Bayesian Analysis of Historical Functional Linear Models with application to air pollution forecasting

by

Yovna Junglee

Minor dissertation presented in partial fulfilment of the requirements for the degree of Master of Science in Advanced Analytics

in the

Department of Statistical Sciences
University of Cape Town

Supervised by

Dr Birgit Erni & Mr Allan Clark

March 2022

Abstract

Historical functional linear models are used to analyse the relationship between a functional response and a functional predictor whereby only the past of the predictor process can affect the current outcome. In this work, we develop a Bayesian framework for the analysis of the historical functional linear model with multiple predictors.

Different from existing Bayesian approaches to historical functional linear models, our proposed methodology is able to handle multiple functional covariates with measurement error and sparseness. The proposed model utilises the well-established connection between non-parametric smoothing and Bayesian methods to reduce sensitivity to the number of basis functions which are used to model the functional regression coefficients. We investigate two methods of estimation within the Bayesian framework. We first propose to smooth the functional predictors independently from the regression model in a two-stage analysis, and secondly, jointly with the regression model. The efficiency of the MCMC algorithms is increased by implementing a Cholesky decomposition to sample from high-dimensional Gaussian distributions and by taking advantage of the orthogonal properties of the functional principal components used to model the functional covariates. Our extensive simulation study shows substantial improvements in both the recovery of the functional regression surface and the true underlying functional response with higher coverage probabilities, when compared to a classical model under which the measurement error is unaccounted for. We further found that the Bayesian two-stage analysis outperforms the joint model under certain conditions.

A major challenge with the collection of environmental data is that they are prone to measurement error, both random and systematic. Hence, our methodology provides a reliable functional data analytic framework for modelling environmental data. Our focus is on the application of our method to forecast the level of daily atmospheric pollutants using meteorological information such as hourly records of temperature, humidity and wind speed from data collected by the City of Cape Town, South Africa. The forecasts provided by the proposed Bayesian two-stage model are highly competitive against the functional autoregressive models which are traditionally used for functional time series.

***K*eywords** Functional regression, Bayesian inference, measurement error, air pollution

Acknowledgements

I am extremely grateful to my supervisors and mentors, Dr Birgit Erni and and Mr Allan Clark for their invaluable advice, continuous support and patience during the past two years. Mentors like you are what make science and academia wonderful! And thank you to Allan Clark for sparking my interest in Bayesian statistics and for your words of encouragement since Honours.

Thank you to my friends for an incredible and memorable time in South Africa.

Thanks to Mr Haithum Wingrove from the City of Cape Town for kindly assisting with the provision of the data used in this dissertation.

Declaration of Authorship

I, Yovna Junglee, declare that this thesis and the work presented in it are my own and has been generated by me as the result of my own original research.

I confirm that:

1. This work was done wholly or mainly while in candidature for a degree at this University;
2. Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
3. Where I have consulted the published work of others, this is always clearly attributed;
4. Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
5. I have acknowledged all main sources of help;
6. Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
7. Either none of this work has been published before submission, or parts of this work have been published as: [please list references below]:

Signed:

Date:

Contents

List of Figures

List of Tables

List of Abbreviations

ACF	Autocorrelation function.
eSNR	Expected signal-to-noise ratio.
FAR(a)	Functional autoregressive model of order a.
FARX(a)	Functional autoregressive model with exogenous variables of order a.
FDA	Functional data analysis.
FLM	Functional linear model.
FPACF	Functional partial autocorrelation function.
FPC	Functional principal components.
FPCA	Functional principal components analysis.
HFLM	Historical functional linear model.
IMAE(.)	Integrated mean absolute error.
IMSE(.)	Integrated mean-squared error.
M-H	Metropolis-Hastings.
MCMC	Markov chain Monte Carlo.
REML	Restricted maximum likelihood.

CHAPTER 1

Introduction

Functional data analysis (FDA) is an emerging area of research in statistics which deals with the statistical analysis of infinite-dimensional data. Traditional statistical methodologies for multivariate data cannot be easily extended to functional data due to the high correlation between the measurements and because they inherently exist on an infinite dimension (Ferraty and Vieu, 2006). Recently, there has been an overwhelming development of methods for regression, classification, clustering and dimension reduction amongst others that account for the functional structure of the data (Wang et al., 2016). Applications to medical data have been extensively covered in the literature for functional data analysis, but to a much lesser extent, to environmental problems. With the advance in sensing technology, environmental data are being collected at increasingly high temporal and spatial resolutions. In most cases, this complex data can be considered functional where the observations consist of daily curves discretely sampled over time.

In this dissertation, we develop a Bayesian framework for the analysis of the historical functional linear model (HFLM) with multiple predictors. The HFLM is used to analyse the relationship between a functional response and functional predictors whereby only the past or recent past of the predictor process can affect the current outcome. Such a model would be useful in many areas of application, not only environmental but the methodological extension was motivated by an environmental application.

More specifically, this research focuses on forecasting levels of atmospheric pollutants using meteorological information such as temperature, wind speed, and relative humidity. Wichmann and Voyi (2012) previously showed that there exists significant daily respiratory, cardiovascular and cerebrovascular mortality risk linked to air pollution exposure in a study conducted in Cape Town, South Africa. This risk is higher than that assessed in developed countries (Wichmann and Voyi, 2012). The provision of real-time forecasts of air pollution levels can be especially helpful for susceptible individuals, enabling them to take necessary precautions in the event of increased pollution. Several methods have been explored in literature to forecast atmospheric pollutants including time series analysis (Salcedo et al., 1999; Pisoni et al., 2009), neural networks (Niska et al., 2004; Ibarra-Berastegi et al., 2008), generalized additive models (Davis and Speckman, 1999). More closely related to our work, Damon and Guillas (2002) proposed a functional autoregressive Hilbertian model with exogeneous variables (FARX) for ozone forecasting using hourly temperature and wind speed measurements.

As atmospheric pollutants will only depend on meteorological data up to the current time, historical functional linear models – a class of constrained fully functional linear models, are of interest. However, a major challenge with the collection of environmental data is that they are prone to measurement error, both random and systematic (Gilbert, 1987). The presence of measurement error in the predictor process has been largely unaccounted for in the literature for Bayesian HFLMs. Consequently, the impact of measurement error in the functional predictors on statistical inference and uncertainty quantification remains unaddressed to the best of our knowledge.

Different from existing Bayesian approaches to HFLMs, our proposed methodology is able to handle multiple functional covariates with measurement error and sparseness. Since data are collected at discrete time points $\{\tau_1, \ldots, \tau_J\}$, sparseness in the context of FDA refers to the spacing between τ_j and τ_{j+1}, hence the larger J is, the denser the functional data (Wang et al., 2016). To illustrate the importance of our method, we investigate the effect of incorporating the smoothing strategy for the functional predictor process on model estimation and inference using a simulation study. As historical functional linear models are heavily parameterised, especially when fitted using multiple predictors, we incorporate recent developments in fast sampling strategies to formulate an efficient MCMC estimation procedure.

This dissertation is organised as follows:

- In Chapter 2, we present a review of the literature on the functional data analytic methods which provide the foundation to this dissertation, and their Bayesian counterparts. We also discuss recent developments in functional time-series models.
- Chapter 3 focusses on the development of the Bayesian historical functional linear model. We begin by elaborating on the current gaps in the literature which led to the development of our proposed method. We motivate our choice of prior distributions for the parameters of our model, and describe the Bayesian smoothing strategies for the functional predictor process estimated within our MCMC algorithms. Finally, we present the techniques used for Bayesian prediction, model inference and residual diagnostics for functional linear models.
- A simulation study is presented in Chapter 4. The purpose of the simulation study is to investigate whether smoothing the functional predictor process improves model estimation and inferences. We present our results for a two-predictor model, varying the noise levels and the number of functional data.
- Finally, we apply our proposed methods to the air pollution data set collected by

the City of Cape Town, South Africa in Chapter 5. We benchmark our forecasts against the functional autoregressive model with exogeneous variables of order 1 (FARX(1)) and the Hyndman-Ullah FAR(1) model.

CHAPTER 2

Literature Review

In this chapter, we first introduce the area of functional data analysis (FDA) and the mathematical underpinnings behind this concept. Since FDA is an evolving field of statistics, we only selectively review the methods that relate most closely to our historical functional linear model. As an emphasis of this dissertation is on the effect of measurement error in the functional predictor process, we focus on functional linear models with functional predictors and their estimation procedures. We also extensively review the Bayesian alternatives of these models. An overview of Bayesian inference including MCMC estimation procedures and assessing convergence can be found in Appendix A. Bayesian formulations of functional linear models provide a flexible framework to 1) estimate smoothing parameters without the need to perform cross-validation over a grid of hyperparameter values and 2) account for measurement error in the predictors, amongst others. Finally, we end this chapter by reviewing functional time-series models.

2.1 Introduction to Functional Data Analysis

In FDA, a data point from a random variable which can be observed at several different times in the range $(\tau_{\min}, \tau_{\max})$ is considered as an observation of the continuous family $\boldsymbol{\chi} = \{X(\tau);\ \tau \in (\tau_{\min}, \tau_{\max})\}$ (Ferraty and Vieu, 2006). More formally, the following definitions of a functional variable and data have been established,

Definition 2.1.1 ((Ferraty and Vieu, 2006))**.** *A random variable $\mathcal{X}$ is called a functional variable if it takes values in an infinite-dimensional space. An observation χ of the random variable $\mathcal{X}$ is known as a functional data.*

Functional data are considered as realisations of an underlying latent stochastic process as the data can only be observed discretely over time on a dense or sparse, fixed or random grid. The latent stochastic process is assumed to exist in a Hilbert space such as an L^2 process which is square integrable and thus satisfies the condition, $\mathbb{E}(\int_{\mathcal{I}} X^2(\tau)dt) < \infty$ (Wang et al., 2016). An example of a functional data set is illustrated in Figure (2.1).

To handle functional data, many tools such as spline smoothing (Ramsay and Silverman, 2005; Kokoszka and Reimherr, 2017) and kernel methods (Ferraty and Vieu, 2006) are commonly used. This is because these non- and semi-parametric methods offer large degrees of flexibility and are able to preserve the information in this infinite-dimensional data (Wang et al., 2016).

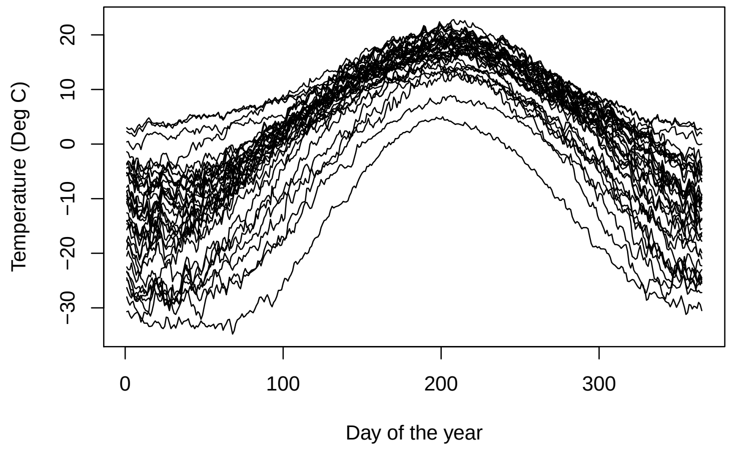


Figure 2.1. *An example of a functional data set using the Canadian temperature data (Ramsay et al., 2020). Each curve consists of daily averages of temperature over a year. The curves represent yearly profiles of temperature across different stations.*

While there have been important methodological and theoretical developments for the classification, clustering and other exploratory tools of functional data, functional regression remains one of the most active area of research in this field. The approach to functional regression depends on whether the response or covariates, or both are functional. Table 2.1 below summarises the three main categories of functional regression models as described by Ramsay and Silverman (2005) where L^2 denotes the space of square-integrable functions on a suitable domain as defined earlier by Wang et al. (2016), and $\mathbb{R}^d$ represents a $d-$dimensional real-valued random vector.

Model	Response	Predictor	Formulation
Scalar-on-function	$\mathbb{R}^d$	L^2	$Y_i = \mu + \int_{\mathcal{S}} \beta(s) X_i(s) ds + e_i$
Function-on-scalar	L^2	$\mathbb{R}^d$	$Y_i(\tau) = \mu(\tau) + \beta X_i + e_i(\tau)$
Function-on-function	L^2	L^2	$Y_i(\tau) = \mu(\tau) + \int_{\mathcal{S}} \beta(s,\tau) X_i(s) ds + e_i(\tau)$

Table 2.1. *Classification of functional regression models.*

For the fully functional, function-on-function linear model, where $\mathcal{S} = \mathcal{T}$, we allow the influence of the functional predictor $X_i(s)$ on $Y_i(\tau)$ to extend over its whole domain.

This gives rise to a bivariate regression surface $\beta(s,\tau)$ which determines the impact of X at time s on Y at time τ. In many realistic settings however, and especially when both s and τ are time indices, the case $s>\tau$ is not possible (Ramsay and Silverman, 2005). Restrictions on the domain of the functional predictor are hence imposed which gave rise to a sub-class of functional linear models.

First considered by Malfait and Ramsay (2003), the historical functional linear model (HFLM) is given by,

$$Y_i(\tau) = \mu(\tau) + \int_0^{\tau} \beta(s,\tau)X_i(s)ds + e_i(\tau), \quad s \in [0,T], \quad \tau \in [0,T],$$

where $\mu(\tau)$ is the functional intercept term, $\beta(s,\tau)$ is a bivariate functional regression surface, and $e_i(\tau)$ is the functional residual process of the i^{th} observation with $\mathbb{E}(e_i(\tau)) = 0$, $\operatorname{cov}(e_i(\tau), e_i(\tau')) = \sigma_e^2$ where $\tau, \tau' \in [0,T]$ and $\tau = \tau'$. This assumption implies identically and independently distributed (i.i.d) errors within each curve and between the curves. Independency between the curve means that the i^{th} residual process, $\{e_i(\tau) : \tau \in [0,T]\}$, is independent of the j^{th} residual process $\{e_j(\tau) : \tau \in [0,T]\}$ when $i \neq j$. Within curve independency implies that $\operatorname{cov}(e_i(\tau), e_i(\tau')) = 0$ when $\tau \neq \tau' \ \forall\, i$. In this model, only the past trajectory, $s \leq \tau$, of the functional predictor can influence the current outcome implying that $\beta(s,\tau) = 0$ for $s > \tau$. In the lag functional linear model (LFLM), only the recent past can affect the current outcome,

$$Y_i(\tau) = \mu(\tau) + \int_{\tau-\delta}^{\tau} \beta(s,\tau)X_i(s)ds + e_i(\tau), \quad s \in [0,T], \quad \tau \in [\delta,T]. \tag{2.1}$$

In eq. (2.1), δ denotes the time lag where the predictor process starts to influence the outcome. The recent lag FLM is an extension to this model where the limits of the integrand are, $s \in [\tau-\delta_1, \tau-\delta_2]$ where $\delta_1 > \delta_2$ (Kim et al., 2011). Yet another model was proposed, the concurrent functional linear model, similar to the time-varying coefficient model, given by,

$$Y_i(\tau) = \mu(\tau) + \beta(\tau)X_i(\tau) + e_i(\tau), \quad s \in [0,T], \quad \tau \in [0,T].$$

Here, the predictor process can only affect the current response when $s = \tau$.

Since we only collect discretised realisations of the function, basis expansions are commonly used to project functional predictors, responses or coefficients onto a lower dimensional space for smoothing. While splines (e.g., B-splines, Fourier expansions,...)

are popular choices of basis functions, functional principal components analysis (FPCA) based expansions have become a key dimension reduction technique as it achieves the highest variation in the functional data explained when compared to other basis functions of the same dimension, K (Wang et al., 2016). However, FPCA relies on the spectral decomposition of the empirical covariance operator. The covariance operator can be challenging to estimate when the functional data are sparse and contaminated with random noise. FPCA is discussed further in §3.3.

2.1.1 Basis expansions and splines

A key element of FDA remains the use of basis expansions to represent functional data. Basis expansions in FDA can be used for smoothing purposes, and for projecting dense functional data onto a lower dimension (Kokoszka and Reimherr, 2017).

Consider discretely observed functional data, $\boldsymbol{X}_i = \{X_i(\tau_j) : \ j \in [1, 2, \ldots, T]\}$, a basis expansion is a linear expansion of the form,

$$X_i(\tau_j) \approx \sum_{l=1}^{L} c_{il}\psi_l(\tau_j), \qquad \text{for } i = 1, \ldots, N, \tag{2.2}$$

where $\psi_l(\tau_j)$ denotes the basis function from a collection of spline functions (De Boor, 1972), wavelet transforms (Rioul and Vetterli, 1991) or Fourier series, with coefficient c_{il}. In matrix form, eq. (2.2) is written as,

$$\boldsymbol{X}_i = \boldsymbol{\psi}\mathbf{c}_i,$$

where $\boldsymbol{\psi}$ is the $T \times L$ matrix of basis functions and $\mathbf{c}_i = [c_{i1}, \ldots, c_{iL}]^{\mathrm{T}}$ is a vector of coefficients.

The number of basis functions is given by L which determines the degree of smoothing achieved. Smaller values of $L < T$ achieves greater smoothing, while for $L = T$, an exact representation of the functional observation is obtained and is commonly used for the interpolation of missing observations (Ramsay and Silverman, 2005). The choice of L is an important step when using basis expansions and can be treated as a model selection problem based on goodness-of-fit measures such as the Akaike information criteria (AIC). However, sensitivity to the choice of L can be reduced through a penalised approach which penalises the degree of roughness of the function as discussed further in §2.2.1.

Splines are the most commonly used bases to represent functional data. They are piecewise polynomials of order m with knots at locations $\kappa_1, \ldots, \kappa_L$ that split the interval over which the function $X_i(\tau)$ is to be estimated into $L+1$ sub-intervals (Ramsay and Silverman, 2005). There exists several types of splines: the smoothing spline (Eubank, 1988), B-splines (De Boor, 1972), P-splines which are simply B-splines with a difference penalty on the coefficients (Eilers and Marx, 1996), thin-plate regression splines (Duchon, 1977) and their low-rank approximations (Wood, 2003).

B-splines are attractive since the basis functions are strictly local which means that they are non-zero over a small interval (Wood, 2006a). Figure (2.2) is an illustration of B-spline basis functions defined over $[0, 1]$ with 10 equidistant knots. Each curve represents a basis function. On the other hand, the thin-plate spline by Duchon (1977) has been described as an ideal smoother which does not require the choice of the number of knots since by default a knot is placed at each observation and the corresponding penalty ensures smoothness (Wood, 2006a). However, they have large computational costs which led to the development of low-rank approximations based on either a prior selection of the knot placements, or on a singular value decomposition of the basis matrix which are computationally more efficient (Wood, 2006b).

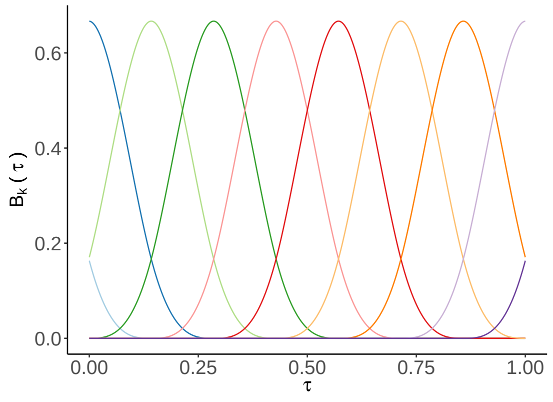


Figure 2.2. *Illustration of 10 B-spline basis functions.*

2.1.2 Multidimensional splines and tensor product bases

While thin-plate bases can extend to estimate multi-dimensional surfaces, bases such as the B-spline are only useful for estimating functions from one predictor variable only (Wood, 2006a). A solution to this problem is to construct a tensor product of marginal bases. Consider the bivariate function, $f(v, \tau)$, which is a function of both v and τ. The first step is to select a low-rank marginal base for each v and τ, $a_k(v)$ and $b_l(\tau)$, such that the bivariate function can be represented in the following form,

$$f(v,\tau) \approx \sum_{l}^{L}\sum_{k}^{K} c_{kl}a_k(v)b_l(\tau).$$

The dimension of the marginal bases are L and K respectively, which gives rise to a LK tensor product basis dimension (Wood, 2006a). Again, the marginal bases can be chosen to be spline basis functions such as B-splines.

2.2 Functional Linear Models with Functional Covariates

In this discussion, we review the methods of estimation for functional linear models with functional covariates.

2.2.1 Methods for a scalar response

We begin with the scalar-on-function type (ref. Table 2.1). In this case, $\beta(s)$ is a univariate functional regression coefficient where $\mathcal{S} = \{s \in (s_{\min}, s_{\max})\}$ is the domain of the functional predictor process, and e_i is the error term with zero mean, and constant variance σ_e^2 for $i = 1, \ldots, n$ (Cardot et al., 1999). A naive approach is to discretise the functional regression coefficient and predictor process with each time s being considered an independent scalar predictor (Ramsay and Silverman, 2005). Assuming that the functional predictor is observed on a regular grid, $s \in [s_1, \ldots, s_R]$, where R is the grid size, an approximation of the scalar-on-function is given by,

$$Y_i = \mu + \sum_{r}^{R} X_i(s_r)\beta(s_r) + e_i.$$

This model is an extension to the multiple regression model and likewise can be estimated using ordinary least squares. This method is impractical for several reasons. For densely observed functional data, a large number of parameters need to be estimated. For τ_j close to $\tau_{j'}$, $X_i(\tau_j)$ will be close to $X_i(\tau_{j'})$ causing multi-collinearity in the regression problem (Kokoszka and Reimherr, 2017). If the functional data are not observed on a common grid, the method is no longer feasible. A solution to this problem is to represent the functional coefficient using a basis expansion of the form $\beta(s) = \sum_{k=1}^{K_\beta} b_k B_k(s)$, where b_k denote the basis coefficients and $B_k(s)$, the basis functions. It can be shown that this reduces to a multiple linear regression model of the form $Y_i = \mu + \sum_{k=1}^{K} c_k x_{ik} + e_i$ where $x_{ik} = \int_{\mathcal{S}} B_k(s)X_i(s)ds$ (Kokoszka and Reimherr, 2017). While this approach requires the choice of the basis dimension K, we can reduce the sensitivity to K by selecting K

large enough and through penalisation,

$$P_\lambda(\mu, \beta) = \sum_{i=1}^{N} \left(Y_i - \mu - \int \beta(s) X_i(s) ds \right)^2 + \lambda \int [D^m \beta(s)]^2 ds,$$

where D^m is an m^{th}-order differential or derivative-based operator on the function $\beta(s)$ and λ is a smoothing parameter (Ramsay and Silverman, 2005; Kokoszka and Reimherr, 2017). Higher values of λ impose larger smoothness constraints on the functional regression coefficient. Methods to find the optimal λ include restricted maximum likelihood and generalised cross-validation methods (Marx and Eilers, 1999; Ruppert, 2002; Ramsay and Silverman, 2005). When expressed as a basis expansion, as shown by Ramsay and Silverman (2005), the penalty is in the form,

$$\mathcal{P}(\mathbf{b}) = \mathbf{b}^{\mathrm{T}} \left[\int D^m \mathbf{B}(s) D^m \mathbf{B}^{\mathrm{T}}(s) ds \right] \mathbf{b} \tag{2.3}$$

$$= \mathbf{b}^{\mathrm{T}} \boldsymbol{\Omega} \mathbf{b}. \tag{2.4}$$

Marx and Eilers (1999) use B-spline basis functions with a second order difference penalty on the coefficients to impose smoothness on the functional regression coefficient.

In addition to the basis expansion of the functional regression, Ramsay and Silverman (2005) have suggested to expand the predictor process using a pre-specified known set of basis functions, $X_i(s) = \sum_{l=1}^{K_X} c_{il} \phi_l(s)$ where c_{il} now denotes the coefficients of the basis functions $\boldsymbol{\phi}(s)$ of dimension K_x used to represent the predictors. Under this approach, the integral can be estimated as $\int_{\mathcal{S}} \beta(s) X_i(s) ds \approx \sum_k \sum_l b_k c_{il} \int_{\mathcal{S}} B_k(s) \phi_l(s) ds$. The coefficients can be estimated using ordinary least squares. In a penalised approach, the smoothing parameter is evaluated using cross-validation (CV).

The functional principal component regression (FPCR) has been suggested by Reiss and Ogden (2007) in which the scalar response is regressed against the functional principal component (FPC) scores of the design matrix $\mathbf{X} = \{x_{ik} = \int_{\mathcal{S}} B_k(s) X_i(s) ds\}$ where $B_k(.)$ represent B-spline functions. Cross-validation is used to select the number of principal components and the authors propose two approaches to estimate λ; 1) by fitting a linear mixed effects model using restricted maximum likelihood (REML), and 2) cross-validation.

The classical scalar-on-function model has been extended to account for more complex structures which may also allow for non-Gaussian data (Müller and Stadtmüller, 2005).

For a longitudinal diffusion tensor imaging study, Goldsmith et al. (2011a, 2012) implemented a two-stage estimation procedure for responses generated from a distribution of the exponential family. The functional predictor is first projected onto a large number of eigenfunctions from an estimated covariance operator to take into account measurement error and for sparse data measured on an irregular grid. The estimated coefficients are then plugged into the regression model to find the coefficients of the basis functions used to represent the functional regression coefficient, $\mathbf{b}$ (eq. (2.4)). Although it is mathematically simple when the basis expansions for the predictor process and the functional regression coefficient are both the orthornormal bases of eigenfunctions, Goldsmith et al. (2011a, 2012) argue against this choice because 1) the FPC basis may not be an appropriate basis for the functional parameter, and 2) the smoothness induced on the functional parameter will depend on the number of eigenfunctions selected to which the functional predictor processes are also sensitive.

Penalised splines are common choices when the functional coefficient is expected to be smooth which may also facilitate interpretation of the latter. However, in some cases $\beta(s)$ may be expected to possess spikes and irregular features. As a result, wavelet-based approaches with a LASSO (Tibshirani, 1996) penalty have been explored (Zhao et al., 2012).

The two-stage analysis of Goldsmith et al. (2011a) has been criticised by Crainiceanu et al. (2009) who showed that this method produces biased estimators with misspecified variability in $\mathbf{b}$, and alternatively proposed a joint Bayesian estimation using a Metropolis-within-Gibbs as a closed form solution of the posterior distribution of the FPC scores could not be derived. Similarly, Crainiceanu and Goldsmith (2010) presented a Bayesian scalar-on-function regression model which jointly obtains smooth estimates of the functional predictor process using a FPC basis expansion within their Metropolis-within-Gibbs sampling algorithm. The same FPC basis expansion was used for the functional regression coefficient. Goldsmith et al. (2011b); McLean et al. (2013) proposed variational Bayes methods (Fox and Roberts, 2012) for fast approximate inference of the scalar-on-function model by Crainiceanu and Goldsmith (2010) which however does not require FPC basis expansions for the functional regression coefficient.

2.2.2 Methods for a functional response

This class of model is the least researched area within functional regression. As illustrated in Table (2.1), models with a functional response and functional covariates give rise to a bivariate regression coefficient surface. In this setting, traditional scalar-on-function methods have been extended for functional responses. We first focus on the unconstrained fully functional linear model. To estimate the regression surface, bivariate basis expansions have been suggested (Ramsay and Silverman, 2005). These include tensor products of splines or thin-plate spline bases by Wood (2003). For tensor products, the regression surface is approximated as, $\beta(s,\tau) = \sum_k \sum_l b_{uk}\psi_k(s)f_l(\tau)$ where $\psi(.)$ and $f_l(.)$ denote the marginal bases. The penalty matrices based on the Kronecker sum of the penalties corresponding to the marginal bases have been discussed by Wood (2016). The model can then be estimated using standard ordinary least squares approaches (Ramsay and Silverman, 2005; Kokoszka and Reimherr, 2017), or using a mixed effect framework for penalised functional regression to estimate the smoothing parameter (Scheipl et al., 2015; Greven and Scheipl, 2017).

Greven and Scheipl (2017) implemented a component-wise gradient boosting algorithm to minimise the expected loss with respect to the linear predictor. Ramsay and Silverman (2005) proposed to represent the functional response using a basis expansion whose coefficients are first estimated before evaluating the regression model. Ivanescu et al. (2015) extended the scalar-on-function mixed effect model by Goldsmith et al. (2011a) for a functional response and multiple functional predictors including a plug-in estimate of the functional predictor process for sparsely or corruptly observed data before estimating the bivariate coefficient. Yao et al. (2005) construct an estimator for the bivariate regression surface based on FPCA designed for sparsely observed covariates with measurement error.

Constrained function-on-function linear models will now be discussed, primarily the historical and lag-based models. Malfait and Ramsay (2003) considered the functional linear model such that only the predictor processes at times $s \leq \tau$ will affect the outcome at time τ, also known as the historical functional linear model (HFLM). Unlike the rectangular support of the unconstrained model, the bivariate surface of the HFLM has a triangular support since $\beta(s,\tau) = 0$ for $s > \tau$. The authors therefore constructed a triangular basis using the finite element method for the expansion of $\beta(s,\tau)$. Malfait and Ramsay (2003) implement a least squares approach to estimate the parameters of the model and the surface can be recovered using $\hat{\beta}(s,\tau) = \sum_k \hat{b}_k\theta_k(s,\tau)$. Harezlak et al. (2007) expanded this model by penalising the horizontal, vertical and parallel directions of the surface, and estimate the model parameters by minimising the penalised least squares criterion. Ivanescu et al. (2015) state that their mixed effect model for functional response can include historical functional effects. For the HFLM, Brockhaus et al. (2017) used a tensor product of spline basis functions to expand the regression surface, and the model is then estimated by a component-wise gradient boosting algorithm. The model was applied to a study aiming at monitoring the cell dry mass during a biotechnological fermentation for the production of a model protein.

Another method of estimation which synthesises varying coefficient models (Hastie and Tibshirani, 1993) and functional regression to obtain a smooth estimate of the regression surface of the recent history model was proposed by Kim (2010). The first step involves expanding the bivariate surface $\beta(s, \tau)$ along s only using spline bases and estimating the coefficient $b_u(\tau_j)$ which denotes the u^{th} basis coefficient at time τ_j. This is estimated in a pointwise approach using least squares at each τ_j. The set of estimated coefficients $b_u(\tau_j)$ is then smoothed using local polynomial regression. The proposed model was applied to investigate the relationship between the flow level at Curlew creek in Florida and precipitation. For sparsely observed data, Kim et al. (2011) extended the model by Kim (2010) and propose an estimation procedure involving a two-dimensional local linear smoothing to the empirical auto- and cross-covariances.

For the lag functional linear model (LFLM), Pomann et al. (2016) propose two estimation procedures, 1) a pointwise approach for each τ_j similar to Kim (2010), 2) and a global approximation using a penalised bivariate thin-plate splines expansion for the bivariate surface estimated within the mixed effect model framework and REML. For sparsely observed data, the predictor process is estimated using a FPCA based expansion. Their results show that the global model outperforms the pointwise model. The selection of the lag parameter is generally treated as a model selection problem whereby the lag which minimises the AIC criterion is selected as the optimal parameter (Kim, 2010; Pomann et al., 2016). Pomann et al. (2016) applied the methods to estimate a magnetic resonance imaging (MRI)-based measure of tissue damage in multiple sclerosis patients.

Within the Bayesian framework, Goldsmith and Schwartz (2017) extended the scalar-on-function model by Goldsmith et al. (2011b) for the functional concurrent linear model with a spike-and-slab prior on the regression coefficients for variable selection. The unconstrained function-on-function linear model by Meyer et al. (2015) on the other hand implemented a loss-less expansion of both the functional response and predictors a-priori to project the functional data onto a lower dimension estimated using a Gibbs sampling algorithm. Meyer et al. (2021) consider prior information using a mixture normal distribution to impose the constraints of the historical functional linear model along with a wavelet-transform basis of the bivariate coefficient surface fitted using a Gibbs sampler.

2.3 Functional Time-Series Models

The most popular model for the analysis of functional time-series is the functional autoregressive process of order 1, FAR(1) developed by Bosq (2000). The model is given by,

$$Y_t(\tau) = \int_s \psi(s,\tau) Y_{t-1}(\tau) ds + e_t(\tau).$$

The estimation of $\psi(s,\tau)$ relies on the eigendecomposition of the estimated lagged covariance operator using a truncated set of eigenfunctions (Kowal et al., 2019). The FAR(1) model was later adapted by including a weighted kernel of the covariance operator which was shown to be more robust to non-stationary time series (Besse et al., 2000). There exist several methods of estimation for the functional time series model. Hyndman and Ullah (2007) propose a functional principal component analysis approach where a time-series analysis is performed on the FPC scores of the observations. This is referred to as the Hyndman-Ullah FAR(1) model.

Aue et al. (2015) implement a multivariate time-series model on the FPC scores of the functional data. More closely related to our application, Damon and Guillas (2002) extended the FAR(1) model by Bosq (2000) to include exogenous variables (FARX(1)). The model is given by,

$$Y_t(\tau) = \int_s \psi(s,\tau) Y_{t-1}(\tau) ds + \int_s \beta(s,\tau) X_t(s) ds + e_t(\tau).$$

Similarly, the estimation of $\beta(s,\tau)$ relies on the eigendecomposition of the empirical covariance operator between $Y_t(\tau)$ and $X_t(\tau)$ using a truncated set of eigenfunctions. The authors applied the methods for ozone forecasting using hourly temperature and wind speed measurements. Their results show that the inclusion of the meteorological information increases predictive power, and the authors suggest that improvements could be made to the ozone forecasts by including more meteorological variables.

Traditional FAR(p) methods however are not suitable for data with measurement error and may result in model misspecification (Kowal et al., 2019). This is then addressed using a Bayesian hierarchical model by including an observation process and a FAR(p) latent process (Kowal et al., 2019). Within the Bayesian framework, functional dynamic linear models (FDLM) were developed and estimated using the Gibbs sampler (Kowal et al., 2017; Kowal, 2021). The FDLM builds upon the dynamic functional factor model by Hays et al. (2012).

CHAPTER 3

Methodology

In this thesis, we consider a Bayesian historical function-on-function linear model (HFLM). To the best of our knowledge, Bayesian analysis of the historical functional linear model is sparse in the literature. Pomann et al. (2016) focused on model estimation and fit assessed using criteria such as AIC and mean-squared error based measures within a mixed effects framework to impose smoothness constraints, and Meyer et al. (2021) established a framework for Bayesian inference and uncertainty quantification for HFLMs with a single predictor assumed to be noiseless. As such, the performance of these models with respect to uncertainty quantification for multiple predictors in the presence of measurement error is unclear.

Therefore, we address this gap by proposing a historical functional linear model which simultaneously estimates the true functional covariate from noisy and sparse data. We expand upon the work done by Crainiceanu et al. (2009); McLean et al. (2013) for the scalar-on-function model to functional regression models with historical effects. Our proposed Bayesian model consists of an *observation equation* to obtain smooth estimates of the true covariates from sparsely observed predictors with measurement error, and a *regression equation* which then models the relationship between the observed functional response and the true functional covariates. We propose two methods of estimation. In the first proposition, we set up an MCMC algorithm in which the observation equation of the predictor process is modelled independently from the regression model. Secondly, the observation equation is estimated jointly with the regression model.

This chapter is outlined as follows: we present the Bayesian formulation of the historical functional linear model in §3.1 and motivate our choice for the prior disrtibution of the functional regression coefficients in §3.2. To account for noisiness in the functional covariates, we describe the smoothing strategy in §3.3, and formulate the corresponding posterior distribution. We outline the MCMC sampling algorithms in §3.4, and describe the Cholesky algorithm implemented for fast sampling from multivariate Gaussian distributions in §3.4.3. We discuss methods to flag non-identifiability in HFLMs in §3.5 and Bayesian inferential methods in §3.6. An R package which implements C++ code via the Rcpp package is being developed to increase computational efficiency, and can be found in the GitHub: yovnajunglee/BayesHFLM.

3.1 A Bayesian Historical Functional Linear Model

Let $\{Y_i(\tau_1), \ldots, Y_i(\tau_T)\}$ be the i^{th} functional response observed at a set of discretised points, $\{\tau_j;\ j = 1, ..., T\}$. Suppose we have G functional predictors $X_{ig}(v_r)$ for $i = 1, \ldots, N$, also discretely observed on a regular grid $\{v_r;\ r = 1, \ldots, S\}$. The aim is to model the relationship between the functional predictors and the functional response, imposing a restriction such that only a subdomain of the predictor $X_{ig}(v)$ is allowed to influence $Y_i(\tau)$. By synthesising semi-parametric methods with functional data analysis, we obtain the following model,

$$Y_i(\tau) = \mu(\tau) + \sum_{g=1}^{\mathrm{G}} \left(\int_0^{\tau} \theta_g(v,\tau) X_{ig}(v) dv \right) + e_i(\tau), \quad e_i(\tau) \sim \mathcal{N}_1\left(0, \sigma_e^2\right), \tag{3.1}$$

where the bivariate regression surface can be represented using a tensor product of splines such that,

$$\theta_g(v,\tau) \approx \sum_{k=1}^{K_Y} \sum_{u=1}^{U} b_{uk,g} \psi_{u,g}(v) f_{k,g}(\tau),$$

and the functional intercept term, $\mu(\tau)$, is expressed using a basis expansion,

$$\mu(\tau) \approx \sum_{m=1}^{K_\mu} f_{\mu,m}(\tau)\mu_m.$$

We assume that the errors are independent between and within the curves.[1] The spline basis functions are denoted by $f_{k,g}(\tau), \psi_{u,g}(v)$ and $f_{\mu,m}(\tau)$, with coefficients $b_{uk,g}$ and μ_m. We can recover the regression surface for the g^{th} functional predictor as

$$\hat{\theta}_g(v,\tau) = \sum_{k=1}^{K_Y} \sum_{u=1}^{U} \hat{b}_{uk,g} \psi_{u,g}(v) f_{k,g}(\tau), \text{ for } 0 \leq v \leq \tau.$$

Consequently, the coefficient surface is only evaluated when $v \leq \tau$, while for $v > \tau$, the surface is set to 0. This gives rise to a triangular coefficient surface.

We follow Kim (2010)'s strategy, and use numerical approximations to calculate the

[1]$\mathcal{N}_D$ denotes a multivariate Gaussian distribution of dimension D.

integrals in (3.1). A (left) Riemann sum approximation is given by,

$$\int_0^\tau \sum_{u=1}^{U} \psi_{u,g}(v_r) b_{uk,g} X_{ig}(v) dv = \sum_{u=1}^{U} b_{uk,g} \int_0^\tau X_{ig}(v_r)\psi_{u,g}(v_r) dv \tag{3.2}$$

$$\approx \sum_{u=1}^{U} b_{uk,g} \sum_{v_r \in [0,\tau)} X_{ig}(v_r)\psi_{u,g}(v_r) \times \Delta(v_r). \tag{3.3}$$

Since we consider a fixed grid with equally spaced intervals, $\Delta(v_r) = (v_r - v_{r-1})$ is constant $\forall\ r \in [1, 2, \ldots, S]$.

Using notations by Brockhaus et al. (2017), consider a fixed regular grid where the N responses $\mathbf{Y} = [Y_i(\tau_j)]_{\substack{i=1,\ldots,N\\ j=1,\ldots,\mathrm{T}}}$ and the g^{th} functional covariate $\mathbf{X}_g = [X_{ig}(v_r)]_{\substack{i=1,\ldots,N\\ r=1,\ldots,S}}$ are both observed on a common grid with size $T = S$. The matrix of the error terms is given by $\mathbf{e} = [\mathrm{e}_i(\tau_j)]_{\substack{i=1,\ldots,N\\ j=1,\ldots,\mathrm{T}}}$. The HFLM can be represented in matrix form,

$$\begin{aligned}
\operatorname{vec}(\boldsymbol{Y}_{N\times T}) = &\left(\underset{T\times K_\mu}{\boldsymbol{f}_\mu} \otimes \underset{N\times 1}{\mathbf{1}_N}\right) \underset{K_\mu \times 1}{\boldsymbol{\mu}} + \\
&\sum_{g=1}^{G} \left(\left[\left\{ \underset{NT\times S}{\boldsymbol{H}} \cdot \left(\underset{N\times S}{\boldsymbol{X}_g} \underset{S\times T}{\boldsymbol{\Delta}} \otimes \underset{T\times 1}{\mathbf{1}_T} \right) \right\} \underset{S\times U}{\boldsymbol{\psi}_g} \right] \odot \left[\underset{N\times 1}{\mathbf{1}_N} \otimes \underset{T\times K_Y}{\boldsymbol{f}_g} \right] \right) \underset{UK_Y\times 1}{\boldsymbol{b}_g} + \\
&\operatorname{vec}(\mathbf{e}_{N\times T}) \\
&= \mathbf{F}\boldsymbol{\mu} + \sum_{g=1}^{G} (\boldsymbol{B}_g \odot \boldsymbol{B}_Y)\, \mathbf{b}_g + \operatorname{vec}(\mathbf{e}_{N\times T}) \\
&= \mathbf{R}\boldsymbol{\alpha} + \operatorname{vec}(\mathbf{e}_{N\times T}),
\end{aligned} \tag{3.4}$$

where vec $(\boldsymbol{Y})$ is the vectorisation of the matrix $\mathbf{Y}$, $\boldsymbol{f}_g$ and $\boldsymbol{\psi}_g$ are the marginal bases for the tensor product used to represent the g^{th} bivariate regression coefficient, of K_Y and U basis functions in the τ- and $v-$ direction respectively, $\boldsymbol{\Delta}$ is a diagonal matrix of integration weights $\Delta(v_r)$, $\mathbf{H}$ is an indicator matrix for the integration limits in eq. (3.3), $\mathbf{b}_g$ is the vector of coefficients for the g^{th} predictor, $\otimes$ and $\odot$ denote a Kronecker product and a row-tensor product respectively. The $N \times (K_\mu + GUK_Y)$ design matrix $\mathbf{R}$ is given by,

$$\mathbf{R} = [\ \mathbf{F} \mid \boldsymbol{B}_1 \odot \boldsymbol{B}_Y \mid \cdots \mid \boldsymbol{B}_G \odot \boldsymbol{B}_Y\ ],$$

with coefficients $\boldsymbol{\alpha}^{\mathrm{T}} = [\boldsymbol{\mu}^{\mathrm{T}}\ \mathbf{b}_1^{\mathrm{T}}\ \ldots\ \mathbf{b}_G^{\mathrm{T}}\ ]$.

3.2 Prior specification for functional regression coefficients

We make a simplifying assumption, and all the regression surfaces are estimated using the same tensor product of splines basis, namely, $\theta_g(v,\tau) \approx \sum_{u,k} f_k(\tau) b_{uk,g} \psi_u(v)$ for $g = 1, \ldots, \mathrm{G}$. Within the Bayesian paradigm, smoothness can be induced using prior information as described by Silverman (1985) and Berry et al. (2002). Under the classical approach, the objective is to minimise the penalised likelihood or least squares,

$$\min_{\boldsymbol{\mu}, \mathbf{b}_g \text{ for } g=1,\ldots,G} \left[\operatorname{vec}(\mathbf{Y}) - \left(\mathbf{F}\boldsymbol{\mu} + \sum_{g=1}^{G} (\boldsymbol{B}_g \odot \boldsymbol{B}_Y) \mathbf{b}_g \right) \right]^{\mathrm{T}} \left[\operatorname{vec}(\mathbf{Y}) - \left(\mathbf{F}\boldsymbol{\mu} + \sum_{g=1}^{G} (\boldsymbol{B}_g \odot \boldsymbol{B}_Y) \mathbf{b}_g \right) \right] + \lambda_\mu \mathcal{P}(\boldsymbol{\mu}) + \sum_{g=1}^{G} \lambda_{b,g} \mathcal{P}(\mathbf{b}_g), \quad (3.5)$$

where $\mathcal{P}(.)$ is a penalty function such as the difference or derivative-based penalties as defined in eq. (2.4) with smoothing parameters $\lambda_{b,g}, \lambda_\mu$. Equivalently, in the Bayesian framework, prior information on the parameters $\boldsymbol{\mu}, \mathbf{b}_g$ are included such that the posterior distribution of the parameters solves eq. (3.5). Following this, the prior for $\boldsymbol{b}_g$ is $\mathcal{N}_{UK_Y}\left(\mathbf{0},\ \sigma^2_{b,g} \boldsymbol{\Omega}_b^{-1}\right)$, where the prior precision matrix, $\boldsymbol{\Omega}_b = \mathcal{P}(\mathbf{b}_g)$.

For tensor products of splines, the Kronecker product of the penalties of the corresponding marginal bases was first suggested by Fahrmeir and Lang (2001). In recognition of the fact that the Kronecker products of penalties may lead to undersmoothing, Wood (2006b) proposed a penalty which measures the wiggliness of the function in the form of a Kronecker sum of the marginal penalties,

$$\lambda_{b,g} \mathcal{P}(\mathbf{b}_g) = \lambda_{b,g} \boldsymbol{\Omega}_b = \lambda_{s,g} \boldsymbol{\Omega}_\psi \otimes \mathbf{I}_{K_Y} + \lambda_{f,g} \mathbf{I}_U \otimes \boldsymbol{\Omega}_f.$$

The penalties for the marginal bases are given by $\boldsymbol{\Omega}_\psi$ and $\boldsymbol{\Omega}_f$ with smoothing parameters $\lambda_{s,g}$ and $\lambda_{f,g}$. We assume equal smoothing parameters in both directions, that is, $\lambda_{s,g} = \lambda_{f,g}$ which we denote by $\lambda_{b,g}$, thus, $\boldsymbol{\Omega}_b = \boldsymbol{\Omega}_\psi \otimes \mathbf{I}_{K_Y} + \mathbf{I}_U \otimes \boldsymbol{\Omega}_f$. Under this assumption, a closed-form solution to the conditional posterior distribution of the smoothing parameter $\lambda_{b,g}$ is obtained which simplifies computation. In cases where $\lambda_{s,g} = \lambda_{f,g}$ may not be plausible, the conditional posterior distribution of each of the smoothing parameters is intractable which led to the development of a sliced sampling strategy by McLean et al. (2013) to obtain samples of the smoothing parameters. Nevertheless, the case of equal smoothing is not implausible for the application which motivates the development of this method in this dissertation.

It can be convenient to reparameterise $\boldsymbol{b}_g = \widetilde{\boldsymbol{\Omega}}_b \widetilde{\boldsymbol{b}}_g$ to obtain a diagonal prior precision matrix for $\widetilde{\boldsymbol{b}}_g$, which improves MCMC mixing properties (Wand and Ormerod, 2008; Kowal et al., 2017) and may simplify computations. From the singular value decomposition of $\boldsymbol{\Omega}_b = \mathbf{U}_\omega \mathbf{D}_\omega \mathbf{U}_\omega^{\mathrm{T}}$ where ${\mathbf{U}_\omega}^{\mathrm{T}} \mathbf{U}_\omega = \mathbf{I}_{UK_Y}$ and $\mathbf{D}_\omega$ is a diagonal matrix (Wand and Ormerod, 2008; Kowal et al., 2017), we set

$$\widetilde{\boldsymbol{\Omega}}_b = \mathbf{U}_\omega \mathbf{D}_\omega^{-0.5},$$

giving rise to a prior for $\widetilde{\boldsymbol{b}}_g \sim \mathcal{N}_{UK_Y}\left(\mathbf{0},\ \sigma_{b,g}^2 \mathbf{I}_{UK_Y}\right)$. Once the bases for $\{\theta_g(v,\tau)\}$ are chosen, the model then depends on the choice of basis dimensions, U and K_Y. Ruppert (2002) shows that the location of the knots is unimportant as long as the dimensions are large enough and that sensitivity to the size of the dimension can be reduced under a penalised approach.

Similarly, the prior for the coefficients of the functional intercept term, $\boldsymbol{\mu} \sim \mathcal{N}_{K_\mu}\left(0,\ \sigma_\mu^2 \boldsymbol{\Omega}_\mu^{-1}\right)$ which is also reparameterised to obtain a diagonal prior precision matrix.

From eq. (3.4) and under the reparameterisation, the model can be written as,

$$\operatorname{vec}(\mathbf{Y}) = \mathbf{F}\widetilde{\boldsymbol{\Omega}}_\mu \widetilde{\boldsymbol{\mu}} + \sum_{g=1}^{G} \left(\boldsymbol{B}_g \odot \boldsymbol{B}_Y\right) \widetilde{\boldsymbol{\Omega}}_b \widetilde{\mathbf{b}}_g + \operatorname{vec}(\boldsymbol{\epsilon}) = \widetilde{\mathbf{R}}\widetilde{\boldsymbol{\alpha}} + \operatorname{vec}(\boldsymbol{e}). \tag{3.6}$$

We choose inverse-gamma priors for the variance parameters are given by, $\sigma_\mu^2, \sigma_{b,g}^2 \sim \mathcal{IG}(0.001, 0.001)$. This IG prior is commonly used as an uninformative prior. Furthermore, a continuous density prior on the smoothing parameters is an automatic way of preventing undersmoothing as the prior probability of doing no smoothing is zero (Berry et al., 2002).

Let $\boldsymbol{D}_{\widetilde{\boldsymbol{\alpha}}}^{-1} = \operatorname{diag}(\sigma_\mu^2 \mathbf{I}_{K_\mu},\ \sigma_{b,1}^2 \mathbf{I}_{UK_Y},\ \ldots,\ \sigma_{b,G}^2 \mathbf{I}_{UK_Y})$. Denote the number of regression coefficients to estimate, $p = K_\mu + GUK_Y$ and the prior distribution for the regression coefficients, $\widetilde{\boldsymbol{\alpha}} | \sigma_\mu^2, \sigma_{b,g}^2 \sim \mathcal{N}_p\left(\mathbf{0},\ \boldsymbol{D}_{\widetilde{\boldsymbol{\alpha}}}^{-1}\right)$. It can then be shown (see Appendix B for derivations) that the conditional posterior distribution of the regression coefficients is given by, $\widetilde{\boldsymbol{\alpha}}| \cdot \sim \mathcal{N}_p(\boldsymbol{\mu}_{\widetilde{\boldsymbol{\alpha}}}, \boldsymbol{\Sigma}_{\widetilde{\boldsymbol{\alpha}}})$, where,

$$\boldsymbol{\Sigma}_{\widetilde{\boldsymbol{\alpha}}} = \left[\frac{1}{\sigma_e^2} \widetilde{\boldsymbol{R}}^{\mathrm{T}} \widetilde{\boldsymbol{R}} + \boldsymbol{D}_{\widetilde{\boldsymbol{\alpha}}}\right]^{-1} \text{ and } \boldsymbol{\mu}_{\widetilde{\boldsymbol{\alpha}}} = \frac{1}{\sigma_e^2} \boldsymbol{\Sigma}_{\widetilde{\boldsymbol{\alpha}}} \widetilde{\boldsymbol{R}}^{\mathrm{T}} \operatorname{vec}(\boldsymbol{Y}). \tag{3.7}$$

The conditional posterior distribution of the variance parameters are given by,

$$\sigma_e^2|\cdot \sim \mathcal{IG}\left(0.5NT + 0.001,\ \frac{1}{2}\left(\mathrm{vec}(\boldsymbol{Y}) - \widetilde{\mathbf{R}}\widetilde{\boldsymbol{\alpha}}\right)^{\mathrm{T}}\left(\mathrm{vec}(\boldsymbol{Y}) - \widetilde{\mathbf{R}}\widetilde{\boldsymbol{\alpha}}\right) + 0.001\right),$$
$$\sigma_\mu^2|\cdot \sim \mathcal{IG}\left(0.5K_\mu + 0.001,\ \frac{1}{2}\widetilde{\boldsymbol{\mu}}^{\mathrm{T}}\widetilde{\boldsymbol{\mu}} + 0.001\right),$$
$$\sigma_{b,g}^2|\cdot \sim \mathcal{IG}\left(0.5UK_Y + 0.001,\ \frac{1}{2}\widetilde{\boldsymbol{b}}_g^{\mathrm{T}}\widetilde{\boldsymbol{b}}_g + 0.001\right).$$

3.3 Modelling functional covariates

To handle sparseness and low signal-to-noise ratio in the covariates, an additional smoothing step is implemented. Our technique is different from that employed by Pomann et al. (2016) in their HFLM where estimates of the true functional covariates are used as plug-in values in their global estimation of the regression model because this method can lead to poor estimates in the functional regression with understated variability in the parameters (Crainiceanu et al., 2009). Hence we expand upon the work done by Crainiceanu et al. (2009), Crainiceanu and Goldsmith (2010) and McLean et al. (2013) for the scalar-on-function model and we implement a fully Bayesian approach to account for the uncertainty in the functional predictor process.

Following the same procedure by McLean et al. (2013), denote the observed predictor process by $\omega_{ig}(v)$ and true latent predictor process, $X_{ig}(v)$, the functional predictors are modelled using a basis expansion and the observation process is given by,

$$\omega_{ig}(v) = X_{ig}(v) + \eta_{ig}(v) \tag{3.8}$$
$$= \mu_g(v) + \sum_{k=1}^{K_g} \xi_{ik,g}\phi_{k,g}(v) + \eta_{ig}(v), \quad \eta_{ig}(v) \sim \mathcal{N}(0, \sigma_{\eta_g}^2) \tag{3.9}$$

where $\phi_k(v)$ are basis expansions with coefficients $\xi_{ik,g}$ and K_g denotes the basis dimension of the g^{th} predictor. We assume i.i.d errors between and within curves. We choose the basis of functional principal components (FPCs), where $\boldsymbol{\phi}_g$ represents a K_g-dimensional matrix of FPCs which are orthonormal, $\boldsymbol{\phi}_g^{\mathrm{T}}\boldsymbol{\phi}_g = \mathbf{I}_{K_g}$. In a Bayesian setting, the orthonormality of the functional principal component bases helps to overcome the computational costs associated with drawing samples from high-dimensional Gaussian distributions. This is because the discretised orthonormality constraint provides an important simplification for evaluating the inverse of the posterior covariance matrix required for sampling in the MCMC. Furthermore, they have good mixing and convergence properties (Crainiceanu and Goldsmith, 2010).

For the latent functional process $X_{ig}(v)$, an estimate of the sample covariance matrix, $\hat{\mathbf{C}}$, where the covariance between time points s and v, is,

$$\hat{c}_g(s,v) = \frac{1}{N-1}\sum_{i=1}^{N}\Big(X_{ig}(s) - \bar{X}_g(s)\Big)\Big(X_{ig}(v) - \bar{X}_g(v)\Big).$$

The functional data means are given by, $\bar{X}_g(s) = \frac{1}{N}\sum_{i=1}^{N} X_{ig}(s)$.

The FPCs are derived from the spectral decomposition of the covariance operator $\hat{c}_g(s,v) = \sum_{k=1}^{\infty} d_{k,g}\phi_{k,g}(s)\phi_{k,g}(v)$ where $\phi_{k,g}(\cdot)$ are the eigenfunctions arranged in decreasing order of the corresponding eigenvalues $d_{k,g}$. However, we do not observe $X_{ig}(v)$ but rather a noisy process $\omega_{ig}(v)$. Consequently, the resulting eigenfunctions tend to not be smooth. There are three main approaches to address this problem. In the first approach, the eigenfunctions are first obtained from the sample covariance matrix and are then smoothed (Ramsay and Silverman, 2005). Another proposition is to smooth the sample covariance matrix and to diagonalise it (Ramsay and Silverman, 2005; Yao et al., 2005). Lastly, the sample curves are first smoothed and the resulting covariance matrix diagonalised (Ramsay and Silverman, 2005). Xiao et al. (2016) proposed a fast bivariate smoothing method for the covariance operator, known as fast covariance estimation (FACE), which synthesises the last two approaches and overcomes the computational costs of operations performed on large covariance matrices in FPCA. Hence from eq. (3.9), $\phi_{k,g}(v)$ represents the functional principal components evaluated using FACE, $\boldsymbol{\xi}_{i,g} = [\xi_{i1,g},\ldots,\xi_{iK_g,g}]^{\mathrm{T}}$ denotes the FPC scores of the i^{th} observation, and K_g denotes the number of FPC chosen for the g^{th} predictor. It is however important to note that the FPC bases are computed a priori. Since FPC based expansions are data-driven approaches, our proposed method does not take into consideration the uncertainty associated from constructing bases using error contaminated functional data. Thus this may lead to understated uncertainty in the resulting estimates of the true functional predictor processes and consequently, the estimated bivariate coefficient surface of the regression model.

3.3.1 Prior specification

We choose the following hierarchical prior for the coefficients; $\xi_{ik,g} \sim \mathcal{N}_1(0, \lambda_{k,g})$ and inverse-gamma for the variance parameter $\sigma^2_{\eta_g}$ with known hyperparameters i_1, i_2. We found that an uninformative inverse-gamma prior for $\lambda_{k,g}$ with scale and shape hyperparameters equal to 0.001 produces inflated estimates of values of $\lambda_{k,g}$ which tend to zero. This was expected since Gelman (2006) has shown that the uninformative

IG prior tends to be informative for small variance parameters. Since empirically $\hat{\lambda}_{k,g}$ correspond to the eigenvalues, a possible solution is to discard FPCs for which $d_{k,g} < \epsilon$ where ϵ is small. However, keeping the number of FPCs large enough is crucial to ensure identifiability of the corresponding design matrix of the functional regression model (Scheipl and Greven, 2016). We choose the prior $\pi(\lambda_{k,g}) \propto \lambda_{k,g}^{-1}$ which has the form of an inverse-χ^2 density with 0 degrees of freedom (Gelman, 2006).

Denote $\boldsymbol{\Lambda}_g = \mathrm{diag}(\lambda_{1,g}, \dots, \lambda_{K_g,g})$. Under this prior specification, it can be shown (see Appendix C for derivations) that the conditional posteriors are given by:

$$\boldsymbol{\xi}_{i,g}|\widetilde{\boldsymbol{\omega}}_i, \cdot \sim \mathcal{N}_{K_g}\left(\boldsymbol{\mu}_{\widetilde{\boldsymbol{\omega}}_{ig}} = \frac{1}{\sigma^2_{\eta_g}}\boldsymbol{\Sigma}_{\widetilde{\boldsymbol{\omega}}_{ig}}\boldsymbol{\phi}_g^{\mathrm{T}}\widetilde{\boldsymbol{\omega}}_{ig},\ \boldsymbol{\Sigma}_{\widetilde{\boldsymbol{\omega}}_{ig}} = \left[\boldsymbol{\Lambda}_g^{-1} + \frac{1}{\sigma^2_{\eta,g}}\mathbf{I}_{K_g}\right]^{-1}\right), \tag{3.10}$$

$$\lambda_{k,g}|\cdot \sim \mathcal{IG}\left(0.5N,\ 0.5\sum_{i=1}^{N}\xi^2_{ik,g}\right), \tag{3.11}$$

$$\sigma^2_{\eta_g}|\cdot \sim \mathcal{IG}\left(i_1 + 0.5NS,\ i_2 + 0.5\sum_{i=1}^{N}(\widetilde{\boldsymbol{\omega}}_{ig} - \boldsymbol{\phi}_g\boldsymbol{\xi}_{i,g})^T(\widetilde{\boldsymbol{\omega}}_{ig} - \boldsymbol{\phi}_g\boldsymbol{\xi}_{i,g})\right), \tag{3.12}$$

where $\widetilde{\omega}_{ig}(v) = \omega_{ig}(v) - \mu_g(v)$. Note that $\boldsymbol{\Sigma}_{\widetilde{\boldsymbol{\omega}}_{ig}} = \left[\boldsymbol{\Lambda}_g^{-1} + \frac{1}{\sigma^2_{\eta_g}}\mathbf{I}_{K_g}\right]^{-1}$ simplifies to $\mathrm{diag}\left(\frac{\lambda_{1,g}\sigma^2_{\eta_g}}{\lambda_{1,g}+\sigma^2_{\eta_g}}, \dots, \frac{\lambda_{K_g,g}\sigma^2_{\eta_g}}{\lambda_{K_g,g}+\sigma^2_{\eta_g}}\right)$. We set $i_1 = i_2 = 0.001$.

3.4 MCMC sampling algorithm

We construct an MCMC sampling algorithm which consists of the following three main blocks; sampling (i) the FPC scores to obtain estimates of the smooth latent predictor process $\{X_{ig}(v)\}$, (ii) the regression coefficients $\{\mu_k, b_{uk,g}\}$, and (iii) the variance components. We consider two different methods of estimation to incorporate the smoothing strategy described above. Firstly, we propose to model the observation process (3.9) independently from the regression equation, secondly, jointly using a Bayesian hierarchical model, as explained below.

3.4.1 Bayesian independent two-stage model

Here, model (3.9) is estimated independently from model (3.1). The uncertainty in the predictor processes is incorporated via a substitution strategy within the Gibbs sampling algorithm. The FPC scores, $\boldsymbol{\xi}_{i,g}$ are drawn from their posterior distributions, $\boldsymbol{\xi}_{i,g}|\widetilde{\boldsymbol{\omega}}_i$ defined in eq. (3.10), and at each iteration, an estimate of the predictor process is obtained using $X_i^{(m)}(v) = \sum_{k=1}^{K_g} \xi_{ik,g}^{(m)} \psi_k(v)$. The design matrix is then updated and used to sample the parameters of the regression model. The Gibbs sampling algorithm is outlined in Algorithm (1).

3.4.2 Joint Bayesian hierarchical model

Under the assumption that there exists a finite representation of $X_{ig}(v) = \mu_g(v) + \sum_{k=1}^{K_g} \xi_{ik,g}\phi_{k,g}(v)$, the Bayesian hierarchical model is then formulated as,

$$Y_i(\tau) = \mu(\tau) + \sum_{g=1}^{G} \left(\int_0^\tau \theta_g(v,\tau) \left(\mu_g(v) + \sum_{k=1}^{K_g} \xi_{ik,g}\phi_{k,g}(v) \right) dv \right) + e_i(\tau), \quad e_i(\tau) \sim \mathcal{N}_1\left(0, \sigma_e^2\right),$$

$$\omega_{ig}(v) = \mu_g(v) + \sum_{k=1}^{K_g} \xi_{ik,g}\phi_k(v) + \eta_{ig}(v), \quad \eta_{ig}(v) \sim \mathcal{N}_1\left(0, \sigma_{\eta_g}^2\right).$$

A directed acyclic graph (DAG) illustrating the dependence between the observed data and model parameters can be found in Figure (3.1). For the joint estimation procedure, we derive the full conditional posterior distributions, $[\boldsymbol{\xi}_{i,g} | \widetilde{\boldsymbol{\omega}}_i, \mathbf{y}_i] \propto [\mathbf{y}_i | \boldsymbol{b}, \boldsymbol{\xi}_{i,g}][\boldsymbol{\xi}_{i,g} | \widetilde{\boldsymbol{\omega}}_i]$. It can be shown that the full conditional posterior distribution is given by

$$\pi(\boldsymbol{\xi}_{i,g} | \widetilde{\boldsymbol{\omega}}_{ig}, \mathbf{y}_i) \propto \exp\left(-\frac{1}{2\sigma_e^2} \sum_{j=1}^{T} \left(Y_i(\tau_j) - \sum_{u,k,g} \widetilde{X}_{igu}(\tau_j) f_k(\tau_j) b_{uk,g} \right)^2 \right) \times \mathcal{N}_{K_g}\left(\boldsymbol{\mu}_{\widetilde{\boldsymbol{\omega}}_{ig}}, \boldsymbol{\Sigma}_{\widetilde{\boldsymbol{\omega}}_{ig}} \right),$$

where $\boldsymbol{\mu}_{\widetilde{\boldsymbol{\omega}}_{ig}}$ and $\boldsymbol{\Sigma}_{\widetilde{\boldsymbol{\omega}}_{ig}}$ are defined in eq. (3.10), and from eq. (3.3),

$$\widetilde{X}_{igu}(\tau) = \sum_{v_i \in [0,\tau)} \sum_{k}^{K_g} \xi_{ik,g}\phi_{k,g}(v_r)\psi_u(v_r)\Delta(v_r).$$

The full conditional posterior distribution is intractable due to the presence of the FPC scores in the likelihood function of the regression model. Hence we set up a Metropolis-Hastings (M-H) within the Gibbs algorithm to sample from the distribution. Due to the long run-times of the Bayesian HFLM, the proposal distribution of the form $\mathcal{N}_1(0, \sigma^2)$ is not feasible as the variance parameter requires tuning. Hence, following Roberts and Rosenthal (2009), at each iteration, the proposal distribution is adaptively updated based on the current estimate of the posterior covariance matrix of the chain so far, $\boldsymbol{\Sigma}^{(m)}$. The proposal distribution at the n^{th} iteration is given by,

Algorithm 1: Gibbs sampling algorithm with substitution

```
1  for $m$ = 1,..., number of iterations do
       /* Obtain smooth estimates of the functional predictors */
2      for $g$ = 1, ..., $G$ do
           /* Sample the FPC scores */
3          for $i$ = 1, ..., $N$ do
4              for $k$ = 1,..., $K_g$ do
5                  Sample $\xi_{ik,g}|\widetilde{\boldsymbol{\omega}}_i, \cdot \sim \mathcal{N}_1\left(\frac{\lambda_{k,g}}{\lambda_{k,g}+\sigma^2_{\eta g}}\boldsymbol{\phi}_g^T\widetilde{\boldsymbol{\omega}}_{ig},\ \frac{\lambda_k\sigma^2_{\eta g}}{\lambda_{k,g}+\sigma^2_{\eta g}}\right)$
6              Set $X_{ig}(v) \longleftarrow \sum_{k}^{K_g}\xi_{ik,g}\psi_k(v)$
           /* Sample variance parameters of the smoothing process */
7          for $k$ = 1,..., $K_g$ do
8              Sample $\lambda_{k,g}|\cdot \sim \mathcal{IG}\left(0.5N,\ 0.5\sum_{i=1}^{N}\xi^2_{ik,g}\right)$
9          Sample $\sigma^2_{\eta g}|\cdot \sim \mathcal{IG}\left(i_1 + 0.5NS,\ i_2 + 0.5\sum_{i=1}^{N}(\widetilde{\boldsymbol{\omega}}_{ig} - \boldsymbol{\phi}_g\boldsymbol{\xi}_{i,g})^T(\widetilde{\boldsymbol{\omega}}_{ig} - \boldsymbol{\phi}_g\boldsymbol{\xi}_{i,g})\right)$,
            where $\widetilde{\omega}_{ig}(v) = \omega_{ig}(v) - \mu_g(v)$
       /* Update design matrix of the HFLM as defined above */
10     Update $\widetilde{\mathbf{R}}$
       /* Sample regression coefficients jointly */
11     Sample $\widetilde{\boldsymbol{\alpha}}|\cdot \sim \mathcal{N}_p(\boldsymbol{\mu}_{\widetilde{\alpha}}, \boldsymbol{\Sigma}_{\widetilde{\alpha}})$ where $\boldsymbol{\Sigma}_{\widetilde{\alpha}} = \left[\frac{1}{\sigma_e^2}\widetilde{\boldsymbol{R}}^{\mathrm{T}}\widetilde{\boldsymbol{R}} + \boldsymbol{D}_{\widetilde{\alpha}}\right]^{-1}$ and
       $\boldsymbol{\mu}_{\widetilde{\alpha}} = \frac{1}{\sigma_e^2}\boldsymbol{\Sigma}_{\widetilde{\alpha}}\widetilde{\boldsymbol{R}}^{\mathrm{T}}\boldsymbol{Y}$
       /* Sample the variance parameters of the regression model */
12     Sample $\sigma_e^2|\cdot \sim \mathcal{IG}\left(0.5NT + 0.001,\ \frac{1}{2}(\mathrm{vec}(\boldsymbol{Y}) - \widetilde{\mathbf{R}}\widetilde{\boldsymbol{\alpha}})^{\mathrm{T}}(\mathrm{vec}(\boldsymbol{Y}) - \widetilde{\mathbf{R}}\widetilde{\boldsymbol{\alpha}}) + 0.001\right)$
13     Sample $\sigma_\mu^2|\cdot \sim \mathcal{IG}\left(0.5K_\mu + 0.001,\ \frac{1}{2}\widetilde{\boldsymbol{\mu}}^{\mathrm{T}}\widetilde{\boldsymbol{\mu}}\right)$
14     for $g$=1,...,$G$ do
15         $\sigma^2_{b,g}|\cdot \sim \mathcal{IG}\left(0.5UK_Y + 0.001,\ \frac{1}{2}\widetilde{\boldsymbol{b}}_g^{\mathrm{T}}\widetilde{\boldsymbol{b}}_g\right)$
```

$$Q_m(x,\cdot) = (1-\beta)\mathcal{N}_{K_g}\left(x, 2.38^2\boldsymbol{\Sigma}^{(m)}/K_g\right) + \beta\mathcal{N}_{K_g}\left(x, (0.1)^2\mathbf{I}_{K_g}/K_g\right),$$

where x is the value of $\boldsymbol{\xi}_{i,g}$ at the $(m-1)^{th}$ iteration and $\boldsymbol{\Sigma}^{(m)} = \boldsymbol{\Sigma}^{(m)}_{\widetilde{\boldsymbol{\omega}}_{ig}}$ defined in eq. (3.10). Roberts and Rosenthal (2009) suggest setting $\beta = 0.05$. This adaptive proposal was inspired by the work of Haario et al. (2001) who showed that an optimal proposal covariance is given by $2.38^2\boldsymbol{\Sigma}/d$ where $\boldsymbol{\Sigma}$ is the covariance matrix of the parameters. The acceptance probability is the ratio between the conditional posterior densities evaluated at the new proposal value of the parameters generated by the proposal distribution, $\boldsymbol{\xi}^*_{i,g}$, and the value of the parameters at the previous iteration, $\boldsymbol{\xi}^{(m-1)}_{i,g}$ because of the symmetricity of the proposal distribution, $Q_m(x,\cdot)$,

$$r = \min(\exp\{\log(\pi(\boldsymbol{\xi}^*_{i,g}|\widetilde{\boldsymbol{\omega}}_i, \mathbf{y}_i) - \log(\pi(\boldsymbol{\xi}^{(m-1)}_{i,g}|\widetilde{\boldsymbol{\omega}}_i, \mathbf{y}_i))\}, 1).$$

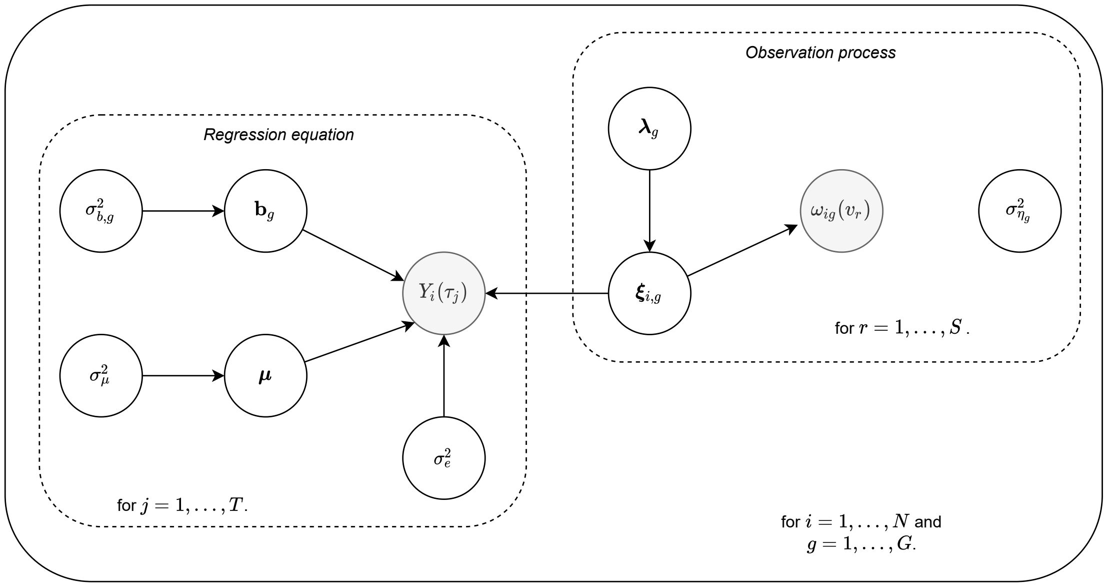


Figure 3.1. *Directed acyclic graph (DAG) illustrating the conditional dependence between the model parameters and observed data of the hierarchical historical functional linear model. Shaded nodes represent the observed data, and unshaded nodes the parameters of the proposed HFLM.*

3.4.3 Fast sampling from high dimensional Gaussian distributions

The regression model requires sampling from a multivariate Gaussian distribution of high dimension as shown in §3.2. It requires the inversion of a $(K_\mu + GUK_Y) \times (K_\mu + GUK_Y) = p \times p$ covariance matrix which can rapidly increase as the dimension of the marginal bases

Algorithm 2: Metropolis-within-Gibbs sampling algorithm for joint estimation

```
1  for $m$ = 1,..., number of iterations do
       /* Obtain smooth estimates of the functional predictors */
2      for $g$ = 1, ..., $G$ do
           /* Sample the FPC scores */
3          for $i$ = 1, ..., $N$ do
               /* Generate a proposal from the proposal distribution Q */
4              $\boldsymbol{\xi}^*_{i,g} \sim Q_m(\boldsymbol{\xi}^{(m-1)}_{i,g}, \cdot)$ ;
               /* Calculate the acceptance probability */
5              $r = \min(\exp\{\log(\pi(\boldsymbol{\xi}^*_{i,g}|\widetilde{\boldsymbol{\omega}}_i, \mathbf{y}_i) - \log(\pi(\boldsymbol{\xi}^{(m-1)}_{i,g}|\widetilde{\boldsymbol{\omega}}_i, \mathbf{y}_i))\}, 1)$
6              $u \sim \mathcal{U}(0,1)$
7              if $u \leq r$ then
                   /* Accept new solution */
8                  $\boldsymbol{\xi}^{(m)}_{i,g} \longleftarrow \boldsymbol{\xi}^*_{i,g}$ ,    $X^{(m)}_{ig}(v) \longleftarrow \sum_{k}^{K_g} \xi^{(m)}_{ik,g}\psi_{k,g}(v)$
9              else
10                 $\boldsymbol{\xi}^{(m)}_{i,g} \longleftarrow \boldsymbol{\xi}^{(m-1)}_{i,g}$,    $X^{(m)}_{ig}(v) \longleftarrow \sum_{k}^{K_g} \xi^{(m-1)}_{ik,g}\psi_{k,g}(v)$
           /* Sample variance parameters of the smoothing process */
11         for $k$ = 1,..., $K_g$ do
12             Sample $\lambda_{k,g}|\cdot \sim \mathcal{IG}\left(N/2,\ 0.5\sum_{i=1}^{N}\xi^2_{ik,g}\right)$
13         Sample $\sigma^2_{\eta_g}|\cdot \sim \mathcal{IG}\left(i_1 + 0.5NS,\ i_2 + 0.5\sum_{i=1}^{N}(\widetilde{\boldsymbol{\omega}}_{ig} - \boldsymbol{\phi}_g\boldsymbol{\xi}_{i,g})^T(\widetilde{\boldsymbol{\omega}}_{ig} - \boldsymbol{\phi}_g\boldsymbol{\xi}_{i,g})\right)$, where
           $\widetilde{\omega}_{ig}(v) = \omega_{ig}(v) - \mu_g(v)$
       /* Update design matrix of the HFLM as defined above */
14     Update $\widetilde{\mathbf{R}}$ /* Sample regression coefficients jointly */
15     Sample $\widetilde{\boldsymbol{\alpha}}|\cdot \sim \mathcal{N}_p(\boldsymbol{\mu}_{\widetilde{\boldsymbol{\alpha}}},\ \boldsymbol{\Sigma}_{\widetilde{\boldsymbol{\alpha}}})$ where $\boldsymbol{\Sigma}_{\widetilde{\boldsymbol{\alpha}}} = \left[\frac{1}{\sigma^2_e}\widetilde{\boldsymbol{R}}^{\mathrm{T}}\widetilde{\boldsymbol{R}} + \boldsymbol{D}_{\widetilde{\boldsymbol{\alpha}}}\right]^{-1}$ and $\boldsymbol{\mu}_{\widetilde{\boldsymbol{\alpha}}} = \frac{1}{\sigma^2_e}\boldsymbol{\Sigma}_{\widetilde{\boldsymbol{\alpha}}}\widetilde{\boldsymbol{R}}^{\mathrm{T}}\boldsymbol{Y}$
       /* Sample the variance parameters of the regression model */
16     Sample $\sigma^2_e|\cdot \sim \mathcal{IG}\left(0.5NT + 0.001,\ \frac{1}{2}(\mathrm{vec}(\boldsymbol{Y}) - \widetilde{\mathbf{R}}\widetilde{\boldsymbol{\alpha}})^{\mathrm{T}}(\mathrm{vec}(\boldsymbol{Y}) - \widetilde{\mathbf{R}}\widetilde{\boldsymbol{\alpha}}) + 0.001\right)$
17     Sample $\sigma^2_\mu|\cdot \sim \mathcal{IG}\left(0.5K_\mu + 0.001,\ \frac{1}{2}\widetilde{\boldsymbol{\mu}}^{\mathrm{T}}\widetilde{\boldsymbol{\mu}}\right)$
18     for $g$=1,...,$G$ do
19         $\sigma^2_{b,g}|\cdot \sim \mathcal{IG}\left(0.5UK_Y + 0.001,\ \frac{1}{2}\widetilde{\boldsymbol{b}}_g^{\mathrm{T}}\widetilde{\boldsymbol{b}}_g\right)$
```

of the tensor product increases. For faster computation, a Cholesky based implementation proposed by Rue (2001) is considered. As shown earlier, the conditional posterior distribution of the regression parameters are given by,

$$\widetilde{\boldsymbol{\alpha}}|\cdot \sim \mathcal{N}_p(\boldsymbol{\mu}_{\tilde{\boldsymbol{\alpha}}}, \boldsymbol{\Sigma}_{\tilde{\boldsymbol{\alpha}}}),$$

where $\boldsymbol{\Sigma}_{\tilde{\boldsymbol{\alpha}}} = \left[\frac{1}{\sigma_e^2}\widetilde{\boldsymbol{R}}^{\mathrm{T}}\widetilde{\boldsymbol{R}} + \boldsymbol{D}_{\tilde{\boldsymbol{\alpha}}}\right]^{-1}$ and $\boldsymbol{\mu}_{\tilde{\boldsymbol{\alpha}}} = \frac{1}{\sigma_e^2}\boldsymbol{\Sigma}_{\tilde{\boldsymbol{\alpha}}}\widetilde{\boldsymbol{R}}^{\mathrm{T}}\operatorname{vec}(\boldsymbol{Y})$. Now let $\boldsymbol{\ell} = \frac{1}{\sigma_e^2}\widetilde{\boldsymbol{R}}^{\mathrm{T}}\operatorname{vec}(\boldsymbol{Y})$, the posterior distribution is of the form,

$$\widetilde{\boldsymbol{\alpha}}|\cdot \sim \mathcal{N}_p\left(\mathbf{Q}_{\tilde{\boldsymbol{\alpha}}}^{-1}\boldsymbol{\ell},\ \mathbf{Q}_{\tilde{\boldsymbol{\alpha}}}^{-1}\right),$$

where $\mathbf{Q}$ denotes the precision matrix. To sample from this distribution, we first compute the band Cholesky decomposition $\mathbf{Q} = \mathbf{L}\mathbf{L}^{\mathrm{T}}$. The following systems are solved: using forward substitution, $\mathbf{L}\mathbf{v} = \boldsymbol{\ell}$, and using backward subsitution, $\mathbf{L}^{\mathrm{T}}\boldsymbol{\mu} = \mathbf{v}$ and $\mathbf{L}^{\mathrm{T}}\mathbf{y} = \mathbf{z}$ where $\mathbf{z} \sim \mathcal{N}_p(\mathbf{0}, \mathbf{I}_p)$. The sample is then obtained using $\boldsymbol{\alpha} = \boldsymbol{\mu} + \mathbf{y}$ (Rue, 2001).

3.5 Identifiability in functional linear models

A model is non-identifiable when there exists more than one unique solutions that can solve the regression problem. Non-identifiability can lead to spurious regression surface estimates despite good fitted values making the interpretability of the fitted models unreliable (Scheipl and Greven, 2016). In their work, Scheipl and Greven (2016) found that non-identifiability can be further exacerberated if the functional covariates are pre-smoothed. However, in cases where estimating the functional process is of importance, the authors suggest that non-identiability can be overcome by selecting $K_g \geq U$ where K_g is the number of FPC, and U is the dimension of the marginal basis $\boldsymbol{\psi}$ of the tensor product. They further suggest that full-rank penalty matrices should be considered with the fixed terms in the basis functions having a corresponding small penalty value instead of zero. In their work, Scheipl and Greven (2016) also present some diagnostic checks to identify potential identifiability which was adapted by Brockhaus et al. (2017) to flag non-identifiability in HFLMs by first identifying numeric rank deficiency in the design matrix and secondly by checking for the uniqueness of the regression coefficients under smoothness constraints. These diagnostic checks are implemented in the `refund` `R` package (Goldsmith et al., 2020).

3.6 Prediction, Inference and Model Checking

3.6.1 Estimates and credibility intervals

The estimates of the parameters are evaluated using the mean of the post burn-in MCMC samples. The point-wise credibility interval (PWCI) for the functional predictors, fitted values and predictions are obtained by calculating the quantiles of the post burn-in MCMC samples at probabilities $\alpha/2$ and $1-\alpha/2$.

3.6.2 Predicting response trajectories

We derive the posterior predictive distributions to generate forecasts. In the case that a smoothing step is added to the estimation procedure, we again generate forecasts based on a two-step procedure which involves 1) obtain estimates of the smoothed out-of-sample predictors, and 2) sample from the posterior predictive distribution.

Denote the parameters of the model by,

$$\mathbf{\Phi} = [\mu(\tau), \theta_1(v,\tau), \ldots, \theta_G(v,\tau), \sigma_e^2, \sigma_{b,1}^2, \ldots, \sigma_{G,1}^2].$$

Let N^* denote the number of future observations, then $\text{vec}(\mathbf{Y}^*)$ is the vectorisation of the matrix of future values, $\mathbf{Y}_i^* = [Y_i^*(\tau_j)]_{\substack{i=1,\ldots,N^* \\ j=1,\ldots,\mathrm{T}}}$ and the g^{th} functional covariate of the future observations is given by, $\mathbf{X}_g^* = [X_{ig}(v_r)]_{\substack{i=1,\ldots,N^* \\ r=1,\ldots,S}}$.

The predictive posterior distribution is given by,

$$[\text{vec}(\mathbf{Y}^*) \mid \text{vec}(\mathbf{Y}), \mathbf{X}^*, \mathbf{X}, \mathbf{\Phi}].$$

Since $\text{vec}(\mathbf{Y}^*)$ is independent of $\text{vec}(\mathbf{Y})$ and $\mathbf{X}$, then,

$$[\text{vec}(\mathbf{Y}^*) \mid \text{vec}(\mathbf{Y}), \mathbf{X}^*, \mathbf{X}, \mathbf{\Phi}] \propto [\text{vec}(\mathbf{Y}^*) \mid \mathbf{X}^*, \mathbf{\Phi}] \times [\mathbf{\Phi} \mid \text{vec}(\mathbf{Y}), \mathbf{X}],$$

where

$$Y_i^*(\tau) \sim \mathcal{N}_1\left(\mu(\tau) + \sum_{g=1}^{G} \theta_g(v,\tau) X_{ig}^*(v)dv, \sigma_e^2\right).$$

The distribution $[\mathbf{\Phi} \mid \text{vec}(\mathbf{Y}), \mathbf{X}]$ is the conditional posterior distribution of the model parameters. In the MCMC algorithm, we iteratively draw samples from the conditional posterior distributions. The sampling algorithm is given as follows,

At each iteration m,

1. Generate samples for the smooth functional covariates. The smoothing procedure is performed as in (1) and (2) for the two-stage and joint HFLM respectively to obtain estimates of $\mathbf{X}^{*^{(m)}}$.

2. Generate samples from the conditional posterior distribution, $[\mathbf{\Phi}^{(m)} | \operatorname{vec}(\mathbf{Y}), \mathbf{X}]$.

3. Sample the forecasts from, $[\operatorname{vec}(\mathbf{Y}^*) \mid \operatorname{vec}(\mathbf{X}^{*^{(m)}}), \mathbf{\Phi}^{(m)}]$.

3.6.3 Autocorrelation of residual processes

Classical techniques used for linear regression models can be extended to functional linear models to assess normality of the residuals such as visually inspecting histograms and quantile-quantile plots. Hence, here we focus on how to identify functional partial autocorrelation functions (FPACF) of the residual processes used to measure the correlation between two lags of the functional time series after controlling for the other lags. Denote the residual processes by $e_t = \{e_t(\tau_j)\ j = 1, \ldots, T\}$. The partial autocovariance between e_t and e_{t+h} is the autocovariance of the residuals of e_t and e_{t+h} produced by performing a penalised functional regression on the intermediate lags $\mathbf{e}_{t,h} = [e_{t+1}, \ldots, e_{t+h-1}]$ and evaluating the corresponding L^2-norm of the residuals of the FLMs which we denote by $\hat{\gamma}_{1,h}$ and $\hat{\gamma}_{2,h}$ (Mestre et al., 2021). An estimate of the partial autocovariance at lag h of the functional time-series is thus given by (Mestre et al., 2021),

$$\hat{C}_{h,h}(u,v) = \frac{1}{T-h} \sum_{t=1}^{T-h} \left[e_t(u) - \hat{\Psi}_{h,1}\left(\boldsymbol{e}_{t,h}\right)(u) \right] \left[e_{t+h}(v) - \hat{\Psi}_{h,2}\left(\boldsymbol{e}_{t,h}\right)(v) \right]$$

$$\hat{\gamma}_{1,h} = \left(\int \frac{1}{T-h} \sum_{t=1}^{T-h} \left[e_t(u) - \hat{\Psi}_{h,1}\left(\boldsymbol{e}_{t,h}\right)(u) \right]^2 du \right)^{1/2}$$

$$\hat{\gamma}_{2,h} = \left(\int \frac{1}{T-h} \sum_{t=1}^{T-h} \left[e_{t+h}(u) - \hat{\Psi}_{h,2}\left(\boldsymbol{e}_{t,h}\right)(u) \right]^2 du \right)^{1/2},$$

where $\hat{\Psi}_{h,j}$ denotes the operator obtained by performing the regularised functional linear regression (Mestre et al., 2021). The empirical partial autocorrelation may then be defined as,

$$\hat{\rho}_{h,h} = \frac{\left\| \hat{C}_{h,h} \right\|}{\hat{\gamma}_{1,h} \hat{\gamma}_{2,h}}.$$

Under the white noise assumption of the residual processes, Mestre et al. (2021) also proposed an upper bound for the prediction interval of the partial autocorrelation function which can be used to assess whether the residuals are white noise processes at a specified lag.

CHAPTER 4

Simulation Study

To assess the effect of smoothing the functional covariates on inferences, we conduct a simulation study by generating data sets with varying noise levels in both the functional response and predictor processes and number of curve data N. Since penalised Bayesian functional regression models are commonly used approaches, our benchmark model is the Bayesian HFLM, simply referred to as HFLM, which does not account for measurement error in the functional predictor processes, and is compared to the measurement error models described in §3.3 with estimation procedures outlined in §3.4 namely estimation under a two-stage analysis which we refer to as the two-stage HFLM, and the joint hierarchical model referred to as joint HFLM.

4.1 Data Generating Process

Similar to Meyer et al. (2015) and Pomann et al. (2016), we generate data from the following function,

$$Y_i(\tau) = f_i(\tau) + e_i(\tau) = \mu(\tau) + \int_0^\tau \theta_1(v,\tau)X_{1i}(v)dv + \int_0^\tau \theta_2(v,\tau)X_{2i}(v)dv + e_i(\tau),$$

on $T = 25$ equally spaced intervals in $[0,1]$. The regression surfaces are set as,

$$\begin{aligned}\theta_1(v,\tau) &= \cos(2\pi\tau)\cos(\pi v),\\ \theta_2(v,\tau) &= \frac{1}{\pi\times 0.4\times 0.3}\exp\left(-\frac{(\tau-0.2)^2}{0.3^2} - \frac{(v-0.3)^2}{0.4^2}\right) + \\ &\quad \frac{1}{\pi\times 0.4\times 0.3}\exp\left(-\frac{(v-0.7)^2}{0.4^2} - \frac{(\tau-0.9)^2}{0.3^2}\right).\end{aligned}$$

for $v \leq \tau$. The first surface θ_1 is irregular with negative and positive peaks, and the second surface θ_2 is a bimodal lagged surface with two peaks along $v - \tau = 0$ which decreases to 0 as $|v - \tau|$ increases. The intercept term is set to $\mu(\tau) = \exp(-(\tau - 0.5)^2)$. The first functional predictor is generated as $X_{1i}(v) = w_{1i}\sin(2\pi v) + z_{1i}\cos(2\pi v)$ where $w_{1i}, z_{1i} \sim \mathcal{N}_1(10, 4)$. The predictor is then standardised by the overall mean and standard deviation. Similarly, the second predictor is simulated as $X_{2i}(v) = u_{1i}\sin(\pi\tau) + v_{1i}\cos(\pi\tau) + u_{2i}\sin(2\pi\tau) + v_{2i}\cos(2\pi\tau)$ where $u_{1i}, v_{1i} \sim \mathcal{N}_1(0,1)$ and $u_{2i}, v_{2i} \sim \mathcal{N}_1(0, 0.25)$. The predictor processes are then perturbed by noise, $\omega_{ig}(v) =$

$X_{ig}(v) + \eta_{ig}(v)$ where $\eta_{ig}(v) \sim \mathcal{N}_1(0, \sigma_x^2)$, which we use to fit our model. The functional predictors were generated on the same grid as the functional response, that is, on $S = 25$ equally spaced intervals in $[0, 1]$. The integral is evaluated using Riemann sums. We simulate the i.i.d. errors from $e_i(\tau) \sim \mathcal{N}_1(0, \sigma_e^2)$ by specifying the expected signal-to-ratio, eSNR $= \sum_{i,j}(\mathbb{E}[Y_i(\tau_j)) - \sum_{i}\mathbb{E}(Y_i(\tau_j))/N)^2/(\sigma_e^2(N-1)T)$ (Pomann et al., 2016). We vary eSNR $\in \{2, 10, 50\}$, $\sigma_x^2 \in \{0.01, 0.1, 1\}$ and $N = \{25, 50, 100\}$. We generate 50 data sets for each configuration. Similarly, we generate a test data set with $N = 10$ observations. An illustration of a simulated data set can be found in Figure (4.1).

To ensure convergence of the MCMC chain, we run 10000 iterations of the MCMC sampler, and only retain the last 50% of the samples. For the HFLM, we randomly set starting parameters from $\mathcal{U}(0, 1)$. For the measurement error models, we generate starting parameters for the regression coefficients from a Gaussian distribution with mean equal to the posterior means and twice the standard deviations of the MCMC chains of the HFLM. Convergence was assessed using the Gelman-Rubin's test for convergence from 3 chains using the `coda` package (Plummer et al., 2006) for a small subset of simulation runs, and a visual inspection of the traceplots, ACF and densities of the post burn-in samples on a subset of parameters. and The results can be found in Appendix F. The coefficients of the regression model are estimated as the posterior mean of the post burn-in samples. The following basis functions and dimensions are used:

- We use 10 thin-plate splines functions for the functional intercept term with a second-order derivative penalty.
- The marginal bases for the tensor product are cubic B-splines along v and τ with $U = K = 10$. Second-order derivative penalties are used for the basis functions.
- The number of FPCs, K_g, is also set to 10 for $g = \{1, 2\}$.

To ensure identifiability of the design matrix from the simulation set up, we perform the diagnostics on $X_{ig}(v)$ i.e., the functional predictor processes under no noise. Non-identifiability was not flagged when tested using the `refund` package (Goldsmith et al., 2020).

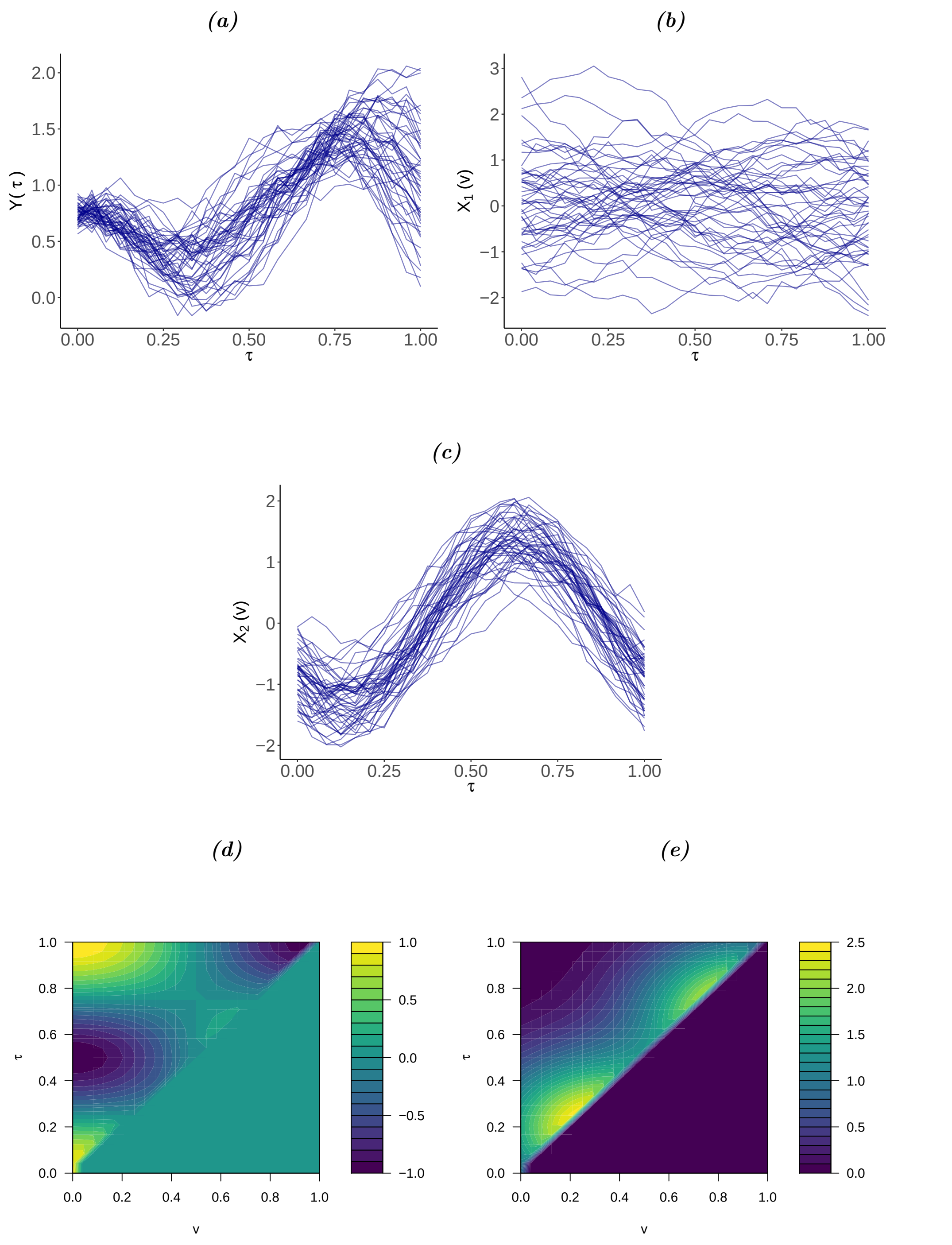


Figure 4.1. *Illustration of a simulated data set for $N = 50$, $T = 25$. (a) The functional response generated from $Y_i(\tau) = f_i(\tau) + e_i(\tau)$ with eSNR = 10. (b)-(c) First and second functional predictor generated from $\omega_{ig}(v) = X_{ig}(v) + \eta_{ig}(v)$ where $\eta_{ig}(v) \sim \mathcal{N}(0, 0.01)$ for v, i, g. (d)-(e) First and second true coefficient surfaces.*

4.2 Methods of comparison

To compare the performances of the models, we evaluate the integrated mean-squared error of each surface,

$$IMSE(\theta_g) = \int\int (\hat{\theta}_g(v,\tau) - \theta_g(v,\tau))^2 dv d\tau,$$

which we approximate using $\sum_{j,r}(\hat{\theta}_g(v_r,\tau_j) - \theta_g(v_r,\tau_j))^2/(TS)$. For each generated data set, we evaluate the point-wise credibility intervals of $\theta_g(v_r,\tau_j)$. Denote the lower and upper bounds of the credibility interval at the α-level of $\theta_g(v_r,\tau_j)$ by $\left[\hat{\theta}_g^{\alpha/2}(v_r,\tau_j),\, \hat{\theta}_g^{1-\alpha/2}(v_r,\tau_j)\right]$ for each $r = 1,\ldots,S$ and $j = 1,\ldots,T$, the coverage is equal to,

$$\begin{cases} 1, & \text{if } \hat{\theta}_g^{\alpha/2}(v_r,\tau_j) \leq \theta_g(v_r,\tau_j) \leq \hat{\theta}_g^{1-\alpha/2}(v_r,\tau_j), \\ 0, & \text{otherwise.} \end{cases}$$

The empirical coverage of the surfaces is then calculated by first averaging the coverage over the historical surface and then over all 50 simulated data sets (Meyer et al., 2021). The IMSE and empirical coverage of the estimated surfaces are only evaluated at time points $v \leq \tau$ to take into account the triangular shape of the surface.

We calculate the in-sample integrated mean-squared error of the fitted values relative to the true underlying function,

$$IMSE(Y) = \frac{1}{N}\sum_{i}^{N}\int_{\mathcal{T}}(\hat{Y}_i(\tau) - f_i(\tau))^2 d\tau.$$

The point-wise credibility intervals are obtained for $\hat{Y}_i(\tau_j)$ for $i = 1,\ldots,N$, $j = 1,\ldots,T$. The empirical coverage is obtained by first averaging over the curve data at each time-point and then over all simulated data sets.

Out-of-sample, we calculate the IMSE of the predictions relative to the observed values in the test set,

$$IMSE(Y_{pred}) = \frac{1}{N}\sum_{i}^{N}\int_{\tau}\left(\hat{Y}_i(\tau) - Y_{pred}(\tau)\right)^2 d\tau.$$

The empirical coverage is also obtained by first averaging over the curve data at each time-point and then over all simulated data sets.

The computations were performed using facilities provided by the University of Cape

Town's ICTS High Performance Computing team: hpc.uct.ac.za.

4.3 Results

We only present results for $N = 50$ here, while results for $N = \{25, 100\}$ can be found in Appendix E. For $N = 50$ and $T = 25$ with the total number of regression coefficients to be estimated $p = 210$ and 10000 iterations in the Gibbs sampler, the run-time for 1) the HFLM without measurement error is approximately 75.2 seconds, 2) the two-stage model, 553.7 seconds and 3) the joint model, 673.7 seconds (8GB MacBook 1.4GHz Intel Core i5).

The IMSE of the surfaces can be found in Figures (4.2)-(4.3). For the first historical surface, the two-stage and joint HFLM both produce the lowest IMSE thus outperforming the classical HFLM under the assumption of no noise. We however note poor estimates of the second historical surface from the joint estimation model when the variance of the noise increases. This problem is further exacerbated when the error variance in the functional response is small. In general, the IMSE of the surfaces is larger for the HFLM than the joint and two-stage model, except when $\sigma_x^2 = 1$ and eSNR$= 50$, the joint estimation model produces estimates with high variation.

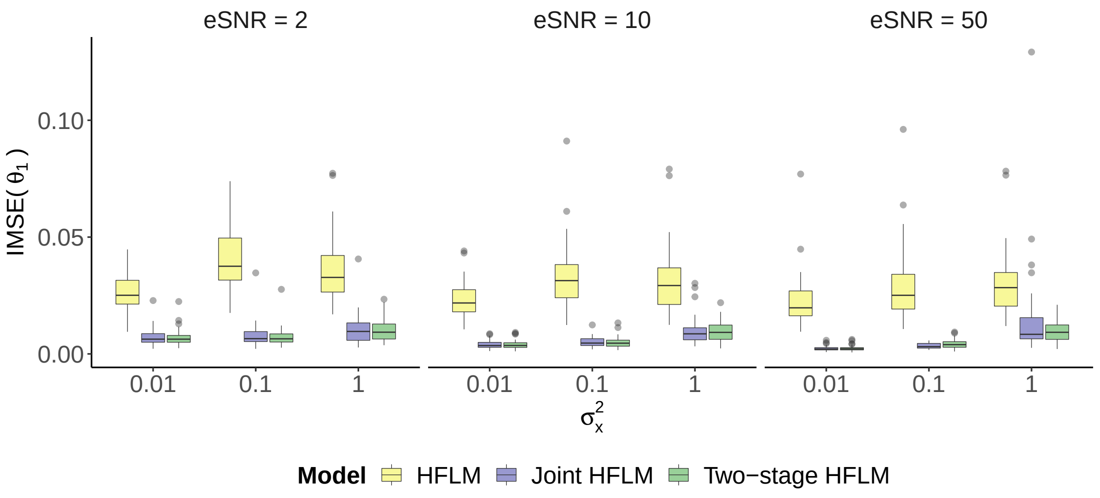


Figure 4.2. *IMSE of the first historical surface, $\theta_1(v, \tau)$.*

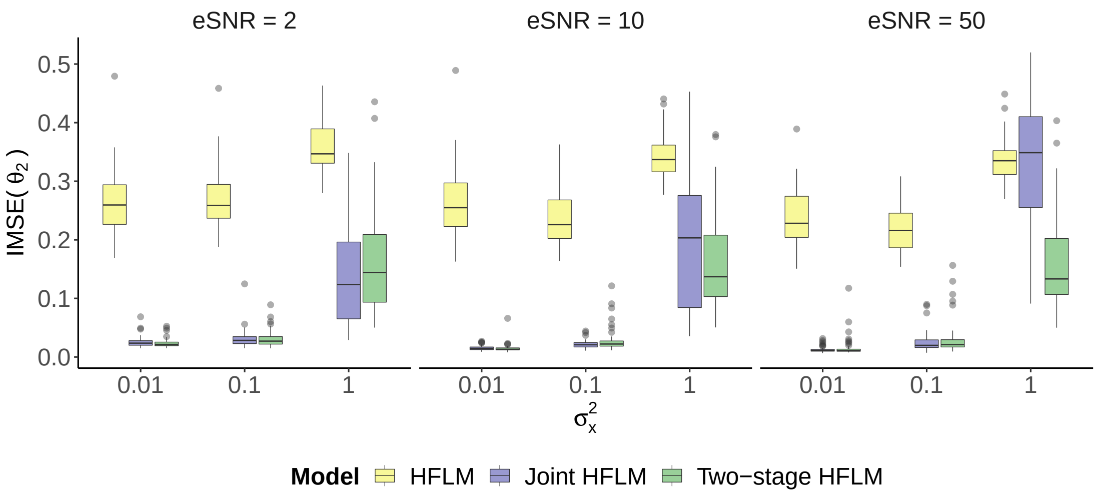


Figure 4.3. *IMSE of the second historical surface, $\theta_2(v, \tau)$.*

The empirical coverage of the surfaces can be found in Figures (4.4)-(4.5). For the first surface, the empirical coverage remains above 95% across all scenarios for the measurement error models, while for the HFLM, the coverage decreases below 95% when σ_x^2 increases. For the second surface, empirical coverage also decreases for large σ_x^2 however we find that the two-stage model provides higher coverage than the joint estimation model when $\sigma_x^2 = 1$ and eSNR$= 50$, that is when the signal-to-noise ratio in the functional response is high and the error in the predictor process is large.

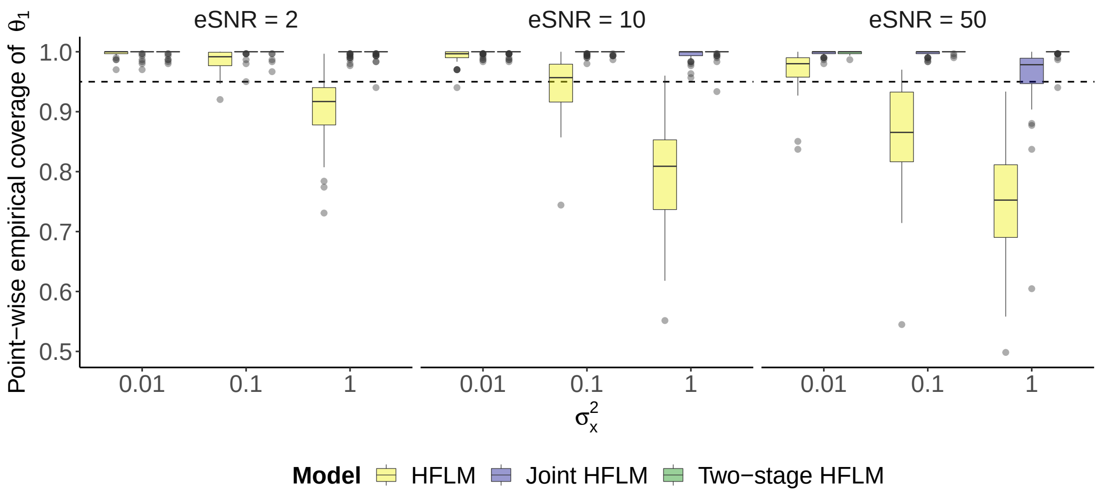


Figure 4.4. *Empirical coverage of the first historical surface, $\theta_1(v, \tau)$.*

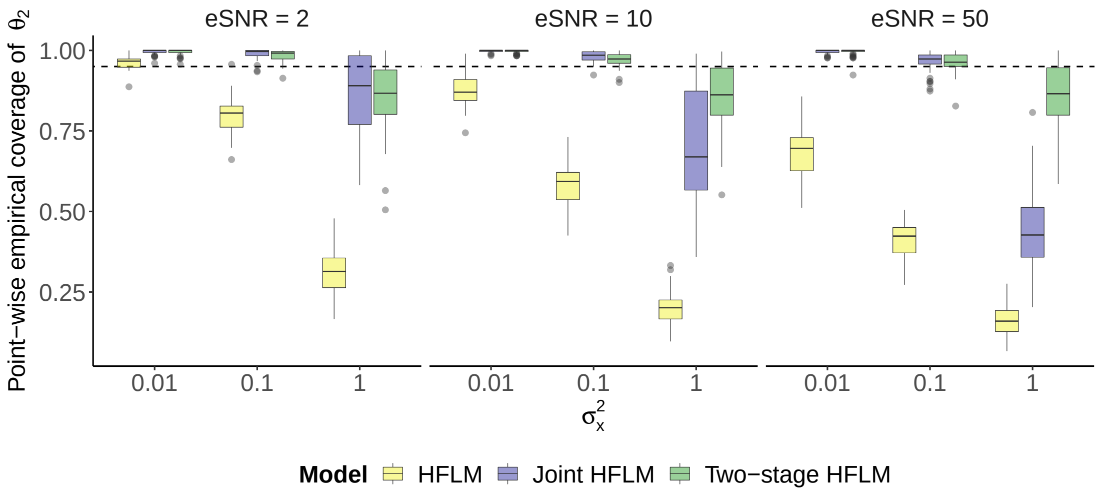


Figure 4.5. *Empirical coverage of the second historical surface,* $\theta_2(v, \tau)$.

Despite poorer coverage and estimates of second historical surface than the two-stage HFLM, the joint estimation model produces the best fitted values with a low IMSE when compared to the true underlying function $f(\tau)$. However the empirical coverage of the measurement error models is higher than the HFLM which suggests that they effectively capture the uncertainty in the predictor processes.

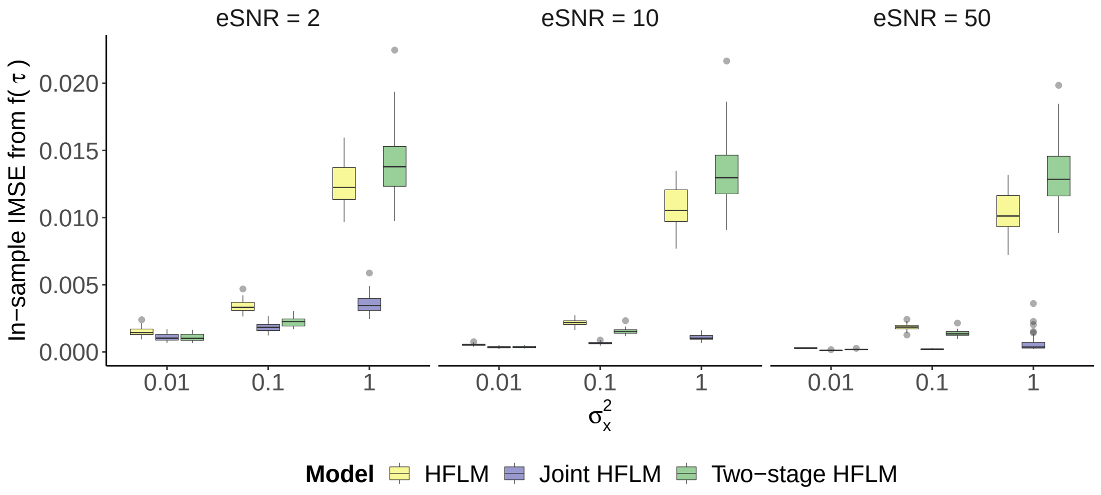


Figure 4.6. *In-sample IMSE relative to the true underlying function,* $f(\tau)$.

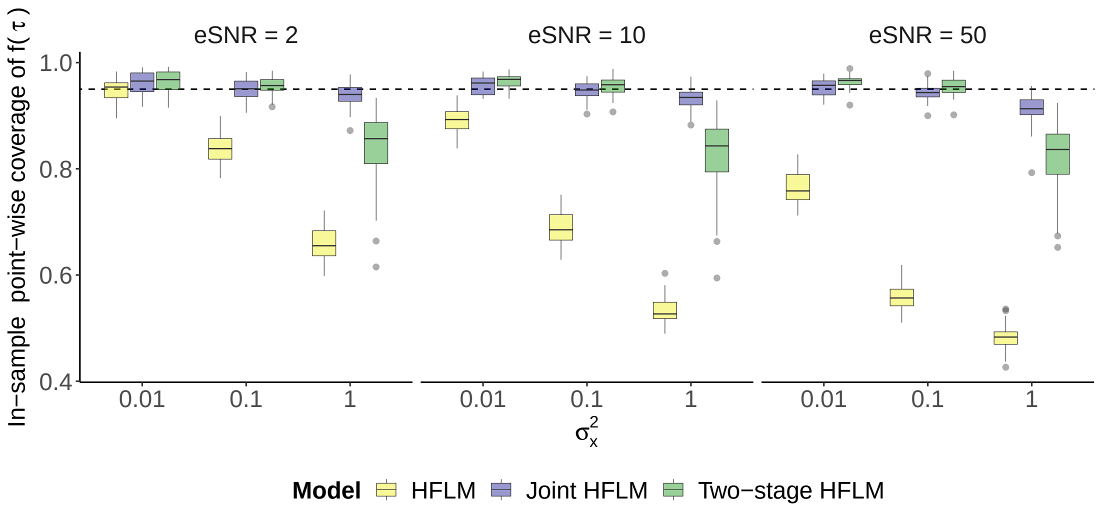


Figure 4.7. *In-sample empirical coverage the true underlying function, $f(\tau)$.*

We now compare the performance of the models out-of-sample. With lower IMSE, the measurement error models generally outperform the HFLM model out-of-sample. However, the two-stage model produces lower IMSE than the joint model despite the joint HFLM performing better in-sample. The empirical coverage of the two-stage model is consistently above 95% for large σ_x^2 while the joint estimate provides poor coverage of the observed values which worsens when the eSNR of the functional response increases while the error in the predictor process also increases.

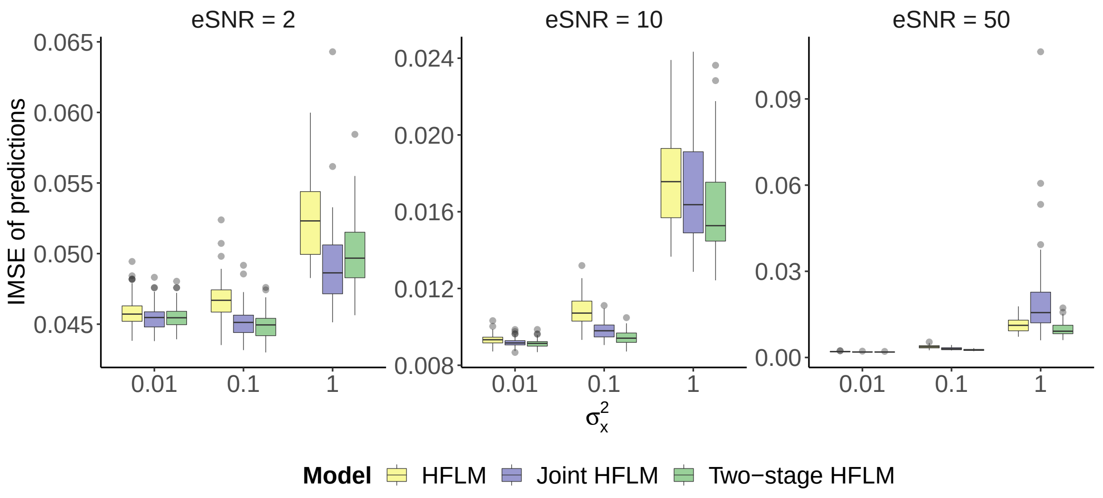


Figure 4.8. *Out-of-sample IMSE relative to the observed test data, $Y_{pred}(\tau)$.*

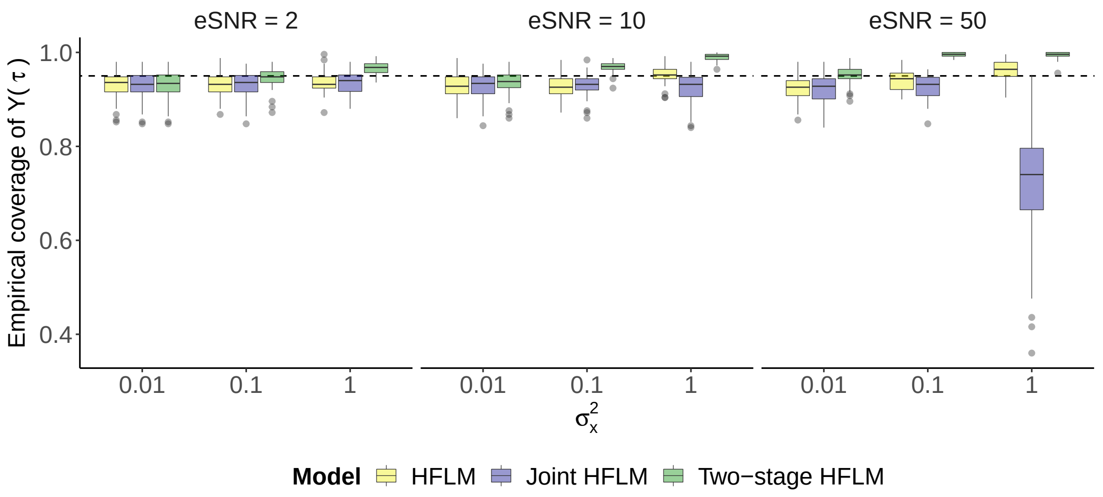


Figure 4.9. *Out-of-sample empirical coverage of the observed test data, $Y_{pred}(\tau)$.*

Experiments show that accounting for measurement error is crucial for accurate estimation and uncertainty quantification. However, we found that the different estimation procedures produce different results and are sensitive to the variance parameters. In general, we find that when eSNR and σ_x^2 are both large, the joint estimation model performs poorly for both the recovery of the true historical surfaces and predictions out-of-sample. Under these conditions, despite poor estimates of the true historical surface with low coverage, the joint HFLM does very well in-sample. We suspect that the joint HFLM is prone to overfitting. The acceptance probability of the FPC scores is on average 16%. This low acceptance rate suggests the MCMC samples of the FPC scores do not mix well. We believe that the M-H step tends to accept FPC scores that support the regression model at the expense of the observation equation, which in turn leads to poor estimates of the true historical surface.

Similar conclusions were drawn for $N = \{25, 100\}$ as shown in Appendix E. Furthermore, as N increases, the IMSE of the historical surfaces and predictions obtained from the HFLM decrease and the performance becomes closer to that of the measurement error models. However, their corresponding empirical coverages are lower than the measurement error models which reflects the fact that the HFLM understates the uncertainty in the estimates.

It is evident from the analysis made above that the measurement errors should be implemented when the presence of sizeable noise is suspected in the predictor processes. However, there are strong arguments in favour of the two-stage Bayesian model over the joint model. The two-stage model is less prone to numerical problems and overfitting, and it further is easier to implement as it does not require the choice of a proposal distribution in a Metropolis-Hastings step. Additionally, this estimation procedure produces accurate estimates of the historical surfaces and forecasts with high coverage while also accounting for the uncertainty in the functional predictor processes.

CHAPTER 5

Application to air pollution forecasting

Air quality management aims to actively monitor the levels of ambient pollutants which can be harmful to population health, the environment and infrastructure. In order to effectively manage air pollution, the City of Cape Town have set up a network of ambient air quality monitoring (AQM) stations across Cape Town, South Africa. There are currently 11 AQM stations in different locations of the City that collect data on nitrogen dioxide (NO_2), sulphur dioxide (SO_2), ozone (O_3), carbon monoxide (CO), benzene (CH_4) and particulate Matter (PM) (Western Cape Government, 2019). Air pollution data are publicly available at the South African Air Quality Information System (SAAQIS) website which serves as a common platform for air quality monitoring in South Africa (SAAQIS et al., 2021).

The purpose of this chapter is to investigate the performance of functional linear models to forecast atmospheric Ozone levels ($\mu g/m^3$) using meteorological information such as temperature, relative humidity, and wind speed. To this end, we fit our Bayesian HFLMs with and without modelling the functional covariates to the data set. We benchmark our HFLMs against the FARX(1) model of Damon and Guillas (2002) which has shown promise for this application, and the functional autoregressive model (Hyndman-Ullah FAR(1)) of Hyndman and Ullah (2007). The FARX(1) is estimated using a truncated estimation of the linear covariance and auto-covariance operator based on the first j eigenfunctions (ref. 2.3). A limitation of the FARX(1) model is that it depends on the current full day's functional predictor processes implying that it allows predictors at time points $s > \tau$ to influence the current outcome at time point τ. The FARX(1) model also does not take into account the measurement error in the functional predictors. The Hyndman-Ullah FAR(1) model on the other hand does not include exogeneous variables and a time-series model such as an autoregressive integrated moving average (ARIMA) or exponential smoothing is performed on the estimated FPC scores of the functional observations.

This chapter is outlined as follows. In §5.1, we describe the air pollution data set used for the analysis, and the pre-processing techniques used to ensure Gaussianity and to handle missing data. The implementation of the models fitted and methods of comparison are discussed in §5.2. The results of this study are presented in §5.3.

5.1 Description of the data

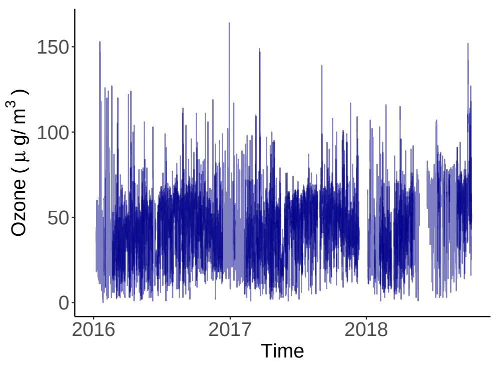
5.1.1 Exploratory data analysis

The Plattekloof air quality monitoring (AQM) station was chosen for this case study as the station had the least number of large missing blocks of data recorded in the City of Cape Town. The data consist of atmospheric ozone ($\mu g/m^3$) measured hourly ($T = 24$) from 01/01/2016 to 09/10/2018 at the Plattekloof AQM station. Observations after 09/10/2018 were discarded since the meteorological information recorded were erroneous since a constant high temperature (>70°C) was recorded throughout this period. Since we expect that daily patterns of ozone levels are different during the weekends, we only select weekday data. Meteorological data from the nearby Bothasig station was used as this information was inconsistently recorded at the Plattekloof AQM station. The predictor processes under consideration for this analysis are hourly records of temperature, wind speed and relative humidity. There are $N = 793$ daily curve data of ozone measurements denoted by $Y_i(t_j)$ where $i \in [1, \ldots, 793]$ represents a day, and $j \in [0, \ldots, 23]$, the time of the day, with the corresponding meteorological data, $X_i(v_r)$ for $r \in [0, \ldots, 23]$. The ozone and meteorological data are observed on the same time grid, $T = R$.

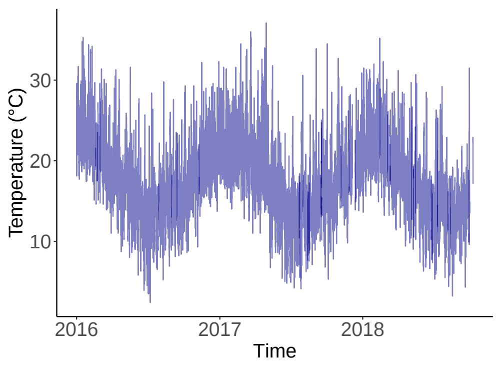
The time-series plots of the ozone data and meteorological processes can be found in Figure (5.1). We found that there exists a slight seasonality component in the time-series ozone data. Since the meteorological processes also exhibit seasonal patterns which could potentially explain the variation in the ozone data, we do not remove seasonality or trend in the ozone data.

5.1.2 Data pre-processing

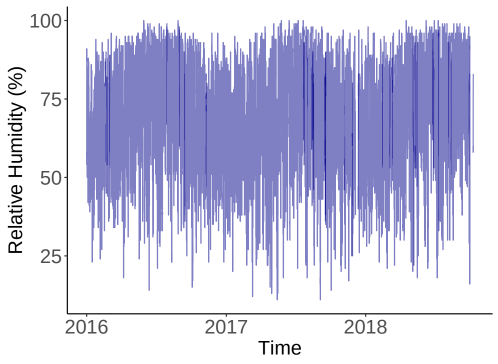
For each curve data, we pre-process the ozone measurements using natural cubic spline interpolation to estimate missing observations with 24 basis functions for functional data with less than 3 missing observations within each curve. The missing observations within the curve occurred during consecutive times. In total, there were 86 curves with more than 2 missing observations within the curve that were discarded, out of which 42 days had no ozone records. They also mostly occurred in large blocks of missing observations. The first 80% of the observations were retained as the training set and the remaining for the test set. The test set consists of data from 05/03/2018 to 09/10/2018. The data were split in a sequential way to be able to generate forecasts into the future. The time indices for both the functional response and predictors were standardised to ensure that $\tau \in [0, 1]$ where $\tau_j = \frac{t_j - \min_j(t_j)}{\max_j(t_j) - \min_j(t_j)}$ for $t_j \in \{0, 1, 2, \ldots, 23\}$.



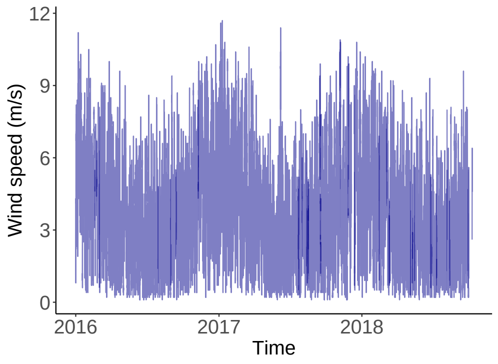
(a) *Ozone*



(b) *Temperature*



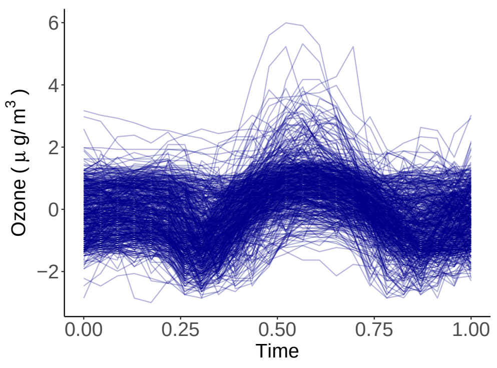
(c) *Relative humidity*



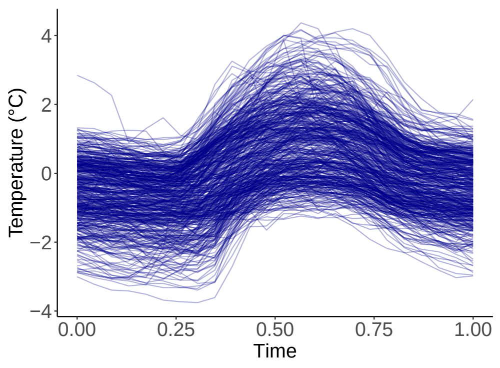
(d) *Wind speed*

Figure 5.1. *Time-series plots of the ozone and meteorological processes from 01/01/2016 to 09/10/2018 measured hourly. Similar cyclic patterns can be observed in the meteorological processes, and a weaker seasonality component can be noted in the yearly ozone processes.*

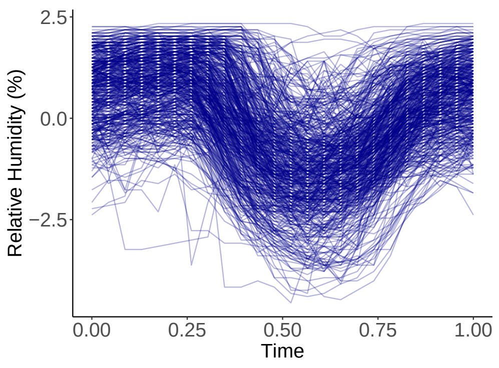
To ensure Gaussianity of the ozone data, we performed a Box-Cox transformation using the training set, $\tilde{Y}_i(\tau) = \dfrac{Y_i(\tau)^\lambda - 1}{\lambda}$ for $i = 1, \ldots, N$ and $\tau \in \mathcal{T}$. Guerrero's method was used, where the parameter λ which minimises the coefficient of variation for sub-series of the data is selected. The `R` package `forecast` was used to perform the Box-Cox transformation which yielded an optimal value $\hat{\lambda} = 0.810$ (Hyndman and Khandakar, 2008). The response and predictor processes in the training set were standardised by their overall mean and standard deviation. Figure (5.2) illustrates the daily curves of the functional response and predictors used in the analysis.



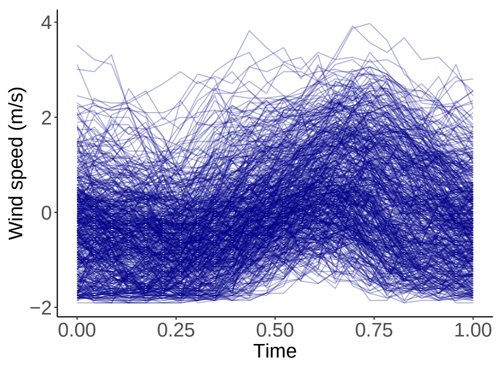
(a) *Ozone*



(b) *Temperature*



(c) *Relative humidity*



(d) *Wind speed*

Figure 5.2. *Post-processed daily curves of the Box-Cox transformed ozone data and meteorological data, both standardised by the overall mean and standard deviation.*

5.2 Implementation

Prior to fitting the HFLMs, preliminary diagnostic checks were carried out to flag possible non-identifiability in the model set-up. The functional intercept term is modelled using 10 thin-plate splines. We performed diagnostics for $U = K = 10$ marginal cubic B-splines using pre-smoothed estimates of the functional predictor processes with $K_g = 10$. Since rank deficiency and non-identifiability was not flagged, we chose to select $U = K = 10$ cubic B-splines basis functions. We ran 10000 iterations of the MCMC sampler with a 50% burn-in. For the HFLM, we randomly set starting parameters from $\mathcal{U}(0, 1)$. For the measurement error models, we generate starting parameters for the regression coefficients from a Gaussian distribution with mean equal to the posterior means and twice the posterior standard deviations of the MCMC chains of the HFLM. Convergence diagnostics can be found in Appendix G. We fit two two-variable HFLM models, one with predictors temperature and relative humidity, and another with temperature and wind speed. Since the focus of this study is not on model selection but to rather highlight the potential of HFLMs for air pollution forecasting, we did not consider other predictors.

The FARX(1) model was implemented using the `far` package (Guillas and Damon, 2015). For the FARX(1) model, it is necessary to select the number of eigenfunctions used to estimate the covariance operators. We perform cross-validation using the `far.cv` function and the optimal dimension is selected based on the minimisation of the cross-validation L^2 error. Out-of-sample one-step ahead predictions are generated based on the estimated covariance operator and the previous day's observed trajectory. A drawback of this implementation is that Damon and Guillas (2002) have not established how prediction intervals must be evaluated in their work. Two two-variable FARX(1) models are also fitted with predictors temperature and relative humidity, and another with temperature and wind speed.

The package `ftsa` was used to fit the Hyndman-Ullah FAR(1) model (Shang, 2013). The predictions for the Hyndman-Ullah FAR(1) are based on the forecasted FPC scores using exponential smoothing, and the functional trajectories are re-constructed using the functional principal components bases. The authors propose a parametric approach based on the Gaussian assumptions of the errors.

To evaluate the importance of the complex Bayesian HFLM, we implement a naive persistence model in which the next functional observation is equal to the previous one (Guillas and Damon, 2015).

To compare the performance of our models, we evaluate the out-of-sample IMSE and IMAE for discretely observed data. We consider a $h = \{10, 30, 110\}$ day forecast horizon. Since the current FARX(1) forecast function from the `far` package only provides one-step ahead forecasts from observed past trajectories of the functional response, we iteratively generate forecasts based on the previous' day out-of-sample forecast for consistent model comparison. For the HFLMs and FARX(1) models, the observed functional predictors at $t + 1$ are used to generate the forecast for the response at $t + 1$.

5.3 Results

5.3.1 Model comparisons

The performance of the Bayesian HFLMs which we refer to as the HFLM, two-stage-HFLM and joint-HFLM, and the functional time-series models, FARX(1) and Hyndman-Ullah FAR using the functional predictors *temperature and relative humidity* are presented in Table (5.1). We find that the Bayesian HFLM without smoothing and using the two-stage analysis outperform the Hyndman-Ullah FAR(1) and are competitive against the FARX(1). For long-time horizons, $h = 110$, the two-stage HFLM outperforms the FARX(1) model. In general, the improvement in performances of the HFLM and two-stage HFLM over existing methods becomes more substantial as the time horizon of the forecasts increases.

However, the joint HFLM does very poorly despite reporting an in-sample IMSE of 0.379 compared to 0.685 for both the HFLM and two-stage HFLM. Figures in (5.3) illustrate the fitted surfaces corresponding to temperature and relative humidity respectively. We found that the estimated surfaces of the joint model differ substantially from the other models. This indicates that the joint model tends to find solutions that minimise in-sample error effectively which however performs poorly out-of-sample. We suspect that in the joint model, there is a tendency to only accept FPC scores of the functional covariates that mostly satisfy the regression model which may lead to overfitting. The average acceptance rate is 13.5% across all observations and lie within 0 to 16% implying that at least one observation has not been updated at all in the MCMC sampler.

Model	**IMSE**			**IMAE**		
	Forecasts (h day time horizon)			Forecasts (h day time horizon)		
	10	30	110	10	30	110
Naive[1]	0.673	1.025	1.242	0.624	0.776	0.816
Hyndman-Ullah FAR(1)	0.701	1.159	3.045	0.678	0.850	1.460
FARX(1)	0.349	0.529	1.088	0.421	0.548	0.860
HFLM	**0.441**	**0.547**	**1.036**	**0.499**	**0.571**	**0.842**
Two-stage HFLM	**0.443**	**0.549**	**1.033**	**0.450**	**0.573**	**0.842**
Joint HFLM	1.232	1.011	1.451	0.850	0.777	0.961

Table 5.1. *Comparison of the models fitted based on the IMSE and IMAE of the forecasts. The FARX(1) and Bayesian HFLMs are fitted using the predictors temperature and relative humidity.*

Table (5.2) and Figures in (5.4) correspond to the performance and estimated surfaces of models fitted with predictor processes temperature and wind speed. Similar findings are made for this model whereby over a long time horizon, the Bayesian HFLMs

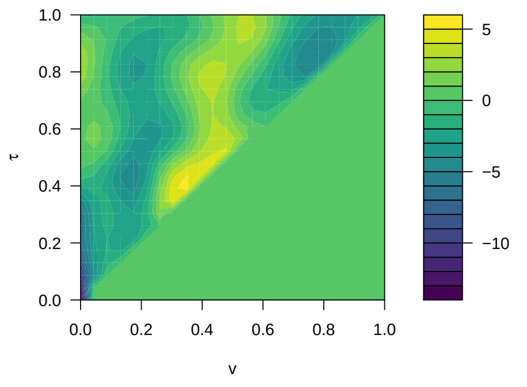
[1]It is important to note the naive model (Guillas and Damon, 2015) uses observations in the test set to predict the next observation, hence care must be taken when comparing the other models to the naive model since the forecasts are generated from previously estimated forecasts and not on actual observed past trajectories for the FAR, FARX and HFLMs.

perform the best with the exception of the joint estimation model. With lower forecasts IMSE and IMAE, there is an indication that temperature and wind speeds are better predictors of ozone than temperature and relative humidity. We further investigate the estimates of the temperature and wind speed processes of the measurement error models in Figures (5.5) and (5.6) by randomly sampling 5 profiles from the training set. For the wind speed profiles where the observations are noisier than the temperature profiles, we find that the two-stage HFLM (95% CI for $\hat{\sigma}^2_{\eta_2}$: $[0.0356, 0.0381]$) provides a more accurate smooth representation of the observed trajectories than the joint HFLM (95% CI for $\hat{\sigma}^2_{\eta_2}$: $[0.0547, 0.0587]$).

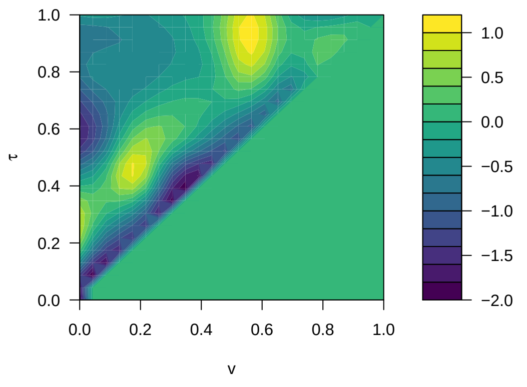
Model	**IMSE**			**IMAE**		
	Forecasts (h day time horizon)			Forecasts (h day time horizon)		
	10	30	110	10	30	110
Naive	0.673	1.025	1.242	0.624	0.776	0.816
Hyndman-Ullah FAR(1)	0.701	1.159	3.045	0.678	0.850	1.460
FARX(1)	0.308	0.493	1.099	0.393	0.524	0.858
HFLM	**0.391**	**0.504**	**0.984**	**0.466**	**0.537**	**0.806**
Two-stage HFLM	**0.391**	**0.509**	**0.982**	**0.463**	**0.539**	**0.807**
Joint HFLM	1.242	0.999	1.161	0.806	0.737	0.839

Table 5.2. *Comparison of the models fitted based on the IMSE and IMAE of the forecasts. The FARX(1) and Bayesian HFLMs are fitted using the predictors temperature and wind speed.*

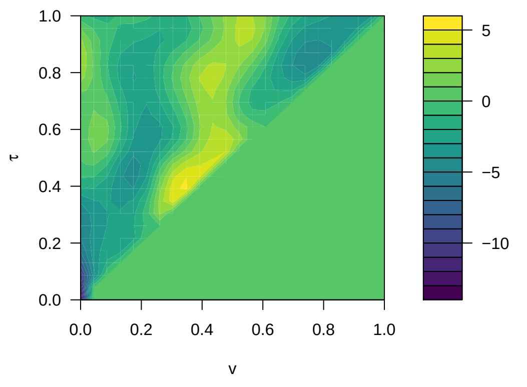
Figure 5.3. *Estimated functional regression surfaces obtained from (a)-(b) the Bayesian HFLM without modelling the functional covariates, (c)-(d) the Bayesian two-stage HFLM and (e)-(f) the joint HFLM fitted with predictors* ***temperature*** *and* ***relative humidity****.*



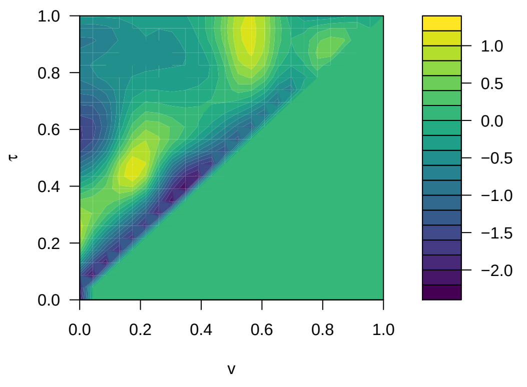
(a) *Temperature*



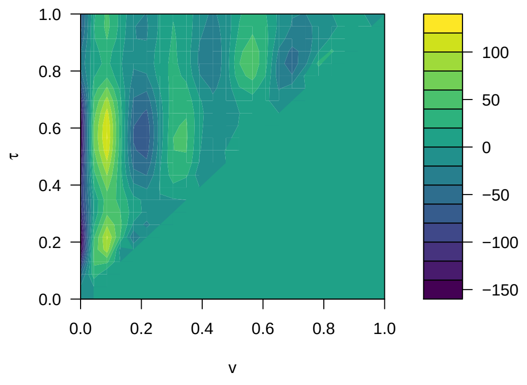
(b) *Relative humidity*



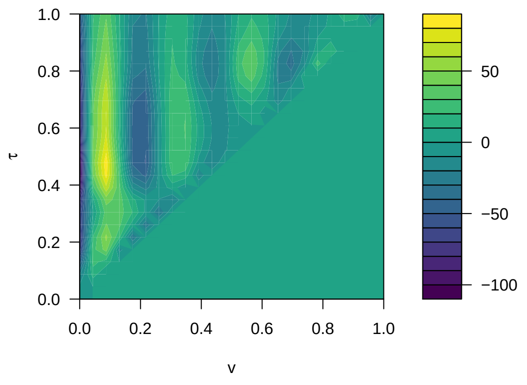
(c) *Temperature*



(d) *Relative humidity*

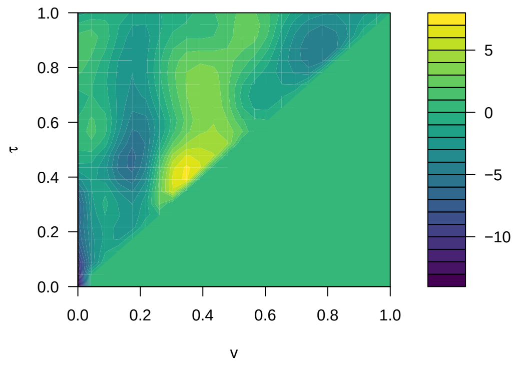


(e) *Temperature*

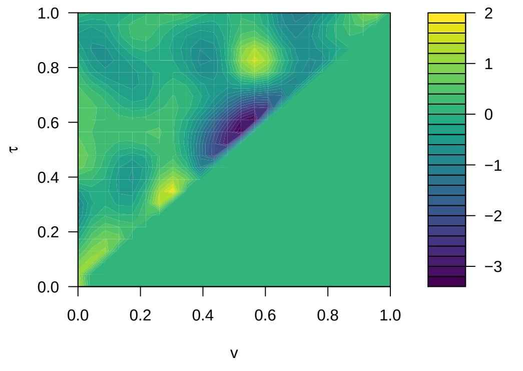


(f) *Relative humidity*

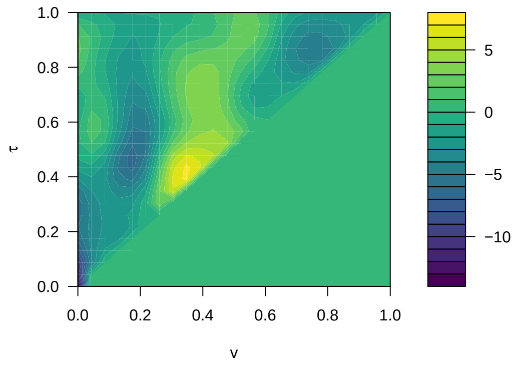
Figure 5.4. *Estimated functional regression surfaces obtained from (a)-(b) the Bayesian HFLM without modelling the functional covariates, (c)-(d) the Bayesian two-stage HFLM and (e)-(f) the joint HFLM fitted with predictors* ***temperature*** *and* ***wind speed****.*



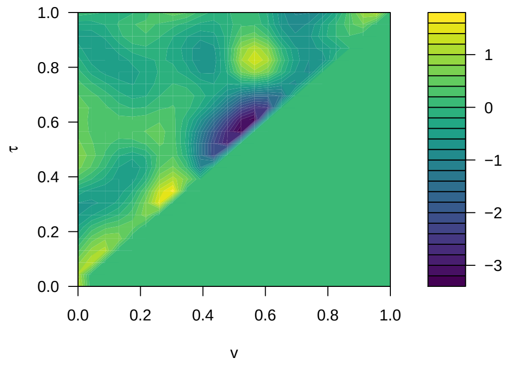
(a) *Temperature*



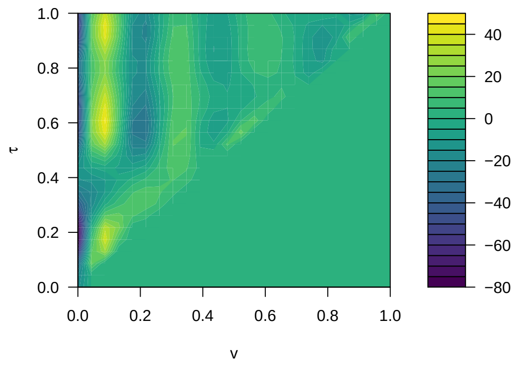
(b) *Wind speed*



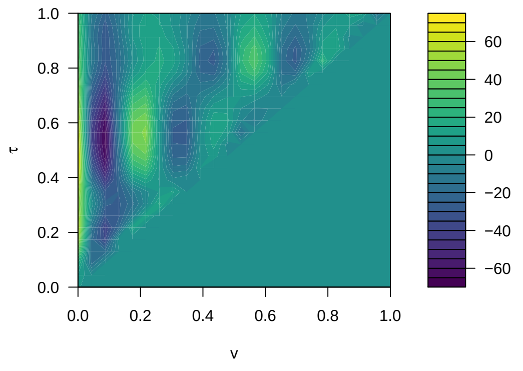
(c) *Temperature*



(d) *Wind speed*



(e) *Temperature*



(f) *Wind speed*

Figure 5.5. *Estimated temperature profiles from the models fitted. The shaded regions represent the 95% credibility interval. The black line corresponds to the observed trajectory.*

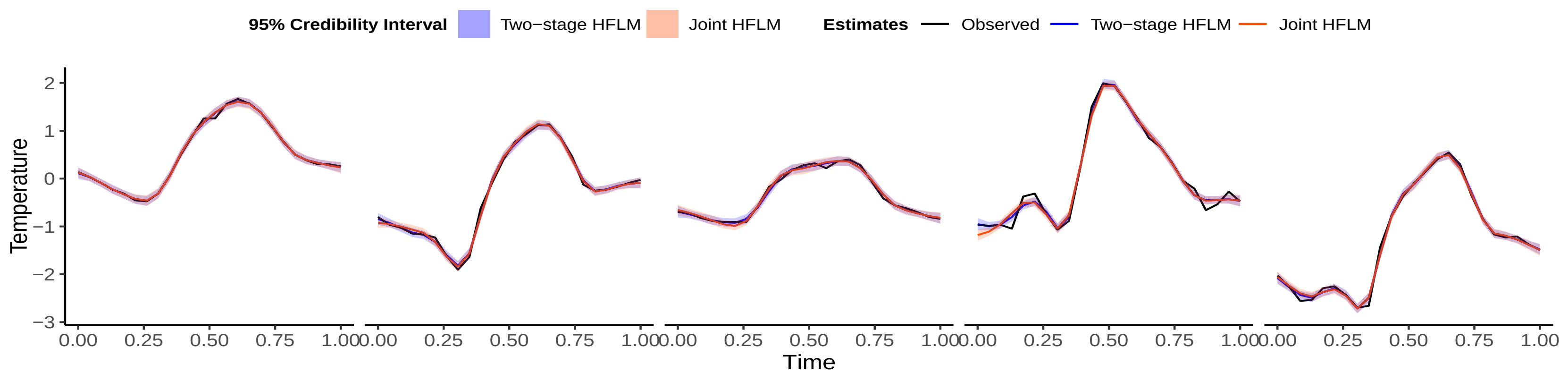
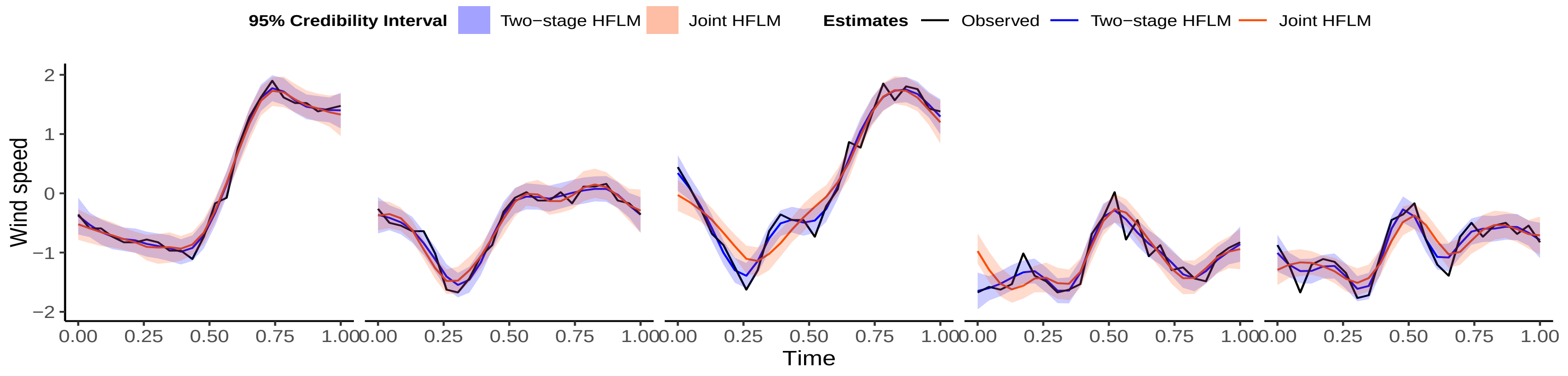


Figure 5.6. *Estimated wind speed profiles from the models fitted. The shaded regions represent the 95% credibility interval. The black line corresponds to the observed trajectory.*



5.3.2 Inference

Empirical coverage from the 95% Bayesian prediction interval calculated by finding the average coverage over the time of the day and over the days is given in Table (5.3). We found that the Bayesian HFLMs outperform the Hyndman-Ullah FAR(1) model with the exception of the joint estimation model. The empirical coverage is high for $h = \{10, 30\}$ day ahead time horizon (>95%), which however decreases for $h = 110$.

	Model	Hyndman-Ullah FAR(1)	HFLM	Two-stage HFLM	Joint HFLM	HFLM	Two-stage HFLM	Joint HFLM
			Temperature, Relative Humidity			*Temperature, Wind Speed*		
Empirical coverage of forecasts	10	0.825	0.971	0.975	0.808	0.975	0.979	0.821
	30	0.729	0.951	0.951	0.842	0.957	0.956	0.849
	110	0.540	0.873	0.877	0.742	0.882	0.882	0.784

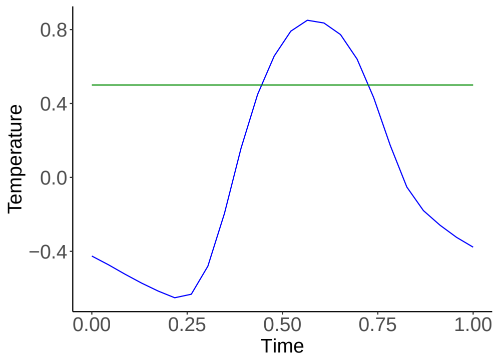
Table 5.3. *Empirical coverage of the forecasts calculated by finding the average over the time of the day and over the days.*

The performances of the HFLM and two-stage HFLM are very similar. This can be attributed to the fact that the number of functional data is high and that the measurement error in the functional predictors is low. We select the two-stage HFLM with predictors temperature and wind speed which has a slightly higher empirical coverage of short-term forecasts for further analysis.

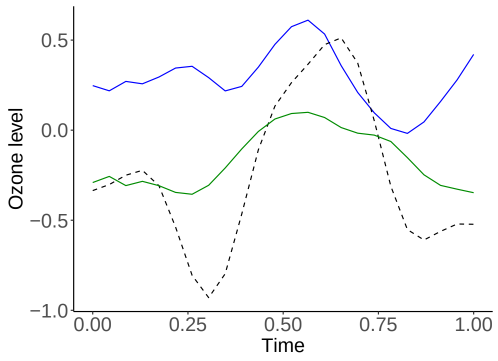
Figure (5.8) illustrates the forecasts from the two-stage HFLM, the FARX(1) with predictors temperature and wind speed, and Hyndman-Ullah FAR(1) for a 15-day horizon. The shaded region corresponds to the 95% prediction interval for the two-stage HFLM and the Hyndman-Ullah FAR(1). As shown by the empirical coverage, the prediction interval of the HFLM covers most of the true observations, but is wide. We further note that both the FARX(1) and the two-stage HFLM are not able to capture the outlying peaks during day 13.

From the estimated regression surfaces displayed in Figure (5.4), we note that temperature has a strong positive peaked effect on ozone levels from around 10am to 12pm. We note that temperature at midday has an important positive relationship with ozone levels throughout the rest of the day. Wind speed also has a negative peaked effect from around 12pm to 14pm. The surface tends to zero as $|v - \tau|$ increases which suggests that current wind speed does not influence ozone levels further ahead in time. For interest's sake, we seek to understand the estimated regression surfaces from the joint estimation model despite the very large absolute estimated coefficients. The joint model suggests that the effects of both temperature and wind speed on ozone levels tend to be more time-specific. The large absolute values along $v \approx 0.1$ suggest that temperature and wind speed levels early in the morning tend to impact ozone levels throughout the rest of the day.

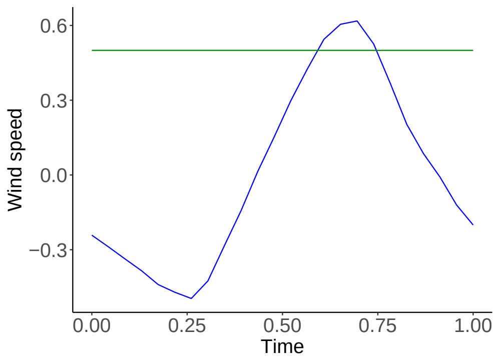
We analyse the marginal effect of the bivariate regression coefficients using the two-stage HFLM with predictors temperature and wind speed in Figure (5.7). The marginal additive effect of the G^{th} predictor on ozone levels at time τ is evaluated as $m_G(\tau) = \int_0^\tau \hat{\beta}_G(s,\tau)\tilde{X}_G(s)ds$ where for $g \neq G$, $\tilde{X}_g(s) = 0$ for $s \in \mathcal{S}$. We consider two different profiles for both temperature and wind speed, 1) an average profile of the smoothed functional predictors, and 2) a constant profile where $\tilde{X}_g(s) = 0.5$ for $s \in \mathcal{S}$. We find that the absolute marginal effect of temperature is greater than zero for $\tau \in \mathcal{T}$ suggesting that temperature has an effect on ozone levels throughout the day. Lower temperature levels in the morning tend to result in slightly higher ozone levels during that morning. When the peak of temperature is high during midday, the ozone level also increases. This is consistent with what we found for the regression surfaces. However, wind speed does not have a strong effect on ozone with $m_2(\tau) \approx 0$ throughout the day.



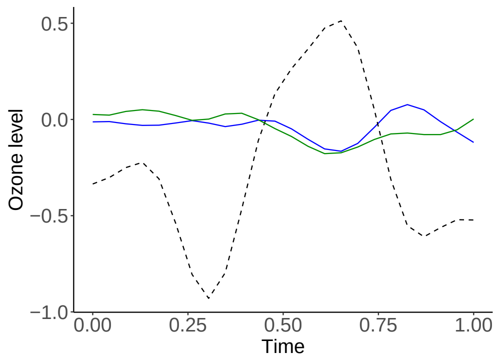
(a) *The average temperature profile in blue, and constant temperature profile in green.*



(b) *Marginal effect on ozone levels for an average temperature profile (blue) and constant temperature profile (green) on the ozone levels.*



(c) *The average wind speed profile in blue, and constant wind speed profile in green.*



(d) *Marginal effect on ozone levels for an average wind speed profile (blue) and constant wind speed profile (green) on the ozone levels.*

Figure 5.7. *Marginal effect of the bivariate regression coefficients for temperature and wind speed processes on the ozone levels. The black dashed line corresponds to the estimated mean profile of the ozone levels, $\hat{\mu}(\tau)$.*

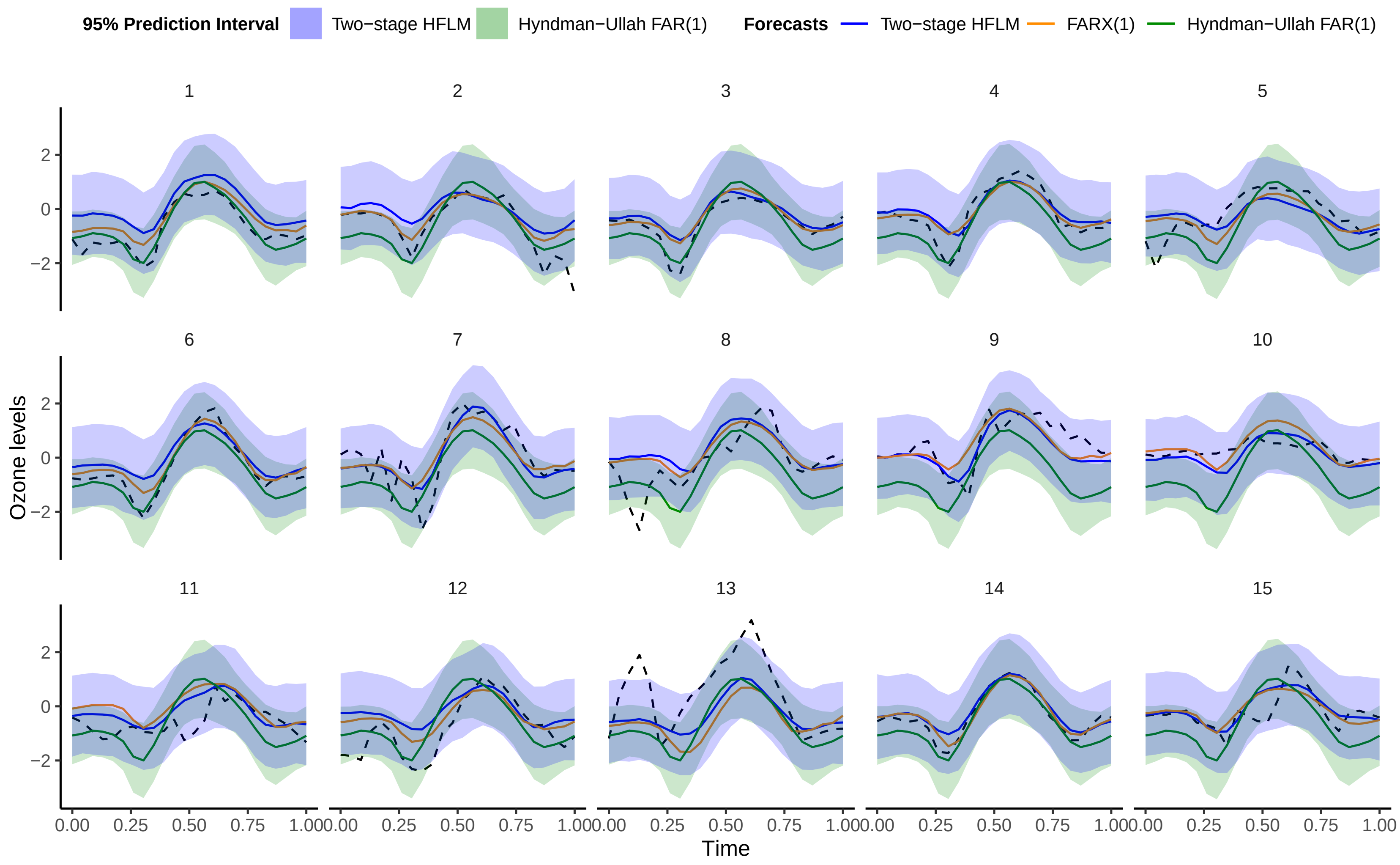
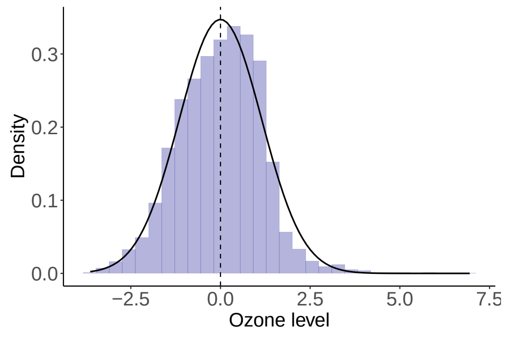


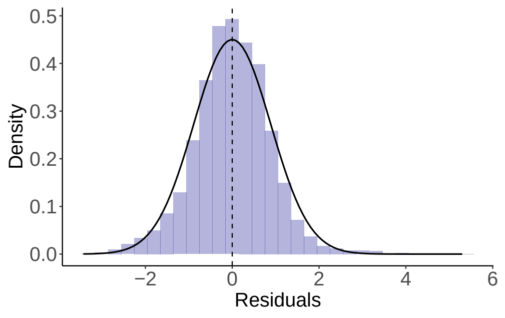
Figure 5.8. *15-day ahead forecasts generated by the models fitted. The shaded regions represent the 95% prediction interval. The black dashed line corresponds to the observed trajectory.*

5.3.3 Model checking

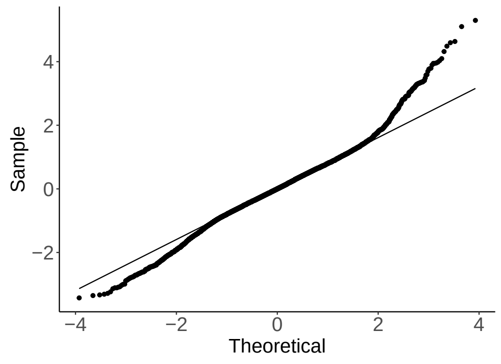
Residual diagnostic checks were performed to evaluate the two-stage HFLM with predictors temperature and wind speed as displayed in Figure (5.9).



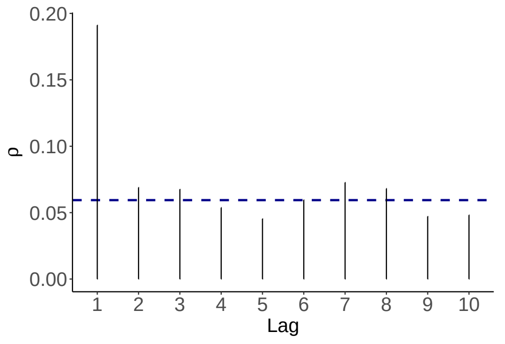
(a)



(b)



(c)



(d)

Figure 5.9. *Residual diagnostics where the residuals are calculated as $\hat{e}_i(\tau) = \hat{Y}_i(\tau) - Y_i(\tau)$. (a) Histogram of the ozone measurements. The density of a Gaussian distribution with parameters equal to the mean and standard deviation of the functional response is shown in black. (b) Histogram of the residual processes. The density of a Gaussian distribution with parameters equal to the mean and standard deviation of the residuals is shown in black. (c) Quantile-quantile plot of the residuals. (d) Correlogram of the residual processes.*

Figure (5.9a) is a histogram of the ozone measurements following Box-Cox transformation and standardisation. Despite the Box-Cox transformation to ensure Gaussianity of the observations, we note that the distribution is positively skewed. The residual processes are centered around 0 in Figure (5.9b), but we note a similar positive skewness in the distribution of the residuals which is confirmed by the quantile-quantile plot of the residuals where there is a strong deviation from the theoretical quantile on the right end. We further test the residual processes for autocorrelation in Figure (5.9d) using the `fdaACF` package (Mestre et al., 2021). We note the presence of a very strong lag 1 autocorrelation in the functional residual series.

5.4 Conclusion

In this chapter, we demonstrated the promise of HFLMs to predict ozone levels. The classical Bayesian HFLM and two-stage HFLM are very competitive against the FARX(1) model with similar IMSE and IMAE of predictions. An advantage of the HFLM is that it allows us to analyse the effects of the predictor processes on ozone levels at different time points while also ensuring that the predictor values ahead in the day do not impact the current outcome. With lower IMSE and IMAE, we concluded that temperature and wind speed are better predictors of ozone than temperature and relative humidity. Upon further investigation, we found that wind speed does not have an important effect on ozone levels while higher temperatures at midday lead to higher ozone levels throughout the rest of the day. In this work, we only investigated the effect of wind speed, relative humidity and temperature on ozone levels, but we expect that the inclusion of other meteorological and environmental information such as rainfall and solar/UV radiation in a HFLM with multiple predictors may increase the predictive performance.

CHAPTER 6

Discussion

In this work, we presented a Bayesian formulation of the historical functional linear model for multiple functional predictors with measurement error. A tensor product of spline basis functions was used to model the bivariate regression surface. Through our choice of prior information, which imposes smoothness constraints, we were able to reduce the sensitivity to the dimensions of the basis functions. We further included a smoothing procedure also estimated within the MCMC sampler to account for measurement error in the functional predictor processes. An FPC based expansion was used to model the predictors. This resulted in a simplified posterior covariance matrix obtained under orthornormal properties of the basis which speeds up computation. We considered two methods of estimation. In our first proposition, the predictors are modelled independently from the regression model within a Gibbs sampling algorithm, and in the second proposition, we sample the FPC scores jointly in an adaptive Metropolis-within-Gibbs sampling algorithm.

Through extensive simulations, we showed that in the presence of noise in the predictor process, it is important to include the smoothing strategy for accurate estimation of the historical surfaces and for higher coverage probabilities. This suggests that the measurement error models effectively capture the uncertainty associated with the predictor process while the HFLM understates the variability of the estimates.

We thereafter applied our methodology to an air pollution data set collected by the City of Cape Town to forecast ozone levels. We showed that our method is highly competitive against the functional time-series models which are commonly used to model functional time series. For long-term forecasts, the Bayesian two-stage HFLM outperforms the latter with better forecasts and higher empirical coverage of the prediction interval. We however found that the joint HFLM produced spurious estimates of the historical regression surface which requires further investigation as it is beyond the scope of this thesis.

We found that the two-stage Bayesian HFLM is easier to implement while being as effective as the joint HFLM, and outperforms the latter under certain conditions. The two-stage Bayesian analysis is less prone to numerical problems and overfitting. Hence, we strongly argue for the use of the two-stage method if sizeable error is suspected in the functional predictors which we discuss further below. Nevertheless, it would be interesting to analyse the sensitivity of the joint estimation model under alternative prior specifications with different proposal distributions.

A limitation of this study remains the long run-times of the models which increase rapidly when both the frequency at which the curve data is being sampled, and the number of observations recorded increase. Experiments show that the size of the bias decreases when N increases and the error in the predictor is low. Consequently, the effect of the measurement error may become trivial as the number of observations increase. It could thus be helpful to establish a practical bound as a function of the number of observations and estimated uncertainty in the functional process beyond which Bayesian smoothing becomes necessary when smoothness constraints are imposed on the historical surface.

In our application, we noted the presence of a dependence structure in the residual processes. Future work could extend this model to a model with a more complex structure for the functional errors such as a FAR(p) model on the residual processes. Moreover, the skewed distribution of the functional response and residuals indicate that other distributional assumptions of the model errors such as a Student's t-distribution which is more robust to long tails must be considered (Lange et al., 1989). We also recommend to analyse the sensitivity of the model to the pre-processing steps performed prior to model fitting including the spline interpolation for observations that are sparsely missing within each curve, and the Box-Cox transformation of the functional response. In this application, we did not need to perform curve registration since all the prominent features that are generally expected in this type of environmental data, are aligned (Ramsay and Silverman, 2005).

We also found that as the time-horizon of forecasts increases, the IMSE increases. There are several reasons that might explain this. First, other dynamic factors such as human physical movement over time that have not been included in the model may explain this variation. Secondly, that the dynamics between the ozone levels and the meteorological processes change over time. Both of these problems can be tackled by including more, not necessarily functional, predictors in the model. More complex models that allow the regression surfaces to vary with season can also be developed to model the change in dynamics between the ozone levels and the meteorological processes.

It is important to highlight that only a few historical predictor values are used to predict responses in the morning whereas for responses at night, almost the whole day's trajectory is used to predict the outcome. Since it is possible that meteorological information recorded early in the morning do not impact ozone levels much later in the day, a lag-based model such as that defined in eq. (2.1) is worth investigating. An advantage of this model is that the same amount of historical predictor values is used to predict every outcome, however, responses at $\tau < \delta$ where δ is the lag parameter cannot be predicted.

In our application, we assumed that the functional predictors in the forecast horizon were known, but in practice, these predictors may also have to be forecasted. Thus, the sensitivity of the Bayesian HFLMs to unknown and predicted functional covariates remains to be investigated.

The two-stage Bayesian HFLMs have generally outperformed the other models under consideration. This can be attributed to the fact that the two-stage HFLM effectively captures the uncertainty associated with the predictor processes through the smoothing strategy while also being easy to implement. While in this work we assumed a fixed regular design for both functional response and covariates, it is relatively straightforward to extend to a design where either the response or covariates are irregularly sampled. Due to the spline-based approach, for the case where the functional response is not observed on a regular grid, the design matrix as defined in eq. (3.4) will require careful construction where the spline basis functions defined at only the observed τ will be included. For irregularly sampled predictor processes, FPC based methods that are robust to such structure will need to be investigated. Finally, this work lays the foundation for other methodological developments in this area such as Bayesian variable selection for this class of models and variational inference for more efficient run-times.

Bibliography

Aue, A., Norinho, D. D., and Hörmann, S. (2015). On the prediction of stationary functional time series. *Journal of the American Statistical Association*, 110(509):378–392.

Berry, S. M., Carroll, R. J., and Ruppert, D. (2002). Bayesian smoothing and regression splines for measurement error problems. *Journal of the American Statistical Association*, 97(457):160–169.

Besse, P. C., Cardot, H., and Stephenson, D. B. (2000). Autoregressive forecasting of some functional climatic variations. *Scandinavian Journal of Statistics*, 27(4):673–687.

Bosq, D. (2000). *Linear processes in function spaces: theory and applications*, volume 149. Springer Science and Business Media.

Brockhaus, S., Melcher, M., Leisch, F., and Greven, S. (2017). Boosting flexible functional regression models with a high number of functional historical effects. *Statistics and Computing*, 27(4):913–926.

Brooks, S. P. and Gelman, A. (1998). General methods for monitoring convergence of iterative simulations. *Journal of Computational and Graphical Statistics*, 7(4):434–455.

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

Chib, S. and Greenberg, E. (1995). Understanding the Metropolis-Hastings algorithm. *The American Statistician*, 49(4):327–335.

Crainiceanu, C. M. and Goldsmith, A. J. (2010). Bayesian functional data analysis using WinBUGS. *Journal of Statistical Software*, 32(11).

Crainiceanu, C. M., Staicu, A.-M., and Di, C.-Z. (2009). Generalized multilevel functional regression. *Journal of the American Statistical Association*, 104(488):1550–1561.

Damon, J. and Guillas, S. (2002). The inclusion of exogenous variables in functional autoregressive ozone forecasting. *Environmetrics*, 13(7):759–774.

Davis, J. and Speckman, P. (1999). A model for predicting maximum and 8 h average ozone in Houston. *Atmospheric Environment*, 33(16):2487–2500.

De Boor, C. (1972). On calculating with B-splines. *Journal of Approximation theory*, 6(1):50–62.

Duchon, J. (1977). Splines minimizing rotation-invariant semi-norms in Sobolev spaces. In *Constructive theory of functions of several variables*, pages 85–100. Springer.

Eilers, P. H. and Marx, B. D. (1996). Flexible smoothing with B-splines and penalties. *Statistical science*, 11(2):89–121.

Eubank, R. L. (1988). *Spline smoothing and nonparametric regression*, volume 90. M. Dekker New York.

Fahrmeir, L. and Lang, S. (2001). Bayesian inference for generalized additive mixed models based on Markov random field priors. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 50(2):201–220.

Ferraty, F. and Vieu, P. (2006). *Nonparametric functional data analysis: theory and practice*. Springer Science and Business Media.

Fox, C. W. and Roberts, S. J. (2012). A tutorial on variational Bayesian inference. *Artificial intelligence review*, 38(2):85–95.

Gelfand, A. E. and Smith, A. F. (1990). Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85(410):398–409.

Gelman, A. (2006). Prior distributions for variance parameters in hierarchical models (comment on article by Browne and Draper). *Bayesian Analysis*, 1(3):515–534.

Gelman, A., Rubin, D. B., et al. (1992). Inference from iterative simulation using multiple sequences. *Statistical Science*, 7(4):457–472.

Geman, S. and Geman, D. (1984). Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on pattern analysis and machine intelligence*, (6):721–741.

Gilbert, R. O. (1987). *Statistical methods for environmental pollution monitoring*. John Wiley and Sons.

Goldsmith, J., Bobb, J., Crainiceanu, C. M., Caffo, B., and Reich, D. (2011a). Penalized functional regression. *Journal of Computational and Graphical Statistics*, 20(4):830–851.

Goldsmith, J., Crainiceanu, C. M., Caffo, B., and Reich, D. (2012). Longitudinal penalized functional regression for cognitive outcomes on neuronal tract measurements. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 61(3):453–469.

Goldsmith, J., Scheipl, F., Huang, L., Wrobel, J., Di, C., Gellar, J., Harezlak, J., McLean, M. W., Swihart, B., Xiao, L., Crainiceanu, C., and Reiss, P. T. (2020). *refund: Regression with Functional Data*. R package version 0.1-23.

Goldsmith, J. and Schwartz, J. E. (2017). Variable selection in the functional linear concurrent model. *Statistics in medicine*, 36(14):2237–2250.

Goldsmith, J., Wand, M. P., and Crainiceanu, C. (2011b). Functional regression via variational Bayes. *Electronic Journal of Statistics*, 5:572.

Greven, S. and Scheipl, F. (2017). A general framework for functional regression modelling. *Statistical Modelling*, 17(1-2):1–35.

Guillas, S. and Damon, J. (2015). *far: Modelization for Functional AutoRegressive Processes*. R package version 0.6-5.

Haario, H., Saksman, E., and Tamminen, J. (2001). An adaptive Metropolis algorithm. *Bernoulli*, pages 223–242.

Harezlak, J., Coull, B. A., Laird, N. M., Magari, S. R., and Christiani, D. C. (2007). Penalized solutions to functional regression problems. *Computational Statistics and Data Analysis*, 51(10):4911–4925.

Hastie, T. and Tibshirani, R. (1993). Varying-coefficient models. *Journal of the Royal Statistical Society: Series B (Methodological)*, 55(4):757–779.

Hastings, W. K. (1970). Monte carlo sampling methods using Markov chains and their applications.

Hays, S., Shen, H., and Huang, J. Z. (2012). Functional dynamic factor models with application to yield curve forecasting. *The Annals of Applied Statistics*, pages 870–894.

Hyndman, R. J. and Khandakar, Y. (2008). Automatic time series forecasting: the forecast package for R. *Journal of Statistical Software*, 26(3):1–22.

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

Ibarra-Berastegi, G., Elias, A., Barona, A., Saenz, J., Ezcurra, A., and de Argandoña, J. D. (2008). From diagnosis to prognosis for forecasting air pollution using neural networks: Air pollution monitoring in Bilbao. *Environmental Modelling and Software*, 23(5):622–637.

Ivanescu, A. E., Staicu, A.-M., Scheipl, F., and Greven, S. (2015). Penalized function-on-function regression. *Computational Statistics*, 30(2):539–568.

Kim, K. (2010). *The recent history functional linear model and its extension to sparse longitudinal data*. PhD thesis, The Pennsylvania State University.

Kim, K., Şentürk, D., and Li, R. (2011). Recent history functional linear models for sparse longitudinal data. *Journal of Statistical Planning and Inference*, 141(4):1554–1566.

Kokoszka, P. and Reimherr, M. (2017). *Introduction to functional data analysis*. Chapman and Hall/CRC.

Kowal, D. R. (2021). Dynamic regression models for time-ordered functional data. *Bayesian Analysis*, 1(1):1–29.

Kowal, D. R., Matteson, D. S., and Ruppert, D. (2017). A bayesian multivariate functional dynamic linear model. *Journal of the American Statistical Association*, 112(518):733–744.

Kowal, D. R., Matteson, D. S., and Ruppert, D. (2019). Functional autoregression for sparsely sampled data. *Journal of Business and Economic Statistics*, 37(1):97–109.

Lange, K. L., Little, R. J., and Taylor, J. M. (1989). Robust statistical modeling using the t-distribution. *Journal of the American Statistical Association*, 84(408):881–896.

Malfait, N. and Ramsay, J. O. (2003). The historical functional linear model. *Canadian Journal of Statistics*, 31(2):115–128.

Marx, B. D. and Eilers, P. H. (1999). Generalized linear regression on sampled signals and curves: a p-spline approach. *Technometrics*, 41(1):1–13.

McLean, M. W., Scheipl, F., Hooker, G., Greven, S., and Ruppert, D. (2013). Bayesian functional generalized additive models with sparsely observed covariates. *arXiv preprint arXiv:1305.3585*.

Mestre, G., Portela, J., Rice, G., San Roque, A. M., and Alonso, E. (2021). Functional time series model identification and diagnosis by means of auto-and partial autocorrelation analysis. *Computational Statistics and Data Analysis*, 155:107108.

Metropolis, N. and Ulam, S. (1949). The Monte Carlo method. *Journal of the American statistical association*, 44(247):335–341.

Meyer, M. J., Coull, B. A., Versace, F., Cinciripini, P., and Morris, J. S. (2015). Bayesian function-on-function regression for multilevel functional data. *Biometrics*, 71(3):563–574.

Meyer, M. J., Malloy, E. J., and Coull, B. A. (2021). Bayesian wavelet-packet historical functional linear models. *Statistics and Computing*, 31(2):1–13.

Müller, H.-G. and Stadtmüller, U. (2005). Generalized functional linear models. *the Annals of Statistics*, 33(2):774–805.

Niska, H., Hiltunen, T., Karppinen, A., Ruuskanen, J., and Kolehmainen, M. (2004). Evolving the neural network model for forecasting air pollution time series. *Engineering Applications of Artificial Intelligence*, 17(2):159–167.

Pisoni, E., Farina, M., Carnevale, C., and Piroddi, L. (2009). Forecasting peak air pollution levels using NARX models. *Engineering Applications of Artificial Intelligence*, 22(4-5):593–602.

Plummer, M., Best, N., Cowles, K., and Vines, K. (2006). CODA: Convergence diagnosis and output analysis for mcmc. *R News*, 6(1):7–11.

Pomann, G.-M., Staicu, A.-M., Lobaton, E. J., Mejia, A. F., Dewey, B. E., Reich, D. S., Sweeney, E. M., Shinohara, R. T., et al. (2016). A lag functional linear model for prediction of magnetization transfer ratio in multiple sclerosis lesions. *The Annals of Applied Statistics*, 10(4):2325–2348.

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

Ramsay, J. O., Graves, S., and Hooker, G. (2020). *fda: Functional Data Analysis*. R package version 5.1.9.

Reiss, P. T. and Ogden, R. T. (2007). Functional principal component regression and functional partial least squares. *Journal of the American Statistical Association*, 102(479):984–996.

Rioul, O. and Vetterli, M. (1991). Wavelets and signal processing. *IEEE signal processing magazine*, 8(4):14–38.

Roberts, G. O. and Rosenthal, J. S. (2009). Examples of adaptive MCMC. *Journal of Computational and Graphical Statistics*, 18(2):349–367.

Rue, H. (2001). Fast sampling of Gaussian Markov random fields. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):325–338.

Ruppert, D. (2002). Selecting the number of knots for penalized splines. *Journal of Computational and Graphical Statistics*, 11(4):735–757.

SAAQIS, Department of Environment, Fisheries and Forestry, Republic of South Africa, and South African Weather Service (2021). South African Air Quality Information System (SAAQIS).

Salcedo, R., Ferraz, M. A., Alves, C., and Martins, F. (1999). Time-series analysis of air pollution data. *Atmospheric Environment*, 33(15):2361–2372.

Scheipl, F. and Greven, S. (2016). Identifiability in penalized function-on-function regression models. *Electronic Journal of Statistics*, 10(1):495–526.

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

Shang, H. L. (2013). ftsa: An R package for analyzing functional time series. *The R Journal*, 5(1):64–72.

Silverman, B. W. (1985). Some aspects of the spline smoothing approach to non-parametric regression curve fitting. *Journal of the Royal Statistical Society: Series B (Methodological)*, 47(1):1–21.

Sinharay, S. (2003). Assessing convergence of the Markov chain Monte Carlo algorithms: A review. *ETS Research Report Series*, 2003(1):i–52.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288.

Tierney, L. (1992). Exploring posterior distributions using Markov chains. Technical report, Department of Statistics, University of Minesota, Minneapolis.

Wand, M. and Ormerod, J. (2008). On semiparametric regression with O'Sullivan penalized splines. *Australian and New Zealand Journal of Statistics*, 50(2):179–198.

Wang, J.-L., Chiou, J.-M., and Müller, H.-G. (2016). Functional data analysis. *Annual*

Review of Statistics and Its Application, 3:257–295.

Western Cape Government (2019). Air quality management.

Wichmann, J. and Voyi, K. (2012). Ambient air pollution exposure and respiratory, cardiovascular and cerebrovascular mortality in Cape Town, South Africa: 2001–2006. *International journal of environmental research and public health*, 9(11):3978–4016.

Wood, S. N. (2003). Thin plate regression splines. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 65(1):95–114.

Wood, S. N. (2006a). *Generalized additive models: an introduction with R*. Chapman and Hall/CRC.

Wood, S. N. (2006b). Low-rank scale-invariant tensor product smooths for generalized additive mixed models. *Biometrics*, 62(4):1025–1036.

Wood, S. N. (2016). P-splines with derivative based penalties and tensor product smoothing of unevenly distributed data. *Statistics and Computing*, 27(4):985–989.

Xiao, L., Zipunnikov, V., Ruppert, D., and Crainiceanu, C. (2016). Fast covariance estimation for high-dimensional functional data. *Statistics and Computing*, 26(1):409–421.

Yao, F., Müller, H.-G., and Wang, J.-L. (2005). Functional linear regression analysis for longitudinal data. *The Annals of Statistics*, pages 2873–2903.

Zhao, Y., Ogden, R. T., and Reiss, P. T. (2012). Wavelet-based LASSO in functional linear regression. *Journal of Computational and Graphical Statistics*, 21(3):600–617.

APPENDIX A

Bayesian inference

A.1 Markov Chain Monte Carlo (MCMC) methods

Markov Chain Monte Carlo (MCMC) algorithms such as the Metropolis algorithms (Metropolis and Ulam, 1949; Hastings, 1970) and the Gibbs sampler (Geman and Geman, 1984; Gelfand and Smith, 1990) have become the standard estimation procedures for complex and possibly intractable posterior distributions arising from Bayesian inference (Gelman et al., 1992).

The goal of MCMC methods is to iteratively generate a sequence of values from a distribution whereby that sequence of values, also referred to as the MCMC samples, converges towards a stationary distribution which is the target distribution (Brooks and Gelman, 1998).

The Gibbs sampler (Geman and Geman, 1984) is the most common sampling algorithm when there is no closed-form solution to the joint posterior distribution but the conditional distributions of the parameters are tractable. In this procedure, samples are iteratively and sequentially drawn from the conditional posterior distributions (Gelfand and Smith, 1990). At each iteration, the parameters of the model are updated with their corresponding posterior samples.

The Metropolis-Hastings (M-H) algorithm (Hastings, 1970) is a generalisation of the Gibbs sampler mostly used when the conditional posterior distributions are intractable. Given a target distribution, $\pi(.)$, the objective is to iteratively sample from a proposal distribution , $x^* \sim q(.)$, and $u \sim \mathcal{U}(0,1)$, such that the new sample is only accepted if $u \leq r$ where the acceptance probability r is given by (Chib and Greenberg, 1995),

$$r = \min\left(\frac{\pi(x^*)q(x^*,x)}{\pi(x)q(x,x^*)}, 1\right),$$

where x is the current value of the parameter. When $q(.)$ is a symmetric distribution such that $q(x^*,x) = q(x,x^*)$, the acceptance probability reduces to,

$$r = \min\left(\frac{\pi(x^*)}{\pi(x)}, 1\right).$$

It can be shown that the MCMC samples converge towards the target distribution (Chib and Greenberg, 1995). The Gibbs sampler is a special case of the M-H algorithm in which the proposal density is the conditional posterior distribution which results in an acceptance probability equal to 1 (Chib and Greenberg, 1995).

It is possible to use a hybrid of MCMC algorithms to construct different sampling strategies. Consider two parameters θ_1 and θ_2 where it is possible to directly sample from $\theta_1|\theta_2$, but not from $\theta_2|\theta_1$. A Metropolis-within-Gibbs sampling scheme can be implemented whereby a Gibbs step is used to sample from $\theta_1|\theta_2$ combined with a Metropolis or M-H step for $\theta_2|\theta_1$ (Tierney, 1992).

A.1.1 A note on MCMC

MCMC estimation procedures can be particularly computationally expensive, and therefore, variational Bayes (VB) can be attractive alternative methods due to their higher computational efficiency. VB is a variational method which aims to find a distribution $Q(.)$ to approximate closely the target distribution where the closeness between the distributions can be assessed using a measure such as the Kullback-Leibler divergence (Fox and Roberts, 2012).

A.2 Assessing convergence

In order to assess convergence of the MCMC chains towards the target distribution, Gelman et al. (1992) proposed the Gelman-Rubin test also known as the potential scale reduction factor (PSRF). For this test, the authors suggest running m independent and parallel chains with different starting values for each chain. The PSRF is a comparison of within-chain and between-chain variances where a value close to 1 implies that the inferences obtained from the MCMC samples of the independent and parallel m chains have converged to the target distribution (Gelman et al., 1992). A cut-off value of 1.2 has been suggested in the literature (Sinharay, 2003).

APPENDIX B

Derivation of the conditional posterior distributions of the parameters of the regression model

For design matrix $\widetilde{\mathbf{R}}$ and vector of coefficients $\widetilde{\boldsymbol{\alpha}}$ defined in eq. (3.6) and under the prior specifications for regression coefficients $\widetilde{\boldsymbol{\alpha}}$, and variance parameters $\sigma_e^2, \sigma_\mu^2, \sigma_{b,g}^2$ defined in §3.2, we derive their corresponding conditional posterior distributions.

The probability distribution functions (PDF) are denoted by $[x]$, and the conditional distributions, $[x|y]$; $[x \mid \cdot\,]$ denotes the PDF of x conditional on the rest of the parameters.

B.1 Regression coefficients

$$
\begin{aligned}
[\widetilde{\boldsymbol{\alpha}} \mid \cdot\,] &= \left[\operatorname{vec}(\mathbf{Y}) \,\middle|\, \widetilde{\boldsymbol{\alpha}}, \sigma_e^2\right] [\widetilde{\boldsymbol{\alpha}}] \\
&\propto \exp\left(-\frac{1}{2\sigma_e^2}\left(\operatorname{vec}(\mathbf{Y}) - \widetilde{\mathbf{R}}\widetilde{\boldsymbol{\alpha}}\right)^{\mathrm{T}}\left(\operatorname{vec}(\mathbf{Y}) - \widetilde{\mathbf{R}}\widetilde{\boldsymbol{\alpha}}\right)\right) \times \exp\left(-\frac{1}{2}\widetilde{\boldsymbol{\alpha}}^{\mathrm{T}}\mathbf{D}_{\boldsymbol{\alpha}}\widetilde{\boldsymbol{\alpha}}\right) \\
&\propto \exp\left(-\frac{1}{2}\left(\frac{1}{\sigma_e^2}\widetilde{\boldsymbol{\alpha}}^{\mathrm{T}}\widetilde{\mathbf{R}}^{\mathrm{T}}\widetilde{\mathbf{R}}\widetilde{\boldsymbol{\alpha}} - \frac{2}{\sigma_e^2}\widetilde{\boldsymbol{\alpha}}^{\mathrm{T}}\widetilde{\mathbf{R}}^{\mathrm{T}}\operatorname{vec}(\mathbf{Y}) + \widetilde{\boldsymbol{\alpha}}^{\mathrm{T}}\mathbf{D}_{\boldsymbol{\alpha}}\widetilde{\boldsymbol{\alpha}}\right) + \dots\right)
\end{aligned}
$$

where $\dots$ represents terms in the exponential that do not depend on $\widetilde{\boldsymbol{\alpha}}$.

Now, looking at the exponential term,

$$
\begin{aligned}
&\frac{1}{\sigma_e^2}\widetilde{\boldsymbol{\alpha}}^{\mathrm{T}}\widetilde{\mathbf{R}}^{\mathrm{T}}\widetilde{\mathbf{R}}\widetilde{\boldsymbol{\alpha}} - \frac{2}{\sigma_e^2}\widetilde{\boldsymbol{\alpha}}^{\mathrm{T}}\widetilde{\mathbf{R}}^{\mathrm{T}}\operatorname{vec}(\mathbf{Y}) + \widetilde{\boldsymbol{\alpha}}^{\mathrm{T}}\mathbf{D}_{\boldsymbol{\alpha}}\widetilde{\boldsymbol{\alpha}} + \dots \\
&= \widetilde{\boldsymbol{\alpha}}^{\mathrm{T}}\left[\frac{1}{\sigma_e^2}\widetilde{\boldsymbol{R}}^{\mathrm{T}}\widetilde{\boldsymbol{R}} + \boldsymbol{D}_{\widetilde{\boldsymbol{\alpha}}}\right]\widetilde{\boldsymbol{\alpha}} - \frac{2}{\sigma_e^2}\widetilde{\boldsymbol{\alpha}}^{\mathrm{T}}\widetilde{\mathbf{R}}^{\mathrm{T}}\operatorname{vec}(\mathbf{Y}) + \dots
\end{aligned}
$$

By matrix factorisation of $\widetilde{\boldsymbol{\alpha}}$,

$$
= \left(\widetilde{\boldsymbol{\alpha}} - \boldsymbol{\mu}_{\widetilde{\boldsymbol{\alpha}}}\right)^{\mathrm{T}} \boldsymbol{\Sigma}_{\widetilde{\boldsymbol{\alpha}}}^{-1} \left(\widetilde{\boldsymbol{\alpha}} - \boldsymbol{\mu}_{\widetilde{\boldsymbol{\alpha}}}\right) + \dots,
$$

where $\boldsymbol{\Sigma}_{\widetilde{\boldsymbol{\alpha}}} = \left[\frac{1}{\sigma_e^2}\widetilde{\boldsymbol{R}}^{\mathrm{T}}\widetilde{\boldsymbol{R}} + \boldsymbol{D}_{\widetilde{\boldsymbol{\alpha}}}\right]^{-1}$ and $\boldsymbol{\mu}_{\widetilde{\boldsymbol{\alpha}}} = \frac{1}{\sigma_e^2}\boldsymbol{\Sigma}_{\widetilde{\boldsymbol{\alpha}}}\widetilde{\boldsymbol{R}}^{\mathrm{T}}\operatorname{vec}(\boldsymbol{Y})$.

Therefore,

$$
[\widetilde{\boldsymbol{\alpha}} \mid \cdot\,] \propto \underbrace{\exp\left(-\frac{1}{2}\left(\widetilde{\boldsymbol{\alpha}} - \boldsymbol{\mu}_{\widetilde{\boldsymbol{\alpha}}}\right)^{\mathrm{T}} \boldsymbol{\Sigma}_{\widetilde{\boldsymbol{\alpha}}}^{-1} \left(\widetilde{\boldsymbol{\alpha}} - \boldsymbol{\mu}_{\widetilde{\boldsymbol{\alpha}}}\right)\right)}_{\text{Kernel of a multivariate normal distribution}}.
$$

Hence, $\widetilde{\boldsymbol{\alpha}}|\cdot \sim \mathcal{N}_p\left(\boldsymbol{\mu}_{\widetilde{\boldsymbol{\alpha}}},\ \boldsymbol{\Sigma}_{\widetilde{\boldsymbol{\alpha}}}\right)$.

B.2 Variance parameters

$$
\begin{aligned}
\left[\sigma_e^2 \,\middle|\, \cdot\,\right] &= \left[\operatorname{vec}(\mathbf{Y}) \,\middle|\, \widetilde{\boldsymbol{\alpha}}, \sigma_e^2\right]\left[\sigma_e^2\right] \\
&\propto (\sigma_e^2)^{-NT/2} \exp\left(-\frac{1}{2\sigma_e^2}\left(\operatorname{vec}(\mathbf{Y}) - \widetilde{\mathbf{R}}\widetilde{\boldsymbol{\alpha}}\right)^{\mathrm{T}}\left(\operatorname{vec}(\mathbf{Y}) - \widetilde{\mathbf{R}}\widetilde{\boldsymbol{\alpha}}\right)\right) \times (\sigma_e^2)^{-0.001-1} \exp\left(-\frac{0.001}{\sigma_e^2}\right) \\
&\propto (\sigma_e^2)^{-NT/2-0.001-1} \times \exp\left(-\frac{1}{\sigma_e^2}\left(0.5\left(\operatorname{vec}(\mathbf{Y}) - \widetilde{\mathbf{R}}\widetilde{\boldsymbol{\alpha}}\right)^{\mathrm{T}}\left(\operatorname{vec}(\mathbf{Y}) - \widetilde{\mathbf{R}}\widetilde{\boldsymbol{\alpha}}\right) + 0.001\right)\right),
\end{aligned}
$$

which is the form of an inverse-gamma distribution with shape and scale parameters $0.5NT + 0.001$ and $0.5\left(\operatorname{vec}(\mathbf{Y}) - \widetilde{\mathbf{R}}\widetilde{\boldsymbol{\alpha}}\right)^{\mathrm{T}}\left(\operatorname{vec}(\mathbf{Y}) - \widetilde{\mathbf{R}}\widetilde{\boldsymbol{\alpha}}\right) + 0.001$, respectively.

Now, given the prior specification for $\widetilde{\boldsymbol{\mu}} \mid \sigma_\mu^2 \sim \mathcal{N}_{K_\mu}\left(\mathbf{0}, \sigma_\mu^2 \mathbf{I}_{K_\mu}\right)$ from partitioning the multivariate normal distribution, $\widetilde{\boldsymbol{\alpha}} \mid \sigma_\mu^2, \sigma_{b,g}^2 \sim \mathcal{N}_p\left(\mathbf{0}, \boldsymbol{D}_{\widetilde{\boldsymbol{\alpha}}}^{-1}\right)$,

$$
\begin{aligned}
\left[\sigma_\mu^2 \,\middle|\, \cdot\,\right] &= \left[\widetilde{\boldsymbol{\mu}} \,\middle|\, \sigma_\mu^2\right]\left[\sigma_\mu^2\right] \\
&\propto (\sigma_\mu^2)^{-K_\mu/2} \exp\left(-\frac{1}{2\sigma_\mu^2}\widetilde{\boldsymbol{\mu}}^{\mathrm{T}}\widetilde{\boldsymbol{\mu}}\right) \times (\sigma_\mu^2)^{-0.001-1} \exp\left(-\frac{0.001}{\sigma_\mu^2}\right) \\
&\propto (\sigma_\mu^2)^{-K_\mu/2-0.001-1} \times \exp\left(-\frac{1}{\sigma_\mu^2}\left(0.5\widetilde{\boldsymbol{\mu}}^{\mathrm{T}}\widetilde{\boldsymbol{\mu}} + 0.001\right)\right),
\end{aligned}
$$

which is the form of an inverse-gamma distribution with parameters $K_\mu/2 + 0.001$ and $0.5\widetilde{\boldsymbol{\mu}}^{\mathrm{T}}\widetilde{\boldsymbol{\mu}} + 0.001$.

Similarly, from the prior for $\widetilde{\boldsymbol{b}}_g \mid \sigma_{b,g}^2 \sim \mathcal{N}_{UK_Y}\left(\mathbf{0}, \sigma_{b,g}^2 \mathbf{I}_{UK_Y}\right)$ for $g = 1, \ldots, G$,

$$
\begin{aligned}
\left[\sigma_{b_g}^2 \,\middle|\, \cdot\,\right] &= \left[\widetilde{\boldsymbol{b}}_g \,\middle|\, \sigma_{b,g}^2\right]\left[\sigma_{b,g}^2\right] \\
&\propto (\sigma_{b_g}^2)^{-UK_Y/2} \exp\left(-\frac{1}{2\sigma_{b_g}^2}\widetilde{\mathbf{b}}_g^{\mathrm{T}}\widetilde{\mathbf{b}}_g\right) \times (\sigma_{b_g}^2)^{-0.001-1} \exp\left(-\frac{0.001}{\sigma_{b_g}^2}\right) \\
&\propto (\sigma_{b_g}^2)^{-UK_Y/2-0.001-1} \times \exp\left(\frac{-1}{\sigma_{b_g}^2}\left(0.5\widetilde{\mathbf{b}}_g^{\mathrm{T}}\widetilde{\mathbf{b}}_g + 0.001\right)\right).
\end{aligned}
$$

This is an inverse-gamma distribution with shape and scale parameters, $UK_Y/2 + 0.001$ and $0.5\widetilde{\mathbf{b}}_g^{\mathrm{T}}\widetilde{\mathbf{b}}_g + 0.001$ respectively.

APPENDIX C

Derivation of the conditional posterior distributions of the parameters of the smoothing step

Given the matrix of K_g orthonormal basis functions $\boldsymbol{\phi}_g$ defined in §3.3 and prior specification in §3.3.1, the derivation for the conditional posterior distribution of the parameters is given below.

C.1 Coefficients

For $i = 1, \ldots, N$ and $g = 1, \ldots, G$,

$$\left[\boldsymbol{\xi}_{i,g} \,\middle|\, \cdot\,\right] = \left[\widetilde{\boldsymbol{\omega}}_{i,g} \,\middle|\, \boldsymbol{\xi}_{i,g}, \sigma^2_{\eta_g}\right] \left[\boldsymbol{\xi}_{i,g} \,\middle|\, \lambda_{k,g}\right]$$

$$\propto \exp\left(\frac{-1}{2\sigma^2_{\eta_g}} \left(\widetilde{\boldsymbol{\omega}}_{i,g} - \boldsymbol{\phi}_g \boldsymbol{\xi}_{i,g}\right)^{\mathrm{T}} \left(\widetilde{\boldsymbol{\omega}}_{i,g} - \boldsymbol{\phi}_g \boldsymbol{\xi}_{i,g}\right)\right) \times \exp\left(\frac{-1}{2} \boldsymbol{\xi}_{i,g}^{\mathrm{T}} \boldsymbol{\Lambda}_g^{-1} \boldsymbol{\xi}_{i,g}\right)$$

where $\operatorname{diag}(\boldsymbol{\Lambda}_g) = (\lambda_{1,g}, \ldots, \lambda_{K_g,g})$.

$$\propto \exp\left(-\frac{1}{2}\left(\frac{1}{\sigma^2_{\eta_g}} \boldsymbol{\xi}_{i,g}^{\mathrm{T}} \boldsymbol{\phi}_g^{\mathrm{T}} \boldsymbol{\phi}_g \boldsymbol{\xi}_{i,g} - \frac{2}{\sigma^2_{\eta_g}} \boldsymbol{\xi}_{i,g}^{\mathrm{T}} \boldsymbol{\phi}_g^{\mathrm{T}} \widetilde{\boldsymbol{\omega}}_{i,g} + \boldsymbol{\xi}_{i,g}^{\mathrm{T}} \boldsymbol{\Lambda}_g^{-1} \boldsymbol{\xi}_{i,g}\right) + \ldots\right)$$

where $\ldots$ represents terms in the exponential that do not depend on $\boldsymbol{\xi}_{i,g}$.

In the exponential term,

$$\frac{1}{\sigma^2_{\eta_g}} \boldsymbol{\xi}_{i,g}^{\mathrm{T}} \boldsymbol{\phi}_g^{\mathrm{T}} \boldsymbol{\phi}_g \boldsymbol{\xi}_{i,g} - \frac{2}{\sigma^2_{\eta_g}} \boldsymbol{\xi}_{i,g}^{\mathrm{T}} \boldsymbol{\phi}_g^{\mathrm{T}} \widetilde{\boldsymbol{\omega}}_{i,g} + \boldsymbol{\xi}_{i,g}^{\mathrm{T}} \boldsymbol{\Lambda}_g^{-1} \boldsymbol{\xi}_{i,g} + \ldots$$

Since $\boldsymbol{\phi}_g^{\mathrm{T}} \boldsymbol{\phi}_g = \mathrm{I}_{K_g}$,

$$= \frac{1}{\sigma^2_{\eta_g}} \boldsymbol{\xi}_{i,g}^{\mathrm{T}} \boldsymbol{\xi}_{i,g} - \frac{2}{\sigma^2_{\eta_g}} \boldsymbol{\xi}_{i,g}^{\mathrm{T}} \boldsymbol{\phi}_g^{\mathrm{T}} \widetilde{\boldsymbol{\omega}}_{i,g} + \boldsymbol{\xi}_{i,g}^{\mathrm{T}} \boldsymbol{\Lambda}_g^{-1} \boldsymbol{\xi}_{i,g} + \ldots$$

$$= \boldsymbol{\xi}_{i,g}^{\mathrm{T}} \left[\frac{1}{\sigma^2_{\eta_g}} \mathrm{I}_{K_g} + \boldsymbol{\Lambda}_g^{-1}\right] \boldsymbol{\xi}_{i,g} - \frac{2}{\sigma^2_{\eta_g}} \boldsymbol{\xi}_{i,g}^{\mathrm{T}} \boldsymbol{\phi}_g^{\mathrm{T}} \widetilde{\boldsymbol{\omega}}_{i,g} + \ldots$$

By matrix factorisation of $\boldsymbol{\xi}_{i,g}$,

$$= \left(\boldsymbol{\xi}_{i,g} - \boldsymbol{\mu}_{\widetilde{\omega}_{i,g}}\right)^{\mathrm{T}} \boldsymbol{\Sigma}^{-1}_{\widetilde{\omega}_{i,g}} \left(\boldsymbol{\xi}_{i,g} - \boldsymbol{\mu}_{\widetilde{\omega}_{i,g}}\right) + \ldots,$$

where $\boldsymbol{\Sigma}_{\widetilde{\omega}_{i,g}} = \left[\boldsymbol{\Lambda}_g^{-1} + \frac{1}{\sigma^2_{\eta_g}} \mathbf{I}\right]^{-1}$ and $\boldsymbol{\mu}_{\widetilde{\omega}_{i,g}} = \frac{1}{\sigma^2_{\eta_g}} \boldsymbol{\Sigma}_{\widetilde{\omega}_{i,g}} \boldsymbol{\phi}_g^{\mathrm{T}} \widetilde{\boldsymbol{\omega}}_{i,g}$. Note that $\boldsymbol{\Sigma}_{\widetilde{\omega}_{i,g}} =$

$\left[\boldsymbol{\Lambda}_g^{-1} + \frac{1}{\sigma^2_{\eta_g}}\mathbf{I}\right]^{-1}$ simplifies to $\operatorname{diag}\left(\frac{\lambda_{1,g}\sigma_{\eta_g^2}}{\lambda_{1,g}+\sigma_{\eta_g^2}}, \ldots, \frac{\lambda_{K_g,g}\sigma_{\eta_g^2}}{\lambda_{K_g,g}+\sigma_{\eta_g^2}}\right)$.

Therefore,

$$\left[\boldsymbol{\xi}_{i,g} \;\middle|\; \cdot\;\right] \propto \exp\left(-\frac{1}{2}\left(\boldsymbol{\xi}_{i,g} - \boldsymbol{\mu}_{\widetilde{\boldsymbol{\omega}}_{i,g}}\right)^{\mathrm{T}} \boldsymbol{\Sigma}^{-1}_{\widetilde{\boldsymbol{\omega}}_{i,g}})\left(\boldsymbol{\xi}_{i,g} - \boldsymbol{\mu}_{\widetilde{\boldsymbol{\omega}}_{i,g}}\right)\right).$$

C.2 Variance parameters

$$\begin{aligned}
\left[\sigma^2_{\eta_g} \;\middle|\; \cdot\;\right] &= \prod_{i=1}^{N}\left[\widetilde{\boldsymbol{\omega}}_{i,g} \;\middle|\; \boldsymbol{\xi}_{i,g}, \sigma^2_{\eta_g}\right] \times \left[\sigma^2_{\eta_g}\right] \\
&\propto (\sigma^2_{\eta_g})^{-NS/2} \exp\left(-\frac{1}{2\sigma^2_{\eta_g}} \sum_{i=1}^{N}\left(\widetilde{\boldsymbol{\omega}}_{i,g} - \boldsymbol{\phi}_g\boldsymbol{\xi}_{i,g}\right)^{\mathrm{T}}\left(\widetilde{\boldsymbol{\omega}}_{i,g} - \boldsymbol{\phi}_g\boldsymbol{\xi}_{i,g}\right)\right) \times (\sigma^2_{\eta_g})^{-a-1}\exp\left(-\frac{b}{\sigma^2_{\eta_g}}\right) \\
&\propto (\sigma^2_{\eta_g})^{-NS/2-a-1} \exp\left(-\frac{1}{\sigma^2_{\eta_g}}\left(0.5\sum_{i=1}^{N}\left(\widetilde{\boldsymbol{\omega}}_{i,g} - \boldsymbol{\phi}_g\boldsymbol{\xi}_{i,g}\right)^{\mathrm{T}}\left(\widetilde{\boldsymbol{\omega}}_{i,g} - \boldsymbol{\phi}_g\boldsymbol{\xi}_{i,g}\right) + b\right)\right).
\end{aligned}$$

This is an inverse-gamma distribution with shape and scale parameters, $NS/2 + a$ and $0.5\sum_{i=1}^{N}\left(\widetilde{\boldsymbol{\omega}}_{i,g} - \boldsymbol{\phi}_g\boldsymbol{\xi}_{i,g}\right)^{\mathrm{T}}\left(\widetilde{\boldsymbol{\omega}}_{i,g} - \boldsymbol{\phi}_g\boldsymbol{\xi}_{i,g}\right) + b$.

Now the prior for $\xi_{ik,g} \sim \mathcal{N}(0, \lambda_{k,g})$ for $i = 1, \ldots, N$,

$$\begin{aligned}
[\lambda_{k,g} \mid \cdot\,] &\propto \prod_{i=1}^{N}[\xi_{ik} \mid \lambda_{k,g}] \times [\lambda_{k,g}] \\
&\propto \lambda_{k,g}^{-N/2}\exp\left(-\frac{1}{2\lambda_{k,g}}\sum_{i=1}^{N}\xi^2_{ik,g}\right) \times \lambda^{-1}_{k,g} \\
&\propto \lambda_{k,g}^{-N/2-1}\exp\left(\frac{-1}{\lambda_{k,g}}0.5\sum_{i=1}^{N}\xi^2_{ik,g}\right).
\end{aligned}$$

Again, this is an inverse-gamma distribution with shape and scale parameters, $N/2$ and $0.5\sum_{i=1}^{N}\xi^2_{ik,g}$.

APPENDIX D

BayesHFLM: An R package for the analysis of Bayesian historical functional linear models

In this Appendix, we provide a documentation of the package being developed to perform the analysis made in this thesis. BayesHFLM implements the method described in this thesis including support functions to construct credibility intervals and to generate predictions. The package can be found in GitHub/yovnajunglee/BayesHFLM. The code is implemented using Rccp with an R user interface. It requires prior installation of the R packages mgcv, refund and fda.

`simulate_hflm` *Simulate data from the lag/historical functional linear model*

Description

Simulates data from a lag/historical functional linear model. This model assumes that the functional observation and functional predictors are observed on a common grid.

Usage

```
simulate_hflm(nobs, n.tau, delta, varx, eSNR, plot = TRUE)
```

Arguments

`nobs`	Number of functional data.
`n.tau`	Size of grid.
`beta1`	True first historical surface. Options include `null` for a null surface, `bimodal-lagged`.
`beta2`	True second historical surface. Same as options as `beta1`.
`varx1`	Variance of noise for first functional predictor.
`varx2`	Variance of noise for second functional predictor.
`eSNR`	Expected signal-to-noise ratio for functional response.
`delta`	Lag parameter for a LFLM. Defaults to `NULL` for a HFLM.

`seed`	Sets the seed when generating random data sets to ensure reproducibility of the simulation study.
`plot`	Logical. If `TRUE` (default), plots the functional response, predictors and true historical surfaces.

Values

Returns a list with:

`ytau`	$N \times T$ matrix of functional responses Y.
`ftau`	$N \times T$ matrix of true underlying function, $f_i(\tau)$.
`x1tau`	$N \times S$ matrix of noisy functional covariates X_1.
`x2tau`	$N \times S$ matrix of noisy functional covariates X_2.
`theta1`	Matrix of the first true historical surface.
`theta2`	Matrix of the second true historical surface.
`vary`	Variance of noise for the functional response calculated from the eSNR.

`bayeshflm` *Fits a two-variable Bayesian historical functional linear model*

Description

Fits a Bayesian historical functional linear model. This model assumes 1) functional observations and predictors are observed on a common regular grid, 2) i.i.d error terms for both functional response and predictors.

Usage

```
bayeshflm( Ymat, Xmat1, Xmat2, taus, Ss, interceptInfo = list(Fk1 =
"ts", K1 = 10), tensorInfo = list(Fk = "bs", Psi = "bs", U = 10, K =
10), fpcaInfo = list(knots = 15, npc = tensorInfo$U), lag = NULL, niter
= 1000, nburn = 0.5, alpha_start = NULL, mcmc_hyper = list(i1 = 0.001,
i2 = 0.001, a = 0.001, b = 0.001), smooth = TRUE, method = "ind" )
```

Arguments

`Ymat`	$N \times T$ matrix of functional responses.
`Xmat1`	$N \times S$ matrix of functional predictors.
`Xmat2`	$N \times S$ matrix of functional predictors.
`taus`	Points at which the functional response is sampled.
`Ss`	Points at which the functional predictors are sampled.
`interceptInfo`	A list with arguments `Fk1` to specify the basis function .e.g. `"bs"` for B-splines, `"ts"` for thin-plate splines. Refer to `smoothCon()` from `mgcv` package. `K1` to specify the dimension of the basis.
`tensorInfo`	A list with arguments `Fk` and `Psi` to specify the marginal bases for the tensor product .e.g. `"bs"` for B-splines, `"ts"` for thin-plate splines. Refer to `smoothCon()` from `mgcv` package. `U,K` to specify the dimension of the marginal bases.
`fpcaInfo`	A list with arguments `npc` for the number of FPCs used to model the functional predictors. Defaults to `U`. `knots` the number of knots used to estimate the FPCs (refer to `fpca.face()` from `refund` package).
`lag`	Lag parameter for a LFLM. Defaults to `NULL` for a HFLM.
`alpha_start`	Starting parameters for the regression parameters. If `NULL`, the starting parameters are generated from $\mathcal{U}(0,\ 1)$.
`nburn`	Percentage of MCMC samples to keep.
`niter`	Number of iterations of the MCMC sampler.
`nburn`	Percentage of MCMC samples to keep.
`mcmc_hyper`	A list of prior hyperparameters. $\mathtt{i}_1, \mathtt{i}_2$ are the hyperparameters for the inverse-gamma prior on $\sigma_e^2, \sigma_{\eta_g}^2$. Defaults to 0.001 for both. $\mathtt{a}, \mathtt{b}$ are the hyperparameters for the inverse-gamma prior on $\sigma_\mu^2, \sigma_{b,g}^2$. Defaults to 0.001 for both.
`smooth`	Logical. If `TRUE` (default) includes a smoothing step for functional predictors.
`method`	`ind` (default) for the two-stage analysis or `joint` for a joint estimation. This is only important if `smooth = TRUE`.

Values

Returns a list with:

`mcmc`	MCMC samples of model parameters and fitted values.
`betahat`	A list of the posterior means of the post burn-in MCMC samples of $\theta_g(v,\tau)$.
`post_mu`	MCMC samples of $\mu(\tau)$.
`post_beta1, post_beta2`	MCMC samples of $\theta_g(v,\tau)$.
`muhat`	Posterior mean of the post burn-in MCMC samples of $\mu(\tau)$.
`Yfit`	Posterior mean of the MCMC samples of the fitted values.
`F1,F2`	Matrix of eigenfunctions for the first and second predictors.
`Lmat`	Indicator matrix for the integration limits.
`smooth`	Logical. Indicates whether the functional predictors were smoothed.

`credint_hflm` *Produces joint or pointwise credibility intervals of the functional parameters of the model.*

Description

Produces joint or pointwise credibility intervals of the functional parameters of the model.

Usage

```
credint_hflm(object, burnin, level = 0.95, type = c("pointwise",
"joint"))
```

Arguments

`object`	Output from `bayeshflm`.
`burnin`	Indices of post burn-in sample. Should be set as `(nburn*niter):niter`.
`level`	The credibility level required.
`type`	The specification of which type of credibility interval should be generates. Only `"pointwise"` or `"joint"` CIs are supported.

Values

A matrix with columns giving the estimated, lower and upper credibility limits for each parameter.

`predict_hflm` *Generates predictions for a HFLM*

Description

Generates predictions for a HFLM where the predictors were not smoothed, or where the predictors were smoothed using the two-stage analysis.

Usage

```
predict_hflm(hflm_mcmc, X1_new, X2_new, taus, burnin, interval = TRUE,
level = 0.95)
```

Arguments

`hflm_mcmc`	Output from `bayeshflm`.
`X1_new, X2_new`	Matrix of functional predictors.
`burnin`	Indices of post burn-in sample. Should be set as `(nburn*niter):niter`.
`interval`	Logical. If `TRUE`, generates credibility intervals for the forecast.
`level`	The credibility level required.

Values

A matrix with columns giving the predictions, and their lower and upper credibility limits.

`predict_hflm_joint` *Generates predictions for the joint estimation HFLM*

Description

Generates predictions for the joint estimation HFLM.

Usage

```
predict_hflm_joint(hflm_mcmc, X1_new, X2_new, taus, burnin, interval =
TRUE, level = 0.95)
```

Arguments

`hflm_mcmc`	Output from `bayeshflm`.
`X1_new, X2_new,`	Matrix of functional predictors.
`burnin`	Indices of post burn-in sample. Should be set as `(nburn*niter):niter`.
`interval`	Logical. If `TRUE`, generates credibility intervals for the forecast.
`level`	The credibility level required.

Values

A matrix with columns giving the predictions, and their lower and upper credibility limits.

APPENDIX E

Simulation study results

Here, we display the results of the simulation set-ups with $N = \{25, 100\}$ curve data.

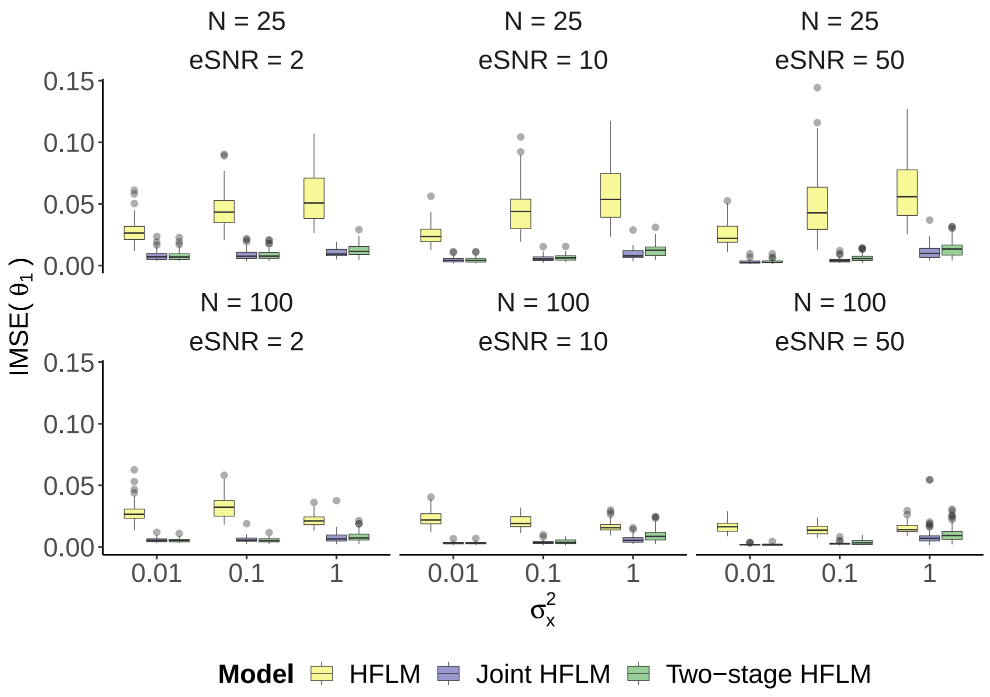


Figure E.1. *IMSE of the first historical surface,* $\theta_1(v, \tau)$.

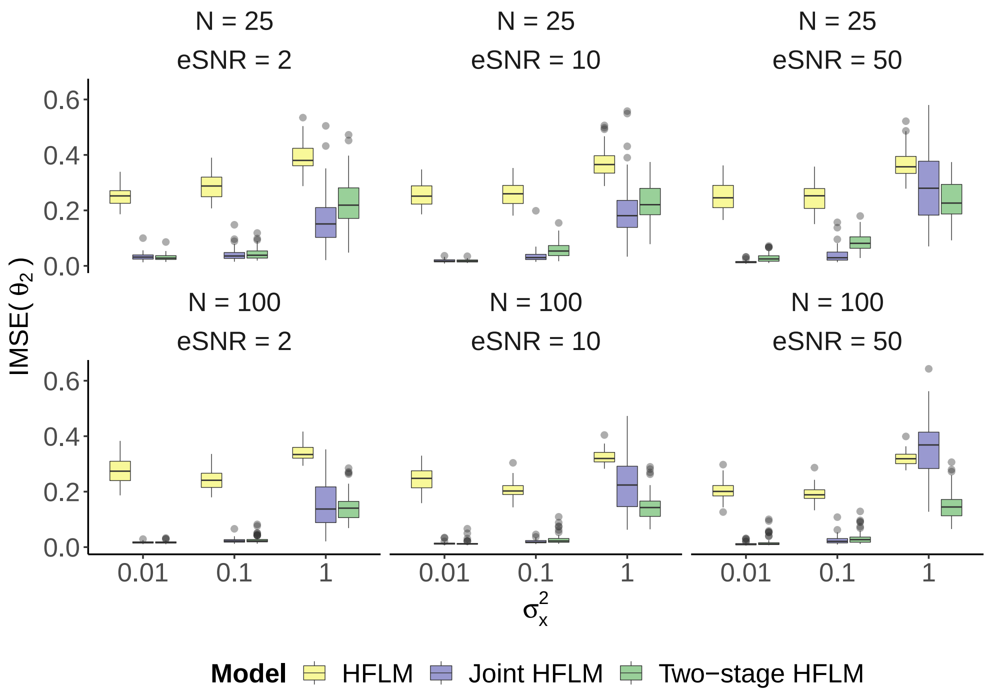


Figure E.2. *IMSE of the second historical surface, $\theta_2(v, \tau)$.*

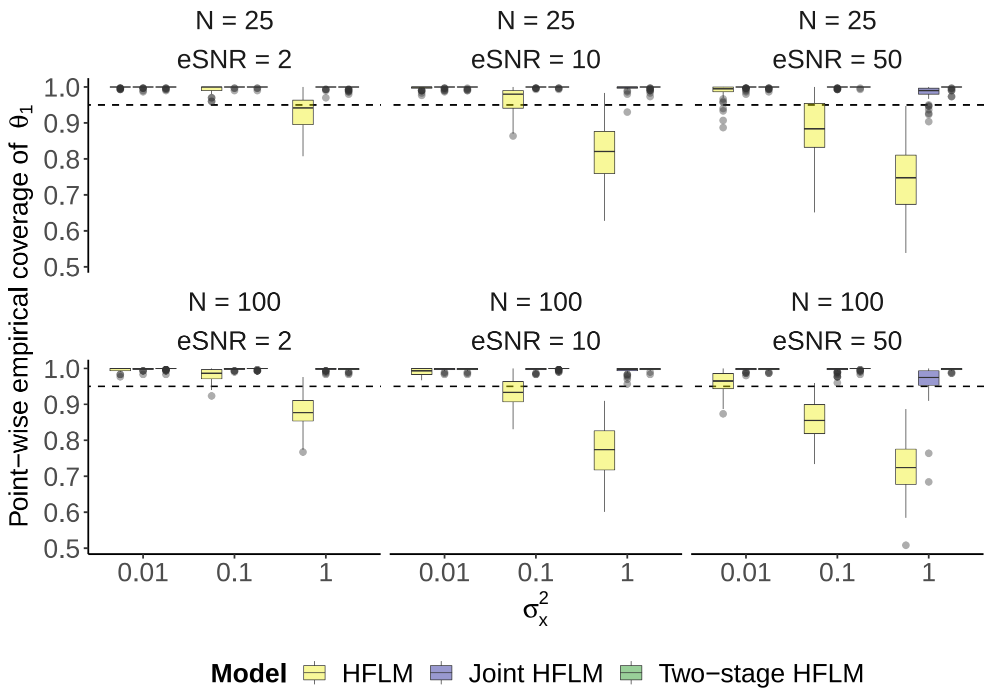


Figure E.3. *Empirical coverage of the first historical surface, $\theta_1(v, \tau)$.*

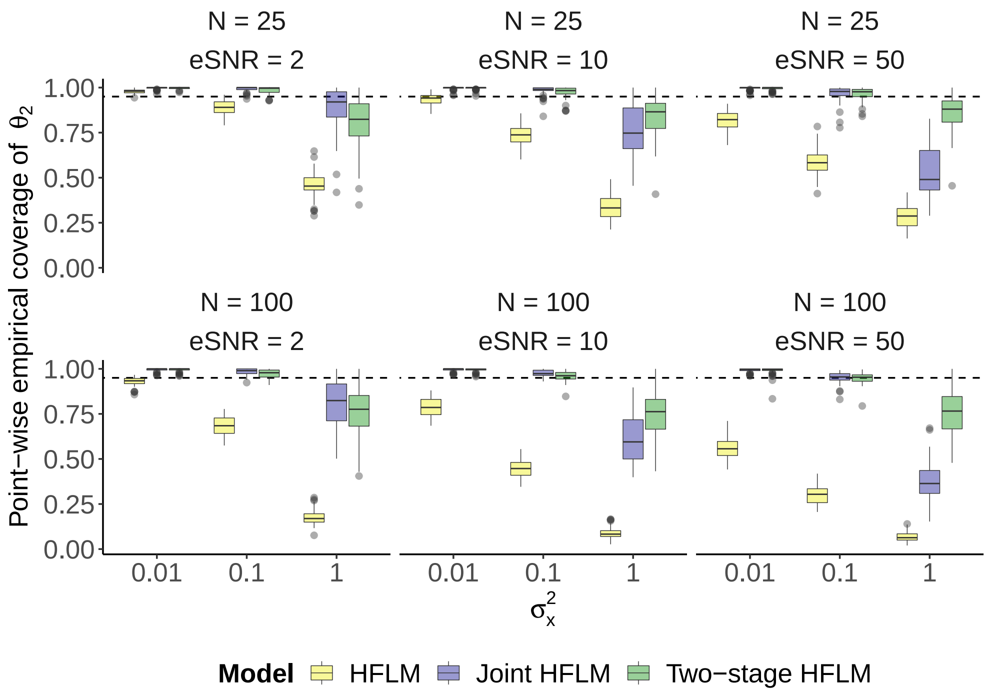


Figure E.4. *Empirical coverage of the second historical surface,* $\theta_2(v, \tau)$.

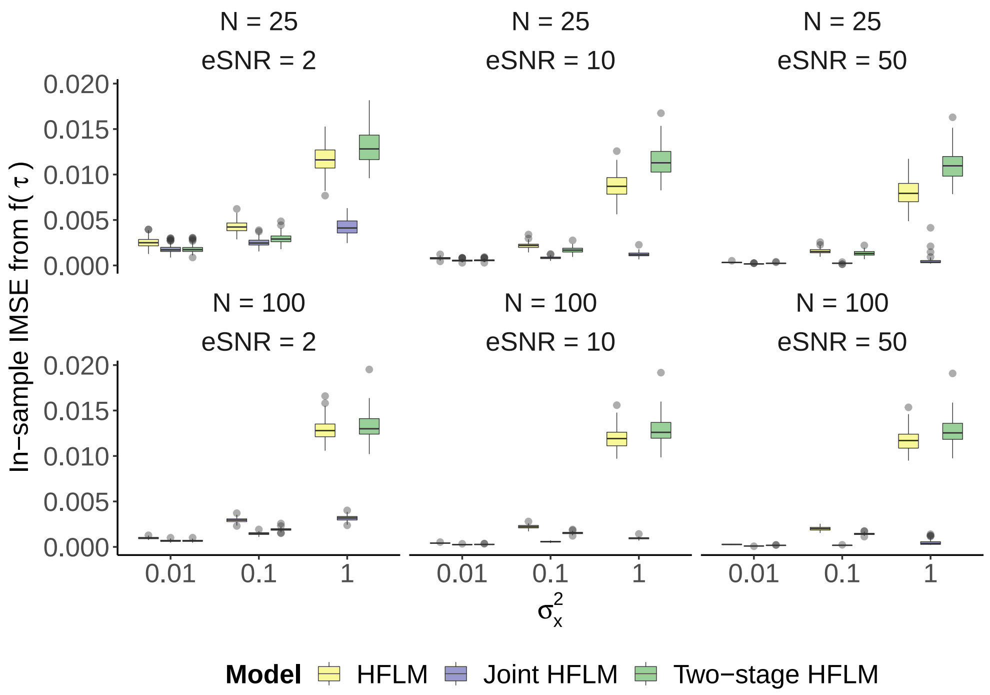


Figure E.5. *In-sample IMSE relative to the true underlying function,* $f(\tau)$.

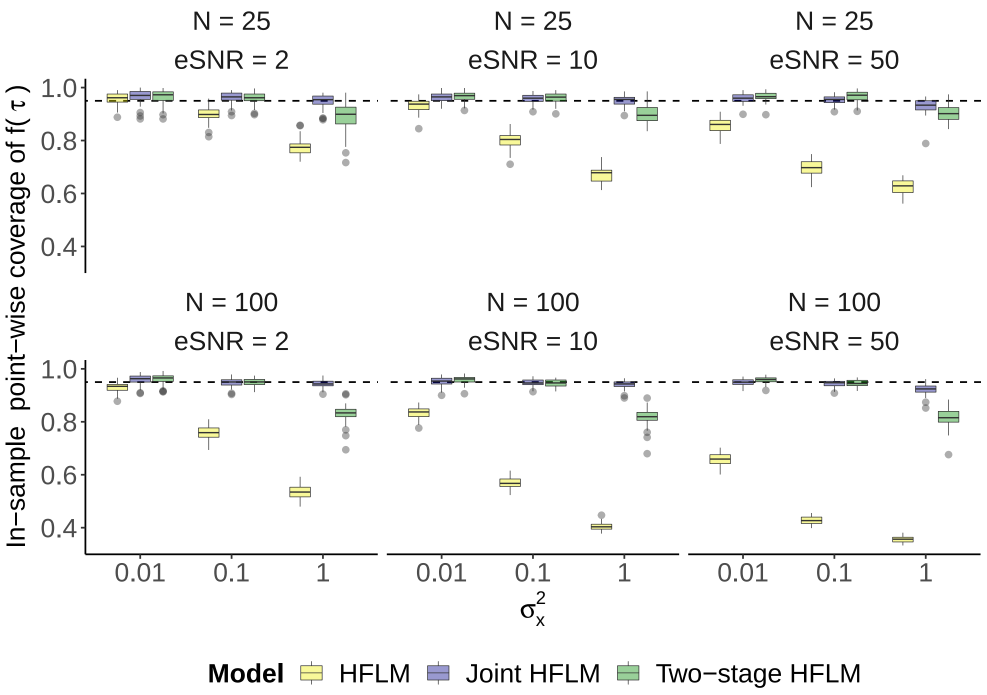


Figure E.6. *In-sample empirical coverage the true underlying function, $f(\tau)$.*

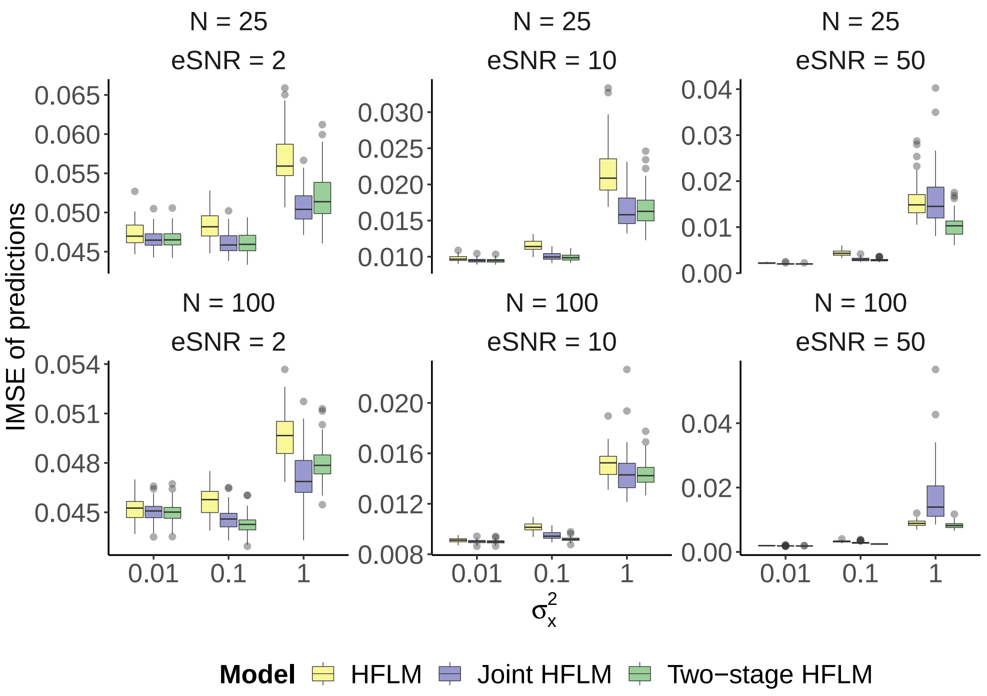


Figure E.7. *Out-of-sample IMSE relative to the observed test data, $Y_{pred}(\tau)$.*

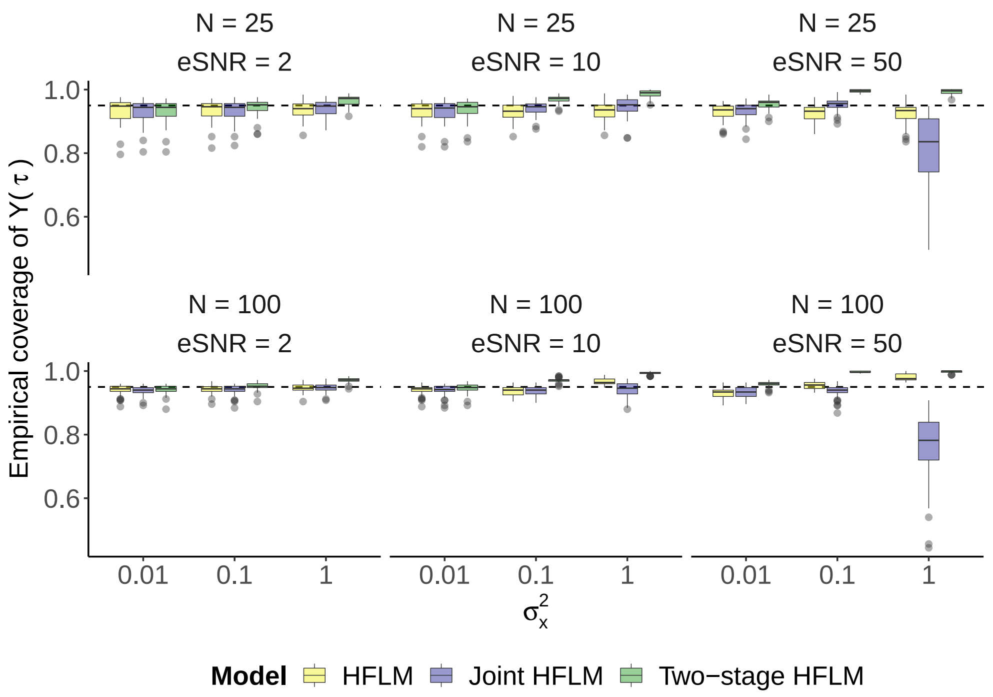


Figure E.8. *Out-of-sample empirical coverage of the observed test data, $Y_{pred}(\tau)$.*

APPENDIX F

Convergence diagnostics for simulation study

Table (F.1) and Figure below correspond to the results of the convergence diagnostics for the simulation set-up: $N = 50$, $T = 25$, eSNR= 10 and $\sigma_x^2 = 0.01$. 10000 iterations of the MCMC sampler was ran and a burn-in sample of 50% was kept. To assess convergence, we ran 3 parallel chains with different starting values used for the Gelman-Rubin's (PSRF) test which gave values close to 1, well below the suggested cut-off 1.2. We also visually inspect the traceplots and obtain lag autocorrelation values of the MCMC chain. We also compare the resulting densities of the post burn-in MCMC samples. Our results show evidence of convergence for all three models. We illustrate the results for the two-stage Bayesian model only by drawing 4 estimated coefficients b_{uk}. The rest of the parameters and the other two models looked the same.

	$\hat{b}_{10,8,1}$	$\hat{b}_{5,10,1}$	$\hat{b}_{9,3,2}$	$\hat{b}_{1,6,2}$
PSRF	1.000	1.000	1.001	1.000
95% upper C.I	1.000	1.001	1.002	1.000

Table F.1. *Simulation Study: Gelman-Rubin's test for convergence.*

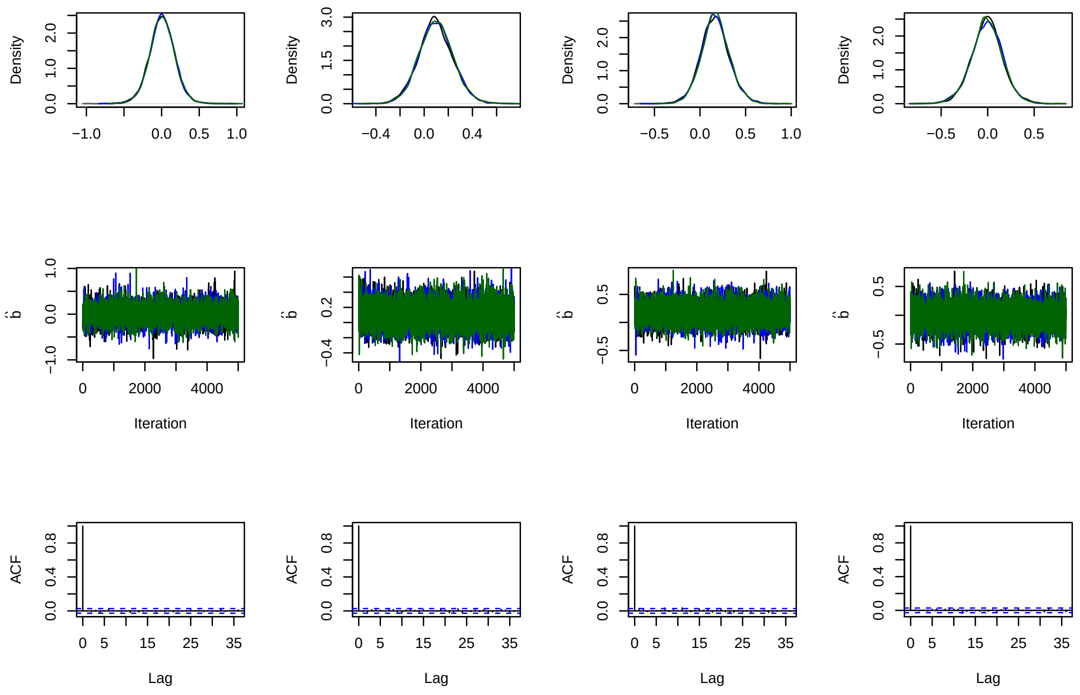


Figure F.1. *Simulation Study: Top row corresponds to the densities of the MCMC chains for each parameter. Middle row are the traceplots of the MCMC chains. Bottom row are the autocorrelation plots.*

APPENDIX G

Convergence diagnostics for application

Similarly, 3 parallel chains were ran and the Gelman-Rubin's test was used to test for convergence. We further investigated traceplots, densities of the chains and ACF plots. Our results showed evidence of convergence. We illustrate the results for the two-stage Bayesian model using predictors temperature and wind speed only by drawing 4 estimated coefficients b_{uk}. The rest of the parameters and the other two models looked the same.

	$\hat{b}_{10,8,1}$	$\hat{b}_{5,10,1}$	$\hat{b}_{9,3,2}$	$\hat{b}_{1,6,2}$
PSRF	1.000	1.000	1.000	1.000
95% upper C.I	1.000	1.001	1.001	1.000

Table G.1. *Application: Gelman-Rubin's test for convergence.*

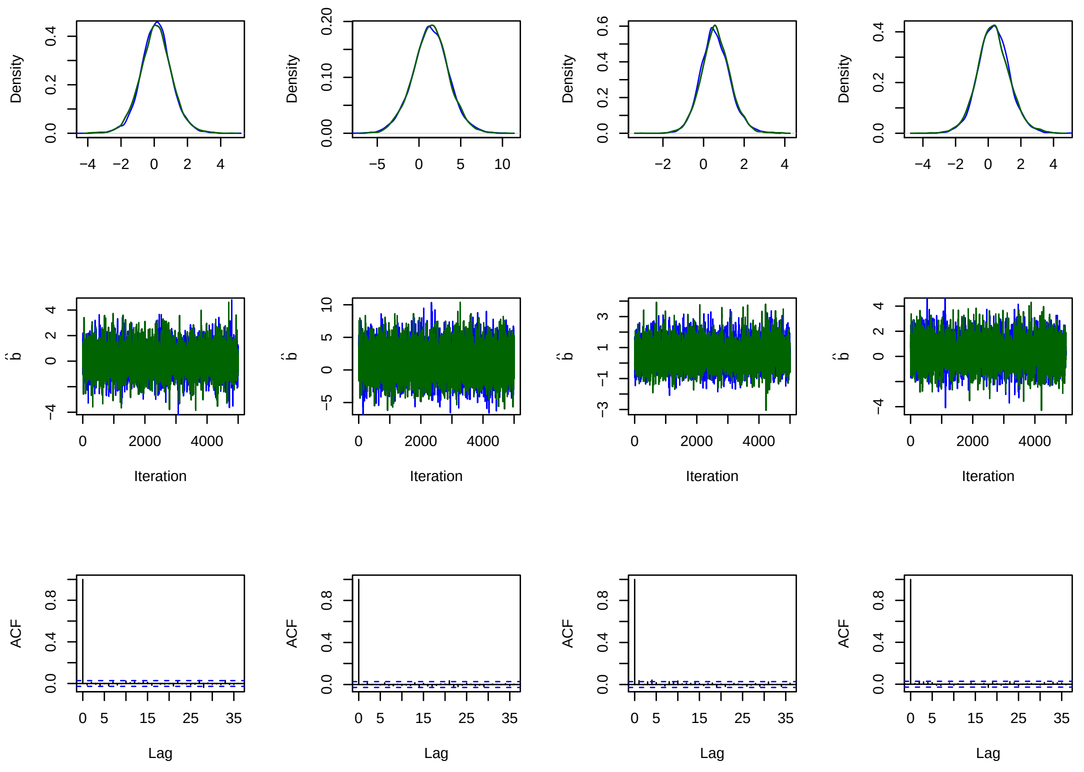


Figure G.1. *Application: Top row corresponds to the densities of the MCMC chains for each parameter. Middle row are the traceplots of the MCMC chains. Bottom row are the autocorrelation plots.*

APPENDIX H

R code: Simulation Study

```r
### Simulation runs for HFLM model

library(BayesHFLM)
library(mgcv)
library(refund)
library(foreach)
library(doMC)

registerDoMC(8)


nsims <-  50 # Number of simulation runs
niter <-  10000 # Number iterations in Gibbs sampler
nburn <-  0.5

varx_sim <-  c(0.01,0.1,1) # var(X) values under consideration
esnr_y <-  c(2, 10, 50) # eSNR(y) values under consideration

nobs <-  c(25, 50, 100)
ntaus <-  25

combs <-  expand.grid(esnr_y, varx_sim, nobs) # all possible
    ↪set-ups for each case

test_seed <-  1546482 # set seed for test set

ntest <-  10 # no. of observations for test set

sim_results <-  matrix(NA, nrow = nrow(combs)*nsims, ncol = 15)
colnames(sim_results) <-  c("esnr","varx","nobs", "mseb1",
    ↪"mseb2", "covpw_b1", "covpw_b2","covjoint_b1","covjoint_b2",
                          "mseft", "covft","mse_oos_y",
                             ↪"mse_oos_f",  "cov_oos_y",
                             ↪"cov_oos_f")


# Function to evalute coverage
coverage <- function(true, lower, upper){

  cc <- NA
  if(true >= lower && true <= upper){
    cc <- 1
  }
  else{cc <- 0}

}

for(cc in 1:nrow(combs)){
  # Keep combination fixed, run simulation

  # Generate new test set with new seed
  test.dat <- simulate_hflm(nobs = ntest, n.tau = 25,
                                      beta1 = "p1",
                                         ↪beta2="bimodal-lagged",
                                         ↪eSNR = combs[cc,1],
                                      varx1 = combs[cc,2],
                                         ↪varx2 = combs[cc,2],
                                      delta = NULL, seed =
                                         ↪test_seed, plot =
                                         ↪FALSE)
  taus <- seq(0,1,length.out = ntaus)
  Ss <-  taus

  # Run model for each simulated data set

  results <-  foreach(nsim = 1:nsims, .combine=rbind, .packages
      ↪= c("BayesHFLM",
                                                               "mgcv",
                                                                  ↪"refund"))
                                                                  ↪%dopar%
                                                                  ↪{
    print(paste0("Combination ", cc, " out of ", nrow(combs), ":
       ↪Simulation no. ", nsim))
    # Simulate data set (set the seed)
    sim.dat = simulate_hflm(combs[cc, 3], ntaus, beta1  = "p1",
       ↪beta2 = "bimodal-lagged", eSNR = combs[cc,1],
                              varx1 = combs[cc,2], varx2 =
                                 ↪combs[cc,2], seed = nsim,
                              plot = FALSE)
    hflm_model <- bayeshflm(sim.dat$ytau, sim.dat$x1tau,
       ↪sim.dat$x2tau,
                              taus, Ss, niter = niter,
                              method = "ind", smooth = FALSE, nburn
                                 ↪= 0.5)

    # Store MSE of betas
    mse_beta1_sim <-  mean((hflm_model$beta_hat$beta1 -
       ↪sim.dat$theta1.s.tau)^2)
    mse_beta2_sim <-   mean((hflm_model$beta_hat$beta2 -
       ↪sim.dat$theta2.s.tau)^2)

    burnin <-  (nburn*niter):niter
    # Obtain credibility intervals
    pw_ci <-  credint_hflm(hflm_model, type = "pointwise")
    joint_ci <-  credint_hflm(hflm_model, type = "joint")

    lag <-  which(c(hflm_model$Lmat) ==1 )
    true_beta <-
       ↪c(c(sim.dat$theta1.s.tau)[lag],c(sim.dat$theta2.s.tau)[lag])
    np = length(c(sim.dat$theta1.s.tau)[lag])
    cov_prob <- NULL

    # Test for coverage
    cov_prob_pw_b <- (apply(cbind(true_beta,
       ↪pw_ci[-c(1:ntaus),]), 1, function(x){
      coverage(x[1], x[2], x[3])
    }))

    cov_prob[1] <- mean(cov_prob_pw_b[1:np])
    cov_prob[2] <- mean(cov_prob_pw_b[(np+1):(2*np)])

    cov_prob_j_b <- (apply(cbind(true_beta,
       ↪joint_ci[-c(1:ntaus),]), 1, function(x){
      coverage(x[1], x[2], x[3])
    }))

    cov_prob[3] <- mean(cov_prob_j_b[1:np])
    cov_prob[4] <- mean(cov_prob_j_b[(np+1):(2*np)])


    # Store predicted values MSE

    test_y <- predict_hflm(hflm_model, test.dat$x1tau,
       ↪test.dat$x2tau, taus, 0, burnin = burnin)
    pred_y <- matrix(test_y[,1], ncol = ntaus, byrow = TRUE)

    #---- OOS fit

    mse_oos_y <- mean((test.dat$ytau-pred_y)^2)
    mse_oos_f <- mean((test.dat$ftau-pred_y)^2)

    # Cov. prob
    cov_oos_y = mean(apply(cbind(c(t(test.dat$ytau)),
       ↪test_y[,2:3]), 1, function(x){
      coverage(x[1], x[2], x[3])
    }))

    cov_oos_f = mean(apply(cbind(c(t(test.dat$ftau)),
       ↪test_y[,2:3]), 1, function(x){
      coverage(x[1], x[2], x[3])
    }))

    # --- In-sample fit

    # MSE on training set (compare to f(t))
    mse_is_ft <-
       ↪mean((c(t(sim.dat$ftau))-c(t(hflm_model$Yfit)))^2)

    #Coverage for f

    cov_ft <-  mean(apply(cbind(c(t(sim.dat$ftau)),
       ↪t(apply(hflm_model$mcmc$Yfit[,burnin], 1, quantile,
       ↪probs = c(0.025, 0.975))
    ) , 1, function(x){
      coverage(x[1], x[2], x[3])
    }))

    rm(hflm_model)
    # Return objects
    c(combs[cc,], mse_beta1_sim, mse_beta2_sim, cov_prob,
       ↪mse_is_ft, cov_ft,
      mse_oos_y,  mse_oos_f,  cov_oos_y,  cov_oos_f)


  }

  sim_results[(temp):(temp + nsims - 1),] <-
    ↪matrix(unlist(results), ncol = 15, byrow = F)
  temp = temp + nsims
}


save.image("~/SimStudy/Results/sim_False_25.RData")
```

APPENDIX I

R code: Application

```r
### Load packages

library(splines)
library(forecast)
library(mgcv)
library(refund)
library(far)
library(BayesHFLM)
library(fdaACF)
library(readxl)
library(lattice)
library(ggplot2)
library(lubridate)
library(reshape2)
library(readr)
library(ftsa)
theme_set(theme_classic()+  theme(axis.text = element_text(size
    ↪= 18), axis.title = element_text(size = 18)))


### Misc. functions for data cleaning and viz.


# ---- Function to plot daily data

plot_daily = function(data, ...){
  plot(data[1,], type = 'l', ylim =
      ↪c(min(na.omit(data)),max(na.omit(data))), ...)
  for(i in 1:nrow(data)){
    lines(data[i,], ...)
  }
}


# ---- Pre-processing

# i) Natural cubic splines interpolation for sparsely missing
# obs. in functional response
```

```r

pre_process <- function(YY, tt){
  # Construct basis functions
  ns_val <- cbind(1,ns(tt, 24))
  post_YY <- matrix(NA, ncol = 24, nrow = nrow(YY))
  # For each row, interpolate curve
  # only if there are missing obs in that curve
  for(i in 1:nrow(YY)){
    if(sum(is.na(YY[i,]))>0){
      find_missing <- which(is.na(YY[i,]))
      NN <- ns_val[-find_missing,]
      y <- matrix(YY[i,-find_missing], ncol = 1)
      Bet <- solve(crossprod(NN)+diag(1e-6, ncol = ncol(NN),
         ↪nrow = ncol(NN)))%*%t(NN)%*%y
      #print(length(ns_val%*%Bet))
      post_YY[i,] = ns_val%*%Bet

    }else{
      post_YY[i,] = YY[i,]
    }
  }

  return( post_YY)
}

# ii) Function to create wide data frame
# (column = hour, row = day)

data.wide <- function(x, ndays, nhours, days, pol){
  df.mat <- matrix(NA, nrow =  ndays, ncol =  nhours)
  nday <- 1
  for(i in days){
    df.mat[nday,]<-x[i:(i+23)]
    nday <- nday + 1
  }
  return(df.mat)
}


data.wide.df <- function(x, ndays, nhours, days, pol){
  df.mat <- data.frame("", nrow =  ndays, ncol =  nhours)
  nday <- 1
  for(i in days){
    df.mat[nday,]<-x[i:(i+23)]
    nday <- nday + 1
  }
  return(df.mat)
}

# iii) Function for Box Cox transformation

box.cox <- function(x, lambda){
  xt <- NA
  if(lambda==0){
    xt <- log(x)
  }
  else{
    xt <- (x^lambda - 1)/lambda
  }
  hist(xt)
  return(xt)
}


# ---- Import data

# i) Ozone data

# 2016
X2016_O3_Data <-
   ↪read_excel("~/Documents/GitHub/airpol-bhflm/Application/2016Data/2016_O3_Data
   ↪skip = 2)

# 2017
X2017_O3_Data <-
   ↪read_excel("~/Documents/GitHub/airpol-bhflm/Application/2017Data/Plattekloof2
   ↪skip = 2)

# 2018
X2018_O3_Data  <-
   ↪read_delim("~/Documents/GitHub/airpol-bhflm/Application/2018Data/Plattekloof2
                               ";", escape_double = FALSE, trim_ws
                                  ↪= TRUE,
                               skip = 2)

# ii) Meteorological data
```

```
plattekloof_Metdata_2016 <-
    ↪read_excel("~/Documents/GitHub/airpol-bhflm/Application/2016Data/Bothasig_Met
    ↪skip = 2)
Bothasig2017 <-
    ↪read_excel("~/Documents/GitHub/airpol-bhflm/Application/2017Data/Bothasig2017
                                   skip = 2)
Bothasig2018 <-
    ↪read_delim("~/Documents/GitHub/airpol-bhflm/Application/2018Data/Bothasig2018
                                   ";", escape_double = FALSE, trim_ws =
                                       ↪TRUE,
                                   skip = 2)[-c(8762:8769), ]
# ---- Pre-processing

plattekloof_o3<- as.numeric(c(X2016_O3_Data$`Plattekloof AQM
    ↪Site`[-(1:2)],
                                       X2017_O3_Data$O3[-1],
                                           ↪X2018_O3_Data$O3[-c(1,8762:8769)]))

# ---- Extract no. of days
nobs <- length(plattekloof_o3);nobs
day16 <- length(X2016_O3_Data$`Plattekloof AQM Site`[-(1:2)])
day17 <- length(X2017_O3_Data$O3[-1])
day18<- length(X2018_O3_Data$O3[-c(1,8762:8769)])
days <- seq(1, nobs, by = 24)

# ----- Define tau
time = 0:23
# Standardise time so that it lies between 0 and 1
std_time = (time-min(time))/(max(time)-min(time))



# ---- Plot data as a time-series
plot(ts(plattekloof_o3))

# ---- Transform data in wide format
plattekloof_o3_df <- data.wide(plattekloof_o3, length(days), 24,
    ↪days)[1:1013,]

# ---- Construct a matrix of time indices
time.mat = data.wide(unique(X2016_O3_Data$`Date &
    ↪Time`[-(1:2)]),length(seq(1, day16, by = 24)), 24, seq(1,
    ↪day16, by = 24))
```

```
time.mat2017 = data.wide(unique(X2017_O3_Data$`Date &
    ↪Time`[-1]),length(seq(1, day17, by = 24)), 24, seq(1,
    ↪day17, by = 24))
time.mat2018 = data.wide(unique(X2018_O3_Data$`Date &
    ↪Time`[-c(1,8762:8769)]),length(seq(1, day18, by = 24)), 24,
    ↪seq(1, day18, by = 24))
all_time <- rbind(time.mat,time.mat2017,time.mat2018)

# ---- label days (eg Monday = 1, Tuesday = 2 ,...)
# ( start from 5 as 1/1/16 was a Friday)
day_labels = c(rep(c(5:7, 1:4),
    ↪ceiling(nrow(plattekloof_o3_df)/7)),5)[1:nrow(plattekloof_o3_df)]


# ---- Extract weekdays only
time.wd = all_time[which(day_labels %in% 1:5),]

# ---- Extract weekdays only
plattekloof_o3_df_wd <- plattekloof_o3_df[which(day_labels %in%
    ↪1:5),]
plot_daily(plattekloof_o3_df_wd)

# ---- Find days for which missing obs >= 3
indx <- apply(plattekloof_o3_df_wd, 1,
    ↪function(x){sum(is.na(x))})

plattekloof_o3_df_wd <- plattekloof_o3_df_wd[-(which(indx>=3)),]

# ---- Set the following rows as train & test set

final = 1:nrow(plattekloof_o3_df_wd )
train = 1:(0.80*length(final))
test = final[-(1:(0.80*length(final)))]

View(time.wd[-(which(indx>=3)),][test, ])
# ---- Extract training and test set and pre-process Ys

plattekloof_o3_wd_train <-
    ↪pre_process(plattekloof_o3_df_wd[train,], std_time)
plot_daily(plattekloof_o3_wd_train)
plattekloof_o3_wd_test =
    ↪pre_process(plattekloof_o3_df_wd[test,], std_time)

```

```r
dim(plattekloof_o3_wd_train)
dim(plattekloof_o3_wd_test)

plot_daily(plattekloof_o3_wd_train)
plot_daily(plattekloof_o3_wd_test)

# ---- Extract meteorological info.

plattekloof_meteo<- plattekloof_Metdata_2016[-1,]


# --- Temperature data

Bothasig2018$Temp_2m <- gsub(",", ".", Bothasig2018$Temp_2m)

Temp = c(as.numeric(plattekloof_meteo$Temp),
         as.numeric(Bothasig2017$Temp_2m[-1]),
         as.numeric(Bothasig2018$Temp_2m[-1]))

# ------- i) In wide format

Temp_mat = data.wide(Temp, length(days), 24, days)[1:1013,]

# ------ ii) Extract weekdays only
#
Temp_mat_wd = Temp_mat[which(day_labels %in%
   ↪1:5),][-which(indx>=3), ]


# ------ ii) Test & Train set
Temp_test = Temp_mat_wd[test,]
Temp_train = Temp_mat_wd[train,]
dim(Temp_train)
plot_daily(Temp_test)

apply(Temp_train, 1, function(x){sum(is.na(x))}) # Find missing
   ↪observations


# --- Relative Humidity data

Bothasig2018$RH <- gsub(",", ".", Bothasig2018$RH)

```

```
RH = c(as.numeric(plattekloof_meteo$RH),
       as.numeric(Bothasig2017$RH)[-1],
          ↪as.numeric(Bothasig2018$RH)[-1])

# ------- i) In wide format

RH_mat = data.wide(RH, length(days), 24, days)[1:1013,]


# ------ ii) Extract weekdays only
RH_mat_wd = RH_mat[which(day_labels %in% 1:5),][-which(indx>=3),]

# ------ ii) Test & Train set
RH_test = RH_mat_wd[test,]
RH_train = RH_mat_wd[train,]

dim(RH_train)
plot_daily(RH_test)

# --- Windspeed data
Bothasig2018$`Wind Speed V` <- gsub(",", ".", Bothasig2018$`Wind
    ↪Speed V`)

WS = c(as.numeric(plattekloof_meteo$`Wind Speed V`),
       as.numeric(Bothasig2017$`Wind Speed V`)[-1],
          ↪as.numeric(Bothasig2018$`Wind Speed V`)[-1])

# ------- i) In wide format

WS_mat = data.wide(WS, length(days), 24, days)[1:1013,]

# ------ ii) Extract weekdays only

WS_mat_wd = WS_mat[which(day_labels %in% 1:5),][-which(indx>=3),]

# ------ ii) Test & Train set

WS_test = WS_mat_wd[test,]
WS_train = WS_mat_wd[train,]
dim(WS_train)
plot_daily(WS_test)

na_temp = which(apply(Temp_train, 1,
↪function(x){sum(is.na(x))})>0)
na_rh = which(apply(RH_train, 1, function(x){sum(is.na(x))})>0)
na_WS= which(apply(WS_train, 1, function(x){sum(is.na(x))})>0)


miss_x <- unique(c(na_temp,na_rh,na_WS)) # Find missing X

# ----- Define X matrices, scaled by overall mean and variance

Temp.mean  =   mean(Temp_train[-miss_x,])
Temp.sd  =  sd(Temp_train[-miss_x,])
Xmat1 =  matrix(apply(Temp_train[-miss_x,] , 1,
    ↪function(x){(x-Temp.mean)/Temp.sd} ), byrow = T, ncol = 24)


RH.mean  = mean(RH_train[-miss_x,])
RH.sd  = sd(RH_train[-miss_x,])
Xmat2 =  matrix(apply(RH_train[-miss_x,] , 1,
    ↪function(x){(x-RH.mean)/RH.sd} ), byrow = T, ncol = 24)

WS.mean  =  mean(WS_train[-miss_x,])
WS.sd  = sd(WS_train[-miss_x,])
Xmat3 <- matrix(apply(WS_train[-miss_x,] , 1,
    ↪function(x){(x-WS.mean)/WS.sd} ), byrow = T, ncol = 24)

plot_daily(Xmat1)
plot_daily(Xmat2)
plot_daily(Xmat3)


# ----  Obtain estimate for lambda of Box-Cox

lamfit = BoxCox.lambda((c(t(plattekloof_o3_wd_train[-miss_x,
    ↪]))));lamfit

# ----  Transform Y

Ymat_bc =
    ↪matrix(box.cox((c(t(plattekloof_o3_wd_train[-miss_x,]))),
    ↪lamfit), byrow = T, ncol =24)
hist(c(t(Ymat_bc)))

#---- Scale by overall mean and sd
```

```r

Ymat_bc1 <- (Ymat_bc-mean(Ymat_bc))/sd(Ymat_bc)
Y.mean = mean(Ymat_bc)
Y.sd = sd(Ymat_bc)
plot_daily(Ymat_bc)
hist(Ymat_bc1)

# --- Run BayesHFLM models

niter = 5000; burnin = (0.5*niter):niter


# A. Predictors: Temperature and RH

pk_o3_F = bayeshflm(Ymat_bc1, Xmat1 = Xmat1, Xmat2 = Xmat2,
    ↪std_time, std_time,
                    interceptInfo = list(Fk1="ts",K1 = 10),
                    tensorInfo = list(Fk = "bs", Psi = "bs", U =
                       ↪10, K = 10),
                    fpcaInfo = list(knots = 15, npc = 10),
                    method = "ind" , niter = niter,
                    smooth = F)

start_params1 <- rowMeans(pk_o3_F$mcmc$alphas[,burnin]) +
    ↪2*apply(pk_o3_F$mcmc$alphas[,burnin],1,sd)*
  rnorm(dim(pk_o3_F2$mcmc$alphas[,burnin])[1])

pk_o3_T = bayeshflm(Ymat_bc1, Xmat1 = Xmat1, Xmat2 = Xmat2,
    ↪std_time, std_time,
                    interceptInfo = list(Fk1="ts",K1 = 10),
                    tensorInfo = list(Fk = "bs", Psi = "bs", U =
                       ↪10, K = 10),
                    fpcaInfo = list(knots = 15, npc = 10),
                    method = "joint" , niter = niter,
                    smooth = T, alpha_start =
                       ↪as.matrix(start_params1))

pk_o3_Tind = bayeshflm(Ymat_bc1, Xmat1 = Xmat1,  Xmat2 = Xmat2,
    ↪taus = std_time, Ss = std_time,
                        interceptInfo = list(Fk1="ts",K1 = 10),
                        tensorInfo = list(Fk = "bs", Psi = "bs",
                           ↪U = 5, K = 5),
                        fpcaInfo = list(knots = 15, npc = 10),
                        method = "ind" , niter = niter,
                        smooth = TRUE, alpha_start =
                          ↪as.matrix(start_params1))

# B. PREDICTORS: Temperature and WS



pk_o3_F2 = bayeshflm(Ymat_bc1, Xmat1 = Xmat1, Xmat2 = Xmat3,
    ↪std_time, std_time,
                    interceptInfo = list(Fk1="ts",K1 = 10),
                    tensorInfo = list(Fk = "bs", Psi = "bs", U =
                       ↪10, K = 10),
                    fpcaInfo = list(knots = 15, npc = 10),
                    method = "ind" , niter = niter,
                    smooth = F)

start_params2 <- rowMeans(pk_o3_F2$mcmc$alphas[,burnin]) +
    ↪2*apply(pk_o3_F2$mcmc$alphas[,burnin],1,sd)*
  rnorm(dim(pk_o3_F2$mcmc$alphas[,burnin])[1])

pk_o3_T2 = bayeshflm(Ymat_bc1, Xmat1 = Xmat1, Xmat2 = Xmat3,
    ↪std_time, std_time,
                    interceptInfo = list(Fk1="ts",K1 = 10),
                    tensorInfo = list(Fk = "bs", Psi = "bs", U =
                       ↪10, K = 10),
                    fpcaInfo = list(knots = 15, npc = 10),
                    method = "joint" , niter = niter,
                    smooth = T, alpha_start =
                       ↪as.matrix(start_params2))

pk_o3_Tind2 = bayeshflm(Ymat_bc1, Xmat1 = Xmat1,  Xmat2 = Xmat3,
    ↪ taus = std_time, Ss = std_time,
                       interceptInfo = list(Fk1="ts",K1 = 10),
                       tensorInfo = list(Fk = "bs", Psi = "bs",
                          ↪U = 5, K = 5),
                       fpcaInfo = list(knots = 15, npc = 10),
                       method = "ind" , niter = niter,
                       smooth = T, alpha_start =
                          ↪as.matrix(start_params2))
```

```
# ------ Plot estimated surfaces

filled.contour(std_time,std_time, pk_o3_Tind$beta_hat$beta1,
               color.palette = viridis,xlab="v", ylab =
                  ↪bquote(tau))

# Indicator matrix of integration limits

lag <- which(c(create.lag.matrix(4, std_time, std_time, 24))==1)


# --- Build FARX(1) models

# 1) Define functional data objects
all_dat = list("Y"=t(Ymat_bc1), "X1" =t(Xmat1), "X2" =t(Xmat2),
   ↪"X3" =t(Xmat3))
fda_obg = as.fdata(all_dat)
plot(fda_obg)

# Model 1= X1, X2
far_pk1 <- far.cv(data = fda_obg, y = "Y", x = c("X1","X2"))
far_fit1 <- far(data = fda_obg, y = "Y", x = c("X1","X2"), kn =
   ↪far_pk1$minL2[1:3])
pred2 = predict(far_fit1, newdata = fda_obg)
t.pred2 = t(pred2[[1]])
mean(abs(t.pred2-Ymat_bc1[-1,]))

# Model 2 = X1, X3
far_pk2 <- far.cv(data = fda_obg, y = "Y", x = c("X1","X3"))
far_fit2 <- far(data = fda_obg, y = "Y", x = c("X1","X3"), kn =
   ↪far_pk2$minL2[1:3])
pred3 = predict(far_fit2, newdata = fda_obg)
t.pred3 = t(pred3[[1]])
mean((t.pred3-Ymat_bc1[-1,])^2)


# ----- Fit Hyndman-Ullah FAR(1) model

fts.y = data.frame(t(Ymat_bc1))

fts.obj = fts(x=1:24, y = t(rbind(Ymat_bc1))) # Create fts
   ↪object for ftsm function
fts.fit = ftsm(fts.obj, order =10)
```

```

# --- Plot some fitted curves

par(mfrow=c(3,2))
for(i in 2:100){

  plot(Ymat_bc1[i,], type = 'l', ylim = c(-3,3), col = 1, ylab =
      ↪"Ozone",
       xlab = "Time index", main = "Temp, RH")
  lines(pk_o3_F$Yfit[i,], col =6 )
  lines(pk_o3_T$Yfit[i,], col =7 )
  lines(t.pred2[i-1,], col = 5)
  legend("topleft", legend = c("Observed","Smooth = TRUE",
      ↪"Smooth = F", "FARX"),
         col = c(1,7,6,5), lty = 1, bty = "n")
}

# ---- Some convergence diagnostics plot

plot(pk_o3_F$mcmc$sigmab1[,burnin], type ='l')
plot(pk_o3_F$mcmc$sigmab2[,burnin], type ='l')
plot(pk_o3_F$mcmc$sigmamu[,burnin], type ='l')


# ----- Mean level

plot(pk_o3_F$mu_hat[1,], type = "l")


# ----- Performance on test set

# Finding missing obs in test set
# Scale obs. by training set mean and SD. (and lambda for Y)
miss_x_test = unique(c(which(apply(Temp_test, 1,
    ↪function(x){sum(is.na(x))})>0),
                       (which(apply(RH_test, 1,
                          ↪function(x){sum(is.na(x))})>0)),
                       (which(apply(WS_test, 1,
                          ↪function(x){sum(is.na(x))})>0))))

Ymat_test <-
    ↪matrix(box.cox((c(t(plattekloof_o3_wd_test[-miss_x_test,
    ↪]))), lamfit), byrow = T, ncol =24)
```

```r
Ymat_test1 <- matrix(apply(Ymat_test, 1,
    ↪function(x){(x-Y.mean)/Y.sd} ), byrow = T, ncol = 24)
dim(Ymat_test)
Xmat1_test <- matrix(apply(Temp_test[-miss_x_test, ] , 1,
    ↪function(x){(x-Temp.mean)/Temp.sd} ), byrow = T, ncol = 24)
Xmat2_test <- matrix(apply(RH_test[-miss_x_test, ], 1,
    ↪function(x){(x-RH.mean)/RH.sd} ), byrow = T, ncol = 24)
Xmat3_test <- matrix(apply(WS_test[-miss_x_test, ], 1,
    ↪function(x){(x-WS.mean)/WS.sd} ), byrow = T, ncol = 24)

# ----- Generate predictions


mypreds1 = predict_hflm(pk_o3_F, Xmat1_test, Xmat2_test,
    ↪std_time, -0.01, burnin=(0.5*niter):niter)
mypreds2 = predict_hflm_joint(pk_o3_T, Xmat1_test, Xmat2_test,
    ↪std_time, -0.01, burnin=(0.5*niter):niter)
mypreds2b = predict_hflm(pk_o3_Tind, Xmat1_test, Xmat2_test,
    ↪std_time, -0.01, burnin=(0.5*niter):niter)

mypreds3 = predict_hflm(pk_o3_F2, Xmat1_test, Xmat3_test,
    ↪std_time, -0.01, burnin=burnin)
mypreds4 = predict_hflm(temp, Xmat1_test, Xmat3_test, std_time,
    ↪-0.01, burnin=burnin)
mypreds4b = predict_hflm(pk_o3_Tind2, Xmat1_test, Xmat3_test,
    ↪std_time, -0.01, burnin=burnin)


Yhat1 = matrix(mypreds1[,1], ncol = ncol(Xmat1_test), byrow
    ↪=TRUE)[-1,]
Yhat2 = matrix(mypreds2$values[,1], ncol = ncol(Xmat1_test),
    ↪byrow =TRUE)[-1,]
Yhat2b = matrix(mypreds2b[,1], ncol = ncol(Xmat1_test), byrow
    ↪=TRUE)[-1,]

Yhat3 = matrix(mypreds3[,1], ncol = ncol(Xmat1_test), byrow
    ↪=TRUE)[-1,]
Yhat4 = matrix(mypreds4[,1], ncol = ncol(Xmat1_test), byrow
    ↪=TRUE)[-1,]
Yhat4b = matrix(mypreds4b[,1], ncol = ncol(Xmat1_test), byrow
    ↪=TRUE)[-1,]


```

```
# 2a) for FARX(1) model

fda_test <- as.fdata(list("Y" = t(Ymat_test1), "X1" =
    ↪t(Xmat1_test), "X2" =t(Xmat2_test), "X3" = t(Xmat3_test)))

# Function to iteratively update forecasts

farx.forecast <- function(farmod, H, kn){
  Xmat1_FAR <- matrix(NA, nrow = 24, ncol = H +1)
  Xmat2_FAR <- matrix(NA, nrow = 24, ncol = H+1)
  Xmat3_FAR <- matrix(NA, nrow = 24, ncol = H+1)
  Ymat_FAR <- matrix(NA, nrow = 24, ncol = H+1)

  Ymat_FAR[,1]=t(Ymat_test1)[,1]
  Xmat1_FAR[,1]=t(Xmat1_test)[,1]
  Xmat2_FAR[,1]=t(Xmat2_test)[,1]
  Xmat3_FAR[,1]=t(Xmat3_test)[,1]

  for(i in 2:(H+1)){
    Ymat_FAR[,i]=t(Ymat_test1)[,i]
    Xmat1_FAR[,i]=t(Xmat1_test)[,i]
    Xmat2_FAR[,i]=t(Xmat2_test)[,i]
    Xmat3_FAR[,i]=t(Xmat3_test)[,i]

    fda_test_1 <- as.fdata(list("Y" = Ymat_FAR[,1:i], "X1" =
       ↪Xmat1_FAR[,1:i], "X2" =Xmat2_FAR[,1:i], "X3"
       ↪=Xmat3_FAR[,1:i]))
    pred_one = predict(farmod, newdata = fda_test_1, kn = kn)
    Ymat_FAR[,i] <- pred_one[[1]][,i-1]

  }

  return(Ymat_FAR[,-1])
}

FARX_10_Temp_RH <- farx.forecast(far_fit1, 10,far_pk1$minL2[1:3])
FARX_30_Temp_RH <- farx.forecast(far_fit1, 30,far_pk1$minL2[1:3])
FARX_110_Temp_RH <- farx.forecast(far_fit1,
    ↪110,far_pk1$minL2[1:3])

FARX_10_Temp_WS <- farx.forecast(far_fit2, 10,
    ↪far_pk2$minL2[1:3])
FARX_30_Temp_WS <- farx.forecast(far_fit2, 30,
    ↪far_pk2$minL2[1:3])
FARX_110_Temp_WS <- farx.forecast(far_fit2, 110,
    ↪far_pk2$minL2[1:3])


# 2b) For Hyndman-Ullah FAR(1)

fts.pred_10  = forecast.ftsm(fts.fit, h = 11, level = 95)
fts.vals_10 = cbind(c(fts.pred_10$mean$y),
    ↪c(fts.pred_10$lower$y),c(fts.pred_10$upper$y))

fts.pred_30  = forecast.ftsm(fts.fit, h = 31, level = 95)
fts.vals_30 = cbind(c(fts.pred_30$mean$y),
    ↪c(fts.pred_30$lower$y),c(fts.pred_30$upper$y))

fts.pred_110  = forecast.ftsm(fts.fit, h = 111, level = 95)
fts.vals_110= cbind(c(fts.pred_110$mean$y),
    ↪c(fts.pred_110$lower$y),c(fts.pred_110$upper$y))


# 2c) Persistence model

pred_int1 = pred.persist(select.fdata(fda_test, date =
    ↪1:111))[[1]]



## -- Functions to evaluate errors

imse <- function(yhat, yobs, H){
  mean((yhat[1:H,] -yobs[1:H,])^2)
}

imae <- function(yhat, yobs, H){
  mean(abs(yhat[1:H,] -yobs[1:H,] ))
}

# Remember that the first obs. doesn't count for FARX so remove
    ↪for analysis
Ytest_comp <- Ymat_test1[-1, ]

# HFLMS with Temp and RH

```

```
imse(Yhat1, Ytest_comp, 10)
imse(Yhat2, Ytest_comp, 10)
imse(Yhat2b, Ytest_comp, 10)
imse(t(FARX_10_Temp_RH), Ytest_comp, 10)
imse(t(fts.pred_10$mean$y[,-1]), Ytest_comp, 10)
imse(t(pred_int1), Ytest_comp, 110)


imse(Yhat1, Ytest_comp, 30)
imse(Yhat2, Ytest_comp, 30)
imse(Yhat2b, Ytest_comp, 30)
imse(t(FARX_30_Temp_RH), Ytest_comp, 30)
imse(t(fts.pred_30$mean$y[,-1]), Ytest_comp, 30)


imse(Yhat1, Ytest_comp, 110)
imse(Yhat2, Ytest_comp, 110)
imse(Yhat2b, Ytest_comp, 110)
imse(t(FARX_110_Temp_RH), Ytest_comp, 110)
imse(t(fts.pred_110$mean$y[,-1]), Ytest_comp, 110)

imae(Yhat1, Ytest_comp, 10)
imae(Yhat2, Ytest_comp, 10)
imae(Yhat2b, Ytest_comp, 10)
imae(t(FARX_10_Temp_RH), Ytest_comp, 10)
imae(t(fts.pred_10$mean$y[,-1]), Ytest_comp, 10)
imae(t(pred_int1), Ytest_comp, 110)

imae(Yhat1, Ytest_comp, 30)
imae(Yhat2, Ytest_comp, 30)
imae(Yhat2b, Ytest_comp, 30)
imae(t(FARX_30_Temp_RH), Ytest_comp, 30)
imae(t(fts.pred_30$mean$y[,-1]), Ytest_comp, 30)


imae(Yhat1, Ytest_comp, 110)
imae(Yhat2, Ytest_comp, 110)
imae(Yhat2b, Ytest_comp, 110)
imae(t(FARX_110_Temp_RH), Ytest_comp, 110)
imae(t(fts.pred_110$mean$y[,-1]), Ytest_comp, 110)


imse(Yhat3, Ytest_comp, 10)
imse(Yhat4, Ytest_comp, 10)
imse(Yhat4b, Ytest_comp, 10)
imse(t(FARX_10_Temp_WS), Ytest_comp, 10)
imse(t(fts.pred_10$mean$y[,-1]), Ytest_comp, 10)
imse(t(pred_int1), Ytest_comp, 110)


imse(Yhat3, Ytest_comp, 30)
imse(Yhat4, Ytest_comp, 30)
imse(Yhat4b, Ytest_comp, 30)
imse(t(FARX_30_Temp_WS), Ytest_comp, 30)
imse(t(fts.pred_30$mean$y[,-1]), Ytest_comp, 30)

imse(Yhat3, Ytest_comp, 110)
imse(Yhat4, Ytest_comp, 110)
imse(Yhat4b, Ytest_comp, 110)
imse(t(FARX_110_Temp_WS), Ytest_comp, 110)
imse(t(fts.pred_110$mean$y[,-1]), Ytest_comp, 110)



imae(Yhat3, Ytest_comp, 10)
imae(Yhat4, Ytest_comp, 10)
imae(Yhat4b, Ytest_comp, 10)
imae(t(FARX_10_Temp_WS), Ytest_comp, 10)
imae(t(fts.pred_10$mean$y[,-1]), Ytest_comp, 10)
imae(t(pred_int1), Ytest_comp, 110)


imae(Yhat3, Ytest_comp, 30)
imae(Yhat4, Ytest_comp, 30)
imae(Yhat4b, Ytest_comp, 30)
imae(t(FARX_30_Temp_WS), Ytest_comp, 30)
imae(t(fts.pred_30$mean$y[,-1]), Ytest_comp, 30)

imae(Yhat3, Ytest_comp, 110)
imae(Yhat4, Ytest_comp, 110)
imae(Yhat4b, Ytest_comp, 110)
imae(t(FARX_110_Temp_WS), Ytest_comp, 110)
imae(t(fts.pred_110$mean$y[,-1]), Ytest_comp, 110)


# Function to evaluate coverage
```

```

coverage <- function(true, lower, upper){

  cc <- NA
  if(true >= lower && true <= upper){
    cc <- 1
  }
  else{cc <- 0}

}

mean(apply(cbind(c(t(Ymat_test1[-1,][1:H,])),
    ↪mypreds1[-(1:24),][1:(H*24),2:3]), 1, function(x){
  coverage(x[1], x[2], x[3])
}))


# ---- Plot forecasts using ggplot and pred interval using ggplot

Yhat_all = data.frame(cbind(id = rep(1:110, each = 24), time =
    ↪rep(std_time, 110),
                            c(t(Ymat_test1[-1,][1:110,])),
                            c(t(Yhat3[1:110, ])),
                               ↪c(t(Yhat4[1:110, ])),
                               ↪c(t(Yhat4b[1:110, ])),
                            c(FARX_110_Temp_WS),
                               ↪c(fts.pred_110$mean$y[,-1])))

pred_interval <- data.frame(cbind(mypreds3[25:(111*24),2:3],
    ↪mypreds4b[25:(111*24),2:3], fts.vals_110[-(1:24),2:3]))
colnames(pred_interval) <- c("lwrF","uprF",
    ↪"lwrT","uprT","lwrHU","uprHU")


my.cols <- c("black","blue","darkorange","green4")#
    ↪heat.colors(3, alpha=1)
my.names <- c("True", "Two-stage HFLM", "FARX(1)",
    ↪"Hyndman-Ullah FAR(1)")
names(my.cols) <- my.names

ggplot(data = cbind(Yhat_all,pred_interval) %>% filter(id %in%
    ↪c(1:15)), aes(x=time, y =V3, group = id)) +
  geom_line(linetype="dashed") + # True]
  geom_line(aes(y=V6, col = "Two-stage HFLM") )+  # TRUE IND
  geom_line(aes(y=V7, col = "FARX(1)")) + #FARX
  geom_line(aes(y=V8, col = "Hyndman-Ullah FAR(1)") )+ #HU FAR
  facet_wrap(~id, ncol = 5)+
  geom_ribbon( aes(ymin = lwrHU, ymax = uprHU, fill =
     ↪"Hyndman-Ullah FAR(1)"), alpha = 0.2)+
  geom_ribbon( aes(ymin = lwrT, ymax = uprT, fill = "Two-stage
     ↪HFLM"), alpha = 0.2) +
  scale_colour_manual("Forecasts", breaks =
     ↪as.character(my.names),  values = my.cols)+
  scale_fill_manual("95% Prediction Interval", breaks =
     ↪as.character(my.names),  values = my.cols)+
  theme(strip.background = element_blank(),
        legend.position = "top",
        legend.title = element_text(size=9, face ="bold"))+
  labs(x="Time", y = "Ozone levels")


#----- Analysing marginal effects of windspeed using two-stage
    ↪HFLM model B

par(mfrow=c(1,3))

# ---- Find fitted values of X

Xind3 <- matrix(apply(pk_o3_Tind2$mcmc$x2[,burnin],1,mean))
X_Profile = apply(matrix(Xind3,ncol = 24, byrow = T),2,mean)
X_Profile_low = rep(0.5,24)

Y_marg <- matrix(X_Profile,nrow
    ↪=1)%*%(pk_o3_Tind2$beta_hat$beta2)*diff(std_time)[1]
Y_marg_low <- matrix(X_Profile_low,nrow
    ↪=1)%*%(pk_o3_Tind2$beta_hat$beta2)*diff(std_time)[1]

df <- data.frame(time= std_time , Y_marg = Y_marg,
                 Xprof = X_Profile,
                 mu = pk_o3_Tind2$mu_hat[1,],
                 Xlow = X_Profile_low,
                 Yup = c(Y_marg_up))

ggplot(df, aes(x=time, y= Y_marg)) + geom_line(col = "blue") +
  geom_line(aes(y=mu), linetype = "dashed") +
     ↪geom_line(aes(y=Ylow),col = "green4")+
  labs(y="Ozone level", x ="Time")+ theme(axis.text =
     ↪element_text(size = 18), axis.title = element_text(size =
     ↪18))

ggplot(df, aes(x=time, y= Xprof)) + geom_line(col = "blue") +
  geom_line(aes(y=Xlow), col = "green4") +labs(x="Time", y =
     ↪"Wind speed")+ theme(axis.text = element_text(size = 18),
     ↪axis.title = element_text(size = 18))



# Residual diagnostics

# -------- Find FPACF

par(mfrow=c(2,2))
fdaACF::FTS_identification(Ymat_bc1 - pk_o3_Tind2$Yfit, v =
    ↪std_time, nlags = 10, figure = TRUE )


ggplot(data=FPCA_df, aes(x=as.factor(lag), y=rho)) +
  geom_hline(yintercept = FPACF$Blueline, lwd = 1, linetype =
     ↪"dashed", col = "darkblue")+
  geom_errorbar(aes(x=as.factor(lag), ymax=rho, ymin=0),
     ↪width=0)+
  labs(y = bquote(rho), x = "Lag")

# Calculate residuals

err =  data.frame(e = c(Ymat_bc1 - pk_o3_Tind2$Yfit))

# Plot histogram of residuals

ggplot(data=ydat, aes(x=yhat)) +
    ↪geom_histogram(aes(y=..density..), fill = "darkblue", alpha
    ↪= .3) +
  geom_vline(xintercept = 0, linetype = "dashed")+
  stat_function(fun = dnorm, args = list(mean = mean(Ymat_bc1),
     ↪sd = sd(Ymat_bc1)), lwd = 0.7 )+
  labs(y="Density", x="Ozone level")

# QQ plots of residuals

```

```
ggplot(err,aes(sample = e)) +stat_qq() + stat_qq_line() +
  labs(y="Sample",x="Theoretical")
```