the previous subsection.

Step II: Given the current value of $\psi_1^{(\ell)}(t)$, we apply the following iterative algorithm to obtain the estimates for $\psi_2^{(\ell)}(t)$ and $\boldsymbol{\alpha}_m^{(\ell)}, m = 1, 2$, by minimizing

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}[y_{ij} - \alpha_{i1}\psi_1^{(\ell)}(t_{ij}) - \alpha_{i2}\psi_2(t_{ij})]^2\,,$$

subject to $\langle\psi_m, \psi_\ell\rangle = \delta_{m\ell}, m, \ell \in \{1, 2\}$. We apply a similar procedure as described in the previous subsection to obtain the estimates for $\psi_2^{(\ell)}(t)$ and $\boldsymbol{\alpha}_m^{(\ell)}, m = 1, 2$, as follows:

(1) Set an initial value for $\psi_2(t)$, denoted as $\psi_2^0(t)$, which satisfies $||\psi_2^0||^2 = 1$ and $\langle\psi_2^0, \psi_1^{(\ell)}\rangle = 0$;

(2) Given the current value of $\psi_2^{(\ell)}(t)$, we obtain the estimate for $\boldsymbol{\alpha}_m^{(\ell)}, m = 1, 2$, by minimizing

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}[y_{ij} - \alpha_{i1}\psi_1^{(\ell)}(t_{ij}) - \alpha_{i2}\psi_2^{(\ell)}(t_{ij})]^2.$$

This is simply a least squares problem. For the $i$-th subject, the corresponding $\boldsymbol{\alpha}_i^{(\ell)} = (\alpha_{i1}^{(\ell)}, \alpha_{2i}^{(\ell)})^T$ is given as

$$\boldsymbol{\alpha}_i^{(\ell)} = (\boldsymbol{\psi}_i^T\boldsymbol{\psi}_i)^{-1}\boldsymbol{\psi}_i^T\mathbf{y}_i,$$

where $\boldsymbol{\psi}_i = (\boldsymbol{\psi}_{i1}^{(\ell)}, \boldsymbol{\psi}_{i2}^{(\ell)})$, $\boldsymbol{\psi}_{i1}^{(\ell)} = (\psi_1^{(\ell)}(t_{i1}), \ldots, \psi_1^{(\ell)}(t_{in_i}))^T$, $\boldsymbol{\psi}_{i2}^{(\ell)} = (\psi_2^{(\ell)}(t_{i1}), \ldots, \psi_2^{(\ell)}(t_{in_i}))^T$ and $\mathbf{y}_i = (y_{i1}, \ldots, y_{in_i})^T$.

(3) Given the value of $\boldsymbol{\alpha}_i^{(\ell)}$, update the value of $\psi_2^{(\ell+1)}(t)$ by minimizing

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}[y_{ij} - \alpha_{i1}^{(\ell)}\psi_1^{(\ell)}(t_{ij}) - \alpha_{2i}^{(\ell)}\psi_2(t_{ij})]^2$$

subject to $\langle \psi_2^{(\ell+1)}, \psi_1^{(\ell)} \rangle = 0$ and $||\psi_2^{(\ell+1)}||^2 = 1$. Because the norm of $\psi_2^{(\ell+1)}(t)$ will not affect the KKT conditions, we can first ignore the norm constraint and the minimization becomes a least square with equality-constraints problem. This problem can also be solved efficiently using the Least Squares with Equalities and Inequalities(LSEI) algorithm proposed by Lawson and Hanson (1974). Then, we normalize the resulting solution such that the norm of $\psi_2^{(\ell+1)}(t)$ is 1.

(4) Repeat step (2) and step (3) until the convergence reaches.

(5) In the end, we obtain the estimate $\psi_2^{(\ell)}(t)$ and $\boldsymbol{\alpha}_m^{(\ell)}, m = 1, 2$, for the given value of $\psi_1^{(\ell)}(t)$ .

Step III: Given the estimated value $\psi_2^{(\ell)}(t)$, we treat $\psi_1(t)$ as an unknown function and apply the same algorithm within Step II to obtain the estimate for $\psi_1^{(\ell+1)}(t)$ and $\boldsymbol{\alpha}_m^{(\ell+1)}, m = 1, 2$.

Step IV: Repeat Step II and III until the algorithm converges.

### 4.3.3 Estimating More FECs

Given the estimates for the first $M$ FECs, $\widehat{\psi}_i(t), i = 1, \ldots, M$, we can obtain the estimate for $\psi_{M+1}(t)$ and the corresponding $\boldsymbol{\alpha}_{M+1}$ using a similar strategy as described in Subsection 4.3.2. To be more specific, we iterate between $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \ldots, \boldsymbol{\alpha}_{M+1}$ and $\psi_{M+1}(t)$ by treating the first $M$ FECs fixed. After we obtain the estimate for $\psi_{M+1}$, we can further refine those estimates for the first $M$ FECs iteratively by treating each of them as unknown at each iteration. In this way, the loss function decreases in the loss function in every iteration. As well, the estimated FECs are always orthogonal to each other.

### 4.3.4 Smoothness Regulation

In order to control the smoothness of the estimated FECs $\psi_m(t), m = 1, \ldots, M$, we can add a roughness penalty in (4.2). That is, for any fixed $M$, we estimate $\psi_1(t), \ldots, \psi_M(t)$ by minimizing

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}\left[y_{ij} - \sum_{m=1}^{M}\alpha_{im}\psi_m(t_j)\right]^2 + \sum_{m=1}^{M}\gamma_m \int \left[\frac{d^2\psi_m(t)}{dt^2}\right]^2 dt, \tag{4.4}$$

subject to $\langle \psi_m, \psi_\ell \rangle = \delta_{m\ell}$, where $m, \ell = 1, \ldots, M$. The algorithm introduced in Subsection 3.1-3.3 can be modified accordingly. For instance, we can estimate the first FEC by modifying Step III in Subsection 3.1 as:

Step III (b) Given the current value of $\boldsymbol{\alpha}_1^{(\ell)}$, we update the estimate of $\psi_1^{(\ell)}(t)$ to $\psi_1^{(\ell+1)}(t)$ by minimizing

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}[y_{ij} - \alpha_{i1}^{(\ell)}\psi_1(t_{ij})]^2 + \gamma_1 \int \left[\frac{d^2\psi_1(t)}{dt^2}\right]^2 dt,$$

subject to $||\psi_1||^2 = 1$.

The above minimization is essentially a quadratically constrained quadratic program(QCQP) problem. We use the R package `Rsolnp` (Ghalanos and Theussl, 2015) based on the SOLNP algorithm proposed by Ye (1987) to numerically solve it. We will demonstrate the performance of this method in our simulation studies.

When estimating each FEC, there is only one tuning parameter involved, i.e., the smoothing parameter $\gamma_m$. The value of $\gamma_m$ controls the amount of smoothness imposed on the $m$-th FEC. We propose to select the tuning parameter based on the leave-one-curve-out cross validation strategy. To be more specific, we treat one curve's observations as the test data set and the data for all other curves as the training data set. For instance, when we estimate the first FEC $\psi_1(t)$, we can first obtain the estimate for the first FEC, $\widehat{\psi}_1^{(-i)}(t)$, using all the training data for any given value of $\gamma_1$, where we suppose to use the $i$-th curve as the test data set. Then, the score for the test curve can be calculated by minimizing

$$\sum_{j=1}^{n_i}(y_{ij} - \alpha_{i1}\widehat{\psi}_1^{(-i)}(t_{ij}))^2.$$

Then the prediction for $y_{ij}$ is $\hat{y}_{ij}^{(-i)} = \widehat{\alpha}_{i1}^{(-i)}\widehat{\psi}_1^{(-i)}(t_{ij})$. The prediction error for the $i$-th curve is

$$\frac{1}{n_i}\sum_{j}(\hat{y}_{ij}^{(-i)} - y_{ij})^2.$$

The cross validation error for $\gamma_1$ is given as

$$\mathrm{CV}(\gamma_1) = \sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}(\hat{y}_{ij}^{(-i)} - y_{ij})^2.$$

For the following FEC, we propose to select the smoothing parameter after treating the previous estimated FECs fixed.

### 4.3.5 Selecting the Number of FECs

We use the AIC criterion proposed by Li et al. (2013) to select the number of FECs:

$$\text{AIC}(M) = N \log(\sigma_M^2) + N + 2nM,$$

in which $M$ denotes the number of FECs, $n$ denotes the number of individual curves, and $N = \sum_{i=1}^{n} n_i$ is the total number of observations. We can estimate the noise variance, $\sigma_M^2$, by using the average square of the residuals. That is,

$$\hat{\sigma}_M^2 = \frac{1}{n}\sum_{i=1} \frac{1}{n_i}(\mathbf{y}_i - \widehat{\mathbf{y}}_{i,M})^T(\mathbf{y}_i - \widehat{\mathbf{y}}_{i,M}), \tag{4.5}$$

where $\widehat{\mathbf{y}}_{i,M} = (\hat{y}_{i1}, \ldots, \hat{y}_{in_i})^T$ represents the fitted $i$-th individual's observations when the number of FECs is selected to be $M$.

## 4.4 Theoretical Results

Theorem 2 shows that our first estimated FEC will asymptotically converge to the true FEC as the number of subjects increases. Similar results are shown in Theorem 3 for the rest estimated FECs.

Consider sparse observations of functional data $y_{ij} = x_i(t_{ij}) + \epsilon_{ij}$, where the observation times $t_{ij}, j = 1, \ldots, n_i$, for subject $i$ are uniformly drawn from $[0,1]$. Let the Mercer representation for the uncentered covariance function $K(s,t) = \mathtt{E}(X(s)X(t))$ of the stochastic process $X(t)$ be

$$K(s,t) = \sum_{m=1}^{\infty} \lambda_m \psi_m^0(s)\psi_m^0(t).$$

Assume $\sum_m \lambda_m < \infty$ and $\int_0^1 [\psi_m^0(t)]^4 dt < \infty$ for each $m = 1, 2, \cdots$.

**Theorem 4.4.1.** Recall the objective function

$$L_n(\boldsymbol{\alpha}, \psi) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{n_i}\sum_{j=1}^{n_i}[y_{ij} - \alpha_{i1}\psi_1(t_{ij})]^2 \tag{4.6}$$

where $\boldsymbol{\alpha}_1 = (\alpha_1 1, \ldots, \alpha_{n1}) \in \mathbb{R}^n$ and $\psi_1(t)$ is a function in $L^2(0,1)$ with constraint $\int_0^1 \psi^2(t)dt = 1$. Then the minimizer $\hat{\psi}_1(t)$ of $L_n$ converges to $\psi_1^0(t)$ in $L^2(0,1)$ almost surely as $n \to \infty$.

**Theorem 4.4.2.** The minimizers $\hat{\psi}_l(t)$, $l = 1, \cdots, M$, of the loss function

$$L_n(\{\boldsymbol{\alpha}_l, \psi_l\}_{l=1}^M) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{n_i}\sum_{j=1}^{n_i}[y_{ij} - \sum_{l=1}^{M} \alpha_{il}\psi_l(t_{ij})]^2 \tag{4.7}$$

converges to $\psi_l^0(t)$, $l = 1, \cdots, M$, in $L^2[0,1]$ almost surely as $n \to \infty$.

The proofs for Theorems 4.4.1 and 4.4.2 are available in the Appendix B. The following lemmas are used to prove the above theorem.

**Lemma 4.4.1.** Let $m_i, i = 1, 2, \ldots,$ be independent positive random variables with mean 1 and $\sum_{i=1}^{\infty} \mathtt{E}(m_i - 1)^2/i^2 < \infty$. For any sequence of positive numbers, $a_i$, such that $\sum_{i=1}^{\infty} \mathtt{E}(m_i - 1)^2 a_i^2/i^2 < \infty$, we have

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} a_i = \lim_{n\to\infty} \frac{1}{\sum_{i=1}^{n} m_i} \sum_{i=1}^{n} m_i a_i \quad a.s.$$

**Lemma 4.4.2.** Let $m_{ij}, i = 1, 2, \ldots, j = 1, 2, \ldots,$ be positive random variables. For each $j = 1, 2, \ldots,$ $m_{ij}, i = 1, 2, \ldots,$ are independently and identically distributed with mean 1 and finite variance. Then for any infinite matrix $A = [A_{ij}]$, with $\lambda_j = \lim_n \frac{1}{n} \sum_{i=1}^{n} a_{ij}^2$ exists for each $j$, as $n \to \infty$,

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{\infty} a_{ij}^2 = \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{\infty} a_{ij}^2 m_{ij} = \sum_{j=1}^{\infty} \lambda_j.$$

**Lemma 4.4.3.** For an $n \times p$ matrix $A$, the r-rank approximation of $A$ under the Frobenius norm of matrices is $\tilde{A} = \sum_{i=1}^{r} \boldsymbol{\alpha}_i \otimes \boldsymbol{\beta}_i$ where $r \leq \min(n, p)$, and $\boldsymbol{\alpha}_i$, $\boldsymbol{\beta}_i$ are the eigenvectors of $AA^T$ and $A^T A$, respectively.

## 4.5 Application: Longitudinal CD4 Percentages

We demonstrate our proposed method by analyzing the longitudinal CD4 counts dataset. The CD4 percentage, which is defined as CD4 counts divided by the total number of lymphocytes, is a commonly used marker to describe the health status of HIV infected persons. The dataset considered here is from the Multi-center AIDS Cohort Study, which includes repeated measurements of CD4 percentages for 283 homosexual men who became HIV positive between 1984 and 1991. All subjects were scheduled to be measured at semi-annual visits. The trajectories of 10 randomly selected subjects are shown in Figure 4.1. It shows that the data are sparse with unequal numbers of repeated measurements and different visit times for individual subjects, because many of them missed scheduled visits and the HIV infections could occur randomly during the study. For all 283 subjects, the number of observations per subject ranged between 1 and 14, with a median of 6 measurements.

Table 4.1: The values of AIC defined in (4.5) for various number of FECs.

| # FECs | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| AIC | 8493.44 | 7632.86 | 7626.01 | 7720.19 | 7913.83 | 8059.46 |

The objective of our analysis is to recover individual longitudinal trajectories from the sparse and irregular observations. The smoothing parameters are selected from $\{0, 10^2, 10^4, 10^8\}$

using the leave-one-curve-out cross-validation and the selected smoothing parameters for the first 5 estimated eigenfunctions are $10^4, 10^2, 10^4, 10^2$ and $10^4$, respectively. Table 4.1 displays the values of AIC defined in (4.5) varying with the number of FECs. It shows that AIC is minimized when the number of FECs is 3.

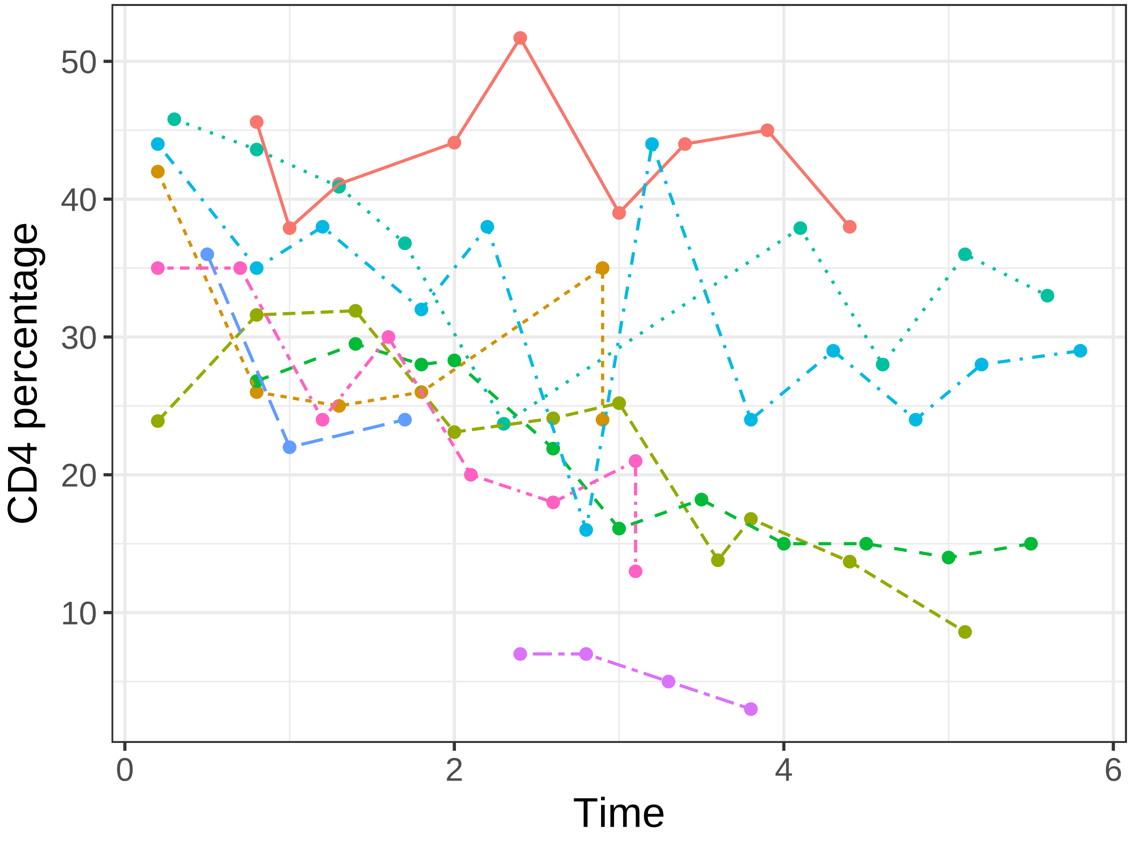

Figure 4.1: The longitudinal CD4 percentage for 10 randomly selected subjects. Each curve represents the measurements for one single subject.

Figure 4.2 shows the estimated three FECs and the estimated mean function. The first estimated FEC, $\hat{\psi}_1(t)$, is decreasing and positive over the whole time interval. The first FEC score can be interpreted as the weighted average of the longitudinal trajectory across time. The second estimated FEC, $\hat{\psi}_2(t)$, changes its sign at time 3. The second FEC score can be interpreted as the change of the longitudinal trajectory between $[0, 3]$ and $[3, 6]$. Similarly, the third estimated FEC, $\hat{\psi}_3(t)$, is positive $[1.6, 4.3]$ and negative elsewhere. So the third FEC score represents the change of the longitudinal trajectory between $[1.6, 4.3]$ and the other periods. The mean function is obtained by taking the average of all the individual predicted trajectories, which shows an overall decreasing trend across individuals.

Figure 4.3 shows the predicted individual trajectories for 4 different individuals with the various number of observations. It shows that all the estimated CD4 trajectories fit

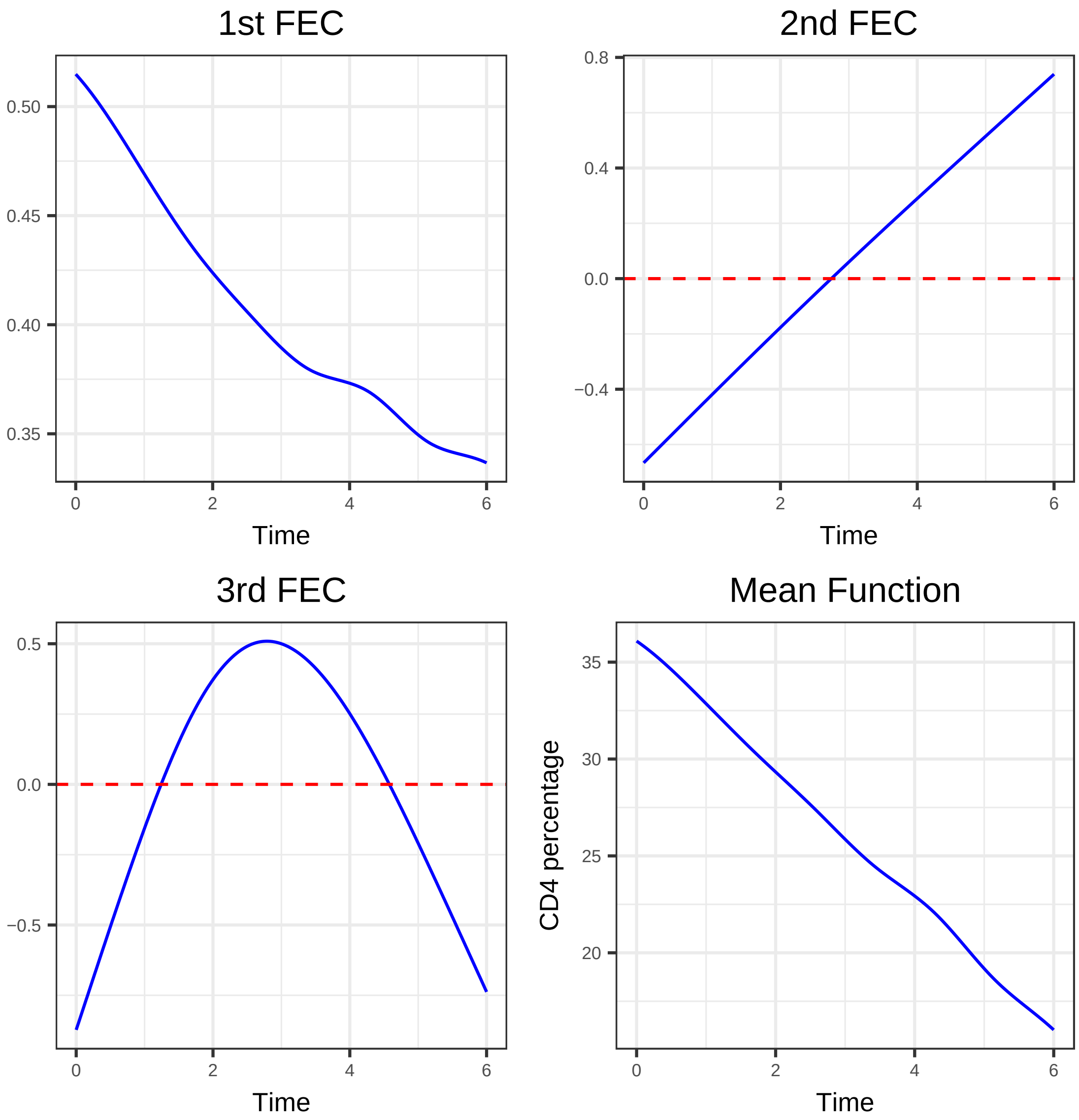

Figure 4.2: The estimated three functional empirical components (FECs) along with the estimated mean function for the CD4 data.

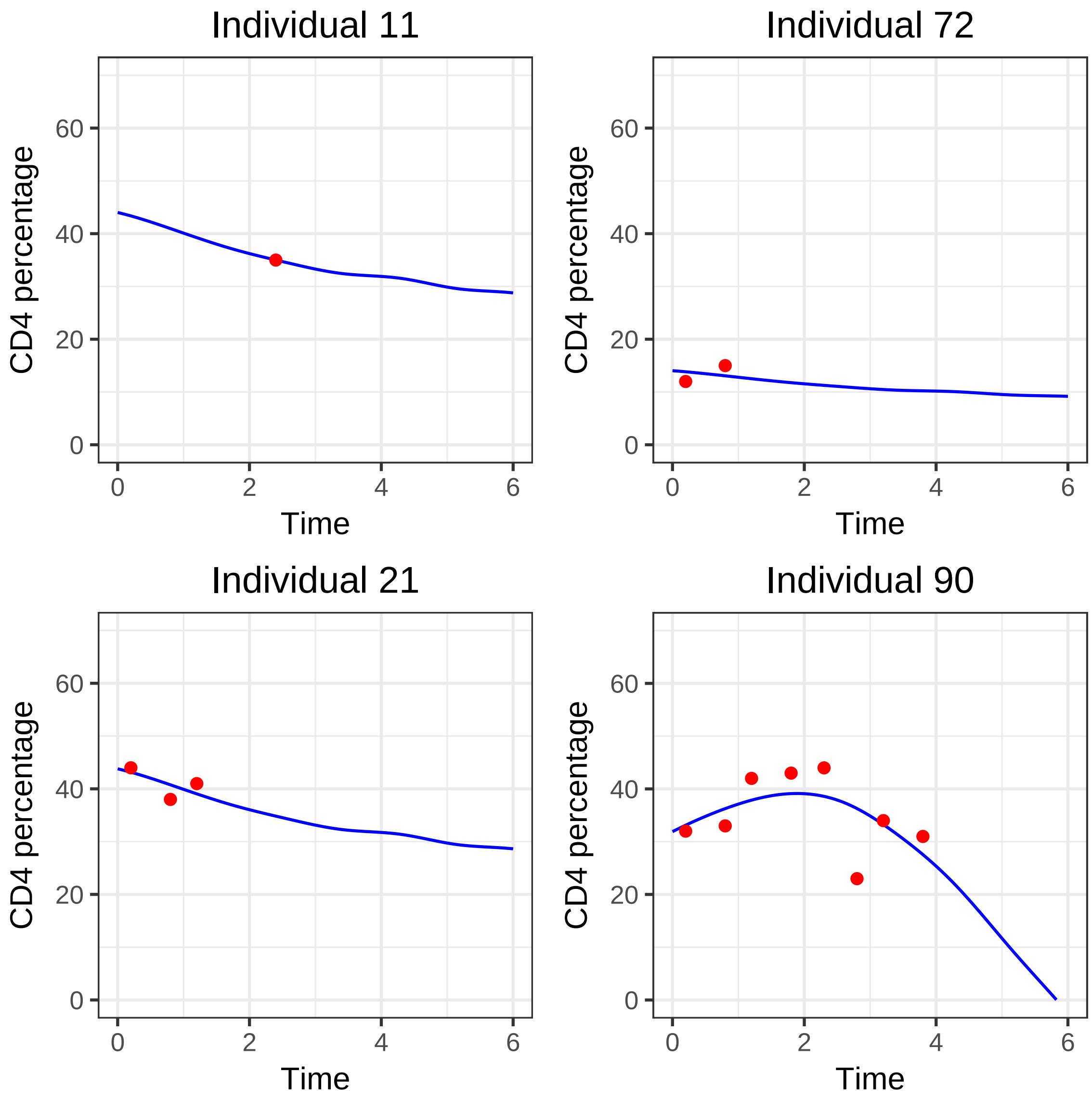

Figure 4.3: The estimated individual trajectories using the SOAP method (solid line) and the corresponding observations (dots) for individual 11, 21, 72 and 90.

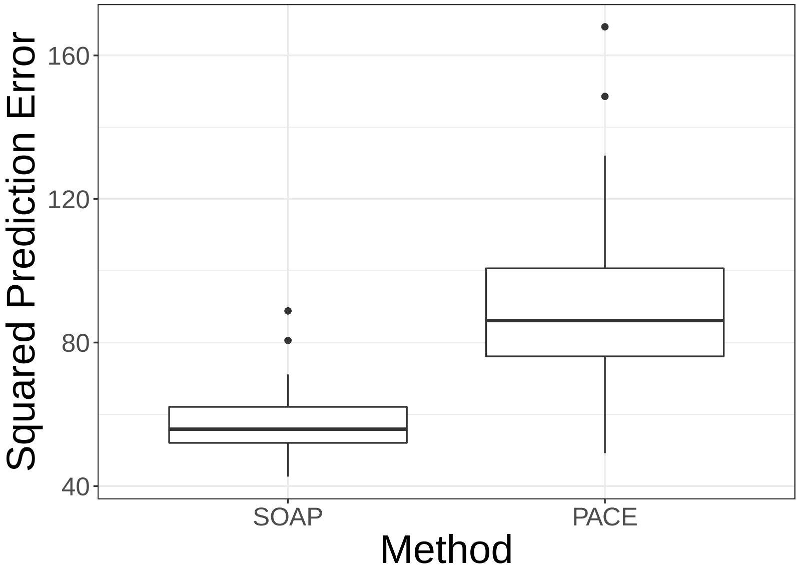

Figure 4.4: Boxplots of the mean square prediction errors for the last observation in the CD4 dataset using the SOAP method and the PACE method in 100 random data-splitting repetitions.

the observations well. An estimated individual trajectory generally displays the overall decreasing trend when the number of observations is small. On the other hand, when there are enough observations for individuals, such as individual 90 shown in Figure 4.3, the estimated individual trajectory is able to capture the individual trend.

To compare the SOAP method with the PACE method (Yao et al., 2005a) with respect to recovering the underlying trajectories, we use the following procedure. First, we randomly select the data of half subjects as the training data set and treat the other half data as the test data set. We estimate the FECs using the training dataset. Next, for each subject in the test data set, we treat the last observation as unknown and predict it based on the previous observations. In the end, we compare the predicted value with the observed value and obtain the mean square prediction error (MSPE) for all individuals in the test data set. We repeat this procedure 100 times. Figure 4.4 are the boxplots of the MSPEs for two methods. Figure 4.4 shows that the SOAP method outperforms the PACE method in predicting the future individual trajectories. For instance, the median of MSPEs is 55.87 for the SOAP method, which is 35% smaller than the PACE method. The 25% and 75% quantiles of MSPEs are 52.05 and 62.08 for our method, which are also 31% and 38% smaller than the PACE method, respectively.

## 4.6 Simulations

To evaluate the performance of our proposed method, we conduct one simulation study in comparison with the PACE method. In order to make our proposed method and the PACE method comparable, we simulate the curves $X_i(t)$ such that $E(X_i(t)) = 0$. Then in this simulation setting, the functional principal components (FPCs) in PACE are equivalent to our proposed functional empirical components (FECs). Therefore, for the rest of this section, we unify both of them as eigenfunctions.

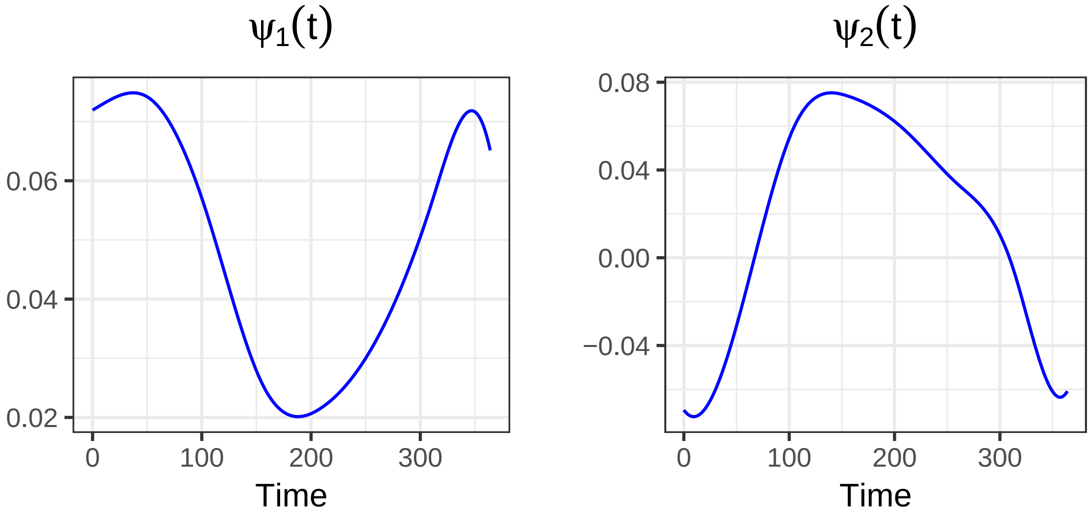

Figure 4.5: The true eigenfunctions used to generate the true underlying individual trajectories. We obtain these two functional empirical components by conducting conventional FPCA on the Canadian temperature Data (Ramsay and Silverman, 2002).

The underlying true trajectories are simulated as $X_i(t) = \alpha_{i1}\psi_1(t) + \alpha_{i2}\psi_2(t), i = 1, \ldots, n$, where the true eigenfunctions, $\psi_1(t)$ and $\psi_2(t)$ are shown in Figure 4.5, satisfying $\langle \psi_i, \psi_j \rangle = \delta_{ij}, i, j = 1, 2$. The corresponding scores $\alpha_{i1}$ and $\alpha_{i2}$ are generated in both Gaussian and non-Gaussian distributions. For the Gaussian scenario, the scores are generated from two independent Gaussian distributions. That is, $\alpha_{i1} \overset{i.i.d}{\sim} N(0, 30)$ and $\alpha_{i2} \overset{i.i.d}{\sim} N(0, 10)$. For the non-Gaussian scenario, the scores are first generated from two independent gamma distributions and then are centered by subtracting the sample mean. That is, $\alpha_{i1} = \alpha'_{i1} - \bar{\alpha}'_{i1}$, where $\alpha'_{i1} \overset{i.i.d}{\sim} \text{Gamma}(1, 0.03)$ and $\alpha_{i2} = \alpha'_{i2} - \bar{\alpha}'_{i2}$, where $\alpha'_{i2} \overset{i.i.d}{\sim} \text{Gamma}(1, 0.1)$. We choose the parameters of these two gamma distributions such that the standard derivations are roughly the same as in the Gaussian scenario. The corresponding observed data for each trajectory are generated as $y_{ij} = X_i(t_{ij}) + \epsilon_{ij}$, in which $\epsilon_{ij} \sim N(0, \sigma^2)$. To achieve the sparseness, the number of time points, $n_i$, for each trajectory is chosen randomly from a discrete uniform distribution on $\{1, 2, 3, 4, 5\}$ and the corresponding time points $t_{ij}, j = 1, \ldots, n_i$, are uniformly generated in the entire time domain $[0, T]$.

To evaluate the performance of the SOAP method, we generate 300 training samples and 300 test samples in each simulation replication. We first use our proposed method to estimate the eigenfunctions using the training dataset, and then predict the test samples' trajectories. The PACE method is also applied to estimate the eigenfunctions from the training data and predict the trajectories for the test samples. These two methods are compared by defining the integrated mean prediction error (IMPE) for the 300 test samples as:

$$\text{IMPE} = \frac{1}{300}\sum_{i=1}^{300}\int [\widehat{x}_i(t) - x_i(t)]^2 dt, \tag{4.8}$$

in which $x_i(t)$ represents the true $i$-th trajectory in the test set and $\widehat{x}_i(t)$ is the corresponding predicted trajectory. We repeat the above procedure for 100 repetitions.

Table 4.2: The summary results for predicting the individual trajectory using the SOAP method and the PACE method for 100 simulation replicates. The table shows the means, standard derivations (SDs), medians, minimums and maximums for the integrated mean prediction errors in (4.8) when the true FPC scores are generated from the Gaussian distribution and non-Gaussian distribution.

| | **Gaussian** | | **Non-Gaussian** | |
|---|---|---|---|---|
| | SOAP | PACE | SOAP | PACE |
| Mean | 159.38 | $1.02 \times 10^5$ | 164.46 | 981.36 |
| SD | 32.45 | $1.18 \times 10^6$ | 53.65 | $4.51{\times}10^4$ |
| Median | 154.16 | 151.83 | 151.59 | 290.30 |
| Minimum | 98.42 | 80.87 | 75.81 | 145.45 |
| Maximum | 283.32 | $1.39 \times 10^7$ | 521.61 | $5.34{\times}10^5$ |

The results are shown in Table 4.2. First of all, we find that the performance of PACE is quite unstable when the true FPC scores are generated from the Gaussian distribution and non-Gaussian distribution in comparison with our proposed method. For instance, the maximum IMPE of PACE goes up to $1.3 \times 10^7$. There are 11 in the Gaussian scenario and 19 in the non-Gaussian scenario out of 100 repetitions that IMPE of PACE is greater than 600. In contrast, all IMPEs from SOAP are less than 600.

We notice that PACE produces poorly predicted trajectories when the time points with observations are relatively close to each other. The reason is that estimating the individual scores involves computing the inverse of the sample covariance matrix, and the inverse of this matrix can be quite unstable when the time points with observations are close. Figure 4.6 shows an example from one simulation run with 5 observations in this scenario. Note that those 3 observations in the middle are relatively close in the time domain to each other in comparison with the last observation. As we can see in the left panel in Figure 4.6, the predicted trajectory produced by PACE overfits the observed data, while the predicted trajectory from SOAP is quite close to the true underlying trajectory.

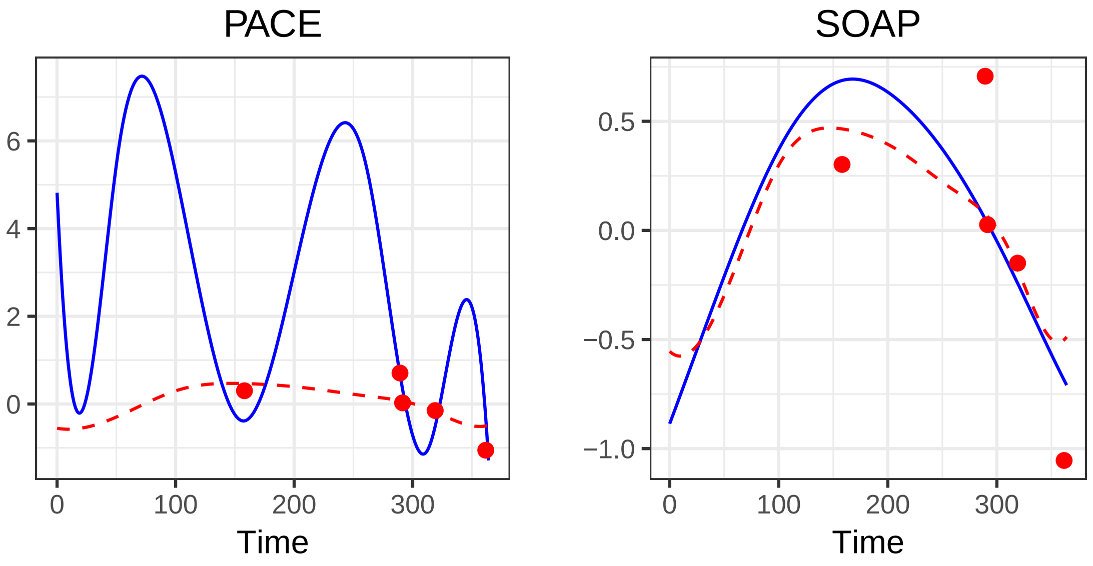

Figure 4.6: The estimated individual trajectory (solid line) using the PACE method (left panel) and the SOAP method (right panel) compared with the true trajectory (dashed line). The dots represent the observations for this curve.

Table 4.3: The summary results for estimating the underlying eigenfunctions using the SOAP method and the PACE method for 100 simulation replicates. The table shows the means, standard derivations(SDs), medians, minimums and maximums for the integrated mean square errors (IMSEs) defined in (4.9) when the true FPC scores are generated from the Gaussian distribution and non-Gaussian distribution.

| | | **Gaussian** | | **Non-Gaussian** | |
|---|---|---|---|---|---|
| | | IMSE($\hat{\psi}_1$) | IMSE($\hat{\psi}_2$) | IMSE($\hat{\psi}_1$) | IMSE($\hat{\psi}_2$) |
| **Mean** | SOAP | 3.20 | 32.05 | 3.56 | 32.90 |
| | PACE | 19.42 | 566.89 | 65.35 | 1012.42 |
| **SD** | SOAP | 1.89 | 3.00 | 2.61 | 3.86 |
| | PACE | 12.20 | 451.37 | 56.31 | 551.54 |
| **Median** | SOAP | 2.75 | 31.56 | 2.72 | 33.02 |
| | PACE | 15.42 | 406.00 | 46.37 | 961.76 |
| **Minimum** | SOAP | 0.47 | 25.81 | 0.45 | 23.59 |
| | PACE | 4.48 | 76.10 | 7.14 | 117.05 |
| **Maximum** | SOAP | 9.59 | 40.33 | 14.40 | 44.95 |
| | PACE | 81.98 | 1803.95 | 356.77 | 2157.66 |

Besides recovering the individual trend, we also compare the estimated eigenfunctions with the true eigenfunctions using the following integrated mean square error (IMSE):

$$\text{IMSE}(\hat{\psi}_i) = \int [\psi_i(t) - \widehat{\psi}_i(t)]^2 dt, i = 1, 2. \tag{4.9}$$

The results are summarized in Table 4.3. First, the estimated eigenfunctions using the SOAP method are much closer to the true underlying eigenfunctions than those estimated with the PACE method under both simulation settings. For instance, the mean IMSE($\hat{\psi}_1$) from the PACE method is 4 times larger than the SOAP method and IMSE($\hat{\psi}_2$) from the PACE method is almost 18 times larger than the SOAP method. In addition, the performance of the SOAP method is not sensitive to the distribution of the underlying scores, but the PACE method's performance significantly drops from the Gaussian to the non-Gaussian scenario. For instance, the mean IMSE($\hat{\psi}_1$) increases from 19.42 (Gaussian) to 65.34 (non-Gaussian). Finally, we notice that the performance of the SOAP method is generally more stable than the PACE method, which is shown by comparing the standard derivations of the IMSEs. For example, the standard deviation of IMSE($\hat{\psi}_2$) is 551.54 using the PACE method in comparison with 3.86 using the SOAP method.

## 4.7 Summary

In this chapter, we propose a novel SOAP method for predicting the underlying individual trajectories as well as the major variation patterns from sparse and irregularly longitudinal observations. The SOAP method directly estimates the empirical functional components from the best approximation perspective. This perspective is different from most conventional methods, such as PACE, which first estimates the de-meaned covariance function from the data and then eigen-decompose the resulting covariance function to obtain the estimated FPCs. This new best approximation perspective enables the SOAP method to recover the individual trajectories without estimating the mean and covariance functions and without requiring that the underlying FPC scores be Gaussian distributed.

We demonstrate the SOAP method by analyzing a CD4 dataset, in which the longitudinal measurements for each individual are sparsely and irregularly observed. Our SOAP method is able to recover the individual CD4 trajectories and explore the major variational sources across all subjects. We also compare the performance of the SOAP method and the PACE method in prediction by treating the last observation of each individual as unknown and find that the SOAP method produces better predictions compared to the PACE method.

Furthermore, we evaluate the performance of the SOAP method and the PACE method in a simulation study. We notice that the PACE method can be numerically unstable when the data are observed in close time points. Generally speaking, the SOAP method outperforms the PACE method in both predicting the individual trajectory and recovering the optimal empirical basis functions.

# Chapter 5

# Estimating Derivatives from Sparse and Irregularly Longitudinal Data

## 5.1 Introduction

In this chapter, we consider the problem of estimating derivatives of a sample of individual trajectories when the data are sparsely observed and contaminated with noise.

Conventional methods estimate the derivative of an individual trajectory by first estimating the individual trajectory directly using some nonparametric regression methods such as kernel smoothing(Gasser and Mülller, 1984), local polynomial regression(Fan and Gijbels, 1995b) and smoothing splines (de Boor, 2001). Then the derivative of the estimated individual trajectory is obtained as the estimated derivative function. However, these method generally requires the data to be densely observed, otherwise the estimated individual trajectory would have large errors and the estimated derivative would have even larger errors.

When the data are sparsely and irregularly observed, Liu and Müller (2009) proposed to first estimate the FPCs of the individual trajectories and then use the derivatives of the functional principal components (FPCs) as the optimal empirical basis functions to represent the derivatives of the individual trajectories. However, the derivatives of the FPCs for individual trajectories are not the FPCs for the derivative functions. Therefore, they are not the optimal empirical basis functions to represent the derivative functions. To overcome this drawback, Dai et al. (2017) proposed the derivative principal component analysis (DFCA) method to estimate the functional principal components of the derivative functions, which are the optimal empirical basis functions to represent the derivative functions. They showed that the covariance of the derivative functions was equivalent to the derivative of the covariance of the original functions. Therefore, Dai et al. (2017) proposed to estimate the covariance of the derivative functions by taking the derivative of the estimated covariance of the original function and then to eigen-decompose the estimated covariance function of the derivative functions to obtain the FPCs of the derivative functions. Their

DFCA method outperformed the method proposed in Liu and Müller (2009) shown in their simulation study.

In this chapter, we propose a new method to directly estimate the optimal empirical basis functions for the derivative functions in the best approximation framework. This method is called the derivative functional empirical component analysis (DeFECA). The novelty of our method is three-fold. First, the estimated empirical basis functions forms the most parsimonious or optimal representation of the derivative functions. Second, our method does not requires the coefficients to the empirical basis functions to be Gaussian distributed. Our method is still applicable when this assumption is invalid or difficult to verify. Third, our method does not require estimating the mean and covariance functions, which may be challenging when the assembled time points with observations for all subjects are not sufficiently dense. Last but not the least, simulation studies shows that our method is numerically stable in comparison to existing methods because our method avoids the inverse of the covariance matrix.

The rest of this chapter is organized as follows. The proposed DeFECA framework is introduced in Section 5.2. An efficient estimation method is given in Section 5.3. In Section 5.4, the proposed method is demonstrated by exploring the dynamics of body fat percentage of 162 girls before and after the first menarche. A carefully-designed simulation is conducted to evaluate the finite sample performance of our proposed method in comparison with other alternative methods in Section 5.5. Section 5.6 provides concluding remarks.

## 5.2 Derivative Functional Empirical Component Analysis

This chapter considers an $L^2$ stochastic process $X(t)$, where we assume $t \in [0, 1]$ without loss of generality. Let $x_i(t)$ denote a random sample of $X(t)$, $i = 1 \ldots, n$. The data $y_{ij}, j = 1 \ldots, n_i$, are observation of $x_i(t)$ at the point $t_{ij}$ with the measurement error $\epsilon_{ij}$. In other words,

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

where $\{\epsilon_{ij}\}$ are i.i.d. with zero mean and $\sigma^2$ variance.

We assume that the derivate, $X'(t)$, is a smooth and square integrable random process. By Mercer's theorem, we have the following representation for the uncentered covariance function $C(s, t) = E(X'(s)X'(t))$:

$$C(s, t) = \sum_{k=1}^{\infty} \lambda_k \phi_k(t)\phi_k(s),$$

in which $\lambda_k$ is the $k$-th positive eigenvalue with the decreasing order $\lambda_1 \geq \lambda_2 \geq \ldots \geq 0$, and $\phi_k(t)$ is the corresponding $k$-th eigenfunctions. Let $x_i'(t)$ denote the derivative function for $x_i(t)$. We can express the derivative function as

$$x_i'(t) = \sum_{k=1}^{\infty} \alpha_{ik}\phi_k(t),$$

in which $\alpha_{ik}$ is the $k$-th coefficient to the $k$th eigenfunction $\phi_k(t)$.

We call $\phi_k(t)$ as the derivative functional empirical component (DeFEC) to distinguish them from the conventional functional principal components (FPCs), which are the eigenfunctions of the covariance function $E[\{X(s) - E(X(s))\}\{X(t) - E(X(t))\}]$ of the original functional data $X(t)$. The coefficient $\alpha_{ik}$ is called the $k$-th DeFEC score of the $i$-th curve.

In practice, one can approximate the derivative function $x_i'(t)$ by using the first $K$ leading DeFECs:

$$x_i'(t) \approx \sum_{k=1}^{K} \alpha_{ik}\phi_k(t).$$

Consequently, the function $x_i(t)$ can be expressed as

$$x_i(t) = x_{i0} + \int_0^t x_i'(s)ds = x_{i0} + \sum_{k=1}^{K} \alpha_{ik} \int_0^t \phi_k(s)ds, \tag{5.1}$$

where $x_{i0}$ represent the value of the $i$-th curve at the starting point 0. Furthermore, the observations can be expressed as

$$y_{ij} = x_{i0} + \sum_{k}^{K} \alpha_{ik} \int_0^{t_{ij}} \phi_k(s)ds + \epsilon_{ij}. \tag{5.2}$$

We propose to estimate the first $K$ DeFECs $\phi_k(t), k = 1, \ldots, K$, in the best approximation framework by minimizing the following criterion:

$$\sum_{i=1}^{n} \frac{1}{n_i} \sum_{j=1}^{n_i} \left\{ y_{ij} - \left( x_{i0} + \sum_{k=1}^{K} \alpha_{ik} \int_0^{t_{ij}} \phi_k(s)ds \right) \right\}^2, \tag{5.3}$$

with constraints that $\phi_k(t), k = 1, \ldots, K$, satisfying $\langle \phi_k, \phi_j \rangle = 1$ if $k = j$ and $\langle \phi_k, \phi_j \rangle = 0$ if $k \neq j$.

## 5.3 Estimation Method

In this section, we first describe the method to estimate the first DeFEC $\phi_1(t)$ and the associated score $\boldsymbol{\alpha}_1$ in Section 5.3.1 and then extend the method to estimate the first $K$ FDCs, $\phi_k(t), k = 1, \ldots, K$, in Section 5.3.2 and Section 5.3.3.

### 5.3.1 Estimating the first DeFEC

The first DeFEC $\phi_1(t)$ is expressed with a linear combination of basis functions: $\phi_1(t) = \boldsymbol{\beta}_1^T \mathbf{b}_1(t)$, where $\boldsymbol{\beta}_1 = (\beta_{11}, \dots, \beta_{1J})^T$ is the vector of basis coefficients and $\mathbf{b}(t) = (b_1(t), \dots, b_J(t))^T$ is the corresponding vector of basis functions. When we focus on estimating the first DeFEC, the loss function in (5.3) will reduce to

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}\left[y_{ij} - x_{i0} - \alpha_{i1}\mathbf{h}_{ij}^T\boldsymbol{\beta}_1\right]^2. \tag{5.4}$$

subject to $||\phi_1||^2 = 1$, where $\mathbf{h}_{ij} = \int_0^{t_{ij}} \mathbf{b}(s)ds$.

We propose to minimize (5.4) in an iterative fashion. For a given $\boldsymbol{\beta}_1$, we estimate $\alpha_{i1}$ and $x_{i0}$ by minimizing the loss function (5.4) with respect to $\alpha_{i1}$ and $x_{i0}$. Then, given the estimated $\alpha_{i1}$ and $x_{i0}$, we estimate $\boldsymbol{\beta}_1$ by minimizing the loss function (5.4) with respect to $\boldsymbol{\beta}_1$. The value of the loss function (5.4) decreases in each iteration. The detailed algorithm is outlined as follows:

Step I: Set the initial value of $\boldsymbol{\beta}_1$ as $\boldsymbol{\beta}_1^{(0)}$ such that the corresponding DeFEC satisfying $||\phi_1^{(0)}||^2 = 1$;

Step II: Given the current value of $\boldsymbol{\beta}_1^{(\ell)}$, $\ell = 0, 1, 2, \dots$, we estimate $x_{i0}^{(\ell)}$ and $\alpha_{1i}^{(\ell)}$ by minimizing

$$\sum_{j=1}^{n_i}\left(y_{ij} - x_{i0} - \alpha_{1i}B_{i1j}^{(\ell)}\right)^2,$$

in which $B_{i1j}^{(\ell)} = \mathbf{h}_{ij}^T\boldsymbol{\beta}_1^{(\ell)}$. This is simply a least square problem. We obtain the estimated intercept $x_{i0}^{(\ell)}$ and the estimated slope $\alpha_{1i}^{(\ell)}$.

Step III: Given the estimates $\alpha_{1i}^{(\ell)}$ and $x_{i0}^{(\ell)}$, we update the estimate of $\boldsymbol{\beta}_1$ to $\boldsymbol{\beta}_1^{(\ell+1)}$ by mimizing

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}\left(y_{ij} - x_{i0}^{(\ell)} - \alpha_{i1}^{(\ell)}\mathbf{h}_{ij}^T\boldsymbol{\beta}_1\right)^2, \tag{5.5}$$

subject to $||\phi_1^{(0)}||^2 = \boldsymbol{\beta}_1^T\mathbf{G}\boldsymbol{\beta}_1 = 1$, in which $\mathbf{G}$ is a $J \times J$ matrix with the $(i, j)-$th element being $\int_0^1 b_i(t)b_j(t)dt$. This is a constrained least squares problem. Fortunately, we can ignore the norm constraint and first obtain the unconstrained least squares minimizer and then scale the estimated $\phi_1(t)$ such that its norm is 1. Let $\tilde{\boldsymbol{\beta}}_1^{(\ell+1)}$ be the minimizer of the loss function (5.5) without the norm constraint. It is expressed as $\tilde{\boldsymbol{\beta}}_1^{'(\ell+1)} = \{(\mathbf{a}^{(\ell)})^T\mathbf{a}^{(\ell)}\}^{-1}(\mathbf{a}^{(\ell)})^T\mathbf{y}^{(\ell)}$, in which $\mathbf{y}^{(\ell)} = ((\mathbf{y}_1^{(\ell)})^T/\sqrt{n_1}, \dots, (\mathbf{y}_n^{(\ell)})^T/\sqrt{n_n})^T$, $\mathbf{y}_i^{(\ell)} = (y_{i1} - x_{i0}^{(\ell)}, \dots, y_{in_i} - x_{n_i0}^{(\ell)})^T$, $\mathbf{a}^{(\ell)} = ((\mathbf{a}_1^{(\ell)})^T, \dots, (\mathbf{a}_n^{(\ell)})^T)^T$, and an $n_i \times J$ matrix $\mathbf{a}_i^{(\ell)}$ has the $(p, q)$-th element as $\alpha_{i1}^{(\ell)}\int_0^{t_{ip}} b_q(s)ds/\sqrt{n_i}$. We then scale the estimate as $\boldsymbol{\beta}_1^{(\ell+1)} =$

$\tilde{\boldsymbol\beta}_1^{(\ell+1)}/\sqrt{(\tilde{\boldsymbol\beta}_1^{(\ell+1)})^T\mathbf{G}\tilde{\boldsymbol\beta}_1^{(\ell+1)}}$. We can easily verify the Karush-Kuhn-Tucker (KKT) condition to show that $\boldsymbol\beta_1^{(\ell+1)}$ is the minimizer for the given value of $\alpha_{1i}^{(\ell)}$ and $x_{i0}^{(\ell)}, i = 1, \dots, n$.

Step IV: Repeat Step II and III until the algorithm converges.

### 5.3.2 Estimating the first and the second DeFECs

The first two DeFECs are estimated by minimizing

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}\left(y_{ij} - x_{i0} - \alpha_{i1}\int_0^{t_{ij}}\phi_1(s)ds - \alpha_{i2}\int_0^{t_{ij}}\phi_2(s)ds\right)^2.$$

subject to $\int_0^1 \phi_i^2(t)dt = 1, i, j = 1, 2$, and $\int_0^1 \phi_1(t)\phi_2(t)dt = 0$. The first two DeFECs are both expressed as a linear combination of basis functions $\phi_i(t) = \boldsymbol\beta_i^T\mathbf{b}(t)$, $i = 1, 2$. We propose to use the following algorithm to simultaneously estimate $\phi_1(t)$ and $\phi_2(t)$:

Step I: Set an initial value $\boldsymbol\beta_1^{(0)}$. We can apply the algorithm described in the previous section to obtained this initial estimate.

Step II: Given the current value of $\boldsymbol\beta_1^{(\ell)}, \ell = 0, 1, 2, \dots$, we estimate $\boldsymbol\beta_2^{(\ell)}, \boldsymbol\alpha_1^{(\ell)} = (\alpha_{11}^{(\ell)}, \dots, \alpha_{n1}^{(\ell)})^T, \boldsymbol\alpha_2^{(\ell)} = (\alpha_{12}^{(\ell)}, \dots, \alpha_{n2}^{(\ell)})^T$ and $\mathbf{x}_0^{(\ell)} = (x_{10}^{(\ell)}, \dots, x_{n0}^{(\ell)})^T$ by minimizing:

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}\left(y_{ij} - x_{i0} - \alpha_{i1}\mathbf{h}_{ij}^T\boldsymbol\beta_1^{(\ell)} - \alpha_{i2}\mathbf{h}_{ij}^T\boldsymbol\beta_2\right)^2,$$

subject to ${\boldsymbol\beta_1^{(\ell)}}^T\mathbf{G}\boldsymbol\beta_2^{(\ell)} = 0$ and ${\boldsymbol\beta_2^{(\ell)}}^T\mathbf{G}\boldsymbol\beta_2^{(\ell)} = 1$, in which $\mathbf{G}$ is a $J \times J$ matrix with the $(i,j)-$th element being $\int_0^1 b_i(t)b_j(t)dt$. We apply the following iterative algorithm to obtain the estimates:

(1) Set an initial value for $\boldsymbol\beta_2$, which is denoted as $\boldsymbol\beta_2^{(0)}$.

(2) In the $m$-th iteration, given the current value of $\boldsymbol\beta_2^{(m)}$, we estimate $\boldsymbol\alpha_1^{(m)}, \boldsymbol\alpha_2^{(m)}, \mathbf{x}_0^{(m)}$ by minimizing

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}\left(y_{ij} - x_{i0} - \alpha_{i1}\mathbf{h}_{ij}^T\boldsymbol\beta_1^{(\ell)} - \alpha_{i2}\mathbf{h}_{ij}^T\boldsymbol\beta_2^{(m)}\right)^2$$

This is simply a least square problem.

(3) Given the estimate $\boldsymbol\alpha_1^{(m)}, \boldsymbol\alpha_2^{(m)}, \mathbf{x}_0^{(m)}$, we obtain the $(m+1)$-th value of $\boldsymbol\beta_2$ by minimizing

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}\left(y_{ij} - x_{i0}^{(m)} - \alpha_{i1}^{(m)}\mathbf{h}_{ij}^T\boldsymbol\beta_1^{(\ell)} - \alpha_{i2}^{(m)}\mathbf{h}_{ij}^T\boldsymbol\beta_2\right)^2$$

subject to $(\boldsymbol{\beta}_1^{(\ell)})^T \mathbf{G} \boldsymbol{\beta}_2 = 0$ and ${\boldsymbol{\beta}_2}^T \mathbf{G} \boldsymbol{\beta}_2 = 1$. This is a least squares problem with the linear constraints, which can also be solved efficiently using the Least Squares with Equalities and Inequalities (LSEI) algorithm proposed by Lawson and Hanson (1974).

(4) Repeat Step (2) and Step (3) until the convergence reaches. In the end, we obtain the estimate of $\boldsymbol{\beta}_2^{(\ell)}$, $\boldsymbol{\alpha}_1^{(\ell)}$, $\boldsymbol{\alpha}_2^{(\ell)}$ and $\mathbf{x}_0^{(\ell)}$ given the value of $\boldsymbol{\beta}_1^{(\ell)}$.

Step III: Given the estimated value $\boldsymbol{\beta}_2^{(\ell)}$, we treat $\boldsymbol{\beta}_1$ as unknown again and apply the same algorithm within Step II to obtain the estimate for $\boldsymbol{\beta}_1^{(\ell+1)}$ and $\boldsymbol{\alpha}_1^{(\ell+1)}$, $\boldsymbol{\alpha}_2^{(\ell+1)}$ and $\mathbf{x}_0^{(\ell+1)}$.

Step IV: Repeat Step II and III until the algorithm converges.

### 5.3.3 Estimating More DeFECs

Given the estimates for the first m DeFECs, $m = 2, 3, \ldots,$ we can obtain the $(m+1)$-th DeFEC and the corresponding DeFEC scores $\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_{m+1}$, and the starting value $\mathbf{x}_0$ using a similar strategy described in the previous section. To be more specific, we can iterate between the values of $\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_{m+1}, \mathbf{x}_0$ and the $(m+1)$-th DeFEC $\phi_{m+1}(t)$ by treating the previous $m$ estimated DeFECs as fixed. After we obtain the estimate of $\phi_{m+1}(t)$, we can further refine our estimate for the first $m$ estimated DeFECs iteratively by treating each DeFEC as unknown in each iteration. For instance, we can treat $\phi_1(t)$ as unknown and updates its estimate conditional on all other estimated DeFECs, $\phi_2(t), \ldots, \phi_{m+1}(t)$. The loss function decreases in every step and we stop until the algorithm converges.

### 5.3.4 Smoothness Regulation

In order to control the smoothness of the estimated DeFECs, $\phi_k(t), k = 1, \ldots, K$, we add a roughness penalty in the loss function (5.3). We estimate $\phi_1(t), \ldots, \phi_K(t)$ by minimizing:

$$\sum_{i=1}^{n} \frac{1}{n_i} \sum_{j=1}^{n_i} \left( y_{ij} - \left[ x_{i0} + \sum_{k=1}^{K} \alpha_{ik} \int_0^{t_{ij}} \phi_k(s) ds \right] \right)^2 + \sum_{k=1}^{K} \gamma_k \int \left[ \frac{d^2 \phi_k(t)}{dt^2} \right]^2 dt, \tag{5.6}$$

subject to $\langle \phi_k, \phi_j \rangle = 1$ if $k = j$ and $\langle \phi_k, \phi_j \rangle = 0$ if $k \neq j$, where $k, j \in \{1, \ldots, K\}$. The algorithm introduced in Subsections 3.1-3.3 will be modified accordingly. For instance, when estimating the first DeFEC, we can update Step III in Subsection 3.1 to:

Step III(b) Given the value of $\alpha_{1i}^{(\ell)}$ and $x_{i0}^{(\ell)}$, we update the estimate of $\boldsymbol{\beta}_1$ to $\boldsymbol{\beta}_1^{(\ell+1)}$ by minimizing

$$\frac{1}{n} \sum_{i=1}^{n} \frac{1}{n_i} \sum_{j=1}^{n_i} \left( y_{ij} - x_{i0}^{(\ell)} - \alpha_{i1}^{(\ell)} \int_0^{t_{ij}} \phi_1(s) ds \right)^2 + \gamma_1 \int \left[ \frac{d^2 \phi_1(t)}{dt^2} \right]^2 dt$$

subject to $||\phi_1||^2 = 1$.

The above minimization is essentially a quadratically constrained quadratic program (QCQP) problem. We use the R package `Rsolnp` (Ghalanos and Theussl, 2015) based on the SOLNP algorithm proposed by Ye (1987) to numerically solve it. We will demonstrate this method is our simulation study.

When estimating each DeFEC, only one tuning parameter is involved, which is the smoothing parameter $\gamma_k$. The value of $\gamma_k$ controls the amount of smoothness imposed on the estimated DeFEC. We propose to select its value using the leave-one-curve-out cross validation strategy. To be more specific, we treat the $i$-th curve as the test data set and all the other curves as the training data set. When we estimate the first DeFEC, we can compute the first DeFEC estimate $\hat{\phi}_1^{(-i)}(t)$ using all the training curves' observations. Then, the estimated score for the test curve, i.e., the $i$-th curve, can be calculated by minimizing

$$\sum_{j=1}^{n_i}(y_{ij} - x_{i0} - \alpha_1 \int_0^{t_{ij}} \hat{\phi}_1^{(-i)}(t)ds)^2,$$

in which $y_{ij}$ represents the $j$-th measurement observed at $t_{ij}$ for the $i$-th curve and $n_i$ denotes the total number of observations for the $i$-th curve. Then the predicted value is $\hat{y}_{ij}^{(-i)} = \hat{x}_{i0}^{(-i)} + \hat{\alpha}_{i1}^{(-i)} \int_0^{t_{ij}} \hat{\phi}_1^{(-i)}(s)ds$. The cross validation error for the $i$-th curve is

$$\frac{1}{n_i}\sum_{j=1}^{n_i}(y_{ij} - \hat{y}_{ij}^{(-i)})^2.$$

The cross validation error for $\gamma_1$ is given as

$$\text{CV}(\gamma_1) = \sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}(y_{ij} - \hat{y}_{ij}^{(-i)})^2.$$

For the following DeFECs, we proposed to select the smoothing parameter by treating the previous estimated DeFECs fixed.

### 5.3.5 Determine the Number of DeFECs

In practice, we need to determine the number of DeFECs used to recovery the derivative function given sparse and irregular observations. An appropriate number of DeFECs should balance between model complexity and goodness of fit. A larger number of DeFECs leads to a better fit of observations but introduces more variability of the estimate. In this chapter, we propose to use the AIC criterion proposed by Li et al. (2013) to select the number of DeFECs:

$$\text{AIC}(K) = N\log(\sigma^2_{[K]}) + N + 2nK,$$

in which $K$ denotes the number of DeFECs, $n$ denotes the number of individual curves and $N$ is the total number of observations. We can estimate variance of the noise, $\sigma^2_{[K]}$, using the mean squares of residuals:

$$\hat{\sigma}^2_{[K]} = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{n_i} (\mathbf{y}_i - \hat{\mathbf{y}}_{i,[K]})^T (\mathbf{y}_i - \hat{\mathbf{y}}_{i,[K]}), \tag{5.7}$$

where $\hat{\mathbf{y}}_{i,[K]}$ is the fitted value for the $i$-th curve when using $K$ DeFECs.

## 5.4 Real Data Application

We demonstrate our proposed method by analyzing the data from a prospective study on body fat accretion in a cohort of 162 girls from the MIT Growth and Development Study (Phillips et al., 2003). At the start of this study, all girls were pre-menarcheal and non-obese, as determined by a triceps skinfold thickness less than the 85th percentile. All girls were followed over time according to a schedule of annual measurements until four years after menarche. The final measurement was scheduled on the fourth anniversary of their reported date of menarche. At each examination, a measure of body fatness was obtained based on bioelectric impedance analysis and a measure of percent body fat percentage was derived. This data set has totally 1049 measurements for the percent body fat, with an average of 6.4 measurements per subject. Figure 5.1 shows the data for 10 randomly-selected individuals.

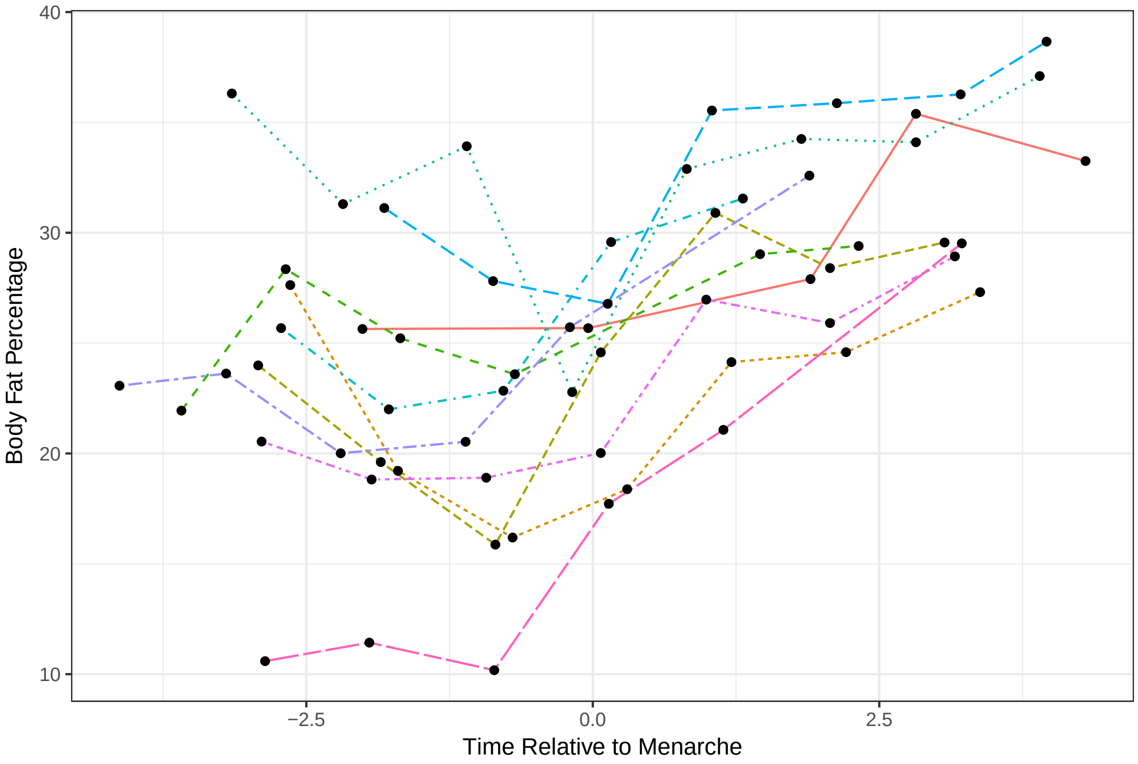

Figure 5.1: The measured body fat percentage before and after menarche for 10 randomly-selected girls in the MIT Growth and Development Study.

Table 5.1: The AIC values for various number of DeFECs when conducting the derivative functional empirical component analysis (DeFECA) for the MIT Growth and Development Study.

| Number of DeFECs | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| AIC | 3270.56 | 3221.03 | 3218.81 | 3267.54 |

The objective of our analysis is to recover the derivatives of individual growth curves and to explore their major variations. The smoothing parameters are selected from $\{0, 0.01, 0.1, 1, 10\}$ using the leave-one-curve-out cross validation. The selected smoothing parameters for the first 3 estimated DeFECs are $0, 0.1$, and $0.01$, respectively. The number of DeFECs is selected to be 3 using the AIC criterion in (5.7), shown in Table 5.1.

Figure 5.2 shows the estimated derivative functions for 4 individuals with various number of measurements. The top 4 panels plot the derivative functions and the bottom 4 panels gives the corresponding estimated trajectories. The estimated trajectories fit the observed data well despite that different individuals have different growth dynamics.

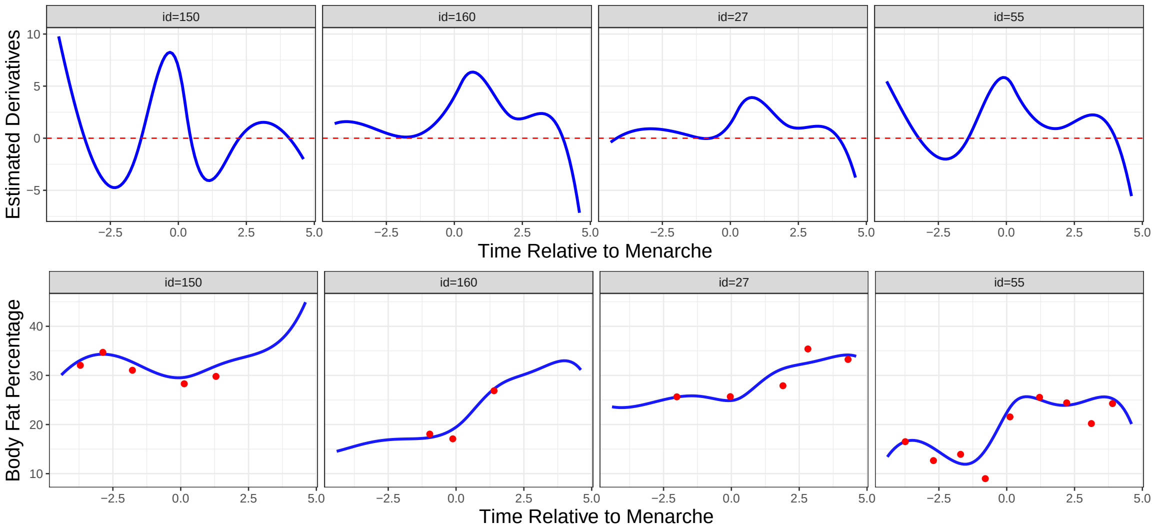

Figure 5.2: The estimated derivatives for 4 selected girls in the top 4 panels. The corresponding estimated trajectories are shown in the bottom 4 panels. The dots indicate the observed data.

We can cluster the subjects' dynamics patterns based on their estimated DeFEC scores. More specifically, we apply the K-means method on their estimated DeFEC scores to cluster them. Three clusters are identified, which are shown in Figure 5.3. Figure 5.4 displays the estimated derivative functions for five subjects in each cluster. It shows that the subjects in different group clearly has different dynamic patterns. For instance, the subjects in Group 1 exhibits positive dynamics around the age of menarche, which also indicates that they tend to gain body fact during this time period. By contrast, subjects in Group 2 have negative dynamics before menarche and positive dynamics after menarche, which means their body fact percentages decreases to its lowest level before menarche and gradually bounce back after menarche. In addition, the body fat percentage of those subjects in Group 3 seems have lower changing rate compared to the first two Groups on average.

Figure 5.5 shows the estimated DeFECs. The first estimated DeFEC represents the first major mode of variation in the girls' growth rates of their body fat percentage. It is positive in the whole time interval except the right boundary. It indicates that the first major mode of variation in the girls' growth rates is the weighted average of the growth rates. The largest weight happens at around one year after menarche, which suggests that the girls have the largest variation in their growth rates at this time point. The second DeFEC represents the second major mode of variation in the girls' growth rates of their body fat percentage, which is negative in the time period from 3 years prior to menarche until menarche and positive in the other time period. It indicates that the second major mode of variation in

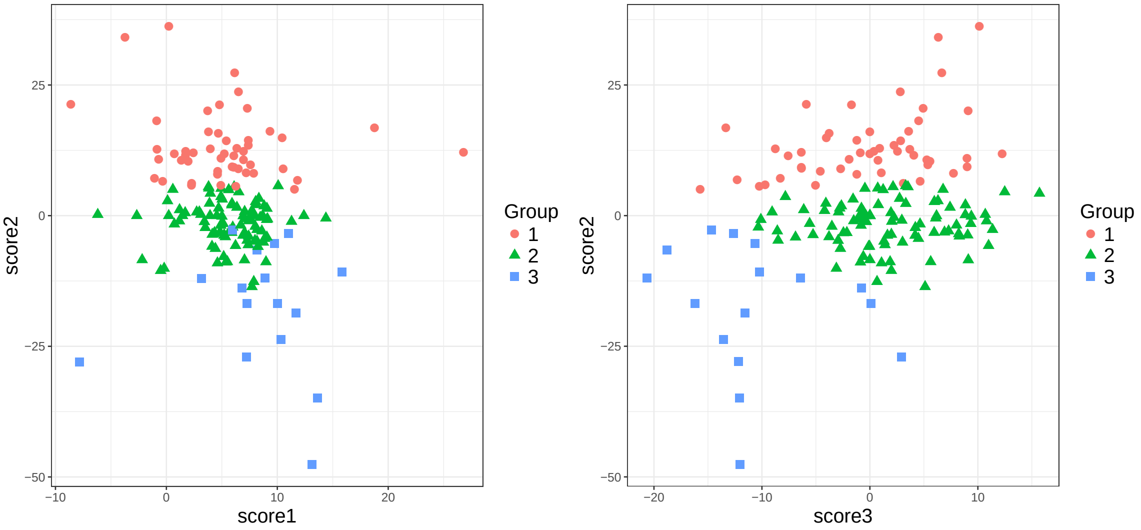

Figure 5.3: The estimated DeFEC scores for all 162 girls. Each dot represents one girl. The left panel shows the first estimated DeFEC score against the second estimated DeFEC score. The right panel shows the second DeFEC score against the third DeFEC score. The three dot shapes represent the three clusters identified by the K-means method.

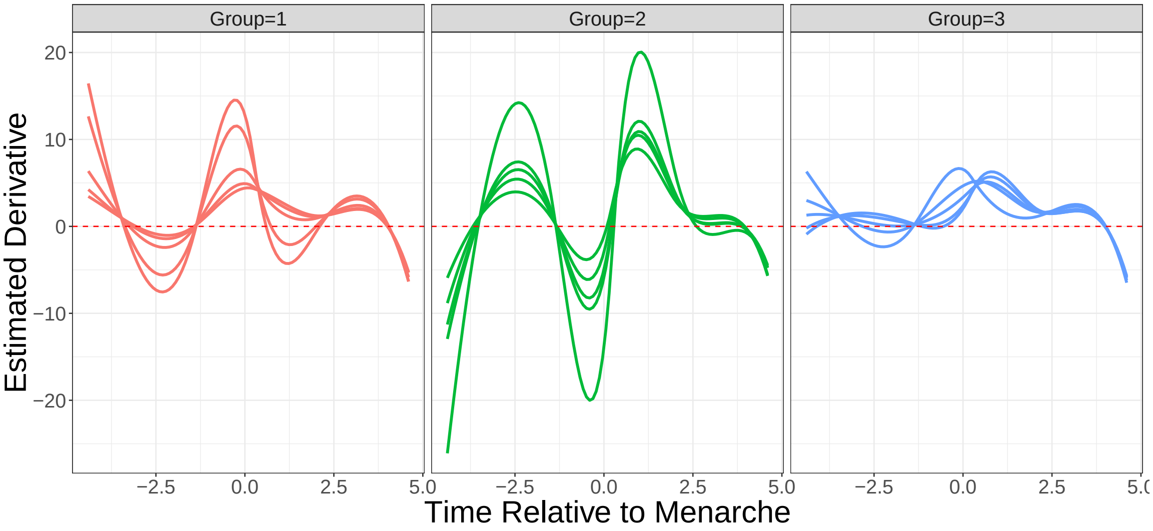

Figure 5.4: The estimated derivative functions for five subjects in each of three clusters identified by the K-means method on the estimated DeFEC scores.

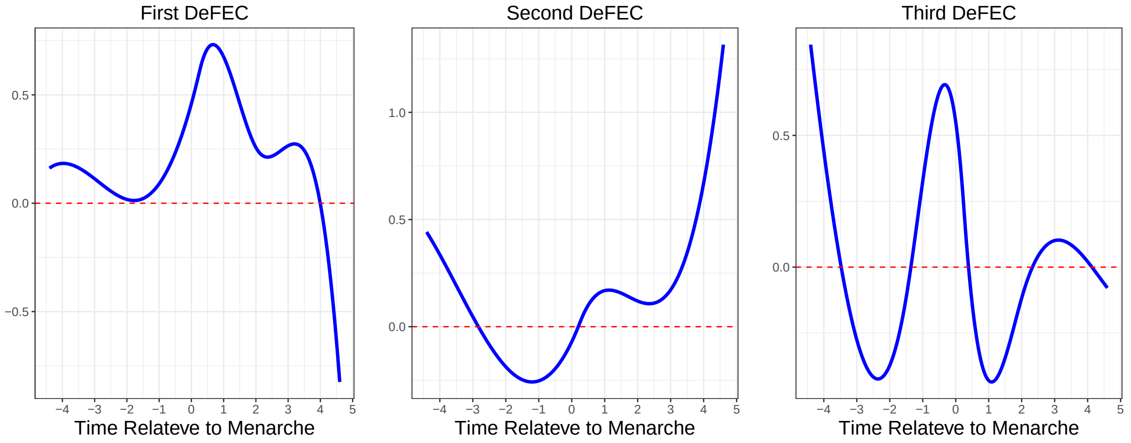

Figure 5.5: The top three derivative functional empirical components (DeFECs) estimated from the MIT Growth and Development Study.

the girls' growth rates is the contrast of the girls' growth rates between the time period from 3 years prior to menarche until menarche and the other time period.

## 5.5 Simulation Studies

To evaluate the performance of the proposed DeFECA method, we conduct the following simulation study to compare with the derivative principal component analysis (DFCA) method proposed in Dai et al. (2017). In order to make the proposed DeFECA method and the DFCA method comparable, we simulate the curves $x_i'(t)$ such that $E(x_i'(t)) = 0$. Then in this simulation setting, the functional principal components (FPCs) for the derivative functions in DFCA are equivalent to our proposed derivative functional empirical components (DeFECs). Therefore, for the rest of this section, we unify both of them as DeFECs.

We simulate the derivative functions $x_i'(t), i = 1, \ldots, n$, as $x_i'(t) = \alpha_{1i}\phi_1(t) + \alpha_{2i}\phi_2(t)$, where the true DeFECs $\phi_1(t)$ and $\phi_2(t)$ are the estimates from the real data application, as shown in the left two panels of Figure 5.5. We generate the true DeFEC scores $\alpha_{i1}$ and $\alpha_{i2}$ in both Gaussian and non-Gaussian distributions. For the Gaussian scenario, the true DeFEC scores are generate from two independent Gaussian distributions. That is, $\alpha_{1i} \overset{i.i.d}{\sim} N(0, 30)$ and $\alpha_{2i} \overset{i.i.d}{\sim} N(0, 10)$. For the non-Gaussian scenario, the scores are generate from two independent gamma distributions. That is, $\alpha_{1i} \overset{i.i.d}{\sim}$ gamma$(1, 0.03)$ and $\alpha_{i2} \overset{i.i.d}{\sim}$ gamma$(1, 0.1)$. We choose the parameters of these two gamma distributions such that the standard derivations are roughly the same as those in the Gaussian case. The corresponding observed data for each $x_i(t)$ is generated by $y_{ij} = x_{i0} + \int_0^{t_{ij}} x_i'(s)ds + \epsilon_{ij}, i = 1, \ldots, n, j = 1, \ldots, n_i$, in which $x_{i0}$ represents the starting value for the $i$th curve, which is generated independently from $N(0, 3^2)$. The measurement errors $\epsilon_{ij}$ is generated from $N(0, 1)$. To achieve the sparseness, the number of data points, $n_i$, for each curve is sampled from a discrete uniform distribution on $\{3, 4, 5, 6, 7\}$. The corresponding time points, i.e.,$\{t_{ij}, j = 1, \ldots, n_i\}$, are uniformly generated in the entire time domain $[0, T]$. We generate $n = 200$ sample curves in each simulation repetition. In each simulation repetition, we apply our proposed DeFECA method to first estimate the DeFECs, i.e., $\phi_1(t)$ and $\phi_2(t)$, of the derivative functions and the DeFEC scores and starting values $x_{i0}$. Then the estimate for the derivative function can be expressed as:

$$\hat{x}_i'(t) = \hat{x}_{i0} + \sum_{k=1}^{\hat{K}} \hat{\alpha}_{ik}\hat{\phi}_k(t).$$

We use leave-one-curve-out cross validation to choose from $\{0, 0.001, 0.1, 1, 10\}$ for the values of the smoothing parameters. The number of DeFECs, $K$, is selected using AIC. Within each repetition, we compute the mean integrated square errors (MISE) between the estimated derivatives and the true derivatives:

$$\text{MISE} = \frac{1}{n}\sum_{i=1}^{n}\int\{x_i'(t) - \hat{x}_i'(t)\}^2 dt, \tag{5.8}$$

in which $x_i'(t)$ represents the true derivative function for the $i$th curve and $\hat{x}_i'(t)$ is the corresponding estimated derivative function. We also apply the DFCA method (Dai et al. (2017)) to estimate the derivative functions and calculate the corresponding MISE. We repeat this procedure for 100 simulation replicates.

The estimation results are summarized in Table 5.2. The DeFECA method yields smaller MISE on average in comparison to the DFCA method in both Gaussian and non-Gaussian scenarios. For instance, the average MISE of the DFCA method are 1110 and 3335 for Gaussian and non-Gaussian scenarios, respectively, both of which are almost three times as large as those of the DeFECA method. The derivative functions estimated from the DeFECA method are also more stable than the DFCA method, because the standard deviations of MISE for the DeFECA method are 83% and 75% smaller than those of the DFCA method for the Gaussian and non-Gaussian scenarios, respectively.

We also calculate the mean integrated square errors (MISE) between the true DeFEC and their estimates

$$\text{MISE}(\phi_k) = \int\{\phi_k'(t) - \hat{\phi}_k'(t)\}^2 dt, k = 1, 2. \tag{5.9}$$

The results are summarized in Table 5.3. First, the estimated DeFECs using the DeFECA method are much closer to the true DeFECs than those estimated with the DFCA method in both Gaussian and non-Gaussian scenarios. For instance, the mean $\text{MISE}_{\phi_1}$ from the DFCA method in the Gaussian scenario is 0.98, which is about 10 times as large as the mean $\text{MISE}_{\phi_1}$ of the DeFECA method. Second, we notice that the performance of the DeFECA method is generally much more stable compared to the DFCA method by comparing the

Table 5.2: The summary results for estimating the derivative functions using the DeFECA method and the DFCA method for 100 simulation replicates. The table shows the means, standard derivations (STDs), medians, minimums and maximums for the mean integrated square errors (MISE) defined in (5.8) for Gaussian and non-Gaussian scenarios.

| | Gaussian | | non-Gaussian | |
|---|---|---|---|---|
| | DFCA | DeFECA | DFCA | DeFECA |
| Mean | 1110 | 379 | 3335 | 1050 |
| STD | 640 | 110 | 1171 | 292 |
| Median | 971 | 354 | 3181 | 952 |
| Minimum | 349 | 237 | 1630 | 815 |
| Maximum | 3894 | 905 | 7005 | 2294 |

standard derivations of the MISEs. For example, the standard deviations of MISE($\psi_1$) is 0.1 using the DFCA method in comparison with 0.03 using the DeFECA method.

Table 5.3: The summary results for estimating the DeFECs ($\phi_1(t)$ and $\phi_2(t)$) using the DeFECA method and the DFCA method for 100 simulation replicates. The table shows the means, standard derivations (STDs), medians, minimums and maximums for the mean integrated square errors (MISEs) defined in (5.9) for the Gaussian and non-Gaussian scenarios.

| | | **Gaussian** | | **Non-Gaussian** | |
|---|---|---|---|---|---|
| | | MISE($\phi_1$) | MISE($\phi_2$) | MISE($\phi_1$) | MISE($\phi_2$) |
| **Mean** | DFCA | 0.98 | 1.49 | 0.93 | 1.54 |
| | DeFECA | 0.11 | 0.22 | 0.08 | 0.27 |
| **STD** | DFCA | 0.10 | 0.29 | 0.11 | 0.30 |
| | DeFECA | 0.03 | 0.12 | 0.01 | 0.13 |
| **Median** | DFCA | 0.97 | 1.52 | 0.93 | 1.62 |
| | DeFECA | 0.11 | 0.19 | 0.08 | 0.23 |
| **Minimum** | DFCA | 0.78 | 1.04 | 0.62 | 0.95 |
| | DeFECA | 0.06 | 0.06 | 0.07 | 0.09 |
| **Maximum** | DFCA | 1.21 | 1.94 | 1.20 | 1.95 |
| | DeFECA | 0.25 | 0.79 | 0.10 | 0.66 |

## 5.6 Conclusions

In this chapter, we propose a novel derivative functional empirical component analysis (DeFECA) method for estimating the individual derivative function as well as the major variation patterns of the underlying dynamics from sparse and irregularly longitudinal observations. The DeFECA method directly estimates the optimal empirical basis functions from the best approximation perspective, while most conventional methods, such as DFCA, first estimate the de-meaned covariance function from the data and then eigen-decompose the estimated covariance function to obtain FPCs. This new best approximatio' perspective enables the DeFECA method to predict the individual derivative function without estimating the mean function and without requiring the DeFECA scores to be Gaussian distributed.

We demonstrate the DeFECA method by analyzing the MIT Growth and Development study, in which the longitudinal body fat percentages are sparsely and irregularly observed before and after the menarche. Our DeFECA method can estimate the individual derivative functions and explore the major variation modes of the growth dynamics among all subjects. Furthermore, we compare the performance of the DeFECA method and the DFCA method in simulation studies. The DeFECA method is shown to outperform the DFCA method in both predicting the individual trajectory and recovering the optimal empirical basis functions.

# Chapter 6

# Sparse Functional Single Index Model

## 6.1 Introduction

In this chapter, we consider the problem of modeling the relationship between a functional predictor and a scalar response. Given a scalar response $Y$ and a functional predictor $X(t)$ observed in $[0, T]$, the conventional functional linear regression model assumes that the scalar response $Y$ is linked with a functional predictor in the following model:

$$E(Y|X) = \beta_0 + \int_0^T \beta(t)X(t)dt,$$

in which $\beta_0$ is the intercept and $\beta(t)$ is the coefficient function. The above functional linear regression model describes a linear relationship between the response and the functional predictor.

A natural way to extend the above functional linear regression model is to use a functional single index model (FuSIM):

$$E(Y|X) = g(\int_0^T \beta(t)X(t)dt), \tag{6.1}$$

in which $\beta(t)$ is the index or weight function and $g(\cdot)$ is the link function describing the relationship between the scalar response $Y$ and the intergral of the functional predictor $X(t)$ and the index function $\beta(t)$. If $g(x)$ takes a known parametric form, such as the identity or logit function, then the functional single index model reduces to the conventional functional linear model (Ramsay and Silverman, 2005) or generalized functional linear model (Müller and Stadtmüller, 2005), respectively.

If we take a closer look at the functional single index model (6.1), the index function $\beta(t)$ determines weights of the contribution of the functional predictor $X(t)$ related to the scalar response through the function $g(\cdot)$ at each time point $t \in [0, T]$. In particular, if $\beta(t) \equiv 0$ on a subinterval $I \subset [0, T]$, then the functional predictor $X(t)$ is not related to the response

$Y$ on this subinterval. It is of great interest to estimate the support region of $\beta(t)$, denoted by $\mathcal{S} = \{t : \beta(t) \neq 0\}$, in which the functional predictor is related to the the response.

In this chapter, we propose the following sparse functional single index model (sparse FuSIM):

$$E(Y|X) = g(\int_{\mathcal{S}} \beta(t)X(t)dt) \tag{6.2}$$

in which $\mathcal{S} \subset [0, T]$ is unknown. Our goal in this chapter is to simultaneously estimate the index function $\beta(t)$, the nonlinear function $g(\cdot)$ without any parametric assumptions, and identify the support region $\mathcal{S}$ on which $X(t)$ is related to $Y$. Such a sparsely-supported estimate of $\beta(t)$ can not only increase the prediction accuracy of the response, but also lead to a better interpretation of the functional single index model.

Several methods have been proposed to estimate the functional single index model (6.1). For instance, Chen et al. (2011) estimated the unknown link function $g(\cdot)$ by nonparametric kernel smoothing with polynomial convergence rate. Ma (2014) considered a functional single-index model with multiple functional predictors. They used B-spline basis functions to estimate both the link function $g(\cdot)$ and the index or slope function $\beta(t)$. They also showed the uniform convergence rates of the proposed spline estimators. In addition, Ait-Saïdi et al. (2008) proposed an alternative cross-validation procedure for the model estimation. Jiang and Wang (2011) and Li et al. (2017) considered another type of index model in which the response is a function and the covariates are scalar and functions respectively. However, all the methods above assume that the support of the index function is the same as the functional predictors. That is, $\mathcal{S} = [0, T]$.

In the conventional functional linear regression model, in which the link function $g(\cdot)$ takes a linear form, this sparse support estimate has been studied by Ramsay et al. (2009), Zhou et al. (2013), and Lin et al. (2016). The “FLiRTI” method proposed by Ramsay et al. (2009) approximates the coefficient function nonparametrically at some discrete grid points and penalizes the $L_1$ norm of $\beta(t)$ and its first several derivatives to determine whether the estimated function and its derivatives are zero at each point. As mentioned by Zhou et al. (2009), the coefficient function estimated by the “FLiRTI” method has a large variance when the number of points increases and the model is often over-parameterized. To overcome this issue, a two-step procedure proposed by Zhou et al. (2009) is used to obtain an initial estimate for the support region of the coefficient function in the first step. The estimated support region is refined in the second step. However, the computation cost of this two-step procedure is quite high and hard to implement. A simple one-step procedure called ‘SloS’ proposed by Lin et al. (2016) yields a smooth and locally sparse estimator of $\beta(t)$. The SloS method regularizes the sparseness and smoothness of the coefficient function using the functional SCAD penalty and the roughness penalty in a single optimization objective function. However, none of the above methods consider the scenario in which the functional predictor is related to the response by an unknown nonlinear link function $g(\cdot)$. To the best of our knowledge, this is the first study that considers the sparse support problem within the functional single index model framework.

The main contribution of our method is two folds. First, we propose a new sparse functional single index model, in which the support region of the coefficient function $\beta(t)$ is a subregion of the entire domain. Second, we propose an efficient algorithm to estimate the coefficient function $\beta(t)$, its support region and the unknown nonlinear link function $g(\cdot)$ simultaneously. An R package `sFuSIM` has been developed to implement the proposed method and is available at `https://github.com/YunlongNie/sFuSIM`.

The remainder of this chapter is organized as follows. In Section 6.2.1, we present the details of sparse FuSIM and the corresponding fitting procedure. In Section 6.3, we introduce an efficient algorithm for the parameter estimations, followed by the tunning parameter selection in the end. Section 6.4 applies sparse FuSIM to a bike rental dataset. The finite-sample performance of our method is evaluated in Section 6.5 using a carefully-designed simulation study. Section 6.6 provides the concluding remarks.

## 6.2 Methodology

### 6.2.1 A sparse Functional Single Index Model

Let $y_i, i = 1, \ldots, n$, represent the observations for the response and $x_i(t)$ be the corresponding functional predictor. We propose to estimate the sparse functional single index model (6.2) by minimizing the following criterion:

$$Q(\beta, g) = \frac{1}{n}\sum_{i=1}^{n}\Big(y_i - g(\int_0^T \beta(t)x_i(t)dt)\Big)^2 + \mathrm{PEN}_\lambda(\beta), \tag{6.3}$$

where the norm of $\beta(t)$ is required to be 1 to ensure the identifiability of $g(\cdot)$ and $\beta(t)$. Note that the integral inside the function $g(\cdot)$ is taken between $[0, T]$ instead of the true support region of $\beta(t)$, i.e., $S$. In order to obtain a sparsely-supported estimator for $\beta(t)$, we add a second term which penalizes the $L_1$ norm of the index function $\beta(t)$. There is a tuning parameter, $\lambda$, in the penalty, which controls the sparseness of the resulting $\beta(t)$. A large value of $\lambda$ will shrink the magnitude of $\beta(t)$ towards zero in some subintervals. Therefore, the resulting $\beta(t)$ will be only nonzero in the other subintervals and the union of these subintervals will be the estimated $S$. On the other hand, when $\lambda = 0$, our model will reduce to the conventional FuSIM model and the support of $\beta(t)$ will always be $[0, T]$.

### 6.2.2 Sparsity Penalty

The penalty term in (6.3) penalizes the $L_1$ norm of $\beta(t)$ to obtain a sparsely- supported estimate. We employ the functional SCAD method proposed by Lin et al. (2016), which is a functional generalization of the SCAD method (Fan and Li, 2001). The functional SCAD method is proposed in Lin et al. (2016) to find a locally sparse estimator for the coefficient function in functional linear regression models. The nice shrinkage property of functional SCAD allows the proposed estimator to locate null subregions of the coefficient function without over shrinking nonzero values of the coefficient functions. More spefically, the functional SCAD penalty in the sparse FuSIM method is defined as:

$$\mathrm{PEN}_\lambda(\beta) = \frac{1}{T}\int_0^T p_\lambda(|\beta(t)|)dt,$$

in which $p_\lambda(\cdot)$ is the SCAD function defined in Fan and Li (2001):

$$p_\lambda(u) = \begin{cases} \lambda u & \text{if } 0 \le u \le \lambda, \\ -\frac{u^2-2a\lambda u+\lambda^2}{2(a-1)} & \text{if } \lambda < u < a\lambda, \\ \frac{(a+1)\lambda^2}{2} & \text{if } u \ge a\lambda, \end{cases}$$

Here $a$ is 3.7, as suggested by Fan and Li (2001), and $\lambda$ is the tuning parameter. We refer to $\lambda$ as the sparsity parameter for the rest of this chapter. A large value of $\lambda$ will penalize the nonzero region of the corresponding $\beta(t)$, hence leading to a sparse estimation of $\beta(t)$. On the other hand, when the sparsity parameter, $\lambda$, is zero, the resulting $\beta(t)$ reduces to the conventional functional single index model.

## 6.3 Summary of Computing Algorithm

Given a specific value of $\lambda$ in the fSCAD penalty, we estimate $\beta(t)$ and the nonlinear function $g(\cdot)$ in an iterative fashion. The algorithm is summarized below:

Step I Set the initial value for $\beta(t)$, denoted as $\beta^{(0)}(t)$;

Step II Given the current value of $\beta^{(j)}(t)$, we estimate $g^{(j)}(\cdot)$ by minimizing

$$Q_1(g|\beta^{(j)}) = \frac{1}{n}\sum_{i=1}^{n}\Big(y_i - g(\int_0^T \beta^{(j)}(t)x_i(t)dt)\Big)^2$$

Step III Given the current value of $g^{(j)}(\cdot)$, we update the estimate of $\beta(t)$ to $\beta^{(j+1)}(t)$ by minimizing

$$\begin{aligned} Q_2(\beta|g^{(j)}) =& \frac{1}{n}\sum_{i=1}^{n}\Big(y_i - g^{(j)}(\int_0^T \beta(t)x_i(t)dt)\Big)^2 \\ &+ \mathrm{PEN}_\lambda(\beta) \end{aligned}$$

Step IV Repeat Step II and III until the algorithm coverages.

### 6.3.1 Estimating the Link Function $g(\cdot)$

We now give the details of estimating the link function $g(\cdot)$ in Step II of the computing algorithm. For simplicity, we omit the index for iteration in this subsection. Given the current estimate of $\hat{\beta}(t)$, the criterion given in Step II becomes

$$Q_1(g|\hat{\beta}(t)) = \frac{1}{n}\sum_{i=1}^{n}\Big(y_i - g(\hat{z}_i)\Big)^2,$$

in which $\hat{z}_i = \int_0^T \hat{\beta}(t)x_i(t)dt$. We propose to estimate $g$ by the local linear regression(Wand and Jones, 1994). The local linear regression estimator of $g$ is given as

$$\hat{g}(z) = \sum_{i=1}^{n}[\hat{m}_0 - \hat{m}_1(z_i - z)]K_h(z_i - z),$$

where $\hat{m}_0, \hat{m}_1$ is obtained by minimizing

$$\frac{1}{n}\sum_{i=1}^{n}\Big([y_i - m_0 - m_1(z_i - z)]K_h(z_i - z)\Big)^2.$$

Here $K_h(\cdot)$ is a kernel function with the bandwidth $h$. The bandwidth is selected by cross-validation as suggested by Fan and Gijbels (1995a).

### 6.3.2 Estimating the Index Function $\beta(t)$

This subsection covers the details of estimating the index function $\beta(t)$ in Step III of the computing algorithm. We also omit the index for iteration in this subsection for simplicity. Given the current estimate of $\hat{g}(\cdot)$, the criterion given in Step III becomes

$$Q_2(\beta|\hat{g}) = \frac{1}{n}\sum_{i=1}^{n}\Big(y_i - \hat{g}(\int_0^T \beta(t)x_i(t)dt)\Big)^2 + \mathrm{PEN}_\lambda(\beta). \tag{6.4}$$

The index function $\beta(t)$ is expressed as a linear combination of basis function: $\beta(t) = \mathbf{b}^T\mathbf{B}(t)$, where $\mathbf{B}(t) = (B_1(t), \dots, B_L(t))^T$ are basis functions and $\mathbf{b} = (b_1, \dots, b_L)^T$ are the corresponding basis coefficients. We choose cubic B-splines as our basis functions, because they have the sparse support property (de Boor, 2001), which is important for efficient computation and identification of the support region of $\beta(t)$. Then the integral inside the least squares term in (6.4)is expressed as

$$\int_0^T \beta(t)x_i(t)dt = \mathbf{b}^T\int_0^T \mathbf{B}(t)x_i(t)dt = \mathbf{Z}_i^T\mathbf{b},$$

in which $\mathbf{Z}_i = (\int B_1(t)x_i(t)dt, \dots, \int B_L(t)x_i(t)dt)^T$. Because $\hat{g}(\cdot)$ is a nonlinear function, minimizing $Q_2(\beta)$ in (6.4) is a nonlinear optimization problem. To solve this nonlinear optimization problem, we propose to apply the local approximation idea and obtain the minimizer in an iterative manner. Based on the current estimates $\beta^{(j)}(t) = \mathbf{b}^{(j)}\mathbf{B}(t)$ in the $j$-th iteration step, $j = 1, 2, \ldots$, the least squares term in (6.4) can be approximated by

$$\begin{aligned}
&\frac{1}{n}\sum_{i=1}^{n}\Big(y_i - \hat{g}(\int_0^T \beta(t)x_i(t)dt)\Big)^2 \\
\approx&\frac{1}{n}\sum_{i=1}^{n}\Big(\Big[y_i - \hat{g}(\mathbf{Z}_i^T\mathbf{b}^{(j)}) - \hat{g}'(\mathbf{Z}_i^T\mathbf{b}^{(j)})\Big]\Big[\mathbf{b}^T\mathbf{Z}_i - \mathbf{Z}_i^T\mathbf{b}^{(j)}\Big]\Big)^2 \\
=&(\mathbf{b}^{(j)} - \mathbf{b})^T\mathbf{G}^{(j)}(\mathbf{b}^{(j)} - \mathbf{b}), \qquad (6.5)
\end{aligned}$$

in which $y_i^{(j)} = y_i - \hat{g}(\mathbf{Z}_i^T\mathbf{b}^{(j)}) - \hat{g}'(\mathbf{Z}_i^T\mathbf{b}^{(j)})$ and the $L \times L$ matrix $\mathbf{G}^{(j)} = \frac{1}{n}\sum_{i=1}^{n}(y_i^{(j)})^2\mathbf{Z}_i^T\mathbf{Z}_i$.

Similarly, we can also use the local quadratic approximation(LQA) proposed in Fan and Li (2001) to approximate the penalty term in (6.4) given the current estimate $\beta^{(j)}(t)$:

$$\frac{1}{T}\int p_\lambda(|\beta(t)|)dt \approx \frac{1}{L-d}\sum_{\ell=1}^{L-d} p_\lambda\Big(\sqrt{\int_{t_{\ell-1}}^{t_\ell}\beta^2(t)dt}\Big),$$

in which $t_0, t_1, \ldots, t_{L-d}$ denote the sequence of the knots of the B-spline basis functions $\mathbf{B}(t)$, and $d$ represents the order of the basis functions. We further define

$$||\beta_{[\ell]}(t)||_2^2 \overset{def}{=} \int_{t_{\ell-1}}^{t_\ell}\beta^2(t)dt = \mathbf{b}^T\mathbf{B}_\ell\mathbf{b},$$

in which $\mathbf{B}_\ell$ denotes a $L \times L$ matrix with the (p,q)-entry as $\int_{t_{j-1}}^{t_j} b_p(t)b_q(t)dt$ when $\ell \leq p, q \leq \ell + d$ and zero elsewhere. Then we can derive that

$$\frac{1}{T}\int p_\lambda(|\beta(t)|)dt \approx \frac{1}{L-d}\Big[\mathbf{b}^T\boldsymbol{W}^{(j)}\mathbf{b} + C(\mathbf{b}^{(j)})\Big], \qquad (6.6)$$

where

$$\boldsymbol{W}^{(j)} = \frac{1}{2}\sum_{\ell=1}^{L-d}\Big(\frac{p'_\lambda(||\beta_{[\ell]}(t)||_2\sqrt{L-d/T})}{||\beta_{[\ell]}(t)||_2\sqrt{T/L-d}}\mathbf{B}_\ell\Big),$$

and

$$\begin{aligned}
C(\mathbf{b}^{(j)}) \equiv&\sum_{\ell=1}^{L} p_\lambda\Big(\frac{||\beta^{(j)}{}_{[\ell]}||_2}{\sqrt{T/L-d}}\Big) \\
&-\frac{1}{2}\sum_{\ell=1}^{L} p'_\lambda\Big(\frac{||\beta^{(j)}{}_{[\ell]}||_2}{\sqrt{T/L-d}}\Big)\frac{||\beta^{(j)}{}_{[\ell]}||_2}{\sqrt{T/L-d}}.
\end{aligned}$$

Putting (6.5) and (6.6) together, given the current estimate $\beta^{(j)}(t)$, the local quadratic approximation of $Q_2$ in (6.4) can be expressed as

$$\begin{aligned} Q_2(\mathbf{b}) \approx & (\mathbf{b}^{(j)} - \mathbf{b})^T \mathbf{G}^{(j)} (\mathbf{b}^{(j)} - \mathbf{b}) \\ & + \frac{1}{L-d}\Big[\mathbf{b}^T \boldsymbol{W}^{(j)} \mathbf{b} + C(\mathbf{b}^{(j)})\Big] + \gamma \mathbf{b}^T \Gamma \mathbf{b}. \end{aligned} \tag{6.7}$$

Taking the derivative of $Q_2$ with respect to $\mathbf{b}$, we have

$$\frac{\partial Q_2(\mathbf{b})}{\partial \mathbf{b}} = -2\mathbf{G}^{(j)}(\mathbf{b}^{(j)} - \mathbf{b}) + 2\frac{1}{L-d}\boldsymbol{W}^{(j)}\mathbf{b} + 2\gamma\Gamma\mathbf{b}.$$

Therefore, the updated estimate is $\widehat{\mathbf{b}} = \Big(\mathbf{G}^{(j)} + \frac{1}{L-d}\boldsymbol{W}^{(j)} + \gamma\Gamma\Big)^{-1} \mathbf{G}^{(j)}\mathbf{b}^{(j)}$. In the end, we can plug in the estimate $\widehat{\mathbf{b}}$ to obtain the estimate for the index function $\beta(t)$ as

$$\hat{\beta}(t) = \widehat{\mathbf{b}}^T \mathbf{B}(t).$$

### 6.3.3 Tuning Parameter Selection

The sparsity parameter $\lambda$ can be selected by information criteria such as AIC and BIC. In the real data application and simulation studies of this chapter, we employ BIC because it encourages sparse models. The BIC criterion is defined as

$$\mathrm{BIC} = n\log(\mathrm{RSS}/n) + \log(n)(p+1),$$

in which $n$ presents the sample size, RSS represents residuals sum of squares and $p$ is the number of nonzero elements in the estimated basis coefficients $\widehat{\mathbf{b}}$.

## 6.4 Application

There has been an increased demand for bicycle rentals in recent years, because renting is considered as a more economical and environmentally-friendly alternative to owning bicycles. Thus, it is of great interest to ensure a sufficient bike supply, which is critical for a successful business in this area. In this section, we try to gain a better understanding of the relationship between the weather conditions and the customers' rental behavior during the weekend days. More specifically, we focus on the capital bike share study (Fanaee-T and Gama, 2014), which are rentals to cyclists without membership in the Capital Bike Share program in Washington, D.C. The total counts of casual bike rentals are recorded from January 1st, 2011, to December 31st, 2012, for a total of 105 weeks. Weather information such as temperature is also collected on an hourly basis.

We restrict our analysis to rentals on Saturdays, in which there is a particularly high demand for casual bike rentals compared to the weekdays' bike rentals. Our goal is to study how Saturday rentals relate to the hourly temperature. Understanding the nature of this association can help predict the casual rental demand based on the weather forecast. We apply the sparse functional single index model (6.2) to address this problem, where the scalar response $Y$ is the total number of Saturday rentals, and the functional covariate $X(t)$ is the hourly temperature.

The sparse functional single index model (6.2) is estimated with our proposed method. The optimal value of the sparsity parameter $\lambda$ is selected from $\{10^{-1}, 10^1, 10^3, 10^5, 10^8\}$ by 10-fold cross-validation. Given the value of $\lambda$, the bandwidth of the local linear regression is selected by the leave-one-out cross-validation, which is implemented using the R package `NonpModelCheck` (Zambom and Akritas, 2017). The details of the bandwidth selection can be found in Fan and Gijbels (1995a). We use the Gaussian kernel function and find that the performance of our method is quite similar among different choices of the kernel function. The optimal value of the sparsity parameter is selected to be $10^5$ and the optimal bandwidth is selected to be 29.

Figure 6.1 displays the estimated coefficient function $\hat{\beta}(t)$. It shows that the support of the estimated coefficient function $\beta(t)$ is between 6 am to 10 pm, which indicates the bike rental is mainly impacted by the temperature in this time period. In addition, the estimated coefficient function starts increasing from 6 am, reaches its peak at around 2 pm and decreases afterward until 10 pm. This observation suggests that temperature around 2 pm is more influential to the bike rentals compared to other times throughout the day.

Figure 6.2 shows the estimated link function $\hat{g}(t)$. The estimated link functions $\hat{g}(\cdot)$ clearly has a nonlinear pattern. For interpretation, we plot the hourly temperature of three days which correspond to three different levels of the integral value $\int X(t)\hat{\beta}(t)dt$ in Figure 6.3. It shows that when the temperature is around zero, the numero of bike rentals are quite low, as shown in Day 2; when the temperature around 2 pm increases to around 20 degrees, the bike rental is at a high level as shown in Day 17; when the temperature around 2 pm is too high to around 38 degrees, the bike rental drops as shown in Day 77. Generally, the hourly temperature between 6 am to 10 pm greatly affects the number of bike rentals.

## 6.5 Simulation

To assess the performance of our method, we conduct a simulation study which mimics the real data application. The response $y_i, i = 1, \ldots, n$, is generated with the following model

$$y_i = g_0(\int \beta_0(t)X(t)dt) + \varepsilon_i,$$

in which the true coefficient function $\beta_0(t)$ and the true link function $g_0(\cdot)$ is taken as the estimated coefficient function in the bike rental application as shown in Figure 6.1 and 6.2 respectively. The random noise $\varepsilon_i$ follows a normal distribution with mean zero and standard deviation $\sigma_\varepsilon$. Note that the true coefficient function is only nonzero in [6,22] instead of the

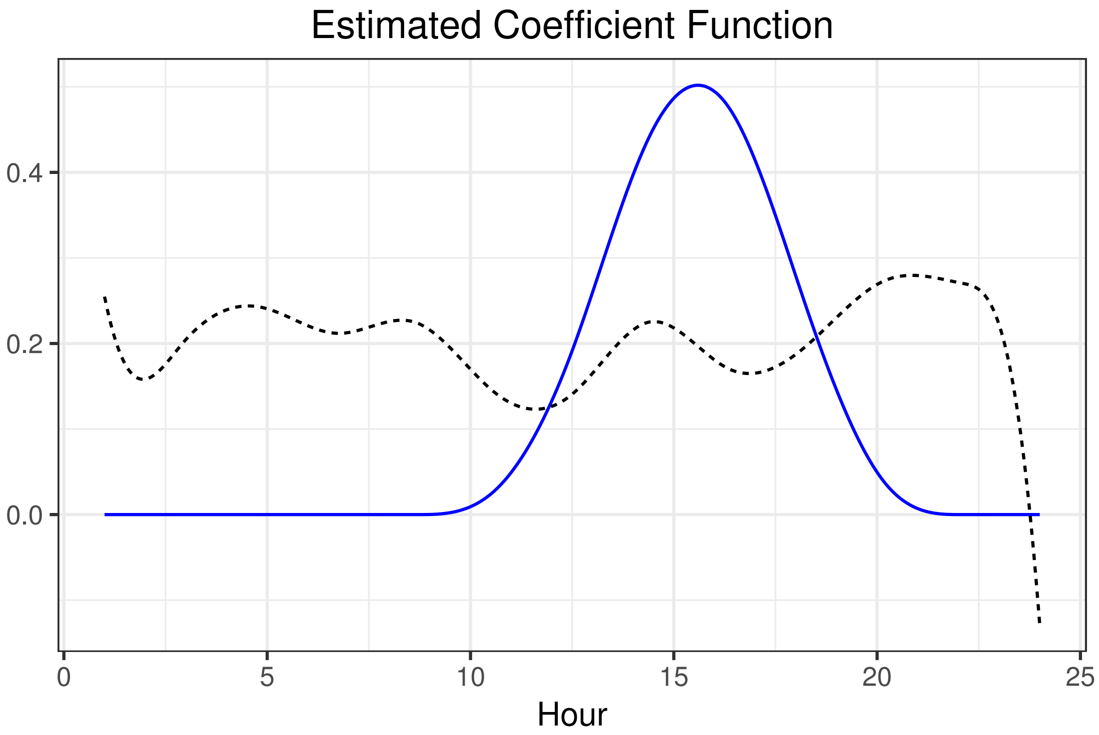

Figure 6.1: The estimated coefficient function $\hat{\beta}(t)$ using the sparse FuSIM (solid line) and the conventional FuSIM (dashed line) respectively.

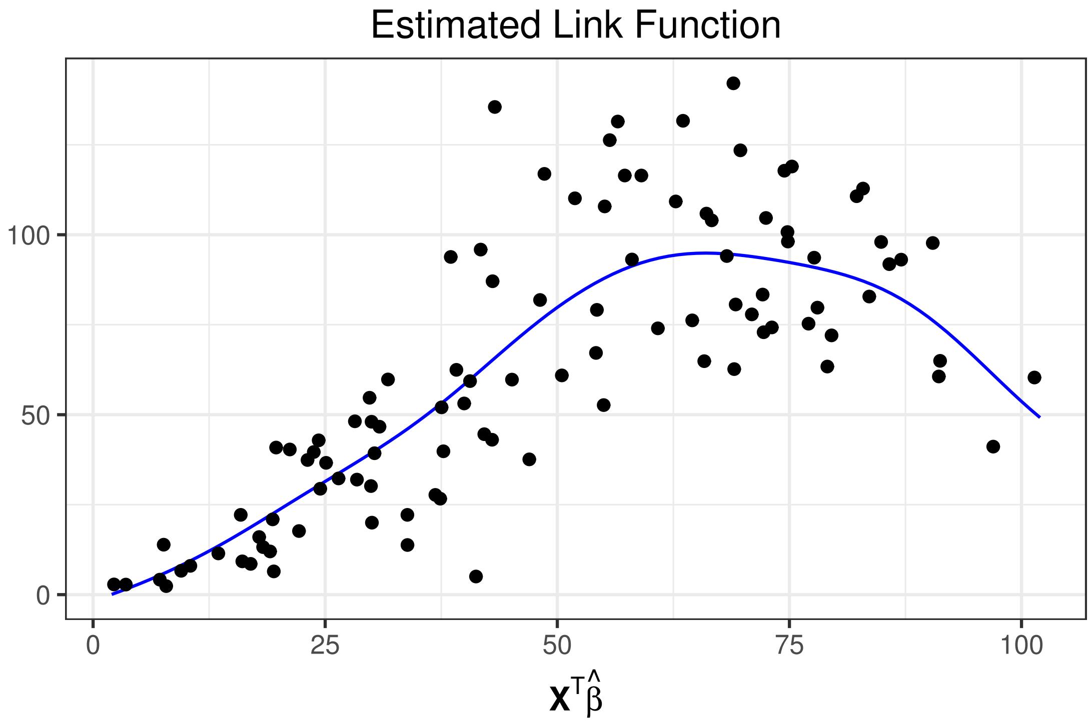

Figure 6.2: The estimated link function $\hat{g}(t)$ for the sparse FuSIM method.

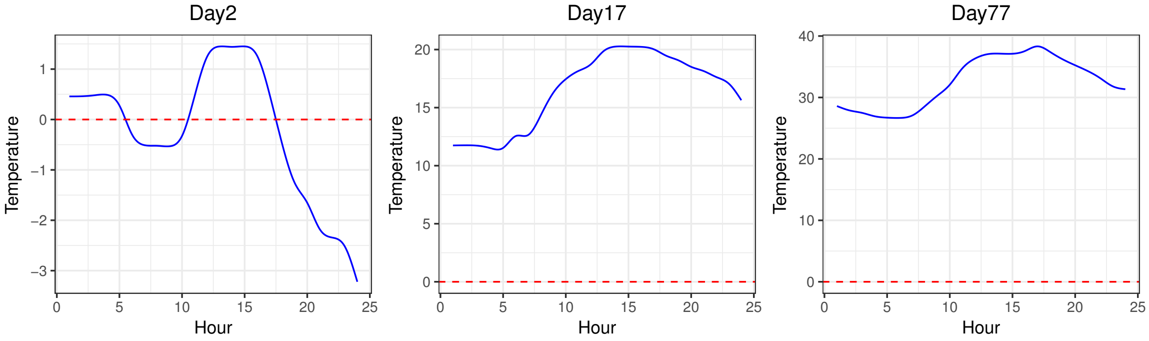

Figure 6.3: The hourly temperature for Day 2, 17 and 77. The estimated integral values are $2.5, 60.6$ and $113.7$.

Table 6.1: The means, standard deviations(SDs), medians, minimums and maximums of MSPEs for 100 Monte Carlo runs using the sparse FuSIM and the conventional FuSIM methods. Here 'sFuSIM' and 'FuSIM 'denote the sparse functional single index model and the conventional functional single index model, respectively.

| | sFuSIM | FuSIM |
|---:|---:|---:|
| Mean | 16.40 | 89.92 |
| SD | 34.84 | 310.91 |
| Median | 6.00 | 24.08 |
| Minimum | 1.34 | 2.18 |
| Maximum | 245.82 | 2433.30 |

entire time domain, i.e., [1,24]. In other words, the functional predictor $X_i(t)$ is only related to response $Y$ when $t \in [6, 22]$. We use the observed hourly temperature for 105 Saturdays in the real data application as our functional predictor $X_i(t), i = 1, \ldots, n$, in this simulation study, in which $n = 105$.

In each simulation repetition, we randomly select 80% of the samples from the observed hourly temperature curves as the training data set and treat the remaining samples' data as the test data set. Then we estimate the coefficient function $\beta(t)$ and the link function $g(\cdot)$ using the training data set only, predict the response variable using the test data set and obtain the mean squared prediction errors (MSPEs). The above procedure is repeated 100 times. The results are shown in Table 6.1. First, the average MSPE produced by the sparse FuSIM is 16.40, which is 5 times smaller than the average MSEP using the conventional FuSIM. In addition, the sparse FuSIM's performance is more stable compared to the conventional FuSIM, because the standard deviation of the sparse FuSIM method is much smaller than that of to the conventional FuSIM.

In addition, we compare the estimated $\hat{\beta}(t)$ with the true coefficient $\beta_0(t)$ by integrated mean square error(IMSE), which is defined as:

$$\text{IMSE} = \int_0^T (\hat{\beta}(t) - \beta_0(t))^2 dt.$$

The results are shown in Table 6.2. As can be seen, the estimated coefficient functions using the sparse FuSIM are much closer to the true coefficient function compared to those estimated from the conventional FuSIM method. For instance, the average IMSEs using our method is only 0.02, whereas the average IMSEs using conventional FuSIM is about 1.88.

To further quantify the ability of identifying the support of $\beta_0(t)$, we calculate the average of proportions of the true support that are correctly identified by the sparse FuSIM model. The quantity is computed as follows. For a given estimated $\hat{\beta}(t)$, we compute its value on a sequence of dense and equally-spaced points in [6, 22], which is the support of $\beta_0(t)$. For example, the sequence is taken to be 6, 6.001, 6.002, . . . , 21.999, 22. Then

Table 6.2: The means, standard deviations(SDs), medians, minimums and maximums of IMSEs for 100 Monte Carlo runs using the sparse FuSIM and the conventional FuSIM methods. Here 'sFuSIM' and 'FuSIM 'denote the sparse functional single index model and the conventional functional single index model, respectively.

| | sFuSIM | FuSIM |
|---:|---:|---:|
| Mean | 0.02 | 1.88 |
| SD | 0.03 | 1.51 |
| Median | 0.01 | 1.22 |
| Minimum | 0.00 | 0.08 |
| Maximum | 0.17 | 3.86 |

Table 6.3: The means, standard deviation(SD), medians, minimums and maximums of true positive(TP) and true negative(TN) proportions in percentage of the sparse FuSIM model for 100 Monte Carlo runs.

| | TP | TN |
|---:|---:|---:|
| Mean | 99.3 | 95.8 |
| SD | 3.1 | 9.6 |
| Median | 100.0 | 100.0 |
| Minimum | 85.7 | 60.0 |
| Maximum | 100.0 | 100.0 |

we compute the proportion of the points at which $\hat{\beta}(t)$ is nonzero among all points in the sequence. In addition, we can calculate the average of proportions of the non-effect regions that are correctly identified by the sparse FuSIM model in a similar fashion. That is, for a given estimated $\hat{\beta}(t)$, we compute its value on a sequence of dense and equally-spaced points within [0,6] and [22 24], in which $\beta_0(t)$ is strictly zero. For example, the sequence is taken to be 0, 0.001, 0.002, . . . , 5.999, 6 and 22, 22.001,. . . , 23.999,24. Then we compute the proportion of the points at which $\hat{\beta}(t)$ is zero among all points in the sequence. Finally, we average these two calculated proportions from 100 repetitions. To distinguish these two proportions, we refer to them as true positive (TP) and true negative (TN) proportions respectively. The results are summarized in Table 6.3. The performance of the sparse FuSIM is quite satisfactory in terms of estimating the support of $\beta(t)$ as the averaged TP and TN proportions are 99.3% and 95.8% respectively.

## 6.6 Summary

In this chapter, we propose a novel sparse functional single index model, which studies the relationship between a scalar response and a functional predictor. Our method is not only able to estimate the nonlinear relationship between the functional predictor and the scalar response without any parametric assumption, but also able to identify the region in which the functional predictor is related to the response.

We apply the sparse functional single index model to predict the daily bike rental counts using the temperature data. We find that bike rental is mainly affected by the temperature between 10 am and 22 pm. In addition, the temperature has a nonlinear impact on bike rental. To be more specific, the bike rentals increase with the temperature from around 0 degrees to around 20 degrees and decreases with temperature from 20 degrees until 38 degrees. We also compare our sparse functional single index model with the conventional functional single index model using a simulation study. The simulation study shows that our model yields better predictions to the response and also provides satisfactory estimation of the true effective region of the functional predictor.

# Chapter 7

# Estimation of Directed Time-varying Gene Regulation Network

## 7.1 Introduction

Gene regulation networks (GRN) have gained a lot of attention from biologists, geneticists, and statisticians in recent years. A variety of methods have been developed to infer gene regulation networks based on gene expression data such as Boolean networks (Thomas, 1973; Mehra et al., 2004; Laubenbacher and Stigler, 2004), information theory models (Steuer et al., 2002; Stuart et al., 2003), and Bayesian networks (Jensen, 1996; Needham et al., 2007). However, these methods only focus on static GRN, i.e., the network with the time-invariant topology given a set of genes. In fact, the regulation effect between a given pair of genes may change dramatically over the course of a biological process (Luscombe et al., 2004). Consequently, the GRN topology may be time-varying.

Ordinary differential equation (ODE) models (Cao and Zhao, 2008; Lu et al., 2011; Wu et al., 2014) have become popular to model the dynamical changes (both decreasing and increasing) of a target gene expression as a function of expression levels of all regulatory genes. The estimated regulation effect is also time-varying due to the variation of the regulatory gene expression. For instance, Cao and Zhao (2008) focused on parameter estimation for the ODE model when the type of regulation effect between two genes is known.

When the number of genes in the network is large, a sparse model is often preferable. But model selection (identification of true regulatory genes) has not been well addressed in the high-dimension context, where the total number of genes available far exceeds the number of gene expression measures. To solve this problem, Lu et al. (2011) reduced the dimension by first clustering genes into modules, then estimating a linear additive ODE model on the module level instead of the gene level. However, this method fails to capture the dynamical regulation effect at the gene level. In addition, the linear assumption on the regulation function may be impractical in many complex scenarios. Wu et al. (2014) modeled the regulation effect using a nonlinear function and solved the curse of dimensionality by adopting shrinkage techniques such as group LASSO (Yuan and Lin, 2006) and adaptive LASSO (Zou, 2006). On the other hand, once the regulatory genes are selected, the global topology of the GRN will stay constant during the whole process. However, in reality, the regulation effect from one gene might exist only in a certain time period rather than during the whole biological process.

Thus, we would prefer a flexible model in which the global topology of the estimated GRN is time-varying. Several methods have been proposed to estimate time-varying networks. For instance, Hanneke et al. (2010) extended exponential random graph models (ERGMs) to model the topology change of a time-varying social network based on a number of evolution statistics such as edge-stability, reciprocity, and transitivity. However, their method can only recover the undirected interactions between the nodes and can only be scaled up to small-scale networks because of the sampling algorithm. Song et al. (2009) and Kolar et al. (2010) proposed a kernel-reweighted logistic regression model with the $L_1$ penalty to estimate a time-varying GRN, which can be scaled up to large networks. Another advantage of their method is to allow both smoothing and sudden changes in the network topology. Kolar and Xing (2009) established the consistency of the kernel-smoothing $L_1$ regularized method. But both Song et al. (2009) and Kolar et al. (2010) only took binarized gene expression data as the input, and were also limited to undirected interactions between the genes. To the best of our knowledge, no existing methods use differential equations to model directed time-varying networks and estimate directed time-varying networks from continuous gene expression data. This is the main focus of this chapter.

Our mothod makes three crucial contributions. First, we model the dynamical feature of directed GRN using a high-dimensional nonlinear ODE model, in which the regulation function is a nonlinear function of the regulatory gene expression and is exactly zero in those intervals when no regulation effect happens. Hence our model allows the global topology of the directed GRN to be time-varying. Second, we propose a carefully-designed shrinkage technique called the functional smoothly clipped absolute deviation (fSCAD) method to do three tasks simultaneously: detecting significant regulatory genes for any given gene, identifying the intervals in which the significant regulatory genes have the regulation effect, and estimating the nonlinear regulation function without any parametric assumption. Finally, our method is computationally efficient and can be scaled up to the high-dimensional context. An R package called ‘flyfuns’ is developed to implement our proposed method. This package can be downloaded at `http://www.sfu.ca/~nyunlong/research/grn/`. A demonstration is also provided on the website.

The rest of this chapter is organized as follows. Details of our method are introduced in Section 7.2. Our method is demonstrated with a real data example in Section 7.3, where we estimate a time-varying directed gene regulation network among 20 Drosophila melanogaster genes during the embryonic stage. Section 7.4 presents a simulation study to investigate the finite sample performance of our method in comparison with conventional methods. Conclusions are given in Section 7.5.

## 7.2 Method

### 7.2.1 An ODE Model for Time-Varying Directed Gene Regulation Networks

Suppose a time-varying directed gene regulation network has $G$ genes in total, and their expressions are measured in a certain time period. The following ODE model relates the rate of change of one target gene expression to the expression of all genes in the network:

$$\dot{X}_\ell(t) = \mu_\ell + \sum_{g=1}^{G} f_{g\ell}(X_g),\ \ell = 1, \ldots, G,\ \ t \in [0, T], \tag{7.1}$$

where $\dot{X}_\ell(t)$ denotes the first derivative of $X_\ell(t)$ at time $t$ for the target gene $\ell$, $\mu_\ell$ is the intercept term, and $f_{g\ell}(X_g)$ represents the regulation function of gene $g$ on gene $\ell$. Here we assume $\dot{X}_\ell(t)$ is known, and in Section 7.2.7 we will discuss how to estimate it. Note that our approach also belongs to the framework of the two-step estimation for ODE parameters (Ramsay and Silverman, 2002; Chen and Wu, 2008).

When the number of genes, $G$, is large, we assume that only a few genes regulate the expression of the target gene $\ell$. In other words, in the ODE model (7.1), only a few regulation functions $f_{g\ell}(X_g) \neq 0$ and all others $f_{g\ell}(X_g) \equiv 0$. This assumption implies the sparsity of the underlying directed GRN structure.

In addition, we assume that the regulation effect of a particular regulatory gene might only be significant when its expression level is within a certain range. We use $S_{g\ell}$ to denote the support or nonzero intervals of the regulation function $f_{g\ell}$. In other words, $f_{g\ell}(X_g) \neq 0$ when $X_g \in S_{gl}$ and $f_{g\ell}(X_g) = 0$ when $X_g \notin S_{gl}$. This assumption results in a dynamical directed GRN with a time-varying topology, because some regulation functions may be nonzero at one time and become zero at some other time.

Without any parametric assumption on the regulation function $f_{g\ell}(X_g)$, we represent $f_{g\ell}(X_g)$ as a linear combination of basis functions

$$f_{g\ell}(X_g) = \sum_{k=1}^{K_{g\ell}} \beta_{g\ell k}\phi_{g\ell k}(X_g) = \boldsymbol{\phi}_{g\ell}^T(X_g)\boldsymbol{\beta}_{g\ell}, \tag{7.2}$$

where $\boldsymbol{\phi}_{g\ell}(X_g) = (\phi_{g\ell 1}(X_g), \phi_{g\ell 2}(X_g), \ldots, \phi_{g\ell K_{g\ell}}(X_g))^T$ denotes the vector of basis functions, $\boldsymbol{\beta}_{g\ell} = (\beta_{g\ell 1}, \ldots, \beta_{g\ell K_{g\ell}})^T$ is the corresponding vector of basis coefficients, and $K_{g\ell}$ denotes the number of basis functions. If all the elements of $\boldsymbol{\beta}_{g\ell}$ are estimated to be zero, then $f_{g\ell}(X_g) \equiv 0$, and the corresponding gene is omitted from the ODE model. On the other hand, if only a few elements of $\boldsymbol{\beta}_{g\ell}$ are estimated to be zero, then the corresponding regulation function $f_{g\ell}(X_g)$ will be strictly zero in certain intervals.

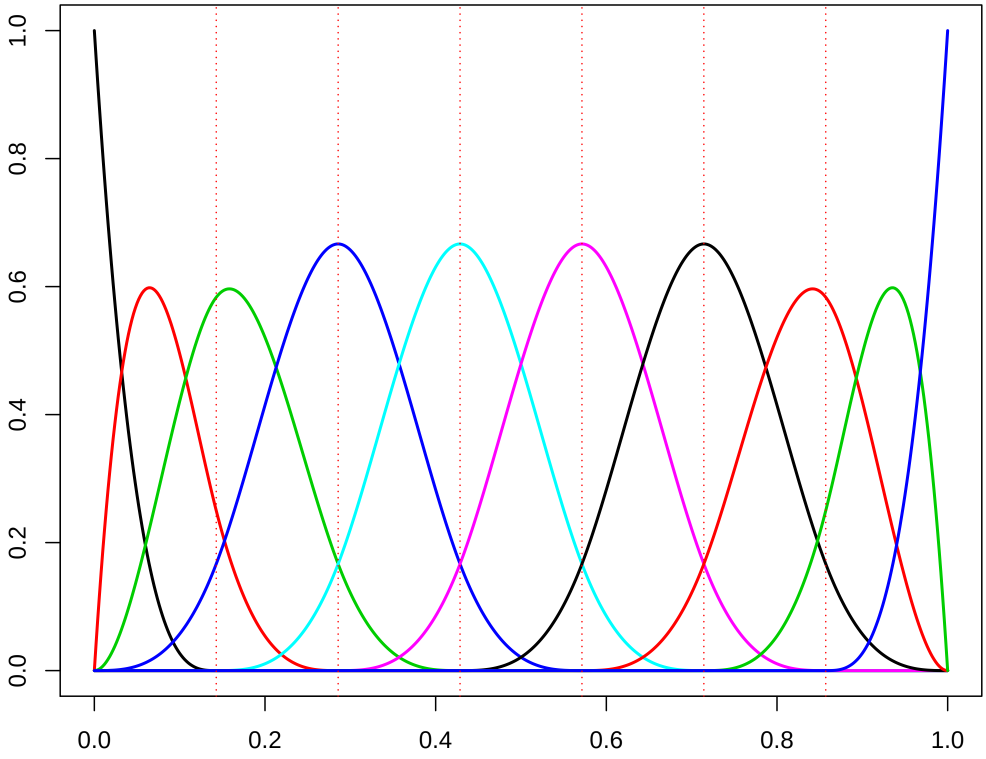

Figure 7.1: Ten cubic B-spline basis functions, defined by six interior knots. The locations of interior knots are indicated by vertical dashed lines.

In this method, we choose B-spline basis functions due to their compact support property: they are only nonzero in a local interval (de Boor, 2001). This property is crucial for the computation efficiency and the sparse estimation of our fSCAD method. Figure 7.1 shows an example of the ten cubic B-spline basis functions, defined by six interior knots. We can see that each of the six basis functions in the center is nonzero over four adjacent sub-intervals. In addition, the three left-most basis functions and the three right-most basis functions are nonzero over no more than four adjacent sub-intervals.

The method proposed in the rest of this section is trying to achieve the following three tasks simultaneously: detecting those significant regulatory genes whose regulation function $f_{g\ell}(X_g) \neq 0$, identifying the nonzero intervals, $S_{g\ell}$, of these regulation functions and estimating the nonlinear regulation function, i.e., $f_{g\ell}(X_g)$, in the corresponding nonzero intervals.

### 7.2.2 Sparsity Penalty

The most common way to achieve sparsity is to add a penalty term to the loss function. Our method belongs to this fashion by carefully choosing the penalty composition. Generally speaking, the main idea is to first partition each regulatory gene's whole expression domain into several subintervals. The penalty term depends on the magnitude of the regulation effect in each subinterval instead of in the entire expression domain.

The functional SCAD method was first proposed by Lin et al. (2016), which could be considered as a functional generalization of the SCAD (Fan et al., 2004). In Lin et al. (2016), they used fSCAD to estimate the coefficient function in the functional linear regression. However, they only had one functional predictor in their model, and did not consider the variable selection problem. In our work, we extend this method to do the variable selection in the high-dimensional differential equation model. At the same time, we use the fSCAD method to identify the nonzero intervals, $S_{g\ell}$, of these regulation functions and estimate the nonlinear regulation function in the estimated nonzero intervals simultaneously. Now we introduce our fSCAD penalty as follows.

The fSCAD penalty in our model is defined as

$$\sum_{g=1}^{G} \frac{M_{g\ell}}{\Delta_{x_g}} \int_{x_{gl}}^{x_{gu}} p_\lambda(|f_{g\ell}(X_g)|)dX_g,$$

where $x_{gl}$ and $x_{gu}$ are the lower and upper bounds of the expression of the $g$-th gene $X_g(t)$, $t \in [0,T]$, $\Delta_{x_g} = x_{gu} - x_{gl}$, and $M_{g\ell}$ is the number of subintervals partitioned by the knots of B-spline basis functions, so $M_{g\ell}$ is the total number of interior knots plus one. We use the cubic B-spline basis functions in our simulation and application, then $M_{g\ell} = K_{g\ell} - 3$, where $K_{g\ell}$ is the number of cubic spline basis functions. Inside the integral, $p_\lambda(\cdot)$ denotes the SCAD penalty function defined in Fan and Li (2001):

$$p_\lambda(u) = \begin{cases} \lambda u & \text{if } 0 \leq u \leq \lambda, \\ -\frac{u^2 - 2a\lambda u + \lambda^2}{2(a-1)} & \text{if } \lambda < u < a\lambda, \\ \frac{(a+1)\lambda^2}{2} & \text{if } u \geq a\lambda, \end{cases}$$

where $a$ is 3.7, as suggested by Fan and Li (2001), and $\lambda$ is the tuning parameter, which controls the sparsity of the regulation functions.

Let $x_0, x_1, \ldots, x_{M_{g\ell}}$ denote the sequence of the knots of B-spline basis function. Lin et al. (2016) has shown that

$$\frac{1}{\Delta_{x_g}} \int_{x_{gl}}^{x_{gu}} p_\lambda(|f_{g\ell}(X_g)|)dX_g = \frac{1}{M_{g\ell}} \lim_{M_{g\ell} \to +\infty} \sum_{j=1}^{M_{g\ell}} p_\lambda\left(\sqrt{\frac{M_{g\ell}}{\Delta_{x_g}} \int_{x_{g,j-1}}^{x_{gj}} [f_{g\ell}(X_g)]^2 dX_g}\right). \quad (7.3)$$

From (7.3), one can see that the fSCAD penalty is essentially the summation of penalties in each subinterval $[x_{g,j-1}, x_{gj}]$. In each subinterval $[x_{g,j-1}, x_{gj}]$, the penalty is governed by the magnitude of the regulation effect quantified by $\int_{x_{g,j-1}}^{x_{gj}} [f_{g\ell}(X_g)]^2 dX_g$. Thus fSCAD compares all gene regulation effects on each subinterval and tends to shrink those insignificant regulation effects towards zero without over-shrinking those significant regulation effects. The value of $\lambda$ determines the size of the shrinkage effect. For instance, a larger value of $\lambda$ will lead to smaller nonzero region for $f_{g\ell}(X_g)$. If the nonzero region of the regulation function does not exist, the corresponding gene is omitted from the ODE model. Thus we identify those genes that have no regulation effects. On the other hand, if the nonzero region does exist, the corresponding gene will have a significant regulation effect when its expression level is within the estimated nonzero region.

In comparison with group LASSO, the advantage of fSCAD is that it is able to discover a strong regulation effect even when this effect only exists for a small subinterval. Essentially, fSCAD penalizes the gene regulation function based on their regulation effects on each subinterval, whereas group LASSO cannot achieve this because its penalty depends on the regulation effect in the whole interval. For instance, if the regulation effect of one gene only exists in a short interval, group LASSO will still shrink the effect to zero and ignore its regulation effect completely even though the magnitude of the regulation effect is quite large in that short interval.

### 7.2.3 Roughness Penalty

We assume that the regulation function $f_{g\ell}(X_g)$ is a smooth function of $X_g$ because the regulation effect is not expected to change dramatically when the regulatory gene's expression has a small change.

In order to obtain a smooth regulation function, we introduce a roughness penalty. For a certain regulatory gene $X_g$, we define the roughness penalty as:

$$\left|\left|\frac{df^2_{g\ell}(X_g(t))}{dt^2}\right|\right|^2 = \int_0^T \left(\frac{d^2 f_{g\ell}(X_g(t))}{dt^2}\right)^2 dt.$$

Based on the basis function expansion for the regulation function $f_{g\ell}(X_g(t))$ defined in (7.2), one can show that the second derivative of $f_{g\ell}(X_g(t))$ can be calculated as

$$\frac{d^2 f_{g\ell}(X_g(t))}{dt^2} = \sum_{k=1}^{K_{g\ell}} \beta_{g\ell k} \frac{d^2 \phi_{g\ell k}(X_g(t))}{dt^2} = \sum_{k=1}^{K_{g\ell}} \beta_{g\ell k} d_{g\ell k},$$

where

$$d_{g\ell k} = \frac{d^2 \phi_{g\ell k}(X_g(t))}{dt^2} = \frac{d^2 \phi_{g\ell k}}{dX_g^2}\left(\frac{dX_g}{dt}\right)^2 + \frac{d\phi_{g\ell k}}{dX_g}\frac{d^2 X_g}{dt^2}. \tag{7.4}$$

The roughness penalty for all $G$ regulation functions is given as

$$R_\ell = \sum_{g=1}^{G} \left|\left| \frac{df_{g\ell}^2(X_g(t))}{dt^2} \right|\right|^2 = \sum_{g=1}^{G} \int_0^T \left( \sum_{k=1}^{K_{g\ell}} \beta_{g\ell k} d_{g\ell k} \right)^2 dt. \tag{7.5}$$

### 7.2.4 Parameter Estimation

Combining the fSCAD penalty (7.3) and the roughness penalty (7.5), we estimate $f_{g\ell}(X_g)$ via minimizing the following loss function:

$$\begin{aligned} Q(\boldsymbol{\beta}_\ell) = &\frac{1}{n} \sum_{i=1}^{n} \Big( \dot{X}_\ell(t_i) - \sum_{g=1}^{G} f_{g\ell}(X_g(t_i)) \Big)^2 + \gamma \sum_{g=1}^{G} \left|\left| \frac{df_{g\ell}^2(X_g(t))}{dt^2} \right|\right|^2 \\ &+ \sum_{g=1}^{G} \frac{M_{g\ell}}{\Delta_{x_g}} \int_{x_{gl}}^{x_{gu}} p_\lambda(|f_{g\ell}(X_g)|) dX_g, \end{aligned} \tag{7.6}$$

where $\boldsymbol{\beta}_\ell = (\boldsymbol{\beta}_{1\ell}^T, \boldsymbol{\beta}_{2\ell}^T, \ldots, \boldsymbol{\beta}_{G\ell}^T)^T$, is a length $GK$ column vector of all basis function coefficients.

The first term in (7.6) quantifies the goodness of fit to the derivative. The second term is the summation of the roughness penalty for each regulation function, and $\gamma$ is the smoothing parameter which controls the smoothness of all regulation functions. The last term corresponds to the fSCAD penalty.

For simplicity, we recast each part of the loss function in (7.6) into a matrix form. Following the notations in (7.2), the first term of the loss function can be expressed as

$$\frac{1}{n} \sum_{i=1}^{n} \Big( \dot{X}_\ell(t_i) - \sum_{g=1}^{G} f_{g\ell}(X_g(t_i)) \Big)^2 = \frac{1}{n} (\dot{\mathbf{X}}_\ell - \boldsymbol{\Phi}_\ell^T \boldsymbol{\beta}_\ell)^T (\dot{\mathbf{X}}_\ell - \boldsymbol{\Phi}_\ell^T \boldsymbol{\beta}_\ell), \tag{7.7}$$

where $\dot{\mathbf{X}}_\ell = (\dot{X}_\ell(t_1), \dot{X}_\ell(t_2), \ldots, \dot{X}_\ell(t_n))^T$ is a length $n$ column vector, $\boldsymbol{\Phi}_\ell = [\boldsymbol{\Phi}_{1\ell n}, \boldsymbol{\Phi}_{2\ell n}, \cdots, \boldsymbol{\Phi}_{G\ell n}]^T$ is a $GK \times n$ matrix,
$\boldsymbol{\Phi}_{g\ell n} = [\boldsymbol{\phi}_{g\ell}(X_g(t_1)), \boldsymbol{\phi}_{g\ell}(X_g(t_2)), \ldots, \boldsymbol{\phi}_{g\ell}(X_g(t_n))]$ is a $K_{g\ell} \times n$ matrix, and
$\boldsymbol{\phi}_{g\ell}(X_g(t_1)) = \Big( \phi_{g\ell 1}(X_g(t_1)), \phi_{g\ell 2}(X_g(t_1)), \ldots, \phi_{g\ell K_{g\ell}}(X_g(t_1)) \Big)^T$. Let $\mathbf{V}_{g\ell}$ be a $K_{g\ell} \times K_{g\ell}$ matrix with entries $v_{g,ij} = \int_0^T d_{g\ell i} d_{g\ell j} dx$, where $1 \leq i, j \leq K_{g\ell}$ and $d_{g\ell i}$ is expressed using (7.4).

Let $\mathbf{V}_\ell = diag(\mathbf{V}_{1\ell}, \mathbf{V}_{2\ell}, \ldots, \mathbf{V}_{G\ell})$ be a matrix ($GK_{g\ell} \times GK_{g\ell}$) with blocks $\mathbf{V}_{1\ell}, \mathbf{V}_{2\ell}, \ldots, \mathbf{V}_{G\ell}$ in its main diagonal and zeros elsewhere. Then the roughness penalty in (7.6) is transformed into the following form:

$$\gamma \sum_{g=1}^{G} \left|\left| \frac{df_{g\ell}^2(X_g(t))}{dt^2} \right|\right|^2 = \gamma \boldsymbol{\beta}_\ell^T \mathbf{V}_\ell \boldsymbol{\beta}_\ell. \tag{7.8}$$

Based on (7.3), the fSCAD penalty can be approximated as

$$\frac{M_{g\ell}}{\Delta_{x_g}}\int_{x_{gl}}^{x_{gu}} p_\lambda(|f_{g\ell}(X_g)|)dX_g \approx \sum_{j=1}^{M_{g\ell}} p_\lambda\Big(\sqrt{\frac{M_{g\ell}}{\Delta_{x_g}}\int_{x_{g,j-1}}^{x_{gj}}[f_{g\ell}(X_g)]^2 dX_g}\Big).$$

In addition, we define

$$||f_{g\ell[j]}||_2^2 \overset{def}{=} \int_{x_{g,j-1}}^{x_{gj}}[f_{g\ell}(X_g)]^2 dX_g = \boldsymbol{\beta}_{g\ell}^T \mathbf{M}_{g\ell j}\boldsymbol{\beta}_{g\ell},$$

where $\mathbf{M}_{g\ell j}$ is a $K_{g\ell}\times K_{g\ell}$ matrix with entries $m_{g\ell j,uv} = \int_{x_{g,j-1}}^{x_{gj}}\phi_{g\ell u}(X_g)\phi_{g\ell v}(X_g)dX_g$, if $j \le u, v \le j+d$ and zero otherwise. Using the local quadratic approximation (LQA) proposed in Fan and Li (2001), given some initial estimate $\boldsymbol{\beta}_{g\ell}^{(0)}$, we can derive that

$$\frac{M_{g\ell}}{\Delta_{x_g}}\int_{x_{gl}}^{x_{gu}} p_\lambda(|f_{g\ell}(X_g)|)dX_g \approx \boldsymbol{\beta}_{g\ell}^T \boldsymbol{W}_{g\ell}^{(0)}\boldsymbol{\beta}_{g\ell} + G(\boldsymbol{\beta}_{g\ell}^{(0)}),$$

where

$$\boldsymbol{W}_{g\ell}^{(0)} = \frac{1}{2}\sum_{j=1}^{M_{g\ell}}\Big(\frac{\dot{p}_\lambda(||f_{g\ell[j]}||_2\sqrt{M_{g\ell}/\Delta_{x_g}})}{||f_{g\ell[j]}||_2\sqrt{\Delta_{x_g}/M_{g\ell}}}\mathbf{M}_{glj}\Big),$$

and

$$G(\boldsymbol{\beta}_{g\ell}^{(0)}) \equiv \sum_{j=1}^{M_{g\ell}} p_\lambda\Big(\frac{||f_{g\ell[j]}||_2}{\sqrt{\Delta_{x_g}/M_{g\ell}}}\Big) - \frac{1}{2}\sum_{j=1}^{M_{g\ell}}\dot{p}_\lambda\Big(\frac{||f_{g\ell[j]}||_2}{\sqrt{\Delta_{x_g}/M_{g\ell}}}\Big)\frac{||f_{g\ell[j]}||_2}{\sqrt{\Delta_{x_g}/M_{g\ell}}}.$$

Adding all the fSCAD penalty for each gene, we have

$$\sum_{g=1}^{G}\frac{M_{g\ell}}{\Delta_{x_g}}\int_{x_{gl}}^{x_{gu}} p_\lambda(|f_{g\ell}(X_g)|)dX_g \approx \boldsymbol{\beta}_\ell^T \mathbf{W}_\ell^{(0)}\boldsymbol{\beta}_\ell + \sum_{g=1}^{G} G(\boldsymbol{\beta}_\ell^{(0)}), \tag{7.9}$$

where $\mathbf{W}_\ell^{(0)} = \text{diag}(\mathbf{W}_{1\ell}^{(0)}, \mathbf{W}_{2\ell}^{(0)}, \ldots, \mathbf{W}_{G\ell}^{(0)})$. Putting (7.7), (7.8) and (7.9) together, we obtain

$$Q(\boldsymbol{\beta}_\ell) = \frac{1}{n}(\dot{\mathbf{X}}_\ell - \boldsymbol{\Phi}_\ell^T\boldsymbol{\beta}_\ell)^T(\dot{\mathbf{X}}_\ell - \boldsymbol{\Phi}_\ell^T\boldsymbol{\beta}_\ell) + \gamma\boldsymbol{\beta}_\ell^T\mathbf{V}_\ell\boldsymbol{\beta}_\ell + \boldsymbol{\beta}_\ell^T\mathbf{W}_\ell^{(0)}\boldsymbol{\beta}_\ell + \sum_{g=1}^{G} G(\boldsymbol{\beta}_{g\ell}^{(0)}).$$

By minimizing $Q(\boldsymbol{\beta}_\ell)$, we obtain the estimate for the basis coefficients

$$\widehat{\boldsymbol{\beta}}_\ell = \frac{1}{n}\Big(\frac{1}{n}\boldsymbol{\Phi}_\ell\boldsymbol{\Phi}_\ell^T + \gamma\mathbf{V}_\ell + \mathbf{W}_\ell^{(0)}\Big)^{-1}\boldsymbol{\Phi}_\ell\dot{\mathbf{X}}_\ell.$$

Then we can plug the estimate, $\widehat{\boldsymbol{\beta}}_\ell$, into (7.2) to obtain the estimates for all regulation functions:

$$\hat{f}_{g\ell}(X_g) = \boldsymbol{\phi}_{g\ell}^T(X_g)\widehat{\boldsymbol{\beta}}_{g\ell}, \quad g = 1, \ldots, G, \quad \ell = 1, \ldots, G.$$

### 7.2.5 Identifiability Issue

Modeling multiple regulatory genes introduces an identifiability problem. For instance, suppose there are only two regulatory genes such that the ODE (7.1) is reduced to

$$\dot{X}_\ell(t) = \mu_\ell + f_{1\ell}(X_1(t)) + f_{2\ell}(X_2(t)).$$

Since simultaneously adding any constant to $f_{1\ell}(\cdot)$ and subtracted it from $f_{2\ell}(\cdot)$ does not affect the model prediction, $f_{1\ell}(\cdot)$ and $f_{2\ell}(\cdot)$ are only estimable up to an additive constant.

To address this issue, we apply a similar strategy as in Wood (2006), which constrains the sum of $f_{g\ell}(\cdot)$ to zero over the entire time domain. That is,

$$E(f_{g\ell}(X_g(t))) = 0, g = 1, \ldots, G. \tag{7.10}$$

This also implies that $\hat{\mu}_\ell = E(\dot{X}_\ell(t))$. In the rest part of this section, we briefly discuss how to include constraints (7.10) into the parameter estimation.

Denote $w_{g\ell k} = \sum_{i=1}^n \phi_{g\ell k}(X_g(t_i))$. The constraint (7.10) can be satisfied in a sample as

$$\sum_{g=1}^{G} \Big( \sum_{k=1}^{K_{g\ell}} \beta_{g\ell k} w_{g\ell k} \Big)^2 = 0. \tag{7.11}$$

Next we can recast the left side of (7.11) into a matrix form

$$\sum_{g=1}^{G} \Big( \sum_{k=1}^{K_{g\ell}} \beta_{g\ell k} w_{g\ell k} \Big)^2 = \boldsymbol{\beta}_\ell^T \boldsymbol{\Omega}_\ell \boldsymbol{\beta}_\ell,$$

where $\boldsymbol{\Omega}_\ell = diag(\boldsymbol{\Omega}_{1\ell}, \boldsymbol{\Omega}_{2\ell}, \ldots, \boldsymbol{\Omega}_{G\ell})$ and

$$\boldsymbol{\Omega}_{g\ell} = \begin{bmatrix} w_{g\ell 1}^2 & w_{g\ell 1} w_{g\ell 2} & \cdots & w_{g\ell 1} w_{g\ell K_{g\ell}} \\ w_{g\ell 2} w_{g\ell 1} & w_{g\ell 2}^2 & \cdots & w_{g\ell 2} w_{g\ell K_{g\ell}} \\ \vdots & \vdots & \cdots & \vdots \\ w_{g\ell K_{g\ell}} w_{g\ell 1} & w_{g\ell K_{g\ell}} w_{g\ell 2} & \cdots & w_{g\ell K_{g\ell}}^2 \end{bmatrix}.$$

We add $\lambda_I \boldsymbol{\beta}_\ell^T \boldsymbol{\Omega}_\ell \boldsymbol{\beta}_\ell$ to the loss function (7.6), in which $\lambda_I$ is a relatively large positive number to make sure that (7.11) holds. Consequently, the estimator $\widehat{\boldsymbol{\beta}}_\ell$ becomes

$$\widehat{\boldsymbol{\beta}}_\ell = \frac{1}{n} \Big( \frac{1}{n} \boldsymbol{\Phi}_\ell \boldsymbol{\Phi}_\ell^T + \gamma \mathbf{V}_\ell + \mathbf{W}_\ell^{(0)} + \lambda_I \boldsymbol{\Omega}_\ell \Big)^{-1} \boldsymbol{\Phi}_\ell \dot{\mathbf{X}}_\ell. \tag{7.12}$$

Note that $\boldsymbol{\Omega}_\ell$ is a singular matrix so that a very large value of $\lambda_I$ might cause $\frac{1}{n} \boldsymbol{\Phi}_\ell \boldsymbol{\Phi}_\ell^T + \gamma \mathbf{V}_\ell + \mathbf{W}_\ell^{(0)} + \lambda_I \boldsymbol{\Omega}_\ell$ in (7.12) to be almost singular. If that is the case, we recommend to try a new value of $\lambda_I$, for instance, half of the previous value.

Below we give the details of our algorithm to compute the estimated coefficients $\widehat{\boldsymbol{\beta}}_\ell$:

Step 1: Compute the initial estimate $\widehat{\boldsymbol{\beta}}_\ell^{(0)} = \frac{1}{n}\left(\frac{1}{n}\boldsymbol{\Phi}_\ell\boldsymbol{\Phi}_\ell^T + \lambda_I\boldsymbol{\Omega}_\ell\right)^{-1}\boldsymbol{\Phi}_\ell\dot{\mathbf{X}}_\ell$.

Step 2: In each iteration, given $\widehat{\boldsymbol{\beta}}_\ell^{(i)}$, compute the corresponding $\boldsymbol{W}_\ell^{(i)}$. Then $\widehat{\boldsymbol{\beta}}_\ell^{(i+1)} = \frac{1}{n}\left(\frac{1}{n}\boldsymbol{\Phi}_\ell\boldsymbol{\Phi}_\ell^T + \gamma\mathbf{V}_\ell + \boldsymbol{W}_\ell^{(i)} + \lambda_I\boldsymbol{\Omega}_\ell\right)^{-1}\boldsymbol{\Phi}_\ell\dot{\mathbf{X}}_\ell$. If a variable is very small in magnitude such that it makes $\left(\frac{1}{n}\boldsymbol{\Phi}_\ell\boldsymbol{\Phi}_\ell^T + \gamma\mathbf{V}_\ell + \boldsymbol{W}_\ell^{(i)} + \lambda_I\boldsymbol{\Omega}_\ell\right)$ almost singular or badly scaled so that inverting $\left(\frac{1}{n}\boldsymbol{\Phi}_\ell\boldsymbol{\Phi}_\ell^T + \gamma\mathbf{V}_\ell + \boldsymbol{W}_\ell^{(i)} + \lambda_I\boldsymbol{\Omega}_\ell\right)$ is unstable, then we manually shrink it into zero.

Step 3: Repeat Step 2 until $\widehat{\boldsymbol{\beta}}_{\boldsymbol{\ell}}^{\boldsymbol{(i)}}$ converges.

### 7.2.6 Choose Tuning parameters

We need to specify four tuning parameters in (7.12): the total number of basis functions used to represent each regulation function, $K_{g\ell}$; the smoothing parameter in the roughness penalty for each regulation function, $\gamma$; the fSCAD penalty for sparsity, $\lambda$; and the identifiability parameter, $\lambda_I$.

First of all, a large value of $K_{g\ell}$ is chosen to obtain a good approximation for each regulation function $f_{g\ell}(\cdot)$. This will not result in a saturated model since the smoothing parameter, $\gamma$, and fSCAD penalty parameter, $\lambda$, will control the roughness of the regulation functions. Second, $\lambda_I \in [10^4, 10^9]$ generally works well according to our experience and this choice is not crucial. We note that the value of $\lambda_I$ only affects the convergence speed. Once $K_{g\ell}$ and $\lambda_I$ are determined, one can use a popular selection criterion such as information criterion (AICc, BIC) or cross validation to search the optimal values for $\gamma$ and $\lambda$ on a discrete grid. Our experience from the real data application suggests that the AICc information criterion tends to work well from a practical perspective.

### 7.2.7 Derivative Estimation

The ODE model in (7.1) uses the derivatives of each gene as the response. In this section, we introduce a smoothing spline method to estimate the derivative of each gene based on the its own observed expression values. Other methods for the derivative estimation can also be used in our framework.

Let $Y_i$ denote the measurement for a particular gene at time $t_i$, $t_i \in [0, T]$. Suppose that $Y_i, i = 1, \ldots, n$, is from an unknown gene expression function $X(t)$. That is,

$$Y_i = X(t_i) + \epsilon_i, i = 1, \ldots, n,$$

where $\epsilon_i$ is independently and identically distributed from a normal distribution $N(0, \sigma_s^2)$. Our goal is to estimate $X(t)$ and $\dot{X}(t)$ from $Y_i,\ i = 1, \ldots, n$.

We first represent $X(t)$ using a linear combination of B-spline basis functions:

$$X(t) = \sum_{j=1}^{J} \theta_i \psi_j(t) = \boldsymbol{\psi}(t)^T \boldsymbol{\theta},$$

in which $\boldsymbol{\theta}$ is the length $J$ vector of coefficients, and $\boldsymbol{\psi}(t)$ is the length $J$ vector of basis functions. Then we estimate the vector of coefficients $\boldsymbol{\theta}$ by minimizing the following loss function:

$$Q_0(\boldsymbol{\theta}) = \sum_{i=1}^{n} \Big( Y_i - X(t_i) \Big)^2 + \lambda_0 \int \Big[ \ddot{X}(t) \Big]^2 dt,\ \lambda_0 > 0. \tag{7.13}$$

Intuitively, the first term in $Q_0(\boldsymbol{\theta})$ quantifies the goodness of fit to the data, and the second one controls the roughness of the estimated function. The relative importance between these two terms is controlled by $\lambda_0$. For instance, a larger value of $\lambda_0$ will lead to a smoother estimate for $X(t)$. Here we suggest using the generalized cross validation (GCV) score in Craven and Wahba (1978) to determine the value of $\lambda_0$.

To estimate the vector of the basis coefficients $\boldsymbol{\theta}$, we can rewrite (7.13) into a matrix form:

$$Q_0(\boldsymbol{\theta}) = (\mathbf{Y} - \boldsymbol{\Psi}\boldsymbol{\theta})^T (\mathbf{Y} - \boldsymbol{\Psi}\boldsymbol{\theta}) + \lambda_0 \boldsymbol{\theta}^T \mathbf{R} \boldsymbol{\theta},$$

where $\mathbf{R}$ is a $J \times J$ matrix with entries $R_{ij} = \int \ddot{\psi}_i(t) \ddot{\psi}_j(t) dt$ and $\boldsymbol{\Psi}$ is an $n \times J$ matrix with entries $\Psi_{ij} = \psi_j(t_i)$. Taking the derivative $Q_0(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$, one can obtain

$$\widehat{\boldsymbol{\theta}} = (\boldsymbol{\Psi}^T \boldsymbol{\Psi} + \lambda_0 \mathbf{R})^{-1} \boldsymbol{\Psi}^T \mathbf{Y}.$$

Thus, the estimated trajectory for $X(t)$ and the derivative $\dot{X}(t)$ can be expressed as $\hat{X}(t) = \boldsymbol{\psi}(t)^T \hat{\boldsymbol{\theta}}$ and $\hat{\dot{X}}(t) = \dot{\boldsymbol{\psi}}(t)^T \hat{\boldsymbol{\theta}}$.

Because the estimated derivatives for gene $\ell$ at observed time points are essentially correlated across time, equation (7) should take this correlation into consideration and be replaced by

$$\frac{1}{n} (\hat{\dot{\mathbf{X}}}_\ell - \boldsymbol{\Phi}_\ell^T \boldsymbol{\beta}_\ell)^T \big[ \widehat{\mathrm{Cov}}(\hat{\dot{\mathbf{X}}}) \big]^{-1} (\hat{\dot{\mathbf{X}}}_\ell - \boldsymbol{\Phi}_\ell^T \boldsymbol{\beta}_\ell),$$

where the estimated variance-covariance matrix of the derivatives $\widehat{\mathrm{Cov}}(\hat{\dot{\mathbf{X}}})$ can be obtained with the delta method,

$$\widehat{\mathrm{Cov}}(\hat{\dot{\mathbf{X}}}) = \dot{\boldsymbol{\Psi}}^T \widehat{\mathrm{Cov}}(\hat{\boldsymbol{\theta}}) \dot{\boldsymbol{\Psi}} = \hat{\sigma}_s^2 \dot{\boldsymbol{\Psi}}^T (\boldsymbol{\Psi}^T \boldsymbol{\Psi} + \lambda_0 \mathbf{R})^{-1} \boldsymbol{\Psi}^T \boldsymbol{\Psi} (\boldsymbol{\Psi}^T \boldsymbol{\Psi} + \lambda_0 \mathbf{R})^{-1} \dot{\boldsymbol{\Psi}}, \tag{7.14}$$

in which $\hat{\dot{\mathbf{X}}} = (\hat{\dot{X}}(t_1), \ldots, \hat{\dot{X}}(t_n))^T$, $\dot{\boldsymbol{\Psi}}$ is a $n \times J$ matrix with entries $\dot{\psi}_j(t_i)$ and $\hat{\sigma}_s^2$ can be obtained by computing the sample variance of the residuals $\mathbf{e}_s = \mathbf{Y} - \boldsymbol{\Psi} \hat{\boldsymbol{\theta}}$.

In fact, as one reviewer suggests, our proposed algorithm given at the end of Section 2.5 is still applicable by simply letting $[\widehat{\mathrm{Cov}}(\hat{\dot{\mathbf{X}}})]^{-1} = \mathbf{L}_\ell^T \mathbf{L}_\ell$ be the Cholesky decomposition of the inverse variance-covariance matrix and then pre-conditioning both $\hat{\dot{\mathbf{X}}}_\ell$ and $\boldsymbol{\Phi}_\ell^T$ with $\mathbf{L}_\ell$. Consequently, equation (12) becomes

$$\widehat{\boldsymbol{\beta}}_\ell = \frac{1}{n}\left(\frac{1}{n}\boldsymbol{\Phi}_\ell \mathbf{L}_\ell^T \mathbf{L}_\ell \boldsymbol{\Phi}_\ell^T + \gamma \mathbf{V}_\ell + \mathbf{W}_\ell^{(0)} + \lambda_I \boldsymbol{\Omega}_\ell\right)^{-1} \boldsymbol{\Phi}_\ell \mathbf{L}_\ell^T \mathbf{L}_\ell \hat{\dot{\mathbf{X}}}_\ell.$$

## 7.3 Application

We consider a data set of 20 Drosophila melanogaster genes involved in the muscle development during the embryonic stage (see Bar-Joseph (2004) for details). The time-course gene expressions are measured at 30 time points in the embryonic stage (Arbeitman et al., 2002).

The time-varying directed gene regulation network of these 20 genes are modeled using the nonlinear ODE model (7.1). The time-varying regulation functions $f_{g\ell}(X_g)$ in (7.1) for each of those 20 genes are estimated in two steps. In the first step, we obtain the estimate for the trajectory of each gene and its derivatives using the smoothing spline method, as introduced in Section 2.7. In the second step, we treat the derivative estimates for each gene as the response and all genes' trajectory estimates as the covariates in ODE model (7.1). We then estimate the basis coefficients for each regulation function via (7.12). The smoothing parameter $\gamma$ and the sparsity parameter $\lambda$ are both determined simultaneously using AICc criterion. The smoothing parameter $\gamma$ is chosen from four candidate values: $10, 10^{-1}, 10^{-3}$ and $10^{-5}$. The sparsity parameter $\lambda$ is selected from five candidate values: $10, 1, 10^{-1}$ and $10^{-2}$. Since the results are not sensitive to specific values of the number of basis functions, $K_{g\ell}$, and the identifiability parameter, $\lambda_I$, we set their values to be $K_{g\ell} = 5$ and $\lambda_I = 10^4$ to ease the computation.

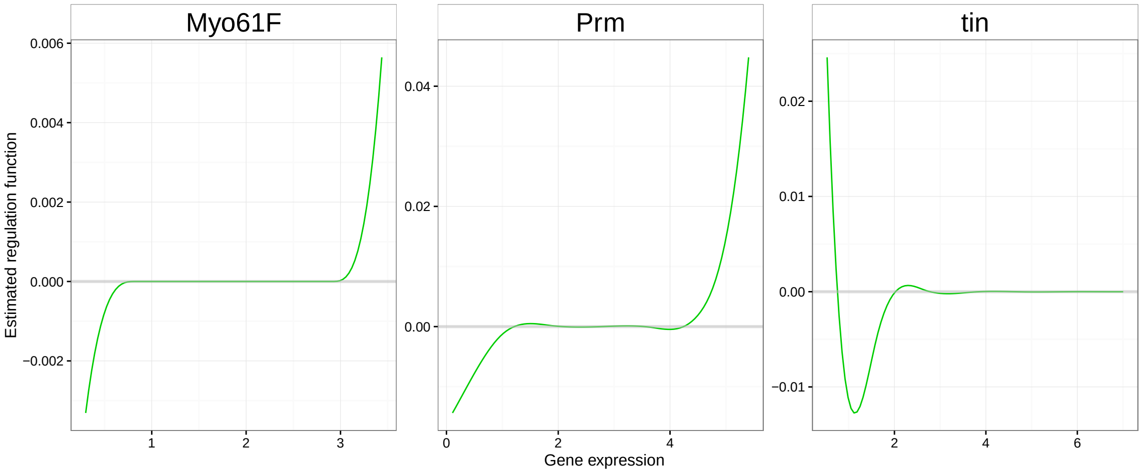

Figure 7.2: Estimated regulation functions on gene *Myo31DF* based in the ODE model (7.1). Three regulatory genes, i.e., *Prm*, *tin* and *Myo61DF* are selected out of 20 genes. All the regulation functions of the rest 17 genes are estimated to be strictly zero during the whole embryonic stage.

Figure 7.2 shows the estimated regulation functions for gene *Myo31DF*. It can be seen that 3 out of 20 genes are selected, which means that the regulation functions of the other 17 genes are estimated to be strictly zero during the entire embryonic stage. Those three estimated regulation functions shown in Figure 7.2 all have non-linear trends and show local sparsity to some extent.

To check whether our finding for gene *Myo31DF* makes biology sense, we conduct a literature search for studies on gene interactions using the Drosophila Interactions Database (Murali et al., 2011) and the GeneMANIA tool (Warde-Farley et al., 2010). We find evidences in the literature about all three regulatory genes *Myo61F*, *Prm*, and *tin* on gene *Myo31DF*. For instance, Hozumi et al. (2006) suggested that both *Myo61F* and *Myo31DF* played a crucial role in generating left-right asymmetry of the embryonic gut. They found that *Myo31DF* was required in the hindgut epithelium for normal embryonic handedness and the overexpression of *Myo61F* reversed the handedness of the embryonic gut, and its knockdown also caused a left-right patterning defect. These two unconventional myosin I proteins might have antagonistic functions in left-right patterning. The results obtained from our analysis match these insights. For instance, Figure 7.2 shows that gene *Myo61F* only regulates gene *Myo31DF* when its expression level is either less than 1 or greater than 2. Thus, either the knockdown or overexpression of *Myo61F* will cause a left-right patterning defect. In addition, Lewis et al. (2005), Ruby et al. (2007), Ruby et al. (2007) and Kheradpour et al. (2007) suggested that gene *Prm* and gene *Myo31DF* shared two common miRNAs, i.e., *mir-iab-4* and *mir-999*. As for gene *tin*, even though there was no direct evidence showing its regulation effect on *Myo31DF*, Fu et al. (1997) found out that *tin* was critical in determining the patterning of the Drosophila heart. Because of gene *Myo31DF*'s role in generating the left-right asymmetry gut, our hypothesis is that *tin* regulates *Myo31DF* to insure the left-right asymmetry formation in the heart. This hypothesis needs to be further investigated in real genetic studies.

Once the regulation functions for all 20 genes are estimated, we can visualize the whole GRN at any given time point. Figure 7.3 shows the estimated GRN at different selected time points during the embryonic stage. One important feature of the estimated GRN is that the regulation effects between genes are time-varying. For example, *Prm* regulates *sls* at the beginning of the embryonic stage, i.e., $t = 3h$, however, *Mef2* replace *Prm*'s role in regulating *sls* in the middle stage. In addition, from the whole network point of view, we observe that genes interact with each other more frequently in the beginning than in the middle or at the end of the embryonic stage. Finally, we find some strong regulators such as *Mef2*, *Myo61F*, *Prm* and *Mhc*, which act as hubs in our estimated GRN.

In Figure 7.3, we highlight those interactions that have been verified in the literature. A red arrow indicates the corresponding directed regulation effect between genes has been verified; a green arrow means the corresponding gene-to-gene interaction has been discovered before but the exactly direction is unclear; and a black arrow means the corresponding interaction has not been found so far. Most regulation effects estimated using our method have been verified previously. Those regulation effects that have not been discovered may be candidate hypotheses for future investigation. It is worth mentioning that the total number of known interactions in the literature is 158 out of 400 possible interactions. In other words, the background interaction rate is 39.5%(=158/400). Using our method, wIt is woe estimate 67 interactions, 58 of which are verified in the literature. The discovery rate for our method is 86.6%(=58/67), which is more than twice the background interaction rate.

Another very important feature of our estimated GRN shown in Figure 7.3 is that the estimated network is sparsely connected. In other words, only a limited number of genes regulate a target gene. Table 7.1 displays a complete list of estimated regulatory genes for all genes. The number of regulatory genes ranges from 2 to 6 with an average 3.35. Furthermore, we prioritize those selected regulatory genes based on their estimated signal strength. The signal strength is defined using the functional $L_2$ norm of the estimated regulation functions in the entire time domain considered. For example, for gene *Actn*, *Mef2* is a stronger regulator in the whole time interval compared to *Prm* and *tin*, as shown in the first row of Table 7.1.

Table 7.1: The regulatory genes for all 20 genes selected by our method. The regulatory genes are sorted by their overall regulation effect on the corresponding target gene. For example, *Mef2* has the largest the overall regulation effect on *Actn* in comparison with *Prm* and *tin*.

| Target Gene | Regulatory Genes |
|---|---|
| Actn | Mef2 Prm tin |
| dpp | Mef2 Prm flw |
| eve | Myo61F Mef2 srp eve |
| fln | tin Prm Actn Mhc Msp300 flw |
| flw | Myo61F Prm |
| how | sls Mlc1 fln |
| lmd | srp Myo61F flw |
| Mef2 | Myo61F up flw lmd |
| Mhc | Msp300 |
| Mlc1 | Msp300 Prm tin |
| Msp300 | tin up Mlc1 Prm Msp300 |
| Myo31DF | Prm tin Myo61F |
| Myo61F | Msp300 tin fln |
| Prm | Msp300 |
| sls | Prm tin how Mef2 |
| srp | Mef2 Prm flw eve twi Msp300 |
| tin | Mef2 lmd Myo61F flw twi |
| twi | Mef2 srp |
| up | Msp300 Prm Mlc1 flw |
| wg | Mef2 twi |

## 7.4 Simulation

In this section, we assess the performance of our fSCAD method using a simulation study. To mimic the real gene regulation process, we use the ODE model for the target gene *Myo31DF* estimated from the real data analysis to generate the true trajectory of the target gene as follows:

$$X_0(t) = \int_0^t \dot{X}_0(\tau)d\tau = \int_0^t \sum_{i=1}^{20} f_i(X_i(\tau))d\tau, \tag{7.15}$$

where $\dot{X}_0(\tau)$ denotes the derivative of the expression for the target gene and $X_i(\tau)$ is the expression function of gene $i$ at time $\tau$, i.e., what was observed empirically at $\tau$. Here we take $\tau \in \{0, 1, 2, \ldots, 23\}$. The three true regulation functions $f_i(X_i), i = 1, 2, 3$ are the same as the estimated regulation functions from the real data shown in Figure 2, and all the remaining 17 true regulation functions are strictly zero in the whole interval. That is, $f_i(X_i) \equiv 0, i = 4, \ldots, 20$. For simplicity, we use $X_1$, $X_2$ and $X_3$ to denote gene *Prm*, *tin* and *Myo61F*, respectively. In addition, we refer to genes with non-zero regulation functions as regulatory genes and genes with strictly-zero regulation functions as non-regulatory genes. To account for the estimation error in estimating the derivative function $\dot{X}_0(t)$ in the first step, as one reviewer suggests, we generate the noisy data by adding a white noise $\epsilon$ to the true $X_0(t)$. The noise level is controlled by the noise-to-signal ratio $\rho$ as $\epsilon \overset{i.i.d}{\sim} N(0, \rho\sigma_x^2)$, where $\sigma_x$ is the sample standard deviation of the true trajectory $X_0(t)$ empirically observed at $\tau \in \{0, 1, 2, \ldots, 23\}$.

We estimate ODE model (7.1) using the group Lasso method and the following three methods:

**Locally sparse method**: the loss function defined in (7.7) with both fSCAD penalty and roughness penalty;

**Smoothing spline method**: the loss function defined in (7.7) with roughness penalty only, i.e., $\lambda = 0$;

**Linear fSCAD method**: the loss function defined in (7.7) with the fSCAD penalty and a very large roughness penalty. More specifically, we fix $\gamma = 100$ to force the estimated regulation functions to be almost linear.

For the locally sparse method and the smoothing spline method, the smoothing parameter $\gamma$ is chosen from four candidate values: $10, 10^{-1}, 10^{-3}$ and $10^{-5}$ using AICc. For both the locally sparse method and the linear fSCAD method, the sparsity parameter $\lambda$ is selected from five candidate values: $10, 1, 10^{-1}$ and $10^{-2}$ using AICc. In addition, the number of basis function $K_{g\ell}$ and the identifiability parameter $\lambda_I$ remain the same as in the real data analysis, i.e., $K_{g\ell} = 5$ and $\lambda_I = 10^4$. For the group Lasso method, we use the 5-fold cross-validation to choose the penalty parameter.

We access the variable-selection accuracy for each method using the false negative error (FN) and the false positive error (FP), which are defined in the gene regulation scenario as follows:

$$\text{FN} = \frac{\#\text{ of incorrectly estimated non-regulatory genes}}{\#\text{ of all true regulatory genes}},$$

$$\text{FP} = \frac{\#\text{ of incorrectly estimated regulatory gene}}{\#\text{ of all estimated regulatory gene}}.$$

The simulation is repeated for 100 times and the results are presented in Table 7.2. First of all, we can see that the locally sparse method yields the lowest FN error among all the methods given the same noise-to-signal level. To be more specific, when the noise-to-signal level is only 1%, the locally sparse method only misselects 25% of all the estimated regulatory genes. In comparison, with no sparsity penalty, smoothing spline method is not able to produce a parsimony model such that the FP error keeps at 85% even when the noise-to-signal level is only 1%. The group Lasso method also fails to detect the true regulatory genes and over 90% of the estimated regulatory genes are actually non-regulatory genes in our simulation settings. This is because the group Lasso method cannot detect local sparsity and always penalize the regulation function in the whole domain. In addition, the linear fSCAD yields the second lowest FP error because of the fSCAD penalty, however, as the large roughness penalty forces the regulation functions to be linear, the resulting model are not as parsimony as the true model is. On the other hand, the smoothing spline method does not make any FN error due to the fact that it estimates all the genes as regulatory genes. Both the linear fSCAD and the locally sparse method yield similar FN errors, which indicates that they seldom select true non-regulatory genes as regulatory genes. Lastly, the performance of group Lasso is still very poor, because over 85% of the estimated non-regulatory genes are actually true regulatory genes even when the noise-to-signal ratio level is 1%.

Table 7.2: The means and standard deviations (SD) of the false positive errors (FP) and the false negative errors (FN) of the four methods in 100 simulation replicates. Here $\rho$ represents the noise-to-signal ratio in the simulated data.

| | | FP | | FN | |
|---|---|---|---|---|---|
| Method | $\rho$ | Mean | SD | Mean | SD |
| Locally Sparse | 1% | 25.0% | 0.0% | 0.0% | 0.0% |
| | 5% | 30.0% | 11.5% | 9.0% | 15.6% |
| Smoothing Spline | 1% | 85.0% | 0.0% | 0.0% | 0.0% |
| | 5% | 85.0% | 0.0% | 0.0% | 0.0% |
| Linear fSCAD | 1% | 63.9% | 5.5% | 4.8% | 11.7% |
| | 5% | 63.9% | 5.5% | 4.8% | 11.7% |
| Group Lasso | 1% | 94.9% | 8.2% | 88.1% | 19.5% |
| | 5% | 99.5% | 3.2% | 99.3% | 4.7% |

Next, we assess the each method's ability to detect the sparsity of the regulation functions. For each regulation function, we divide the corresponding entire interval into 100 subintervals equally. The specificity for each estimated regulation function in one simulation run is calculated as the percentage of those strictly-zero subintervals which are falsely estimated as non-zero. Then we take the average specificity for each method across all regulation functions in 100 simulation replicates. The complete results are shown in Table 7.4. We find that the locally sparse method yields the lowest specificity among all methods considered. Less than 10% of the true strictly-zero subintervals are incorrectly estimated as non-zero even when the noise-to-signal ratio is high. The smoothing spline method cannot produce sparse regulation function estimations, therefore its specificity is always 1. With a very large roughness penalty, the linear fSCAD method forces the estimated regulation function to be close to linear forms, and this method fails to detect the change points between zero regions and non-zero regions. Therefore, the linear fSCAD method yields the second highest specificity among all four methods. With the group Lasso penalty, the group Lasso method tends to shrink the entire regulation function to zero and the corresponding specificity is about 16%. On the other hand, we find that the the group Lasso method always shrinks those three true regulation function into strictly zero even when the noise-to-ratio level is only 1%. In constrast, the locally sparse method estimated regulation function are much closer to the the true regulation functions. The average of estimated regulation functions compared to the true regulation functions along with the experimental point-wise confidence bands using the locally sparse method are shown in Figure 7.4 and 7.5.

| Method | $\rho$ | Mean | Standard Deviation |
|---|---|---|---|
| Locally sparse | 1% | 6.8% | 1.0% |
| | 5% | 8.4% | 2.9% |
| Smoothing spline | 1% | 100.0% | 0.0% |
| | 5% | 100.0% | 0.0% |
| Linear fSCAD | 1% | 24.3% | 2.6% |
| | 5% | 30.5% | 5.0% |
| Group Lasso | 1% | 17.9% | 13.3% |
| | 5% | 16.0% | 5.4% |

Table 7.3: The mean and standard deviation of the false positive rates using four methods in 100 simulation replicates. Here $\rho$ represents the noise-to-signal ratio in the simulated data.

Although the main focus of our method is to estimate time-varying directed GRN, we also compare the prediction performance of our proposed method with the group Lasso method. More specifically, we hold out the last observation, i.e., $X_i(23), i = 0, 1, \ldots, 20$, for all the genes in the network and estimate the regulation functions using the first 23 observations only. Then we use our method and the group Lasso method to estimate the gene regulation functions in the ODE model (7.1). We then use the estimated ODE model (7.1) to predict the value of $X_0(\tau)$ at $\tau = 23$ and compute the squared prediction error. We also compare their prediction performances with two other methods: the constant expression method and an autoregressive model, AR1. The constant expression model simply takes the sample mean from previously observed trajectories values as the prediction value. The AR1 method is fitted using the maximum likelihood approach. The detailed results are presented in Table 7.4. Table 7.4 shows that the locally sparse method yields the lowest mean squared prediction error among all methods, which is only about 10% compared to the group Lasso method and the AR1 model.

| Method | $\rho$ | Mean | Standard Deviation |
|---|---|---|---|
| Locally Sparse | 1% | 0.23 | 0.15 |
| | 5% | 0.96 | 1.27 |
| Group Lasso | 1% | 6.72 | 0.60 |
| | 5% | 6.80 | 2.97 |
| AR1 | 1% | 8.66 | 0.62 |
| | 5% | 8.70 | 3.00 |
| Constant Expression | 1% | 568.80 | 1.07 |
| | 5% | 568.45 | 5.37 |

Table 7.4: The mean and standard deviation of squared prediction errors using four methods in 100 simulation replicates. Here $\rho$ represents the noise-to-signal ratio in the simulated data.

In summary, our proposed method can correctly select the true regulatory genes without misselecting those true non-regulatory genes in the ODE model compared to other alternative methods. In addition, it can also successfully identify the strictly-zero subregions of all regulation functions. Finally, it outperforms popular method such as group Lasso in term of the forward prediction accuracy.

## 7.5 Summary

ODE models are widely used to model a dynamical system in many fields such as biology, economics, and physics. In this chapter, we use a high-dimensional nonlinear ODE model to describe a time-varying direct GRN. It is worth mentioning, as one reviewer suggests, the ODE model itself is time-stationary in the sense that all the regulation functions are deterministic functions of the regulatory gene expressions, but the edges may implicitly emerge or disappear over time, and the strength of the edge may vary with time, because the expressions of regulatory genes change with time. We propose the fSCAD method to estimate the unknown regulation functions in the high-dimensional ODE model from the time-course gene expression data.

In the real data application, we show that our method can simultaneously detect the significant regulatory genes, estimate the nonlinear regulation functions without any parametric assumption, and identify the intervals with no regulation effects. The resulting GRN with the estimated regulation functions has many potential implications. First, based on the estimated edges and their corresponding directions, new hypotheses for gene regulation mechanism can be proposed as candidate relationships for future investigations. For those edges that have already been verified in the literature, we can prioritize them based on the estimated signal strength. In addition, when no prior knowledge about the direction of the regulation effect is available, our method can be a good starting point for the direction detection. Furthermore, our method can not only suggest potentially unverified regulation relationships between genes, but also give clues in which time periods the regulation effects are most likely to be detected. This advantage can greatly facilitate the future biology experiment designs for detecting gene regulation effects.

Furthermore, our simulation study shows that our method is able to estimate the true regulation functions under different levels of noises in the data more accurately in comparison with the group Lasso method. Finally, our method avoids solving the ODEs numerically, making it computational efficient and feasible in the high-dimensional context. Our method can be extended to model and estimate other high-dimensional directed networks from time-course or longitudinal data.

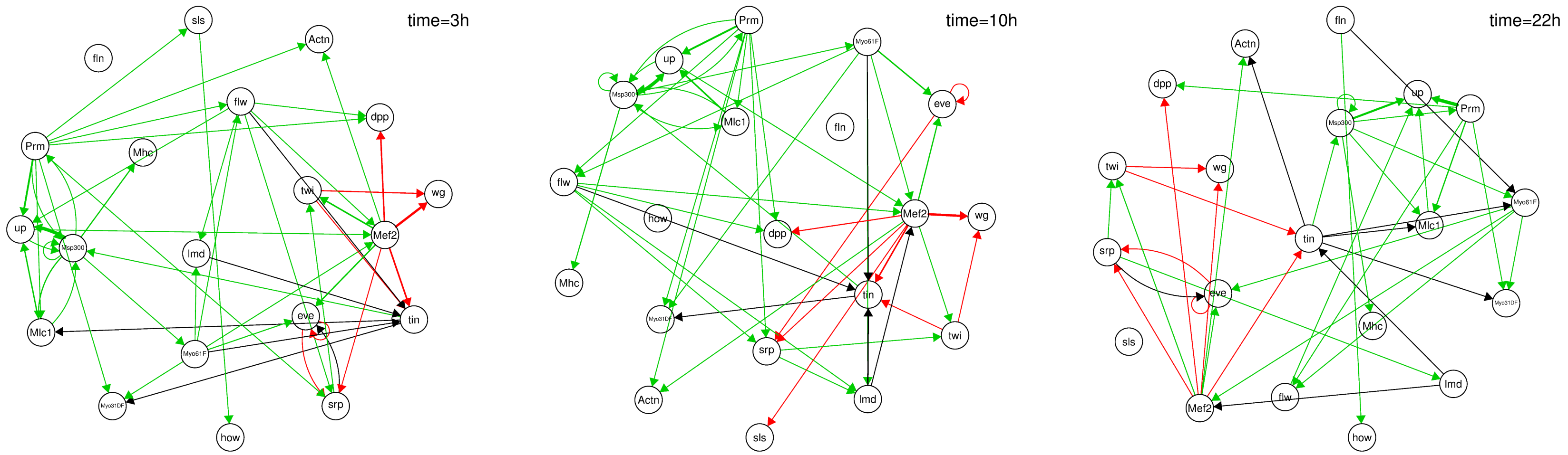

Figure 7.3: The estimated time-varying directed gene regulation network of 20 genes in the muscle development pathway at three time points during the embryonic stage. The connection lines represent the existence of regulation effects between genes. The line color indicates whether the regulations have been verified in the literature: red (verified regulation effects), green (verified gene-to-gene interactions) and black (unverified regulation effects). Details can be found in the excel file at `http://www.sfu.ca/~nyunlong/research/grn/`. This figure is generated using the qgraph package (Epskamp et al., 2012).

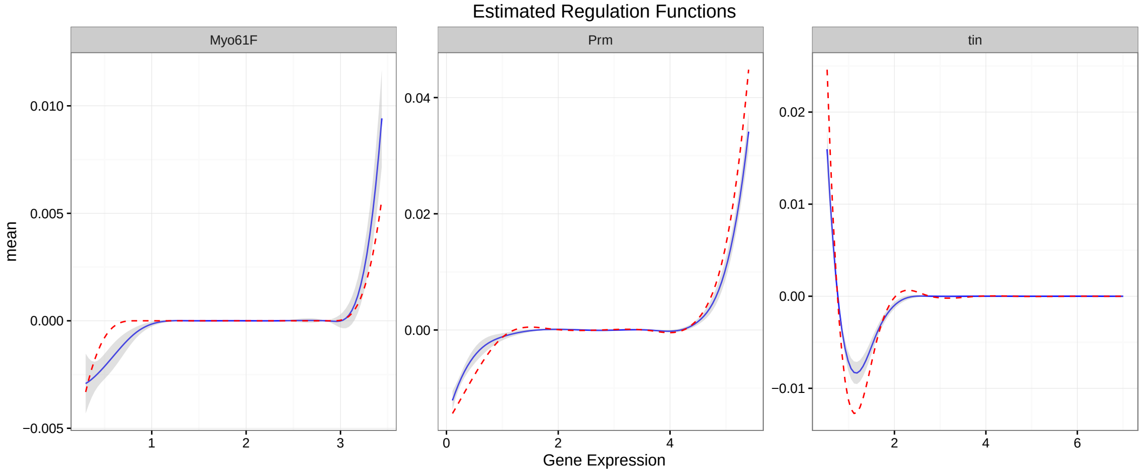

Figure 7.4: Estimated regulation functions from the simulated data with the noise-to-signal ratio of the simulated data $\rho = 1\%$ using the locally sparse method. The dashed red and solid blue lines represent the true regulation functions and the mean of the estimated regulation functions in 100 simulation replicates. The grey bands denote the pointwise 95% confidence interval of the estimated regulation functions.

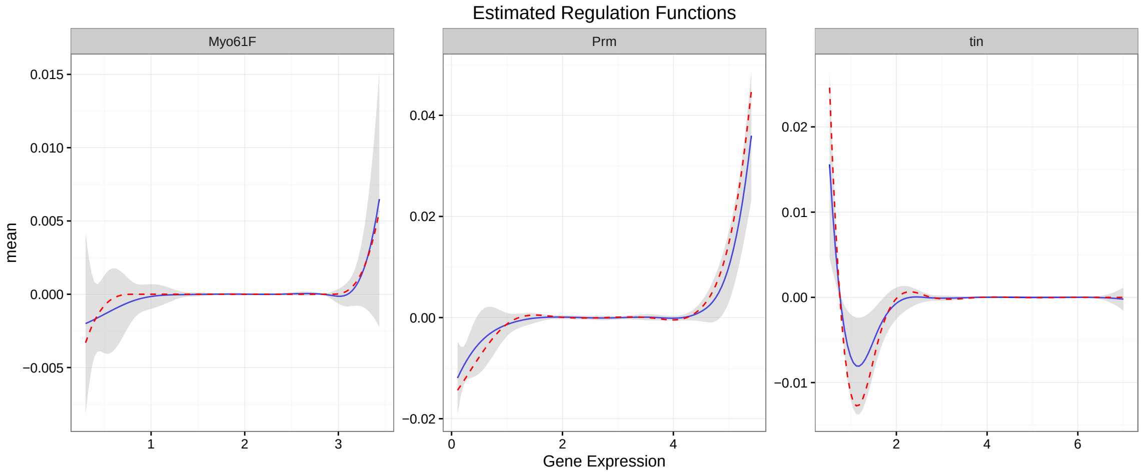

Figure 7.5: Estimated regulation functions from the simulated data with the noise-to-signal ratio of the simulated data $\rho = 5\%$ using the locally sparse method. The dashed red and solid blue lines represent the true regulation functions and the mean of the estimated regulation functions in 100 simulation replicates. The grey bands denote the pointwise 95% confidence interval of the estimated regulation functions.

# Bibliography

Ait-Saïdi, A., F. Ferraty, R. Kassa, and P. Vieu (2008). Cross-validated estimations in the single-functional index model. *Statistics 42*(6), 475–494.

Arbeitman, M. N., E. E. Furlong, F. Imam, E. Johnson, B. H. Null, B. S. Baker, M. A. Krasnow, M. P. Scott, R. W. Davis, and K. P. White (2002). Gene expression during the life cycle of drosophila melanogaster. *Science 297*(5590), 2270–2275.

Bair, E., T. Hastie, D. Paul, and R. Tibshirani (2006). Prediction by supervised principal components. *Journal of the American Statistical Association 101*(473), 119–137.

Bar-Joseph, Z. (2004). Analyzing time series gene expression data. *Bioinformatics 20*(16), 2493–2503.

Bosq, D. (2000). *Linear Processes in Function Spaces: Theory and Applications*. Springer-Verlag, New York.

Cao, J. and H. Zhao (2008). Estimating dynamic models for gene regulation networks. *Bioinformatics 24*(14), 1619–1624.

Cardot, H., R. Faivre, and M. Goulard (2003). Functional approaches for predicting land use with the temporal evolution of coarse resolution remote sensing data. *Journal of Applied Statistics 30*(10), 1185–1199.

Cardot, H., F. Ferraty, and P. Sarda (1999). Functional linear model. *Statistics & Probability Letters 45*(1), 11–22.

Chen, D., P. Hall, and H.-G. Müller (2011). Single and multiple index functional regression models with nonparametric link. *The Annals of Statistics 39*(3), 1720–1747.

Chen, J. and H. Wu (2008). Estimation of time-varying parameters in deterministic dynamic models. *Statistica Sinica 18*(3), 987–1006.

Chen, K. and J. Lei (2015). Localized functional principal component analysis. *Journal of the American Statistical Association 110*(511), 1266–1275.

Clark, N., M. Gautam, W. Wayne, D. Lyons, G. Thompson, and B. Zielinska (2007). Heavy-duty vehicle chassis dynamometer testing for emissions inventory, air quality modeling, source apportionment and air toxics emissions inventory. *Coordinating Research Council, incorporated.*

Craven, P. and G. Wahba (1978). Smoothing noisy data with spline functions. *Numerische Mathematik 31*(4), 377–403.

Dai, X., H.-G. Müller, and W. Tao (2017). Derivative principal component analysis for representing the time dynamics of longitudinal and functional data. *arXiv preprint arXiv:1707.04360*.

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. (2001). *A Practical Guide to Splines.* Applied Mathematical Sciences. Springer, New York.

Du, P., G. Cheng, and H. Liang (2012). Semiparametric regression models with additive nonparametric components and high dimensional parametric components. *Computational Statistics & Data Analysis 56*(6), 2006–2017.

Du, P. and X. Wang (2014). Penalized likelihood functional regression. *Statistica Sinica*, 1017–1041.

Epskamp, S., A. O. J. Cramer, L. J. Waldorp, V. D. Schmittmann, and D. Borsboom (2012). qgraph: Network visualizations of relationships in psychometric data. *Journal of Statistical Software 48*(4), 1–18.

Fan, J. and I. Gijbels (1995a). Data-driven bandwidth selection in local polynomial fitting: variable bandwidth and spatial adaptation. *Journal of the Royal Statistical Society. Series B 57*(2), 371–394.

Fan, J. and I. Gijbels (1995b). *Local Polynomial Modelling and Its Applications.* Chapman and Hall, London.

Fan, J. and R. Li (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association 96*(456), 1348–1360.

Fan, J., H. Peng, et al. (2004). Nonconcave penalized likelihood with a diverging number of parameters. *The Annals of Statistics 32*(3), 928–961.

Fanaee-T, H. and J. Gama (2014). Event labeling combining ensemble detectors and background knowledge. *Progress in Artificial Intelligence 2*(2-3), 113–127.

Fu, Y., P. Ruiz-Lozano, and S. M. Evans (1997). A rat homeobox gene, rnkx-2.5, is a homologue of the tinman gene in drosophila and is mainly expressed during heart development. *Development Genes and Evolution 207*(5), 352–358.

Fukunaga, K. and W. L. Koontz (1970). Representation of random processes using the finite karhunen-loeve expansion. *Information and Control 16*(1), 85–101.

Gasser, T. and H.-G. Mülller (1984). Estimating regression functions and their derivatives by the kernel method. *Scandinavian Journal of Statistics 11*(3), 171–185.

Ghalanos, A. and S. Theussl (2015). *Rsolnp: General Non-linear Optimization Using Augmented Lagrange Multiplier Method.* R package version 1.16.

Hall, P. and G. Hooker (2015). Truncated linear models for functional data. *Journal of the Royal Statistical Society: Series B 78*(3), 637–653.

Hall, P. and J. L. Horowitz (2007). Methodology and convergence rates for functional linear regression. *The Annals of Statistics 35*(1), 70–91.

Hall, P., H.-G. Müller, and J.-L. Wang (2006). Properties of principal component methods for functional and longitudinal data analysis. *The Annals of Statistics 34*(3), 1493–1517.

Hanneke, S., W. Fu, and E. P. Xing (2010). Discrete temporal models of social networks. *Electronic Journal of Statistics 4*, 585–605.

Hozumi, S., R. Maeda, K. Taniguchi, M. Kanai, S. Shirakabe, T. Sasamura, P. Spéder, S. Noselli, T. Aigaki, R. Murakami, et al. (2006). An unconventional myosin in drosophila reverses the default handedness in visceral organs. *Nature 440*(7085), 798–802.

Huang, J. Z., H. Shen, and A. Buja (2009). The analysis of two-way functional data using two-way regularized singular value decompositions. *Journal of the American Statistical Association 104*(488), 1609–1620.

Jensen, F. V. (1996). *An Introduction to Bayesian Networks.* University College London Press, London.

Jiang, C.-R. and J.-L. Wang (2011, feb). Functional single index models for longitudinal data. *The Annals of Statistics 39*(1), 362–388.

Kheradpour, P., A. Stark, S. Roy, and M. Kellis (2007). Reliable prediction of regulator targets using 12 drosophila genomes. *Genome Research 17*(12), 1919–1931.

Kolar, M., L. Song, A. Ahmed, and E. P. Xing (2010). Estimating time-varying networks. *Annals of Applied Statistics 4*(1), 94–123.

Kolar, M. and E. P. Xing (2009). Sparsistent estimation of time-varying discrete markov random fields. *arXiv:0907.2337*.

Laubenbacher, R. and B. Stigler (2004). A computational algebra approach to the reverse engineering of gene regulatory networks. *Journal of Theoretical Biology 229*(4), 523–537.

Lawson, C. L. and R. J. Hanson (1974). *Solving Least Squares Problems.* Englewood Cliffs, Prentice-Hall, NJ.

Lewis, B., C. Burge, and D. Bartel (2005). Conserved seed pairing, often flanked by adenosines, indicates that thousands of human genes are microrna targets. *Cell 120*(1), 15.

Li, G., H. Shen, and J. Z. Huang (2016). Supervised sparse and functional principal component analysis. *Journal of Computational and Graphical Statistics 25*(3), 859–878.

Li, G., D. Yang, A. B. Nobel, and H. Shen (2016). Supervised singular value decomposition and its asymptotic properties. *Journal of Multivariate Analysis 146*, 7–17.

Li, J., C. Huang, Z. Hongtu, and A. D. N. Initiative (2017). A functional varying-coefficient single-index model for functional response data. *Journal of the American Statistical Association 112*(519), 1169–1181.

Li, Y., N. Wang, and R. J. Carroll (2013). Selecting the number of principal components in functional data. *Journal of the American Statistical Association 108*(504), 1284–1294.

Lin, Z., J. Cao, L. Wang, and H. Wang (2016). Locally sparse estimator for functional linear regression models. *Journal of Computational and Graphical Statistics doi:10.1080/10618600.2016.1195273*, 1–41.

Lin, Z., L. Wang, and J. Cao (2016). Interpretable functional principal component analysis. *Biometrics 72*(3), 846–854.

Liu, B. and H.-G. Müller (2009). Estimating derivatives for samples of sparsely observed functions, with application to online auction dynamics. *Journal of the American Statistical Association 104*(486), 704–717.

Lu, T., H. Liang, H. Li, and H. Wu (2011). High-dimensional ODEs coupled with mixed-effects modeling techniques for dynamic gene regulatory network identification. *Journal of the American Statistical Association 106*(496), 1242–1258.

Luscombe, N. M., M. M. Babu, H. Yu, M. Snyder, S. A. Teichmann, and M. Gerstein (2004). Genomic analysis of regulatory network dynamics reveals large topological changes. *Nature 431*(7006), 308–312.

Ma, S. (2014, oct). Estimation and inference in functional single-index models. *Annals of the Institute of Statistical Mathematics 68*(1), 181–208.

Mas, A. (2002). Weak convergence for the covariance operators of a hilbertian linear process. *Stochastic Processes and Their Applications 99*(1), 117–135.

Mehra, S., W.-S. Hu, and G. Karypis (2004). A boolean algorithm for reconstructing the structure of regulatory networks. *Metabolic Engineering 6*(4), 326–339.

Müller, H.-G. (2005). Functional modelling and classification of longitudinal data. *Scandinavian Journal of Statistics 32*(2), 223–240.

Müller, H.-G. and U. Stadtmüller (2005). Generalized functional linear models. *The Annals of Statistics 33*(2), 774–805.

Murali, T., S. Pacifico, J. Yu, S. Guest, G. G. Roberts, and R. L. Finley (2011). Droid 2011: a comprehensive, integrated resource for protein, transcription factor, rna and gene interactions for drosophila. *Nucleic Acids Research 39*(suppl 1), D736–D743.

Needham, C. J., J. R. Bradford, A. J. Bulpitt, and D. R. Westhead (2007, 08). A primer on learning in bayesian networks for computational biology. *PLOS Computational Biology 3*(8), 1–8.

Nie, Y., L. Wang, and J. Cao (2017). Estimating time-varying directed gene regulation networks. *Biometrics 73*(4), 1231–1242.

Nie, Y., L. Wang, B. Liu, and J. Cao (2018). Supervised functional principal component analysis. *Statistics and Computing 28*(3), 713–723.

Peng, J. and H.-G. Müller (2008). Distance-based clustering of sparsely observed stochastic processes, with applications to online auctions. *The Annals of Applied Statistics 2*(3), 1056–1077.

Peng, J. and D. Paul (2009). A geometric approach to maximum likelihood estimation of the functional principal components from sparse longitudinal data. *Journal of Computational and Graphical Statistics 18*(4), 995–1015.

Pezzulli, S. (1993). Some properties of smoothed principal components analysis for functional data. *Computational Statistics 8*(1), 1–16.

Phillips, S. M., L. G. Bandini, D. V. Compton, E. N. Naumova, and A. Must (2003). A longitudinal comparison of body composition by total body water and bioelectrical impedance in adolescent girls. *The Journal of Nutrition 133*(5), 1419–1425.

Pollard, D. (1991). Asymptotics for least absolute deviation regression estimators. *Econometric Theory 7*(2), 186–199.

Ramsay, J., G. Hooker, and S. Graves (2009). *Functional data analysis with R and MATLAB*. Springer, New York.

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

Ramsay, J. O. and B. W. Silverman (2005). *Functional data analysis*. Second Edition. Springer-Verlag, New York.

Ratcliffe, S. J., G. Z. Heller, and L. R. Leader (2002). Functional data analysis with application to periodically stimulated foetal heart rate data. ii: Functional logistic regression. *Statistics in medicine 21*(8), 1115–1127.

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

Ruby, J. G., C. H. Jan, and D. P. Bartel (2007). Intronic microrna precursors that bypass drosha processing. *Nature 448*(7149), 83–86.

Ruby, J. G., A. Stark, W. K. Johnston, M. Kellis, D. P. Bartel, and E. C. Lai (2007). Evolution, biogenesis, expression, and target predictions of a substantially expanded set of drosophila micrornas. *Genome Research 17*(12), 1850–1864.

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

Song, L., M. Kolar, and E. P. Xing (2009). Keller: estimating time-varying interactions between genes. *Bioinformatics 25*(12), i128–i136.

Steuer, R., J. Kurths, C. O. Daub, J. Weise, and J. Selbig (2002). The mutual information: detecting and evaluating dependencies between variables. *Bioinformatics 18*(2), S231–S240.

Stuart, J. M., E. Segal, D. Koller, and S. K. Kim (2003). A gene-coexpression network for global discovery of conserved genetic modules. *Science 302*(5643), 249–255.

Teicher, H. and Y. S. Chow (1978). *Probability Theory: Independence, Interchangeability, Martingales.* Springer-Verlag, New York.

Thomas, R. (1973). Boolean formalization of genetic control circuits. *Journal of Theoretical Biology 42*(3), 563–585.

Tran, N. M. (2008). *An introduction to theoretical properties of functional principal component analysis.* Ph. D. thesis, Department of Mathematics and Statistics, The University of Melbourne.

Wand, M. P. and M. C. Jones (1994). *Kernel smoothing.* Chapman and Hall/CRC, London.

Warde-Farley, D., S. L. Donaldson, O. Comes, K. Zuberi, R. Badrawi, P. Chao, M. Franz, C. Grouios, F. Kazi, C. T. Lopes, et al. (2010). The genemania prediction server: biological network integration for gene prioritization and predicting gene function. *Nucleic Acids Research 38*(suppl 2), W214–W220.

Wood, S. (2006). *Generalized additive models: an introduction with R.* Chapman and Hall/CRC, London.

Wu, H., T. Lu, H. Xue, and H. Liang (2014). Sparse additive ordinary differential equations for dynamic gene regulatory network modeling. *Journal of the American Statistical Association 109*(506), 700–716.

Yao, F., H.-G. Müller, and J.-L. Wang (2005a). Functional data analysis for sparse longitudinal data. *Journal of the American Statistical Association 100*(470), 577–590.

Yao, F., H.-G. Müller, and J.-L. Wang (2005b). Functional linear regression analysis for longitudinal data. *The Annals of Statistics 33*(6), 2873–2903.

Ye, Y. (1987). *Interior Algorithms for Linear, Quadratic, and Linearly Constrained Non-Linear Programming.* Ph. D. thesis, Department of Earth System Science, Stanford University.

Yuan, M. and Y. Lin (2006). Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B 68*(1), 49–67.

Zambom, A. Z. and M. G. Akritas (2017). NonpModelCheck: An R package for nonparametric lack-of-fit testing and variable selection. *Journal of Statistical Software 77*(10), 1–28.

Zhang, X. L., H. Begleiter, B. Porjesz, W. Wang, and A. Litke (1995). Event related potentials during object recognition tasks. *Brain Research Bulletin 38*(6), 531–538.

Zhou, J., N.-Y. Wang, and N. Wang (2013). Functional linear model with zero-value coefficient function at sub-regions. *Statistica Sinica 23*(1), 25.

Zhou, X.-H., D. K. McClish, and N. A. Obuchowski (2009). *Statistical methods in diagnostic medicine*, Volume 569. John Wiley & Sons.

Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American Statistical Association 101*(476), 1418–1429.

# Appendix A

# Theoretical Results in Section 3.4

We show the proof of Theorem 3.4.1 and 3.4.2 in Section 4 of the main manuscript in details. These two theorem shows that the empirical FPCs can be obtained by minimizing the mean $L^2$ errors to the observed function data $x_i(t)$.

**Theorem 4.4.1.** For any $\tau > 0$, let

$$\hat{\beta}(t) = \arg\min \frac{1}{n}\sum_{i=1}^{n} \Big|\Big|x_i(t) - \alpha(t)\langle\beta, x_i\rangle\Big|\Big|^2 + \tau \int \beta^2(t)dt \tag{A.1}$$

subject to $||\alpha||^2 = 1$, then $\hat{\beta}(t) = c\hat{\phi}_1(t)$, where $\hat{\phi}_1(t)$ is the first empirical eigenfunctions of the sample covariance function $g(s,t) = \frac{1}{n}\sum_{i=1}^{n} x_i(s)x_i(t)$ and $c$ is a constant scale factor.

*Proof.* For given $\alpha(t)$, each $x_i(t)$ can be expressed as $x_i(t) = \langle\alpha, x_i\rangle\alpha(t) + \eta_i(t)$, in which $\eta_i \perp \alpha$, then the loss function

$$\begin{aligned}
&\frac{1}{n}\sum_{i=1}^{n} ||x_i(t) - \alpha(t)\langle\beta, x_i\rangle||^2 + \tau \int \beta^2(t)dt \\
=&\frac{1}{n}\sum_{i=1}^{n} ||(\langle\alpha, x_i\rangle - \langle\beta, x_i\rangle)\alpha(t)||^2 + \tau \int \beta^2(t)dt + \sum_{i=1}^{n}||\eta_i(t)||^2 \\
=&\frac{1}{n}\sum_{i=1}^{n} ||\int(\alpha(t) - \beta(t))x_i(t)dt||^2 + \tau \int \beta^2(t)dt + \sum_{i=1}^{n}||\eta_i(t)||^2,
\end{aligned} \tag{A.2}$$

Note that the last term in (A.2) does not depend on $\beta(t)$. Therefore, minimizing (A.2) is equivalent to minimizing the sum of those first two terms. In addition, we can express both $\alpha(t)$ and $\beta(t)$ using the functional principal components $\boldsymbol{\phi} = (\phi_1(t), \ldots, \phi_K(t))$ obtained by decomposing the sample covariance function $g(s,t) = \frac{1}{n}\sum_{i=1}^{n} x_i(s)x_i(t)$ of the functional data. To show that $\alpha(t)$ and $\beta(t)$ can be expanded using $\boldsymbol{\phi}$, we denote $\alpha(t) = \alpha_1(t) + \alpha_2(t)$ and $\beta(t) = \beta_1(t) + \beta_2(t)$, in which $\alpha_1(t) = \mathbf{a}_1^T\boldsymbol{\phi}$, $\alpha_2(t) \perp \text{span}\{\phi_1, \ldots, \phi_K\}$, $\beta_1(t) = \mathbf{b}_1^T\boldsymbol{\phi}$, and $\beta_2(t) \perp \text{span}\{\phi_1, \ldots, \phi_K\}$. Then $\alpha_2(t) \perp x_i(t)$ and $\beta_2(t) \perp x_i(t)$. The loss function

$$\begin{aligned}
&\frac{1}{n}\sum_{i=1}^{n} || \int (\alpha(t) - \beta(t)) x_i(t) dt||^2 + \tau \int \beta^2(t) dt \\
=&\frac{1}{n}\sum_{i=1}^{n} || \int \alpha_1(t) x_i(t) dt - \int \beta_1(t) x_i(t) dt||^2 + \tau \int \beta_1^2(t) dt + \tau \int \beta_2^2(t) dt \\
=&\frac{1}{n}\sum_{i=1}^{n} ||\mathbf{a}_1^T \mathbf{s}_i - \mathbf{b}_1^T \mathbf{s}_i||^2 + \tau ||\mathbf{b}_1||^2 + \tau ||\boldsymbol{\beta}_2||^2 \\
=&\frac{1}{n}||\mathbf{S}\mathbf{a}_1 - \mathbf{S}\mathbf{b}_1||^2 + \tau ||\mathbf{b}_1||^2 + \tau ||\boldsymbol{\beta}_2||^2,
\end{aligned}$$

where $\mathbf{s}_i = (\int x_i(t)\phi_1(t)dt, \ldots, \int x_i(t)\phi_K(t)dt)^T$, and $\mathbf{S} = (\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_n)^T$ is an $n \times K$ score matrix. Then the above loss function becomes a ridge regression problem with the solution

$$\widehat{\mathbf{b}}_1 = (\mathbf{S}^T\mathbf{S} + n\tau\mathbf{I})^{-1}\mathbf{S}^T\mathbf{S}\mathbf{a}_1 \text{ and } \hat{\beta}_2(t) \equiv 0.$$

Substituting the estimate $\hat{\beta}(t) = \widehat{\mathbf{b}}_1^T \boldsymbol{\phi}(t)$ into the loss function, we have

$$\begin{aligned}
&\frac{1}{n}\sum_{i=1}^{n} ||x_i(t) - \alpha(t) \int \hat{\beta}(t) x_i(t) dt||^2 + \tau \int \hat{\beta}^2(t) dt \\
=&\frac{1}{n}\sum_{i=1}^{n} ||\mathbf{s}_i^T \boldsymbol{\phi}(t) - (\mathbf{a}_1^T \boldsymbol{\phi}(t) + \alpha_2(t)) \widehat{\mathbf{b}}_1^T \mathbf{s}_i||^2 + \tau \hat{\mathbf{b}}_1^T \hat{\mathbf{b}}_1 \\
=&\frac{1}{n}\sum_{i=1}^{n} ||\mathbf{s}_i^T \boldsymbol{\phi}(t) - \widehat{\mathbf{b}}_1^T \mathbf{s}_i \mathbf{a}_1^T \boldsymbol{\phi}(t)||^2 + \tau \hat{\mathbf{b}}_1^T \hat{\mathbf{b}}_1 + \frac{1}{n}\sum_{i=1}^{n} ||\alpha_2(t) \hat{\mathbf{b}}_1^T \mathbf{s}_i||^2 \\
=&\frac{1}{n}\sum_{i=1}^{n} ||\mathbf{s}_i - \mathbf{a}_1 \hat{\mathbf{b}}_1^T \mathbf{s}_i||^2 + \tau \hat{\mathbf{b}}_1^T \hat{\mathbf{b}}_1 + \frac{1}{n}\sum_{i=1}^{n} ||\alpha_2(t) \hat{\mathbf{b}}_1^T \mathbf{s}_i||^2 \\
=&Tr(\mathbf{S}^T\mathbf{S}) - 2\mathbf{a}_1^T \mathbf{S}^T \mathbf{S} \hat{\mathbf{b}}_1 + \hat{\mathbf{b}}_1^T (n\tau \mathbf{I} + \mathbf{S}^T\mathbf{S}) \hat{\mathbf{b}}_1 + \frac{1}{n}\sum_{i=1}^{n} ||\alpha_2(t) \hat{\mathbf{b}}_1^T \mathbf{s}_i||^2 \\
=&Tr(\mathbf{S}^T\mathbf{S}) - \mathbf{a}_1^T \mathbf{S}^T \mathbf{S} (n\tau \mathbf{I} + \mathbf{S}^T\mathbf{S})^{-1} \mathbf{S}^T\mathbf{S}\mathbf{a}_1 + \frac{1}{n}\sum_{i=1}^{n} ||\alpha_2(t) \hat{\mathbf{b}}_1^T \mathbf{s}_i||^2. \qquad (A.3)
\end{aligned}$$

The last term in (A.3) is reduced to zero only when $\alpha_2(t) = 0$. In addition, since

$$\mathbf{S}^T\mathbf{S}(n\tau\mathbf{I} + \mathbf{S}^T\mathbf{S})^{-1}\mathbf{S}^T\mathbf{S} = \text{diag}(\frac{\lambda_1^2}{n\tau + \lambda_1}, \ldots, \frac{\lambda_K^2}{n\tau + \lambda_K}),$$

the second term in (A.3) is maximized when $\widehat{\boldsymbol{\alpha}}_1 = (1, 0, 0, \ldots, 0)$, due to the fact that $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_K$. Therefore, $\hat{\alpha}(t) = \phi_1(t)$ and $\widehat{\beta}(t) = \frac{\lambda_1}{n\tau + \lambda_1}\phi_1(t)$. □

The next theorem extends Theorem 3.4.1 into the first $J$ leading FPCs.

**Theorem 4.4.2.** Let $\boldsymbol{\alpha}(t) = (\alpha_1(t), \ldots, \alpha_J(t))$ and $\boldsymbol{\beta}(t) = (\beta_1(t), \ldots, \beta_J(t))$. For any $\tau > 0$, let

$$(\widehat{\boldsymbol{\alpha}}(t), \widehat{\boldsymbol{\beta}}(t)) = \arg\min \frac{1}{n}\sum_{i=1}^{n} ||x_i(t) - \sum_{j=1}^{J} \alpha_j(t) \langle \beta_j, x_i \rangle dt||^2 + \tau \sum_{j=1}^{J} \int \beta_j^2(t) dt$$

subject to $\langle \alpha_i, \alpha_j \rangle = \delta_{ij}$ and $\delta_{ij}$ is the Kronecker delta, then $\hat{\beta}_j(t) = c_j \hat{\phi}_j(t), j = 1, \dots, J$, where $\hat{\phi}_j(t)$ is the $j$-th empirical eigenfunctions of the sample covariance function $g(s,t) = \frac{1}{n}\sum_{i=1}^{n} x_i(s)x_i(t)$ and $c_j$ is a scale factor.

*Proof.* For given $\boldsymbol{\alpha}(t)$, each $x_i(t)$ can be expressed as $x_i(t) = \sum_{j=1}^{J}\langle x_i, \alpha_j\rangle \alpha_j(t) + \eta_i(t)$, in which $\eta_i \perp span\{\alpha_1, \dots, \alpha_J\}$, then

$$
\begin{aligned}
&\frac{1}{n}\sum_{i=1}^{n} ||x_i(t) - \sum_{j=1}^{J} \alpha_j(t)\langle \beta_j, x_i\rangle dt||^2 + \tau \sum_{j=1}^{J}\int \beta_j^2(t)dt \\
=&\frac{1}{n}\sum_{i=1}^{n} ||\Big(\langle \boldsymbol{\alpha}(t), x_i(t)\rangle - \langle \boldsymbol{\beta}(t), x_i(t)\rangle\Big)^T \boldsymbol{\alpha}||^2 + \tau||\boldsymbol{\beta}||^2 + \sum_{i=1}^{n}||\eta_i(t)||^2 \\
=&\frac{1}{n}\sum_{i=1}^{n} ||\langle \boldsymbol{\alpha}, x_i(t)\rangle - \langle \boldsymbol{\beta}, x_i(t)\rangle||^2 + \tau||\boldsymbol{\beta}||^2 + \sum_{i=1}^{n}||\eta_i(t)||^2.
\end{aligned}
$$

Due to the fact that $\alpha_j(t)$ is orthogonal to each other, we can maximize the above loss function for each $\beta_j(t), j = 1, \dots, J$ separately. More specifically,

$$
\hat{\beta}_j(t) = \arg\min \frac{1}{n}\sum_{i=1}^{n} ||\langle \alpha_j(t), x_i(t)\rangle - \langle \beta_j(t), x_i(t)\rangle||^2 + \tau||\beta_j(t)||^2.
$$

Similarly, we can express both $\beta_j(t)$ and $\alpha_j(t)$ using the FPCs $\boldsymbol{\phi}(t)$ as $\beta_j(t) = \mathbf{b}_j^T\boldsymbol{\phi}(t)$ and $\alpha_j(t) = \mathbf{a}_j^T\boldsymbol{\phi}(t)$. Then the solution is given as

$$
\widehat{\mathbf{b}}_j = (\mathbf{S}^T\mathbf{S} + n\tau\mathbf{I})^{-1}\mathbf{S}^T\mathbf{S}\mathbf{a}_j,
$$

in which $\mathbf{s}_i = (\int x_i(t)\phi_1(t)dt, \dots, \int x_i(t)\phi_K(t)dt)^T$ and $\mathbf{S} = (\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_n)^T$ is an $n \times K$ score matrix. Now substituting $\widehat{\mathbf{b}}_j$ into the loss function, we have

$$
\begin{aligned}
&\frac{1}{n}\sum_{i=1}^{n} ||x_i(t) - \sum_{j=1}^{J} \alpha_j(t)\langle \beta_j, x_i\rangle dt||^2 + \tau \sum_{j=1}^{J}\int \beta_j^2(t)dt \\
=&Tr(\mathbf{S}^T\mathbf{S}) - 2Tr(\mathbf{a}^T\mathbf{S}^T\mathbf{S}\hat{\mathbf{b}}) + Tr(\hat{\mathbf{b}}^T(\mathbf{S}^T\mathbf{S} + n\tau\mathbf{I})\hat{\mathbf{b}} \\
=&Tr(\mathbf{S}^T\mathbf{S}) - 2\sum_{j=1}^{J} Tr(\mathbf{a}_j^T\mathbf{S}^T\mathbf{S}\hat{\mathbf{b}}_j) + \sum_{j=1}^{J} Tr(\hat{\mathbf{b}}_j^T(\mathbf{S}^T\mathbf{S} + n\tau\mathbf{I})\hat{\mathbf{b}}_j) \\
=&Tr(\mathbf{S}^T\mathbf{S}) - \sum_{j=1}^{J} Tr(\mathbf{a}_j^T\mathbf{S}^T\mathbf{S}(\mathbf{S}^T\mathbf{S} + n\tau\mathbf{I})^{-1}\mathbf{S}^T\mathbf{S}\mathbf{a}_j).
\end{aligned}
$$

Note that

$$
\mathbf{S}^T\mathbf{S}(\mathbf{S}^T\mathbf{S} + \tau\mathbf{I})^{-1}\mathbf{S}^T\mathbf{S} = \mathrm{diag}(\frac{\tau_1^2}{n\tau + \lambda_1}, \dots, \frac{\lambda_K^2}{n\tau + \lambda_K}),
$$

so the loss function is minimized when $\hat{\alpha}_j(t) = \hat{\phi}_j(t)$ and $\hat{\beta}_j(t) = \frac{\lambda_j}{\lambda_j + n\tau}\hat{\phi}_j(t)$.

□

# Appendix B

# Theoretical Results in Section 4.2 and Section 4.4

**Theorem 4.2.1.** For any given value of $M$

$$\{\hat{\psi}_1, \dots, \hat{\psi}_M\} = \arg\min \frac{1}{n}\sum_{i=1}^{n}\Big(\int [x_i(t) - \sum_{m=1}^{M}\alpha_{mi}\psi_m(t)]^2 dt\Big), \tag{B.1}$$

with subject to $\langle \psi_i, \psi_j \rangle = \delta_{ij}$. Then $\{\hat{\psi}_1, \dots, \hat{\psi}_M\}$ is the first $M$ eigenfunctions of $\hat{K}(s,t) = \frac{1}{n}\sum_{i=1}[x_i(s)x_i(t)]$ and $\alpha_{ki} = \langle x_i, \psi_k \rangle$.

*Proof.* We start with $M = 1$, then the problem above becomes

$$\hat{\psi}_1 = \arg\min \frac{1}{n}\sum_{i=1}^{n}\Big(\int [x_i(t) - \alpha_{1i}\psi_1(t)]^2 dt\Big),$$

subject to $||\psi_1||^2 = 1$. For every $x_i(t)$, we can express it as $x_i(t) = \langle x_i, \psi \rangle \psi_1(t) + \eta_i(t)$, in which $\eta_i \perp \psi_1$.

$$\begin{aligned}
\frac{1}{n}\sum_{i=1}^{n}\Big(\int [x_i(t) - \alpha_{1i}\psi_1(t)]^2 dt\Big) &= \frac{1}{n}\Big(\int [\langle x_i, \psi_1 \rangle \psi_1(t) + \eta_i(t) - \alpha_{1i}\psi_1(t)]^2 dt\Big) \\
&= \frac{1}{n}\sum_{i=1}^{n}\Big(\int [(\langle x_i, \psi_1 \rangle - \alpha_{1i})\psi_1(t) + \eta_i(t)]^2 dt\Big) \\
&= \frac{1}{n}\sum_{i=1}^{n}\int \Big[(\langle x_i, \psi_1 \rangle - \alpha_{1i})\psi_1(t)\Big]^2 dt + \frac{1}{n}\sum_{i=1}^{n}\int \eta_i^2(t) dt
\end{aligned}$$

The first term is minimized when $\alpha_{1i} = \langle x_i, \psi_1 \rangle$ and the second term is minimized only when $\psi(t)$ is the first eigenfunction of $\hat{K}(s,t)$. This is because

$$\begin{aligned}
\frac{1}{n}\sum_{i=1}^{n}\Big(\int [x_i(t) - \langle x_i, \psi_1\rangle \psi_1(t)]^2 dt\Big) &= \frac{1}{n}\sum_{i=1}^{n}\Big(\int x_i(t)^2 dt - 2\int \langle x_i, \psi_1\rangle x_i(t)\psi_1(t)dt \\
&+ \langle x_i, \psi_1\rangle^2 \int \psi_1^2(t)dt\Big) \\
&= \frac{1}{n}\sum_{i=1}^{n}\int x_i^2(t)dt - \frac{1}{n}\sum_{i=1}^{n}\langle x_i, \psi_1\rangle^2 \\
&= \frac{1}{n}\sum_{i=1}^{n}\int x_i^2(t)dt - \frac{1}{n}\sum_{i=1}^{n}\int x_i(t)\psi_1(t)dt \int x_i(s)\psi_1(s)ds \\
&= \frac{1}{n}\sum_{i=1}^{n}\int x_i^2(t)dt - \frac{1}{n}\sum_{i=1}^{n}\int\int x_i(t)\psi_1(t)x_i(s)\psi_1(s)dsdt \\
&= \frac{1}{n}\sum_{i=1}^{n}\int x_i^2(t)dt - \int\int \psi_1(t)\hat{K}(s,t)\psi_1(s)dsdt.
\end{aligned}$$

We can easily see that the second term is maximized when $\psi_1(t)$ is the first eigenfunction of $\hat{K}(s,t)$ .

When $M > 1$, we can write each $x_i(t) = \sum_{m=1}^{M}\langle x_i, \psi_m\rangle\psi_m(t) + \eta_i(t)$ and $\eta_i \perp span\{\psi_1, \ldots, \psi_M\}$. Following the same strategy, we can first show that $\alpha_{mi} = \langle x_i, \psi_m\rangle$. Then the problem becomes minimizing

$$\frac{1}{n}\sum_{i=1}^{n}\Big(\int [x_i(t) - \sum_{m=1}^{M}\langle x_i, \psi_m\rangle\psi_m(t)]^2 dt\Big) = \frac{1}{n}\sum_{i=1}^{n}\int x_i^2(t)dt - \sum_{m=1}^{M}\int\int \psi_m(t)\hat{K}(s,t)\psi_m(s)dsdt,$$

which is equivalent to maximizing the second term. It is only when $\hat{\psi}_1, \ldots, \hat{\psi}_K$ are the first $K$ leading eigenfunctions of the sample covariance function $\hat{K}(s,t)$ that the second term is maximized.

□

**Lemma 4.4.1.** Let $(m_i)$ be independent positive random variables with mean 1 and $\sum_{i=1}^{\infty} \mathtt{E}(m_i - 1)^2/i^2 < \infty$. For any sequence $(a_i)$ of positive numbers such that $\sum_{i=1}^{\infty} \mathtt{E}(m_i - 1)^2 a_i^2/i^2 < \infty$ we have

$$\lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n} a_i = \lim_{n\to\infty}\frac{1}{\sum_{i=1}^{n} m_i}\sum_{i=1}^{n} m_i a_i \quad a.s.$$

*Proof.* Since $\sum_{i=1}^{n} m_i/n \to 1$ a.s. under the assumption and by Kolmogorov's strong law of large numbers, e.g. Theorem 1, p. 121 in Teicher and Chow (1978), it suffices to prove that

$$\lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}(m_i - 1)a_i = 0 \quad a.s.$$

Another application of Kolmogorov's strong law of large numbers completes the proof. □

**Lemma 4.4.2.** Let $\{m_{ij} : i = 1, 2, \ldots, j = 1, 2, \ldots\}$ be positive random variables. For each $j = 1, 2, \ldots$, $\{m_{ij} : i = 1, 2, \ldots\}$ are independently and identically distributed with mean 1 and finite variance. Then for any infinite matrix $A = [A_{ij}]$, with $\lambda_j = \lim_n \frac{1}{n} \sum_{i=1}^n a_{ij}^2$ exists for each $j$, as $n \to \infty$,

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{\infty} a_{ij}^2 = \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{\infty} a_{ij}^2 m_{ij} = \sum_{j=1}^{\infty} \lambda_j.$$

*Proof.* First note that $\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^n a_{ij}^2 < \infty$ implies $\sum_{i=1}^\infty a_{ij}^2/i^2 < \infty$. The conditions in Lemma 4.4.1 for $\{m_{ij} : i = 1, 2, \ldots\}$ read

$$\sum_{i=1}^{\infty} \mathtt{E}(m_{ij} - 1)^2/i^2 < \infty$$
$$\sum_{i=1}^{\infty} \mathtt{E}(m_{ij} - 1)^2 a_{ij}^2/i^2 < \infty,$$

which are true by assumption. Hence for fixed $p$, we have from Lemma 4.4.1, almost surely,

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{p} a_{ij}^2 = \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{p} a_{ij}^2 m_{ij}.$$

Since by Tonelli's theorem,

$$\lim_n \lim_p \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{p} a_{ij}^2 = \lim_p \lim_n \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{p} a_{ij},$$

and

$$\lim_n \lim_p \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{p} a_{ij}^2 m_{ij} = \lim_p \lim_n \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{p} a_{ij} m_{ij}, \ a.s.$$

the proof is complete. □

**Lemma 4.4.3.** For a matrix $A_{n\times p} = [a_{ij}]$, the r-rank with $r \leq \min(n, p)$ approximation of $A$ under the Frobenius norm of matrices is $\tilde{A} = \sum_{i=1}^r \alpha_i \otimes \beta_i$ where $\alpha_i$ and $\beta_i$ are the r left and r right singular eigenvectors of A; i.e. the eigenvectors of $AA^T$ and $A^T A$.

*Proof.* By singular value decomposition of $A$, we can write $A = \sum_{k=1}^{\min(n,p)} d_k \mathbf{u}_k \otimes \mathbf{v}_k$, where $(\mathbf{u}_k)_{k=1}^n$ and $(\mathbf{v}_k)_{k=1}^p$ are orthogonal basis in $\mathbb{R}^n$ and $\mathbb{R}^p$. Hence $A - \sum_{i=1}^r \boldsymbol{\alpha}_i \otimes \boldsymbol{\beta}_i$ has minimum squared Frobenius norm $\sum_{i=r+1}^{\min(n,p)} d_i^2$ with minimizing $\boldsymbol{\alpha}_i = d_i \mathbf{u}_i$ and $\boldsymbol{\beta}_i = \mathbf{v}_i$. □

**Remark 1.** If $A$ has orthogonal columns, $A^T A$ has unit basis vectors as eigenvectors; i.e., the $\boldsymbol{\beta}_i$'s are $\mathbf{e}_i \in \mathbb{R}^p$, where $\mathbf{e}_i$ has only the $i$-th entry non-zero, with value 1.

Given sparse observations of functional data $y_{ij} = x_i(t_{ij}) + \epsilon_{ij}$, where the observation times $t_{ij}, j = 1, \ldots, n_i$, for subject $i$ are uniformly drawn from $[0, 1]$, recall the objective function

$$L_n(\boldsymbol{\alpha}, \psi) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{n_i} \sum_{j=1}^{n_i} [y_{ij} - \alpha_i \psi(t_{ij})]^2 \tag{B.2}$$

where $\boldsymbol{\alpha} = (\alpha_i) \in \mathbb{R}^n$ and $\psi(t)$ is a function on $[0, 1]$ with constraint $\int_0^1 \psi^2(t)dt = 1$.

**Theorem 4.4.1.** Let the Mercer expansion with kernel $k(s, t) = \mathtt{E}X(s)X(t)$ of the stochastic process $X(t)$ be $\{\lambda_k, \psi_k^0 : k = 1, 2, \cdots\}$. Assume $\sum_k \lambda_k < \infty$ and $\int_0^1 [\psi_k^0(t)]^4 dt < \infty$ for each $k = 1, 2, \cdots$, then the minimizer $\hat{\psi}(t)$ of $L_n$ converges to $\psi_1^0(t)$ in $L^2(0, 1)$ almost surely as $n \to \infty$.

*Proof.* Since $\Psi^0 := \{\psi_k^0 : k = 1, 2, \cdots\}$ is a complete orthonorml system in $L^2[0, 1]$, we represnet $\psi(t)$ as $\psi(t) = \boldsymbol{\beta}^T \Psi^0$ for coefficient vector $\boldsymbol{\beta} = (\beta_i)$. We also represent $x_i(t) = \sum_{k=1}^{\infty} a_{ik} \psi_k^0(t)$ for $i = 1, \cdots, n$, then (B.2) becomes

$$\begin{aligned} L_n(\boldsymbol{\alpha}, \boldsymbol{\beta}) &= \frac{1}{n} \sum_{i=1}^{n} \frac{1}{n_i} \sum_{j=1}^{n_i} [y_{ij} - \alpha_i \psi(t_{ij})]^2 \\ &= \frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{\infty} (a_{ik} - \alpha_i \beta_k)^2 m_{ik} \\ &\quad + \frac{-2}{n} \sum_{i=1}^{n} \sum_{k=1}^{\infty} (a_{ik} - \alpha_i \beta_k) \frac{1}{n_i} \sum_{j=1}^{n_i} \epsilon_{ij} \\ &\quad + \frac{1}{n} \sum_{i=1}^{n} \frac{1}{n_i} \sum_{j=1}^{n_i} \epsilon_{ij}^2 \end{aligned}$$

where $m_{ik} = \frac{1}{n_i} \sum_{j=1}^{n_i} [\psi_k^0(t_{ij})]^2$. Since $\epsilon_{ij}$ are i.i.d. mean zero normal errors, the second term in the last equality in the above display converges to 0 and the third $\sigma^2$ by strong law of large numbers. Since $t_{ij}$ are uniformly drawn from $[0, 1]$, then

$$\mathtt{E}m_{ik} = \frac{1}{n_i} \sum_{j=1}^{n_i} \int_0^1 [\psi_k^0(t_{ij})]^2 \, dt_{ij} = 1,$$

as $\psi_k$ are constrained to have norm 1 in $L^2[0, 1]$. By Lemma 2, almost surely,

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{\infty} (a_{ik} - \alpha_i \beta_k)^2 m_{ik} = \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{\infty} (a_{ik} - \alpha_i \beta_k)^2 =: L(\boldsymbol{\alpha}, \boldsymbol{\beta})$$

hence the minimizer $\hat{\boldsymbol{\beta}}$ of $L_n(\boldsymbol{\alpha}, \boldsymbol{\beta})$ will converge to the minimizer $\boldsymbol{\beta}$ of $L(\boldsymbol{\alpha}, \boldsymbol{\beta})$ (as $L_n(\boldsymbol{\alpha}, \boldsymbol{\beta})$ are convex functions of $(\boldsymbol{\alpha}, \boldsymbol{\beta})$, see e.g. the *Convexity Lemma* in Pollard (1991)), which is the first eigenvector of $A^T A$, with $A = (a_{ik})_{i=1,2,\ldots}^{k=1,2,\ldots}$ and we know A has orthogonal columns, thus $\boldsymbol{\beta} = (1, 0, \cdots)$. Hence $\hat{\psi}(t) := \hat{\boldsymbol{\beta}}^T \Psi^0(t)$ converges to $\psi_1^0(t)$ in $L^2[0, 1]$ almost surely. □

**Theorem 4.4.2.** The minimizers $\hat{\psi}_l(t)$, $l = 1, \cdots, M$, of the loss function

$$L_n(\{\boldsymbol{\alpha}_l, \psi_l\}_{l=1}^M) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}[y_{ij} - \sum_{l=1}^{M}\alpha_{il}\psi_l(t_{ij})]^2 \tag{B.3}$$

converges to $\psi_l^0(t)$, $l = 1, \cdots, M$ in $L^2[0,1]$ almost surely as $n \to \infty$.

*Proof.* We denote $\boldsymbol{\beta}_l$ as the coefficients of projection of $\psi_l(t)$ on the basis $(\psi_k^0)$ of $L^2[0,1]$, then the objective function (B.3) becomes

$$L_n(\{\boldsymbol{\alpha}_l, \boldsymbol{\beta}_l\}_{l=1}^M) = \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{\infty}(a_{ik} - (\sum_{l=1}^{M}\boldsymbol{\alpha}_l \otimes \boldsymbol{\beta}_l)_{ik})^2 m_{ik} + o_P(1)$$

which have minimizers $\hat{\boldsymbol{\beta}}_l$ such that $\hat{\boldsymbol{\beta}}_l \to \mathbf{e}_l$ in $\ell_2$ for each $1 \leq l \leq M$ by the same argument as for $M = 1$. □