ABSTRACT

LI, CAI. Statistical Methods for Functional and Complex Data. (Under the direction of Luo Xiao and Ana-Maria Staicu.)

This dissertation contains a few independent research projects with emphasis on developing statistical and computational methods for functional and complex data.

In the first chapter, we propose a novel covariance smoothing method based on penalized splines and associated software. Smoothing of noisy sample covariances is an important component in functional data analysis. The proposed method is a bivariate spline smoother that is designed for covariance smoothing and can be used for sparse functional or longitudinal data. We propose a fast algorithm for covariance smoothing using leave-one-subject-out cross validation. Our simulations show that the proposed method compares favorably against several commonly used methods. The method is applied to a study of child growth led by one of coauthors and to a public dataset of longitudinal CD4 counts.

In the second chapter, we study the design problem for optimal classification of functional data. The goal is to select sampling time points so that functional data observed at these time points can be classified as accurately as possible. We propose optimal designs that are applicable for a pilot study with either dense or sparse functional data. Using linear discriminant analysis, we formulate our design objectives as explicit functions of the sampling points. We study the theoretical properties of the proposed design objectives and provide a practical implementation. The performance of the proposed design is assessed through simulations and real data applications.

In the third chapter, we propose a novel modeling framework to study the effect of covariates of various types on the conditional distribution of the response. The methodology accommodates flexible model structure, allows for joint estimation of the quantiles at all levels, and involves a computationally efficient estimation algorithm. Extensive numerical investigation confirms good performance of the proposed method. The methodology is motivated by and applied to a lactating sow study, where the primary interest is to understand how the dynamic change of minute-by-

minute temperature in the farrowing rooms within a day (functional covariate) is associated with low quantiles of feed intake of lactating sows, while accounting for other sow-specific information (vector covariate).

In the fourth chapter, we focus on an important computational aspect of matrix learning problems. The nuclear norm regularization is frequently employed to estimate a low rank matrix. To minimize this class of convex functions, a vital and most time-consuming step is singular value thresholding. We provide a `MATLAB` wrapper function `svt` that implements singular value thresholding, which is not available for using the built-in `MATLAB` functions. It encompasses both top singular value decomposition and thresholding, handles both large sparse matrices and structured matrices, and reduces the computation cost in the algorithms.

In the final chapter, we discuss future directions for research in optimal design for functional data and present some partial results.

© Copyright 2017 by Cai Li

All Rights Reserved

Statistical Methods for Functional and Complex Data

by
Cai Li

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Statistics

Raleigh, North Carolina

2017

APPROVED BY:

Hua Zhou	Zhilin Li
Luo Xiao Chair of Advisory Committee	Ana-Maria Staicu Vice-chair of Advisory Committee

DEDICATION

To my family and friends

BIOGRAPHY

The author was born in China. He earned a Bachelor of Engineering degree in Bioinformatics from Huazhong University of Science and Technology and was awarded a Master of Science degree in Genetics from Chinese Academy of Sciences. In 2011, he was selected for the Ph.D. program of Bioinformatics at North Carolina State University. In 2014, he switched to the Ph.D. program of Statistics and received his master's degrees in statistics and bioinformatics. He will graduate with a Ph.D. in Statistics in 2017.

ACKNOWLEDGEMENTS

First and foremost, I would like to take this opportunity to express my deepest and wholehearted gratitude to my advisor Dr. Luo Xiao for his endless support, infinite patience and outstanding guidance throughout the years. His extensive knowledge, deep insight into statistics and sharp thinking have been a great source of inspiration for my research. His overwhelming passion for research, taking great care of students and professionalism have a far-reaching impact on my career. Most definitely I am indebted to him for dedicating countless time and tireless effort to walking through the technical details with me, correcting my English writing and improving my presentation skills. I am not only deeply grateful for his generous financial support, but also his tremendous help on my career development. I am fortunate enough to have him as my advisor and friend.

I would like to thank all my committee members for their suggestions and comments on this dissertation. They have been overwhelmingly supportive and helpful through my Ph.D. studies and job searching. I am deeply grateful to Dr. Ana-Maria Staicu for her invaluable help and guidance in my research, and for leading me into the world of functional data analysis. Her enthusiasm for research have always motivated me. I owe my special thanks to Dr. Hua Zhou for all he has taught me about statistical computing and optimization. Without his constant help and encouragement I could not have made it this far. I also thank Dr. Zhilin Li for kindly serving on my committee.

I extend my sincere appreciation to other faculty members and staff in both Department of Statistics and Bioinformatics Research Center for offering excellent courses and creating a great academic environment. I thank my friends, fellow students and everyone who helped me through this journey.

Thanks to Bill & Melinda Gates Foundation for supporting my research. I also thank Dr. Shasha Jumbe for his visionary leadership of the Healthy Birth, Growth, and Development Knowledge Integration program.

Finally, I would like to thank my family for their boundless love and unconditional support. I especially thank my parents for providing such a wonderful home to grow up in.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Fast Covariance Estimation for Sparse Functional Data[1]

1.1 Introduction

The covariance function is a crucial ingredient in functional data analysis. Sparse functional or longitudinal data are ubiquitous in scientific studies, while functional principal component analysis has become one of the first-line approaches to analyzing this type of data; see, e.g., Besse and Ramsay (1986); Ramsay and Dalzell (1991); Kneip (1994); Besse et al. (1997); Staniswalis and Lee (1998); Yao et al. (2003, 2005).

Given a sample of functions observed at a finite number of locations and, often, with sizable measurement error, there are usually three approaches for obtaining smooth functional principal components: 1) smooth the functional principal components of the sample covariance function; 2) smooth each curve and diagonalize the resulting sample covariance of the smoothed curves; and 3) smooth the sample covariance function and then diagonalize it.

The sample covariance function is typically noisy and difficult to interpret. Therefore, bivariate smoothing is usually employed. Local linear smoothers (Fan and Gijbels, 1996), tensor-product bi-

[1]This chapter is based on a joint work with Luo Xiao, William Checkley and Ciprian Crainiceanu, which has been accepted by *Statistics and Computing*.

variate P-splines (Eilers and Marx, 2003) and thin plate regression splines (Wood, 2003) are among the popular methods for smoothing the sample covariance function. For example, the *fpca.sc* function in the R package *refund* (Huang et al., 2015) uses the tensor-product bivariate P-splines. However, there are two known problems with these smoothers: 1) they are general-purpose smoothers that are not designed specifically for covariance operators; and 2) they ignore that the subject, instead of the observation, is the independent sampling unit and assume that the empirical covariance surface is the sum between an underlying smooth covariance surface and independent random noise. The FACE smoothing approach proposed by Xiao et al. (2013) was designed specifically to address these weaknesses of off-the-shelf covariance smoothing software. The method is implemented in the function *fpca.face* in the *refund* R package (Huang et al., 2015) and has proven to be reliable and fast in a range of applications. However, FACE was developed for high-dimensional dense functional data and the extension to sparse data is far from obvious. One approach that attempts to solve these problems was proposed by Yao et al. (2003). In their paper they used leave-one-subject-out cross-validation to choose the bandwidth for local polynomial smoothing methods. This approach is theoretically sound, but computationally expensive. This may be the reason why the practice is to either try multiple bandwidths and visually inspect the results or completely ignore within-subject correlations.

Several alternative methods for covariance smoothing of sparse functional data also exist in the literature: James et al. (2000) used reduced rank spline mixed effects models, Cai and Yuan (2012) considered nonparametric covariance function under the reproducing kernel Hilbert space framework, and Peng and Paul (2009) proposed a geometric approach under the framework of marginal maximum likelihood estimation.

Our paper has two aims. First, we propose a new automatic bivariate smoother that is specifically designed for covariance function estimation and can be used for sparse functional data. Second, we propose a fast algorithm for selecting the smoothing parameter of the bivariate smoother using leave-one-subject-out cross validation. The code for the proposed method is publicly available in the *face* R package (Xiao et al., 2017b).

1.2 Model

Suppose that the observed data take the form $\{(y_{ij}, t_{ij}), j = 1, \ldots, m_i, i = 1, \ldots, n\}$, where t_{ij} is in the unit interval $[0, 1]$, n is the number of subjects, and m_i is the number of observations for subject i. The model is

$$y_{ij} = f(t_{ij}) + u_i(t_{ij}) + \epsilon_{ij}, \tag{1.1}$$

where f is a smooth mean function, $u_i(t)$ is generated from a zero-mean Gaussian process with covariance operator $\mathcal{C}(s, t) = \text{Cov}\{u_i(s), u_i(t)\}$, and ϵ_{ij} is white noise following a normal distribution $\mathcal{N}(0, \sigma_\epsilon^2)$. We assume that the random terms are independent across subjects and from each other. For longitudinal data, m_i's are usually much smaller than n.

We are interested in estimating the covariance function $\mathcal{C}(s, t)$. A standard procedure employed for obtaining a smooth estimate of $\mathcal{C}(s, t)$ consists of two steps. In the first step, an empirical estimate of the covariance function is constructed. Let $r_{ij} = y_{ij} - f(t_{ij})$ be the residuals and $\mathcal{C}_{ij_1j_2} = r_{ij_1} r_{ij_2}$ be the auxiliary variables. Because $\mathbb{E}(\mathcal{C}_{ij_1j_2}) = \mathcal{C}(t_{ij_1}, t_{ij_2})$ if $j_1 \neq j_2$, $\{\mathcal{C}_{ij_1j_2} : 1 \leq j_1 \neq j_2 \leq m_i, i = 1, \ldots, n\}$ is a collection of unbiased empirical estimates of the covariance function. In the second step, the empirical estimates are smoothed using a bivariate smoother. Smoothing is required because the empirical estimates are usually noisy and scattered in time. Standard bivariate smoothers are local linear smoothers (Fan and Gijbels, 1996), tensor-product bivariate P-splines (Eilers and Marx, 2003) and thin plate regression splines (Wood, 2003). In the following section we propose a statistically efficient, computationally fast and automatic smoothing procedure that serves as an alternative to these approaches.

To carry out the above steps, we assume a mean function estimator $\hat{f}$ exists. Then we let $\hat{r}_{ij} = y_{ij} - \hat{f}(t_{ij})$ and $\widehat{C}_{ij_1j_2} = \hat{r}_{ij_1} \hat{r}_{ij_2}$. Note that we use the hat notation on variables when f is substituted by $\hat{f}$ and when we define a variable with a hat notation, the same variable without a hat notation is similarly defined using the true f. In our software, we estimate f using a P-spline smoother (Eilers and Marx, 1996) with the smoothing parameter selected by leave-one-subject-out cross validation. See Appendix A.1 for details.

1.3 Methodology

We model the covariance function $\mathcal{C}(s,t)$ as a tensor-product splines $H(s,t) = \sum_{1\le\kappa\le c, 1\le\ell\le c}\theta_{\kappa\ell}B_\kappa(s)B_\ell(t)$, where $\mathbf{\Theta} = (\theta_{\kappa\ell})_{1\le\kappa\le c,1\le\ell\le c}$ is a coefficient matrix, $\{B_1(\cdot),\ldots,B_c(\cdot)\}$ is the collection of B-spline basis functions in the unit interval, and c is the number of interior knots plus the order (degree plus 1) of the B-splines. Note that the locations and number of knots as well as the polynomial degrees of splines determine the forms of the B-spline basis functions (de Boor, 1978). We use equally-spaced knots and enforce the following constraint on $\mathbf{\Theta}$:

$$\theta_{\kappa\ell} = \theta_{\ell\kappa}, 1 \le \kappa, \ell \le c.$$

With this constraint, $H(s,t)$ is always symmetric in s and t, a desired property for estimates of covariance functions.

Unlike the other approaches covariance function estimation methods described before, our method applies a joint estimation of covariance function and error variance and incorporates the correlation structure of the auxiliary variables $\{\widehat{C}_{ij_1j_2} : 1 \le j_1 \le j_2 \le m_i, i = 1,\ldots,n\}$ in a two-step procedure to boost statistical efficiency. Because we use a relatively large number of knots, estimating $\mathbf{\Theta}$ by least squares or weighted least squares tends to overfit. Thus, we estimate $\mathbf{\Theta}$ by minimizing a penalized weighted least squares. Let $n_i = m_i(m_i+1)/2$, $\widehat{\mathbf{C}}_{ij} = \left\{\widehat{C}_{ijj}, \widehat{C}_{ij(j+1)},\ldots,\widehat{C}_{ijm_i}\right\}^T \in \mathbb{R}^{m_i-j+1}$, $\mathbf{H}_{ij} = \{H(t_{ij},t_{ij}), H(t_{ij},t_{i(j+1)}),\ldots,H(t_{ij},t_{im_i})\}^T \in \mathbb{R}^{m_i-j+1}$, and $\boldsymbol{\delta}_{ij} = (1, \mathbf{0}_{m_i-j}^T)^T \in \mathbb{R}^{m_i-j+1}$ for $1 \le j \le m_i$. Then let $\widehat{\mathbf{C}}_i = (\widehat{\mathbf{C}}_{i1}^T, \widehat{\mathbf{C}}_{i2}^T,\ldots,\widehat{\mathbf{C}}_{im_i}^T)^T \in \mathbb{R}^{n_i}$ be the vector of all auxiliary variables $\widehat{C}_{ij_1j_2}$ for subject i with $j_1 \le j_2$. Here $\widehat{\mathbf{C}}_i$ contains the nugget terms $\widehat{C}_{ijj}$ and note that $\mathbb{E}(\mathcal{C}_{ijj}) = r(t_{ij},t_{ij}) + \sigma_\epsilon^2$. Similarly, we let $\mathbf{H}_i = (\mathbf{H}_{i1}^T, \mathbf{H}_{i2}^T,\ldots,\mathbf{H}_{im_i}^T)^T \in \mathbb{R}^{n_i}$, and $\boldsymbol{\delta}_i = (\boldsymbol{\delta}_{i1}^T, \boldsymbol{\delta}_{i2}^T,\ldots,\boldsymbol{\delta}_{im_i}^T)^T \in \mathbb{R}^{n_i}$. Also let $\mathbf{W}_i \in \mathbb{R}^{n_i\times n_i}$ be a weight matrix for capturing the correlation of $\widehat{\mathbf{C}}_i$ and will be specified later. The weighted least squares is given by $\sum_{i=1}^n \left(\mathbf{H}_i + \boldsymbol{\delta}_i\sigma_\epsilon^2 - \widehat{\mathbf{C}}_i\right)^T \mathbf{W}_i \left(\mathbf{H}_i + \boldsymbol{\delta}_i\sigma_\epsilon^2 - \widehat{\mathbf{C}}_i\right)$. Let $\|\cdot\|_F$ denote the Frobenius norm and let $\mathbf{D} \in \mathbb{R}^{c\times(c-2)}$ be a second-order differencing matrix (Eilers and Marx, 1996). Then we estimate

$\boldsymbol{\Theta}$ and σ_ϵ^2 by

$$(\widehat{\boldsymbol{\Theta}}, \hat{\sigma}_\epsilon^2) = \arg \min_{\boldsymbol{\Theta}:\boldsymbol{\Theta}=\boldsymbol{\Theta}^T, \sigma_\epsilon^2} \left\{ \text{WLS} + \lambda \|\boldsymbol{\Theta}\mathbf{D}\|_F^2 \right\}, \tag{1.2}$$

where λ is a smoothing parameter that balances model fit and smoothness of the estimate.

The penalty term $\|\boldsymbol{\Theta}\mathbf{D}\|_F^2$ is essentially equivalent to the penalty $\iint_{s,t} \left\{ \frac{\partial^2 H}{\partial s^2}(s,t) \right\}^2 \mathrm{d}s\mathrm{d}t$ and can be interpreted as the row penalty in bivariate P-splines (Eilers and Marx, 2003). Note that when $\boldsymbol{\Theta}$ is symmetric, as in our case, the row and column penalties in bivariate P-splines become the same. Therefore, our proposed method can be regarded as a special case of bivariate P-splines that is designed specifically for covariance function estimation. Another note is that when the smoothing parameter goes to infinity, the penalty term forces $H(s,t)$ to become linear in both the s and the t directions. Finally, if $\hat{\theta}_{\kappa\ell}$ denotes the (κ,ℓ)th element of $\widehat{\boldsymbol{\Theta}}$, then our estimate of the covariance function $\mathcal{C}(s,t)$ is given by $\widetilde{C}(s,t) = \sum_{1\le\kappa\le c, 1\le\ell\le c} \hat{\theta}_{\kappa\ell} B_\kappa(s) B_\ell(t)$.

1.3.1 Estimation

Let $\mathbf{b}(t) = \{B_1(t), \ldots, B_c(t)\}^T$ be a vector. Let $\text{vec}(\cdot)$ be an operator that stacks the columns of a matrix into a vector and denote $\otimes$ the Kronecker product operator. Then $H(s,t) = \{\mathbf{b}(t) \otimes \mathbf{b}(s)\}^T \text{vec}\,\boldsymbol{\Theta}$. Let $\boldsymbol{\theta} = \text{vech}\,\boldsymbol{\Theta}$, where $\text{vech}(\cdot)$ is an operator that stacks the columns of the lower triangle of a matrix into a vector, and let $\mathbf{G}_c$ be the duplication matrix (page 246, Seber 2007) such that $\text{vec}\,\boldsymbol{\Theta} = \mathbf{G}_c\boldsymbol{\theta}$. It follows that $H(s,t) = \{\mathbf{b}(t) \otimes \mathbf{b}(s)\}^T \mathbf{G}_c\boldsymbol{\theta}$.

Let $\mathbf{B}_{ij} = [\mathbf{b}(t_{ij}), \ldots, \mathbf{b}(t_{im_i})] \otimes \mathbf{b}(t_{ij})$, $\mathbf{B}_i = [\mathbf{B}_{i1}^T, \ldots, \mathbf{B}_{im_i}^T]^T$ and $\mathbf{B} = [\mathbf{B}_1^T, \ldots, \mathbf{B}_n^T]^T$. Also let $\mathbf{X}_i = [\mathbf{B}_i\mathbf{G}_c, \boldsymbol{\delta}_i]$ and $\mathbf{X} = [\mathbf{X}_1^T, \ldots, \mathbf{X}_n^T]^T$. $\boldsymbol{\alpha} = (\boldsymbol{\theta}^T, \sigma_\epsilon^2)^T$. Finally let $\widehat{\mathbf{C}} = (\widehat{\mathbf{C}}_i^T, \ldots, \widehat{\mathbf{C}}_n^T)^T$, $\boldsymbol{\delta} = (\boldsymbol{\delta}_1^T, \cdots, \boldsymbol{\delta}_n^T)^T$ and $\mathbf{W} = \text{blockdiag}(\mathbf{W}_1, \cdots, \mathbf{W}_n)$. Note that $\mathbf{X}$ can also be written as $[\mathbf{B}\mathbf{G}_c, \boldsymbol{\delta}]$. Then,

$$\mathbb{E}(\widehat{\mathbf{C}}_i) = \mathbf{H}_i + \boldsymbol{\delta}_i\sigma_\epsilon^2 = [\mathbf{B}_i\mathbf{G}_c, \boldsymbol{\delta}_i] \left(\boldsymbol{\theta}, \sigma_\epsilon^2\right) = \mathbf{X}_i\boldsymbol{\alpha},$$

and

$$\text{WLS} = \left(\widehat{\mathbf{C}} - \mathbf{X}\boldsymbol{\alpha}\right)^T \mathbf{W} \left(\widehat{\mathbf{C}} - \mathbf{X}\boldsymbol{\alpha}\right). \tag{1.3}$$

Next let $\mathrm{tr}(\cdot)$ be the trace operator such that for a square matrix $\mathbf{A}$, $\mathrm{tr}(\mathbf{A})$ is the sum of the diagonals of $\mathbf{A}$. We can derive that (page 241, Seber 2007)

$$\begin{aligned} \|\mathbf{\Theta D}\|_F^2 &= \mathrm{tr}(\mathbf{\Theta D D}^T\mathbf{\Theta}^T), \\ &= (\mathrm{vec}\,\mathbf{\Theta})^T(\mathbf{I}_c \otimes \mathbf{D D}^T)\mathrm{vec}\,\mathbf{\Theta}. \end{aligned}$$

Because $\mathrm{vec}\,\mathbf{\Theta} = \mathbf{G}_c\boldsymbol{\theta}$, we obtain that

$$\begin{aligned} \|\mathbf{\Theta D}\|_F^2 &= \boldsymbol{\theta}^T\mathbf{G}_c^T(\mathbf{I}_c \otimes \mathbf{D D}^T)\mathbf{G}_c^T\boldsymbol{\theta} \\ &= \begin{pmatrix} \boldsymbol{\theta}^T & \sigma_\epsilon^2 \end{pmatrix} \begin{pmatrix} \mathbf{P} & \mathbf{0} \\ \mathbf{0} & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{\theta} \\ \sigma_\epsilon^2 \end{pmatrix} \\ &= \boldsymbol{\alpha}^T\mathbf{Q}\boldsymbol{\alpha}, \end{aligned} \tag{1.4}$$

where $\mathbf{P} = \mathbf{G}_c^T(\mathbf{I}_c \otimes \mathbf{D D}^T)\mathbf{G}_c^T$ and $\mathbf{Q}$ is the block matrix containing $\mathbf{P}$ and zeros.

By (1.3) and (1.4), the objective function in (1.2) can be rewritten as

$$\widehat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha}} \left(\widehat{\mathbf{C}} - \mathbf{X}\boldsymbol{\alpha}\right)^T \mathbf{W} \left(\widehat{\mathbf{C}} - \mathbf{X}\boldsymbol{\alpha}\right) + \lambda\boldsymbol{\alpha}^T\mathbf{Q}\boldsymbol{\alpha}. \tag{1.5}$$

Now we obtain an explicit form of $\widehat{\boldsymbol{\alpha}}$

$$\widehat{\boldsymbol{\alpha}} = \begin{pmatrix} \hat{\boldsymbol{\theta}} \\ \hat{\sigma}_\epsilon^2 \end{pmatrix} = \left(\mathbf{X}^T\mathbf{W}\mathbf{X} + \lambda\mathbf{Q}\right)^{-1} \left(\mathbf{X}^T\mathbf{W}\widehat{\mathbf{C}}\right). \tag{1.6}$$

We need to specify the weight matrices $\mathbf{W}_i$'s. One sensible choice for $\mathbf{W}_i$ is the inverse of $\mathrm{Cov}\,(\mathbf{C}_i)$, where $\mathbf{C}_i$ is defined similar to $\widehat{\mathbf{C}}_i$, except that the true mean function f is used. However, $\mathrm{Cov}\,(\mathbf{C}_i)$ may not be invertible or may be close to being singular. Thus, we specify $\mathbf{W}_i$ as

$$\mathbf{W}_i^{-1} = (1-\beta)\mathrm{Cov}\,(\mathbf{C}_i) + \beta\mathrm{diag}\left\{\mathrm{diag}\left\{\mathrm{Cov}\,(\mathbf{C}_i)\right\}\right\}, 1 \le i \le n,$$

for some constant $0 < \beta < 1$. The above specification ensures that $\mathbf{W}_i$ exists and is stable. We will use $\beta = 0.05$, which works well in practice.

We now derive $\mathrm{Cov}\,(\mathbf{C}_i)$ in terms of $\mathcal{C}$ and σ_ϵ^2. First note that $\mathbb{E}(r_{ij_1} r_{ij_2}) = \mathrm{Cov}(r_{ij_1}, r_{ij_2}) = \mathcal{C}(t_{ij_1}, t_{ij_2}) + \delta_{j_1 j_2}\sigma_\epsilon^2$, where $\delta_{j_1 j_2} = 1$ if $j_1 = j_2$ and 0 otherwise.

Proposition 1. *Define* $\mathbf{M}_{ijk} = \left\{\mathcal{C}(t_{ij}, t_{ik}), \delta_{jk}\sigma_\epsilon^2\right\}^T \in \mathbb{R}^2$. *Then,*

$$\mathrm{Cov}\,(\mathcal{C}_{ij_1 j_2}, \mathcal{C}_{ij_3 j_4}) = \mathbf{1}^T(\mathbf{M}_{ij_1 j_3} \otimes \mathbf{M}_{ij_2 j_4} + \mathbf{M}_{ij_1 j_4} \otimes \mathbf{M}_{ij_2 j_3}).$$

The proof of Proposition 1 is provided in Appendix A.2. Now we see that $\mathbf{W}_i$ also depends on $(\mathcal{C}, \sigma_\epsilon^2)$. Hence, we employ a two-stage estimation. We first estimate $(\mathcal{C}, \sigma_\epsilon^2)$ by using penalized ordinary least squares, i.e., $\mathbf{W}_i = \mathbf{I}$ for all i. Then we obtain the plug-in estimate of $\mathbf{W}_i$ and estimate $(\mathcal{C}, \sigma_\epsilon^2)$ using penalized weighted least squares. The algorithm for the two-stage estimation is summarized as Algorithm 1.

Algorithm 1: Estimation algorithm

Input: data, specification of settings of univariate marginal basis functions and the smoothing parameter λ

Output: estimate of $\mathcal{C}$ and σ_ϵ^2

1 Initialize $\widehat{\mathbf{C}}$, $\mathbf{X}$ and $\mathbf{Q}$;

2 $\widehat{\boldsymbol{\alpha}}^{(0)} \leftarrow \arg\min_{\boldsymbol{\alpha}} (\widehat{\mathbf{C}} - \mathbf{X}\boldsymbol{\alpha})^T(\widehat{\mathbf{C}} - \mathbf{X}\boldsymbol{\alpha}) + \lambda \boldsymbol{\alpha}^T \mathbf{Q}\boldsymbol{\alpha}$;

3 $\widehat{\mathbf{W}} \leftarrow \mathbf{W}\{\widehat{\boldsymbol{\alpha}}^{(0)}\}$;

4 $\widehat{\boldsymbol{\alpha}} \leftarrow \arg\min_{\boldsymbol{\alpha}} (\widehat{\mathbf{C}} - \mathbf{X}\boldsymbol{\alpha})^T \widehat{\mathbf{W}} (\widehat{\mathbf{C}} - \mathbf{X}\boldsymbol{\alpha}) + \lambda \boldsymbol{\alpha}^T \mathbf{Q}\boldsymbol{\alpha}$;

1.3.2 Selection of the Smoothing Parameter

For selecting the smoothing parameter, we use leave-one-subject-out cross validation, a popular approach for correlated data; see, for example, Yao et al. (2003), Reiss and Ogden (2010) and Xiao et al. (2015). Compared to the leave-one-observation-out cross validation, which ignores the correlation, leave-one-subject-out cross-validation was reported to be more robust against overfit.

However, such an approach is usually computationally expensive. In this section, we derive a fast algorithm for approximating the leave-one-subject-out cross validation.

Let $\widetilde{\mathbf{C}}_i^{[i]}$ be the prediction of $\widehat{\mathbf{C}}_i$ by applying the proposed method to the data without the data from the ith subject, then the cross-validated error is

$$\text{iCV} = \sum_{i=1}^{n} \|\widetilde{\mathbf{C}}_i^{[i]} - \widehat{\mathbf{C}}_i\|^2. \tag{1.7}$$

There is a simple formula for iCV. First we let $\mathbf{S} = \mathbf{X}(\mathbf{X}^T\mathbf{W}\mathbf{X} + \lambda\mathbf{Q})^{-1}\mathbf{X}^T\mathbf{W}$, which is the smoother matrix for the proposed method. $\mathbf{S}$ can be written as $(\mathbf{X}\mathbf{A})[\mathbf{I} + \lambda\text{diag}(\mathbf{s})]^{-1}(\mathbf{X}\mathbf{A})^T\mathbf{W}$ for some square matrix $\mathbf{A}$ and $\mathbf{s}$ is a column vector; see, for example, Xiao et al. (2013). In particular, both $\mathbf{A}$ and $\mathbf{s}$ do not depend on λ.

Let $\mathbf{S}_i = \mathbf{X}_i(\mathbf{X}^T\mathbf{W}\mathbf{X} + \lambda\mathbf{Q})^{-1}\mathbf{X}^T\mathbf{W}$ and $\mathbf{S}_{ii} = \mathbf{X}_i(\mathbf{X}^T\mathbf{W}\mathbf{X} + \lambda\mathbf{Q})^{-1}\mathbf{X}_i^T\mathbf{W}_i$. Then $\mathbf{S}_i$ is of dimension $n_i \times N$, where $N = \sum_{i=1}^n n_i$, and $\mathbf{S}_{ii}$ is of dimension $n_i \times n_i$.

Lemma 1. *The* iCV *in* (1.7) *can be simplified as*

$$\text{iCV} = \sum_{i=1}^{n} \|(\mathbf{I}_{n_i} - \mathbf{S}_{ii})^{-1}(\mathbf{S}_i\widehat{\mathbf{C}} - \widehat{\mathbf{C}}_i)\|^2.$$

The proof of Lemma 1 is the same as that of Lemma 3.1 in Xu and Huang (2012) and thus is omitted. Similar to Xu and Huang (2012), we further simplify iCV by using the approximation $(\mathbf{I}_{n_i} - \mathbf{S}_{ii}^T)^{-1}(\mathbf{I}_{n_i} - \mathbf{S}_{ii})^{-1} = \mathbf{I}_{n_i} + \mathbf{S}_{ii} + \mathbf{S}_{ii}^T$. This approximation leads to the generalized cross validation, which we denote as iGCV,

$$\begin{aligned} \text{iGCV} &= \sum_{i=1}^{n}(\mathbf{S}_i\widehat{\mathbf{C}} - \widehat{\mathbf{C}}_i)^T(\mathbf{I}_{n_i} + \mathbf{S}_{ii} + \mathbf{S}_{ii}^T)(\mathbf{S}_i\widehat{\mathbf{C}} - \widehat{\mathbf{C}}_i) \\ &= \|\widehat{\mathbf{C}} - \mathbf{S}\widehat{\mathbf{C}}\|^2 + 2\sum_{i=1}^{n}\left(\mathbf{S}_i\widehat{\mathbf{C}} - \widehat{\mathbf{C}}_i\right)^T\mathbf{S}_{ii}\left(\mathbf{S}_i\widehat{\mathbf{C}} - \widehat{\mathbf{C}}_i\right). \end{aligned} \tag{1.8}$$

While iGCV in (1.8) is much easier to compute than iCV in (1.7), the formula in (1.8) is still computationally expensive as the smoother matrix $\mathbf{S}$ is of dimension $N \times N$, where $N = 2,000$ if

$n = 100$ and $m_i = m = 5$ for all i. Thus, we further simplify iGCV.

Let $\mathbf{F}_i = \mathbf{X}_i\mathbf{A}$, $\mathbf{F} = \mathbf{X}\mathbf{A}$ and $\widetilde{\mathbf{F}} = \mathbf{F}^T\mathbf{W}$. Define $\mathbf{f}_i = \mathbf{F}_i^T\widehat{\mathbf{C}}_i$, $\mathbf{f} = \mathbf{F}^T\widehat{\mathbf{C}}$, $\widetilde{\mathbf{f}} = \widetilde{\mathbf{F}}\widehat{\mathbf{C}}$, $\mathbf{J}_i = \mathbf{F}_i^T\mathbf{W}_i\widehat{\mathbf{C}}_i$, $\mathbf{L}_i = \mathbf{F}_i^T\mathbf{F}_i$ and $\widetilde{\mathbf{L}}_i = \mathbf{F}_i^T\mathbf{W}_i\mathbf{F}_i$. To simplify notation we will denote $[\mathbf{I} + \lambda\text{diag}(\mathbf{s})]^{-1}$ as $\widetilde{\mathbf{D}}$, a symmetric matrix, and its diagonal as $\widetilde{\mathbf{d}}$. Let $\odot$ be the Hadamard product such that for two matrices of the same dimensions $A = (a_{ij})$ and $B = (b_{ij})$, $A \odot B = (a_{ij}b_{ij})$.

Proposition 2. *The* iGCV *in* (1.8) *can be simplified as*

$$\begin{aligned}\text{iGCV} &= \|\widehat{\mathbf{C}}\|^2 - 2\widetilde{\mathbf{d}}^T(\widetilde{\mathbf{f}} \odot \mathbf{f}) + (\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}})^T(\mathbf{F}^T\mathbf{F})(\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}}) + 2\widetilde{\mathbf{d}}^T\mathbf{g} \\ &- 4\widetilde{\mathbf{d}}^T\mathbf{G}\widetilde{\mathbf{d}} + 2\widetilde{\mathbf{d}}^T\left[\sum_{i=1}^n \left\{\mathbf{L}_i(\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}})\right\} \odot \left\{\widetilde{\mathbf{L}}_i(\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}})\right\}\right],\end{aligned}$$

where $\mathbf{g} = \sum_{i=1}^n \mathbf{J}_i \odot \mathbf{f}_i$ *and* $\mathbf{G} = \sum_{i=1}^n (\mathbf{J}_i\widetilde{\mathbf{f}}^T) \odot \mathbf{L}_i$.

The proof of Proposition 2 is provided in Appendix A.2.

Algorithm 2: Tuning algorithm

Input: $\mathbf{X}$, $\widehat{\mathbf{C}}$, $\mathbf{Q}$, $\mathbf{W}$, $\boldsymbol{\lambda} = \{\lambda_1, \ldots, \lambda_k\}^T$
Output: λ^*
Initialize $\mathbf{s}$, $\widetilde{\mathbf{f}}$, $\mathbf{f}$, $\mathbf{F}$, $\mathbf{g}$, $\mathbf{G}$, $\mathbf{L}_i$, $\widetilde{\mathbf{L}}_i$, $i = 1, \ldots, n$;
foreach λ *in* $\boldsymbol{\lambda}$ **do**
  $\widetilde{\mathbf{d}} \leftarrow \text{diag}([\mathbf{I} + \lambda\text{diag}(\mathbf{s})]^{-1})$;
  $I \leftarrow -2\widetilde{\mathbf{d}}^T(\widetilde{\mathbf{f}} \odot \mathbf{f})$;
  $II \leftarrow (\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}})^T(\mathbf{F}^T\mathbf{F})(\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}})$;
  $III \leftarrow 2\widetilde{\mathbf{d}}^T\mathbf{g}$;
  $IV \leftarrow -4\widetilde{\mathbf{d}}^T\mathbf{G}\widetilde{\mathbf{d}}$;
  $V \leftarrow 2\widetilde{\mathbf{d}}^T\left[\sum_{i=1}^n \left\{\mathbf{L}_i(\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}})\right\} \odot \left\{\widetilde{\mathbf{L}}_i(\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}})\right\}\right]$;
  iGCV $\leftarrow I + II + III + IV + V$;
end
$\lambda^* \leftarrow \arg\min_\lambda$ iGCV;

While the formula in Proposition 2 looks complex, it can be efficiently computed. Indeed, only the term $\widetilde{\mathbf{d}}$ depends on the smoothing parameter λ and it can be easily computed; all other terms

including $\mathbf{g}$ and $\mathbf{G}$ can be pre-calculated just once. Suppose the number of observations per subject is $m_i = m$ for all i. Let $K = c(c+1)/2 + 1$ and $M = m(m+1)/2$. Note that K is the number of unknown coefficients and M is the number of raw covariances from each subject. Then the pre-calculation of terms in the iGCV formula requires $O(nMK^2 + nM^2K + K^3 + M^3)$ computation time and each calculation of iGCV requires $O(nK^2)$ computation time. To see the efficiency of the simplified formula in Proposition 2, we note that a brute force evaluation of iCV in Lemma 1 requires computation time of the order $O(nM^3 + nK^3 + n^2M^2K)$, quadratic in the number of subjects n.

When the number of observations per subject m is small, i.e., $m < c$, the number of univariate basis functions, the iGCV computation time increases linearly with respect to m; when m is relatively large, i.e., $m > c$ but $m = o(n)$, then the iGCV computation time increases quadratically with respect to m. Therefore, the iGCV formula is most efficient with a small m, i.e., sparse data. As for the case that m is very large and the proposed method becomes very slow, then the method in Xiao et al. (2016) might be preferred.

1.4 Curve Prediction

In this section, we consider the prediction of $X_i(t) = f(t) + u_i(t)$, the ith subject curve. We assume that $X_i(t)$ is generated from a Gaussian process. Suppose we would like to predict $X_i(t)$ at $\{s_{i1}, \ldots, s_{im}\}$ for $m \geq 1$. Let $\mathbf{y}_i = (\mathbf{y}_{i1}, \ldots, \mathbf{y}_{im_i})^T$, $\mathbf{f}_i^o = \{f(t_{i1}), \ldots, f(t_{im_i})\}^T$, and $\mathbf{x}_i = \{X_i(s_{i1}), \ldots, X_i(s_{im})\}^T$. Let $\mathbf{H}_i^o = [\mathbf{b}(t_{i1}), \ldots, \mathbf{b}(t_{im_i})]^T$ and $\mathbf{H}_i^n = [\mathbf{b}(s_{i1}), \ldots, \mathbf{b}(s_{im})]^T$. It follows that

$$\begin{pmatrix} \mathbf{y}_i \\ \mathbf{x}_i \end{pmatrix} \sim \mathcal{N}\left\{ \begin{pmatrix} \mathbf{f}_i^o \\ \mathbf{f}_i^n \end{pmatrix}, \begin{pmatrix} \mathbf{H}_i^o \mathbf{\Theta} \mathbf{H}_i^{o,T} & \mathbf{H}_i^o \mathbf{\Theta} \mathbf{H}_i^{n,T} \\ \mathbf{H}_i^n \mathbf{\Theta} \mathbf{H}_i^{o,T} & \mathbf{H}_i^n \mathbf{\Theta} \mathbf{H}_i^{n,T} \end{pmatrix} + \sigma_\epsilon^2 \mathbf{I}_{m_i+m} \right\}.$$

We derive that

$$\mathbb{E}(\mathbf{x}_i | \mathbf{y}_i) = \left(\mathbf{H}_i^n \mathbf{\Theta} \mathbf{H}_i^{o,T} \right) \mathbf{V}_i^{-1} (\mathbf{y}_i - \mathbf{f}_i^o) + \mathbf{f}_i^n,$$

where $\mathbf{V}_i = \mathbf{H}_i^o \mathbf{\Theta} \mathbf{H}_i^{o,T} + \sigma_\epsilon^2 \mathbf{I}_{m_i}$, and

$$\mathrm{Cov}(\mathbf{x}_i|\mathbf{y}_i) = \mathbf{V}_i^n - \left(\mathbf{H}_i^n \mathbf{\Theta} \mathbf{H}_i^{o,T}\right) \mathbf{V}_i^{-1} \left(\mathbf{H}_i^n \mathbf{\Theta} \mathbf{H}_i^{o,T}\right)^T,$$

where $\mathbf{V}_i^n = \mathbf{H}_i^n \mathbf{\Theta} \mathbf{H}_i^{n,T} + \sigma_\epsilon^2 \mathbf{I}_m$. Because f, $\mathbf{\Theta}$ and σ_ϵ^2 are unknown, we need to plug in their estimates $\hat{f}$, $\widehat{\mathbf{\Theta}}$ and $\hat{\sigma}_\epsilon^2$, respectively, into the above equalities. Thus, we could predict $\mathbf{x}_i$ by

$$\hat{\mathbf{x}}_i = \{\hat{x}_i(s_{i1}), \ldots, \hat{x}_i(s_{im})\}^T = \left(\mathbf{H}_i^n \widehat{\mathbf{\Theta}} \mathbf{H}_i^{o,T}\right) \widehat{\mathbf{V}}_i^{-1} (\mathbf{y}_i - \hat{\mathbf{f}}_i^o) + \hat{\mathbf{f}}_i^n,$$

where $\hat{\mathbf{f}}_i^o = \{\hat{f}(t_{i1}), \ldots, \hat{f}(t_{im_i})\}^T$, $\hat{\mathbf{f}}_i^n = \{\hat{f}(s_{i1}), \ldots, \hat{f}(s_{im})\}^T$, and $\widehat{\mathbf{V}}_i = \mathbf{H}_i^o \hat{\mathbf{\Theta}} \mathbf{H}_i^{o,T} + \hat{\sigma}_\epsilon^2 \mathbf{I}_{m_i}$. Moreover, an approximate covariance matrix for $\hat{\mathbf{x}}_i$ is

$$\widehat{\mathrm{Cov}}(\hat{\mathbf{x}}_i|\mathbf{y}_i) = \widehat{\mathbf{V}}_i^n - \left(\mathbf{H}_i^n \widehat{\mathbf{\Theta}} \mathbf{H}_i^{o,T}\right) \widehat{\mathbf{V}}_i^{-1} \left(\mathbf{H}_i^n \widehat{\mathbf{\Theta}} \mathbf{H}_i^{o,T}\right)^T,$$

where $\widehat{\mathbf{V}}_i^n = \mathbf{H}_i^n \widehat{\mathbf{\Theta}} \mathbf{H}_i^{n,T} + \hat{\sigma}_\epsilon^2 \mathbf{I}_m$.

Note that one may also use the standard Karhunen-Loeve decomposition representation of $X_i(t)$ for prediction; see, e.g., Yao et al. (2005). An advantage of the above formulation is that we avoid the evaluation of the eigenfunctions extracted from the covariance function $\mathcal{C}$; indeed, we just need to compute the B-spline basis functions at the desired time points, which is computationally simple.

1.5 Simulations

1.5.1 Simulation Settings

We generate data using model (1.1). The number of observations for each random curve is generated from a uniform distribution on either $\{3, 4, 5, 6, 7\}$ or $\{j : 5 \leq j \leq 15\}$, and then observations are sampled from a uniform distribution in the unit interval. Therefore, on average, each curve has $m = 5$ or $m = 10$ observations. The mean function is $\mu(t) = 5\sin(2\pi t)$. For the covariance function $\mathcal{C}(s,t)$, we consider two cases. For case 1 we let $\mathcal{C}_1(s,t) = \sum_{\ell=1}^3 \lambda_\ell \psi_\ell(s) \psi_\ell(t)$, where ψ_ℓ's

are eigenfunctions and λ_ℓ's are eigenvalues. Here $\lambda_\ell = 0.5^{\ell-1}$ for $\ell = 1, 2, 3$ and $\psi_1(t) = \sqrt{2}\sin(2\pi t)$, $\psi_2(t) = \sqrt{2}\cos(4\pi t)$ and $\psi_3(t) = \sqrt{2}\sin(4\pi t)$. For case 2 we consider the Matern covariance function

$$C(d; \phi, \nu) = \frac{1}{2^{\nu-1}\Gamma(\nu)} \left(\frac{\sqrt{2\nu}d}{\phi} \right)^{\nu} K_\nu \left(\frac{\sqrt{2\nu}d}{\phi} \right)$$

with range $\phi = 0.07$ and order $\nu = 1$. Here K_ν is the modified Bessel function of order ν. The top two eigenvalues for this covariance function are 0.209 and 0.179, respectively. The noise term ϵ_{ij}'s are assumed normal with mean zero and variance σ_ϵ^2. We consider two levels of signal to noise ratio (SNR): 2 and 5. For example, if

$$\sigma_\epsilon^2 = \frac{1}{2} \int_{s=0}^{1} \int_{t=0}^{1} \mathcal{C}(s,t) \mathrm{d}s\mathrm{d}t,$$

then the signal to noise ratio in the data is 2. The number of curves is $n = 100$ or 400 and for each covariance function 200 datasets are drawn. Therefore, we have 16 different model conditions to examine.

1.5.2 Competing Methods and Evaluation Criterion

We compare the proposed method (denoted by FACEs) with the following methods: 1) The *fpca.sc* method in Goldsmith et al. (2011), which uses tensor-product bivariate P-splines (Eilers and Marx, 2003) for covariance smoothing and is implemented in the R package *refund*; 2) a variant of *fpca.sc* that uses thin plate regression splines for covariance smoothing, denoted by TPRS, and is coded by the authors; 3) the MLE method in Peng and Paul (2009), implemented in the R package *fpca*; and 4) the local polynomial method in Yao et al. (2003), denoted by *loc*, and is implemented in the MATLAB toolbox *PACE*. The underlying covariance smoothing R function for *fpca.sc* and TPRS is *gam* in the R package *mgcv* (Wood, 2013). For FACEs, we use $c = 10$ marginal cubic B-spline bases in each dimension. To evaluate the effect of the weight matrices in the proposed objective function (2), we also report results of FACEs without using weight matrices; we denote the one stage fit by FACEs (1-stage). For *fpca.sc*, we use its default setting, which uses 10 B-spline bases in each

dimension and the smoothing parameters are selected by "REML". We also code *fpca.sc* ourselves because the *fpca.sc* function in the *refund* R package incorporates other functionalities and may become very slow. For TPRS, we also use the default setting in *gam*, with the smoothing parameter selected by "REML". For bivariate smoothing, the default TPRS uses 27 nonlinear basis functions, in addition to the linear basis functions. We also consider TPRS with 97 nonlinear basis functions to match the basis dimension used in *fpca.sc* and FACEs. For the method MLE, we specify the range for the number of B-spline bases to be $[6, 10]$ and the range of possible ranks to be $[2, 6]$. We will not evaluate the method using a reduced rank mixed effects model (James et al., 2000) because it has been shown in Peng and Paul (2009) that the MLE method is more superior.

We evaluate the above methods using four criterions. The first is the integrated squared errors (ISE) for estimating the covariance function. The next two criterions are based on the eigendecomposition of the covariance function: $\mathcal{C}(s,t) = \sum_{\ell=1}^{\infty} \lambda_\ell \psi_\ell(s)\psi_\ell(t)$, where $\lambda_1 \geq \lambda_2 \geq \ldots$ are eigenvalues and $\psi_1(t), \psi_2(t), \ldots$ are the associated orthonormal eigenfunctions. The second criterion is the integrated squared errors (ISE) for estimating the top 3 eigenfunctions from the covariance function. Let $\psi(t)$ be the true eigenfunction and $\hat{\psi}(t)$ be an estimate of $\psi(t)$, then the integrated squared error is

$$\min \left[\int_{t=0}^{1} \{\psi(t) - \hat{\psi}(t)\}^2 \mathrm{d}t, \int_{t=0}^{1} \{\psi(t) + \hat{\psi}(t)\}^2 \mathrm{d}t \right].$$

It is easy to show that the range of integrated squared error for eigenfunction estimation is $[0, 2]$. Note that for the method MLE, if rank 2 is selected then only two eigenfunctions can be extracted. In this case, to evaluate accuracy of estimating the third eigenfunction, we will let ISE be 1 for a fair comparison. The third criterion is the squared errors (SE) for estimating the top 3 eigenvalues. The last criterion is the methods' computation speed.

1.5.3 Simulation Results

The detailed simulation results are presented in Appendix A.3. Here we provide summaries of the results along with some illustrations. In terms of estimating the covariance function, for most model conditions, FACEs gives the smallest medians of integrated squared errors and has the smallest

inter-quarter ranges (IQRs). MLE is the 2nd best for case 1 while *loc* is the 2nd best for case 2. See Figure 1.1 and Figure 1.2 for illustrations under some model conditions.

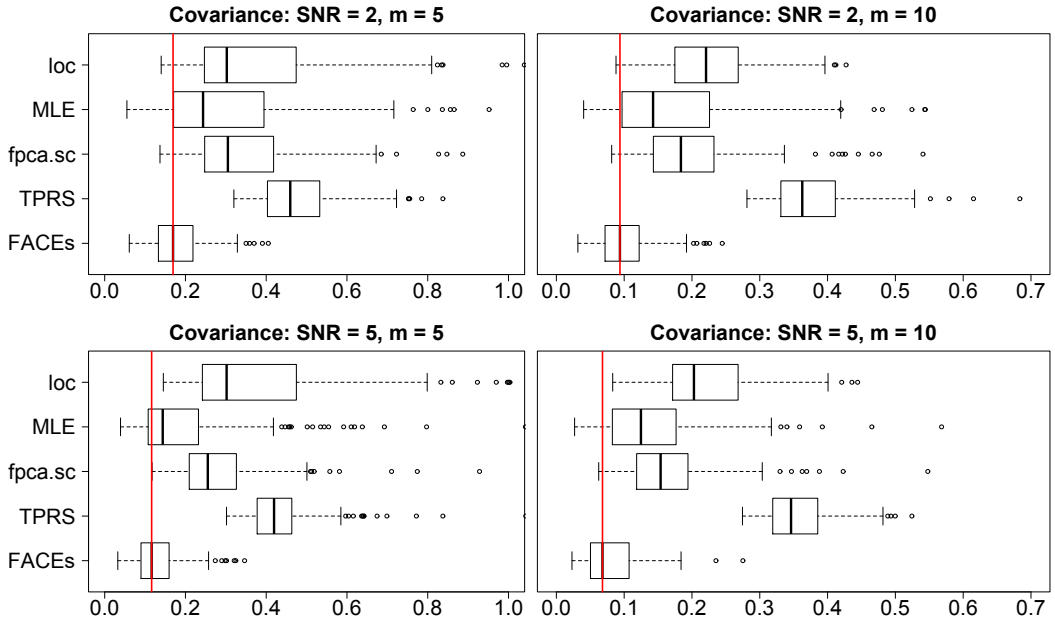


Figure 1.1: Boxplots of ISEs of five estimators for estimating the covariance functions of case 1, $n = 100$.

In terms of estimating the eigenfunctions, FACEs tends to outperform other approaches in most scenarios, while for the remaining scenarios, its performance is still comparable with the best one. MLE performs well for case 1 but relatively poorly for case 2, while the opposite is true for *loc*. TPRS and *fpca.sc* perform quite poorly for estimating the 2nd and 3rd eigenfunctions in both case 1 and case 2. Figure 1.3 illustrates the superiority of FACEs for estimating eigenfunctions when $n = 100$, $m = 5$.

As for estimation of eigenvalues, we have the following findings: 1) FACEs performs the best for estimating the first eigenvalue in case 1; 2) *loc* performs the best for estimating the first eigenvalue in case 2; 3) MLE performs overall the best for estimating 2nd and 3rd eigenvalues in both cases, while the performance of FACEs is very close and can be better than MLE under some model scenarios;

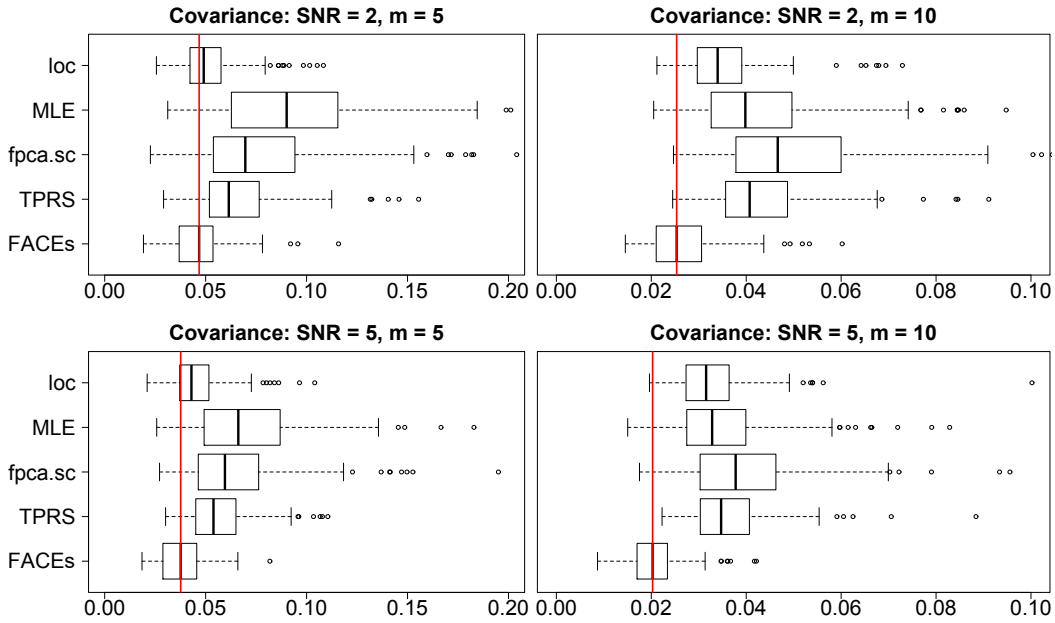


Figure 1.2: Boxplots of ISEs of five estimators for estimating the covariance functions of case 2, $n = 100$.

4) TPRS, *fpca.sc* and *loc* perform quite poorly for estimating the 2nd and 3rd eigenvalues in most scenarios. We conclude that FACEs shows overall very competitive performance and never deviates much from the best performance. Figure 1.4 illustrates the patterns of eigenvalue estimation for $n = 100$, $m = 5$.

We now compare run times of the various methods; see Figure 1.5 for an illustration. When $m = 5$, FACEs takes about four to seven times the computation times of TPRS and *fpca.sc*; but it is much faster than MLE and *loc*, the speed-up is about 15 and 35 folds, respectively. When $m = 10$, although FACEs is still slower than TPRS and *fpca.sc*, the computation times are similar ; computation times of MLE and *loc* are over 9 and 10 folds of FACEs, respectively. Because TPRS and *fpca.sc* are naive covariance smoothers, their fast speed is offset by their tendency to have inferior performance in terms of estimation of covariance functions, eigenfunctions, and eigenvalues.

Finally, by comparing results of FACEs with its 1-stage counterpart (see the online supplement), we see that taking into account of the correlations in the raw covariances boosts the estimation

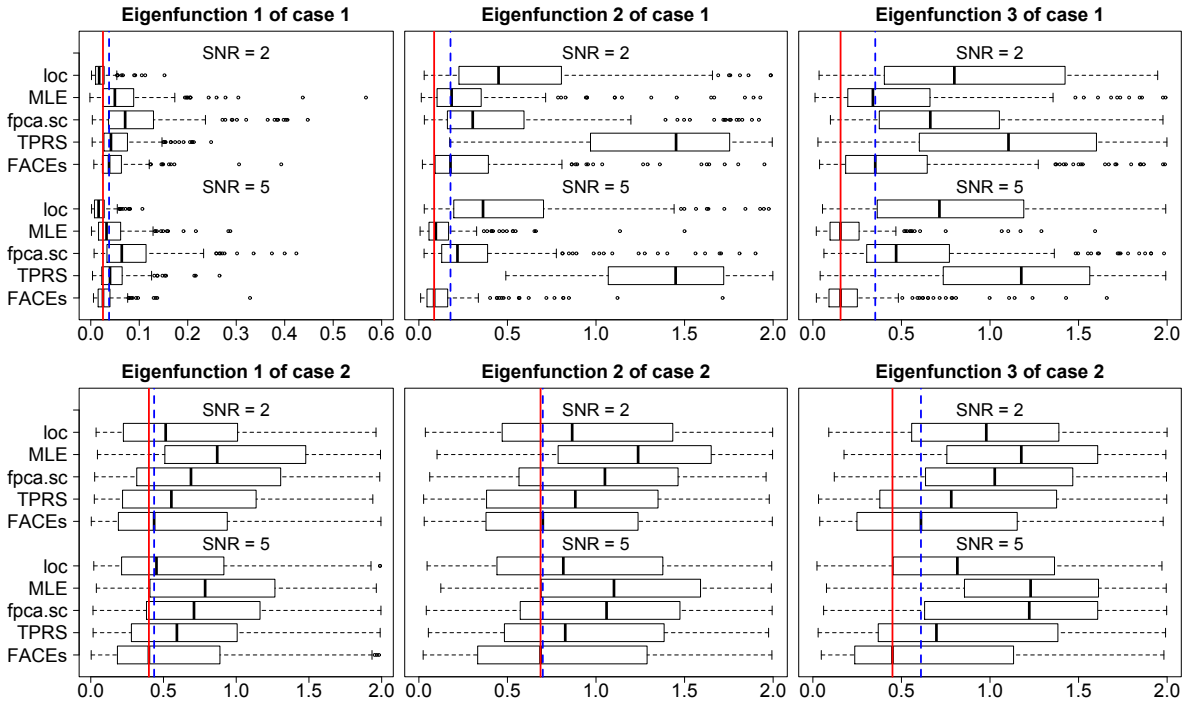


Figure 1.3: Boxplots of ISEs of five estimators for estimating the top 3 eigenfunctions when $n = 100$, $m = 5$. Note that the straight lines are the medians of FACEs when $SNR = 5$ and the dash lines are the medians of FACEs when $SNR = 2$.

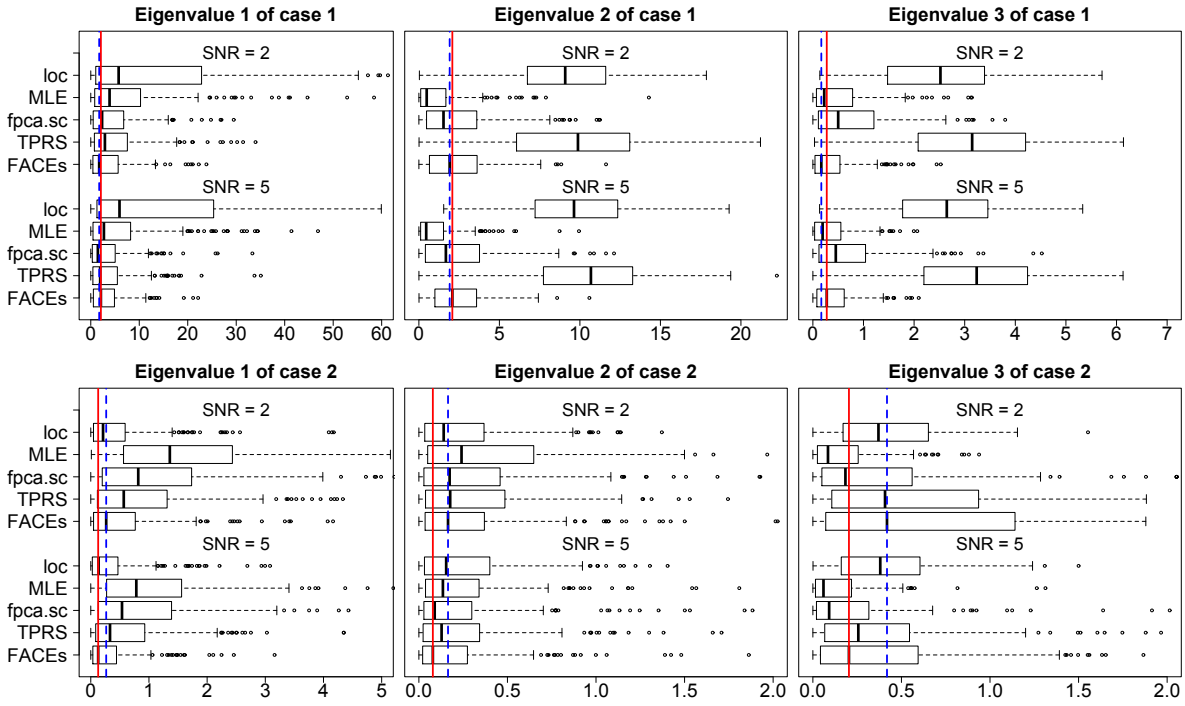


Figure 1.4: Boxplots of 100× SEs of five estimators for estimating the eigenvalues when $n = 100$, $m = 5$. Note that the straight lines are the medians of FACEs when $SNR = 5$ and the dash lines are the medians of FACEs when $SNR = 2$.

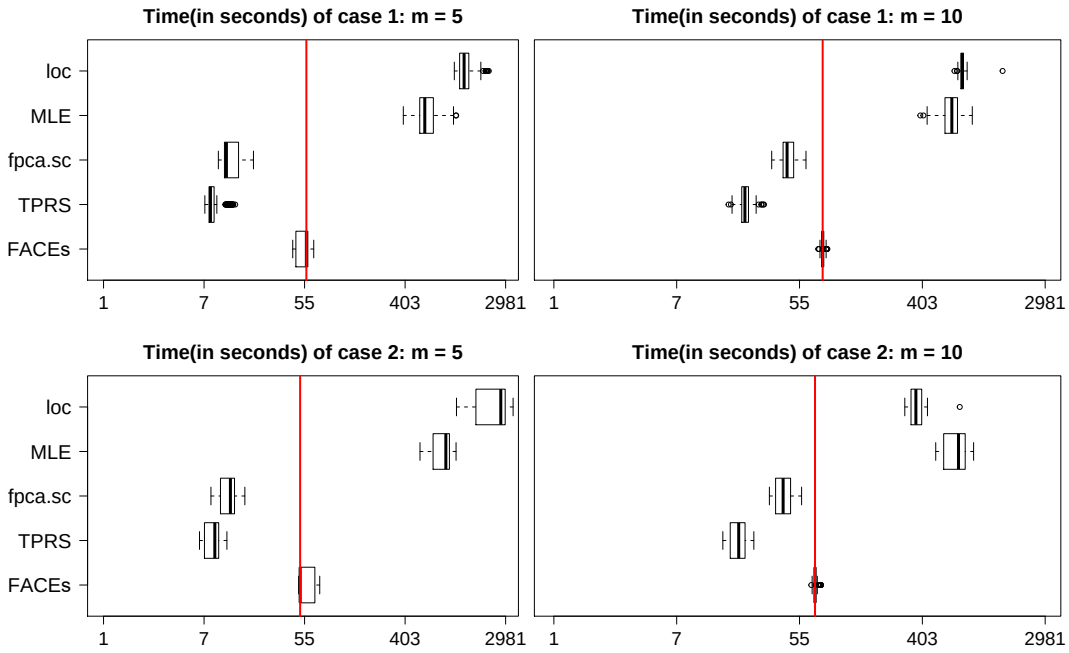


Figure 1.5: Boxplots of computation times (in seconds) of five estimators for estimating the covariance functions when $n = 400$, $SNR = 2$. Note that the x-axis is not equally spaced.

accuracies of FACEs a lot. The 1-stage FACEs is of course faster. It is interesting to note that the 1-stage FACEs is actually also very competitive against other methods.

To summarize, FACEs is a relatively fast method coupled with competing performance against the methods examined above.

1.5.4 Additional Simulations for Curve Prediction

We conduct additional simulations to evaluate the performance of the FACEs method for curve prediction. We focus on case 1 and use the same simulation settings in Section 1.5.1 for generating the training data and the testing data. We generate 200 new subjects for testing. The number of observations for the subjects are generated in the same way as the training data.

In addition to the conditional expectation approach outlined in Section 1.4, Cederbaum et al. (2016) proposed a new prediction approach (denoted by FAMM). As functional data have a mixed effects representation conditional on eigenfunctions, the standard prediction procedure for mixed

effects models can be used for curve prediction. The FAMM requires estimates of eigenfunctions and is applicable to any covariance smoothing method. Finally, direct estimation of subject-specific curves has also been proposed in the literature (Durban et al., 2005; Chen and Wang, 2011; Scheipl et al., 2015).

We will compare the following methods: 1) the conditional expectation method using FACEs; 2) the conditional expectation method using *fpca.sc*; 3) the conditional FAMM method using FACEs; 4) the conditional FAMM method using *fpca.sc*; 5) the conditional expectation method using *loc*; and 6) the spline-based approach in Scheipl et al. (2015) without estimating covariance function, denoted by *pffr*, and is implemented in the R package *refund*. This method uses direct estimation of subject-specific curves. For the conditional FAMM approach, we follow Cederbaum et al. (2016) and fix smoothing parameters at the ratios of the estimated eigenvalues and error variance from covariance function. Fixing smoothing parameters significantly reduces the computation times of the FAMM approach.

We evaluate the above methods using the integrated squared errors and the results are summarized in Table 1.1. The results show that either approach (conditional expectation or conditional FAMM) using FACEs has overall smaller prediction errors than competing approaches. The conditional FAMM approach using FACEs is slightly better than the conditional expectation approach. The results suggest that better estimation of the covariance function leads to more accurate prediction of subject-specific curves.

Table 1.1: Median and IQR (in parenthesis) of ISEs for curve fitting for case 1. The results are based on 200 replications. Numbers in boldface are the smallest of each row.

n	m	SNR	FACEs	*FAMM*(FACEs)	*fpca.sc*	*FAMM*(*fpca.sc*)	*loc*	*pffr*
100	5	2	0.714 (0.085)	**0.699** (0.102)	0.790 (0.156)	0.765 (0.147)	0.826 (0.135)	1.178 (0.092)
400	5	2	**0.592** (0.058)	0.596 (0.058)	0.625 (0.077)	0.639 (0.076)	0.735 (0.082)	1.181 (0.093)
100	10	2	0.369 (0.047)	**0.355** (0.044)	0.420 (0.066)	0.405 (0.069)	0.456 (0.076)	0.880 (0.060)
400	10	2	0.323 (0.027)	**0.317** (0.031)	0.330 (0.036)	0.336 (0.035)	0.406 (0.042)	0.872 (0.065)
100	5	5	0.497 (0.074)	**0.476** (0.082)	0.617 (0.171)	0.585 (0.147)	0.636 (0.106)	1.080 (0.109)
400	5	5	0.375 (0.042)	**0.372** (0.042)	0.416 (0.060)	0.419 (0.055)	0.523 (0.066)	1.050 (0.101)
100	10	5	0.218 (0.044)	**0.202** (0.040)	0.259 (0.056)	0.246 (0.053)	0.294 (0.058)	0.734 (0.071)
400	10	5	0.164 (0.019)	**0.160** (0.021)	0.182 (0.028)	0.180 (0.026)	0.243 (0.034)	0.740 (0.066)

1.6 Data Applications

We illustrate the proposed method on a publicly available dataset. Another application on a child growth dataset is provided in Appendix A.4.

CD4 cells are a type of white blood cells that could send signals to the human body to activate the immune response when they detect viruses or bacteria. Thus, the CD4 count is an important biomarker used for assessing the health of HIV infected persons as HIV viruses attack and destroy the CD4 cells. The dataset analyzed here is from the Multicenter AIDS Cohort Study (MACS) and is available in the *refund* R package (Huang et al., 2015). The observations are CD4 cell counts for 366 infected males in a longitudinal study (Kaslow et al., 1987). With a total of 1888 data points, each subject has between 1 and 11 observations. Statistical analysis based on this or related datasets were done in Diggle et al. (1994), Yao et al. (2005), Peng and Paul (2009) and Goldsmith et al. (2013).

For our analysis we consider log (CD4 count) since the counts are skewed. We plot the data in Figure 1.6 where the x-axis is months since seroconversion (i.e., the time at which HIV becomes detectable). The overall trend seem to be decreasing, as can be visually confirmed by the estimated mean function plotted in Figure 1.6. The estimated variance and correlation functions are displayed in Figure 1.7. It is interesting to see that the minimal value of the estimated variance function occurs at month 0 since seroconversion. Finally we display in Figure 1.8 the predicted trajectory of log (CD4 count) for 4 males and the corresponding pointwise confidence bands.

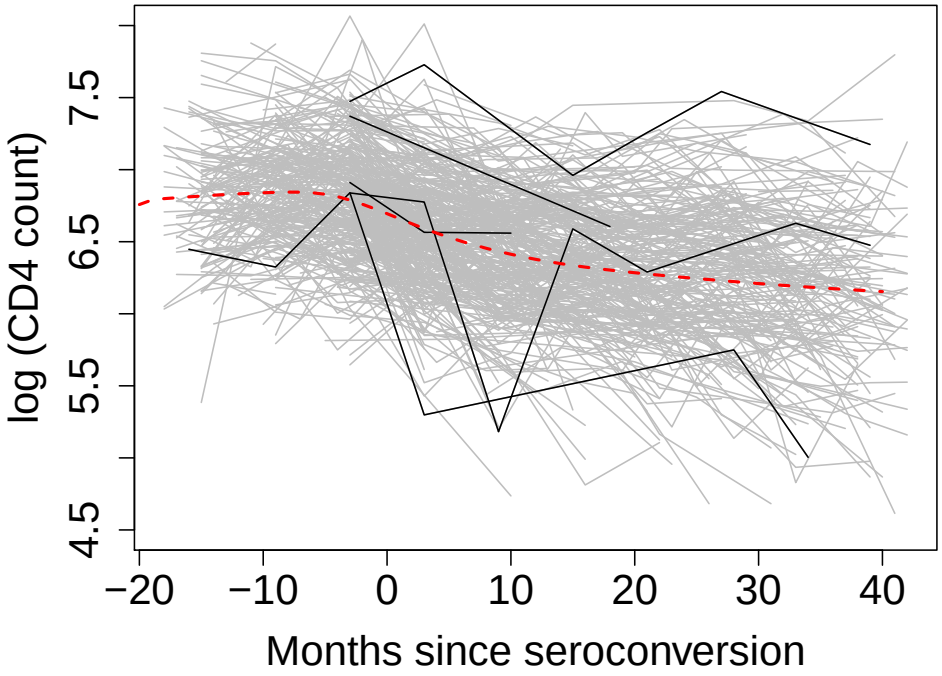


Figure 1.6: Observed log (CD4 count) trajectories of 366 HIV-infected males. The estimated population mean is the black solid line.

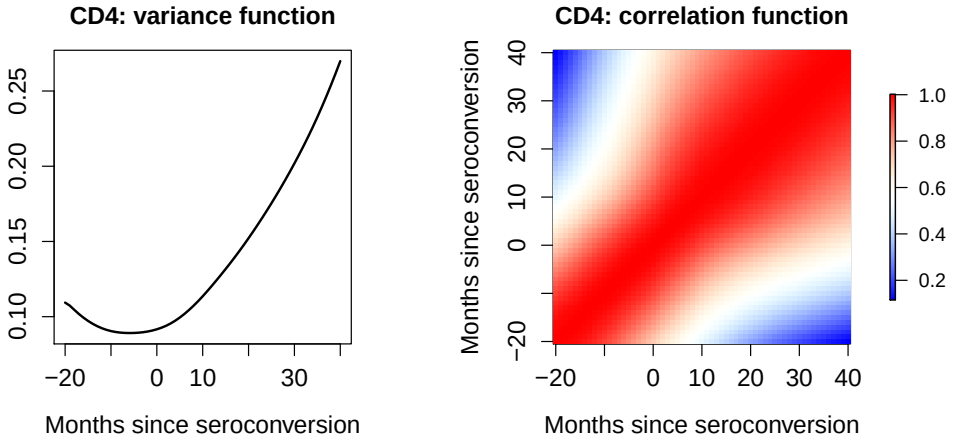


Figure 1.7: Estimated variance function (left panel) and correlation function (right panel) for the log (CD4 count).

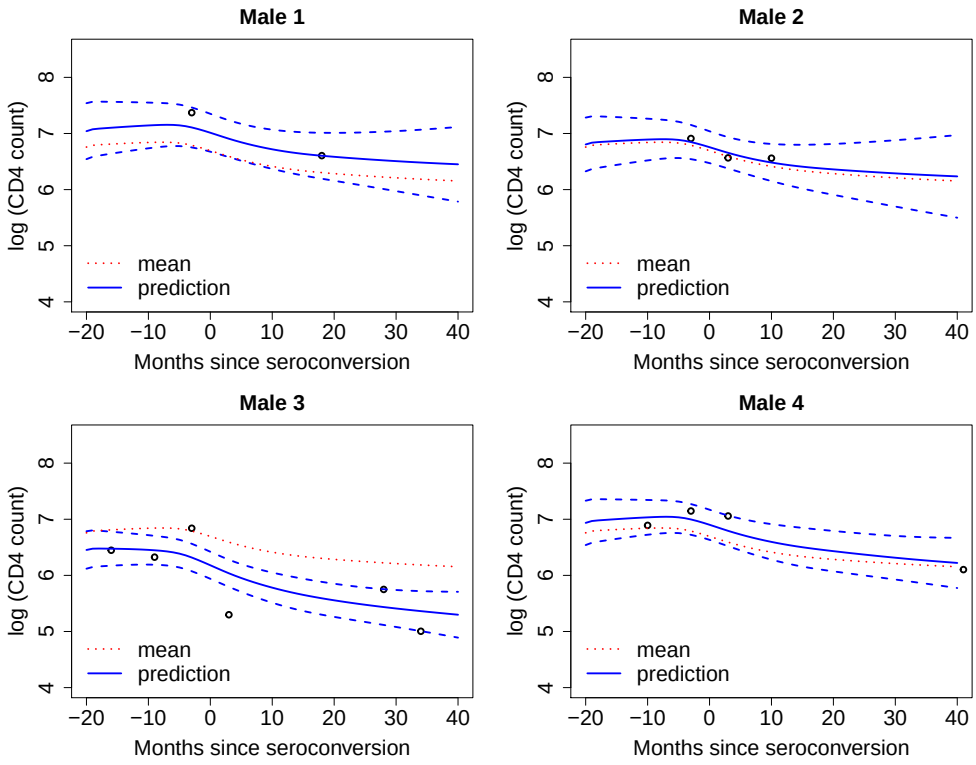


Figure 1.8: Predicted subject-specific trajectories of log (CD4 count) and associated 95% confidence bands for 4 males. The estimated population mean is the dotted line.

1.7 Discussion

Estimating and smoothing covariance operators is an old problem with many proposed solutions. Automatic and fast covariance smoothing is not fully developed and, in practice, one still does not have a method that is used consistently. The reason why the practical solution to the problem has been quite elusive is the lack of automatic covariance smoothing software. The novelty of our proposal is that it directly tackles this problem from the point of view of practicality. Here we proposed a method that we are already using extensively in practice and which is becoming increasingly popular among practitioners.

The ingredients of the proposed approach are not all new, but their combination leads to a

complete product that can be used in practice. The fundamentally novel contributions that make everything practical are: 1) use a particular type of penalty that respects the covariance matrix format; 2) provide a very fast fitting algorithm for leave-one-subject-out cross validation; and 3) ensure the scalability of the approach by controlling the overall complexity of the algorithm.

Smoothing parameters are an important component in smoothing and usually selected by either cross validation or likelihood-based approaches. The latter make use of the mixed model representation of spline-based smoothing (Ruppert et al., 2003) and tend to perform better than cross validation (Reiss and Todd Ogden, 2009; Wood, 2011). New optimization techniques have been developed (Rodríguez-Álvarez et al., 2015; Wood and Fasiolo, 2017) for likelihood-based selection of smoothing parameters. Likelihood-based approaches seem impractical to smoothing of raw covariances because the entries are products of normal residuals. Moreover, the raw covariances are not dependent within subjects, which imposes additional challenge. Developing some kind of likelihood-based selection of smoothing parameters for covariance smoothing is of interest but beyond the scope of the paper.

To make methods transparent and reproducible, the method has been made publicly available in the *face* package and will be incorporated in the function *fpca.face* in the *refund* package later. The current *fpca.face* function (Xiao et al., 2016) deals with high-dimensional functional data observed on the same grid and has been used extensively by our collaborators. We have a long track-record of releasing functional data analysis software and the final form of the function will be part of the next release of *refund*.

Chapter 2

Optimal Design for Classification of Functional Data[1]

2.1 Introduction

Functional data analysis (FDA) deals with data that take values over a continuum, such as curves over a time interval and surfaces over a spatial field. During the last two decades, FDA has become popular in statistics and found applications in a wide range of fields, including neuroscience (Reiss and Ogden, 2010; Lindquist, 2012; Goldsmith et al., 2012; Jiang et al., 2016), genetics and genomics (Leng and Müller, 2006; Reimherr and Nicolae, 2014, 2016), chemometrics (Ferraty and Vieu, 2002; Ferraty et al., 2010) and wearable computing (Morris et al., 2006; Xiao et al., 2015). See Ramsay and Silverman (2002, 2005), Ferraty and Vieu (2006) and Horváth and Kokoszka (2012) for a comprehensive treatment of this topic.

We consider functional data that are univariate functions and can be indexed by a time interval. Generally there are two types of functional data. When each function is observed at a common grid and the grid is dense, we say the data are "dense functional data". When each function is observed at only a few sampling time points that differ from curve to curve, we say the data are "sparse

[1]This chapter is based on a joint work with Luo Xiao.

functional data"; see, e.g., Yao et al. (2003). It is interesting to see that many functional data methods have been developed to deal with only one type of functional data. Thus it is important to develop methods that could handle both dense and sparse functional data.

Here, we are concerned with the optimal sampling design for classification of functional data. The goal is to determine sampling time points so that functional data observed at these time points can be classified accurately. Specifically, given a pilot study with functional data that have known classes and may be either densely observed or sparsely observed, we are interested in selecting the optimal sampling time points to collect observations for a new subject based on which the subject's class can be accurately predicted. A closely related problem is how many sampling time points are needed. This problem might be important because data collection can be costly, especially in longitudinal studies.

Classification for functional data has been well studied in the literature, see, e.g., James and Hastie (2001), Ferraty and Vieu (2003), Müller (2005), Leng and Müller (2006), Zhu et al. (2012), Delaigle and Hall (2012), Delaigle et al. (2012), Delaigle and Hall (2013), Galeano et al. (2015), Meister (2016) and Dai et al. (2016). In contrast, optimal design for classification has received relatively little attention. While not directly aiming for sampling design, recently Delaigle et al. (2012) proposed cross-validation-based methods to select a few observations from functional data and use them for classification or clustering. However, their method is restricted to data that are observed on a common grid and is not applicable to irregularly observed sparse functional data, such as the data shown in Figure 2.1, a focus of this paper. Additional drawbacks of the method in Delaigle et al. (2012) include intense computation and lack of theoretical results.

We propose an optimal design method for classification of functional data using linear discriminant analysis (LDA). The proposed method only utilizes the mean and covariance functions of functional data, which can be estimated from the pilot functional data that are either observed on a common grid or at sampling points varying between subjects. To the best of our knowledge, this is the first proposed design for functional data classification that is applicable to both types of functional data. Moreover, we formulate an explicit design objective as a function of the

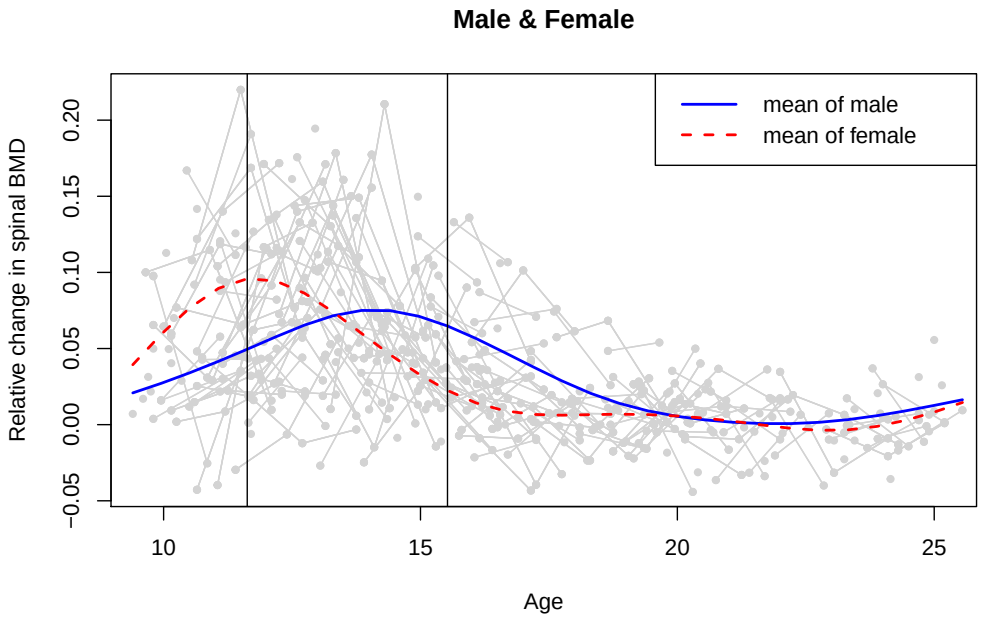


Figure 2.1: Spaghetti plot for relative spinal bone mineral density of 261 North American adolescents from a longitudinal study (Bachrach et al., 1999); see Section 2.6.1 for more details. The black vertical bars indicate two most predictive sampling points identified by the proposed optimal design method for classifying sex of subjects.

sampling points, which provides both theoretical and computational improvements over existing methods. Indeed, with the proposed design objective function, the optimal design can be easily implemented and the theoretical properties can be established. For example, we have established the monotonicity (see Theorem 1) of the design objective function when the number of sampling points is increased. Finally, we provide a practical and easy-to-implement method for determining the number of sampling points.

Our method is related to some recent developments in optimal sampling design for functional data and multivariate data. Optimal design for functional data (Ferraty et al., 2010; Wu, 2013; Ji and Müller, 2016; Park et al., 2016) have been proposed to predict either or both individual curves and a scalar outcome. For multivariate data, Dobbin and Simon (2007); Sánchez et al. (2016) considered the design for high dimensional multivariate data classification and focused on the sample size calculation. In contrast, the design for functional data, such as in our case, is to determine the sampling time points where data will be collected.

The rest of the chapter is organized as follows. In Section 2.2, we describe the linear discriminant analysis for functional data classification, propose two optimal designs for the classification of functional data, and discuss model estimation via a pilot study. In Section 2.3, we study the theoretical properties of the proposed designs. In Section 2.4, we discuss the implementation details of the proposed designs. We assess the empirical performance of the proposed designs via simulations in Section 2.5 and extensive real data applications in Section 2.6. All the technical proofs are enclosed in the Appendix.

2.2 Methodology

2.2.1 Linear Discriminant Analysis for Functional Data

Consider a random function $X(t)(t \in \mathcal{T})$ defined over a continuous time domain $\mathcal{T}$. Assume that $X(\cdot)$ follows a Gaussian process with mean function $\mu_I(t)$ and covariance function $\text{Cov}\{X(s), X(t)\} = r(s,t)$. Here I is the class indicator and takes the values of 0 or 1. This implies that the two classes

of functional data differ only in the mean functions. Let $\mathbf{t} = (t_1, t_2, \ldots, t_p)^T \in \mathcal{T}^p$ be a vector of p time points and suppose that the observed functional data is contaminated with white noises, i.e.,

$$W(t_j) = X(t_j) + \epsilon_j, j = 1, 2, \ldots, p, \tag{2.1}$$

where ϵ_js are i.i.d. $\mathcal{N}(0, \sigma_\epsilon^2)$ and are uncorrelated with $X(\cdot)$. We assume that $X(\cdot)$ is square integrable in $\mathcal{T}$ and without loss of generality, we let $\mathcal{T} = [0, 1]$.

By Mercer's theorem, $r(s, t)$ admits the decomposition $\sum_{\ell=1}^{\infty} \lambda_\ell \phi_\ell(s)\phi_\ell(t)$, where λ_ℓs are the eigenvalues of the covariance function and are ordered decreasingly, and $\phi_\ell(\cdot)$s are the corresponding eigenfunctions satisfying $\int_{\mathcal{T}} \phi_\ell(t)\phi_{\ell'}(t)dt = 1_{\{\ell=\ell'\}}$. Here $1_{\{\ell=\ell'\}}$ is 1 if $\ell = \ell'$, and 0 otherwise.

Given $p \geq 1$ and assume that p observations will be collected from a function $X_{i^*}(\cdot)$, where i^* is the subject index, at the collection of sampling points $\mathbf{t}$. Denote by $\mathbf{W}_{i^*}(\mathbf{t}) = \{W_{i^*}(t_1), \cdots, W_{i^*}(t_p)\}'$ the noisy observations following model (2.1). Denote by $f_k(\mathbf{W}_{i^*})$ the conditional density function for $k \in \{0, 1\}$ and let π_k be the prior probability of class k. Given $\mathbf{W}_{i^*}$, its class I_{i^*} can be assigned according to the Bayes rule,

$$\mathbb{P}(I_{i^*} = k|\mathbf{W}_{i^*}) = \frac{f_k(\mathbf{W}_{i^*})\pi_k}{f_1(\mathbf{W}_{i^*})\pi_1 + f_0(\mathbf{W}_{i^*})\pi_0}.$$

The associated linear classifier $\lambda(\mathbf{W}_{i^*})$ is

$$\lambda(\mathbf{W}_{i^*}) = \log \frac{f_1(\mathbf{W}_{i^*})}{f_0(\mathbf{W}_{i^*})} + \log \frac{\pi_1}{1 - \pi_1},$$

and the classification rule is, if $\lambda(\mathbf{W}_{i^*}) > 0$, then subject i^* is classified to class 1; otherwise its class is 0.

Let $\boldsymbol{\mu}_0(\mathbf{t}) = \{\mu_0(t_1), \cdots, \mu_0(t_p)\}'$, $\boldsymbol{\mu}_1(\mathbf{t}) = \{\mu_1(t_1), \cdots, \mu_1(t_p)\}'$, and let $\boldsymbol{\Sigma}(\mathbf{t}) = \mathrm{Cov}\{\mathbf{W}_{i^*}(\mathbf{t})\}$ be of dimension $p \times p$. For simplicity, we suppress the dependence of $\boldsymbol{\mu}_0$, $\boldsymbol{\mu}_1$ and $\boldsymbol{\Sigma}$ on $\mathbf{t}$. Denote by $\boldsymbol{\Lambda}$ an infinite dimensional diagonal matrix with $\boldsymbol{\Lambda}_{\ell\ell} = \lambda_\ell$ and $\boldsymbol{\Phi}(\mathbf{t}) = [\phi_\ell(t_j)]_{1 \leq j \leq p, 1 \leq \ell \leq \infty}$. Then,

$\mathbf{\Sigma} = \mathbf{\Phi}(\mathbf{t})\mathbf{\Lambda}\mathbf{\Phi}(\mathbf{t})' + \sigma_\epsilon^2 \mathbf{I}_p$. It follows that the linear classifier takes the form

$$\lambda(\mathbf{W}_{i^*}) = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)'\mathbf{\Sigma}^{-1}\mathbf{W}_{i^*} - \frac{1}{2}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0)'\mathbf{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) + \log\frac{\pi_1}{1-\pi_1}. \tag{2.2}$$

2.2.2 Optimal Design for Classification of Functional Data

The design objective is to maximize the classification accuracy of a subject's class based on only a few observations collected from that subject. More specifically, we would like to determine p optimal sampling time points in $\mathcal{T}$ such that a classification method using observations collected at these points achieves the highest accuracy.

Using the linear discriminant classifier, we define the probability of correct classification (PCC) as

$$\text{PCC}(\mathbf{t}) = \pi_1 \mathbb{P}\{\lambda(\mathbf{W}_{i^*}) > 0 | I_{i^*} = 1\} + (1-\pi_1)\mathbb{P}\{\lambda(\mathbf{W}_{i^*}) < 0 | I_{i^*} = 0\},$$

where $\mathbf{W}_{i^*}$ is the vector of noisy observations at $\mathbf{t}$ and is defined in Section 2.2.1. Then the optimal design is defined as $\mathbf{t}^{\text{opt}} = (t_1^{\text{opt}}, \ldots, t_p^{\text{opt}})' := \arg\max_{\mathbf{t}\in\mathcal{T}^p} \text{PCC}(\mathbf{t})$. Note that in this paper we implicitly impose the assumption that $t_1^{\text{opt}} \leq \cdots \leq t_p^{\text{opt}}$ as any permutation of these time points leads to the same PCC. Similar to Dobbin and Simon (2007), we derive that

$$\text{PCC}(\mathbf{t}) = \pi_1 \Phi\left(\frac{\frac{1}{2}s(\mathbf{t}) + \log\frac{\pi_1}{1-\pi_1}}{\sqrt{s(\mathbf{t})}}\right) + (1-\pi_1)\Phi\left(\frac{\frac{1}{2}s(\mathbf{t}) - \log\frac{\pi_1}{1-\pi_1}}{\sqrt{s(\mathbf{t})}}\right), \tag{2.3}$$

where $\Phi(\cdot)$ is the cumulative distribution function of the standard normal and

$$s(\mathbf{t}) = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)'\mathbf{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0), \tag{2.4}$$

which is the squared Mahalanobis distance. When $\pi_1 = 1/2$, $\text{PCC}(\mathbf{t})$ reduces to $\text{PCC}(\mathbf{t}) = \Phi\left(\frac{1}{2}\sqrt{s(\mathbf{t})}\right)$.

An alternative design objective criteria could be the area under the receiver operating char-

acteristic curve (AUROC). Specifically, we write the linear combination of noisy observations as $U = \boldsymbol{\beta}'\mathbf{W}_{i^*}$, where $\boldsymbol{\beta} = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)$. The two classes are corresponding to two populations of scalars U_1 and U_0 with density functions f_{U_1} and f_{U_0}, respectively. Define the true positive rate (TPR) as $\mathbb{P}\{\lambda(\mathbf{W}_{i^*}) > 0 | I_{i^*} = 1\} = \mathbb{P}(U_1 > c)$ and the false positive rate (FPR) as $\mathbb{P}\{\lambda(\mathbf{W}_{i^*}) > 0 | I_{i^*} = 0\} = \mathbb{P}(U_0 > c)$, where $c = (\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0)'\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)/2 - \log\{\pi_1/(1-\pi_1)\}$. It can be shown that

$$\text{AUROC}(\mathbf{t}) = \int_0^1 \text{TPR}\; d(\text{FPR}) = \mathbb{P}(U_1 > U_0).$$

And the corresponding optimal design is $\mathbf{t}^{\text{opt}} := \arg\max_{\mathbf{t}\in\mathcal{T}^p} \text{AUROC}(\mathbf{t})$. It follows that

$$\text{AUROC}(\mathbf{t}) = \Phi\left(\frac{1}{\sqrt{2}}\sqrt{s(\mathbf{t})}\right). \tag{2.5}$$

2.2.3 Model Estimation via a Pilot Study

The design objective functions requires the mean functions $\mu_0(t)$, $\mu_1(t)$, covariance function $r(s,t)$, error variance σ_ϵ^2, as well as the class prevalence rates π_0 and π_1. In practice, these model components are unknown and have to be estimated from a pilot study.

Suppose that the functional data of a pilot study are $\{t_{ij}, W_i(t_{ij})\}_{1\le i\le n, 1\le j\le m_i}$, where i indexes subjects, j indexes repeated observations, n is the number of subjects and m_i is the number of observations for subject i. For the case of dense data, we have $m_i = m$, $t_{ij} = t_i$. Similar to before, let I_i indicate the class for subject i, $I_i = 0$.

The prior probabilities π_0 and π_1 can be estimated by the proportions of subjects in each class in the pilot data. The mean functions can be estimated by a univariate smoother applied separately to the pilot data from each class. We use penalized splines (Eilers and Marx, 1996). Many functional principal components analysis (fPCA) approaches are available to estimate the covariance function for sparse functional data, e.g., local polynomial regression (Yao et al., 2005), mixed effects model (James et al., 2000), and geometric PCA (Peng and Paul, 2009). We use the fast covariance estimation method (FACEs) of Xiao et al. (2017a) to estimate the covariance function $r(s,t)$ and σ_ϵ^2, which was found to be more accurate and computationally efficient for

sparse functional data and is implemented in the R package *face* (Xiao et al., 2017b). For densely observed functional data, we use the sandwich smoother proposed in Xiao et al. (2016), which is implemented in the function *fpca.face* in the R package *refund* (Huang et al., 2015). The sandwich smoother is computationally fast and scalable to high dimensional functional data.

With the estimates $\hat{\mu}_0(t)$, $\hat{\mu}_1(t)$, $\hat{r}(s,t)$, $\hat{\sigma}_\epsilon^2$, as well as $\hat{\pi}_0$ and $\hat{\pi}_1$, we plug the estimates into (2.2) to obtain the practical classifier, denoted by $\hat{\lambda}(\mathbf{W}_{i^*})$. Similarly, we plug the estimates into (2.3) and (2.5) to obtain $\widehat{\mathrm{PCC}}(\mathbf{t})$ and $\widehat{\mathrm{AUROC}}(\mathbf{t})$, respectively.

2.3 Theoretical Properties

In this section we study the properties of the two design objective functions $\mathrm{PCC}(\mathbf{t})$ and $\mathrm{AUROC}(\mathbf{t})$ defined in (2.3) and (2.5), respectively. The practical implications of the theoretical results for the use of the estimated design objective functions will be discussed at the end of this section. We assume that $\sum_{k=1}^{\infty} \lambda_k < \infty$, i.e., $X(\cdot)$ is square integrable. All the proofs are provided in Appendix B. As both objective functions depend on $s(\mathbf{t})$ in (2.4), we first introduce a lemma to characterize $s(\mathbf{t})$.

Lemma 2. *Suppose* $\mathbf{t} \in \mathcal{T}^p$, $\widetilde{\mathbf{t}} \in \mathcal{T}^{p+c}$ *for some fixed integer* $c > 0$ *and* $\mathbf{t} \subseteq \widetilde{\mathbf{t}}$, *then* $s(\mathbf{t}) \leq s(\widetilde{\mathbf{t}})$.

Theorem 1 (*Monotonicity*)**.** *Suppose* $\mathbf{t} \in \mathcal{T}^p$, $\widetilde{\mathbf{t}} \in \mathcal{T}^{p+c}$ *for some fixed integer* $c > 0$ *and* $\mathbf{t} \subseteq \widetilde{\mathbf{t}}$, *then (i)* $\mathrm{PCC}(\mathbf{t}) \leq \mathrm{PCC}(\widetilde{\mathbf{t}})$*; (ii)* $\mathrm{AUROC}(\mathbf{t}) \leq \mathrm{AUROC}(\widetilde{\mathbf{t}})$.

Theorem 1 implies that more observations per function, i.e. bigger p, result in larger values of the objective functions $\mathrm{PCC}(\mathbf{t})$ and $\mathrm{AUROC}(\mathbf{t})$.

Theorem 2. *Fix* p, *for any* $t_p^* \in \mathcal{T}^p$ *that maximizes* $\mathrm{AUROC}(\mathbf{t})$, *it also maximizes* $\mathrm{PCC}(\mathbf{t})$ *with any* $0 < \pi_0 = 1 - \pi_1 < 1$.

The proof of Theorem 2 is straightforward by using the fact that PCC is strictly increasing as a function of $\mathbf{s}(\mathbf{t})$; see the expression of PCC in (2.3) and the proof of Theorem 1. Theorem 2 implies that in terms of optimal sampling time points, the relative ratio of the two classes of functional

data does not matter and the optimal sampling points determined by AUROC are the same as those determined by PCC. Thus, in the simulations and data applications, we shall focus only on PCC.

Theorem 3 (*Limit*)**.** *Suppose that the assumptions in the Appendix B hold. For the fixed design where* $\mathbf{t}_p = \{0, 1/p, 2/p, \cdots, (p-1)/p\}'$*, let* $\mu(t) = \mu_1(t) - \mu_0(t)$ *and assume* $\mu(t) = \sum_{k\geq 1} \phi_k(t)\mu_k$*. Then*

(i) $\lim\limits_{p\to\infty,\, p\in\mathbb{N}} \mathrm{PCC}(\mathbf{t}_p) = \pi_1 \Phi\left(\frac{\frac{1}{2}s_{\max} + \log\frac{\pi_1}{1-\pi_1}}{\sqrt{s_{\max}}}\right) + (1-\pi_1)\Phi\left(\frac{\frac{1}{2}s_{\max} - \log\frac{\pi_1}{1-\pi_1}}{\sqrt{s_{\max}}}\right)$*;*

(ii) $\lim\limits_{p\to\infty,\, p\in\mathbb{N}} \mathrm{AUROC}(\mathbf{t}_p) = \Phi\left(\frac{1}{\sqrt{2}}\sqrt{s_{\max}}\right)$*,*

where $s_{\max} = \sum_{k\geq 1} \lambda_k^{-1}\mu_k^2$*.*

Corollary 1. *When* $\pi_1 = 1/2$*, the limit of PCC simplifies to* $\Phi\left(\frac{1}{2}\sqrt{s_{\max}}\right)$*, which was obtained in Delaigle and Hall (2012) for classification of fully observed functional data.*

When $s_{\max} = \infty$, then the limits of PCC and AUROC are both 1 and near perfect classification is achievable. This phenomenon was discovered by Delaigle and Hall (2012). In practice, the limits may be estimated by plugging the estimates of $s_{\max}$ and π_1 into the expressions in Theorem 3.

Because we use penalized spline estimators for model estimation and the theoretical properties of penalized spline estimators in functional data analysis remain largely unknown, we are unable to establish any theoretical results relating $\widehat{\mathrm{PCC}}$ to PCC. However, for the local polynomial method (Yao et al., 2005), the uniform consistencies of all model components in functional principal component analysis have been well established; see Li and Hsing (2010) and Zhang and Wang (2016). Thus, if the local linear regression is used for model estimation, the uniform consistency of $\widehat{\mathrm{PCC}}(\mathbf{t}_p)$ as an estimator of $\mathrm{PCC}(\mathbf{t}_p)$ over $\mathbf{t}_p \in \mathcal{T}^p$ can be easily proved with appropriate assumptions. Furthermore, one can establish the following: Fix p and let $\mathbf{t}_p^*$ be the p optimal sampling time points that maximize the true objective function $\mathrm{PCC}(\cdot)$ and let $\widehat{\mathbf{t}}_p^*$ be the p selected sampling points that maximize the estimated objective function $\widehat{\mathrm{PCC}}(\cdot)$. Then, $\mathrm{PCC}(\widehat{\mathbf{t}}_p^*)$ converges to $\mathrm{PCC}(\mathbf{t}_p^*)$ uniformly. We shall empirically illustrate the convergence of $\mathrm{PCC}(\widehat{\mathbf{t}}_p^*)$ to $\mathrm{PCC}(\mathbf{t}_p^*)$ via simulation studies.

The theoretical results provide insight into the behavior of the estimated design objective function and guide our simulation studies. For example, we shall study if the estimated optimal sampling points will lead to larger PCCs when the number of estimated sampling points increases. The theorem also motivates two practical criteria for determining the number of estimated sampling points; see Section 2.4.2.

2.4 Implementation

2.4.1 Search of Optimal Sampling Points

We fix the number of sampling time points to be selected as p; the selection of p will be discussed in Section 2.4.2. In the literature, usually a grid of candidate sampling points is predetermined. If p is small, a full search can be conducted. Otherwise if p is large, Ferraty et al. (2010) and Ji and Müller (2016) used a greedy search algorithm and Wu (2013) proposed a Metropolis sampling method. Here, we propose a computationally efficient approach for finding optimal sampling points, which does not require the specification of candidate sampling points. The proposed approach exploits the fact that our design objective function is an explicit analytic function of the sampling points $\mathbf{t}$ in (2.4) (see equation (2.3)).

First note that by the proof of Theorem 1, maximizing $\mathrm{PCC}(\mathbf{t})$ is equivalent to maximizing $s(\mathbf{t})$. Thus, below we focus on $s(\mathbf{t})$.

Theorem 4. *Assume that the mean functions $\mu_0(t)$, $\mu_1(t)$ and the covariance function $r(s,t)$ are differentiable. Let $d(t) = \mu_1(t) - \mu_0(t)$. Then*

$$\frac{\partial s(\mathbf{t})}{\partial \mathbf{t}} = 2\{\mathbf{\Sigma}^{-1}(\mathbf{t})d(\mathbf{t})\} \odot \left\{\dot{d}(\mathbf{t}) - \dot{r}(\mathbf{t},\mathbf{t})\mathbf{\Sigma}^{-1}(\mathbf{t})d(\mathbf{t})\right\},$$

where $d(\mathbf{t}) = \{d(t_1), \ldots, d(t_p)\}'$, $\dot{d}(t)$ denotes the derivative of d, $\dot{d}(\mathbf{t}) = \{\dot{d}(t_1), \ldots, \dot{d}(t_p)\}'$, $\dot{r}(s,t)$ denotes the derivative of $r(s,t)$ with respect to s, $\dot{r}(\mathbf{t},\mathbf{t}) = \{\dot{r}(t_k,t_\ell)\}_{1\le k,\ell\le p} \in \mathbb{R}^{p\times p}$, and $\odot$ denotes the Hadamard pointwise product.

The proof is provided in Appendix B. The gradient of $s(\mathbf{t})$ derived in Theorem 4 can be easily evaluated when the model components are replaced by their estimated counterparts. Indeed, because we use spline methods for estimating the mean functions as well as the covariance function (Section 2.2.3), evaluations of the derivatives $\dot{d}(t)$ and $\dot{r}(s,t)$ are straightforward; see Lemma 6 in Appendix B. Note that local linear estimators, however, do not enjoy such computational convenience.

For a practical implementation, we use the *constrOptim* function in R to search for optimal sampling points with the *BFGS* method. In particular, given the invariance of PCC when the sampling points are permuted, we impose the linear constraint that $t_1 \leq t_2 \leq \cdots \leq t_p$. We find that the computation times are significantly reduced by using the above derived gradient and the linear constraints. However, the optimization method may still lead to sampling points that are only locally optimal. Thus, to help find global optimal sampling points, multiple starting points for the *constrOptim* function are provided and are found to work well.

2.4.2 Selection of Number of Sampling Points

In many real-data applications, the number of sampling time points p is not known *a priori* and hence needs to be determined as well.

Existing methods for determining the number of sampling points are mostly based on cross-validation methods. In the context of functional regression, Ferraty et al. (2010) proposed to increase the number of design points until the relative decreasing change of the cross-validation error is small. Thus, such a selection strategy is mainly designed for the regression setting in which the relative changes of prediction errors are small. But the relative change in PCC can be fairly large and new methods are needed. Similar ideas based on cross validation were also adopted in Delaigle et al. (2012) and Wu (2013).

We use the theoretical results derived in Section 2.3 to formulate two new selection criteria. According to Theorem 3, $\mathrm{PCC}(s_{\max})$ and $\mathrm{AUROC}(s_{\max})$ are the limits of the objective functions. Thus, a sensible approach is to select the smallest p so that the design objective function evaluated with those p optimal points achieves a high proportion of the limit. Let $0 < h < 1$, then the selection

method is

$$p_1^* := \min\left\{p \in \mathbb{N} : \min_{\mathbf{t}\in\mathcal{T}^p} \mathrm{PCC}(\mathbf{t}) \geq h\mathrm{PCC}(s_{\max})\right\}. \tag{2.6}$$

The above method does not directly incorporate the cost associated with collecting more sampling points, which may have to be taken into account for longitudinal studies. Thus, we propose a second penalized criterion,

$$p_2^* := \min\left\{p \in \mathbb{N} : \min_{\mathbf{t}\in\mathcal{T}^p} 1 - \mathrm{PCC}(\mathbf{t}) + p \cdot \delta\right\}. \tag{2.7}$$

The penalized criterion implies that the gain in PCC using any $p_2^* + 1$ sampling points is smaller than δ, compared to the highest PCC using p_2^* sampling points.

In practice, we use the estimated PCC and thus the two selection methods become

$$\hat{p}_1^* := \min\left\{p \in \mathbb{N} : \min_{\mathbf{t}\in\mathcal{T}^p} \widehat{\mathrm{PCC}}(\mathbf{t}) \geq h\widehat{\mathrm{PCC}}(\hat{s}_{\max})\right\}, \tag{2.8}$$

where $\hat{s}_{\max}$ is an estimate of $s_{\max}$ in (2.4), and

$$\hat{p}_2^* := \min\left\{p \in \mathbb{N} : \min_{\mathbf{t}\in\mathcal{T}^p} 1 - \widehat{\mathrm{PCC}}(\mathbf{t}) + p \cdot \delta\right\}. \tag{2.9}$$

2.5 Simulations

We carry out simulations to evaluate the performance of the proposed method for selecting optimal sampling time points. We focus on the design using PCC as the objective function and consider the following evaluation criteria.

(i) Fix p and let $\mathbf{t}_p^*$ be the p optimal sampling time points that maximize the true objective function $\mathrm{PCC}(\cdot)$ and let $\widehat{\mathbf{t}}_p^*$ be the p selected sampling points that maximize the estimated objective function $\widehat{\mathrm{PCC}}(\cdot)$. We define the absolute relative error,

$$ARE_p = \frac{|\mathrm{PCC}(\mathbf{t}_p^*) - \mathrm{PCC}(\widehat{\mathbf{t}}_p^*)|}{\mathrm{PCC}(\mathbf{t}_p^*)}. \tag{2.10}$$

We measure the closeness of PCC($\mathbf{t}_p^*$) and PCC($\widehat{\mathbf{t}}_p^*$), rather than $\mathbf{t}_p^*$ and $\widehat{\mathbf{t}}_p^*$ for two reasons. First, we aim to maximize PCC, therefore it is sensible to focus on the estimation accuracy of PCC itself. Second, the maximizer of PCC might not be unique and may cause identifiability problems.

(ii) In terms of the out-of-sample classification accuracy, similar to Ji and Müller (2016), we compare the performance of the proposed optimal design and two other designs: (a) random design, for which sampling points are randomly selected; (b) oracle design, where we assume that the true design objective function is known. The oracle design, while impractical, serves as a golden standard for evaluating the performance of the proposed design.

(iii) For fixed h and δ, p_1^* and p_2^* can be determined according to (2.6) and (2.7), respectively. We then select $\widehat{p}_1^*$ and $\widehat{p}_2^*$ via (2.8) and (2.9), respectively. We evaluate the performance of the two selection methods in terms of the proportion of correctly selecting the true number of sampling points, i.e., $\widehat{p}_1^* = p_1^*$ or $\widehat{p}_2^* = p_2^*$.

2.5.1 Simulation Settings

We generate data according to model (2.1) and only consider sparse functional data, with each class having the same number of subjects. Two cases of mean functions are considered (to be specified later). We generate $X_i(t) - \mu(t)$ by $\sum_{\ell=1}^{5} \xi_{i\ell}\phi_\ell(t)$, where $\{\phi_1, \cdots, \phi_5\}$ is a set of orthonormal eigenfunctions and $\xi_{i\ell}$ are assumed normal with mean zero and variance $\lambda_\ell = 10/2^\ell$, $\ell = 1, \cdots, 5$. We use a periodic covariance $r(s,t)$ induced by five Fourier bases: $\phi_\ell(t) = \sqrt{2}\sin\{(\ell+1)\pi t\}$ for odd ℓ and $\phi_\ell(t) = \sqrt{2}\cos(\ell\pi t)$ for even ℓ. The covariance function is periodic because $r(s,t) = r(1-s,t)$. The random errors ϵ_{ij}s are sampled independently from another normal distribution with mean zero and variance σ_ϵ^2. The variance σ_ϵ^2 is selected such that the signal to noise ratio is one, i.e., $\sum_{\ell=1}^{5} \lambda_\ell/\sigma_\epsilon^2 = 1$. The number of observations per subject m_i varies across subjects with mean m and the sampling time points t_{ij} are drawn from the uniform distribution in the unit interval. We design a factorial experiment with three factors:

- Mean functions:
  - (i) Case 1: $\mu_1 = 5\sin(2\pi t)$ and $\mu_0 = 3\sin(2\pi t)$.

– (ii) Case 2: $\mu_1 = 50t^{1.1}(1-t)$ and $\mu_0 = 40t(1-t)^{1.1}$.

- Number of observations per subject:

  (i) $m_i \sim Uniform\{3, \cdots, 7\}$ and (ii) $m_i \sim Uniform\{7, \cdots, 13\}$.

- Number of subjects for each class:

  (i) $n = 100$, (ii) $n = 400$ and (iii) $n = 800$.

Each generated dataset is accompanied by a testing dataset containing 1000 subjects from each class. Our experiment creates 12 different sets of model conditions. For each model condition, 200 Monte Carlo samples are drawn.

2.5.2 Simulation Results

Table 2.1 displays the medians and interquartile ranges (IQRs) of AREs and demonstrates that the predictive power of the selected optimal sampling time points using the estimated PCC is close to that of the true optimal sampling time points. The predictive power of the selected optimal sampling time points increases when either the sample size or the average number of observations per subject increases, due to the improved accuracy of the plugged in model estimates.

Table 2.1: Median and IQR (in parentheses) of AREs with different p under various model conditions.

Case	n	$p=2$		$p=3$		$p=4$		$p=5$	
		$m=5$	$m=10$	$m=5$	$m=10$	$m=5$	$m=10$	$m=5$	$m=10$
1	100	0.017 (0.035)	0.008 (0.013)	0.020 (0.030)	0.010 (0.018)	0.019 (0.020)	0.014 (0.012)	0.017 (0.015)	0.013 (0.012)
	400	0.003 (0.007)	0.001 (0.002)	0.004 (0.008)	0.002 (0.004)	0.009 (0.012)	0.004 (0.008)	0.009 (0.009)	0.006 (0.006)
	800	0.001 (0.002)	0.001 (0.002)	0.003 (0.004)	0.001 (0.002)	0.005 (0.010)	0.001 (0.006)	0.007 (0.008)	0.003 (0.005)
Case	n	$p=2$		$p=3$		$p=4$		$p=5$	
		$m=5$	$m=10$	$m=5$	$m=10$	$m=5$	$m=10$	$m=5$	$m=10$
2	100	0.006 (0.018)	0.001 (0.006)	0.010 (0.016)	0.004 (0.008)	0.013 (0.021)	0.007 (0.008)	0.012 (0.021)	0.004 (0.008)
	400	0.001 (0.006)	0.000 (0.001)	0.004 (0.009)	0.001 (0.003)	0.006 (0.010)	0.003 (0.006)	0.005 (0.011)	0.002 (0.005)
	800	0.001 (0.006)	0.000 (0.001)	0.003 (0.005)	0.001 (0.003)	0.005 (0.008)	0.001 (0.006)	0.004 (0.008)	0.002 (0.003)

Figure 2.2 presents the finite sample performance of the proposed (and estimated) design for classifying new subjects, for the model conditions with n = 800 and m = 10, along with the oracle

and random designs. We see that that the proposed optimal design performs comparably with the oracle design and outperforms the random design significantly. In addition, using more sampling points tends to improve the performance of the optimal design, which empirically confirms Theorem 1 for the estimated design. Indeed, from $p = 2$ to $p = 5$, the medians of classification accuracies using $\widehat{\mathbf{t}}_p^*$ in case 1 are 0.720, 0.728, 0.735 and 0.738, respectively. In case 2, the corresponding medians are 0.883, 0.922, 0.950 and 0.964, also in an increasing order.

Finally, we evaluate the finite sample performance of the two proposed criteria (2.8) and (2.9) for selecting the number of sampling time points. For case 1, we fix $h = 0.70$ and $\delta = 0.03$ so that the true number of optimal points is 2. For case 2, we fix $h = 0.90$ and $\delta = 0.03$ so that the true number of optimal points is 3. Table 2.2 shows the performance of the proposed methods for correctly selecting the number of optimal sampling points. With larger sample size and more observations per subject on average, the proportions of correct selection of the true number of optimal sampling points increase. The simulation results shows that the penalized criterion (2.7) is more accurate; we also find that it provides more stable performance as we vary δ slightly (results not shown). Therefore, we shall use the penalized criterion in the data applications.

Table 2.2: Proportions of correct selection of the true number of sampling points for the two proposed methods in (2.8) and (2.9).

	Case 1				Case 2			
n	Criterion (2.8)		Criterion (2.9)		Criterion (2.8)		Criterion (2.9)	
	$m = 5$	$m = 10$	$m = 5$	$m = 10$	$m = 5$	$m = 10$	$m = 5$	$m = 10$
100	0.35	0.53	0.65	0.75	0.47	0.55	0.74	0.81
400	0.78	0.93	0.84	0.91	0.49	0.76	0.70	0.92
800	0.93	0.97	0.92	0.99	0.73	0.83	0.82	0.97

2.6 Data Applications

In this section, we apply the proposed optimal design to five publicly available datasets, which we introduce below.

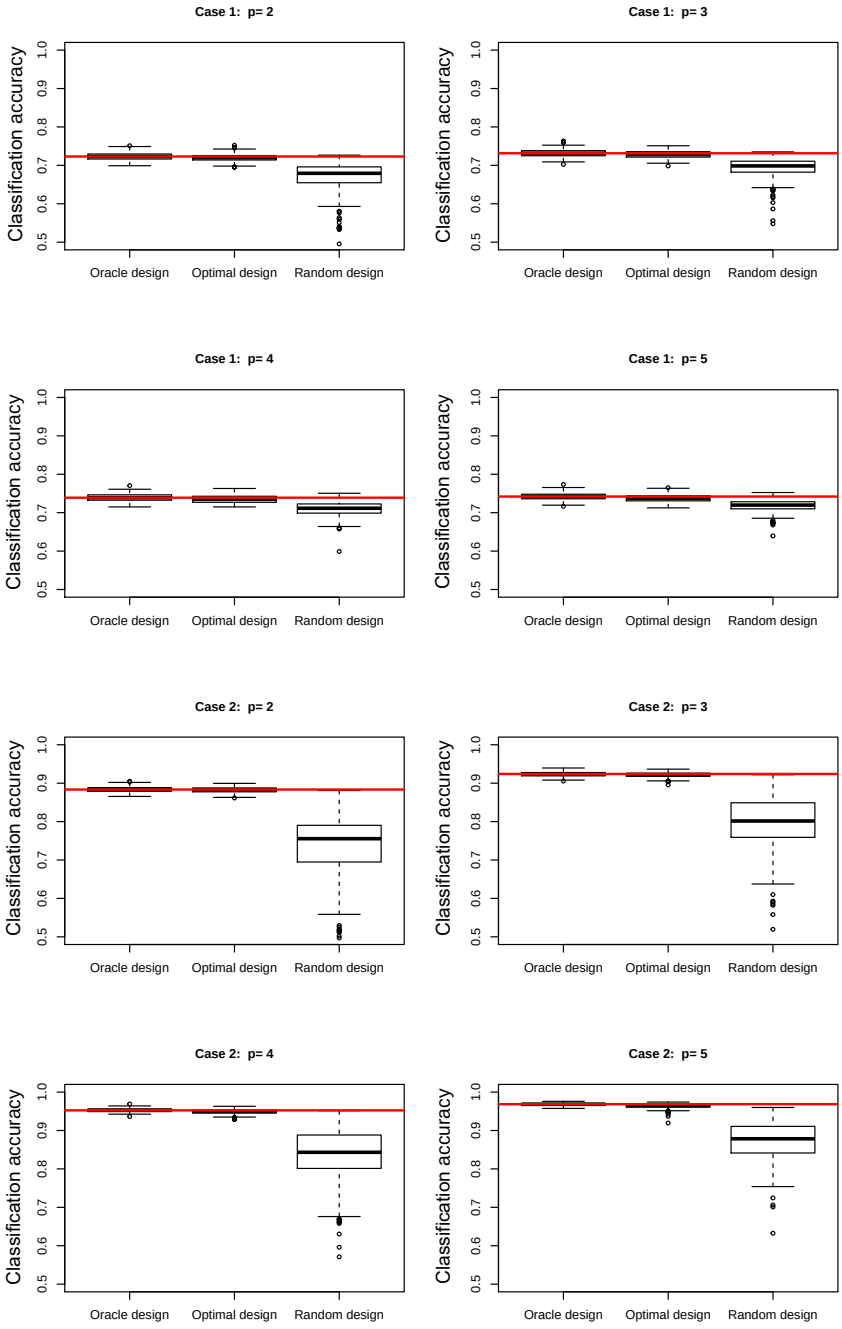


Figure 2.2: Boxplots of percentages of correct classifications using the oracle design, (estimated) optimal design and the random design. The red solid lines represent the medians of classification accuracies using the oracle design.

2.6.1 Data

The Berkeley growth data in the R package *fda* (Ramsay et al., 2011) contains the heights of 39 boys and 54 at 31 non-equidistant ages from age 1 to age 18 (Tuddenham and Snyder, 1954). We aim to predict the gender using the growth curves.

The Tecator data consists of 215 near infrared absorbance spectra of pure meat recorded on a Tecator Infratec Food and Feed Analyzer, at 100 equally spaced sampling points of the wavelength ranging from 850 to 1050 nm. The dataset is available at `http://lib.stat.cmu.edu/datasets/tecator`. There are 77 subjects with high fat content (more than 20%) and 138 subjects with low fat content (less than 20%). Functional data analysis based on this dataset were done in Ferraty and Vieu (2003), Delaigle et al. (2012) and Galeano et al. (2015). Following the above works, we use the second derivatives of the original curves for classification of subjects with either high or low fat content.

The phoneme data, available at `https://statweb.stanford.edu/~tibs/ElemStatLearn/data.html`, contains 1717 log-periodograms constructed from 32 ms long recordings of speakers pronouncing phonemes. There are two phonemes pronounced by the male speakers, 695 subjects are from the class of phoneme "aa" as in the first vowel of "dark", 1022 subjects are form the class of phoneme "ao" as in the first vowel of "water". The curves are recorded at 256 equally spaced frequencies.

The Mediterranean fruit fly data, denoted by "medfly" and available at `http://anson.ucdavis.edu/~mueller/data/data.html`, consists of 789 female Mediterranean fruit flies with daily egg-laying records at 25 consecutive days from birth (Carey et al., 1998). The profile of egg-laying during the first 25 days can be used to predict the number of eggs to be laid during the remaining lifetime. Since the longevity of Mediterranean fruit flies can be gauged by reproductive potential (Müller et al., 2001), we consider 509 flies with more than 150 remaining eggs to be laid in lifetime as the population of high life expectancy, and the remaining 280 flies are in the population of low life expectancy. An optimal design in terms of regression based on this data was conducted in Ji and Müller (2016).

The bone mineral density (BMD) data, available at `https://statweb.stanford.edu/~tibs/`

`ElemStatLearn/data.html`, contains relative spinal bone mineral density measurements on 261 North American adolescents (Bachrach et al., 1999). There are 116 males and 145 females. As shown in Figure 2.1, this is a longitudinal study concerning gender differences. The sampling points range from age 9 to age 26 and the average number of observations per subject is less than 2. The method in Delaigle et al. (2012) is not designed for such data.

To summarize, the growth data, the Tecator data, the phoneme data and the medfly data are dense functional data, and the BMD data are sparse functional data.

We can only use dense functional data to assess the out-of-sample classification accuracy of the proposed method. The reason is that if the observations are not available at the selected time points, it is not possible to evaluate the performance of the selected sampling points. For each dense dataset, we randomly select 2/3 of the subjects as the training set and 1/3 of the subjects as the testing set. Additionally, we sparsify the training data by randomly keeping m_i observations from subject i, where m_i is drawn independently from a uniform distribution with mean m. Two uniform distributions are used: the uniform distribution in the interval $[3, 7]$ with $m = 5$ and the uniform distribution in the interval $[7, 13]$ with $m = 10$. Each experiment scenario is replicated 200 times.

For the BMD data, we apply the proposed method to locate the optimal sampling time points.

2.6.2 Comparison of Dense and Sparse Training Data

We use the four dense datasets to illustrate the proposed optimal design for selecting 2 sampling points when the training data are either dense or sparse. For the sparse training data, either $m = 5$ or 10 observations on average per subject are sampled. For the random design, the classifier is built on the estimated model from the dense training data. Figure 2.3 displays the box plots of classification accuracy for various types of training data using either the optimal or random design. We observe that the classification accuracy of the proposed optimal design is relatively consistent across the different types of training data. Moreover, the classification accuracy of the proposed method built on the sparse training data with $m = 5$ is only slightly inferior to that using full data. In addition, the proposed optimal design substantially outperforms the random design. Finally,

the classification accuracy of the proposed design increases as the number of sampling time points increase from 1 to 2.

2.6.3 Optimal Designs

For the four dense datasets, we apply the proposed optimal design to identify two optimal sampling time points; see Figure 2.4. Figure 2.4 provides results that are sensible and consistent with previous research. More specifically, for the growth data, the two optimal points are selected at ages 13.5 and 15.5, which are the beginning and end of adolescence. For the Tecator data and the phoneme data, Delaigle et al. (2012) estimated similar selected sampling points for classification using cross-validation based methods. For the medfly data, interestingly, Ji and Müller (2016) identified the consecutive days 25 and 26 as two optimal points for the purpose of functional linear regression, while we found the days 24 and 25 to be optimal. The interpretation is that the reproductive potential of female Mediterranean fruit flies is associated with the decay rate of egg production (Müller et al., 2001).

Table 2.3 summarizes the optimal design for the BMD data. We pinpoint the interpretable design points for different p (the number of selected sampling points) and report the associated estimates of PCCs. The optimal sampling points tend to cluster around age 12 and age 16, again, which are the beginning and the end of adolescence. See Figure 2.1 for an illustration of the selected two most predictive sampling points. Using criterion (2.9) with $\delta = 0.01$ (last column of Figure 2.1), we determine the number of sampling points to be 3.

Table 2.3: Optimal designs for the BMD dataset.

p	Design points (age)	$\widehat{\text{PCC}}$	Criterion (2.9) with $\delta = 0.01$
1	16.1	0.700	0.310
2	11.6, 16.1	0.754	0.266
3	11.6, 15.5, 16.1	0.776	0.254
4	11.6, 15.5, 16.1 25.6	0.777	0.263
5	9.4 11.6, 15.5, 16.1 20.0	0.778	0.272

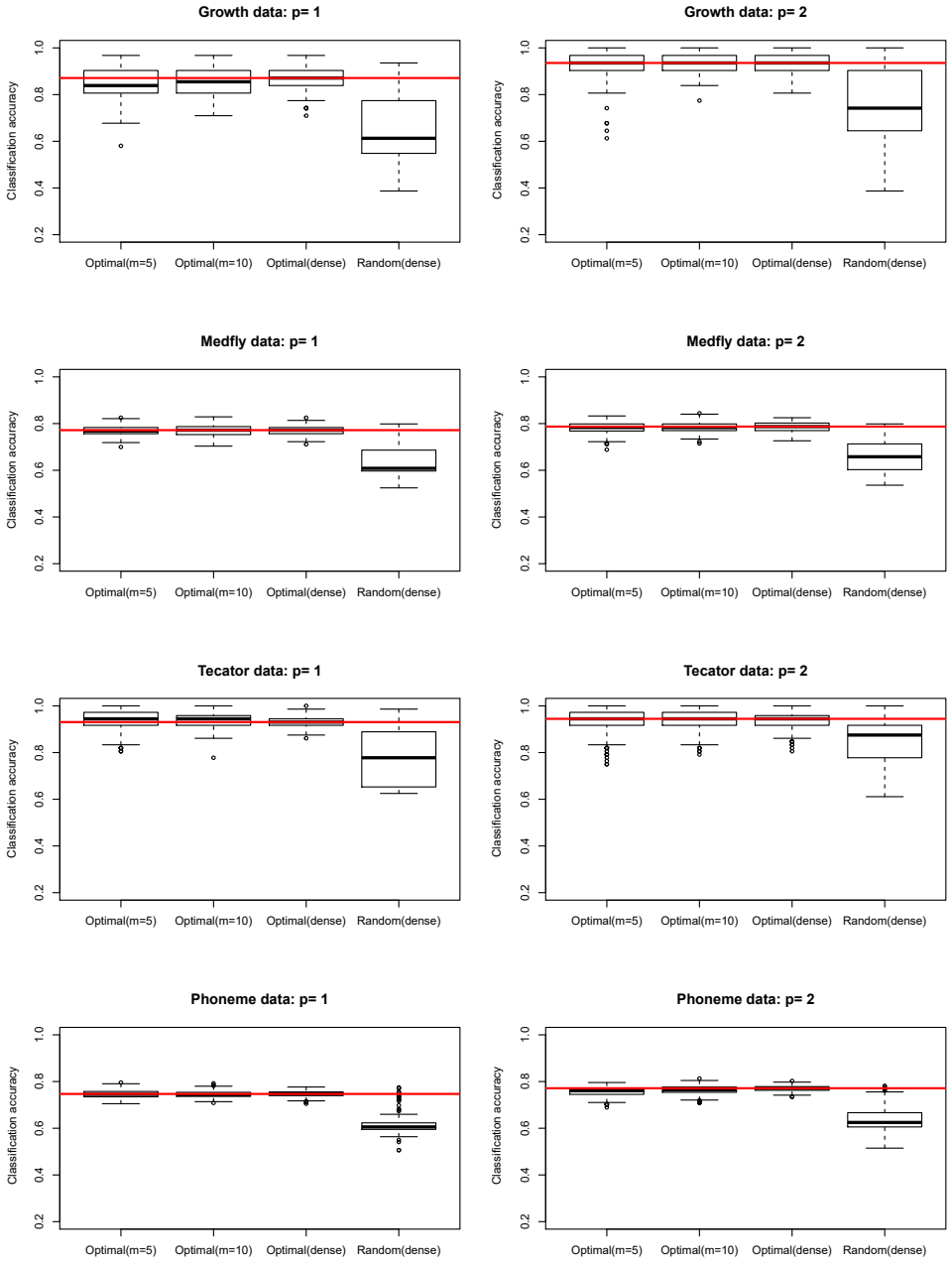


Figure 2.3: Comparison of dense and sparse training data in terms of classification accuracy. The red solid lines are the medians of classification accuracies using the selected optimal sampling points when the training data are dense.

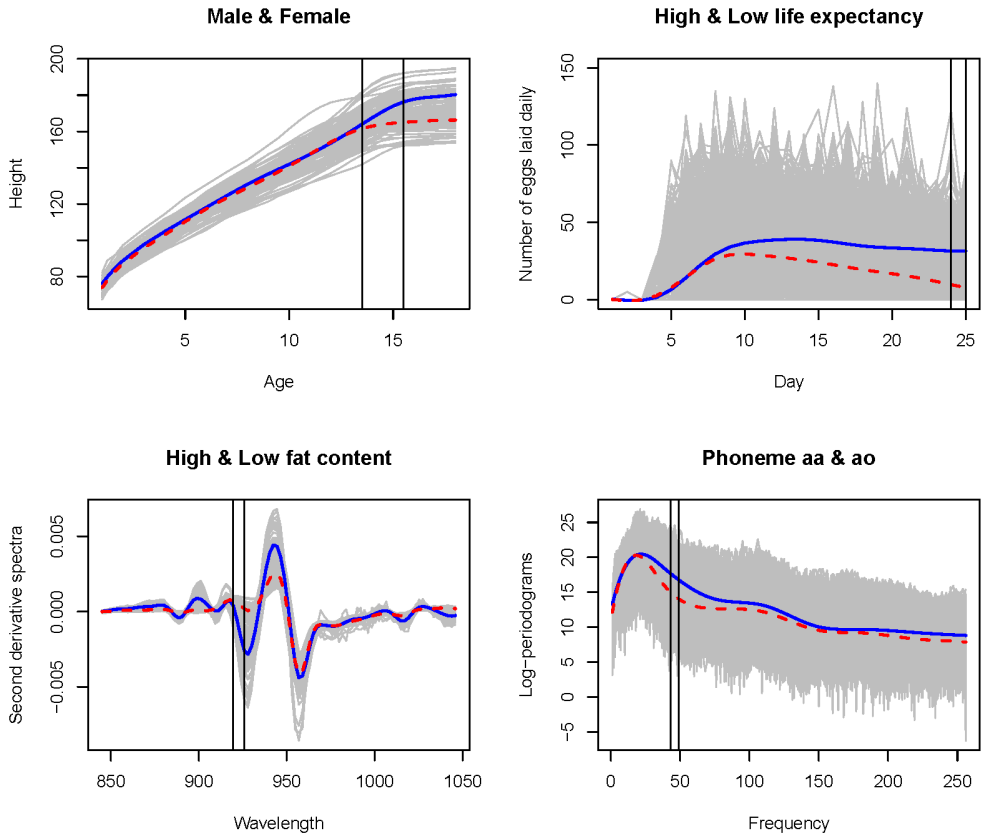


Figure 2.4: Selected two optimal sampling points. The blue solid and red dashed lines represent the estimated group mean functions and the black vertical bars indicate two most predictive points for each dataset.

2.7 Discussion

In this chapter, we proposed optimal designs for selecting the sampling points for classification of functional data. We established the theoretical properties of the proposed design and also provided a practical implementation.

The proposed method requires the pilot functional data which can be either densely or sparsely observed, an advantage of the proposed method compared to existing methods. Given the increasing use of functional data methods for longitudinal studies, an optimal design that is applicable to longitudinal data seems desirable.

The proposed design is limited to the linear classifier and its extension to other classifiers such as the quadratic classifier requires further research and is beyond the scope of the work.

Chapter 3

Conditional Analysis for Mixed Covariates[1]

3.1 Introduction

Many modern applications routinely collect data on study participants comprising scalar responses and covariates of various types, vector, function, image, and the main question of interest is to examine how the covariates affect the response. For example in our data application the aim is to study how the min-by-min temperature or humidity during a day in the farrowing rooms, that is rooms where piglets are born and nursed by the sow until they are weaned, affect the feed intake of lactating sows in their first 21 lactation days, while accounting for other sow-specific information. The response is feed intake amount at a day, and covariates are the min-by-min temperature for that day, average humidity, age of the respect saw.

A popular approach in these cases is to use a semi-parametric framework and assume that the mixed covariates solely affect the mean response, which is the average feed intake; see Cardot et al. (1999); Ramsay and Silverman (2002); James (2002); Ferraty and Vieu (2006, 2009); Goldsmith et al. (2011); McLean et al. (2014) and others. While it is still important to study the average feed

[1]This chapter is based on a joint work with So Young Park, Zhen Han, Santa-Maria Mendoza, Eric van Heugten and Ana-Maria Staicu.

intake, animal scientists are often more concerned with the left tail of the feed intake distribution because low feed intake of lactating sows could lead to many serious issues, including decrease in milk production and negative impact on the sows reproductive system; see, for reference, Quiniou and Noblet (1999); Renaudeau and Noblet (2001); St-Pierre et al. (2003) among others. In this chapter we focus on regression models that study the effects of covariates on the entire distribution of response. Our contribution is the development of a modeling framework that accommodates a comprehensive study of various types functional (temperature and humidity recorded at five-minute intervals) and vector (number of previous pregnancies) on a scalar response (feed intake amount).

Quantile regression models the effect of scalar/vector covariates beyond the mean response and has attracted great interest (Koenker and Bassett Jr, 1978; Koenker, 2005). This approach offers a more comprehensive picture of the effects of the covariates on the response distribution. For pre-specified quantile levels, quantile regression models the conditional quantiles of the response as a function of the observed covariates; this approach has been extended more recently to ensure non-crossing of quantile functions (Bondell et al., 2010). Quantile regression has been also extended to handle functional covariates. Cardot et al. (2005) discussed quantile regression models by employing a smoothing spline modeling-based approach. Kato (2012) considered the same problem and used a functional principal component (fPC) based approach. Both papers mainly discussed the case of having a single functional covariate and it is not clear how to extend them to the case where there are multiple functional covariates or mixed covariates (vector and functional). Ferraty et al. (2005) and Chen and Müller (2012) considered a different perspective and studied the effect of a functional predictor on the quantiles of the response by first positing a model for the conditional distribution of the response and then inverting it.

More recently, Tang and Cheng (2014), Lu et al. (2014), and Yu et al. (2015) studied quantile regression when the covariates are of mixed types and introduced the partial functional linear quantile regression model framework. The first two publications used fPC basis while the last one considered partial quantile regression (PQR) basis. These approaches all are suitable when the interest is in studying the effect of covariate at a particular quantile level, and do not handle the

study of covariate effects at simultaneous quantile levels due to the well-known crossing-issue.

In this chapter we fill this gap and propose a modeling framework that allows to study the effect of mixed type covariates on the distribution of a scalar response. Our approach is inspired from Chen and Müller (2012) (CM, henceforth); nevertheless it has important differences as we explain in detail in Section 3.2. Specifically let $Q_{Y|X}(\tau|X)$ denote the τth conditional quantile of Y given a functional covariate $X(\cdot)$, and let $F_{Y|X}(y)$ denote the conditional distribution of Y given X. Using the relationship between $Q_{Y|X}(\tau|X)$ and $F_{Y|X}(y)$ that $Q_{Y|X}(\tau|X) = \inf\{y : F_{Y|X}(y) \geq \tau\}$ for $0 < \tau < 1$, CM proposed to estimate the quantile function $Q_{Y|X}(\tau|X)$ in two steps: 1) estimate the conditional distribution of Y given $X(\cdot)$, $F(y) = \mathbb{E}[\mathbb{1}(Y < y)|X(\cdot)]$ by positing a mean regression model for an auxiliary variable $Z(y) := \mathbb{1}(Y \leq y)$ and the functional covariate $X(\cdot)$ by using a common scalar-on-function framework; and 2) estimate $Q_{Y|X}(\tau|X)$ by inverting the estimated conditional distribution function. Their estimation approach is restrictive to one functional covariate and a direct extension to accommodate mixed covariates is computationally expensive. We propose a unifying modeling framework and estimation technique that easily accommodate mixed types of covariates in a computationally efficient manner by equivalently representing the problem using a function-on-function regression framework. Our contribution is two fold: 1) our method is spline-based, and as a result can easily accommodate smooth effects of scalar variables as well as of functional covariates; and 2) our estimation approach is based on a single step function-on-function (or function-on-scalar) penalized regression; this enables efficient implementation by exploiting off-the-shelf software and leads to competitive computations.

The remainder of this chapter is structured as follows. Section 3.2 discusses the details of the proposed method and Section 3.3 performs a thorough simulation study evaluating the performance of the proposed method and competitors. We apply the proposed method to analyze the sow data in Section 3.4. We conclude the chapter with a discussion in Section 3.5.

3.2 Modeling Framework

Let X_1 be a scalar covariate and $X_2(\cdot)$ be a functional covariate defined on a closed domain $\mathcal{T}$. We propose the following model for the conditional distribution of Y given X_1 and $X_2(\cdot)$:

$$\mathbb{E}[\mathbb{1}(Y < y)] = g^{-1}\left\{\beta_0(y) + X_1\beta_1(y) + \int X_2(t)\beta_2(t,y)dt\right\}, \tag{3.1}$$

where $g(\cdot)$ is a known, monotone link function, $\beta_1(y)$ is unknown and smooth function and $\beta_2(t,y)$ is unknown and smooth bi-variate function over y and t. The parameters $\beta_1(y)$ and $\beta_2(\cdot,y)$ quantify the effect of the covariates X_1 and $X_2(\cdot)$ respectively onto the distribution of the response. Chen and Müller (2012) (CM) considered a similar model; their approach is restrictive to a single functional covariate. Their estimation is based on functional principal component bases and a direct extension to accommodate mixed covariate is computationally expensive.

The proposed modeling and estimation method is discussed first for the case of a scalar covariate in Section 3.2.1. Section 3.2.2 considers the extension to the case of a functional covariate and then to the case of mixed covariates. Section 3.2.3 further extends the method to handle sparse and noisy functional covariates. We briefly discuss the monotonization of the estimated conditional distribution in Section 3.2.4.

3.2.1 Conditional Distribution of the Response Given Scalar Covariate

To explain ideas, consider the case of a single scalar covariate X first. Specifically let the data be $\{(X_i, Y_i) : i = 1, \ldots, n\}$, where X_i and Y_i are independent realizations of real-valued scalar random variables X and Y, respectively. For instance in the sow data application X is the average daily humidity, and Y is daily feed intake. Define $Z_i(y) = \mathbb{1}(Y_i < y)$ for $y \in \mathbb{R}$, where $\mathbb{1}(\cdot)$ is an indicator function; for each y, we view $Z_i(y)$ as a binary-valued random variable that is independent and identically distributed as $Z(y) = \mathbb{1}(Y < y)$. It follows the conditional distribution function $F_{Y|X}(y) = \mathbb{E}[Z(y)|X]$. Here we propose to model the conditional distribution, $F_{Y|X}(y)$, using a generalized function-on-scalar regression model Scheipl et al. (2015) between the 'artificial' binary

functional response $Z_i(y)$ and the scalar covariate X_i. Specifically, for each $y \in \mathbb{R}$, consider

$$\mathbb{E}[Z_i(y)|X_i] = g^{-1}\big\{\beta_0(y) + X_i\beta_1(y)\big\}, \tag{3.2}$$

where $g(\cdot)$ is a known, monotonic link function, and $\beta_0(\cdot)$ and $\beta_1(\cdot)$ are unknown, smooth coefficient functions. Here we use the logit function defined as $g(x) = \log\{x/(1+x)\}$. If the slope parameter $\beta_1(\cdot)$ is null then the covariate X_i has no effect on the distribution of the response Y_i, which is equivalent to X_i having no effect on any quantile level of Y_i.

We model $\beta_0(y)$ and $\beta_1(y)$ by using pre-specified, truncated univariate basis, $\{B_{0,d}(\cdot) : d = 1, \ldots, \kappa_0\}$ and $\{B_{1,d}(\cdot) : d = 1, \ldots, \kappa_1\}$: $\beta_0(y) = \sum_{d=1}^{\kappa_0} B_{0,d}(y)\theta_{0,d}$ and $\beta_1(y) = \sum_{d=1}^{\kappa_1} B_{1,d}(y)\theta_{1,d}$, where $\theta_{0,d}$'s and $\theta_{1,d}$'s are unknown basis coefficients. Then model (3.2) can be represented as the following generalized additive model

$$\mathbb{E}[Z_i(y)|X_i] = g^{-1}\Big\{\sum_{d=1}^{\kappa_0} B_{0,d}(y)\theta_{0,d} + \sum_{d=1}^{\kappa_1} B_{x,d}(y)\theta_{1,d}\Big\}, \tag{3.3}$$

where for convenience we use the notation $B_{x_i,d}(y) = X_i B_{1,d}(y)$. The general idea is to set the basis dimensions κ_0 and κ_1 to be sufficiently large to capture the complexity of the coefficient functions and control the smoothness of the estimator through roughness penalties $P_0(\boldsymbol{\theta}_0)$ and $P_1(\boldsymbol{\theta}_1)$, where $\boldsymbol{\theta}_l$, is a vector of all basis coefficients $\{\theta_{l,d} : l = 1, \ldots, D_l\}$ for $l = 1, 2$. This approach of using roughness penalties has been widely used; see, for example, Eilers and Marx (1996); Ruppert (2002); Wood (2003, 2006) among many others.

In the following, we detail the estimation algorithm. Let $\{y_j : j = 1, \ldots, J\}$ be a set of equi-spaced points in the range of the response variable, Y_i's. For each i and j, we define $Z_{ij} = Z_i(y_j) = \mathbb{1}(Y_i < y_j)$; it follows that conditional on X_i, the Z_{ij} are independently distributed as Bernoulli distribution with mean (μ_{ij}), where μ_{ij} is such that $g(\mu_{ij}) = \mathbf{B}_{0,j}^T\boldsymbol{\theta}_0 + \mathbf{B}_{x,j}^T\boldsymbol{\theta}_1$. Here $\mathbf{B}_{0,j}^T$ is a $\kappa_0 \times 1$ vector of $\{B_{0,d}(y_j) : d = 1, \ldots, \kappa_0\}$ and $\mathbf{B}_{x_i,j}^T$ is a $\kappa_1 \times 1$ vector of $\{X_i B_{1,d}(y_j) : d = 1, \ldots, \kappa_1\}$. The

basis coefficients, $\boldsymbol{\theta}_0$ and $\boldsymbol{\theta}_1$, are estimated by maximizing the penalized log likelihood criterion,

$$2\log\mathcal{L}(\boldsymbol{\theta}_0, \boldsymbol{\theta}_1 | \{Z_i(y_j), X_i : \forall i, j\}) - \lambda_0 P_0(\boldsymbol{\theta}_0) - \lambda_1 P_1(\boldsymbol{\theta}_1), \tag{3.4}$$

where $\mathcal{L}$ is the likelihood function of data $\{Z_{ij} : j = 1, \cdots, J\}_i$, $P_0(\boldsymbol{\theta}_0)$ and $P_1(\boldsymbol{\theta}_1)$ are penalties, and λ_0 and λ_1 are smoothing parameters. We use quadratic penalties which penalize the size of the curvature of the estimated coefficient functions. Let $P_0(\boldsymbol{\theta_0}) = \boldsymbol{\theta}_0^T \mathbf{D}_0 \boldsymbol{\theta}_0$ and $P_1(\boldsymbol{\theta}_1) = \boldsymbol{\theta}_1^T \mathbf{D}_1 \boldsymbol{\theta}_1$, where $\mathbf{D}_0$ and $\mathbf{D}_1$ are $\kappa_0 \times \kappa_0$ and $\kappa_1 \times \kappa_1$ dimensional matrices based on the basis used (see Wood (2006) for example; the (s, s') element of $\mathbf{D}_0$ is $\int B''_{0,s}(y) B''_{0,s'}(y) dy$ and $\mathbf{D}_1$ is defined similarly). The smoothing parameters λ_0 and λ_1 control the trade-off between the goodness of fit and smoothness of the fit. The smoothing parameters are estimated using restricted maximum likelihood (REML).

The criterion (3.4) can be viewed as the penalized quasi-likelihood (PQL) of the corresponding generalized linear mixed model

$$Z_{ij} | \boldsymbol{\theta}_0, \boldsymbol{\theta}_1 \sim \text{Bernoulli}(\mu_{ij}); \qquad \boldsymbol{\theta}_0 \sim N\Big(\mathbf{0}, \lambda_0^{-1} \mathbf{D}_0^-\Big); \qquad \boldsymbol{\theta}_1 \sim N\Big(\mathbf{0}, \lambda_1^{-1} \mathbf{D}_1^-\Big), \tag{3.5}$$

where $\mathbf{D}_0^-$ is the generalized inverse matrix of $\mathbf{D}_0$ and $\mathbf{D}_1^-$ is defined similarly. Wood (2006) discusses an alternative way to deal with the rank-deficient matrices, $\mathbf{D}_0$ and $\mathbf{D}_1$, in the context of restricted maximum likelihood (REML) estimation. See also Ivanescu et al. (2015) who uses the mixed model representation of a similar regression model to (3.5), but with a Gaussian functional response.

Let $\{\widehat{\theta}_{l,d} : d = 1, \ldots, \kappa_l\}$ for $l = 0, 1$ be the estimated basis coefficients. It follows that the estimated distribution function is $\widehat{F}_{Y|X_i}(y) = g^{-1}\{\sum_{d=1}^{\kappa_0} B_{0,d}(y)\widehat{\theta}_{0,d} + \sum_{d=1}^{\kappa_1} B_{x_i,d}(y)\widehat{\theta}_{1,d}\}$. The τth conditional quantile are estimated as by inverting the estimated distribution $\widehat{F}_{Y|X_i}(\cdot)$, $\widehat{Q}_{Y|X_i}(\tau|X_i) = \min_j\{y_j : \widehat{F}_{Y|X_i}(y_j) \geq \tau\}$. This approach relates the τth level quantile of the response in a non-linear manner to the covariate. The estimated distribution function, $\widehat{F}_{Y|X_i}(y)$, is not a monotonic function yet. In practice we suggest to first apply a monotonization method as described in the Supplementary Material Section 3.2.4, and then estimate the conditional quantiles by inverting the resulting estimated distribution.

3.2.2 Extension to Mixed Covariates

The modeling approach discussed in Section 3.2.1 is quite powerful as it can easily accommodate covariates of various types, as we show next. Let's first assume that there is a single functional covariate $X_i(\cdot)$; for convenience, assume that $X_i(\cdot)$ is observed at fine grid of points and without noise. The regression model (3.2) changes to $\mathbb{E}[Z(y)|X] = g^{-1}\big\{\beta_0(y) + \int X(t)\beta_1(t,y)dt\big\}$, where $\beta_1(\cdot,\cdot)$ is an unknown bi-variate coefficient function. In terms of modeling and estimation, the main difference from Section 3.2.1 is that the coefficient function, $\beta_1(t,y)$, is now bivariate and it requires appropriate pre-specified basis function and corresponding penalty term.

We represent $\beta_1(t,y)$ using the tensor product of two univariate bases functions, $\{B^t_{1,d_t}(t) : d_t = 1,\ldots,\kappa_{1,t}\}$ and $\{B^y_{1,d_y}(y) : d_y = 1,\ldots,\kappa_{1,y}\}$; $\beta_1(t,y) = \sum_{d_t=1}^{\kappa_{1,t}}\sum_{d_y=1}^{\kappa_{1,y}} B^t_{1,d_t}(t)B^y_{1,d_y}(y)\theta_{1,d_t,d_y}$. Subsequently the previous penalty matrix $\mathbf{D}_1$ should be also appropriately modified to control the smoothness of $\beta_1(t,y)$ in directions of both t and y. There are several choices to define the penalty matrix in nonparametric regression (see Marx and Eilers (2005); Xiao et al. (2013)). For bivariate smoothing we use $\mathbf{D}_1 = \{\mathbf{P}_t \otimes \mathbf{I}_{\kappa_{1,y}} + \mathbf{I}_{\kappa_{1,t}} \otimes \mathbf{P}_y\}$, where (s,s') element of $\mathbf{P}_t$ is $\int\{\partial^2 B^t_{1,s}(t)/\partial t^2\}\{\partial^2 B^t_{1,s'}(t)/\partial t^2\}dt$ and (s,s') element of $\mathbf{P}_y$ is $\int\{\partial^2 B^y_{1,s}(y)/\partial y^2\}\{\partial^2 B^y_{1,s'}(y)/\partial y^2\}dy$ introduced by Wood (2006). In practice the integration term $\int X_i(t)\beta_1(t,y)dt$ is approximated by Riemann integration $\int X_i(t)\beta_1(t,y)dt = \sum_{l=1}^{L} X_i(t_l)\beta_1(t_l,y)(t_{l+1}-t_l)$, but other numerical approximation scheme can be also used. The estimation of parameters proceeds similarly to Section 3.2.1.

It is important to emphasize that even in the case of a single functional covariate, our methodology differs from Chen and Müller (2012) (CM) in few directions: 1) our proposed method is based on modeling the unknown smooth coefficient functions using pre-specified basis function expansion and using penalties to control their roughness. In contrast, CM uses data-driven basis, chooses the number of basis functions through the percentage of explained variance (PVE) of the functional predictors. This key difference allows us to accommodate covariates of different types. 2) Our estimation approach is based on a single step penalized criterion while CM uses pointwise estimation based on the residual sum of square criterion and thus requires fitting multiple generalized regressions. This is an important advantage in terms of computational efficiency.

Next assume the data include covariates: (i) $\mathbf{X}_{1i}$, a vector of nuisance covariates; (ii) X_{2i}, a scalar covariate of interest; and (iii) $X_{3i}(\cdot)$, a functional covariate of interest. We posit the model

$$F_{Y|X}(y) = g^{-1}\left\{\beta_0(y) + \mathbf{X}_{1i}^T\boldsymbol{\beta_1} + X_{2i}\beta_2(y) + \int X_{3i}(t)\beta_3(t,y)dt\right\}, \tag{3.6}$$

Here $\mathbf{X}_{1i}$ is assumed to have a linear effect $\boldsymbol{\beta_1}$ that is invariant to y; and X_{2i} and $X_{3i}(t)$ to have a linear effect that is varying with y. It is easy to see that a null effect, say $\beta_3(\cdot,\cdot) \equiv 0$ is equivalent to the fact that the respective covariate $X_3(\cdot)$ has no effect on any quantile level of the response. Fitting models in (3.2) or (3.6) can be done by extending the ideas of Ivanescu et al. (2015) for Gaussian functional response; the extension of the model to the non-Gaussian functional response has recently been studied and implemented by Scheipl et al. (2015) using the `pffr` function in `refund` package (Huang et al., 2015).

One advantage of the proposed framework is that it can be easily extended to allow for more flexible effects. In particular, the smooth effect $X_{2i}\beta_2(y)$ can be replaced by $h_1(X_{2i})$, and the linear effect $\int X_{i3}(t)\beta_3(t,y)dt$ by $\int h_2\{X_2(t),t,y\}dt$, where $h_1(\cdot)$ and $h_2(\cdot,\cdot,\cdot)$ are unknown univariate and trivariate smooth functions, respectively; these changes require little additional computational burden. We consider the nonlinear model in the simulation study for the case of having a scalar covariate only and the corresponding results are presented in Section C.1.1 of the Supplementary Materials. The results show excellent prediction performance as the competitive nonlinear quantile regression method, namely Constrained B-Spline Smoothing (COBS) (Ng and Maechler, 2007).

3.2.3 Extension to Sparse and Noisy Functional Covariates

In practice the functional covariates are often observed at irregular times across the units and also are possibly corrupted with measurement errors. In such case, one needs to first smooth and denoise the trajectories before fitting. When the sampling design of the functional covariate is dense, the common approach is to take each trajectory and smooth it using spline or local polynomial smoothing, as proposed in Ramsay and Silverman (2005) and Zhang et al. (2007). When the design

is sparse, the smoothing can be done by pooling all the subjects and following the PACE method proposed in Yao et al. (2005), which is implemented in `MATLAB` and the `fpca.sc` function of `R` package `refund` (Huang et al., 2015). In our simulation study we used `fpca.sc`, irrespective of a sampling design (dense or sparse). Alternatively one can use `fpca.face` (Xiao et al., 2016) in `refund` for regular dense design and `face.sparse` (Xiao et al., 2017a) in the `R` package `face` for irregular sparse design.

3.2.4 Monotonization

While a conditional quantile function is nondecreasing, the resulting estimated quantiles may not be. We consider monotonization as proposed by Chernozhukov et al. (2009). Chernozhukov et al. (2009) showed that in this way the monotonized estimator gives the same or better fit than the original estimator. Two approaches are widely used; one is to monotonize the estimated conditional distribution function $\widehat{F}_{Y|X}(y)$, and the other is to monotonize the estimated conditional quantile function $\widehat{Q}_{Y|X}(\tau)$. We choose the former as we already have $\widehat{F}_{Y|X}(y)$ evaluated at dense grid points y_j's and there is no need to obtain the estimated conditional quantile at fine grid points of the quantile level $\tau \in [0, 1]$. We use an isotonic regression model (Barlow et al., 1972) for monotonization, which fits a nonparametric model with an order restriction. The isotonic regression model is fitted through $[\{y_j, \widehat{F}_{Y|X}(y_j)\} : j = 1, \ldots, J]$ using the `R` function `isoreg`. This idea was also employed in Kato (2012).

3.3 Simulations

In this section we evaluate numerically the performance of the proposed method. We present results for the case when we have both functional and scalar covariates; additional results when there is only single functional or single scalar covariate are discussed in the Supplementary Materials, Section C.1. We adapt the simulation settings of Chen and Müller (2012) for the cases that involve a functional covariate.

Suppose the observed data for the ith subject is $[Y_i, X_{1i}, \{(W_{i1}, t_{i1}), \cdots, (W_{im_i}, t_{im_i})\}]$, $t_{ij} \in$

$[0, 10]$, where $X_{1i} \overset{i.i.d}{\sim} Unif(-16, 16)$, $W_{ij} = X_{2i}(t_{ij}) + \epsilon_{ij} = \mu(t_{ij}) + \sum_{k=1}^{4} \xi_{ik}\phi_k(t_{ij}) + \epsilon_{ij}$, $1 \leq i \leq n, 1 \leq j \leq m_i$. Set the mean function $\mu(t) = t + \sin(t)$, and the eigenfunctions $\phi_k(t) = \cos\{(k+1)\pi t/10\}/\sqrt{5}$ for odd values of k, $\phi_k(t) = \sin\{k\pi t/10\}/\sqrt{5}$ for even values of k. Here assume that scores $\xi_{ik} \overset{iid}{\sim} N(0, \lambda_k)$, where $(\lambda_1, \lambda_2, \lambda_3, \lambda_4) = \{16, 9, 7.56, 5.06\}$, and $\epsilon_{ij} \overset{iid}{\sim} N(0, \sigma_\epsilon^2)$. We assume two cases:

(i) normal distribution $Y_i|X_{1i}, X_{2i}(\cdot) \sim N(2\int X_{2i}(t)\beta(t)dt + 2X_{1i}, 5^2)$; this yields the quantile regression model $Q_{Y|X_1,X_{2(\cdot)}}(\tau) = 2\int X_{2i}(t)\beta(t)dt + 2X_{1i} + 5\Phi^{-1}(\tau)$, where $\Phi(\cdot)$ is the distribution function of the standard normal and $\beta(t) = \sum_{k=1}^{4} \beta_k\phi_k(t)$;

(ii) mixture of normal distributions
$Y_i|X_{1i}, X_{2i}(\cdot) \sim 0.5N(\int X_{2i}(t)\beta(t)dt + X_{1i}, 1^2) + 0.5\ N(3\int X_{2i}(t)\beta(t)dt + 3X_{1i}, 4^2)$;
this yields the quantile regression model $Q_{Y|X_1,X_{2(\cdot)}}(\tau) = 2\int X_{2i}(t)\beta(t)dt + 2X_{1i} + \sqrt{(\int X_{2i}(t)\beta(t)dt + X_{1i})^2 + 8.5}\Phi^{-1}(\tau)$, where $\beta(t) = \sum_{k=1}^{4} \beta_k\phi_k(t)$.

Three noise levels are considered: low ($\sigma_\epsilon = 0.50$), moderate ($\sigma_\epsilon = 4.33$), and high ($\sigma_\epsilon = 6.13$). The three levels are such that the signal to noise ratio (SNR), which are calculated as $SNR = \sqrt{\sum_{k=1}^{4} \lambda_k}/\sigma_\epsilon$, are equal to $SNR = 150$, 2, and 1, respectively. Results are presented for sample sizes, $n = 100$ (small) and $n = 1000$ (large).

The performance is evaluated on a test set of size 100. Two sampling designs are considered: (i) *dense design*, where the sampling points $\{t_{ij} : j = 1, \ldots, m_i\}$ are a set of $m_i = 30$ equi-spaced time points in $[0, 10]$; and (ii) *sparse design*, where $\{t_{ij} : j = 1, \ldots, m_i\}$ are $m_i = 15$ randomly selected points from a set of 30 equi-spaced grids in $[0, 10]$. The quantile functions with our approach are estimated as described in Section 3.2 by first creating an artificial binary response $Z_i(\cdot)$ and then fitting a penalized function-on-function regression model and using the logit link function; we use the `pffr` function (Ivanescu et al., 2015; Scheipl et al., 2015) in the `refund` package (Huang et al., 2015) in `R` for binomial responses, denoted by Joint QR.

We compare our method with three alternative approaches: (1) a variant of our proposed approach using pointwise fitting, denoted by Pointwise QR, and hence fitting multiple regression

models with binomial link function as implemented by the penalized functional regression `pfr` of the `refund` package for generalized scalar responses, developed by Goldsmith et al. (2011); (2) a modified version of the CM method, denoted by Mod CM, that we developed to account for additional scalar covariates, and which estimates pointwise using multiple generalized linear models; (3) a linear quantile regression approach using the quantile loss function and the partial quantile regression bases for functional covariates, proposed by Yu et al. (2015) and denoted by PQR. Notice that (1) and (2) account for a varying effect of the covariates on the response distribution, but do not ensure that this effect is smooth.

The `R` function `pfr` can incorporate both scalar/vector and functional predictors by adopting a mixed effects model framework. The functional covariates are pre-smoothed by fPC analysis; pre-smoothing the functional covariates before fitting the regression model has been also considered by Goldsmith et al. (2011) and Ivanescu et al. (2015). Throughout the simulation study we used REML to select the smoothing parameters and a percentage of variance explained (PVE) to determin trancation for the fPC analysis. Other basis settings are set to the defaults given by developers. The performance is evaluated in terms of mean absolute error (MAE) for quantile levels $\tau = 0.05, 0.1, 0.25$, and 0.5.

Numerical results are provided in Tables 3.1, 3.2 and 3.3. Table 3.1 gives results for the two settings (normal and mixture) when the functional covariate is observed on dense design and the sample size is $n = 100$; Table 3.2 shows the corresponding results for $n = 1000$. We obtained similar findings for the sparse scenario and hence are not reported for brevity.

Consider first the case when conditional distribution of the response is normal. When sample size is large ($n = 1000$) the proposed method (Joint QR) yields the best MAE for the SNR and the quantile levels considered. Even with low-moderate sample size ($n = 100$) the Joint QR remains performing the best for extreme quantiles ($\tau = 0.05, 0.10$ and 0.25) and relatively large noises ($\sigma_\epsilon = 4.33$ and 6.13). When sample size is small-moderate the PQR method also performs very well for the small noise ($\sigma_\epsilon = 0.50$) and for middle quantiles ($\tau = 0.05$ or $\tau = 0.10$). The Pointwise QR and Mod CM methods perform similarly, where the Pointwise QR tends to do better for low-

moderate sample sizes ($n = 100$) while the Mod CM tends to do better for larger one ($n = 1000$). All of the four methods are affected by the level of SNR; the higher it is, the better MAE is.

When the conditional distribution of the response follows the mixture of normals, there is no uniformly best method across quantiles levels or SNR levels whe sample size is large ($n = 1000$). It seems that all four methods have similar performance with some being the best for some situations while others for other situations. Overall the Joint QR method tends to perform better for extreme quantiles ($\tau = 0.05$ or $\tau = 0.10$) while the other three methods tend to predict better the middle quantiles ($\tau = 0.25$ or $\tau = 0.50$). Other findings are relatively similar to the ones for the normal case.

Table 3.3 compares the three methods that involve estimating the conditional distribution - Joint QR, Pointwise QR and Mod CM - in terms of the computational time required for fitting; the times correspond to using a computer with a 2.3 GHz CPU and 8 GB of RAM. Not surprisingly by fitting the model a single time, Joint QR is the fastest, in some cases being order of magnitude faster than the rest. Pointwise QR can be up to twice as fast as Mod CM.

For completeness, we also compare our proposed method to the appropriate competitive methods for the cases (1) when there is a single scalar covariate and (2) when there is a single functional covariate. The Supplementary Materials, Section C.1.1 discusses the former case and compares Joint QR and Pointwise QR with the linear quantile regression and the nonlinear quantile regression (as implemented by the `cobs` function in the `R` package `COBS` (Ng and Maechler, 2007)) in an extensive simulation experiment that involves both linear quantile settings and nonlinear quantile settings. Overall the results show that the proposed methods have similar behavior as LQR; see Table S1. Furthermore we consider the proposed methods with nonlinear modeling of the conditional distribution as discussed in Section 3.2, which we denote with Joint QR (NL) for joint fitting and Pointwise QR (NL) for pointwise fitting. Nonlinear versions of the proposed methods have an excellent MAE performance, which is comparable to or better than that of the COBS method.

Finally, Section C.1.2 in the Supplementary Materials discusses the simulation study for the case of having a single functional covariate and compares the proposed methods with CM in terms

of MAE as well as computational time; see results displayed in Tables S2 and S3. The results show that the proposed Joint QR is superior to CM both in terms of the prediction accuracy and computation efficiency. In our simulation study we also consider the joint fitting of the model by treating the binary response as normal and use `pffr` (Ivanescu et al., 2015) with Gaussian link, denoted by Joint QR (G).

Table 3.1: Average MAE and standard error (in parentheses) of the predicted τ-level quantile for sample size $n = 100$ based on 500 replications.

Distribution	σ_ϵ	Method	$\tau = 0.05$	$\tau = 0.1$	$\tau = 0.25$	$\tau = 0.5$
Normal	0.50	Joint QR	3.30(0.03)	3.15(0.03)	2.90(0.02)	2.71(0.02)
		Pointwise QR	4.41 (0.03)	4.06 (0.03)	3.67 (0.02)	3.59 (0.02)
		Mod CM	4.32 (0.03)	4.04 (0.03)	3.67 (0.02)	3.53 (0.02)
		PQR	**2.97** (0.04)	**2.72** (0.04)	**2.50** (0.04)	**2.45** (0.04)
Normal	4.33	Joint QR	**8.13** (0.04)	**7.57** (0.03)	**7.13** (0.03)	7.02 (0.03)
		Pointwise QR	9.37 (0.05)	8.46 (0.04)	7.63 (0.03)	7.36 (0.03)
		Mod CM	8.88 (0.04)	8.69 (0.04)	8.52 (0.04)	8.60 (0.04)
		PQR	8.76 (0.06)	8.04 (0.05)	**7.13** (0.04)	**6.76** (0.03)
Normal	6.13	Joint QR	**9.90** (0.05)	**9.05** (0.04)	**8.32** (0.03)	8.13 (0.03)
		Pointwise QR	11.07 (0.06)	9.90 (0.05)	8.78 (0.03)	8.41 (0.03)
		Mod CM	10.27 (0.04)	10.00 (0.04)	9.79 (0.04)	9.84 (0.04)
		PQR	10.67 (0.07)	9.71 (0.06)	8.40 (0.04)	**7.86** (0.03)
Mixture	0.50	Joint QR	**6.59** (0.06)	**5.77** (0.06)	**6.17** (0.06)	**4.45** (0.05)
		Pointwise QR	7.53 (0.07)	6.13 (0.06)	6.07 (0.06)	4.64 (0.06)
		Mod CM	6.66 (0.06)	5.94 (0.06)	6.37 (0.06)	4.95 (0.06)
		PQR	8.08 (0.07)	7.06 (0.05)	6.32 (0.04)	5.91 (0.14)
Mixture	4.33	Joint QR	**10.40** (0.06)	**9.06** (0.05)	**8.62** (0.05)	6.89 (0.05)
		Pointwise QR	11.70 (0.08)	9.65 (0.06)	8.61 (0.05)	6.88 (0.05)
		Mod CM	11.40 (0.06)	10.89 (0.06)	10.68 (0.06)	9.34 (0.07)
		PQR	12.00 (0.08)	10.28 (0.06)	8.68 (0.04)	**6.09** (0.06)
Mixture	6.13	Joint QR	**11.79** (0.07)	**10.15** (0.05)	9.38 (0.05)	7.53 (0.05)
		Pointwise QR	12.94 (0.09)	10.68 (0.06)	**9.37** (0.05)	7.48 (0.05)
		Mod CM	12.61 (0.07)	11.95 (0.06)	11.60 (0.06)	10.23 (0.07)
		PQR	13.40 (0.10)	11.47 (0.07)	9.48 (0.04)	**6.59** (0.05)

Table 3.2: Average MAE and standard error (in parentheses) of the predicted τ-level quantile for sample size $n = 1000$ based on 500 replications.

Distribution	σ_ϵ	Method	$\tau = 0.05$	$\tau = 0.1$	$\tau = 0.25$	$\tau = 0.5$
Normal	0.50	Joint QR	**1.27** (0.01)	**1.34** (0.01)	**1.33** (0.01)	**1.22** (0.01)
		Pointwise QR	1.61 (0.01)	1.59 (0.01)	1.54 (0.01)	1.43 (0.01)
		Mod CM	1.43 (0.01)	1.40 (0.01)	1.39 (0.01)	1.36 (0.01)
		PQR	1.74 (0.02)	1.71 (0.02)	1.67 (0.02)	1.67 (0.02)
Normal	4.33	Joint QR	**7.79** (0.03)	**6.91** (0.02)	**6.22** (0.02)	**6.11** (0.02)
		Pointwise QR	8.03 (0.03)	7.06 (0.03)	6.31 (0.02)	6.19 (0.02)
		Mod CM	7.94 (0.03)	7.11 (0.03)	6.43 (0.02)	6.32 (0.02)
		PQR	8.36 (0.04)	7.55 (0.04)	6.60 (0.03)	6.22 (0.02)
Normal	6.13	Joint QR	**9.84** (0.04)	**8.59** (0.03)	**7.52** (0.03)	**7.30** (0.03)
		Pointwise QR	10.09 (0.04)	8.76 (0.03)	7.61 (0.03)	7.35 (0.03)
		Mod CM	9.95 (0.04)	8.79 (0.03)	7.77 (0.03)	7.55 (0.03)
		PQR	10.39 (0.05)	9.26 (0.04)	7.89 (0.03)	7.33 (0.02)
Mixture	0.50	Joint QR	4.33 (0.03)	2.89 (0.02)	4.62 (0.03)	3.54 (0.03)
		Pointwise QR	**4.04** (0.03)	2.42 (0.02)	4.25 (0.03)	**3.45** (0.03)
		Mod CM	4.12 (0.03)	**2.28** (0.02)	**4.08** (0.03)	3.65 (0.03)
		PQR	7.74 (0.04)	6.40 (0.03)	5.31 (0.02)	3.75 (0.10)
Mixture	4.33	Joint QR	**9.62** (0.04)	7.46 (0.03)	7.10 (0.03)	5.74 (0.03)
		Pointwise QR	9.64 (0.04)	**7.37** (0.03)	**6.86** (0.03)	5.67 (0.03)
		Mod CM	9.64 (0.04)	7.60 (0.03)	7.18 (0.03)	6.12 (0.03)
		PQR	11.74 (0.05)	9.65 (0.04)	7.82 (0.03)	**4.64** (0.02)
Mixture	6.13	Joint QR	**11.47** (0.04)	8.94 (0.03)	8.00 (0.03)	6.33 (0.03)
		Pointwise QR	11.55 (0.05)	**8.91** (0.03)	**7.82** (0.03)	6.22 (0.03)
		Mod CM	11.48 (0.04)	9.17 (0.04)	8.22 (0.04)	6.81 (0.03)
		PQR	13.33 (0.06)	10.87 (0.04)	8.65 (0.03)	**5.37** (0.02)

Table 3.3: Comparison of the average computing time (in seconds) for the three approaches that involve estimating the conditional distribution.

Distribution	Method	$n = 100$	$n = 1000$
Normal	Joint QR	17	200
	Pointwise QR	148	271
	Mod CM	278	511
Mixture	Joint QR	18	282
	Pointwise QR	151	296
	Mod CM	327	532

3.4 Sow Data Application

Our motivating application is an experimental study carried out at a commercial farm in Oklahoma from July 21, 2013 to August 19, 2013 (Rosero et al., 2016). The study comprises of 480 lactating sows of different parities (i.e. number of previous pregnancies, which serves as a surrogate for age and body weight) that were observed during their first 21 lactation days; their feed intake was recorded daily as the difference between the feed offer and the feed refusal. In addition the study contains information on the temperature and humidity of the farrowing rooms, each recorded at five minute intervals. The final dataset we used for the analysis consists of 475 sows after five sows with unreliable measurements were removed by the experimenters.

The experiment was conducted to gain better insights into the way that the ambient temperature and humidity of the farrowing room affects the feed intake of lactating sows. Previous studies seem to suggest a reduction in the sow's feed intake due to heat stress: above 29°C sows decrease feed intake by 0.5 kg per additional degree in temperature Quiniou and Noblet (1999). Studying the effect of heat stress on lactating sows is a very important scientific question because of a couple of reasons. First, the reduction of feed intake of the lactating sows is associated with a decrease in both their bodyweight (BW) and milk production, as well as the weight gain of their litter (Johnston et al., 1999; Renaudeau and Noblet, 2001; Renaudeau et al., 2001). Sows with poor feed intake during lactation continue the subsequent reproductive period with negative energy balance (Black et al., 1993), which leads to prevent the onset of a new reproductive cycle. Second, heat stress reduces farrowing rate (number of sows that deliver a new litter) and number of piglets born (Bloemhof et al., 2013); the reduction in reproductivity due to seasonality is estimated to cost 300 million dollars per year for the swine industry (St-Pierre et al., 2003). Economic losses are estimated to increase (Nelson et al., 2009) because high temperatures are likely to occur more frequently due to global warming (Melillo et al., 2014).

Our primary goal is to understand the thermal needs of the lactating sows for proper feeding behavior during the lactation time. We are interested in how the interplay between the temperature and humidity of the farrowing room affects the feed intake demeanor of lactating sows of different

parities. We focus on three specific times during the lactation period: beginning (lactation day 4), middle (day 11) and end (day 18) and consider two types of responses that are meant to assess the feed intake behavior using the current and the previous lactation days. The first one quantifies the absolute change in the feed intake over two consecutive days and the second one quantifies the relative change and takes into account the usual sow's feed intake. We define them formally after introducing some notation.

Let FI_{ij} be the jth measurement of the feed intake observed for the ith sow and denote by LD_{ij} the lactation day when FI_{ij} is measured; here $j = 1, \ldots, n_i$. Most sows are observed for every day within the first 21 lactation days and thus have $n_i = 21$. First define the absolute change in the feed intake between two consecutive days as $\Delta^{(1)}_{i(j+1)} = FI_{i(j+1)} - FI_{ij}$ for j that satisfies $LD_{i(j+1)} - LD_{ij} = 1$. For instance $\Delta^{(1)}_{i(j+1)} = 0$ means there was no change in feed intake of sow i between the current day and the previous day, while $\Delta^{(1)}_{i(j+1)} < 0$ means that the feed intake consumed by the ith sow in the current day is smaller than the feed intake consumed in the previous day. However, the same amount of change in the feed intake may reflect some stress level for a sow who typically eats a lot and a more serious stress level for a sow that usually has a lower appetite. For this, we define the relative change in the feed intake by $\Delta^{(2)}_{i(j+1)} = (FI_{i(j+1)} - FI_{ij})/\{(LD_{i(j+1)} - LD_{ij}) \cdot TA_i\}$, where TA_i is the trimmed average of feed intake of ith sow calculated as the average feed intake after removing the lowest 20% and highest 20% of the feed intake measurements $\{FI_{i1}, \ldots, FI_{in_i}\}$ taken for the corresponding sow. Here TA_i is surrogate for the usual amount of feed intake of the ith sow. Trimmed average is used instead of the common average, to remove outliers of very low feed intakes in first few lactating days. For example, consider the situation of two sows: sow i that *typically* consumes 10lb food per day and sow i' that consumes 5lb food per day. A reduction of 5lb in the feed intake over two consecutive days corresponds to $\Delta^{(2)}_{i(j+1)} = -50\%$ for the ith sow and $\Delta^{(2)}_{i'(j+1)} = -100\%$ for the i'th sow. Clearly both sows are stressed (negative value) but the second sow is stressed more, as its absolute relative change is larger; in view of this we refer to the second response as the *stress index*. Due to the definition of the two types of responses, the data size varies, so for the first response, $\Delta^{(1)}_{i(j+1)}$, we have sample

sizes of 233, 350, and 278 for lactation days 4 ($j = 3$), 11 ($j = 10$), and 18 ($j = 17$), respectively, whereas for $\Delta^{(2)}_{i(j+1)}$ the sample sizes are 362, 373, and 336 respectively.

In this analysis we center the attention on the effect of the ambient temperature and humidity on the *1st quartile* of the proxy stress measures and gain more understanding of the food consumption of sows that are most susceptible to heat stress. While the association between the feed intake of lactating sows and the ambient conditions of the farrowing room has been an active research area for some time, accounting for the temperature daily profile has not been considered yet hitherto. Figure 3.1 displays the temperature and humidity daily profiles recorded at a frequency of 5-minute window intervals for three different days. Preliminary investigation reveals that temperature is negatively correlated with humidity at each time; this phenomenon is caused because the farm uses cool cell panels and fans to control the ambient temperature. Furthermore, it appears that there is strong pointwise correlation between temperature and humidity. In view of these observations we consider the daily average of humidity in our analysis. Exploratory analysis of the feed intake behavior of the sows suggest similarities for the sows with parity greater than older sows (ones who are at their third pregnancy or higher); thus we use a parity indicator instead of the actual parity of the sow. The parity indicator P_i is defined as one, if the ith sow has parity one and zero otherwise.

For the analysis we smooth daily temperature measurements of each sow using univariate smoother with 15 cubic regression bases and quadratic penalty; REML is used to estimate smoothing parameter. The smoothed temperature curve for sow i's jth repeated measure is denoted by $T_{ij}(t)$, $t \in [0, 24)$, and the corresponding daily average humidity is denoted by AH_{ij}. Both temperature and average humidity are centered before being used in the analysis.

For convenience we denote the response with Δ_{ij} by removing the superscript. We first estimate the conditional distribution of Δ_{ij} given temperature $T_{ij}(t)$, average humidity AH_{ij}, parity P_i, and interaction $AH_{ij} \cdot T_{ij}(t)$. Specifically for each of $j = 3, 10$ and 17 we create a set of 100 equi-spaced grid of points between the fifth smallest and fifth largest values of Δ_{ij}'s and denote the grids with $\mathcal{D} = \{d_\ell : \ell = 1, \ldots, 100\}$. Then we create artificial binary response, $\{\mathbb{1}(\Delta_{ij} < d_\ell) : \ell = 1, \ldots, 100\}$,

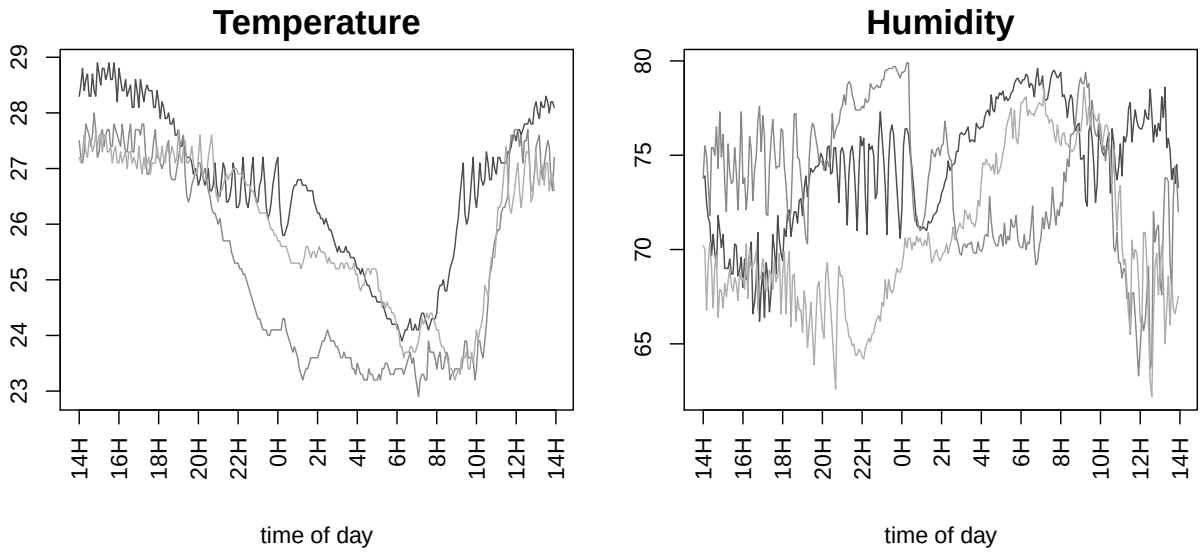


Figure 3.1: Temperature (°C) and humidity (%) observed profiles (dashed) for three randomly selected days and the corresponding smoothed ones (solid); the x-axis begins at 14H (2PM).

and fit the following model for $F_{ij}(d_\ell) = \mathbb{E}\left[\mathbb{1}\left(\Delta_{ij} < d_\ell\right)\middle|T_{ij}(t), AH_{ij}, P_i\right]$:

$$\mathbb{E}\left[\mathbb{1}\left(\Delta_{ij} < d_\ell\right)\middle|T_{ij}(t), AH_{ij}, P_i\right] = g^{-1}\Big\{\beta_0(d_\ell) + \beta_1(d_\ell)P_i + \beta_2(d_\ell)AH_{ij} + \int \beta_3(d_\ell, t)T_{ij}(t)dt + AH_{ij}\int \beta_4(d_\ell, t)T_{ij}(t)dt\Big\},$$

where $\beta_0(\cdot)$ is a smooth intercept, $\beta_1(\cdot)$ quantifies the smooth effect of young sows, $\beta_2(\cdot)$ describes the effect of the humidity, and $\beta_3(\cdot, t)$ and $\beta_4(\cdot, t)$ quantify the effect of the temperature at time t as well as the interaction between the temperature at time t and average humidity. We model $\beta_0(\cdot)$ using 20 univariate basis functions, $\beta_1(\cdot)$ and $\beta_2(\cdot)$ using five univariate basis functions, $\beta_3(\cdot, \cdot)$ and $\beta_4(\cdot, \cdot)$ using tensor product of two univariate bases functions (total of 25 functions). Throughout the analysis, cubic B-spline bases are used and REML is used for estimating smoothing parameters. The estimated conditional distribution, denoted by $\widehat{F}_{ij}(d)$, is monotonized by fitting isotonic regression to $\{(d_\ell, \widehat{F}_{ij}(d_\ell)) : \ell = 10, \ldots, 90\}$; ten smallest and ten largest d_ℓ and the corresponding values of $\widehat{F}_j(d_\ell)$ are removed to avoid boundary effects. By abuse of notation, $\widehat{F}_{ij}(d)$ denotes the resulting monotonized estimated distribution. Finally, we obtain estimated first quartiles, i.e. quantiles at

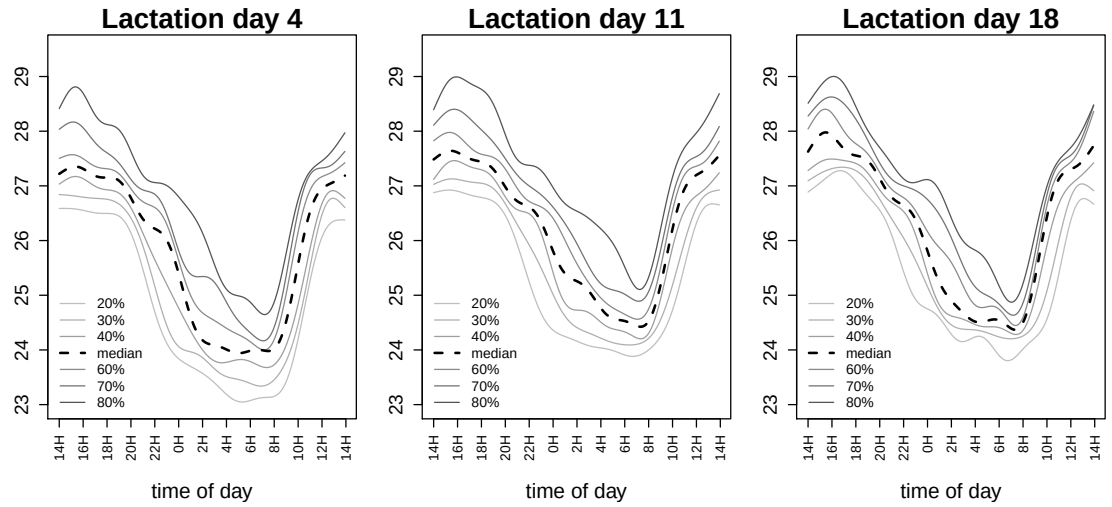


Figure 3.2: Temperature curves with which prediction of quantiles is made. Dashed black line is pointwise average of temperature curves and solid lines are pointwise quartiles; all curves are smoothed.

$\tau = 0.25$ level, by inverting $\widehat{F}_{ij}(d)$, namely $\widehat{Q}\left(\tau = 0.25 \mid T_{ij}(t), AH_{ij}, P_i\right) = \inf\{d : \widehat{F}_{ij}(d) > 0.25\}$.

To understand the relationship between lactating sows' feed intake and thermal condition of the farrowing room, we systematically compare and study predicted quantiles of two responses at combinations of different values of temperature, humidity, and parity. For each of three lactation days ($j = 3, 10, 17$) we consider three values of average humidity (first quartile, median, and third quartile) and two levels of parity (0 for older sows and 1 for younger sows). Based on the experimenters' interest, for the functional covariate $T_{ij}(\cdot)$ we consider seven smooth temperature curves given in Figure 3.2. Each of these curves are obtained by first calculating pointwise quantiles of temperature at five-minute intervals for a specific level and then smoothing it; we considered quantiles levels $\eta = 0.2, 0.3, \ldots,$ and 0.8. In short, for each of three lactation days we obtain the first quartile of two responses for 42 different combinations (3 humidity values $\times$ 2 parity levels $\times$ 7 temperature curves) using the proposed method. To avoid extrapolation we ascertain that (i) there are reasonably many observed measurements at each of the combinations and (ii) bottom 25% of the responses are not dominantly from one of the parity group; see distribution of each response by the parity in Figure 3.3.

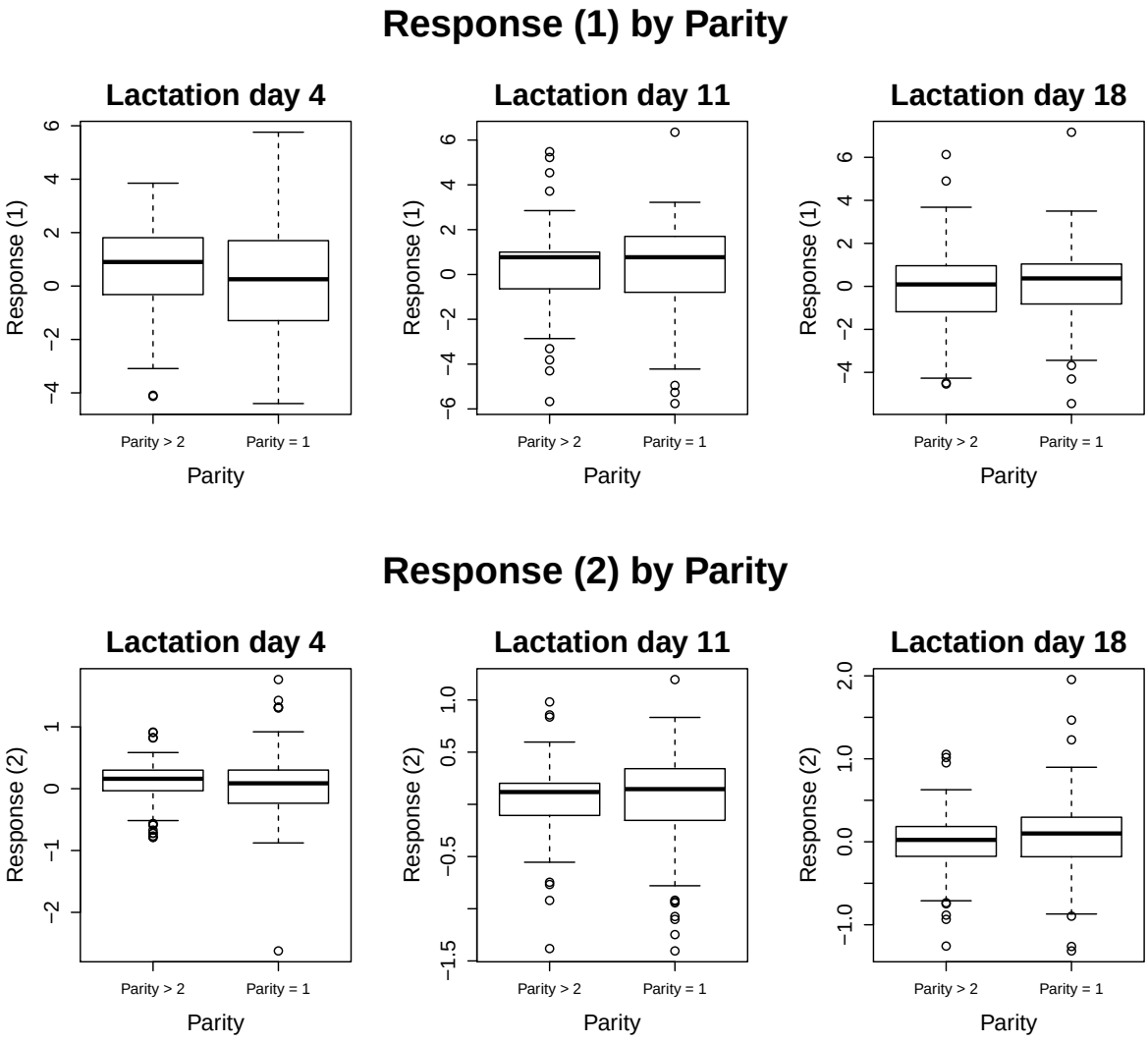


Figure 3.3: Distribution of responses by Parity

The resulting predicted quantiles are shown in Figure 3.4. Here we focus our discussion on predicted quantile of $\Delta^{(2)}_{i(j+1)}$ at quantile level $\tau = 0.25$ for lactation day 4 ($j = 3$) - the first plot of the second row in Figure 3.4. The results suggest that the feed intake of older sows (parity $P_i = 0$; grey lines) are less affected by high temperatures than that of younger sows (black lines); this finding is in agreement with Bloemhof et al. (2013), who also found that younger sows are more sensitive to ambient changes than sows with higher parity. We also observe that the effects of humidity and temperature on feed intake change are strongly interwined. For illustation, let's focus on lacation day 4 ($j = 3$) again for younger sows (black lines). For medium humidity (dashed) their feed intake stays pretty constant as temperature increases, while for low and high humidity levels (solid and dotted lines, respectively) it changes with an opposite direction. Specifically when temperature increases the predicted first quartile of $\Delta^{(2)}_{i(j+1)}$ increases for low humidity (solid line) whereas it decreases for high humidity (dotted line). Our results imply that high humidity (dotted line) is related to a negative impact of high temperature on feed intake while low humidity (solid line) alleviates it; and this finding is consistent with a previous study (Bergsma and Hermesch, 2012). The analysis result suggests to keep low humidity levels in order to maintain healthy feed intake behavior, when ambient temperature is above 60th percentile; high humidity levels are desirable for cooler ambient temperature.

The results corresponding to other lactation days can also be interpreted similarly. While the effects of covariates on feed intake are less apparent toward the end of lactation period, we still observe similar pattern across all three lactation days. For lactation day 11 ($j = 10$) we observe that when temperature is above 40th percentile the predicted first quartile starts increasing with low humidity while it continues decreasing with high humidity. Similarly for lactation day 18 ($j = 17$) when temperature is above 60th percentile the predicted first quartile increases with low humidity while it decreases with high humidity. The effect of temperature on feed intake seems less obvious for lactation days 11 and 18 than for day 4; while the effect may be due to lactation day, it may also be a result of other factors, such as more fluctuation and variability in temperature curves on day 4 than other two days (see Figure 3.2). Overall we conclude that high humidity and temperature

affect sows' feed intake behavior negatively and young sows (parity one) are more sensitive to heat stress than older sows (higher parity), especially in the beginning of lactation period.

3.5 Discussion

The proposed modeling framework opens up a couple of future research directions. A first research avenue is to develop significance tests of null covariate effect. Testing for the null effect of a covariate on the conditional distribution of the response is equivalent to testing that the corresponding regression coefficient function is equal to zero in the associated function-on-function mean regression model. Such significance tests have been studied when the functional response is continuous (Shen and Faraway, 2004; Zhang et al., 2007); however their study for binary-valued functional responses is an open problem in functional data literature. Another research avenue is to do variable selection in the setting where there are many scalar covariates and functional covariates. Many current applications collect data with increasing number of mixed covariates and selecting the ones that have an effect on the conditional distribution of the response is very important. This problem is an active research area in functional mean regression where the response is normal (Gertheiss et al., 2013; Chen et al., 2015). The proposed modeling framework has the potential to facilitate studying such problem.

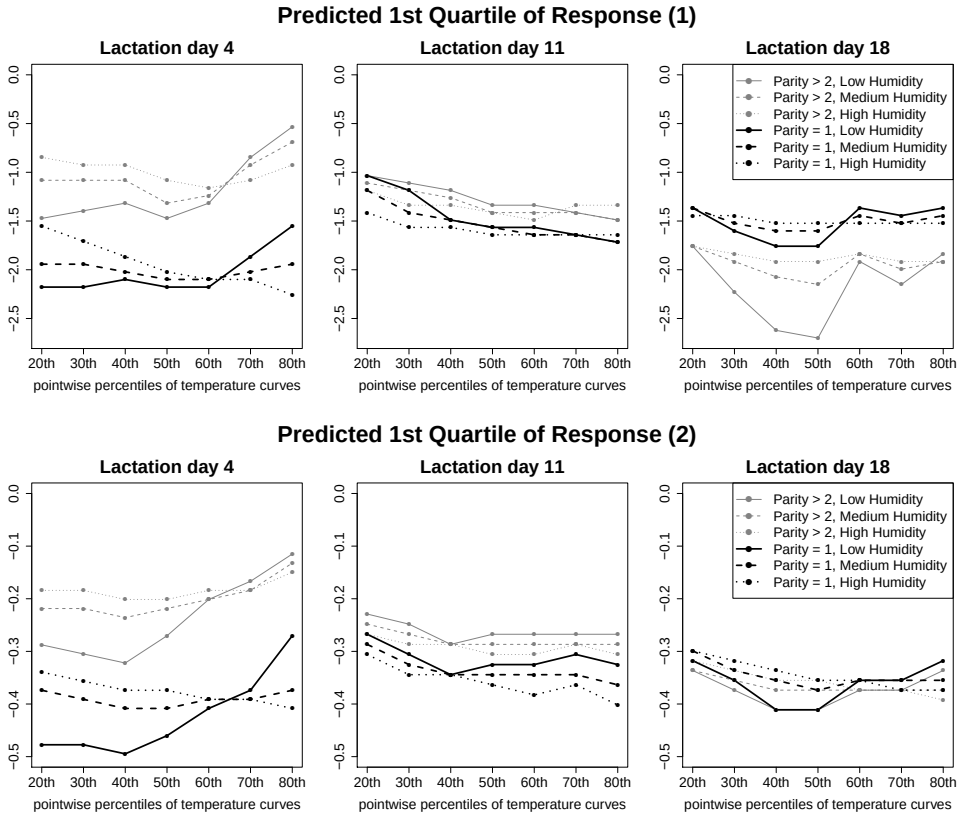


Figure 3.4: Displayed are the predicted quantiles of $\Delta^{(1)}_{i(j+1)}$ and $\Delta^{(2)}_{i(j+1)}$ for different parities, average humidity, and temperature levels. In each of all six panels, black thick lines correspond to the young sows ($P_i = 1$) and grey thin lines correspond to the old sows ($P_i = 0$). Line types indicate different average humidity levels; solid, dashed, and dotted correspond to low, medium, and high average humidity levels (given by the first quartile, median, and the third quartiles of AH_{ij}), respectively. The seven grids in x-axis of each panel correspond to the 7 temperature curves given in the respective panel of Figure 3.2.

Chapter 4

svt: Singular Value Thresholding in MATLAB[1]

4.1 Introduction

Many modern statistical learning problems concern estimating a matrix-valued parameter. Examples include matrix completion, regression with matrix covariates, and multivariate response regression. Matrix completion (Candès and Recht, 2009; Mazumder et al., 2010) aims to recover a large matrix of which only a small fraction of entries are observed. The problem has sparked intensive research in recent years and is enjoying a broad range of applications such as personalized recommendation system (ACM SIGKDD and Netflix, 2007) and imputation of massive genomic data (Chi et al., 2013). In matrix regression (Zhou and Li, 2014), the predictors are two dimensional arrays such as images or measurements on a regular grid. Thus it requires a regression coefficient array of same size to completely capture the effects of matrix predictors. Another example is regression with multiple responses (Yuan et al., 2007; Zhang et al., 2015), which involves a matrix of regression coefficients instead of a regression coefficient vector.

In these matrix estimation problems, the nuclear norm regularization is often employed to

[1]This chapter is based on a joint work with Hua Zhou, which has been conditionally accepted by *Journal of Statistical Software*.

achieve a low rank solution and shrinkage simultaneously. This leads to a general optimization problem

$$\text{minimize} \quad \ell(\mathbf{B}) + \lambda \|\mathbf{B}\|_*, \tag{4.1}$$

where ℓ is a relevant loss function, $\mathbf{B} \in \mathbb{R}^{m \times n}$ is a matrix parameter, $\|\mathbf{B}\|_* = \sum_i \sigma_i(\mathbf{B}) = \|\sigma(\mathbf{B})\|_1$ (sum of singular values of $\mathbf{B}$) is the nuclear norm of $\mathbf{B}$, and λ is a positive tuning parameter that balances the trade-off between model fit and model parsimony. The nuclear norm plays the same role in low-rank matrix approximation that the ℓ_1 norm plays in sparse regression. Generic optimization methods such as accelerated proximal gradient algorithm, majorization-minorization (MM) algorithm, and ADMM have been invoked to solve optimization problem (4.1). See, e.g., Mazumder et al. (2010); Boyd et al. (2011); Parikh and Boyd (2013); Chi et al. (2013); Lange et al. (2014) for matrix completion algorithms and Zhou and Li (2014); Zhang et al. (2015) for the accelerated proximal gradient method for solving nuclear norm penalized regression. All these algorithms involve repeated singular value thresholding, which is the proximal mapping associated with the nuclear norm regularization term

$$\mathbf{A} \mapsto \arg\min \frac{1}{2}\|\mathbf{X} - \mathbf{A}\|_{\mathrm{F}}^2 + \lambda \|\mathbf{X}\|_*. \tag{4.2}$$

Let the singular value decomposition of $\mathbf{A}$ be $\mathbf{U}\mathrm{diag}(\sigma_i)\mathbf{V}^\top = \sum_i \sigma_i \mathbf{u}_i \mathbf{v}_i^\top$. The solution of (4.2) is given by $\sum_i (\sigma_i - \lambda)_+ \mathbf{u}_i \mathbf{v}_i^\top$ (Cai et al., 2010). Some common features characterize the singular value thesholding operator in applications. First the involved matrices are often large. For matrix completion problem, m, n can be at order of $10^3 \sim 10^6$. Second only the singular values that exceed λ and their associated singular vectors are needed. Third the involved matrix is often structured. In this article, we say a matrix is *structured* if matrix-vector multiplication is fast. For example, in matrix completion problem, $\mathbf{A}$ is of the form "sparse + low rank". That is $\mathbf{A} = \mathbf{M} + \mathbf{L}\mathbf{R}^\top$, where $\mathbf{M}$ is sparse and $\mathbf{L} \in \mathbb{R}^{m \times r}$ and $\mathbf{R} \in \mathbb{R}^{n \times r}$ are low rank $r \ll \min\{m, n\}$. Although $\mathbf{A}$ is not sparse itself, matrix-vector multiplications $\mathbf{A}\mathbf{v}$ and $\mathbf{w}^\top \mathbf{A}$ costs $O(m + n)$ flops instead of $O(mn)$. Storing

the sparse matrix $\mathbf{M}$ and $\mathbf{L}$ and $\mathbf{R}$ also takes much less memory than the full matrix $\mathbf{A}$. All these characteristics favor the iterative algorithms for singular value decomposition such as the Lanczos bidiagonalization method (Golub and Van Loan, 1996).

Most algorithms for aforementioned applications are developed in `MATLAB`, which however lacks a convenient singular value thresholding functionality. The most direct approach for SVT is applying full SVD through `svd` and then soft-threshold the singular values. This approach is in practice used in many matrix learning problems according to the distributed code, e.g., Kalofolias et al. (2014); Chi et al. (2013); Parikh and Boyd (2013); Yang et al. (2013); Zhou et al. (2014); Zhou and Li (2014); Zhang et al. (2015); Otazo et al. (2015); Goldstein et al. (2015), to name a few. However, the built-in function `svd` is for full SVD of a dense matrix, and hence is very time-consuming and computationally expensive for large-scale problems. Another built-in function `svds` wraps the `eigs` function to calculate top singular triplets using iterative algorithms. However the current implementation of `svds` is efficient only for sparse matrix input, while matrix estimation algorithm involves singular value thresholding of dense but structured matrices. Another layer of difficulty is that the number of singular values exceeding a threshold is often unknown. Therefore singular value thresholding involves successively computing more and more top singular values and vectors until hitting below the threshold.

To address these issues, we develop a `MATLAB` wrapper function `svt` for the SVT computation. It is compatible with `MATLAB`'s `svds` function in terms of computing a fixed number of top singular values and vectors of sparse matrices. However it is able to take functional handle input, offering the flexibility to exploit matrix structure. More importantly, it automatically performs singular value thresholding with a user-supplied threshold and can be easily used as a plug-in subroutine in many matrix learning algorithms.

We discuss implementation details in Section 4.2 and describe syntax and example usage in Section 4.3. Section 4.4 evaluates numerical performance of the `svt` function in various situations. We conclude with a discussion in Section 4.5.

4.2 Algorithm and Implementation

Our implementation hinges upon a well-known relationship between the singular value decomposition of a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, $m \geq n$, and the eigenvalue decomposition of the symmetric augmented matrix $\begin{pmatrix} \mathbf{0} & \mathbf{A}^\top \\ \mathbf{A} & \mathbf{0} \end{pmatrix}$ (Golub and Van Loan, 1996, Section 8.6). Let the singular value decomposition of $\mathbf{A}$ be $\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$, where $\mathbf{U} \in \mathbb{R}^{m \times n}$, $\mathbf{\Sigma} \in \mathbb{R}^{n \times n}$ and $\mathbf{V} \in \mathbb{R}^{n \times n}$. Then

$$\begin{pmatrix} \mathbf{0} & \mathbf{A}^\top \\ \mathbf{A} & \mathbf{0} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} \mathbf{V} & \mathbf{V} \\ \mathbf{U} & -\mathbf{U} \end{pmatrix} \cdot \begin{pmatrix} \mathbf{\Sigma} & \mathbf{0} \\ \mathbf{0} & -\mathbf{\Sigma} \end{pmatrix} \cdot \frac{1}{\sqrt{2}} \begin{pmatrix} \mathbf{V} & \mathbf{V} \\ \mathbf{U} & -\mathbf{U} \end{pmatrix}^\top . \tag{4.3}$$

Therefore SVD of $\mathbf{A}$ can be computed via the eigen-decomposition of the augmented matrix. Our wrapper function utilizes `MATLAB`'s built-in `eigs` function for computing top eigenvalue and eigenvectors of large, sparse or structured matrix.

In absence of a threshold, `svt` is similar to `svds` and calculates the top singular values and vectors. Since we allow function handle input, users can always take advantage of special structure in matrices by writing a user defined function for calculating matrix-vector multiplication. This is one merit of `svt` comparing with `MATLAB`'s `svds`.

With a user input threshold, `svt` does singular value thresholding in a sequential manner. It first computes the top k (default is 6) singular values and vectors. Two methods have been implemented to gradually build up the requested subspace. Let $\mathbf{U}_r$, $\mathbf{V}_r$ and σ_i, $i = 1, \ldots, r$, be the singular values and vectors accrued so far. In the `deflation` method (Algorithm 3), we obtain next batch of *incre* (default is 5) singular values and vectors by working on the deflated matrix $\mathbf{A} - \mathbf{U}_r \mathrm{diag}(\sigma_1, \ldots, \sigma_r)\mathbf{V}_r^\top$. In the `succession` method (Algorithm 4), originally hinted in Cai et al. (2010), we work on $\mathbf{A}$ directly and retrieve top k, $k + incre$, $k + 2 * incre$, ... singular values and vectors of the original matrix $\mathbf{A}$ successively. Both algorithms terminate as soon as a singular value below the threshold is identified. Efficiency of these two algorithms are compared in Section 4.4.5.

Algorithm 3: Singular value thresholding based on deflation method

Initialization: $mat = [0, A^\top; A, 0]$, $iter = \min(m, n)$;
while $iter > 0$ **do**
  $[eigvec, eigval] \leftarrow eigs(mat, k)$;
  $i \leftarrow i_{\{eigval<=\lambda\}}$;
  if $i \neq na$ **then**
    $w \leftarrow [w, eigvec_{(:,1:i-1)}]$;
    $e \leftarrow [e, eigval_{(1:i-1)}]$;
    break
  else
    $w \leftarrow [w, eigvec]$;
    $e \leftarrow [e, eigval]$;
  end
  $iter \leftarrow iter - k$;
  $k \leftarrow \min(incre, iter)$;
  $mat \leftarrow mat - w * e * w^\top$;
end
$S \leftarrow e$;
$w \leftarrow \sqrt{2} * w$;
$U \leftarrow w_{(n+1:end,:)}$;
$V \leftarrow w_{(1:n,:)}$;
return $[U, S, V]$

Algorithm 4: Singular value thresholding based on succession method

Initialize $mat = [0, A^\top; A, 0]$, $iter = \min(m, n)$;
while $iter > 0$ **do**
  $[eigvec, eigval] \leftarrow eigs(mat, k)$;
  $i \leftarrow i_{\{eigval<=\lambda\}}$;
  if $i \neq na$ **then**
    $w \leftarrow eigvec_{(:,1:i-1)}$;
    $e \leftarrow eigval_{(1:i-1)}$;
    break
  else
    $w \leftarrow eigvec$;
    $e \leftarrow eigval$;
  end
  $iter \leftarrow iter - k$;
  $k \leftarrow min(k + incre, iter)$;
end
$S \leftarrow e$;
$w \leftarrow \sqrt{2} * w$;
$U \leftarrow w_{(n+1:end,:)}$;
$V \leftarrow w_{(1:n,:)}$;
return $[U, S, V]$

4.3 The MATLAB Function Aspect

We demonstrate various usages of `svt` in this section. A complete demonstration script with output is available on the software webpage `http://hua-zhou.github.io/svt/`.

To find the top k singular values and vectors of a matrix $\mathbf{A}$, the usage is same as `MATLAB`'s build-in function `svds`. $\mathbf{A}$ can be either full or sparse. By default, it computes the top 6 singular

values and vectors

```
>[U, S, V] = svt(A)
```

To request top 15 singular values and vectors, we use

```
>[U, S, V] = svt(A, 'k', 15)
```

User can also supply a function handle, instead of the matrix itself, that computes matrix-vector multiplication. This allows `svt` to utilize special structure other than sparsity. For example, suppose $\mathbf{A}$ is a 1000-by-1000 "sparse plus low rank" matrix $\mathbf{M}+\mathbf{L}\mathbf{R}^\top$, where $\mathbf{M}$ is sparse and $\mathbf{L}, \mathbf{R} \in \mathbb{R}^{1000\times 5}$ are two skinny and tall matrices. To compute the top 15 singular values and vectors, we first define a function that computes $\mathbf{A}\mathbf{v}$ or $\mathbf{w}^\top\mathbf{A}$ for arbitrary vectors $\mathbf{v}, \mathbf{w}$ of compatible dimensions

```
function Av = Afun(v, trans)
if trans
Av = (v' * M)' + R * (v' * L)';
else
Av = M * v + L * (R' * v);
end
end
```

and then call

```
>[U, S, V] = svt(Afun, 'k', 15, 'm', 1000, 'n', 1000)
```

Note the function `Afun` needs to have access to the variables `M`, `L` and `R` and is best declared as a sub-function in the main computation routine. The dimensions of matrix are required when using a functional handle. `'m'` is the number of rows and `'n'` is the number of columns.

Great convenience of `svt` comes from singular value thresholding. That is to compute the singular values that exceed a threshold λ and associated singular vectors. The code

```
>[U, S, V] = svt(A, 'lambda', 0.1)
```

computes the singular values and vectors of a matrix **A** that exceed 0.1. **A** can be either full or sparse. For a non-sparse, structured matrix, we can use the same function handle for singular value thresholding

```
>[U, S, V] = svt(Afun, 'lambda', 0.1, 'm', 1000, 'n', 1000)
```

Again the dimensionality of the matrix must be specified by setting `'m'` and `'n'`. By default, `svt` uses the `deflation` method for locating all singular values and vectors above the threshold. Users can change to the succession method by

```
>[U, S, V] = svt(A, 'lambda', 0.1, 'method', 'succession')
```

or

```
>[U, S, V] = svt(Afun, 'lambda', 0.1, 'm', 1000, 'n', 1000, 'method', 'succession')
```

For singular value thresholding, users can specify the number of top singular values to try in the first iteration and the increment size in subsequent iterations by the `'k'` and `'incre'` options respectively. The command

```
>[U, S, V] = svt(A, 'lambda', 0.1, 'k', 15, 'incre', 3)
```

computes top 15 singular values and vectors in the first iteration and then add 3 more in each subsequent iteration until hitting the singular values below threshold 0.1. This option is useful when users have a rough idea how many singular values are above the threshold and can save considerable computation time. The default values are $k = 6$ and $incre = 5$.

4.4 Numerical Experiments

In this section, we evaluate the numerical performance of `svt` in different scenarios and compare with `MATLAB` built-in functions `svd` and `svds`. We conduct these experiments on a desktop with an Intel Quad Core CPU @ 3.20 GHz and 12 GB of RAM. Computing environment is Linux `MATLAB` R2013a 64-bit version. For testing purpose, we use 5 square sparse matrices and 4 rectangular

sparse matrices of varying sizes downloaded from the University of Florida sparse matrix collection (Davis and Hu, 2011). For each numerical task, 10 replicate runs are performed and the average run time and standard error are reported, unless stated otherwise. Sparsity of a matrix $\mathbf{A}$ is defined as the proportion of zero entries, `1 - nnz(A)/numel(A)`.

4.4.1 Top k Singular Values and Vectors of Sparse Matrices

Table 4.1 reports the run times of `svt`, `svds` and `svd` for computing the top 6 singular values and associated vectors of sparse matrices. In this case, `svt` internally calls `svds` thus their run times should be indistinguishable. The huge gain of `svt`/`svds` in large sparse matrices simply demonstrates the advantage of iterative method over the full decomposition method implemented in `svd`.

Matrix	Size	Sparsity	`svt`	`svds`	`svd`
bfwb398	398	0.9816	0.0396(0.0003)	0.0393(0.0004)	0.0450(0.0001)
rdb800l	800	0.9928	0.0944(0.0008)	0.0941(0.0009)	0.2184(0.0007)
tols1090	1090	0.9970	0.0549(0.0007)	0.0592(0.0005)	0.4377(0.0006)
mhd4800b	4800	0.9988	0.0579(0.0029)	0.0536(0.0026)	249.1995(0.0143)
cryg10000	10000	0.9995	0.1550(0.0019)	0.1580(0.0017)	1773.6812(0.2014)

Table 4.1: Top 6 singular values and vectors of sparse matrices by `svt`, `svds` and `svd`. Reported are the average run time (in seconds) and standard error (in parentheses) based on 10 runs.

4.4.2 Top k Singular Values and Vectors of "Sparse + Low rank" Matrices

This example tests the capability of `svt` to take functional handle input. We generate structured matrices by adding a low rank perturbation to a sparse matrix. Let $\mathbf{M} \in \mathbb{R}^{n \times n}$ be a sparse test matrix. We form a "sparse + low rank" matrix $\mathbf{A} = \mathbf{M} + \mathbf{L}\mathbf{R}^\top$, where $\mathbf{L}, \mathbf{R} \in \mathbb{R}^{n \times 10}$ have independent standard normal distributed entries. Table 4.2 shows the average run times of `svt` with function handle input and `svds` with input $\mathbf{A}$ itself to compute top 6 singular values and vectors based on 10 simulation replicates. It clearly shows the advantage of exploiting the special matrix

structure over applying the iterative algorithm to the full matrix directly. The speed-up is up to 100 fold for large matrices.

Matrix	Size	Sparsity	`svt` (fh input)	`svds`
bfwb398	398	0.9816	0.0176(0.0011)	0.0408(0.0009)
rdb800l	800	0.9928	0.0240(0.0005)	0.2115(0.0014)
tols1090	1090	0.9970	0.0780(0.0009)	0.9396(0.0079)
mhd4800b	4800	0.9988	0.0471(0.0001)	5.6700(0.0166)
cryg10000	10000	0.9995	0.1909(0.0022)	44.2213(0.4373)

Table 4.2: Top 6 singular values and vectors of "sparse + low rank" matrices by `svt` and `svds`. Structured matrices are formed by adding a random rank-10 matrix to the original sparse test matrix. Reported are the average run time (in seconds) and standard error (in parentheses) based on 10 simulation replicates.

4.4.3 Singular Value Thresholding of Sparse Matrices

In this example we compare the singular value thresholding capability of `svt` with the strategy of full singular value decomposition by `svd` followed by thresholding on sparse test matrices. The threshold value is pre-determined such that the top 50 singular values are above threshold. By default, `svt` starts with $k = 6$ singular values and then add more than 5 in each subsequent iteration. Results are presented in Table 4.3. For matrices of size less than 1000, `svt` is less efficient due to the overhead of repeated calling iterative algorithms until hitting the threshold. For large matrices, `svt` shows $100 \sim 1000$ fold speed-ups.

4.4.4 Singular Value Thresholding of "Sparse + Low rank" Matrices

This example investigates singular value thresholding of structured matrices. "Sparse + low rank" matrices are generated by the same mechanism as in Section 4.4.2. Results in Table 4.4 show roughly the same pattern as in Table 4.4. Speed-up of `svt` is most eminent for large matrices. To evaluate the effectiveness of exploiting structure in singular value thresholding, we also call `svt` with input

Matrix	Size	Sparsity	`svt`	`svd`
bfwb398	398	0.9816	0.3633(0.0012)	0.0456(0.0001)
rdb800l	800	0.9928	0.7716(0.0047)	0.2237(0.0005)
tols1090	1090	0.9970	0.4295(0.0012)	0.4451(0.0011)
mhd4800b	4800	0.9988	1.3733(0.0075)	249.4558(0.0423)
cryg10000	10000	0.9995	3.1157(0.0152)	1773.0692(0.3403)

Table 4.3: Singular value thresholding of sparse matrices by `svt` and `svd`. Reported are the average run time (in seconds) and standard error (in parentheses) based on 10 runs. The threshold value is pre-determined to catch the top 50 singular values.

A directly, which apparently compromises efficiency.

Matrix	Size	Sparsity	`svt` (fh input)	`svt` (matrix input)	`svd`
bfwb398	398	0.9816	1.3540(0.1486)	1.4209(0.1744)	0.0502(0.0002)
rdb800l	800	0.9928	2.4144(0.0089)	2.8655(0.0194)	0.2569(0.0005)
tols1090	1090	0.9970	0.5100(0.0023)	1.3044(0.0051)	0.4455(0.0005)
mhd4800b	4800	0.9988	5.6852(0.1462)	89.9854(3.3959)	48.9117(0.0122)
cryg10000	10000	0.9995	3.5793(0.0145)	104.0540(0.2411)	443.3518(0.1034)

Table 4.4: Singular value thresholding of "sparse + low rank" matrices. Reported are the average run time (in seconds) and standard error (in parentheses) based on 10 simulation replicates. Structured matrices are formed by adding a random rank-10 matrix to the original sparse test matrix. The threshold value is pre-determined to catch the top 50 singular values.

4.4.5 Deflation versus Succession Method for Singular Value Thresholding

Table 4.5 compares the efficiency of the deflation and succession strategies for singular value thresholding of sparse test matrices. The threshold value is pre-determined such that the top 50 singular values are above threshold. Both methods start with $k = 6$ singular values and then add 5 more in each subsequent iteration. Deflation method is in general faster than the succession method.

Similar comparison is done on "Sparse + Low rank" structured matrices, which are generated in the same way as in Section 4.4.2. Threshold is again set at the 50th singular value of each matrix.

Matrix	Size	Sparsity	Deflation	Succession
bfwb398	398	0.9816	0.3626(0.0012)	0.4055(0.0026)
rdb800l	800	0.9928	0.7636(0.0048)	0.8670(0.0019)
tols1090	1090	0.9970	0.4250(0.0016)	0.5167(0.0013)
mhd4800b	4800	0.9988	1.3761(0.0110)	2.3382(0.0227)
cryg10000	10000	0.9995	3.1782(0.0173)	5.8789(0.0648)

Table 4.5: Comparison of deflation and succession methods for singular value thresholding of sparse matrices. Reported are the average run time (in seconds) and standard error (in parentheses) based on 10 runs. The threshold is pre-determined to catch the top 50 singular values.

The average run time and standard error are reported in Table 4.6. We found non-convergence of underlying `ARPACK` routine when applying deflation method to `rdb800l` and `mhd4800` matrices. The non-convergence are caused by clustered eigenvalues. It is well known that `ARPACK` works best for finding eigenvalues with large separation between unwanted ones, and non-convergence is typical when dealing with ill conditioned matrices (Lehoucq and Sorensen, 1996). When this happens, we restart with the succession method and continue from the current subspace.

Matrix	Size	Sparsity	Deflation	Succession
bfwb398	398	0.9816	1.2936(0.0588)	2.0956(0.0059)
rdb800l	800	0.9928	2.3758(0.0187)	1.3863(0.0118)
tols1090	1090	0.9970	0.5084(0.0022)	0.6088(0.0011)
mhd4800b	4800	0.9988	5.6008(0.1598)	4.4027(0.0396)
cryg10000	10000	0.9995	3.5636(0.0129)	6.3697(0.0621)

Table 4.6: Comparison of deflation and succession methods for singular value thresholding of "sparse + low rank" matrices. Reported are the average run time (in seconds) and standard error (in parentheses) based on 10 simulation replicates. The threshold is pre-determined to catch the top 50 singular values.

4.4.6 Large-Scale Singular Value Thresholding

The purpose of this section is to demonstrate the performance of `svt` on large rectangular matrices. For the first two test matrices (`bibd_20_10` and `bibd_22_8`), "sparse + low rank" matrices are generated by the same mechanism as in Section 4.4.2. For the other two matrices (`sotrmG2_1000` and `tp-6`), singular value thresholding is performed on the original sparse matrices. Threshold is set at the 5th, 20th, and 50th singular value of each matrix respectively. Table 4.7 displays the run time of `svt` from one replicate. Due to the memory limitation, `svd` cannot handle these 4 problems on this test machine.

Matrix	Size	Sparsity	5th	20th	50th
bibd_20_10	(190, 184756)	0.7632	0.0350	0.6152	11.2083
bibd_22_8	(231, 319770)	0.8788	0.0372	2.6438	4.1058
stormG2_1000	(528185, 1377306)	0.9999	0.2518	1.0394	12.6890
tp-6	(142752, 1014301)	0.9999	1.6373	20.2409	41.3021

Table 4.7: Singular value thresholding of large rectangular matrices. Reported are the run time (in minutes) of `svt` from one replicate. The threshold value is pre-determined to catch the top 5, 20, and 50 singular values respectively.

4.4.7 Application to Matrix Completion Problem

To demonstrate the effectiveness of `svt` as a plug-in computational routine in practice, we conduct a numerical experiment on the spectral regularization algorithm for matrix completion (Mazumder et al., 2010), which minimizes

$$\frac{1}{2}\sum_{(i,j)\in\Omega}(x_{ij}-y_{ij})^2+\lambda\|\mathbf{X}\|_* \tag{4.4}$$

at a grid of tuning parameter values λ. Here Ω indexes the observed entries y_{ij} and $\mathbf{X}=(x_{ij})$ is the completed matrix. Algorithm 5 lists the computational algorithm, which involves repeated

singular value thresholding (lines 4-6). See Chi et al. (2013) for a derivation from the majorization-minimization (MM) point of view. Although $\mathbf{A}^{(t)}$ is is a dense matrix, it can be written as

$$\begin{aligned} \mathbf{A}^{(t)} &= P_{\Omega}(\mathbf{Y}) + P_{\Omega^{\perp}}(\mathbf{X}^{(t)}) \\ &= [P_{\Omega}(\mathbf{Y}) - P_{\Omega}(\mathbf{X}^{(t)})] + \mathbf{X}^{(t)}, \end{aligned}$$

where $\mathbf{X}^{(t)}$ is a low rank matrix at large values of λ (only few singular values survive after thresholding). Fortunately, in many applications, large values of λ are the regime of interest, which encourages low rank solutions. That means most of time $\mathbf{A}^{(t)}$ is of the special form "sparse + low rank" that enables extremely fast matrix-vector multiplication.

Algorithm 5: MM algorithm for minimizing the penalized loss (4.4).

1 Initialize $\mathbf{X}^{(0)}$;
2 **repeat**
3 $\quad \mathbf{A}^{(t)} \leftarrow P_{\Omega}(\mathbf{Y}) + P_{\Omega^{\perp}}(\mathbf{X}^{(t)})$;
4 $\quad$ SVD $\mathbf{U}\text{diag}(\mathbf{a}^{(t)})\mathbf{V}^{\top} \leftarrow \mathbf{A}^{(t)}$;
5 $\quad \mathbf{x}^{(t+1)} \leftarrow (\mathbf{a}^{(t)} - \lambda)_{+}$;
6 $\quad \mathbf{X}^{(t+1)} \leftarrow \mathbf{U}\text{diag}(\mathbf{x}^{(t+1)})\mathbf{V}^{\top}$;
7 **until** *objective value converges*;

In the numerical experiment, we generate a rank-5 matrix by multiplying two matrices $\mathbf{M} = \mathbf{L}\mathbf{R}^{\top}$, where $\mathbf{L}, \mathbf{R} \in \mathbb{R}^{n \times 5}$ have independent standard normal distributed entries. Then, we add independent standard Gaussian noise to corrupt the original parameter matrix $\mathbf{M}$, that is $\mathbf{Y} = \mathbf{M} + \boldsymbol{\epsilon}$. 5% entries of $\mathbf{Y}$ are randomly chosen to be observed. The dimension n of our synthetic data ranges from 500 to 5000. For each n, we minimize (4.4) at a grid of 20 points. The grid is set up in a linear manner as in (Mazumder et al., 2010)

```
>lambdas = linspace(maxlambda * 0.9, maxlambda / 5, 20)
```

Here `maxlambda` is the largest singular value of the input matrix $\mathbf{Y}$ with missing entries set at 0. Warm start strategy is used. That is solution at a previous λ is used as the start point for the next

λ. Path following is terminated whenever all 20 grid points are exhausted or the rank of solution exceeds 10 (twice the true rank). Three methods for singular value thresholding are tested: `svt` using functional handle input, `svt` using matrix input $\mathbf{A}^{(t)}$, and full singular value thresholding by `svd` followed by thresholding. Table 4.8 shows the run time in minutes for obtaining the whole solution path. Speed-up of `svt` increases with matrix size and utilizing the "sparse + low rank" structure via functional handle boosts the performance.

Size	Sparsity	Rank	Grid points	`svt` (fh input)	`svt` (matrix input)	`svd`
500	0.95	5	15	0.8541	0.7335	0.5466
1000	0.95	5	19	2.9359	4.0803	4.2763
2000	0.95	5	20	10.3611	35.7058	40.1562
3000	0.95	5	20	20.6781	69.3164	138.8011
4000	0.95	5	20	52.2150	175.2524	335.2373
5000	0.95	5	20	71.8051	246.3738	630.2729

Table 4.8: Run time of matrix completion problem using different singular value thresholding methods. Reported are the run time (in minutes) for whole solution path. Path following is terminated whenever 20 grid points are exhausted or the rank of solution goes beyond 10 (twice the true rank).

4.5 Discussion

We develop a `MATLAB` wrapper function `svt` for singular value thresholding. When a fixed number of top singular values and vectors are requested, `svt` expands the capability of `MATLAB`'s built-in function `svds` by allowing function handle input. This enables application of the iterative method to dense but structured large matrices. More conveniently, `svt` provides a simple interface for singular value thresholding, the key step in many matrix learning algorithms. Our numerical examples have demonstrated efficiency of `svt` in various situations. The `svt` package is continuously developed and maintained at GitHub `http://hua-zhou.github.io/svt/`.

We describe a few future directions here. Our wrapper function utilizes the well-known rela-

tionship between SVD and eigen-decomposition of the augmented matrix (4.3) and builds on the `MATLAB`'s `eigs` function, which in turn calls the `ARPACK` subroutines (Lehoucq et al., 1997) for solving large scale eigenproblems. An alternative is to use the `PROPACK` library (Larsen, 1998), an efficient package for singular value decomposition of sparse or structured matrices. This involves distributing extra source code or compiled program but may further improves efficiency. Both `ARPACK` and `PROPACK` implement Krylov subspace method and compute a fixed number of top eigenvalues or singular values. Thus singular value thresholding has to be done in a sequential manner. The recent `FEAST` package (Plizzi, 2012) is an innovative method for solving standard or generalized eigenvalue problem, and is able to compute all the eigenvalues and eigenvectors within a given search interval, which is particularly attractive for the singular value thresholding task. However user must provide an initial value for the number of eigenvalues in the search interval. If the initial guess is too small, the program will exit. In real applications of singular value thresholding, such an estimate may be hard to obtain. Further investigation of the feasibility of using `FEAST` for singular value thresholding is underway.

Chapter 5

Future Directions on Optimal Design for Functional Data

In this chapter, we describe a list of open problems in optimal design for functional data and present some partial results of the ongoing projects. We follow the notations introduced in Chapter 2 and assume the functional data $X(\cdot)$ has zero mean.

Park et al. (2016) proposed to formulate optimal designs with target of recovering individual functions and predicting a scalar outcome by using a general design objective function

$$\mathcal{M}_{\mathbf{B}}(\mathbf{t}) = \mathrm{tr}(\mathbf{B}\boldsymbol{\Lambda}) - \mathrm{tr}\left\{\mathbf{B}\mathbf{S}(\mathbf{t})\right\}, \tag{5.1}$$

where $\mathbf{S}(\mathbf{t}) = \boldsymbol{\Lambda}\boldsymbol{\Phi}(\mathbf{t})'\boldsymbol{\Sigma}^{-1}\boldsymbol{\Phi}(\mathbf{t})\boldsymbol{\Lambda}$ and $\mathbf{B}$ is an arbitrary positive semidefinite matrix. We extend this framework to cover some other important regression models in functional data analysis and provide a computationally efficient approach for selecting optimal sampling points.

5.1 Optimal Design for Functional Concurrent Linear Model

Consider the concurrent functional linear model

$$y_i(t) = \alpha(t) + \beta(t)X_i(t) + \epsilon_i(t), \tag{5.2}$$

where $y_i(t)$ is a functional response for subject i measured at $t \in \mathcal{T}$, $\alpha(t)$ is a functional intercept, $\beta(t)$ is a functional slope and is assumed to be smooth and square integrable in $\mathcal{T}$, $\epsilon_i(t)$ is an error process with mean zero and is independent of $X_i(t)$. For simplicity, we assume $\alpha(t) = 0$.

We define the objective function as the mean squared error for predicting the new subject's mean outcome,

$$\mathcal{M}_{\mathbf{B}_1}(\mathbf{t}) := \mathbb{E}\int_{\mathcal{T}} \left[X_{i^*}(t)\beta(t) - \mathbb{E}\left\{X_{i^*}(t)|\mathbf{W}_{i^*}(\mathbf{t})\right\}\beta(t)\right]^2 dt. \tag{5.3}$$

It can shown that

$$\begin{aligned}
\mathcal{M}_{\mathbf{B}_1}(\mathbf{t}) &= \mathbb{E}\int \left\{X(t)_{i^*}\beta(t)\right\}^2 dt - \mathbb{E}\int \left[\mathbb{E}\left\{X_{i^*}(t)|\mathbf{W}_{i^*}(\mathbf{t})\right\}\beta(t)\right]^2 dt \\
&= \int \beta^2(t)r(t,t)dt - \int \beta^2(t)r(t,\mathbf{t})'\mathbf{\Sigma}^{-1}r(t,\mathbf{t})dt \\
&= \int \beta^2(t)r(t,t)dt - \mathrm{tr}\left\{\left[\int \beta^2(t)\mathbf{\Phi}(t)\mathbf{\Phi}(t)'dt\right]\mathbf{\Lambda}\mathbf{\Phi}(\mathbf{t})'\mathbf{\Sigma}^{-1}\mathbf{\Phi}(\mathbf{t})\mathbf{\Lambda}\right\}.
\end{aligned}$$

Therefore, $\mathcal{M}_{\mathbf{B}_1}(\mathbf{t})$ can be simplified into the form in (5.1), where $\mathbf{B}_1 = \int \beta^2(t)\mathbf{\Phi}(t)\mathbf{\Phi}(t)'dt$ is a positive semidefinite matrix with $[\mathbf{B}_1]_{\ell,\ell'} = \int \beta^2(t)\phi_\ell(t)\phi_{\ell'}(t)dt$.

5.2 Optimal Design for Function-on-Function Linear Model

Consider the function-on-function linear model

$$y_i(t) = \alpha(t) + \int_{\mathcal{T}} \beta(s,t)X_i(s)ds + \epsilon_i(t), \tag{5.4}$$

where $y_i(t)$ is a functional outcome measured at $t \in \mathcal{T}'$, $\alpha(t)$ is a fixed intercept function, $\beta(t,s)$ is assumed to be smooth and square integrable over the domains, $\epsilon_i(t)$ is an error process with mean zero and is uncorrelated with $X_i(s)$. Without loss of generality we assume $\alpha(t) = 0$.

Then the objective function is defined similarly

$$\mathcal{M}_{\mathbf{B}_2}(\mathbf{t}) := \mathbb{E} \int_{\mathcal{T}'} \left[\int_{\mathcal{T}} X_{i^*}(s)\beta(s,t)ds - \int_{\mathcal{T}} \mathbb{E}\left\{ X_{i^*}(s) | \mathbf{W}_{i^*}(\mathbf{t}) \right\} \beta(s,t)ds \right]^2 dt. \tag{5.5}$$

We simplify $\mathcal{M}_{\mathbf{B}_2}(\mathbf{t})$ as

$$\begin{aligned}
\mathcal{M}_{\mathbf{B}_2}(\mathbf{t}) &= \mathbb{E} \int \left\{ \int X_{i^*}(s)\beta(s,t)ds \right\}^2 dt - \mathbb{E} \int \left[\int \mathbb{E}\left\{ X_{i^*}(s) | \mathbf{W}_{i^*}(\mathbf{t}) \right\} \beta(s,t)ds \right]^2 dt \\
&= \int \left\{ \int \int \beta(s_1,t) r(s_1,s_2) \beta(s_2,t) ds_1 ds_2 \right\} dt - \int \left\{ \int \beta(s,t) r(s,\mathbf{t}) ds \right\}' \mathbf{\Sigma}^{-1} \left\{ \int \beta(s,t) r(s,\mathbf{t}) ds \right\} dt.
\end{aligned}$$

Let $\beta(s,t) = \sum_\ell \phi_\ell(s)\beta_\ell(t)$, it leads to

$$\begin{aligned}
& \int \left\{ \int \int \beta(s_1,t) r(s_1,s_2) \beta(s_2,t) ds_1 ds_2 \right\} dt \\
= & \int \left[\int \int \left\{ \sum_\ell \phi_\ell(s_1)\beta_\ell(t) \sum_\ell \phi_\ell(s_1)\phi_\ell(s_2)\lambda_\ell \sum_\ell \phi_\ell(s_2)\beta_\ell(t) \right\} ds_1 ds_2 \right] dt \\
= & \sum_\ell \lambda_\ell \int \beta_\ell(t)\beta_\ell(t) dt.
\end{aligned}$$

Let $\boldsymbol{\beta}(t) = (\beta_1(t), \beta_2(t), \cdots)'$ be an infinite dimensional vector. We have

$$\begin{aligned}
& \int \left\{ \int \beta(s,t) r(s,\mathbf{t}) ds \right\}' \mathbf{\Sigma}^{-1} \left\{ \int \beta(s,t) r(s,\mathbf{t}) ds \right\} dt \\
= & \int \operatorname{tr} \left\{ \boldsymbol{\beta}(t)' \mathbf{\Lambda} \mathbf{\Phi}(\mathbf{t})' \mathbf{\Sigma}^{-1} \mathbf{\Phi}(\mathbf{t}) \mathbf{\Lambda} \boldsymbol{\beta}(t) \right\} dt \\
= & \operatorname{tr} \left\{ \int \boldsymbol{\beta}(t)\boldsymbol{\beta}(t)' dt \mathbf{S}(\mathbf{t}) \right\}.
\end{aligned}$$

Hence, $\mathcal{M}_{\mathbf{B}_2}(\mathbf{t})$ takes the simplified form in (5.1), where $\mathbf{B}_2 = \int \boldsymbol{\beta}(t)\boldsymbol{\beta}(t)' dt$ is an infinite dimensional matrix with $[\mathbf{B}_2]_{\ell,\ell'} = \int \beta_\ell(t)\beta_{\ell'}(t)dt$.

5.3 Optimal Design for Historical Functional Linear Model

Consider the historical function linear model

$$y_i(t) = \alpha(t) + \int_0^t \beta(s,t) X_i(s) ds + \epsilon_i(t). \tag{5.6}$$

This model differs from model (5.4) in that the bounds of integration are from 0 to t now.

The design objective function is defined as

$$\mathcal{M}_{\mathbf{B}_3}(\mathbf{t}) := \mathbb{E} \int_{\mathcal{T}} \left[\int_0^t X_{i^*}(s)\beta(s,t)ds - \int_0^t \mathbb{E}\left\{ X_{i^*}(s) | \mathbf{W}_{i^*}(\mathbf{t}) \right\} \beta(s,t) ds \right]^2 dt. \tag{5.7}$$

It can be further simplified as

$$\mathcal{M}_{\mathbf{B}_3}(\mathbf{t}) \;=\; \mathbb{E} \int \left\{ \int_0^t X_{i^*}(s)\beta(s,t)ds \right\}^2 dt - \mathbb{E} \int \left[\int_0^t \mathbb{E}\left\{ X_{i^*}(s) | \mathbf{W}_{i^*}(\mathbf{t}) \right\} \beta(s,t) ds \right]^2 dt.$$

Let $\beta_\ell(t) = \int_0^t \beta(s,t)\phi_\ell(s)ds$. We have

$$\begin{aligned}
& \mathbb{E} \int \left\{ \int_0^t X_{i^*}(s)\beta(s,t)ds \right\}^2 dt \\
= & \int \left\{ \int_0^t \int_0^t \beta(s_1,t) r(s_1,s_2) \beta(s_2,t) ds_1 ds_2 \right\} dt \\
= & \int \left\{ \sum_\ell \left[\lambda_\ell \int_0^t \beta(s_1,t)\phi_\ell(s_1)ds_1 \int_0^t \phi_\ell(s_2)\beta(s_2,t)ds_2 \right] \right\} dt \\
= & \sum_\ell \lambda_\ell \int \beta_\ell(t)\beta_\ell(t)dt.
\end{aligned}$$

Then we derive that

$$\begin{aligned}
& \mathbb{E}\int\left[\int_0^t \mathbb{E}\left\{X_{i^*}(s)|\mathbf{W}_{i^*}(\mathbf{t})\right\}\beta(s,t)ds\right]^2 dt \\
= & \int\left\{\int_0^t \beta(s,t)r(s,\mathbf{t})ds\right\}' \mathbf{\Sigma}^{-1}\left\{\int_0^t \beta(s,t)r(s,\mathbf{t})ds\right\}dt \\
= & \int \mathrm{tr}\left\{\boldsymbol{\beta}(t)'\mathbf{\Lambda}\mathbf{\Phi}(\mathbf{t})'\mathbf{\Sigma}^{-1}\mathbf{\Phi}(\mathbf{t})\mathbf{\Lambda}\boldsymbol{\beta}(t)\right\}dt \\
= & \mathrm{tr}\left\{\int \boldsymbol{\beta}(t)\boldsymbol{\beta}(t)'dt\mathbf{S}(\mathbf{t})\right\}.
\end{aligned}$$

Thus, we have shown that $\mathcal{M}_{\mathbf{B}_3}(\mathbf{t})$ has the simplified form in (5.1), where $\mathbf{B}_3 = \int \boldsymbol{\beta}(t)\boldsymbol{\beta}(t)'dt$ with $[\mathbf{B}_3]_{\ell,\ell'} = \int\{\int_0^t \beta(s,t)\phi_\ell(s)ds \int_0^t \beta(s,t)\phi_{\ell'}(s)ds\}dt$.

5.4 Search of Optimal Sampling Points

Following Section 2.4.1, instead of conducting an exhaustive search, we propose a computationally efficient approach for selecting optimal sampling points, which exploits the fact that the design objective function in (5.1) is an explicit analytic function of the sampling points $\mathbf{t}$.

Theorem 5. *Assume that the eigenfunctions $\phi(t)$ and the covariance function $r(s,t)$ are differentiable. Let $\widetilde{\mathbf{B}} = \mathbf{\Lambda}\mathbf{B}\mathbf{\Lambda}$ and $\mathbf{A} = \mathbf{\Sigma}^{-1}\mathbf{\Phi}(\mathbf{t})\widetilde{\mathbf{B}} - \mathbf{\Sigma}^{-1}\mathbf{\Phi}(\mathbf{t})\widetilde{\mathbf{B}}\mathbf{\Phi}(\mathbf{t})'\mathbf{\Sigma}^{-1}\mathbf{\Phi}(\mathbf{t})\mathbf{\Lambda}$. Then*

$$\frac{\partial \mathcal{M}_{\mathbf{B}}(\mathbf{t})}{\partial \mathbf{t}} = -2diag\left\{\mathbf{A}\dot{\phi}(\mathbf{t})'\right\},$$

where $\dot{\phi}_\ell(t)$ denotes the derivative of $\phi_\ell(t)$, $\dot{\phi}(\mathbf{t}) = \{\dot{\phi}_\ell(t_k)\}_{1\le k\le p, 1\le \ell\le \infty} \in \mathbb{R}^{p\times\infty}$.

The proof is provided in Appendix D. We can easily evaluate the gradient of $\mathcal{M}_{\mathbf{B}}(\mathbf{t})$ derived in Theorem 5 with the estimated model components, and then use off-the-shelf optimization solver *constrOptim* function in R to search for optimal sampling points with the *BFGS* method, see Section 2.4.1 for details.

REFERENCES

ACM SIGKDD and Netflix (2007). Proceedings of KDD Cup and workshop. In *Proceedings of KDD Cup and Workshop*.

Bachrach, L., T. Hastie, M. Wang, B. Narasimhan, and R. Marcus (1999). Bone mineral acquisition in healthy asian, hispanic, black, and caucasian youth: A longitudinal study 1. *The Journal of Clinical Endocrinology & Metabolism 84*, 4702–4712.

Barlow, R. E., D. J. Bartholomew, J. Bremner, and H. D. Brunk (1972). *Statistical inference under order restrictions: the theory and application of isotonic regression*. Wiley New York.

Bergsma, R. and S. Hermesch (2012). Exploring breeding opportunities for reduced thermal sensitivity of feed intake in the lactating sow. *Journal of animal science 90*(1), 85–98.

Besse, P., H. Cardot, and F. Ferraty (1997). Simultaneous nonparametric regressions of unbalanced longitudinal data. *Comput. Statist. Data Anal. 24*, 255–270.

Besse, P. and J. O. Ramsay (1986). Principal components analysis of sampled functions. *Psychometrika 51*, 285–311.

Black, J., B. Mullan, M. Lorschy, and L. Giles (1993). Lactation in the sow during heat stress. *Livestock production science 35*(1), 153–170.

Bloemhof, S., P. Mathur, E. Knol, and E. Van der Waaij (2013). Effect of daily environmental temperature on farrowing rate and total born in dam line sows. *Journal of animal science 91*(6), 2667–2679.

Bondell, H., B. Reich, and H. Wang (2010). Noncrossing quantile regression curve estimation. *Biometrika 97*(4), 825–838.

Boyd, S., N. Parikh, E. Chu, B. Peleato, and J. Eckstein (2011, January). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn. 3*(1), 1–122.

Bunea, F. and L. Xiao (2015). On the sample covariance matrix estimator of reduced effective rank population matrices, with applications to fpca. *Bernoulli 21*, 1200–1230.

Cai, J.-F., E. J. Candès, and Z. Shen (2010). A singular value thresholding algorithm for matrix completion. *SIAM J. Optim. 20*(4), 1956–1982.

Cai, T. and M. Yuan (2012). Nonparametric covariance function estimation for functional and longitudinal data. Technical report, Univ. Pennsylvania, Philadelphia, PA.

Candès, E. J. and B. Recht (2009). Exact matrix completion via convex optimization. *Found. Comput. Math. 9*(6), 717–772.

Cardot, H., C. Crambes, and P. Sarda (2005). Quantile regression when the covariates are functions. *Nonparametric Statistics 17*(7), 841–856.

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

Carey, J., P. Liedo, H. Müller, J. Wang, and J. Chiou (1998). Relationship of age patterns of fecundity to mortality, longevity, and lifetime reproduction in a large cohort of mediterranean fruit fly females. *The Journals of Gerontology Series A: Biological Sciences and Medical Sciences 53*(4), B245–B251.

Cederbaum, J., M. Pouplier, P. Hoole, and S. Greven (2016). Functional linear mixed models for irregularly or sparsely sampled data. *Statistical Modelling 16*(1), 67–88.

Chen, H. and Y. Wang (2011). A penalized spline approach to functional mixed effects model analysis. *Biometrics 67*(3), 861–870.

Chen, K. and H.-G. Müller (2012). Conditional quantile analysis when covariates are functions, with application to growth data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 74*(1), 67–89.

Chen, Y., J. Goldsmith, and T. Ogden (2015). Variable selection in function-on-scalar regression.

Chernozhukov, V., I. Fernandez-Val, and A. Galichon (2009). Improving point and interval estimators of monotone functions by rearrangement. *Biometrika*, asp030.

Chi, E. C., H. Zhou, G. K. Chen, D. O. Del Vecchyo, and K. Lange (2013). Genotype imputation via matrix completion. *Genome Research 23*(3), 509–518.

Dai, X., H. Müller, and F. Yao (2016). Optimal bayes classifiers for functional data and density ratios. Available at `https://arxiv.org/abs/1605.03707`.

Davis, T. A. and Y. Hu (2011). The University of Florida sparse matrix collection. *ACM Trans. Math. Software 38*(1), Art. 1, 25.

de Boor, C. (1978). *A Practical Guide to Splines*. Berlin: Springer.

Delaigle, A. and P. Hall (2012). Achieving near perfect classification for functional data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 74*, 267–286.

Delaigle, A. and P. Hall (2013). Classification using censored functional data. *Journal of the American Statistical Association 108*, 1269–1283.

Delaigle, A., P. Hall, and N. Bathia (2012). Componentwise classification and clustering of functional data. *Biometrika 99*, 299.

Diggle, P., P. Heagerty, K.-Y. Liang, and S. Zeger (1994). *Analysis of longitudinal data*. Oxford, U.K.: Oxford University Press.

Dobbin, K. and R. Simon (2007). Sample size planning for developing classifiers using high-dimensional dna microarray data. *Biostatistics 8*, 101–117.

Durban, M., J. Harezlak, M. P. Wand, and R. J. Carroll (2005). Simple fitting of subject-specific curves for longitudinal data. *Statistics in Medicine 24*(8), 1153–1167.

Eilers, P. and B. Marx (1996). Flexible smoothing with B-splines and penalties (with Discussion). *Statist. Sci. 11*, 89–121.

Eilers, P. and B. Marx (2003). Multivariate calibration with temperature interaction using two-dimensional penalized signal regression. *Chemometrics and Intelligent Laboratory Systems 66*, 159–174.

Fan, J. and I. Gijbels (1996). *Local polynomial modelling and its applications*. London: Chapman&Hall/CRC.

Fanaee-T, H. and J. Gama (2013). Event labeling combining ensemble detectors and background knowledge. *Progress in Artificial Intelligence*, 1–15.

Ferraty, F., P. Hall, and P. Vieu (2010). Most-predictive design points for functional data predictors. *Biometrika 97*, 807.

Ferraty, F., A. Rabhi, and P. Vieu (2005). Conditional quantiles for dependent functional data with application to the climatic" el niño" phenomenon. *Sankhyā: The Indian Journal of Statistics*, 378–398.

Ferraty, F. and P. Vieu (2002). The functional nonparametric model and application to spectrometric data. *Computational Statistics 17*, 545–564.

Ferraty, F. and P. Vieu (2003). Curves discrimination: a nonparametric functional approach. *Computational Statistics & Data Analysis 44*, 161–173.

Ferraty, F. and P. Vieu (2006). *Nonparametric Functional Data Analysis*. New York: Springer.

Ferraty, F. and P. Vieu (2009). Additive prediction and boosting for functional data. *Computational Statistics & Data Analysis 53*(4), 1400–1413.

Galeano, P., E. Joseph, and R. Lillo (2015). The mahalanobis distance for functional data with applications to classification. *Technometrics 57*, 281–291.

Gertheiss, J., A. Maity, and A.-M. Staicu (2013). Variable selection in generalized functional linear models. *Stat 2*(1), 86–101.

Goldsmith, J., J. Bobb, C. Crainiceanu, B. Caffo, and D. Reich (2011). Penalized functional regression. *J. Comput. Graph. Statist. 20*, 830–851.

Goldsmith, J., C. Crainiceanu, B. Caffo, and D. Reich (2012). Longitudinal penalized functional regression for cognitive outcomes on neuronal tract measurements. *Journal of the Royal Statistical Society: Series C (Applied Statistics) 61*, 453–469.

Goldsmith, J., S. Greven, and C. Crainiceanu (2013). Corrected confidence bands for functional data using principal components. *Biometrics 69*(1), 41–51.

Goldstein, T., C. Studer, and R. Baraniuk (2015, January). `FASTA`: A generalized implementation of forward-backward splitting.

Golub, G. H. and C. F. Van Loan (1996). *Matrix Computations* (Third ed.). Johns Hopkins Studies in the Mathematical Sciences. Baltimore, MD: Johns Hopkins University Press.

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

Huang, L., F. Scheipl, J. Goldsmith, J. Gellar, J. Harezlak, M. Mclean, B. Swihart, L. Xiao, C. Crainiceanu, P. Reiss, Y. Chen, S. Greven, L. Huo, M. Kundu, and J. Wrobel (2015). R package *refund*: Methodology for regression with functional data (version 0.1-13). URL: `https://cran.r-project.org/web/packages/refund/index.html`.

Isserlis, L. (1918, November). On a formula for the product-moment coefficient of any order of a normal frequency distribution in any number of variables. *Biometrika 12*(1-2), 134–139.

Ivanescu, A. E., A.-M. Staicu, F. Scheipl, and S. Greven (2015). Penalized function-on-function regression. *Computational Statistics*, 1–30.

James, G. and T. Hastie (2001). Functional linear discriminant analysis for irregularly sampled curves. *Journal of the Royal Statistical Society. Series B (Statistical Methodology) 63*, 533–550.

James, G., T. Hastie, and C. Sugar (2000). Principal component models for sparse functional data. *Biometrika 87*, 587–602.

James, G. M. (2002). Generalized linear models with functional predictors. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 64*(3), 411–432.

Ji, H. and H. Müller (2016). Optimal designs for longitudinal and functional data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) n/a*, n/a–n/a.

Jiang, C., J. Aston, and J. Wang (2016). A functional approach to deconvolve dynamic neuroimaging data. *Journal of the American Statistical Association 111*, 1–13.

Johnston, L., M. Ellis, G. Libal, V. Mayrose, and W. Weldon (1999). Effect of room temperature and dietary amino acid concentration on performance of lactating sows. ncr-89 committee on swine management. *Journal of animal science 77*(7), 1638–1644.

Kalofolias, V., X. Bresson, M. M. Bronstein, and P. Vandergheynst (2014). Matrix completion on graphs. *CoRR abs/1408.1717*.

Kaslow, R. A., D. G. Ostrow, R. Detels, J. P. Phair, B. F. Polk, and C. R. Rinaldo (1987). The multicenter aids cohort study: rationale, organization, and selected characteristics of the participants. *American Journal of Epidemiology 126*(2), 310–318.

Kato, K. (2012). Estimation in functional linear quantile regression. *The Annals of Statistics 40*(6), 3108–3136.

Kneip, A. (1994). Nonparametric estimation of common regressors for similar curve data. *Ann. Statist. 22*, 1386–1427.

Koenker, R. (2005). *Quantile regression*. Number 38. Cambridge university press.

Koenker, R. and G. Bassett Jr (1978). Regression quantiles. *Econometrica: journal of the Econometric Society*, 33–50.

Lange, K., E. C. Chi, and H. Zhou (2014). A brief survey of modern optimization for statisticians. *International Statistical Review 82*(1), 46–70.

Larsen, R. M. (1998). Lanczos bidiagonalization with partial reorthogonalization.

Lehoucq, R. B. and D. C. Sorensen (1996, October). Deflation techniques for an implicitly restarted arnoldi iteration. *SIAM J. Matrix Anal. Appl. 17*(4), 789–821.

Lehoucq, R. B., D. C. Sorensen, and C. Yang (1997). Arpack users guide: Solution of large scale eigenvalue problems by implicitly restarted arnoldi methods.

Leng, X. and H. Müller (2006). Classification using functional data analysis for temporal gene expression data. *Bioinformatics 22*, 68–76.

Li, Y. and T. Hsing (2010). Uniform convergence rates for nonparametric regression and principal component analysis in functional/longitudinal data. *Ann. Statist. 38*(6), 3321–3351.

Lichman, M. (2013). UCI machine learning repository.

Lindquist, M. (2012). Functional causal mediation analysis with an application to brain connectivity. *Journal of the American Statistical Association 107*, 1297–1309.

Lu, Y., J. Du, and Z. Sun (2014). Functional partially linear quantile regression model. *Metrika 77*(2), 317–332.

Marx, B. and P. Eilers (2005). Multidimensional Penalized Signal Regression. *Technometrics 47*, 13–22.

Mazumder, R., T. Hastie, and R. Tibshirani (2010). Spectral regularization algorithms for learning large incomplete matrices. *Journal of Machine Learning Research 11*, 2287–2322.

McLean, M., G. Hooker, A.-M. Staicu, F. Scheipl, and R. Ruppert (2014). Functional generalized additive models. *Journal of Computational and Graphical Statistics 23*, 249–269.

Meister, A. (2016). Optimal classification and nonparametric regression for functional data. *Bernoulli 22*, 1729–1744.

Melillo, J. M., T. T. Richmond, and G. Yohe (2014). Climate change impacts in the united states. *Third National Climate Assessment*.

Morris, J., C. Arroyo, B. Coull, L. Ryan, R. Herrick, and S. Gortmaker (2006). Using wavelet-based functional mixed models to characterize population heterogeneity in accelerometer profiles: A case study. *Journal of the American Statistical Association 101*, 1352–1364.

Müller, H. (2005). Functional modelling and classification of longitudinal data. *Scandinavian Journal of Statistics 32*, 223–240.

Müller, H., J. R. Carey, D. Wu, P. Liedo, and J. W. Vaupel (2001). Reproductive potential predicts longevity of female mediterranean fruitflies. *Proceedings of the Royal Society of London B: Biological Sciences 268*(1466), 445–450.

Nelson, G. C., M. W. Rosegrant, J. Koo, R. Robertson, T. Sulser, T. Zhu, C. Ringler, S. Msangi, A. Palazzo, M. Batka, et al. (2009). *Climate change: Impact on agriculture and costs of adaptation*, Volume 21. Intl Food Policy Res Inst.

Ng, P. and M. Maechler (2007). A fast and efficient implementation of qualitatively constrained quantile smoothing splines. *Statistical Modelling 7*(4), 315–328.

Otazo, R., E. Candès, and D. K. Sodickson (2015). Low-rank plus sparse matrix decomposition for accelerated dynamic mri with separation of background and dynamic components. *Magnetic Resonance in Medicine 73*, 1125–1136.

Parikh, N. and S. Boyd (2013). Proximal algorithms. *Found. Trends Mach. Learn. 1*(3), 123–231.

Park, S., L. Xiao, J. Willbur, A. Staicu, and N. Jumbe (2016). A joint optimal design for functional data with application to scheduling ultrasound scans. Manuscript.

Peng, J. and D. Paul (2009). A geometric approach to maximum likelihood estimation of functional principal components from sparse longitudinal data. *J. Comput. Graph. Stat. 18*, 995–1015.

Plizzi, E. (2012). A high-performance numerical library for solving eigenvalue problems: Feast solver user's guide.

Quiniou, N. and J. Noblet (1999). Influence of high ambient temperatures on performance of multiparous lactating sows. *Journal of animal science 77*, 2124–2134.

Ramsay, J. and C. J. Dalzell (1991). Some tools for functional data analysis (with Discussion). *J. R. Statist. Soc. B 53*, 539–572.

Ramsay, J. and B. Silverman (2005). *Functional data analysis*. New York: Springer.

Ramsay, J. and B. W. Silverman (2002). *Applied Functional data analysis: Methods and Case Studies*. New York: Springer.

Ramsay, J. O., H. Wickham, S. Graves, and G. Hooker (2011). R package *fda*: Functional data analysis in r data (version 2.2.6). URL:`https://cran.r-project.org/web/packages/fda/index.html`.

Reich, B., H. Bondell, and H. Wang (2010). Flexible bayesian quantile regression for independent and clustered data. *Biostatistics 11*(2), 337–352.

Reimherr, M. and D. Nicolae (2014). A functional data analysis approach for genetic association studies. *The Annals of Applied Statistics 8*, 406–429.

Reimherr, M. and D. Nicolae (2016). Estimating variance components in functional linear models with applications to genetic heritability. *Journal of the American Statistical Association 111*, 407–422.

Reiss, P. and R. Ogden (2010). Functional generalized linear models with images as predictors. *Biometrics 66*, 61–69.

Reiss, P. T. and R. Todd Ogden (2009). Smoothing parameter selection for a class of semiparametric linear models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 71*(2), 505–523.

Renaudeau, D. and J. Noblet (2001). Effects of exposure to high ambient temperature and dietary protein level on sow milk production and performance of piglets. *Journal of animal science 79*(6), 1540–1548.

Renaudeau, D., N. Quiniou, and J. Noblet (2001). Effects of exposure to high ambient temperature and dietary protein level on performance of multiparous lactating sows. *Journal of Animal Science 79*(5), 1240–1249.

Rodríguez-Álvarez, M. X., D.-J. Lee, T. Kneib, M. Durbán, and P. Eilers (2015). Fast smoothing parameter separation in multidimensional generalized p-splines: the sap algorithm. *Statistics and Computing 25*(5), 941–957.

Rosero, D. S., R. D. Boyd, M. McCulley, J. Odle, and E. van Heugten (2016). Essential fatty acid supplementation during lactation is required to maximize the subsequent reproductive performance of the modern sow. *Animal Reproduction Science 168*, 151–163.

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

Ruppert, D., M. Wand, and R. Carroll (2003). *Semiparametric Regression*. Cambridge: Cambridge University Press.

Sánchez, B., M. Wu, P. Song, and W. Wang (2016). Study design in high-dimensional classification analysis. *Biostatistics n/a*, n/a–n/a.

Scheipl, F., A.-M. Staicu, and S. Greven (2015). Functional additive mixed models. *Journal of Computational and Graphical Statistics 24*(2), 477–501.

Seber, G. (2007). *A Matrix Handbook for Statisticians*. New Jersey: Wiley-Interscience.

Shen, Q. and J. Faraway (2004). An f test for linear models with functional responses. *Statistica Sinica*, 1239–1257.

St-Pierre, N., B. Cobanov, and G. Schnitkey (2003). Economic losses from heat stress by us livestock industries. *Journal of dairy science 86*, E52–E77.

Staniswalis, J. and J. Lee (1998). Nonparametric regression analysis of longitudinal data. *J. Amer. Statist. Assoc. 93*, 1403–1418.

Tang, Q. and L. Cheng (2014). Partial functional linear quantile regression. *Science China Mathematics 57*(12), 2589–2608.

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

Wood, S. (2003). Thin plate regression splines. *J. R. Statist. Soc. B 65*, 95–114.

Wood, S. (2006). Low-rank scale-invariant tensor product smooths for generalized additive mixed models. *Biometrics 62*, 1025–1036.

Wood, S. (2013). R package *mgcv*: Mixed GAM computation vehicle with GCV/AIC/REML, smoothese estimation (version 1.7-24). URL:`http://cran.r-project.org/web/packages/mgcv/index.html`.

Wood, S. N. (2011). Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 73*(1), 3–36.

Wood, S. N. and M. Fasiolo (2017). A generalized fellner-schall method for smoothing parameter optimization with application to tweedie location, scale and shape models. *Biometrics*, n/a–n/a.

Wu, M. (2013). *Study Design for Longitudinal and High Dimensional Measures*. Ph. D. thesis, The University of Michigan.

Xiao, L., L. Huang, J. Schrack, L. Ferrucci, V. Zipunnikov, and C. Crainiceanu (2015). Quantifying the life-time circadian rhythm of physical activity: a covariate-dependent functional approach. *Biostatistics 16*, 352–367.

Xiao, L., C. Li, W. Checkley, and C. Crainiceanu (2017a). Fast covariance estimation for sparse functional data. *Statistics and Computing n/a*, 1–12.

Xiao, L., C. Li, W. Checkley, and C. Crainiceanu (2017b). R package *face*: Fast covariance estimation for sparse functional data (version 0.1-3). URL:`http://cran.r-project.org/web/packages/face/index.html`.

Xiao, L., Y. Li, and D. Ruppert (2013). Fast bivariate P-splines: the sandwich smoother. *J. R. Statist. Soc. B 75*, 577–599.

Xiao, L., D. Ruppert, V. Zipunnikov, and C. Crainiceanu (2016). Fast covariance function estimation for high-dimensional functional data. *Stat. Comput. 26*, 409–421.

Xu, G. and J. Huang (2012). Asymptotic optimality and efficient computation of the leave-subject-out cross-validation. *Ann. Statist. 40*, 3003–3030.

Yang, C., L. Wang, S. Zhang, and H. Zhao (2013). Accounting for non-genetic factors by low-rank representation and sparse regression for eqtl mapping. *Bioinformatics 29*(8), 1026–1034.

Yao, F., H. Müller, A. Clifford, S. Dueker, J. Follett, Y. Lin, B. Buchholz, and J. Vogel (2003). Shrinkage estimation for functional principal component scores with application to the population kinetics of plasma folate. *Biometrics 20*, 852–873.

Yao, F., H. Müller, and J. Wang (2005). Functional data analysis for sparse longitudinal data. *J. Amer. Statist. Assoc. 100*, 577–590.

Yu, D., L. Kong, and I. Mizera (2015). Partial functional linear quantile regression for neuroimaging data analysis. *arXiv preprint arXiv:1511.00632*.

Yuan, M., A. Ekici, Z. Lu, and R. Monteiro (2007). Dimension reduction and coefficient estimation in multivariate linear regression. *J. R. Stat. Soc. Ser. B Stat. Methodol. 69*(3), 329–346.

Zhang, J.-T., J. Chen, et al. (2007). Statistical inferences for functional data. *The Annals of Statistics 35*(3), 1052–1079.

Zhang, X. and J.-L. Wang (2016). From sparse to dense functional data and beyond. *Ann. Statist. 44*(5), 2281–2321.

Zhang, Y., H. Zhou, J. Zhou, and W. Sun (2015). Regression models for multivariate count data. *revision submitted*.

Zhou, H. and L. Li (2014). Regularized matrix regressions. *Journal of Royal Statistical Society, Series B 76(2)*, 463–483.

Zhou, X., J. Liu, X. Wan, and W. Yu (2014). Piecewise-constant and low-rank approximation for identification of recurrent copy number variations. *Bioinformatics 30*(14), 1943–1949.

Zhu, H., P. Brown, and J. Morris (2012). Robust classification of functional and quantitative image data using functional mixed models. *Biometrics 68*, 1260–1268.

APPENDICES

Appendix A

Supplement for Chapter 1

A.1 *P*-spline Mean Function Estimation

A.1.1 Estimation

Given the observed data $\{(y_{ij}, t_{ij}), j = 1, \ldots, m_i, i = 1, \ldots, n\}$, where t_{ij} is in the unit interval $[0, 1]$, n is the number of subjects, and m_i is the number of observations for subject i, we have $\mathbb{E}(y_{ij}) = f(t_{ij})$. We model the smooth mean function $f(t)$ by basis expansion $\sum_{1\leq\kappa\leq c}\theta_\kappa B_\kappa(t)$, where $\boldsymbol{\theta} = (\theta_\kappa)_{1\leq\kappa\leq c}$ is a coefficient vector, $\mathbf{B}(t) = \{B_1(t), \ldots, B_c(t)\}^T$ is the collection of B-spline basis functions evaluated at sampling point t, and c is the number of interior knots plus the order (degree plus 1) of the B-splines. We use 10 cubic basis functions, i.e., $c = 10$. Stack all the observations as a column vector $\mathbf{y} = [\mathbf{y}_1^T, \cdots, \mathbf{y}_n^T]^T$, where $\mathbf{y}_i = (y_1, \cdots, y_{m_i})^T$. Define $\mathbf{B}_i$ as the basis matrix $\mathbf{B}_i = [\mathbf{B}(t_{i1}), \ldots, \mathbf{B}(t_{im_i})]^T$ and $\mathbf{B} = [\mathbf{B}_1^T, \ldots, \mathbf{B}_n^T]^T$. Let $\mathbf{D}$ be the second-order differencing matrix, and let λ denote smoothing parameter. The smoother matrix can be constructed by using *P*-spline (Eilers and Marx, 1996). Moreover, we can add some pre-specified weights for each subject. In this paper, we use $\omega_i = 1/m_i$. Then $\mathbf{W}$ is a diagonal matrix with each ω_i repeating m_i times along the diagonal. Therefore, we estimate $\boldsymbol{\theta}$ by minimizing penalized

weighted least squares

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} (\mathbf{y} - \mathbf{B}\boldsymbol{\theta})^T \mathbf{W} (\mathbf{y} - \mathbf{B}\boldsymbol{\theta}) + \lambda \|\mathbf{D}\boldsymbol{\theta}\|_2^2. \tag{A.1}$$

The fitted mean function is given by $\hat{f}(t) = \mathbf{B}(t)^T (\mathbf{B}^T\mathbf{W}\mathbf{B} + \lambda \mathbf{D}^T\mathbf{D})^{-1}\mathbf{B}^T\mathbf{W}\mathbf{y}$. For simplicity, we write $\widetilde{\mathbf{S}}(\lambda) = (\mathbf{B}^T\mathbf{W}\mathbf{B} + \lambda\mathbf{D}^T\mathbf{D})^{-1}\mathbf{B}^T\mathbf{W}$ and suppress λ. It leads to $\hat{f}(t) = \mathbf{B}(t)^T\widetilde{\mathbf{S}}\mathbf{y}$. Next, we have $\mathrm{Var}(\hat{f}(t)) = \mathbf{B}(t)^T\widetilde{\mathbf{S}}\boldsymbol{\Lambda}\widetilde{\mathbf{S}}^T\mathbf{B}(t)$, where $\mathrm{Cov}(\mathbf{y}) = \boldsymbol{\Lambda}$ is a block diagonal matrix with subject covariance as each block. Specifically, for curve $\mathbf{y}_i$, $\mathrm{Cov}(\mathbf{y}_i) = \boldsymbol{\Lambda}_i + \sigma_\epsilon^2 \mathbf{I}_{m_i}$ is the covariance of underlying trajectory contaminated by white noise with error variance σ_ϵ^2, where $\boldsymbol{\Lambda}_i$ is the covariance of the ith true trajectory. The confidence band for mean function $f(t)$ can be constructed accordingly. Indeed, a 95% pointwise confidence interval for $f(t)$ is

$$\hat{f}(t) \pm 1.96\sqrt{\mathbf{B}(t)^T\widetilde{\mathbf{S}}\widehat{\boldsymbol{\Lambda}}\widetilde{\mathbf{S}^T}\mathbf{B}(t)}.$$

A.1.2 Selection of Smoothing Parameter

We use leave-one-subject-out cross validation to select smoothing parameter for mean function. A fast algorithm is derived for approximating the leave-one-subject-out cross validation.

Let $\widetilde{\mathbf{y}}_i^{[i]}$ be the prediction of $\mathbf{y}_i$ by applying the proposed method to the data without the data from the ith subject, then the cross-validated error is

$$\mathrm{iCV} = \sum_{i=1}^{n} \|\widetilde{\mathbf{y}}_i^{[i]} - \mathbf{y}_i\|^2. \tag{A.2}$$

Now we introduce a shortcut formula for iCV. First we let $\mathbf{S} = \mathbf{B}(\mathbf{B}^T\mathbf{W}\mathbf{B} + \lambda\mathbf{D}^T\mathbf{D})^{-1}\mathbf{B}^T\mathbf{W}$, which is the smoother matrix for the proposed method. $\mathbf{S}$ can be written as $(\mathbf{B}\mathbf{A})[\mathbf{I} + \lambda\mathrm{diag}(\mathbf{s})]^{-1}(\mathbf{B}\mathbf{A})^T\mathbf{W}$ for some square matrix $\mathbf{A}$ and $\mathbf{s}$ is a column vector. Furthermore, both $\mathbf{A}$ and $\mathbf{s}$ do not depend on λ.

Let $\mathbf{S}_i = \mathbf{B}_i(\mathbf{B}^T\mathbf{W}\mathbf{B} + \lambda\mathbf{D}^T\mathbf{D})^{-1}\mathbf{B}^T\mathbf{W}$ and $\mathbf{S}_{ii} = \mathbf{B}_i(\mathbf{B}^T\mathbf{W}\mathbf{B} + \lambda\mathbf{D}^T\mathbf{D})^{-1}\mathbf{B}_i^T w_i$. Then $\mathbf{S}_i$ is of dimension $m_i \times N$, where $N = \sum_{i=1}^n m_i$, and $\mathbf{S}_{ii}$ is symmetric and of dimension $m_i \times m_i$.

Lemma 3. *The* iCV *in* (A.2) *can be simplified as*

$$\text{iCV} = \sum_{i=1}^{n} \|(\mathbf{I}_{m_i} - \mathbf{S}_{ii})^{-1}(\mathbf{S}_i\mathbf{y} - \mathbf{y}_i)\|^2.$$

The proof of Lemma S. 3 is the same as that of Lemma 3.1 in Xu and Huang (2012) and thus is omitted. We further simplify iCV by using the approximation $(\mathbf{I}_{m_i} - \mathbf{S}_{ii}^T)^{-1}(\mathbf{I}_{m_i} - \mathbf{S}_{ii})^{-1} = \mathbf{I}_{m_i} + \mathbf{S}_{ii} + \mathbf{S}_{ii}^T$. This approximation leads to the generalized cross validation iGCV,

$$\text{iGCV} = \sum_{i=1}^{n}(\mathbf{S}_i\mathbf{y}-\mathbf{y}_i)^T(\mathbf{I}_{m_i}+\mathbf{S}_{ii}+\mathbf{S}_{ii}^T)(\mathbf{S}_i\mathbf{y}-\mathbf{y}_i) = \|\mathbf{y}-\mathbf{S}\mathbf{y}\|^2+2\sum_{i=1}^{n}(\mathbf{S}_i\mathbf{y}-\mathbf{y}_i)^T\mathbf{S}_{ii}(\mathbf{S}_i\mathbf{y}-\mathbf{y}_i). \quad \text{(A.3)}$$

We further simplify iGCV. Let $\mathbf{F}_i = \mathbf{B}_i\mathbf{A}$, $\mathbf{F} = \mathbf{B}\mathbf{A}$ and $\widetilde{\mathbf{F}} = \mathbf{F}\mathbf{W}$. Define $\mathbf{f}_i = \mathbf{F}_i^T\mathbf{y}_i$, $\mathbf{f} = \mathbf{F}^T\mathbf{y}$ and $\widetilde{\mathbf{f}} = \widetilde{\mathbf{F}}^T\mathbf{y}$. To simplify notation we will denote $[\mathbf{I} + \lambda\text{diag}(\mathbf{s})]^{-1}$ as $\widetilde{\mathbf{D}}$, a symmetric matrix, and its diagonal as $\widetilde{\mathbf{d}}$. Let $\odot$ be the Hadamard product such that for two matrices of the same dimensions $A = (a_{ij})$ and $B = (b_{ij})$, $A \odot B = (a_{ij}b_{ij})$.

Proposition 3. *The* iGCV *in* (A.3) *can be simplified as*

$$\text{iGCV} = \|\mathbf{y}\|^2-2\widetilde{\mathbf{d}}^T(\widetilde{\mathbf{f}}\odot\mathbf{f})+(\widetilde{\mathbf{f}}\odot\widetilde{\mathbf{d}})^T(\mathbf{F}^T\mathbf{F})(\widetilde{\mathbf{f}}\odot\widetilde{\mathbf{d}})+2\widetilde{\mathbf{d}}^T\mathbf{g}-4\widetilde{\mathbf{d}}^T\mathbf{G}_1\widetilde{\mathbf{d}}+2\widetilde{\mathbf{d}}^T\mathbf{G}_2\left\{(\widetilde{\mathbf{f}}\odot\widetilde{\mathbf{d}})\otimes(\widetilde{\mathbf{f}}\odot\widetilde{\mathbf{d}})\right\},$$

where $\mathbf{g} = \sum_{i=1}^n w_i\mathbf{f}_i\odot\mathbf{f}_i$, $\mathbf{G}_1 = \sum_{i=1}^n w_i(\mathbf{f}_i\widetilde{\mathbf{f}}^T)\odot(\mathbf{F}_i^T\mathbf{F}_i)$, *and* $\mathbf{G}_2 = \sum_{i=1}^n w_i(\mathbf{F}_i^T\mathbf{F}_i)\circ(\mathbf{F}_i^T\mathbf{F}_i)$. *Here* $\circ$ *is the row-wise Khatri-Rao product such that for two matrices with the same number of rows* $A = [\mathbf{a}_1,\ldots,\mathbf{a}_m]^T$ *and* $B = [\mathbf{b}_1,\ldots,\mathbf{b}_m]^T$, $A\circ B = [\mathbf{a}_1\otimes\mathbf{b}_1,\ldots,\mathbf{a}_m\otimes\mathbf{b}_m]^T$.

Remark 1. *While the above formula looks complex, it can be efficiently computed. Indeed, only the term* $\widetilde{\mathbf{d}}$ *depend on the smoothing parameter* λ *and it can be easily computed; all other terms including* $\mathbf{g}$, $\mathbf{G}_1$, $\mathbf{G}_2$ *can be pre-calculated just for once.*

A.1.3 Proof of Proposition 3

Proof. We will use the following Lemma (page 241, Seber 2007).

Lemma 4. *Let* $\mathbf{A}$, $\mathbf{B}$ *and* $\mathbf{C}$ *and* $\mathbf{D}$ *be compatible matrices. Then*

$$\text{tr}(\mathbf{ABCD}) = (\text{vec}\mathbf{D})^T(\mathbf{A}\otimes\mathbf{C}^T)\text{vec}\mathbf{B}^T.$$

We write iGCV as a sum

$$\text{iGCV} = \mathcal{I} + 2\sum_{i=1}^{n}(\mathcal{II}_i + 2\mathcal{III}_i + \mathcal{IV}_i), \tag{A.4}$$

where $\mathcal{I} = \|\mathbf{y} - \mathbf{Sy}\|^2$, $\mathcal{II}_i = \mathbf{y}_i^T\mathbf{S}_{ii}\mathbf{y}_i$, $\mathcal{III}_i = (\mathbf{S}_i\mathbf{y})^T\mathbf{S}_{ii}\mathbf{y}_i$ and $\mathcal{IV}_i = (\mathbf{S}_i\mathbf{y})^T\mathbf{S}_{ii}(\mathbf{S}_i\mathbf{y})$. Note that we have the following equalities that will be used later:

$$\mathbf{S} = (\mathbf{BA})[\mathbf{I} + \lambda\text{diag}(\mathbf{s})]^{-1}(\mathbf{BA})^T\mathbf{W} = \mathbf{F}\widetilde{\mathbf{D}}\widetilde{\mathbf{F}}^T,$$

$$\mathbf{S}_i = (\mathbf{B}_i\mathbf{A})[\mathbf{I} + \lambda\text{diag}(\mathbf{s})]^{-1}(\mathbf{BA})^T\mathbf{W} = \mathbf{F}_i\widetilde{\mathbf{D}}\widetilde{\mathbf{F}}^T,$$

$$\mathbf{S}_{ii} = (\mathbf{B}_i\mathbf{A})[\mathbf{I} + \lambda\text{diag}(\mathbf{s})]^{-1}(\mathbf{B}_i\mathbf{A})^T w_i = w_i\mathbf{F}_i\widetilde{\mathbf{D}}\mathbf{F}_i^T.$$

We first compute $\mathcal{I}$. We have

$$\mathcal{I} = \|\mathbf{y} - \mathbf{Sy}\|^2 = \|\mathbf{y} - \mathbf{F}\widetilde{\mathbf{D}}\widetilde{\mathbf{f}}\|^2 = \|\mathbf{y}\|^2 - 2\mathbf{f}^T\widetilde{\mathbf{D}}\widetilde{\mathbf{f}} + \widetilde{\mathbf{f}}^T\widetilde{\mathbf{D}}\mathbf{F}^T\mathbf{F}\widetilde{\mathbf{D}}\widetilde{\mathbf{f}}.$$

Thus,

$$\mathcal{I} = \|\mathbf{y}\|^2 - 2\widetilde{\mathbf{d}}^T(\widetilde{\mathbf{f}}\odot\mathbf{f}) + (\widetilde{\mathbf{f}}\odot\widetilde{\mathbf{d}})^T(\mathbf{F}^T\mathbf{F})(\widetilde{\mathbf{f}}\odot\widetilde{\mathbf{d}}). \tag{A.5}$$

Second, we compute $\mathcal{II}_i$. We have

$$\mathcal{II}_i = \mathbf{y}_i^T\mathbf{S}_{ii}\mathbf{y}_i = w_i\mathbf{f}_i^T\widetilde{\mathbf{D}}\mathbf{f}_i = w_i\widetilde{\mathbf{d}}^T(\mathbf{f}_i\odot\mathbf{f}_i). \tag{A.6}$$

Third, we compute $\mathcal{III}_i$. Note that $\mathbf{S}_i\mathbf{y} = \mathbf{F}_i\widetilde{\mathbf{D}}\widetilde{\mathbf{f}}$ and hence

$$\mathcal{III}_i = (\mathbf{S}_i\mathbf{y})^T\mathbf{S}_{ii}\mathbf{y}_i = w_i\widetilde{\mathbf{f}}^T\widetilde{\mathbf{D}}\mathbf{F}_i^T\mathbf{F}_i\widetilde{\mathbf{D}}\mathbf{f}_i = w_i\text{tr}(\mathbf{f}_i\widetilde{\mathbf{f}}^T\widetilde{\mathbf{D}}\mathbf{F}_i^T\mathbf{F}_i\widetilde{\mathbf{D}}) = w_i\widetilde{\mathbf{d}}^T\left\{(\mathbf{f}_i\widetilde{\mathbf{f}}^T)\odot(\mathbf{F}_i^T\mathbf{F}_i)\right\}\widetilde{\mathbf{d}}.$$

Thus we have

$$\mathcal{III}_i = w_i \widetilde{\mathbf{d}}^T \left\{ (\mathbf{f}_i \widetilde{\mathbf{f}}^T) \odot (\mathbf{F}_i^T \mathbf{F}_i) \right\} \widetilde{\mathbf{d}}. \tag{A.7}$$

Fourth, we compute $\mathcal{IV}_i$. We derive that

$$\begin{aligned}
\mathcal{IV}_i &= (\mathbf{S}_i \mathbf{y})^T \mathbf{S}_{ii} (\mathbf{S}_i \mathbf{y}) \\
&= w_i \widetilde{\mathbf{f}}^T \widetilde{\mathbf{D}} \mathbf{F}_i^T \mathbf{F}_i \widetilde{\mathbf{D}} \mathbf{F}_i^T \mathbf{F}_i \widetilde{\mathbf{D}} \widetilde{\mathbf{f}} \\
&= w_i \mathrm{tr}(\widetilde{\mathbf{f}}^T \widetilde{\mathbf{D}} \mathbf{F}_i^T \mathbf{F}_i \widetilde{\mathbf{D}} \mathbf{F}_i^T \mathbf{F}_i \widetilde{\mathbf{D}} \widetilde{\mathbf{f}}) \\
&= w_i \mathrm{tr}(\mathbf{F}_i^T \mathbf{F}_i \widetilde{\mathbf{D}} \mathbf{F}_i^T \mathbf{F}_i \widetilde{\mathbf{D}} \widetilde{\mathbf{f}} \widetilde{\mathbf{f}}^T \widetilde{\mathbf{D}}) \\
&= w_i \mathrm{tr} \left\{ \widetilde{\mathbf{D}} (\mathbf{F}_i^T \mathbf{F}_i)(\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}})(\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}})^T (\mathbf{F}_i^T \mathbf{F}_i) \right\} \\
&= w_i \widetilde{\mathbf{d}}^T \left[\left\{ (\mathbf{F}_i^T \mathbf{F}_i)(\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}}) \right\} \odot \left\{ (\mathbf{F}_i^T \mathbf{F}_i)(\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}}) \right\} \right].
\end{aligned}$$

Hence we obtain

$$\mathcal{IV}_i = w_i \widetilde{\mathbf{d}}^T \left\{ (\mathbf{F}_i^T \mathbf{F}_i) \circ (\mathbf{F}_i^T \mathbf{F}_i) \right\} \left\{ (\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}}) \otimes (\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}}) \right\}. \tag{A.8}$$

Now with (A.4), (A.5), (A.6), (A.7) and (A.8), we obtain that

$$\begin{aligned}
\mathrm{iGCV} &= \|\mathbf{y}\|^2 - 2\widetilde{\mathbf{d}}^T (\widetilde{\mathbf{f}} \odot \mathbf{f}) + (\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}})^T (\mathbf{F}^T \mathbf{F})(\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}}) + 2 \sum_{i=1}^n w_i \widetilde{\mathbf{d}}^T (\mathbf{f}_i \odot \mathbf{f}_i) \\
&\quad + 2 \sum_{i=1}^n \left[2 w_i \widetilde{\mathbf{d}}^T \left\{ (\mathbf{f}_i \widetilde{\mathbf{f}}^T) \odot (\mathbf{F}_i^T \mathbf{F}_i) \right\} \widetilde{\mathbf{d}} + w_i \widetilde{\mathbf{d}}^T \left\{ (\mathbf{F}_i^T \mathbf{F}_i) \circ (\mathbf{F}_i^T \mathbf{F}_i) \right\} \left\{ (\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}}) \otimes (\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}}) \right\} \right] \\
&= \|\mathbf{y}\|^2 - 2\widetilde{\mathbf{d}}^T (\widetilde{\mathbf{f}} \odot \mathbf{f}) + (\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}})^T (\mathbf{F}^T \mathbf{F})(\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}}) + 2\widetilde{\mathbf{d}}^T \left\{ \sum_{i=1}^n w_i \mathbf{f}_i \odot \mathbf{f}_i \right\} \\
&\quad - 4\widetilde{\mathbf{d}}^T \left\{ \sum_{i=1}^n w_i (\mathbf{f}_i \widetilde{\mathbf{f}}^T) \odot (\mathbf{F}_i^T \mathbf{F}_i) \right\} \widetilde{\mathbf{d}} + 2\widetilde{\mathbf{d}}^T \left\{ \sum_{i=1}^n w_i (\mathbf{F}_i^T \mathbf{F}_i) \circ (\mathbf{F}_i^T \mathbf{F}_i) \right\} \left\{ (\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}}) \otimes (\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}}) \right\},
\end{aligned}$$

which completes the proof. □

A.2 Proofs of Proposition 1 and Proposition 2

Proof of Proposition 1: We will use the following Lemma (Isserlis, 1918) in our proof.

Lemma 5. *If* $(x_1, \cdots, x_{2n})$ *is a zero mean multivariate normal random vector, then*

$$\mathbb{E}(x_1, \cdots, x_{2n}) = \sum \prod \mathbb{E}(x_i x_j),$$

where $x_i x_j$ *are all distinct pairs over* $x_1, x_2, \cdots, x_{2n}$.

First, we have

$$\begin{aligned} \mathrm{Cov}(C_{ijj'}, C_{ikk'}) &= \mathbb{E}(C_{ijj'} C_{ikk'}) - \mathbb{E}(C_{ijj'})\mathbb{E}(C_{ikk'}), \\ \mathbb{E}(C_{ijj'}) &= \mathcal{C}(t_{ij}, t_{ij'}) + \delta_{jj'}\sigma_\epsilon^2, \\ \mathbb{E}(C_{ikk'}) &= \mathcal{C}(t_{ik}, t_{ik'}) + \delta_{kk'}\sigma_\epsilon^2. \end{aligned}$$

Then,

$$\mathbb{E}(C_{ijj'})\mathbb{E}(C_{ikk'}) = \mathcal{C}(t_{ij}, t_{ij'})\mathcal{C}(t_{ik}, t_{ik'}) + \mathcal{C}(t_{ij}, t_{ij'})\delta_{kk'}\sigma_\epsilon^2 + \mathcal{C}(t_{ik}, t_{ik'})\delta_{jj'}\sigma_\epsilon^2 + \delta_{jj'}\sigma_\epsilon^2\delta_{kk'}\sigma_\epsilon^2.$$

Next, we derive $\mathbb{E}(C_{ijj'}C_{ikk'})$ that

$$\begin{aligned}
\mathbb{E}(C_{ijj'}C_{ikk'}) &= \mathbb{E}(y_{ij}y_{ij'}y_{ik}y_{ik'}) \\
&= \mathbb{E}\{(x_i(t_{ij})+\epsilon_{ij})(x_i(t_{ij'})+\epsilon_{ij'})(x_i(t_{ik})+\epsilon_{ik})(x_i(t_{ik'})+\epsilon_{ik'})\} \\
&= \mathbb{E}\{x_i(t_{ij})x_i(t_{ij'})x_i(t_{ik})x_i(t_{ik'})\} + \mathbb{E}\{x_i(t_{ij})x_i(t_{ij'})x_i(t_{ik})\}\mathbb{E}(\epsilon_{ik'}) \\
&+ \mathbb{E}\{x_i(t_{ij})x_i(t_{ij'})x_i(t_{ik'})\}\mathbb{E}(\epsilon_{ik}) + \mathbb{E}\{x_i(t_{ij})x_i(t_{ij'})\}\mathbb{E}(\epsilon_{ik}\epsilon_{ik'}) \\
&+ \mathbb{E}\{x_i(t_{ij})x_i(t_{ik})x_i(t_{ik'})\}\mathbb{E}(\epsilon_{ij'}) + \mathbb{E}\{x_i(t_{ij})x_i(t_{ik})\}\mathbb{E}(\epsilon_{ij'}\epsilon_{ik'}) \\
&+ \mathbb{E}\{x_i(t_{ij})x_i(t_{ik'})\}\mathbb{E}(\epsilon_{ij'}\epsilon_{ik}) + \mathbb{E}\{x_i(t_{ij})\}\mathbb{E}(\epsilon_{ij'}\epsilon_{ik}\epsilon_{ik'}) \\
&+ \mathbb{E}\{x_i(t_{ij'})x_i(t_{ik})x_i(t_{ik'})\}\mathbb{E}(\epsilon_{ij}) + \mathbb{E}\{x_i(t_{ij'})x_i(t_{ik})\}\mathbb{E}(\epsilon_{ij}\epsilon_{ik'}) \\
&+ \mathbb{E}\{x_i(t_{ij'})x_i(t_{ik'})\}\mathbb{E}(\epsilon_{ij}\epsilon_{ik}) + \mathbb{E}\{x_i(t_{ij'})\}\mathbb{E}(\epsilon_{ij}\epsilon_{ik}\epsilon_{ik'}) \\
&+ \mathbb{E}\{x_i(t_{ik})x_i(t_{ik'})\}\mathbb{E}(\epsilon_{ij}\epsilon_{ij'}) + \mathbb{E}\{x_i(t_{ik})\}\mathbb{E}(\epsilon_{ij}\epsilon_{ij'}\epsilon_{ik'}) \\
&+ \mathbb{E}\{x_i(t_{ik'})\}\mathbb{E}(\epsilon_{ij}\epsilon_{ij'}\epsilon_{ik}) + \mathbb{E}(\epsilon_{ij}\epsilon_{ij'}\epsilon_{ik}\epsilon_{ik'}) \\
&= \mathbb{E}\{x_i(t_{ij})x_i(t_{ij'})x_i(t_{ik})x_i(t_{ik'})\} + \mathbb{E}(\epsilon_{ij}\epsilon_{ij'}\epsilon_{ik}\epsilon_{ik'}) \\
&+ \mathcal{C}(t_{ij},t_{ij'})\delta_{kk'}\sigma_\epsilon^2 + \mathcal{C}(t_{ij},t_{ik})\delta_{j'k'}\sigma_\epsilon^2 + \mathcal{C}(t_{ij},t_{ik'})\delta_{j'k}\sigma_\epsilon^2 \\
&+ \mathcal{C}(t_{ij'},t_{ik})\delta_{jk'}\sigma_\epsilon^2 + \mathcal{C}(t_{ij'},t_{ik'})\delta_{jk}\sigma_\epsilon^2 + \mathcal{C}(t_{ik},t_{ik'})\delta_{jj'}\sigma_\epsilon^2.
\end{aligned}$$

Finally, with $\mathbb{E}(C_{ijj'})\mathbb{E}(C_{ikk'})$ and $\mathbb{E}(C_{ijj'}C_{ikk'})$, we have

$$\begin{aligned}
\mathrm{Cov}(C_{ijj'},C_{ikk'}) &= \mathcal{C}(t_{ij},t_{ij'})\mathcal{C}(t_{ik},t_{ik'}) + \mathcal{C}(t_{ij},t_{ik})\mathcal{C}(t_{ij'},t_{ik'}) + \mathcal{C}(t_{ij},t_{ik'})\mathcal{C}(t_{ij'},t_{ik}) \\
&+ \delta_{jj'}\sigma_\epsilon^2\delta_{kk'}\sigma_\epsilon^2 + \delta_{jk}\sigma_\epsilon^2\delta_{j'k'}\sigma_\epsilon^2 + \delta_{jk'}\sigma_\epsilon^2\delta_{j'k}\sigma_\epsilon^2 \\
&+ \mathcal{C}(t_{ij},t_{ij'})\delta_{kk'}\sigma_\epsilon^2 + \mathcal{C}(t_{ij},t_{ik})\delta_{j'k'}\sigma_\epsilon^2 + \mathcal{C}(t_{ij},t_{ik'})\delta_{j'k}\sigma_\epsilon^2 \\
&+ \mathcal{C}(t_{ij'},t_{ik})\delta_{jk'}\sigma_\epsilon^2 + \mathcal{C}(t_{ij'},t_{ik'})\delta_{jk}\sigma_\epsilon^2 + \mathcal{C}(t_{ik},t_{ik'})\delta_{jj'}\sigma_\epsilon^2 \\
&- \mathcal{C}(t_{ij},t_{ij'})\mathcal{C}(t_{ik},t_{ik'}) - \mathcal{C}(t_{ij},t_{ij'})\delta_{kk'}\sigma_\epsilon^2 - \mathcal{C}(t_{ik},t_{ik'})\delta_{jj'}\sigma_\epsilon^2 - \delta_{jj'}\sigma_\epsilon^2\delta_{kk'}\sigma_\epsilon^2 \\
&= \mathcal{C}(t_{ij},t_{ik})\mathcal{C}(t_{ij'},t_{ik'}) + \mathcal{C}(t_{ij},t_{ik'})\mathcal{C}(t_{ij'},t_{ik}) + \delta_{jk}\delta_{j'k'}\sigma_\epsilon^4 + \delta_{jk'}\delta_{j'k}\sigma_\epsilon^4 \\
&+ \mathcal{C}(t_{ij},t_{ik})\delta_{j'k'}\sigma_\epsilon^2 + \mathcal{C}(t_{ij},t_{ik'})\delta_{j'k}\sigma_\epsilon^2 + \mathcal{C}(t_{ij'},t_{ik})\delta_{jk'}\sigma_\epsilon^2 + \mathcal{C}(t_{ij'},t_{ik'})\delta_{jk}\sigma_\epsilon^2.
\end{aligned}$$

By the definition $\mathbf{M}_{ijk} = \left\{\mathcal{C}(t_{ij}, t_{ijk}), \delta_{jk}\sigma_\epsilon^2\right\}^T$, we derive that

$$\begin{aligned}
\mathrm{Cov}(C_{ijj'}, C_{ikk'}) &= \left\{\mathcal{C}(t_{ij}, t_{ik})\mathcal{C}(t_{ij'}, t_{ik'}) + \mathcal{C}(t_{ij}, t_{ik})\delta_{j'k'}\sigma_\epsilon^2 + \mathcal{C}(t_{ij'}, t_{ik'})\delta_{jk}\sigma_\epsilon^2 + \delta_{jk}\delta_{j'k'}\sigma_\epsilon^4\right\} \\
&+ \left\{\mathcal{C}(t_{ij}, t_{ik'})\mathcal{C}(t_{ij'}, t_{ik}) + \mathcal{C}(t_{ij}, t_{ik'})\delta_{j'k}\sigma_\epsilon^2 + \mathcal{C}(t_{ij'}, t_{ik})\delta_{jk'}\sigma_\epsilon^2 + \delta_{jk'}\delta_{j'k}\sigma_\epsilon^4\right\} \\
&= \mathbf{1}^T(\mathbf{M}_{ij_1j_3} \otimes \mathbf{M}_{ij_2j_4} + \mathbf{M}_{ij_1j_4} \otimes \mathbf{M}_{ij_2j_3}),
\end{aligned}$$

which proves the proposition.

Proof of Proposition 2: We first write iGCV as a sum

$$\mathrm{iGCV} = \mathcal{I} + 2\sum_{i=1}^{n}(\mathcal{II}_i - 2\mathcal{III}_i + \mathcal{IV}_i), \tag{A.9}$$

where $\mathcal{I} = \|\widehat{\mathbf{C}} - \mathbf{S}\widehat{\mathbf{C}}\|^2$, $\mathcal{II}_i = \widehat{\mathbf{C}}_i^T\mathbf{S}_{ii}\widehat{\mathbf{C}}_i$, $\mathcal{III}_i = (\mathbf{S}_i\widehat{\mathbf{C}})^T\mathbf{S}_{ii}\widehat{\mathbf{C}}_i$ and $\mathcal{IV}_i = (\mathbf{S}_i\widehat{\mathbf{C}})^T\mathbf{S}_{ii}(\mathbf{S}_i\widehat{\mathbf{C}})$.

Note that we have the following equalities that will be used later:

$$\mathbf{S} = (\mathbf{XA})[\mathbf{I} + \lambda\mathrm{diag}(\mathbf{s})]^{-1}(\mathbf{XA})^T\mathbf{W} = \mathbf{F}\widetilde{\mathbf{D}}\widetilde{\mathbf{F}},$$

$$\mathbf{S}_i = (\mathbf{X}_i\mathbf{A})[\mathbf{I} + \lambda\mathrm{diag}(\mathbf{s})]^{-1}(\mathbf{XA})^T\mathbf{W} = \mathbf{F}_i\widetilde{\mathbf{D}}\widetilde{\mathbf{F}},$$

$$\mathbf{S}_{ii} = (\mathbf{X}_i\mathbf{A})[\mathbf{I} + \lambda\mathrm{diag}(\mathbf{s})]^{-1}(\mathbf{X}_i\mathbf{A})^T\mathbf{W}_i = \mathbf{F}_i\widetilde{\mathbf{D}}\mathbf{F}_i^T\mathbf{W}_i.$$

We first compute $\mathcal{I}$. We have

$$\mathcal{I} = \|\widehat{\mathbf{C}} - \mathbf{S}\widehat{\mathbf{C}}\|^2 = \|\widehat{\mathbf{C}} - \mathbf{F}\widetilde{\mathbf{D}}\widetilde{\mathbf{f}}\|^2 = \|\widehat{\mathbf{C}}\|^2 - 2\mathbf{f}^T\widetilde{\mathbf{D}}\widetilde{\mathbf{f}} + \widetilde{\mathbf{f}}^T\widetilde{\mathbf{D}}\mathbf{F}^T\mathbf{F}\widetilde{\mathbf{D}}\widetilde{\mathbf{f}}.$$

Thus,

$$\mathcal{I} = \|\widehat{\mathbf{C}}\|^2 - 2\widetilde{\mathbf{d}}^T(\widetilde{\mathbf{f}} \odot \mathbf{f}) + (\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}})^T(\mathbf{F}^T\mathbf{F})(\widetilde{\mathbf{f}} \odot \widetilde{\mathbf{d}}). \tag{A.10}$$

Second, we compute $\mathcal{II}_i$. We have

$$\mathcal{II}_i = \widehat{\mathbf{C}}_i^T\mathbf{S}_{ii}\widehat{\mathbf{C}}_i = \mathbf{f}_i^T\widetilde{\mathbf{D}}\mathbf{F}_i^T\mathbf{W}_i\widehat{\mathbf{C}}_i = \mathbf{f}_i^T\widetilde{\mathbf{D}}\mathbf{J}_i = \widetilde{\mathbf{d}}^T(\mathbf{J}_i \odot \mathbf{f}_i). \tag{A.11}$$

Third, we compute $\mathcal{III}_i$. Note that $\mathbf{S}_i\widehat{\mathbf{C}} = \mathbf{F}_i\widetilde{\mathbf{D}}\widetilde{\mathbf{f}}$ and hence

$$\mathcal{III}_i = (\mathbf{S}_i\widehat{\mathbf{C}})^T\mathbf{S}_{ii}\widehat{\mathbf{C}}_i = \widetilde{\mathbf{f}}^T\widetilde{\mathbf{D}}\mathbf{F}_i^T\mathbf{F}_i\widetilde{\mathbf{D}}\mathbf{J}_i = \mathrm{tr}(\mathbf{J}_i\widetilde{\mathbf{f}}^T\widetilde{\mathbf{D}}\mathbf{F}_i^T\mathbf{F}_i\widetilde{\mathbf{D}}) = \widetilde{\mathbf{d}}^T\{(\mathbf{J}_i\widetilde{\mathbf{f}}^T)\odot(\mathbf{F}_i^T\mathbf{F}_i)\}\widetilde{\mathbf{d}}.$$

Thus we have

$$\mathcal{III}_i = \widetilde{\mathbf{d}}^T\{(\mathbf{J}_i\widetilde{\mathbf{f}}^T)\odot(\mathbf{F}_i^T\mathbf{F}_i)\}\widetilde{\mathbf{d}}. \tag{A.12}$$

Fourth, we compute $\mathcal{IV}_i$. We derive that

$$\begin{aligned}\mathcal{IV}_i &= (\mathbf{S}_i\widehat{\mathbf{C}})^T\mathbf{S}_{ii}(\mathbf{S}_i\widehat{\mathbf{C}})\\ &= \widetilde{\mathbf{f}}^T\widetilde{\mathbf{D}}\mathbf{F}_i^T\mathbf{F}_i\widetilde{\mathbf{D}}\mathbf{F}_i^T\mathbf{W}_i\mathbf{F}_i\widetilde{\mathbf{D}}\widetilde{\mathbf{f}}\\ &= (\widetilde{\mathbf{f}}\odot\widetilde{\mathbf{d}})^T(\mathbf{F}_i^T\mathbf{F}_i)\widetilde{\mathbf{D}}(\mathbf{F}_i^T\mathbf{W}_i\mathbf{F}_i)(\widetilde{\mathbf{f}}\odot\widetilde{\mathbf{d}}).\end{aligned}$$

Hence we obtain

$$\mathcal{IV}_i = \widetilde{\mathbf{d}}^T\left[\left\{(\mathbf{F}_i^T\mathbf{F}_i)(\widetilde{\mathbf{f}}\odot\widetilde{\mathbf{d}})\right\}\odot\left\{(\mathbf{F}_i^T\mathbf{W}_i\mathbf{F}_i)(\widetilde{\mathbf{f}}\odot\widetilde{\mathbf{d}})\right\}\right]. \tag{A.13}$$

Now with (A.9), (A.10), (A.11), (A.12) and (A.13), we obtain that

$$\begin{aligned}\mathrm{iGCV} &= \|\widehat{\mathbf{C}}\|^2 - 2\widetilde{\mathbf{d}}^T(\widetilde{\mathbf{f}}\odot\mathbf{f}) + (\widetilde{\mathbf{f}}\odot\widetilde{\mathbf{d}})^T(\mathbf{F}^T\mathbf{F})(\widetilde{\mathbf{f}}\odot\widetilde{\mathbf{d}}) + 2\sum_{i=1}^n\widetilde{\mathbf{d}}^T(\mathbf{J}_i\odot\mathbf{f}_i)\\ &\quad + 2\sum_{i=1}^n\left[-2\widetilde{\mathbf{d}}^T\left\{(\mathbf{J}_i\widetilde{\mathbf{f}}^T)\circ(\mathbf{F}_i^T\mathbf{F}_i)\right\}\widetilde{\mathbf{d}} + \widetilde{\mathbf{d}}^T\left\{\mathbf{L}_i\odot(\mathbf{F}_i^T\mathbf{F}_i)\right\}\left\{(\widetilde{\mathbf{f}}\odot\widetilde{\mathbf{d}})\otimes(\widetilde{\mathbf{f}}\odot\widetilde{\mathbf{d}})\right\}\right]\\ &= \|\widehat{\mathbf{C}}\|^2 - 2\widetilde{\mathbf{d}}^T(\widetilde{\mathbf{f}}\odot\mathbf{f}) + (\widetilde{\mathbf{f}}\odot\widetilde{\mathbf{d}})^T(\mathbf{F}^T\mathbf{F})(\widetilde{\mathbf{f}}\odot\widetilde{\mathbf{d}}) + 2\widetilde{\mathbf{d}}^T\left\{\sum_{i=1}^n\mathbf{J}_i\odot\mathbf{f}_i\right\}\\ &\quad - 4\widetilde{\mathbf{d}}^T\left\{\sum_{i=1}^n(\mathbf{J}_i\widetilde{\mathbf{f}}^T)\circ(\mathbf{F}_i^T\mathbf{F}_i)\right\}\widetilde{\mathbf{d}} + 2\widetilde{\mathbf{d}}^T\left[\sum_{i=1}^n\left\{(\mathbf{F}_i^T\mathbf{F}_i)(\widetilde{\mathbf{f}}\odot\widetilde{\mathbf{d}})\right\}\odot\left\{(\mathbf{F}_i^T\mathbf{W}_i\mathbf{F}_i)(\widetilde{\mathbf{f}}\odot\widetilde{\mathbf{d}})\right\}\right].\end{aligned}$$

which proves the proposition.

A.3 Additional Simulation Results

In this subsection, we provide additional simulation results. The summaries in terms of median and standard deviation of ISEs for estimating covariance functions are provided in Table A.1. The summaries in terms of median and IQR of ISEs for estimating the 1st eigenfunction are provided in Table A.2. The summaries in terms of median and IQR of ISEs for estimating the 2nd eigenfunction are provided in Table A.3. The summaries in terms of median and IQR of ISEs for estimating the 3rd eigenfunction are provided in Table A.4. The summaries in terms of median and IQR of SEs for estimating the 1st eigenvalue are provided in Table A.5. The summaries in terms of median and IQR of SEs for estimating the 2nd eigenvalue are provided in Table A.6. The summaries in terms of median and IQR of SEs for estimating the 3rd eigenvalue are provided in Table A.7. The summaries in terms of median and IQR of computation times for estimating covariance functions are provided in Table A.8.

For the three additional competitors, 1) FACEs(1-Stage) only estimates the covariance function with independence assumption, that is, we don't take account of correlation structure in the estimation procedure but keep other default settings unchanged; 2) TPRS(50) uses $m + 50$ as the number of knots, where m is the dimension of the null space; 3) TPRS(97) uses $m + 97$ as the number of knots.

A.4 Additional Application: Child Growth Data

The Contents study was conducted in Pampas de San Juan Miraflores and Nuevo Paraso, two peri-urban shanty towns with high population density, 25 km south of central Lima. These peri-urban communities are comprised of 50,000 residents, the majority of whom are immigrants from rural areas of the Peruvian Andes who settled nearly 35 years ago and later claimed unused land on the outskirts of Lima. In the last two decades Pampas has undergone many economic and social developments. The study contains 197 children with anthropomorphic measurements taken from birth. Here we focus on the length curves from birth to 1 year. Each child has 10 to 32

Table A.1: Median and IQR (in parenthesis) of ISEs of eight estimators for estimating the covariance functions. The results are based on 200 replications.

Case	n	m	SNR	FACEs	FACEs(1-Stage)	TPRS	TPRS(50)	TPRS(97)	*fpca.sc*	MLE	*loc*
Case 1	100	5	2	**0.169** (0.085)	0.257 (0.118)	0.460 (0.129)	0.309 (0.152)	0.332 (0.168)	0.305 (0.170)	0.244 (0.224)	0.302 (0.226)
	400	5	2	**0.060** (0.028)	0.083 (0.036)	0.319 (0.041)	0.113 (0.051)	0.132 (0.061)	0.122 (0.053)	0.102 (0.077)	0.307 (0.071)
	100	10	2	**0.094** (0.050)	0.150 (0.076)	0.363 (0.080)	0.165 (0.086)	0.197 (0.097)	0.184 (0.089)	0.143 (0.129)	0.221 (0.093)
	400	10	2	**0.034** (0.019)	0.045 (0.024)	0.285 (0.022)	0.051 (0.023)	0.061 (0.025)	0.057 (0.023)	0.069 (0.050)	0.226 (0.062)
	100	5	5	**0.116** (0.070)	0.207 (0.116)	0.419 (0.085)	0.262 (0.113)	0.280 (0.132)	0.255 (0.116)	0.144 (0.125)	0.302 (0.230)
	400	5	5	**0.034** (0.017)	0.065 (0.026)	0.305 (0.029)	0.086 (0.036)	0.103 (0.038)	0.094 (0.038)	0.079 (0.060)	0.313 (0.067)
	100	10	5	**0.068** (0.056)	0.121 (0.064)	0.346 (0.066)	0.141 (0.075)	0.166 (0.081)	0.154 (0.075)	0.125 (0.094)	0.203 (0.096)
	400	10	5	**0.018** (0.011)	0.036 (0.021)	0.280 (0.017)	0.044 (0.019)	0.050 (0.021)	0.047 (0.021)	0.063 (0.042)	0.229 (0.062)
Case 2	100	5	2	**0.047** (0.017)	0.059 (0.023)	0.061 (0.024)	0.069 (0.035)	0.072 (0.042)	0.070 (0.040)	0.090 (0.053)	0.049 (0.015)
	400	5	2	**0.019** (0.006)	0.024 (0.008)	0.031 (0.009)	0.030 (0.012)	0.032 (0.015)	0.030 (0.015)	0.028 (0.010)	0.029 (0.006)
	100	10	2	**0.025** (0.010)	0.034 (0.012)	0.041 (0.013)	0.042 (0.017)	0.049 (0.022)	0.047 (0.022)	0.040 (0.017)	0.034 (0.009)
	400	10	2	**0.009** (0.003)	0.013 (0.003)	0.022 (0.003)	0.017 (0.004)	0.017 (0.005)	0.016 (0.005)	0.015 (0.004)	0.021 (0.004)
	100	5	5	**0.038** (0.017)	0.049 (0.023)	0.054 (0.020)	0.056 (0.023)	0.062 (0.029)	0.060 (0.030)	0.066 (0.038)	0.043 (0.014)
	400	5	5	**0.014** (0.004)	0.019 (0.005)	0.027 (0.006)	0.025 (0.007)	0.025 (0.008)	0.024 (0.009)	0.023 (0.008)	0.027 (0.004)
	100	10	5	**0.020** (0.006)	0.030 (0.009)	0.035 (0.010)	0.035 (0.013)	0.039 (0.015)	0.038 (0.016)	0.033 (0.012)	0.032 (0.009)
	400	10	5	**0.007** (0.003)	0.012 (0.003)	0.020 (0.003)	0.015 (0.003)	0.014 (0.004)	0.013 (0.004)	0.013 (0.003)	0.020 (0.003)

Table A.2: Median and IQR (in parenthesis) of ISEs of eight estimators for estimating the 1st eigenfunction. The results are based on 200 replications.

Case	n	m	SNR	FACEs	FACEs(1-Stage)	TPRS	TPRS(50)	TPRS(97)	*fpca.sc*	MLE	*loc*
Case 1	100	5	2	0.038 (0.039)	0.050 (0.059)	0.042 (0.048)	0.067 (0.078)	0.075 (0.083)	0.071 (0.092)	0.050 (0.064)	**0.017** (0.017)
	400	5	2	0.013 (0.011)	0.019 (0.016)	0.015 (0.014)	0.025 (0.024)	0.029 (0.024)	0.025 (0.024)	**0.012** (0.012)	0.018 (0.016)
	100	10	2	**0.020** (0.026)	0.037 (0.043)	0.026 (0.025)	0.053 (0.056)	0.060 (0.058)	0.058 (0.062)	0.029 (0.034)	0.025 (0.025)
	400	10	2	0.008 (0.009)	0.012 (0.012)	0.009 (0.007)	0.015 (0.015)	0.017 (0.016)	0.016 (0.016)	**0.005** (0.007)	0.010 (0.008)
	100	5	5	0.025 (0.024)	0.045 (0.051)	0.040 (0.041)	0.062 (0.074)	0.066 (0.080)	0.064 (0.080)	0.032 (0.046)	**0.016** (0.020)
	400	5	5	**0.009** (0.010)	0.018 (0.017)	0.013 (0.010)	0.022 (0.022)	0.025 (0.024)	0.023 (0.024)	**0.009** (0.010)	0.018 (0.016)
	100	10	5	**0.014** (0.017)	0.032 (0.046)	0.024 (0.025)	0.046 (0.051)	0.053 (0.061)	0.052 (0.060)	0.025 (0.028)	0.021 (0.025)
	400	10	5	0.005 (0.006)	0.010 (0.012)	0.009 (0.007)	0.013 (0.015)	0.015 (0.016)	0.014 (0.015)	**0.004** (0.004)	0.008 (0.008)
Case 2	100	5	2	0.436 (0.748)	**0.359** (0.549)	0.554 (0.916)	0.675 (1.024)	0.666 (0.940)	0.689 (0.981)	0.869 (0.959)	0.515 (0.781)
	400	5	2	0.329 (0.556)	0.277 (0.500)	0.434 (0.698)	0.424 (0.699)	0.457 (0.684)	0.464 (0.730)	0.398 (0.718)	**0.209** (0.455)
	100	10	2	0.437 (0.610)	**0.393** (0.683)	0.513 (0.709)	0.597 (0.706)	0.621 (0.676)	0.645 (0.713)	0.578 (0.696)	0.408 (0.638)
	400	10	2	0.205 (0.517)	0.238 (0.517)	0.236 (0.491)	0.261 (0.513)	0.283 (0.513)	0.272 (0.516)	0.217 (0.457)	**0.154** (0.333)
	100	5	5	0.400 (0.701)	**0.366** (0.813)	0.592 (0.720)	0.609 (0.724)	0.674 (0.743)	0.710 (0.777)	0.786 (0.859)	0.452 (0.701)
	400	5	5	0.225 (0.496)	0.240 (0.495)	0.329 (0.477)	0.353 (0.524)	0.381 (0.507)	0.396 (0.644)	0.280 (0.604)	**0.167** (0.271)
	100	10	5	0.390 (0.643)	0.409 (0.710)	0.499 (0.735)	0.527 (0.767)	0.588 (0.794)	0.593 (0.828)	0.468 (0.668)	**0.362** (0.666)
	400	10	5	0.161 (0.327)	0.157 (0.375)	0.209 (0.323)	0.211 (0.326)	0.239 (0.321)	0.248 (0.327)	0.147 (0.298)	**0.095** (0.210)

Table A.3: Median and IQR (in parenthesis) of ISEs of eight estimators for estimating the 2nd eigenfunction. The results are based on 200 replications.

Case	n	m	SNR	FACEs	FACEs(1-Stage)	TPRS	TPRS(50)	TPRS(97)	*fpca.sc*	MLE	*loc*
Case 1	100	5	2	**0.179** (0.300)	0.387 (0.521)	1.452 (0.783)	0.339 (0.467)	0.343 (0.466)	0.304 (0.427)	0.185 (0.249)	0.450 (0.573)
	400	5	2	0.037 (0.037)	0.060 (0.065)	1.623 (0.468)	0.074 (0.067)	0.087 (0.077)	0.079 (0.071)	**0.036** (0.034)	0.146 (0.127)
	100	10	2	**0.069** (0.079)	0.128 (0.137)	1.505 (0.583)	0.127 (0.148)	0.146 (0.151)	0.141 (0.147)	0.070 (0.089)	0.133 (0.136)
	400	10	2	0.017 (0.021)	0.030 (0.036)	1.738 (0.323)	0.033 (0.035)	0.038 (0.036)	0.037 (0.036)	**0.015** (0.014)	0.073 (0.047)
	100	5	5	**0.086** (0.117)	0.196 (0.321)	1.449 (0.647)	0.237 (0.347)	0.256 (0.339)	0.218 (0.258)	0.096 (0.111)	0.362 (0.505)
	400	5	5	**0.020** (0.024)	0.042 (0.054)	1.675 (0.419)	0.059 (0.065)	0.065 (0.070)	0.061 (0.063)	0.021 (0.022)	0.124 (0.107)
	100	10	5	**0.043** (0.058)	0.091 (0.115)	1.560 (0.475)	0.094 (0.114)	0.105 (0.121)	0.101 (0.114)	0.050 (0.051)	0.107 (0.120)
	400	10	5	0.011 (0.012)	0.023 (0.025)	1.742 (0.309)	0.030 (0.025)	0.032 (0.029)	0.032 (0.030)	**0.009** (0.009)	0.070 (0.036)
Case 2	100	5	2	0.700 (0.856)	**0.529** (0.749)	0.883 (0.967)	1.011 (0.968)	0.966 (0.908)	1.050 (0.892)	1.238 (0.862)	0.865 (0.959)
	400	5	2	0.543 (0.823)	0.467 (0.783)	0.701 (0.946)	0.769 (0.936)	0.785 (0.976)	0.838 (0.962)	0.828 (0.782)	**0.450** (0.686)
	100	10	2	**0.640** (0.985)	0.684 (0.871)	0.779 (0.887)	0.862 (0.924)	0.959 (0.956)	0.969 (0.960)	0.971 (0.941)	0.729 (0.860)
	400	10	2	0.374 (0.658)	0.439 (0.612)	0.438 (0.785)	0.466 (0.738)	0.493 (0.706)	0.508 (0.760)	0.374 (0.583)	**0.271** (0.424)
	100	5	5	0.687 (0.951)	**0.600** (0.939)	0.826 (0.901)	0.889 (0.895)	1.012 (0.892)	1.060 (0.901)	1.101 (0.900)	0.815 (0.935)
	400	5	5	0.449 (0.676)	0.424 (0.627)	0.587 (0.704)	0.642 (0.785)	0.683 (0.809)	0.706 (0.841)	0.624 (0.934)	**0.364** (0.425)
	100	10	5	**0.701** (0.963)	0.704 (0.929)	0.876 (0.986)	0.993 (0.944)	0.998 (0.976)	1.035 (0.906)	0.881 (0.878)	0.723 (0.861)
	400	10	5	0.257 (0.446)	0.310 (0.487)	0.340 (0.492)	0.365 (0.589)	0.375 (0.576)	0.384 (0.579)	0.264 (0.437)	**0.203** (0.288)

Table A.4: Median and IQR (in parenthesis) of ISEs of eight estimators for estimating the 3rd eigenfunction. The results are based on 200 replications.

Case	n	m	SNR	FACEs	FACEs(1-Stage)	TPRS	TPRS(50)	TPRS(97)	*fpca.sc*	MLE	*loc*
Case 1	100	5	2	0.352 (0.455)	0.640 (0.707)	1.104 (0.998)	0.679 (0.828)	0.716 (0.689)	0.663 (0.670)	**0.339** (0.462)	0.799 (1.020)
	400	5	2	0.069 (0.061)	0.115 (0.099)	1.527 (0.553)	0.139 (0.121)	0.179 (0.136)	0.163 (0.114)	**0.057** (0.058)	0.229 (0.248)
	100	10	2	**0.110** (0.124)	0.193 (0.203)	1.425 (0.784)	0.190 (0.161)	0.238 (0.193)	0.226 (0.165)	0.117 (0.120)	0.200 (0.251)
	400	10	2	0.035 (0.040)	0.058 (0.044)	1.740 (0.355)	0.054 (0.042)	0.066 (0.043)	0.060 (0.044)	**0.021** (0.017)	0.078 (0.070)
	100	5	5	**0.156** (0.160)	0.435 (0.612)	1.177 (0.813)	0.497 (0.494)	0.511 (0.536)	0.470 (0.466)	**0.156** (0.164)	0.715 (0.827)
	400	5	5	0.036 (0.029)	0.094 (0.071)	1.558 (0.505)	0.099 (0.085)	0.132 (0.094)	0.119 (0.089)	**0.027** (0.025)	0.197 (0.194)
	100	10	5	**0.054** (0.066)	0.134 (0.149)	1.489 (0.629)	0.128 (0.126)	0.160 (0.164)	0.150 (0.143)	0.056 (0.061)	0.148 (0.188)
	400	10	5	0.014 (0.013)	0.041 (0.033)	1.723 (0.372)	0.040 (0.030)	0.049 (0.031)	0.046 (0.030)	**0.010** (0.009)	0.067 (0.052)
Case 2	100	5	2	0.610 (0.900)	**0.433** (0.738)	0.781 (0.992)	0.961 (0.842)	0.966 (0.824)	1.027 (0.823)	1.177 (0.852)	0.979 (0.827)
	400	5	2	0.363 (0.506)	**0.317** (0.431)	0.511 (0.728)	0.626 (0.793)	0.711 (0.804)	0.760 (0.795)	0.759 (0.795)	0.408 (0.489)
	100	10	2	0.663 (0.952)	**0.580** (0.745)	0.721 (0.917)	0.963 (0.897)	1.070 (0.896)	1.111 (0.869)	1.093 (0.907)	0.744 (0.826)
	400	10	2	0.308 (0.471)	0.331 (0.507)	0.319 (0.491)	0.421 (0.604)	0.495 (0.652)	0.505 (0.704)	0.392 (0.598)	**0.279** (0.242)
	100	5	5	0.449 (0.894)	**0.409** (0.718)	0.698 (1.010)	0.874 (0.957)	1.147 (0.931)	1.222 (0.972)	1.229 (0.757)	0.816 (0.910)
	400	5	5	**0.321** (0.582)	0.341 (0.463)	0.425 (0.646)	0.547 (0.682)	0.627 (0.681)	0.709 (0.698)	0.626 (0.725)	0.372 (0.362)
	100	10	5	0.812 (0.898)	**0.625** (0.915)	0.791 (0.858)	0.951 (0.776)	1.060 (0.776)	1.096 (0.789)	1.121 (0.846)	0.788 (0.823)
	400	10	5	**0.193** (0.252)	0.231 (0.371)	0.267 (0.375)	0.325 (0.423)	0.354 (0.436)	0.362 (0.455)	0.257 (0.300)	0.219 (0.199)

Table A.5: 100× Median and IQR (in parenthesis) of SEs of eight estimators for estimating the 1st eigenvalue. The results are based on 200 replications.

Case	n	m	SNR	FACEs	FACEs(1-Stage)	TPRS	TPRS(50)	TPRS(97)	*fpca.sc*	MLE	*loc*
Case 1	100	5	2	**1.736** (5.183)	2.030 (5.342)	2.914 (6.793)	2.647 (6.376)	2.306 (6.328)	2.335 (6.261)	3.877 (9.450)	5.749 (21.780)
	400	5	2	**0.501** (1.213)	0.551 (1.516)	0.624 (1.742)	0.599 (1.450)	0.613 (1.483)	0.603 (1.510)	4.911 (6.375)	11.312 (6.953)
	100	10	2	**1.212** (3.228)	2.206 (4.500)	1.772 (5.345)	1.520 (4.691)	1.554 (4.488)	1.486 (4.460)	4.009 (9.774)	4.366 (8.535)
	400	10	2	**0.258** (0.700)	0.314 (0.788)	0.290 (0.792)	0.301 (0.799)	0.286 (0.825)	0.284 (0.845)	4.840 (4.729)	8.184 (4.808)
	100	5	5	2.087 (4.322)	1.699 (4.230)	1.945 (5.038)	1.516 (4.750)	1.460 (4.605)	**1.463** (4.701)	2.708 (7.754)	5.927 (23.788)
	400	5	5	**0.370** (0.855)	0.410 (1.316)	0.602 (1.434)	0.581 (1.146)	0.524 (1.113)	0.540 (1.165)	4.783 (5.177)	11.382 (5.761)
	100	10	5	1.387 (3.910)	**1.291** (4.211)	1.582 (4.156)	1.335 (3.815)	1.347 (3.876)	1.337 (3.598)	4.295 (8.905)	4.576 (7.862)
	400	10	5	**0.192** (0.534)	0.514 (1.149)	0.311 (0.721)	0.302 (0.679)	0.306 (0.696)	0.308 (0.690)	4.888 (4.324)	8.222 (5.181)
Case 2	100	5	2	0.265 (0.716)	0.351 (0.715)	0.567 (1.193)	0.774 (1.425)	0.859 (1.659)	0.816 (1.530)	1.359 (1.865)	**0.211** (0.542)
	400	5	2	0.071 (0.185)	0.117 (0.281)	0.104 (0.312)	0.140 (0.407)	0.178 (0.440)	0.156 (0.435)	0.180 (0.281)	**0.047** (0.098)
	100	10	2	0.115 (0.269)	0.190 (0.534)	0.241 (0.556)	0.349 (0.663)	0.426 (0.808)	0.417 (0.786)	0.307 (0.601)	**0.084** (0.273)
	400	10	2	0.032 (0.068)	0.051 (0.143)	0.046 (0.092)	0.055 (0.104)	0.059 (0.123)	0.056 (0.113)	0.045 (0.093)	**0.026** (0.065)
	100	5	5	**0.127** (0.405)	0.214 (0.658)	0.331 (0.840)	0.435 (1.073)	0.555 (1.258)	0.537 (1.242)	0.783 (1.289)	0.134 (0.437)
	400	5	5	0.034 (0.105)	0.068 (0.170)	0.069 (0.202)	0.076 (0.239)	0.097 (0.273)	0.095 (0.254)	0.113 (0.229)	**0.033** (0.086)
	100	10	5	**0.063** (0.194)	0.161 (0.458)	0.154 (0.383)	0.181 (0.479)	0.230 (0.514)	0.225 (0.503)	0.207 (0.483)	0.074 (0.225)
	400	10	5	0.028 (0.080)	0.055 (0.133)	0.031 (0.081)	0.037 (0.098)	0.040 (0.109)	0.038 (0.106)	0.046 (0.091)	**0.023** (0.052)

Table A.6: 100× Median and IQR (in parenthesis) of SEs of eight estimators for estimating the 2nd eigenvalue. The results are based on 200 replications.

Case	n	m	SNR	FACEs	FACEs(1-Stage)	TPRS	TPRS(50)	TPRS(97)	*fpca.sc*	MLE	*loc*
Case 1	100	5	2	1.912 (2.943)	3.230 (5.565)	9.879 (6.997)	2.578 (4.115)	1.940 (3.333)	1.530 (3.118)	**0.486** (1.564)	9.088 (4.855)
	400	5	2	0.313 (0.889)	0.867 (1.936)	9.780 (3.503)	0.675 (1.248)	0.581 (1.095)	0.451 (0.927)	**0.146** (0.304)	11.891 (2.358)
	100	10	2	0.869 (1.573)	1.552 (2.927)	9.662 (4.534)	0.699 (1.730)	0.619 (1.673)	0.565 (1.554)	**0.261** (0.736)	7.351 (3.697)
	400	10	2	0.205 (0.444)	0.336 (0.929)	9.901 (2.740)	0.178 (0.418)	0.173 (0.487)	0.152 (0.435)	**0.096** (0.273)	8.810 (2.439)
	100	5	5	2.079 (2.585)	2.826 (5.186)	10.684 (5.510)	2.374 (4.433)	2.132 (3.606)	1.677 (3.346)	**0.455** (1.420)	9.634 (5.118)
	400	5	5	0.356 (0.769)	0.557 (1.523)	10.164 (3.484)	0.381 (0.973)	0.339 (0.945)	0.288 (0.762)	**0.100** (0.292)	12.017 (1.920)
	100	10	5	1.795 (1.835)	1.322 (2.811)	10.207 (4.581)	0.673 (1.509)	0.533 (1.356)	0.492 (1.327)	**0.242** (0.681)	7.831 (4.057)
	400	10	5	0.311 (0.441)	0.302 (0.595)	9.793 (1.964)	0.124 (0.380)	0.128 (0.323)	0.123 (0.295)	**0.055** (0.175)	9.106 (2.132)
Case 2	100	5	2	0.165 (0.333)	0.300 (0.766)	0.177 (0.447)	0.123 (0.397)	0.157 (0.430)	0.174 (0.430)	0.241 (0.596)	**0.140** (0.333)
	400	5	2	**0.021** (0.060)	0.040 (0.090)	0.027 (0.065)	0.024 (0.070)	0.028 (0.077)	0.027 (0.081)	0.035 (0.077)	0.098 (0.196)
	100	10	2	0.063 (0.158)	0.077 (0.172)	0.061 (0.146)	0.057 (0.160)	0.064 (0.200)	0.067 (0.207)	**0.046** (0.144)	0.091 (0.212)
	400	10	2	0.015 (0.035)	0.018 (0.050)	**0.012** (0.050)	0.013 (0.054)	0.016 (0.054)	0.014 (0.054)	0.015 (0.040)	0.056 (0.121)
	100	5	5	**0.079** (0.252)	0.180 (0.461)	0.128 (0.316)	0.090 (0.254)	0.089 (0.278)	0.089 (0.270)	0.136 (0.302)	0.154 (0.367)
	400	5	5	**0.016** (0.042)	0.031 (0.074)	0.029 (0.071)	0.027 (0.075)	0.027 (0.075)	0.029 (0.069)	0.028 (0.069)	0.110 (0.216)
	100	10	5	0.041 (0.102)	0.058 (0.109)	0.043 (0.115)	0.046 (0.118)	0.048 (0.126)	0.043 (0.125)	**0.038** (0.098)	0.115 (0.211)
	400	10	5	0.012 (0.037)	0.016 (0.041)	**0.010** (0.034)	0.012 (0.031)	0.012 (0.030)	0.012 (0.033)	0.011 (0.036)	0.068 (0.126)

Table A.7: 100× Median and IQR (in parenthesis) of SEs of eight estimators for estimating the 3rd eigenvalue. The results are based on 200 replications.

Case	n	m	SNR	FACEs	FACEs(1-Stage)	TPRS	TPRS(50)	TPRS(97)	*fpca.sc*	MLE	*loc*
Case 1	100	5	2	**0.167** (0.491)	0.791 (1.532)	3.147 (2.115)	0.717 (1.254)	0.466 (1.074)	0.500 (1.095)	0.224 (0.711)	2.521 (1.915)
	400	5	2	**0.077** (0.197)	0.154 (0.367)	2.778 (1.909)	0.164 (0.464)	0.155 (0.420)	0.157 (0.406)	0.088 (0.218)	2.986 (0.959)
	100	10	2	0.115 (0.325)	0.406 (0.834)	2.964 (1.778)	0.329 (0.679)	0.262 (0.531)	0.285 (0.552)	**0.114** (0.371)	2.337 (1.448)
	400	10	2	0.037 (0.084)	0.082 (0.187)	3.113 (1.452)	0.107 (0.216)	0.093 (0.196)	0.094 (0.186)	**0.028** (0.087)	2.581 (0.833)
	100	5	5	0.274 (0.537)	0.759 (1.449)	3.237 (2.045)	0.615 (1.031)	0.438 (0.949)	0.453 (0.914)	**0.192** (0.518)	2.646 (1.684)
	400	5	5	**0.034** (0.084)	0.113 (0.269)	3.049 (1.690)	0.132 (0.340)	0.127 (0.269)	0.122 (0.290)	0.038 (0.086)	3.066 (0.808)
	100	10	5	0.130 (0.288)	0.308 (0.613)	3.059 (1.669)	0.216 (0.565)	0.191 (0.470)	0.193 (0.427)	**0.093** (0.224)	2.416 (1.112)
	400	10	5	**0.019** (0.054)	0.043 (0.128)	2.960 (1.349)	0.055 (0.151)	0.040 (0.123)	0.038 (0.116)	0.021 (0.063)	2.680 (0.590)
Case 2	100	5	2	0.418 (1.067)	0.644 (1.441)	0.407 (0.826)	0.200 (0.517)	0.168 (0.539)	0.184 (0.501)	**0.085** (0.230)	0.370 (0.474)
	400	5	2	0.029 (0.083)	0.060 (0.150)	0.056 (0.138)	0.039 (0.083)	0.033 (0.076)	0.030 (0.067)	**0.016** (0.039)	0.258 (0.311)
	100	10	2	0.072 (0.154)	0.146 (0.299)	0.097 (0.188)	0.047 (0.122)	0.038 (0.109)	0.035 (0.100)	**0.019** (0.065)	0.227 (0.294)
	400	10	2	0.012 (0.032)	0.014 (0.034)	0.018 (0.048)	0.014 (0.035)	0.013 (0.035)	0.013 (0.037)	**0.011** (0.027)	0.162 (0.151)
	100	5	5	0.204 (0.547)	0.441 (1.065)	0.257 (0.478)	0.143 (0.328)	0.095 (0.259)	0.092 (0.294)	**0.060** (0.203)	0.380 (0.443)
	400	5	5	**0.013** (0.041)	0.029 (0.079)	0.039 (0.090)	0.028 (0.068)	0.022 (0.050)	0.019 (0.051)	0.017 (0.048)	0.224 (0.236)
	100	10	5	0.044 (0.102)	0.081 (0.215)	0.067 (0.149)	0.039 (0.090)	0.026 (0.071)	0.026 (0.072)	**0.023** (0.054)	0.217 (0.251)
	400	10	5	**0.010** (0.028)	0.016 (0.033)	0.013 (0.037)	0.012 (0.031)	0.013 (0.030)	0.014 (0.029)	**0.010** (0.025)	0.154 (0.159)

Table A.8: Median and IQR (in parenthesis) of computation times (in seconds) of eight estimators for estimating the covariance functions on a desktop with a 2.3 GHz CPU and 8 GB of RAM. The results are based on 200 replications.

Case	n	m	SNR	FACEs	FACEs(1-Stage)	TPRS	TPRS(50)	TPRS(97)	*fpca.sc*	MLE	*loc*
Case 1	100	5	2	14.1 (1.1)	5.6 (0.2)	4.1 (1.2)	6.1 (0.2)	9.8 (0.4)	2.4 (1.2)	141.7 (52.7)	477.3 (43.8)
	400	5	2	56.6 (12.3)	15.3 (0.5)	8.5 (2.9)	11.1 (0.7)	79.8 (44.2)	11.0 (7.1)	615.1 (209.6)	1304.8 (239.0)
	100	10	2	19.3 (1.0)	5.3 (0.3)	7.5 (1.4)	10.5 (1.1)	37.8 (13.4)	12.2 (4.4)	153.1 (41.7)	1680.5 (258.9)
	400	10	2	79.6 (2.3)	21.5 (6.4)	23.3 (4.0)	34.7 (8.4)	200.6 (68.5)	55.7 (17.5)	652.8 (149.3)	772.8 (31.5)
	100	5	5	15.6 (0.2)	4.7 (0.2)	4.6 (0.2)	5.6 (0.2)	16.7 (5.3)	2.9 (0.3)	174.2 (26.0)	462.5 (83.8)
	400	5	5	66.4 (3.9)	21.5 (2.3)	10.0 (2.0)	14.3 (1.9)	55.0 (11.6)	14.1 (4.5)	767.3 (216.4)	2462.8 (387.2)
	100	10	5	16.0 (1.1)	6.2 (0.5)	6.9 (1.5)	12.6 (1.6)	36.6 (6.0)	10.9 (3.3)	143.9 (46.5)	2781.0 (426.8)
	400	10	5	81.0 (1.7)	21.0 (1.6)	23.6 (4.9)	33.8 (3.1)	217.6 (75.7)	55.6 (17.4)	758.3 (137.2)	486.0 (36.7)
Case 2	100	5	2	16.7 (3.9)	5.8 (0.3)	4.1 (0.3)	6.3 (0.4)	17.1 (6.6)	2.3 (0.3)	166.6 (28.7)	1035.5 (112.5)
	400	5	2	50.0 (17.5)	16.3 (0.8)	8.9 (2.4)	11.4 (0.7)	81.5 (39.4)	11.4 (4.1)	907.6 (262.2)	2704.5 (1303.5)
	100	10	2	21.4 (1.4)	6.2 (0.7)	6.2 (0.6)	12.3 (2.0)	37.0 (12.5)	9.6 (2.1)	150.3 (23.1)	2302.8 (461.4)
	400	10	2	70.3 (1.7)	20.7 (2.1)	19.5 (3.7)	31.4 (3.9)	186.5 (64.9)	42.7 (13.7)	701.2 (117.8)	362.8 (63.3)
	100	5	5	15.2 (0.1)	3.7 (0.2)	4.1 (0.2)	4.5 (0.3)	9.7 (1.6)	2.4 (0.2)	179.8 (17.8)	739.4 (60.1)
	400	5	5	57.3 (2.9)	19.4 (0.6)	9.0 (0.6)	12.7 (0.8)	19.2 (5.1)	13.4 (1.7)	835.1 (97.7)	2041.2 (981.4)
	100	10	5	14.8 (0.4)	5.6 (0.4)	6.9 (0.3)	11.3 (1.1)	28.9 (3.2)	11.3 (1.1)	172.0 (13.0)	2719.9 (1005.6)
	400	10	5	58.5 (1.3)	22.9 (0.6)	22.1 (5.3)	35.1 (2.2)	56.5 (7.4)	52.9 (17.8)	772.8 (210.8)	1532.4 (247.8)

measurements of length, with 4320 data points in total. Figure A.1 displays the sample length trajectories. We apply the proposed method to the data. The estimated population mean is plotted in Figure A.1 as a dashed line. The estimated mean curve is generally increasing with age, which is expected. In Figure A.2, we plot the estimated variance and correlation functions. The estimated variance function is increasing as a function of age. The estimated correlation function is shown as a heat map. Each point in the heat map represents the estimated correlation between two days and the color corresponds to the correlation values with red indicating higher correlation and blue indicating lower correlation. The diagonal is dark red indicating perfect correlation, while the minimum correlation is about 0.2. Given the estimated covariance function, using the framework described in Section 1.4, we predict each child's length trajectories. Figure A.3 displays for 4 children the predicted length trajectories with point-wise confidence bands. We randomly sample varying numbers of observations to see the effect of the number of observations on the prediction. We see that the point-wise confidence intervals are generally narrower with a larger number of observations, which is also as expected.

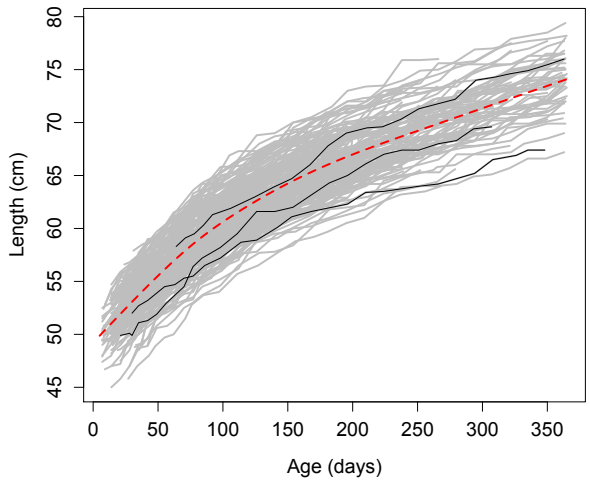


Figure A.1: Length trajectories of about 200 children from birth to 1 year old. The estimated population mean is the dashed red line.

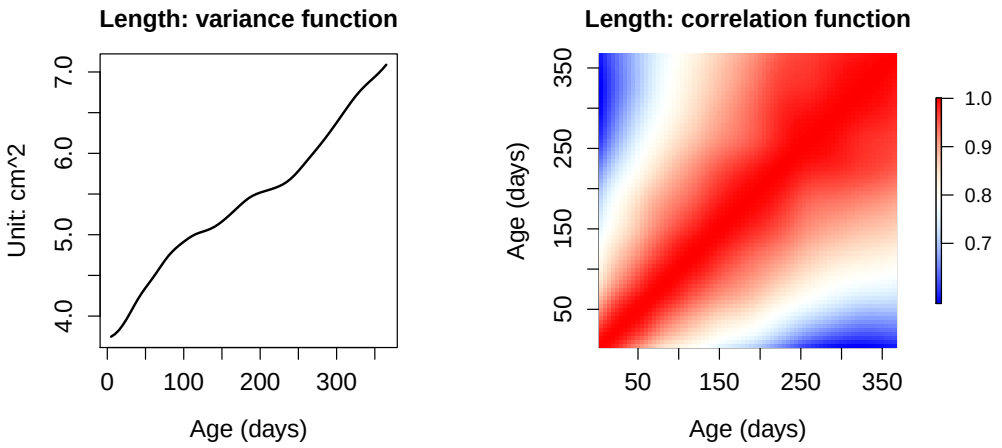


Figure A.2: Estimated variance function (left panel) and correlation function (right panel) for the length of children from birth to 1 year old.

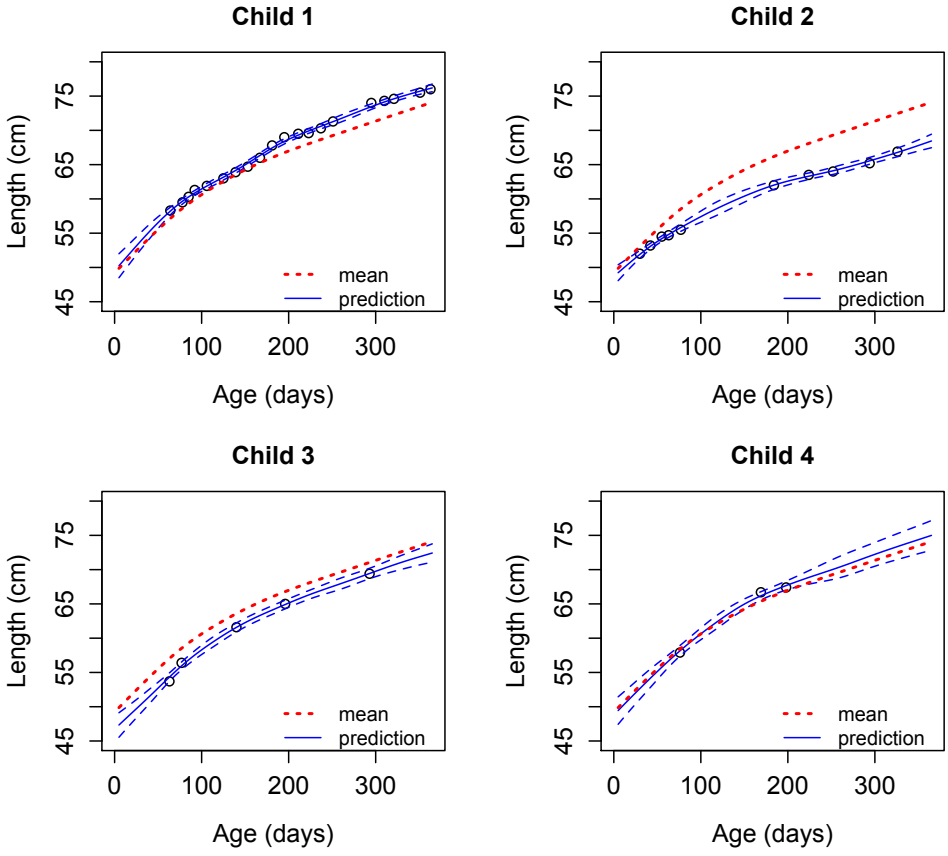


Figure A.3: Predicted child-specific length trajectories from birth to 1 year old and associated 95% confidence bands for 4 children. The estimated population mean is the dotted red line.

Appendix B

Supplement for Chapter 2

Assumptions needed for Theorem 3: Assume Assumptions A, B, C, D, E, and F in Bunea and Xiao (2015) hold for the covariance function $r(s,t)$. Moreover, we assume that the non-zero eigenvalues of $r(s,t)$ are distinct.

B.1 Proofs of Theorems in Section 2.3

Proof of Lemma 2. Without loss of generality, we assume $\sigma_\epsilon^2 = 1$. Let $\boldsymbol{\mu}$ be the difference of the two mean functions. Let $\mathbf{A} = \boldsymbol{\Phi}(\widetilde{\mathbf{t}})\boldsymbol{\Lambda}^{1/2}$ and $\mathbf{A}_1 = \boldsymbol{\Phi}(\mathbf{t})\boldsymbol{\Lambda}^{1/2}$. Then $\mathbf{A}$ can be partitioned as $\mathbf{A} = (\mathbf{A}_1^T, \mathbf{A}_2^T)^T$ after some proper permutation of the index in $\widetilde{\mathbf{t}}$, where $\mathbf{A}_1$ and $\mathbf{A}_2$ are of dimensions $p \times \infty$ and $c \times \infty$, respectively. Similarly, $\boldsymbol{\mu}$ can also be partitioned as $\boldsymbol{\mu} = (\boldsymbol{\mu}_1^T, \boldsymbol{\mu}_2^T)^T$, where $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ are of dimensions $p \times 1$ and $c \times 1$. It leads to

$$\boldsymbol{\Phi}(\widetilde{\mathbf{t}})\boldsymbol{\Lambda}\boldsymbol{\Phi}(\widetilde{\mathbf{t}})^T + \mathbf{I}_{p+c} = \mathbf{A}\mathbf{A}^T + \mathbf{I}_{p+c} = \begin{pmatrix} \mathbf{A}_1\mathbf{A}_1^T + \mathbf{I}_p & \mathbf{A}_1\mathbf{A}_2^T \\ \mathbf{A}_2\mathbf{A}_1^T & \mathbf{A}_2\mathbf{A}_2^T + \mathbf{I}_c \end{pmatrix}.$$

Next, let $\mathbf{A}_{11} = \mathbf{A}_1\mathbf{A}_1^T + \mathbf{I}_p$, $\mathbf{A}_{12} = \mathbf{A}_{21} = \mathbf{A}_1\mathbf{A}_2^T$ and $\mathbf{A}_{22} = \mathbf{A}_2\mathbf{A}_2^T + \mathbf{I}_c$. Then

$$(\mathbf{A}\mathbf{A}^T + \mathbf{I}_{p+c})^{-1} = \begin{pmatrix} \mathbf{A}_{11}^{-1} + \mathbf{A}_{11}^{-1}\mathbf{A}_{12}\mathbf{A}_{22.1}^{-1}\mathbf{A}_{21}\mathbf{A}_{11}^{-1} & -\mathbf{A}_{11}^{-1}\mathbf{A}_{12}\mathbf{A}_{22.1}^{-1} \\ -\mathbf{A}_{22.1}^{-1}\mathbf{A}_{12} - \mathbf{A}_{11}^{-1} & \mathbf{A}_{22.1}^{-1} \end{pmatrix},$$

where $\mathbf{A}_{22.1} = \mathbf{A}_{22} - \mathbf{A}_{21}\mathbf{A}_{11}^{-1}\mathbf{A}_{12}$. Note that (i) $\quad s(\mathbf{t}) = \boldsymbol{\mu}_1^T\mathbf{A}_{11}^{-1}\boldsymbol{\mu}_1$ and

$$\begin{aligned} (ii) \quad s(\widetilde{\mathbf{t}}) &= \boldsymbol{\mu}^T(\mathbf{A}\mathbf{A}^T + \mathbf{I}_{p+c})^{-1}\boldsymbol{\mu} = (\boldsymbol{\mu}_1^T, \boldsymbol{\mu}_2^T)(\mathbf{A}\mathbf{A}^T + \mathbf{I}_{p+c})^{-1}(\boldsymbol{\mu}_1^T, \boldsymbol{\mu}_2^T)^T \\ &= \boldsymbol{\mu}_1^T(\mathbf{A}_{11}^{-1} + \mathbf{A}_{11}^{-1}\mathbf{A}_{12}\mathbf{A}_{22.1}^{-1}\mathbf{A}_{21}\mathbf{A}_{11}^{-1})\boldsymbol{\mu}_1 - \boldsymbol{\mu}_1^T\mathbf{A}_{11}^{-1}\mathbf{A}_{12}\mathbf{A}_{22.1}^{-1}\boldsymbol{\mu}_2 \\ &- (\boldsymbol{\mu}_1^T\mathbf{A}_{11}^{-1}\mathbf{A}_{12}\mathbf{A}_{22.1}^{-1}\boldsymbol{\mu}_2)^T + \boldsymbol{\mu}_2^T\mathbf{A}_{22.1}^{-1}\boldsymbol{\mu}_2. \end{aligned}$$

It leads to

$$\begin{aligned} s(\widetilde{\mathbf{t}}) - s(\mathbf{t}) &= \boldsymbol{\mu}_1^T\mathbf{A}_{11}^{-1}\mathbf{A}_{12}\mathbf{A}_{22.1}^{-1}\mathbf{A}_{21}\mathbf{A}_{11}^{-1}\boldsymbol{\mu}_1 - \boldsymbol{\mu}_1^T\mathbf{A}_{11}^{-1}\mathbf{A}_{12}\mathbf{A}_{22.1}^{-1}\boldsymbol{\mu}_2 \\ &- (\boldsymbol{\mu}_1^T\mathbf{A}_{11}^{-1}\mathbf{A}_{12}\mathbf{A}_{22.1}^{-1}\boldsymbol{\mu}_2)^T + \boldsymbol{\mu}_2^T\mathbf{A}_{22.1}^{-1}\boldsymbol{\mu}_2 \\ &= \mathbf{b}1^T\mathbf{b}_1 - \mathbf{b}_1^T\mathbf{b}_2 - \mathbf{b}_2^T\mathbf{b}_1 + \mathbf{b}_2^T\mathbf{b}_2 \\ &= (\mathbf{b}_1 - \mathbf{b}_2)^T(\mathbf{b}_1 - \mathbf{b}_2), \end{aligned}$$

where $\mathbf{b}_1 = \mathbf{A}_{22.1}^{-1/2}\mathbf{A}_{21}\mathbf{A}_{11}^{-1}\boldsymbol{\mu}_1$ and $\mathbf{b}_2 = \mathbf{A}_{22.1}^{-1/2}\boldsymbol{\mu}_2$. Thus, we have shown that

$$s(\widetilde{\mathbf{t}}) - s(\mathbf{t}) = (\mathbf{b}_1 - \mathbf{b}_2)^T(\mathbf{b}_1 - \mathbf{b}_2) \geq 0.$$

□

Proof of Theorem 1. Since by Lemma 2, $s(\widetilde{\mathbf{t}}) - s(\mathbf{t}) \geq 0$ if $\mathbf{t} \subseteq \widetilde{\mathbf{t}}$, it follows that AUROC($\widetilde{\mathbf{t}}$) $\geq$ AUROC($\mathbf{t}$). To prove PCC($\widetilde{\mathbf{t}}$) $\geq$ PCC($\mathbf{t}$) if $\mathbf{t} \subseteq \widetilde{\mathbf{t}}$, we only need to prove the monotonicity of the following generic function

$$F(x) = \pi_1\Phi\left(x + \frac{b}{x}\right) + \pi_0\Phi\left(x - \frac{b}{x}\right),$$

where $\pi_1 + \pi_0 = 1$, and without loss of generality $b = \frac{1}{2}\log\frac{\pi_1}{\pi_0}$ is assumed greater than 0 and $x > 0$. Let f denote the probability density function of the standard normal distribution, then

$$\begin{aligned}
F'(x) &= \pi_1 f\left(x + \frac{b}{x}\right)\left(1 - \frac{b}{x^2}\right) + \pi_0 f\left(x - \frac{b}{x}\right)\left(1 + \frac{b}{x^2}\right) \\
&= \frac{1}{\sqrt{2\pi}}\exp\left\{-\frac{x^2 + \frac{b^2}{x^2}}{2}\right\}\left\{\pi_1 e^{-b}\left(1 - \frac{b}{x^2}\right) + \pi_0 e^{b}\left(1 + \frac{b}{x^2}\right)\right\} \\
&= \frac{1}{\sqrt{2\pi}}\exp\left\{-\frac{x^2 + \frac{b^2}{x^2}}{2}\right\}\left\{\pi_1\sqrt{\frac{\pi_0}{\pi_1}}\left(1 - \frac{b}{x^2}\right) + \pi_0\sqrt{\frac{\pi_1}{\pi_0}}\left(1 + \frac{b}{x^2}\right)\right\} \\
&= \frac{\sqrt{2\pi_0\pi_1}}{\sqrt{\pi}}\exp\left\{-\frac{x^2 + \frac{b^2}{x^2}}{2}\right\} > 0,
\end{aligned}$$

which implies $\mathrm{PCC}(s(\mathbf{t}))$ is monotone increasing with respect to $s(\mathbf{t})$. Hence, we have proved that if $\mathbf{t} \subseteq \widetilde{\mathbf{t}}$, then $\mathrm{PCC}(\widetilde{\mathbf{t}}) \geq \mathrm{PCC}(\mathbf{t})$. □

Proof of Theorem 3. First, assume $\sigma_\epsilon^2 = 1$ without loss of generality. Let $\boldsymbol{\mu}(\mathbf{t}_p) = \boldsymbol{\mu}_1(\mathbf{t}_p) - \boldsymbol{\mu}_0(\mathbf{t}_p) = \boldsymbol{\mu}^T\boldsymbol{\Phi}(\mathbf{t}_p)$. Let $\mathbf{UDU}^T$ be the eigen-decomposition of $\boldsymbol{\Phi}(\mathbf{t}_p)\boldsymbol{\Lambda}\boldsymbol{\Phi}(\mathbf{t}_p)^T$. Define $\mathbf{V} = \mathbf{U}^T\boldsymbol{\Phi}(\mathbf{t}_p)\boldsymbol{\Lambda}^{1/2}$, then $\boldsymbol{\Phi}(\mathbf{t}_p)\boldsymbol{\Lambda}^{1/2} = \mathbf{UV}$. It follows that

$$\begin{aligned}
s(\mathbf{t}_p) &= \boldsymbol{\mu}^T\boldsymbol{\Lambda}^{-1/2}\boldsymbol{\Lambda}^{1/2}\boldsymbol{\Phi}(\mathbf{t}_p)^T(\mathbf{UDU}^T + \mathbf{I}_p)^{-1}\boldsymbol{\Phi}(\mathbf{t}_p)\boldsymbol{\Lambda}^{1/2}\boldsymbol{\Lambda}^{-1/2}\boldsymbol{\mu} \\
&= \boldsymbol{\mu}^T\boldsymbol{\Lambda}^{-1/2}(\mathbf{UV})^T(\mathbf{UDU}^T + \mathbf{I}_p)^{-1}(\mathbf{UV})\boldsymbol{\Lambda}^{-1/2}\boldsymbol{\mu} \\
&= \boldsymbol{\mu}^T\boldsymbol{\Lambda}^{-1/2}\mathbf{V}^T(\mathbf{D} + \mathbf{I})^{-1}\mathbf{V}\boldsymbol{\Lambda}^{-1/2}\boldsymbol{\mu}.
\end{aligned}$$

Then we have

$$\begin{aligned}
& |\|\boldsymbol{\Lambda}^{-1/2}\boldsymbol{\mu}\|^2 - s(\mathbf{t}_p)| \\
= \; & |\mathrm{tr}\{\boldsymbol{\mu}^T\boldsymbol{\Lambda}^{-1/2}\boldsymbol{\Lambda}^{-1/2}\boldsymbol{\mu}\} - s(\mathbf{t}_p)| \\
= \; & |\mathrm{tr}\{\boldsymbol{\mu}^T\boldsymbol{\Lambda}^{-1/2}[\mathbf{I} - \mathbf{V}^T(\mathbf{D} + \mathbf{I})^{-1}\mathbf{V}]\boldsymbol{\Lambda}^{-1/2}\boldsymbol{\mu}\}| \\
\leq \; & \|\boldsymbol{\Lambda}^{-1/2}\boldsymbol{\mu}\|^2\mathrm{tr}\{\mathbf{I} - \mathbf{V}^T(\mathbf{D} + \mathbf{I})^{-1}\mathbf{V}\}
\end{aligned}$$

and equivalently

$$\left|1 - \frac{s(\mathbf{t}_p)}{\|\mathbf{\Lambda}^{-1/2}\boldsymbol{\mu}\|^2}\right| \leq \mathrm{tr}\{\mathbf{I} - \mathbf{V}^T(\mathbf{D}+\mathbf{I})^{-1}\mathbf{V}\}.$$

Then we will prove $\mathrm{tr}\{\mathbf{I} - \mathbf{V}^T(\mathbf{D}+\mathbf{I})^{-1}\mathbf{V}\} \to 0$ as $p \to \infty$. Denote the jth diagonal of $\mathbf{D}$ by d_j. Let $\mathbf{V} = (v_{jk})$ and $\mathbf{U} = [\mathbf{u}_1, \cdots, \mathbf{u}_p]$. Since $\mathbf{V} = \mathbf{U}^T\mathbf{\Phi}(\mathbf{t}_p)\mathbf{\Lambda}^{1/2}$, we obtain that $v_{jk} = \mathbf{u}^T\boldsymbol{\phi}_k\lambda_k^{1/2}$, where $\boldsymbol{\phi}_k = \{\phi_k(0), \phi_k(1/p), \cdots, \phi_k(1-p)/p\}^T$. Then

$$\mathrm{tr}\{\mathbf{I} - \mathbf{V}^T(\mathbf{D}+\mathbf{I})^{-1}\mathbf{V}\} = \sum_{k\geq 1}\left(1 - \sum_{j\geq 1}\frac{v_{jk}^2}{d_j+1}\right),$$

Let λ_1 and λ_∞ denote the largest and smallest diagonal element of $\mathbf{\Lambda}$ respectively. Note that

$$\frac{1}{\lambda_1}\sum_{k\geq 1}\lambda_k\left(1 - \sum_{j\geq 1}\frac{v_{jk}^2}{d_j+1}\right) \leq \sum_{k\geq 1}\left(1 - \sum_{j\geq 1}\frac{v_{jk}^2}{d_j+1}\right) \leq \frac{1}{\lambda_\infty}\sum_{k\geq 1}\lambda_k\left(1 - \sum_{j\geq 1}\frac{v_{jk}^2}{d_j+1}\right).$$

Fix any arbitrary small number $0 < \epsilon < 1/2$. Then there exists an integer K such that $\sum_{k>K}\lambda_k < \epsilon$. Then by proposition 5.1 in Bunea and Xiao (2015), for any sufficiently large m (depending on ϵ),

$v_{kk}^2/(p\lambda_k) \geq 1-\epsilon$, $|d_k - p\lambda_k| \leq \epsilon$ for all $k \leq K$. It follows that

$$\begin{aligned}
\sum_{k\geq 1}\lambda_k\left(1-\sum_{j\geq 1}\frac{v_{jk}^2}{d_j+1}\right) &= \sum_{k\geq 1}\lambda_k\left(1-\sum_{j\geq 1}\frac{v_{jk}^2}{p\lambda_k}\frac{p\lambda_k}{d_j+1}\right) \\
&\leq \sum_{1\leq k\leq K}\lambda_k\left(1-\frac{v_{kk}^2}{p\lambda_k}\frac{p\lambda_k}{d_k+1}\right)+\sum_{k>K}\lambda_k \\
&\leq \sum_{1\leq k\leq K}\lambda_k\left\{1-\frac{p\lambda_k}{p\lambda_k+\epsilon+1}(1-\epsilon)\right\}+\epsilon \\
&\leq \sum_{1\leq k\leq K}\lambda_k\left(\frac{p\lambda_k\epsilon+\epsilon^2+\epsilon+1}{p\lambda_k+\epsilon+1}\right)+\epsilon \\
&\leq \sum_{1\leq k\leq K}\lambda_k\left(\frac{1}{p\lambda_k+\epsilon+1}+\epsilon\right)+\epsilon \\
&\leq \epsilon\sum_{1\leq k\leq K}\lambda_k+\sum_{k\geq 1}\lambda_k\frac{1}{p\lambda_k+\epsilon+1}+\epsilon \\
&\leq \epsilon\sum_{1\leq k\leq K}\lambda_k+K/p+\epsilon.
\end{aligned}$$

Therefore, with sufficient large p, we can make $\mathrm{tr}[\mathbf{\Lambda}\{\mathbf{I}-\mathbf{V}^T(\mathbf{D}+\mathbf{I})^{-1}\mathbf{V}\}]$ smaller than ϵ, up to some finite multiplicative constant. We have shown that $\mathrm{tr}\{\mathbf{I}-\mathbf{V}^T(\mathbf{D}+\mathbf{I})^{-1}\mathbf{V}\} \to 0$ and hence $\frac{s(\mathbf{t}_p)}{\|\mathbf{\Lambda}^{-1/2}\boldsymbol{\mu}\|^2} \to 1$ as $p \to \infty$ for the fixed design.

Next, we will determine $\mathrm{PCC}(\mathbf{t}_p)$ and $\mathrm{AUROC}(\mathbf{t}_p)$ according to $\|\mathbf{\Lambda}^{-1/2}\boldsymbol{\mu}\|^2 = \sum_{k\geq 1}\lambda_k^{-1}\mu_k^2$. Recall that

$$\begin{aligned}
\mathrm{PCC}(\mathbf{t}_p) &= \pi_1\Phi\left(\frac{1}{2}\sqrt{s(\mathbf{t}_p)}+\frac{\log\frac{\pi_1}{1-\pi_1}}{\sqrt{s(\mathbf{t}_p)}}\right)+(1-\pi_1)\Phi\left(\frac{1}{2}\sqrt{s(\mathbf{t}_p)}-\frac{\log\frac{\pi_1}{1-\pi_1}}{\sqrt{s(\mathbf{t}_p)}}\right), \\
\mathrm{AUROC}(\mathbf{t}_p) &= \Phi\left(\frac{1}{\sqrt{2}}\sqrt{s(\mathbf{t}_p)}\right).
\end{aligned}$$

(*a*) If $\sum_{k\geq 1}\lambda_k^{-1}\mu_k^2 < \infty$, then $s(\mathbf{t}_p) \to \sum_{k\geq 1}\lambda_k^{-1}\mu_k^2$ as p diverges and optimal but imperfect

classification is obtained,

$$\begin{aligned}
\mathrm{PCC}(\mathbf{t}_p) &= \pi_1 \Phi\left(\frac{1}{2}\sqrt{\sum_{k\geq 1}\lambda_k^{-1}\mu_k^2} + \frac{\log\frac{\pi_1}{1-\pi_1}}{\sqrt{\sum_{k\geq 1}\lambda_k^{-1}\mu_k^2}}\right) \\
&+ (1-\pi_1)\Phi\left(\frac{1}{2}\sqrt{\sum_{k\geq 1}\lambda_k^{-1}\mu_k^2} - \frac{\log\frac{\pi_1}{1-\pi_1}}{\sqrt{\sum_{k\geq 1}\lambda_k^{-1}\mu_k^2}}\right), \\
\mathrm{AUROC}(\mathbf{t}_p) &= \Phi\left(\frac{1}{\sqrt{2}}\sqrt{\sum_{k\geq 1}\lambda_k^{-1}\mu_k^2}\right).
\end{aligned}$$

(*b*) If $\sum_{k\geq 1}\lambda_k^{-1}\mu_k^2 = \infty$, then $s(\mathbf{t}_p) \to \infty$ as p diverges and $\mathrm{PCC}(\mathbf{t}_p) = 1$ and $\mathrm{AUROC}(\mathbf{t}_p) = 1$, which implies perfect classification. □

B.2 Technical Details for Section 2.4.1

Proof of Theorem 4. Note that $s(\mathbf{t}) = d(\mathbf{t})'\mathbf{\Sigma}^{-1}(\mathbf{t})d(\mathbf{t})$ with $\mathbf{\Sigma}(\mathbf{t}) = r(\mathbf{t},\mathbf{t}) + \sigma_\epsilon^2\mathbf{I}_p$. For simplicity, let $\mathbf{a} = (a_1,\ldots,a_p)' = \mathbf{\Sigma}^{-1}d(\mathbf{t})$. Thus,

$$\frac{\partial s(\mathbf{t})}{\partial t_k} = 2\dot{d}(t_k)a_k - \dot{d}(\mathbf{t})'\mathbf{\Sigma}^{-1}(\mathbf{t})\left\{\frac{\partial r(\mathbf{t},\mathbf{t})}{\partial t_k}\right\}\mathbf{\Sigma}^{-1}(\mathbf{t})d(\mathbf{t}),$$

and

$$\frac{\partial r(\mathbf{t},\mathbf{t})}{\partial t_k} = \dot{r}(t_k,\mathbf{t})\otimes\mathbf{e}_k' + \mathbf{e}_k\otimes\dot{r}(t_k,\mathbf{t})',$$

where $\mathbf{e}_k$ is a length p vector of 0s except that the kth element is 1, $\otimes$ denotes the Kronecker product. Then simple algebra proves Theorem 4. □

Evaluation of derivatives of spline estimators:

The following Lemma can be easily established by de Boor (1978).

Lemma 6. *Let $\{B_{\ell,k}(t)\}_{1\leq\ell\leq L} \in \mathbb{R}^L$ be the collection of B-spline basis functions of order k evaluated at sampling point t with the sequence of knots $\{t_1,t_2,\ldots,t_L\}$. Let $d(t) = \sum_{\ell=1}^L \alpha_\ell B_{\ell,k}(t)$, where $\{\alpha_\ell\}_{1\leq\ell\leq L}$ is a coefficient vector, Let $r(s,t) = \sum_{\ell_1,\ell_2} B_{\ell_1,k}(s)B_{\ell_2,k}(t)\theta_{\ell_1,\ell_2}$, where $(\theta_{\ell_1,\ell_2})_{1\leq\ell_1,\ell_2\leq L}$ is*

a coefficient matrix. Then,

$$\begin{aligned}
\frac{\partial d(t)}{\partial t} &= (k-1)\sum_{\ell}\frac{\alpha_{\ell}-\alpha_{\ell-1}}{t_{\ell+k-1}-t_{\ell}}B_{\ell,k-1}(t), \\
\frac{\partial r(s,t)}{\partial s} &= (k-1)\sum_{\ell_1,\ell_2}\frac{\theta_{\ell_1,\ell_2}-\theta_{\ell_1-1,\ell_2}}{t_{\ell_1+k-1}-t_{\ell_1}}B_{\ell_1,k-1}(s)B_{\ell_2,k}(t).
\end{aligned}$$

Appendix C

Supplement for Chapter 3

C.1 Additional Simulation Results

C.1.1 Simulation Study for Case of Scalar Covariate

We compare the performance of the proposed method to the conventional quantile regression, and consider generating five models with different settings but all with scalar-only covariates as follows. The first three models follow Reich et al. (2010) and are all with linear quantiles,

$$\begin{aligned} \text{Model 1} \qquad Y_i &= 1 + X_i + \epsilon_i \\ \text{Model 2} \qquad Y_i &= 1 + X_i + \pi_i\epsilon_{1i} + (1-\pi_i)\epsilon_{2i} \\ \text{Model 3} \qquad Y_i &= 1 + X_i + (1.1 - X_i)\epsilon_i, \end{aligned}$$

Model 1 is generated as above, here $X_i \overset{i.i.d}{\sim} N(0,1)$, $\epsilon_i \overset{i.i.d}{\sim} N(0,1)$. Since $Y_i \sim N(1+X_i, 1)$, the linear quantile can be analytically determined by $Q_\tau = 1 + X_i + \Phi^{-1}(\tau)$. Model 2 is similar to the first one, the setting here is model with mixture error. Where $X_i \overset{i.i.d}{\sim} N(0,1)$, $\pi_i \overset{i.i.d}{\sim} Unif(0,1)$, $\epsilon_{1i} \overset{i.i.d}{\sim} N(0,1) \perp\!\!\!\perp \epsilon_{2i} \overset{i.i.d}{\sim} N(3,3)$. Since $Y_i \sim N(1+X_i, \pi_i^2 + 3(1-\pi_i)^2)$, $Q_\tau = 1 + X_i + 3(1-\pi_i) + (3-2\pi_i)\Phi^{-1}(\tau)$. Model 3 is generated with heteroscedastic error, where $X_i \overset{i.i.d}{\sim} Unif(-1,1)$, $\epsilon_i \overset{i.i.d}{\sim} N(0,1)$. True linear quantile can be obtained according to $Y_i \sim N(1+X_i, (1.1-X_i)^2)$,

$Q_\tau = 1 + X_i + (1.1 - X_i)\Phi^{-1}(\tau)$. Model 4 and model 5 are generated under the same frame of non-linear quantile considered in Bondell et al. (2010),

$$\begin{aligned} Y_i &= f(X_i) + g(X_i)\epsilon_i \\ \text{Model 4} \qquad f(X) &= 0.5 + 2X + \sin(2\pi X - 0.5) \qquad g(X) = 1 \\ \text{Model 5} \qquad f(X) &= 3X \qquad g(X) = 0.5 + 2X + \sin(2\pi X - 0.5), \end{aligned}$$

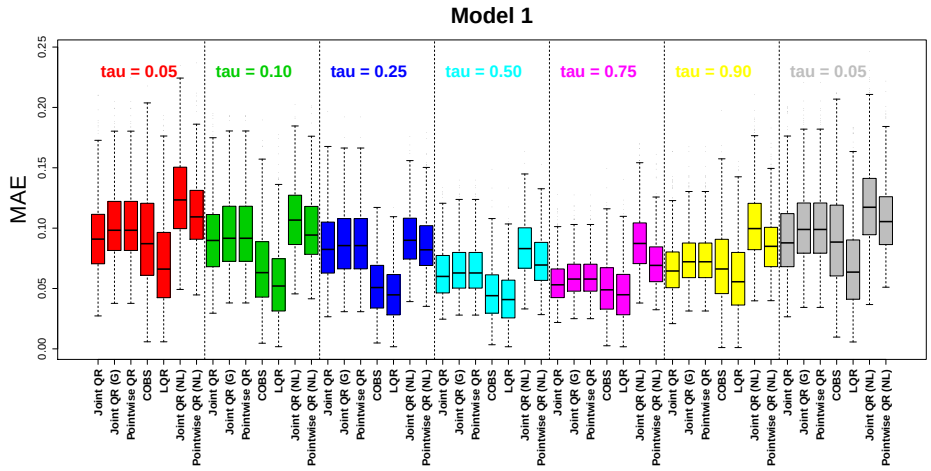
where $X_i \overset{i.i.d}{\sim} Unif(-1, 1)$, $\epsilon_i \overset{i.i.d}{\sim} N(0, 1)$. As in model 4, $Y_i \sim N(f(X_i), 1)$, $Q_\tau = 0.5 + 2X_i + \sin(2\pi X_i - 0.5) + \Phi^{-1}(\tau)$; for model 5 with heteroscedastic error, $Y_i \sim N(3X_i, g(X_i)^2)$, $Q_\tau = 3X_i + (0.5 + 2X_i + \sin(2\pi X_i - 0.5))\Phi^{-1}(\tau)$. All covariates and error terms appearing in the models are mutually independent. The performance of each method is evaluated in terms of MAE again on seven quantile levels of interests ranging from 0.05 to 0.95.

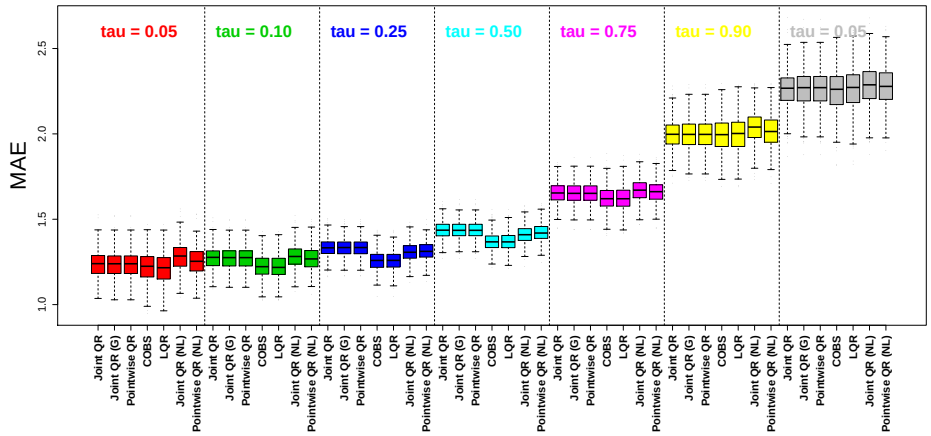
We compare the proposed method with several alternatives, pointwise QR, LQR as introduced before, COBS (Constrained B-Spline Smoothing) implemented by `cobs` from `COBS` package (Ng and Maechler, 2007) in `R` and a variant of our approach by ignoring the binary-valued nature of the functional response, and thus using identity link function (Joint QR (G): `pffr`, Gaussian). Moreover we also consider the proposed methods (Pointwise QR and Joint QR) with non-linear modeling of the conditional distribution, i.e. $F_{Y|X}(y) = g^{-1}\big\{\beta_0(y) + h(X)\big\}$, where $\beta_0(\cdot)$ and $h(\cdot)$ are unknown smooth functions; the methods are denoted by Pointwise QR (NL) and Joint QR (NL). Note that `pfr` cannot only take scalar covariates as input directly, thus we actually implement a generalized linear model.

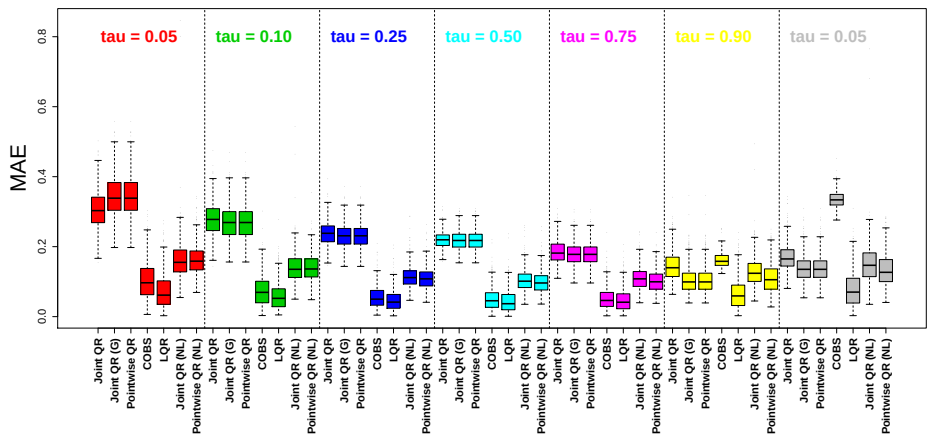
As before, set sample size $n = 1000$ as a training set, and use the additional 100 as a testing set. Totally 500 Monte Carlo samples are generated. Results of our numerical study are presented as in Table C.1.

Table C.1: Average MAE and standard error (in parentheses) of the predicted τ-level quantile for the case of only having a scalar covariate based on 500 replications.

Model	Method	$\tau = 0.05$	$\tau = 0.1$	$\tau = 0.25$	$\tau = 0.5$	$\tau = 0.75$	$\tau = 0.9$	$\tau = 0.95$
	Joint QR	0.09 (0.14)	0.09 (0.15)	0.09 (0.13)	0.06 (0.10)	0.06 (0.08)	0.07 (0.10)	0.09 (0.15)
	Joint QR (G)	0.38 (0.26)	0.34 (0.19)	0.23 (0.14)	0.15 (0.13)	0.29 (0.14)	0.40 (0.19)	0.44 (0.25)
	Pointwise QR	0.10 (0.14)	0.10 (0.14)	0.09 (0.13)	0.07 (0.10)	0.06 (0.08)	0.08 (0.10)	0.10 (0.14)
Model 1	COBS	0.09 (0.19)	0.07 (0.15)	**0.05** (0.11)	0.05 (0.11)	**0.05** (0.12)	0.07 (0.15)	0.09 (0.19)
	LQR	**0.07** (0.17)	**0.06** (0.14)	**0.05** (0.10)	**0.04** (0.10)	**0.05** (0.11)	**0.06** (0.14)	**0.07** (0.16)
	Joint QR (NL)	0.13 (0.16)	0.11 (0.13)	0.09 (0.11)	0.08 (0.10)	0.09 (0.12)	0.10 (0.13)	0.12 (0.16)
	Pointwise QR (NL)	0.11 (0.14)	0.10 (0.13)	0.09 (0.11)	0.07 (0.10)	0.07 (0.10)	0.09 (0.11)	0.11 (0.13)
	Joint QR	1.24 (0.35)	1.27 (0.29)	1.33 (0.23)	1.44 (0.22)	1.65 (0.28)	**2.00** (0.40)	2.26 (0.51)
	Joint QR (G)	**0.94** (0.43)	**0.98** (0.34)	**1.17** (0.26)	1.48 (0.23)	1.83 (0.28)	2.14 (0.40)	2.43 (0.58)
	Pointwise QR	1.24 (0.35)	1.27 (0.29)	1.33 (0.24)	1.44 (0.23)	1.65 (0.28)	**2.00** (0.40)	2.27 (0.51)
Model 2	COBS	1.22 (0.41)	1.22 (0.33)	1.26 (0.26)	**1.37** (0.25)	**1.62** (0.33)	**2.00** (0.48)	**2.25** (0.60)
	LQR	1.21 (0.40)	1.22 (0.33)	1.26 (0.25)	**1.37** (0.24)	**1.62** (0.32)	**2.00** (0.46)	2.26 (0.57)
	Joint QR (NL)	1.28 (0.36)	1.28 (0.29)	1.31 (0.25)	1.41 (0.24)	1.67 (0.30)	2.04 (0.42)	2.29 (0.53)
	Pointwise QR (NL)	1.26 (0.37)	1.27 (0.30)	1.31 (0.24)	1.42 (0.23)	1.66 (0.29)	2.01 (0.42)	2.27 (0.52)
	Joint QR	0.30 (0.25)	0.28 (0.22)	0.24 (0.15)	0.22 (0.11)	0.19 (0.15)	0.15 (0.19)	0.17 (0.17)
	Joint QR (G)	0.68 (0.71)	0.41 (0.25)	0.15 (0.13)	0.20 (0.14)	0.16 (0.11)	0.31 (0.23)	0.49 (0.75)
	Pointwise QR	0.34 (0.27)	0.27 (0.22)	0.23 (0.15)	0.22 (0.12)	0.18 (0.14)	0.11 (0.17)	0.14 (0.17)
Model 3	COBS	0.10 (0.26)	0.08 (0.21)	0.06 (0.15)	0.05 (0.15)	**0.05** (0.15)	0.16 (0.10)	0.34 (0.11)
	LQR	**0.07** (0.23)	**0.06** (0.18)	**0.05** (0.14)	**0.04** (0.14)	**0.05** (0.15)	**0.07** (0.19)	**0.08** (0.23)
	Joint QR (NL)	0.17 (0.20)	0.14 (0.18)	0.11 (0.14)	0.10 (0.14)	0.10 (0.15)	0.11 (0.19)	0.13 (0.21)
	Pointwise QR (NL)	0.16 (0.28)	0.14 (0.20)	0.11 (0.15)	0.10 (0.14)	0.11 (0.15)	0.13 (0.20)	0.15 (0.26)
	Joint QR	0.61 (0.22)	0.60 (0.18)	0.59 (0.15)	0.58 (0.14)	0.58 (0.15)	0.60 (0.20)	0.64 (0.26)
	Joint QR (G)	0.73 (0.34)	0.65 (0.22)	0.58 (0.17)	0.55 (0.14)	0.59 (0.19)	0.64 (0.25)	0.69 (0.33)
	Pointwise QR	0.62 (0.23)	0.60 (0.19)	0.58 (0.15)	0.58 (0.14)	0.58 (0.15)	0.60 (0.20)	0.65 (0.27)
Model 4	COBS	0.35 (0.17)	0.34 (0.15)	0.32 (0.13)	0.32 (0.12)	0.32 (0.12)	0.33 (0.15)	0.34 (0.16)
	LQR	0.62 (0.21)	0.60 (0.19)	0.58 (0.16)	0.58 (0.14)	0.60 (0.16)	0.63 (0.19)	0.65 (0.22)
	Joint QR (NL)	**0.14** (0.15)	**0.12** (0.13)	**0.11** (0.11)	**0.10** (0.10)	**0.10** (0.11)	**0.12** (0.13)	**0.14** (0.16)
	Pointwise QR (NL)	0.18 (0.16)	0.15 (0.13)	0.12 (0.11)	0.11 (0.10)	0.11 (0.11)	0.14 (0.13)	0.17 (0.17)
	Joint QR	0.74 (0.34)	0.57 (0.25)	0.34 (0.19)	0.22 (0.13)	0.30 (0.13)	0.55 (0.20)	0.78 (0.29)
	Joint QR (G)	1.11 (0.93)	0.81 (0.33)	0.57 (0.21)	0.27 (0.14)	0.60 (0.23)	1.07 (0.42)	1.52 (0.63)
	Pointwise QR	0.76 (0.34)	0.56 (0.24)	0.33 (0.19)	0.22 (0.14)	0.30 (0.14)	0.54 (0.21)	0.79 (0.30)
Model 5	COBS	0.38 (0.23)	0.29 (0.19)	**0.16** (0.17)	**0.03** (0.11)	**0.16** (0.19)	0.24 (0.22)	0.33 (0.24)
	LQR	0.84 (0.30)	0.59 (0.21)	0.30 (0.13)	**0.03** (0.09)	0.30 (0.13)	0.59 (0.20)	0.84 (0.29)
	Joint QR (NL)	0.34 (0.22)	0.27 (0.18)	0.17 (0.14)	0.14 (0.15)	0.17 (0.16)	0.24 (0.21)	0.30 (0.29)
	Pointwise QR (NL)	**0.32** (0.24)	**0.25** (0.21)	0.18 (0.17)	0.15 (0.14)	**0.16** (0.15)	**0.22** (0.18)	**0.28** (0.23)


Model 1
MAE
tau = 0.05
tau = 0.10
tau = 0.25
tau = 0.50
tau = 0.75
tau = 0.90
tau = 0.05
Joint QR
Joint QR (G)
Pointwise QR
COBS
LQR
Joint QR (NL)
Pointwise QR (NL)


Model 2
MAE
tau = 0.05
tau = 0.10
tau = 0.25
tau = 0.50
tau = 0.75
tau = 0.90
tau = 0.05
Joint QR
Joint QR (G)
Pointwise QR
COBS
LQR
Joint QR (NL)
Pointwise QR (NL)


Model 3
MAE
tau = 0.05
tau = 0.10
tau = 0.25
tau = 0.50
tau = 0.75
tau = 0.90
tau = 0.05
Joint QR
Joint QR (G)
Pointwise QR
COBS
LQR
Joint QR (NL)
Pointwise QR (NL)

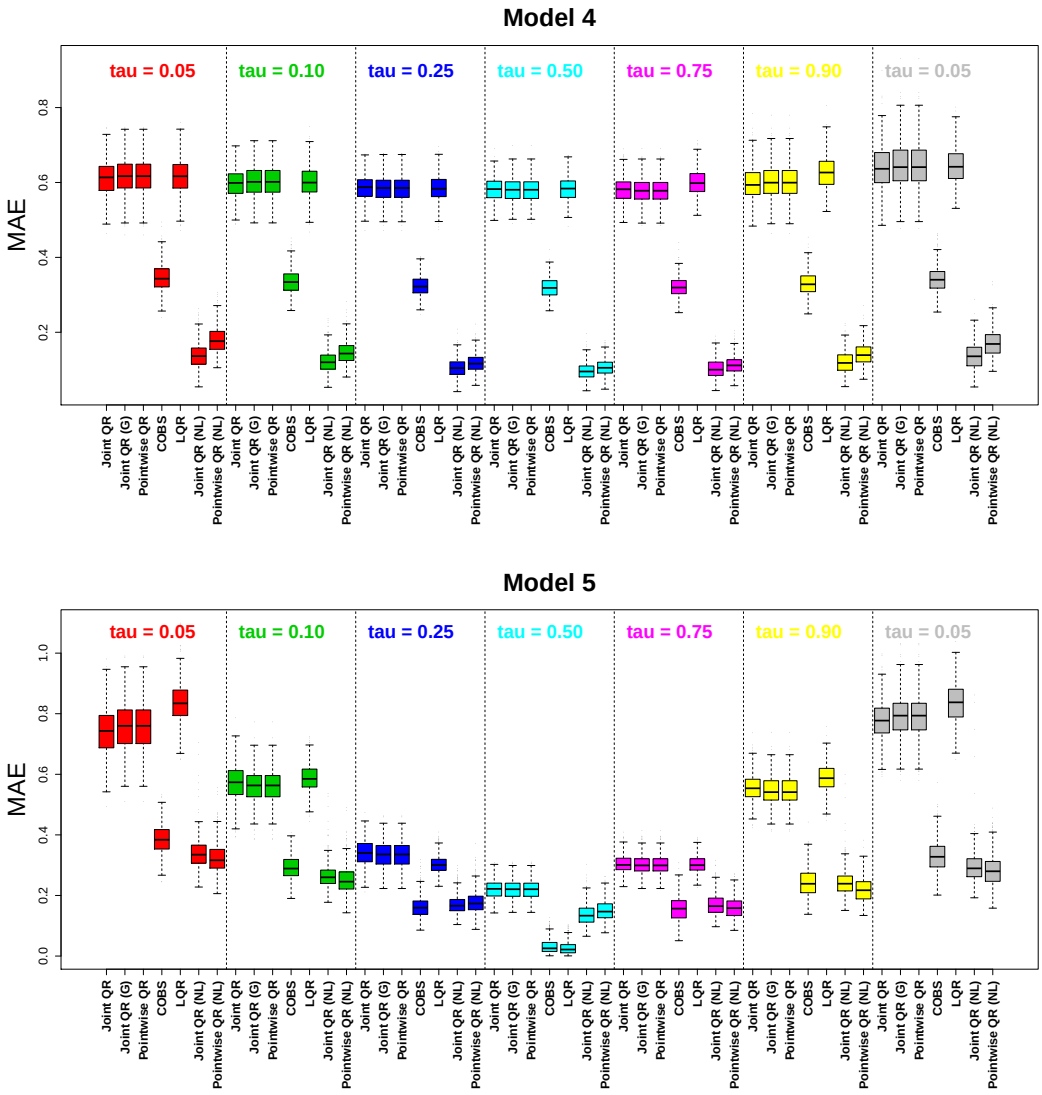


Figure C.1: Boxplots of MAEs of the predicted τ-level quantile for sample size $n = 1000$ for the case of having a scalar covariate only. Results are based on 500 replication.

C.1.2 Simulation Study for Case of Functional Covariate

Consider when there is only a single functional covariate. The observed data for the ith subject is $\{Y_i, (W_{i1}, t_{i1}), \cdots, (W_{im_i}, t_{im_i})\}$. For brevity, we omit the same settings described in simulation 1, see Section 3.3 for details.

We consider two scenarios as before:

(i) normal distribution $Y_i|X_i(\cdot) \sim N(2\sum_{k=1}^{4}\xi_{ik}, 5^2)$;

(ii) mixture distribution $Y_i|X_i(\cdot) \sim 0.5N(\sum_{k=1}^{4}\xi_{ik}, 1^2) + 0.5N(3\sum_{k=1}^{4}\xi_{ik}, 4^2)$.

Sample size and replication number are set as 1000 and 500, respectively. We compare the performance of our method with three alternatives: Pointwise QR, CM and Joint QR (G). The methods are compared as before, and results are presented in Tables C.2 and C.3.

C.2 Additional Application: Bike Sharing Data

In this section we illustrate the proposed method using the bike sharing data (Fanaee-T and Gama, 2013) available at the University of California, Irvine (UCI) Machine Learning Repository (Lichman, 2013). The data set consists of hourly counts of bike rentals in Washington, D.C. recorded by the Capital Bike Sharing (CBS) system from January 1, 2011 to December 31, 2012, with additional information on corresponding users, such as membership (casual vs. registered), and on corresponding days, such as season, temperature, and etc. More detailed descriptions on the data set are provided in Fanaee-T and Gama (2013).

The CBS system is an automated bike sharing system, where users can rent bikes from (and return them to) any bike docks located across Washington, D.C. on an hourly basis. One of main challenges associated with the system is to understand and forecast a daily demand of bike rentals in a supply chain. In most cases, such demand analysis involves studying a quantile of the response instead of its average; for example, a supplier would probably be more interested in covering 90% of a demand than only an average demand. Here we use the proposed method to study the total number

Table C.2: Average MAE and standard error (in parentheses) of the predicted τ-level quantile for the case of only having a functional covariate based on 500 replications.

Distribution	σ_ϵ	Method	$\tau = 0.05$	$\tau = 0.1$	$\tau = 0.25$	$\tau = 0.5$
Normal	0.5	Joint QR	**1.18** (0.01)	1.20 (0.01)	1.18 (0.01)	**1.09** (0.00)
		Joint QR (G)	9.60 (0.04)	7.72 (0.02)	4.85 (0.01)	2.57 (0.01)
		Pointwise QR	1.33 (0.01)	1.30 (0.01)	1.27 (0.01)	1.19 (0.01)
		CM	1.31 (0.01)	**1.17** (0.01)	**1.11** (0.01)	1.14 (0.01)
		PQR	1.57 (0.03)	1.53 (0.02)	1.48 (0.02)	1.47 (0.02)
Normal	4.33	Joint QR	**7.91** (0.03)	**7.04** (0.03)	**6.24** (0.02)	**6.05** (0.02)
		Joint QR (G)	10.95 (0.04)	9.28 (0.03)	7.18 (0.02)	6.30 (0.02)
		Pointwise QR	8.06 (0.03)	7.13 (0.03)	6.28 (0.02)	6.06 (0.02)
		CM	7.99 (0.03)	7.14 (0.03)	6.35 (0.02)	6.15 (0.02)
		PQR	8.20 (0.04)	7.41 (0.04)	6.47 (0.03)	6.08 (0.02)
Normal	6.13	Joint QR	**10.02** (0.04)	**8.83** (0.03)	**7.59** (0.03)	**7.23** (0.02)
		Joint QR (G)	11.80 (0.04)	10.18 (0.04)	8.17 (0.03)	7.34 (0.03)
		Pointwise QR	10.17 (0.04)	8.91 (0.03)	7.62 (0.03)	7.24 (0.03)
		CM	10.10 (0.04)	8.95 (0.03)	7.73 (0.03)	7.35 (0.03)
		PQR	10.29 (0.05)	9.16 (0.04)	7.80 (0.03)	7.25 (0.03)
Mixture	0.5	Joint QR	2.24 (0.01)	1.98 (0.01)	2.63 (0.02)	1.97 (0.02)
		Joint QR (G)	13.60 (0.07)	8.72 (0.04)	4.55 (0.01)	**1.11** (0.01)
		Pointwise QR	**2.12** (0.02)	**1.66** (0.01)	2.48 (0.02)	1.85 (0.02)
		CM	2.99 (0.01)	2.08 (0.01)	**2.25** (0.02)	2.12 (0.02)
		PQR	5.56 (0.05)	4.40 (0.04)	3.21 (0.02)	2.23 (0.03)
Mixture	4.33	Joint QR	**9.61** (0.04)	**7.74** (0.03)	6.32 (0.03)	4.59 (0.02)
		Joint QR (G)	14.45 (0.07)	10.28 (0.04)	7.16 (0.02)	**4.29** (0.01)
		Pointwise QR	9.88 (0.04)	7.77 (0.03)	**6.27** (0.03)	4.53 (0.02)
		CM	9.81 (0.04)	7.90 (0.03)	6.43 (0.03)	4.73 (0.02)
		PQR	11.37 (0.05)	9.15 (0.04)	6.78 (0.02)	4.38 (0.02)
Mixture	6.13	Joint QR	**12.03** (0.04)	**9.60** (0.03)	7.53 (0.03)	5.27 (0.02)
		Joint QR (G)	15.15 (0.07)	11.19 (0.04)	8.12 (0.03)	**5.10** (0.02)
		Pointwise QR	12.39 (0.04)	9.69 (0.03)	**7.51** (0.03)	5.23 (0.02)
		CM	12.30 (0.04)	9.85 (0.03)	7.70 (0.03)	5.43 (0.02)
		PQR	13.51 (0.05)	10.78 (0.04)	7.95 (0.03)	5.14 (0.02)

Table C.3: Comparison of the average computing time (in seconds) for the approaches.

Distribution	Joint QR	Joint QR (G)	Pointwise QR	CM
Normal	179	125	310	477
Mixture	206	134	313	314

of bikes rented by casual users on Saturday (Y), using average of hourly 'feeling' temperatures on Saturday (X_1) and the hourly counts of bike rentals by casual users on the previous day ($X_2(t)$ for hour of a day $t = 1, 2, \ldots, 24$). In this analysis short term weather-related forecasts are assumed to be accurate.

There are total of $n = 104$ Saturdays during the period that the data were collected, with about four to five weekly measurements per each month. Raw responses of total counts of Saturday bike rentals, $\{Y_i : i = 1, 2, \ldots, 104\}$, suggest strong seasonal and year effects; see Figure C.4. To remove these effects prior to the analysis, we fit a linear model to the log-transformed response, $\log(Y + 1)$, using three dummy variables for season and one dummy variable for year and obtain residuals shown in Figure C.5. The resulting residuals are then used as our response variable in the subsequent analysis procedure; henceforth, with abuse of notation, denote the resulting residuals by Y. Alternatively, we also consider fitting a generalized linear model with a Poisson distribution to the original responses. However, as two approaches give a similar conclusion in our analysis, we exclude the results obtained using the second approach.

While the data set includes hourly 'feeling' temperatures on Saturday, we use their average as one of the predictors because a forecast for the average daily temperature is expected to be less variable than that for the hourly temperatures. Specifically we consider the *centered* average daily 'feeling' temperature of Saturday as our scalar covariate in the analysis. Whereas hourly counts of bike rentals by casual users on Fridays are considered in the analysis as noisy functional observations with mild missingness (about 0.35% of the data is missing). Prior to the analysis we pre-smooth the observed functional predictor and center it; by following an approach taken by Goldsmith et al. (2011) and Ivanescu et al. (2015), pre-smoothing is done using the principal analysis by conditional expectation (PACE) method (Yao et al., 2005) with percentage of variance explained (PVE) equal to 0.99. With abuse of notation, we denote the resulting, pre-processed scalar and functional covariates for the ith Saturday with $X_{1,i}$ and $X_{2,i}(t)$, respectively; see Figures C.6 - C.8.

In the following we first consider three different quantile models:

(M1) a model with a scalar covariate X_1 only

(M2) a model with a functional covariate $X_2(t)$ only

(M3) a model with both covariates X_1 and $X_2(t)$

When applicable, we also compare the prediction accuracy of our proposed method with other available methods that are closely related to ours, namely LQR and CM. We quantify prediction accuracy using leave-one-out cross validation (LOO-CV) with the loss function typically used in a quantile regression: $\rho_\tau(q) = (1-\tau)\sum_{y_i<q}(y_i - q) + \tau\sum_{y_i \geq q}(y_i - q)$. Specifically we define out-of-sample prediction error for given τ as OUT-PE$_\tau = n^{-1}\sum_{i=1}^{n}\rho_\tau\left(\widehat{Q}^{[-i]}_{Y|X}(\tau)\right)$, where $\widehat{Q}^{[-i]}_{Y|X}(\tau)$ is the predicted τth quantile obtained using the ith training set; we prepare the ith training set by leaving the ith observations out and then pre-processing the rest as described earlier in this section. Covariates in the testing sets are also appropriately pre-processed. For example, consider the ith testing set; the scalar covariate, $X_{1,i}$, is centered by subtracting the average of $X_{1,i}$'s in the ith training set, whereas the noisy functional covariate, $X_{2,i}(t)$, is smoothed and is centered using the mean and principal component (PC) estimates obtained using $X_{2,i}(t)$'s in the ith training set.

In terms of prediction accuracy we compare the proposed method (Joint QR) with Pointwise QR for all of the three models (M1) - (M3) considered. Additionally LQR and CM are considered for the models (M1) and (M2), respectively. OUT-PE$_\tau$ is obtained for $\tau = 0.1, 0.2, \ldots, 0.9$ and the results are summarized in Figure C.2. The results are consistent with the simulation results given in Sections 3.3 and C.1. As shown in Figure C.2, Pointwise QR and Joint QR have very similar prediction accuracy for all three models (M1)-(M3). When considering only a functional covariate, prediction accuracy of the CM method also lies in the same range as Pointwise QR and Joint QR; this is consistent with the simulation results given in Section C.1.2. When considering a scalar covariate only, the LQR estimation method has substantially higher OUT-PE$_\tau$ than Pointwise QR and Joint QR for all τ's. It implies that the relationship between quantiles of the total number of bike rentals of Saturday and average daily 'feeling' temperature on Saturday may not be linear. In summary for models (M1) and (M2), the propose method provides as reliable prediction as other

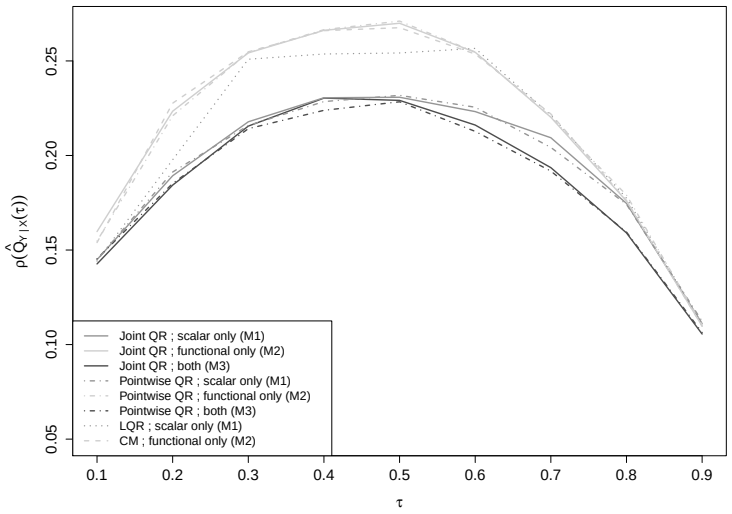
existing methods and there is no obvious estimation method that outperforms the others.

However the existing methods are not applicable for model (M3), when there is both scalar and functional covariates X_1 and $X_2(t)$. One of advantages of the proposed methods (Joint and Pointwise QR) is that now we can compare among models (M1) - (M3) and investigate which covariates have more effects on the conditional distribution of the response, and consequently on the quantiles. For example, based on OUT-PE$_\tau$'s for Joint QR for (M1)-(M3) shown in Figure C.9 we can see that average daily 'feeling' temperature on Saturday (X_1) provides more information on the total number of bike rental demands on Saturday (Y) than hourly counts of bike rentals on Friday ($X_2(t)$) does for all quantile levels considered. It seems that incorporating a functional covariate $X_2(t)$ in addition to a scalar covariate, X_1, does not have any effect on prediction accuracy for low quantile levels ($\tau < 0.5$). However, a model that incorporates both covariates, X_1 and $X_2(t)$, has slightly better prediction accuracy than a model with only scalar covariate, X_1, for higher quantile levels ($\tau > 0.5$).

As we are interested in studying demand of bike rentals, predicting quantiles at *high* quantile levels are particularly of interest in this analysis. Thus based on prediction accuracy of different models and estimation methods for $\tau > 0.5$, we choose a model that incorporates covariates of both types (X_1 and $X_2(t)$): $\mathbb{E}\left[I(Y < y)|X_1, X_2(t)\right] = \beta_0 + \beta_{X_1}(y)X_1 + \int X_2(t)\beta_{X_2}(t, y)dt$ and use the proposed method for estimation. The estimated coefficients, $\widehat{\beta}_{X_1}(y)$ and $\widehat{\beta}_{X_2}(t, y)$, are given in Figure C.10. As in the analysis of the sow data in Section 3.4, we investigate the relationship between covariates and quantiles of the response (i.e. bike rental demands) for different quantile levels by studying predicted quantiles. While fixing $X_2(t)$ equal to the pointwise average of hourly Friday bike rentals, we predict quantiles of total demands of Saturday's bike rentals for fine grids of average feeling temperatures (X_1); again $\tau = 0.1, 0.2, \ldots, 0.9$ are considered. The predicted quantiles, $\widehat{Q}_{Y|X}(\tau)$, are given in Figure C.3.

First, it is clear from the predicted quantiles curves that the relationship between average feeling temperature and quantile of Y is positive and nonlinear for all τ's we consider. This is probably why prediction accuracy of LQR is noticeably inferior to the proposed approach when

Figure C.2: Out-of-sample prediction error, $\rho_\tau\left(\widehat{Q}_{Y|X}(\tau)\right)$ for three models (M1) - (M3) and four different methods (Joint QR, Pointwise QR, CM, LQR)



only scalar covariate (X_1) in earlier comparison. Secondly, gaps between curves are wider for higher τ's imply that the density function of total bike rental demands on Saturday is right skewed; while there is implication of right-skewness across all values of average feeling temperature, we notice particularly large right-skewness around -3°C average feeling temperature and a steep increase in 90% quantiles between -5°C to 0°C. Specifically, the predicted 90% quantiles, $\widehat{Q}_{Y|X}(0.9)$, for -5°C and 0°C average Saturday feeling temperatures are about -0.5 and 0.38 respectively; the result suggest that, with increase in average Saturday feeling temperature from -5°C to 0°C, we expect 90% quantile of Saturday bike rentals demands to increase by more than twice (from 168 to 407) for winter of 2012.

Figure C.3: Predicted quantiles against average Saturday feeling temperatures for $X_2(t)$ equal to pointwise average of hourly Friday bike rentals

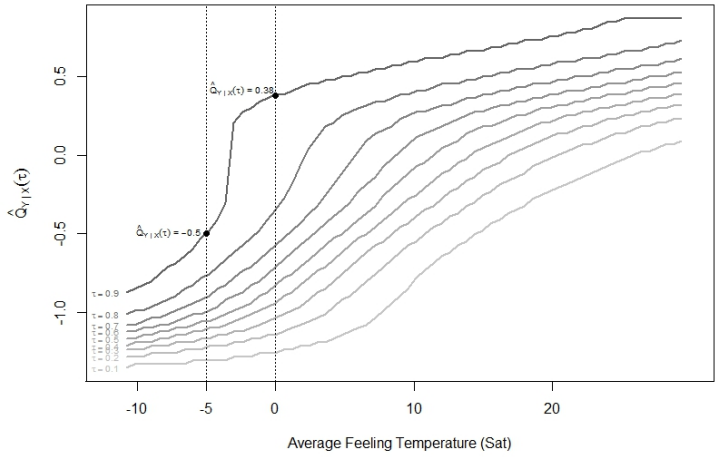


Figure C.4: Total counts of bike rentals on Saturday, Y_i

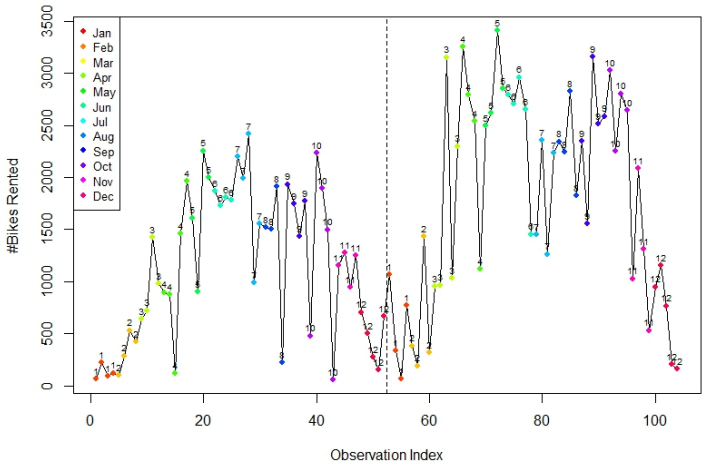


Figure C.5: Pre-processed response; transformed responses, $\log(1 + Y)$ (top) and transformed responses without season and year effects (bottom)

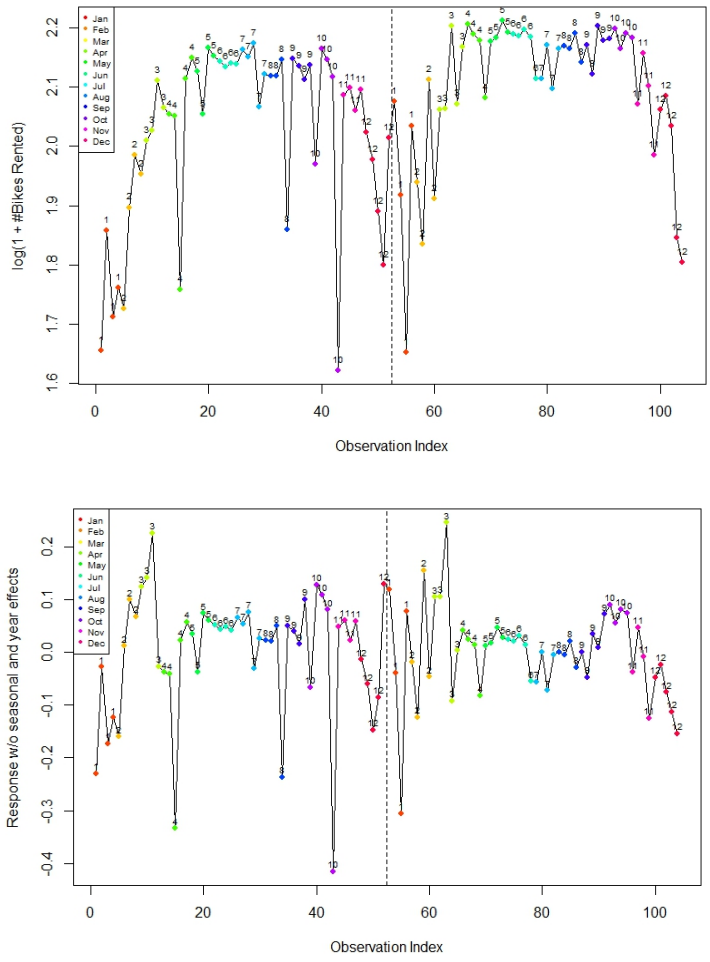


Figure C.6: Average feeling temperatures on Saturday

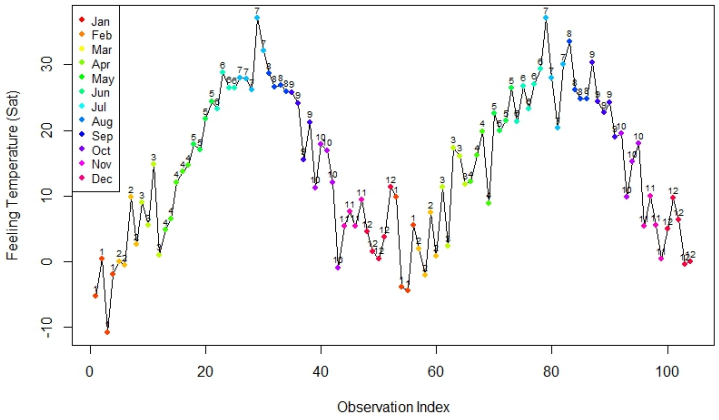


Figure C.7: Spaghetti plots of hourly number of bike rentals on Friday; observed (top) and smoothed (bottomed)

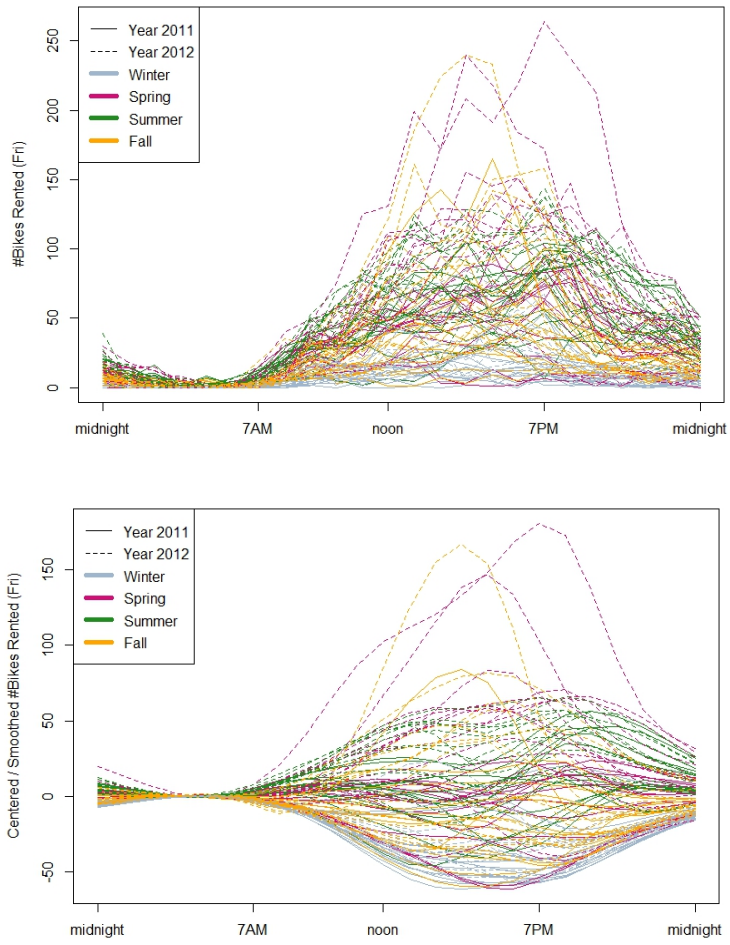


Figure C.8: Lasagna plots of hourly number of bike rentals on Friday; observed (top) and smoothed (bottomed)

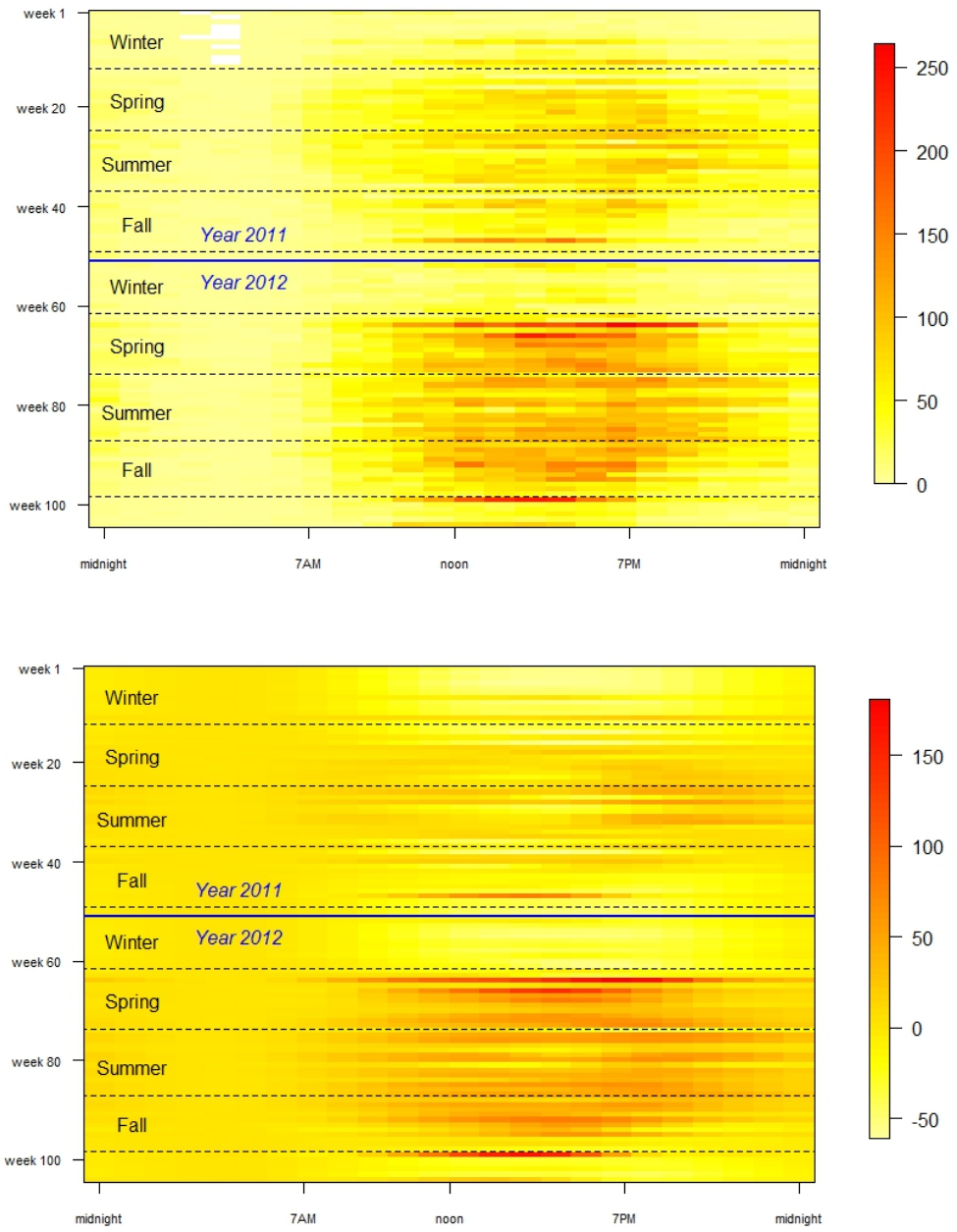


Figure C.9: Out-of-sample prediction error, $\rho_\tau\left(\widehat{Q}_{Y|X}(\tau)\right)$, when the proposed estimation method (Joint QR) is used

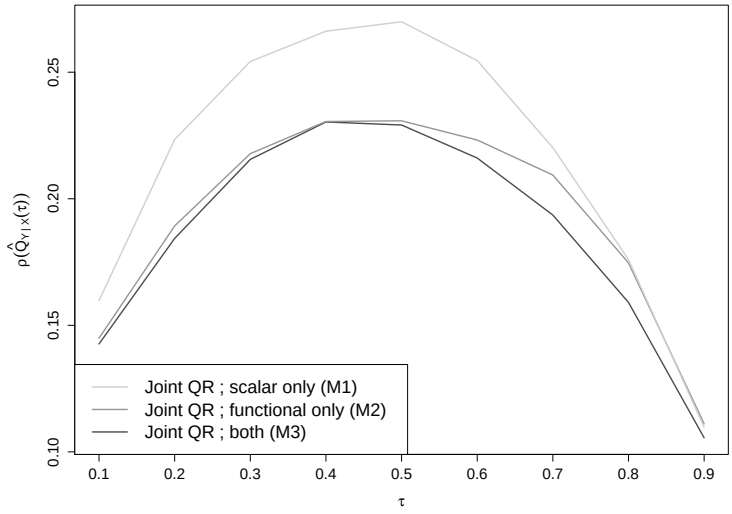
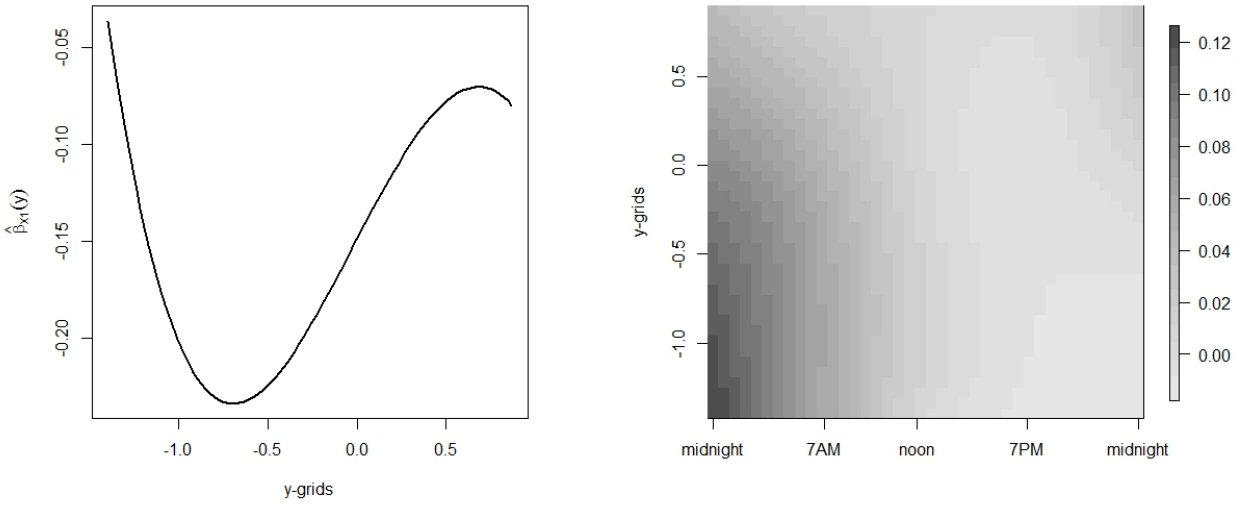


Figure C.10: Estimated coefficients, $\widehat{\beta}_{X_1}(y)$ and $\widehat{\beta}_{X_2}(y,t)$



Appendix D

Supplement for Chapter 5

D.1 Proof of Theorem 5

Proof. Note that $\mathcal{M}_{\mathbf{B}}(\mathbf{t}) = \text{tr}(\mathbf{B}\mathbf{\Lambda}) - \text{tr}\left\{\mathbf{\Lambda}\mathbf{B}\mathbf{\Lambda}\mathbf{\Phi}(\mathbf{t})'\mathbf{\Sigma}^{-1}\mathbf{\Phi}(\mathbf{t})\right\}$ with $\mathbf{\Sigma} = r(\mathbf{t},\mathbf{t}) + \sigma_\epsilon^2\mathbf{I}_p$ and $r(\mathbf{t},\mathbf{t}) = \mathbf{\Phi}(\mathbf{t})\mathbf{\Lambda}\mathbf{\Phi}(\mathbf{t})'$. Thus,

$$\frac{\partial\mathcal{M}(\mathbf{t})}{\partial t_k} = -2\text{tr}\left\{\widetilde{\mathbf{B}}\frac{\partial\mathbf{\Phi}(\mathbf{t})'}{\partial t_k}\mathbf{\Sigma}^{-1}\mathbf{\Phi}(\mathbf{t})\right\} + 2\text{tr}\left\{\widetilde{\mathbf{B}}\mathbf{\Phi}(\mathbf{t})'\mathbf{\Sigma}^{-1}\mathbf{\Phi}(\mathbf{t})\mathbf{\Lambda}\frac{\partial\mathbf{\Phi}(\mathbf{t})'}{\partial t_k}\mathbf{\Sigma}^{-1}\mathbf{\Phi}(\mathbf{t})\right\},$$

and

$$\frac{\partial\mathbf{\Phi}(\mathbf{t})'}{\partial t_k} = \dot{\phi}(t_k)\otimes\mathbf{e}_k',$$

where $\dot{\phi}(t_k) = \{\dot{\phi}_1(t_k), \dot{\phi}_2(t_k), \cdots\}'$. Specifically, $\dot{\phi}_\ell(t_k)$ can be obtained through the following identity:

$$\dot{\phi}_\ell(t_k) = \frac{1}{\lambda_\ell}\int \dot{r}(t_k,s)\phi_\ell(s)ds,$$

where $\dot{r}(t_k,s)$ denotes the derivative of $r(t_k,s)$ with respect to t_k and can be obtained according to Lemma 6 in Appendix B. Then simple algebra proves Theorem 5. □