Modeling and learning with mixtures of experts for functional data and distributed data

Nhat Thien Pham

Normandie Université

THÈSE

Pour obtenir le diplôme de doctorat

Spécialité MATHEMATIQUES

Préparée au sein de l'Université de Caen Normandie

Modeling and Learning with Mixtures of Experts for Functional Data and Distributed Data

Présentée et soutenue par
NHAT THIEN PHAM

Thèse soutenue le 22/11/2022 devant le jury composé de		
MME AGATHE GUILLOUX	Directeur de recherche, INRIA Paris	Rapporteur du jury
M. VINCENT VANDEWALLE	Professeur des universités, Université côte d'Azur	Rapporteur du jury
M. VAN HA HOANG	Maître de conférences, VIETNAM NATIONAL UNIVERSITY - VNU	Membre du jury
M. NICOLAS WICKER	Professeur des universités, Université de Lille	Président du jury

Thèse dirigée par FAICEL CHAMROUKHI (Laboratoire de Mathématiques 'Nicolas Oresme' (Caen))

UNIVERSITÉ CAEN NORMANDIE

Contents

Abstract

Mixture of experts (ME) models are popular in statistics and machine learning and have been studied for high-dimensional vectorial data and that are centralized. However, in many problems, we observe time series data and applying the existing ME models directly to these raw data may limit the performance because they ignore the correlation between variables, an intrinsic nature of functional data, and therefore do not adequately capture the inherent functional structure of the data. In many applications the data may also be not available at a centralized mode, and there is therefore a need to develop adapted distributed strategies allowing for efficient parallel computations. From the modeling point of view, there is indeed a lack of ME models for functional data. From the estimation point of view, there is a lack of approaches to estimate ME models in a distributed fashion. This work addresses both of these problems. This thesis studies mixture of experts (ME) models in functional data and large-scale problems, in a heterogeneous scenario. The main objectives are *i)* to deal with situations in which we are given functional predictors (e.g., time series) to predict a potentially functional response, and *ii)* to learn from distributed data. We first propose in this thesis a new family of ME models, called functional mixture of experts (FME), which includes scalar-on-function FME, function-on-scalar FME, and function-on-function FME. We introduce regularized model estimation approaches via appropriate regularizations that encourage sparse and smooth estimates, while being interpretable. We develop efficient EM algorithms to maximize the corresponding observed-data log-likelihood functions and conduct extensive experimental studies to highlight the performance of the proposed models and algorithms. Then, to scale-up the ME estimation to a potentially distributed data, we develop a distributed learning strategy of ME models. It performs standard inference on local machines to obtain local estimators, then transmits them to a central machine where they are aggregated. Based on minimizing a proposed expected transportation divergence, the local estimators are aggregated to obtain a reduced estimator that is consistent with the global one, *i.e.*, the estimator that could be possibly constructed upon the full dataset. Experimental studies demonstrate the performance of our approach.
Mixture of experts (ME) models are popular in statistics and machine learning and have been studied for high-dimensional vectorial data and that are centralized. However, in many problems, we observe time series data and applying the existing ME models directly to these raw data may limit the performance because they ignore the correlation between variables, an intrinsic nature of functional data, and therefore do not adequately capture the inherent functional structure of the data. In many applications the data may also be not available at a centralized mode, and there is therefore a need to develop adapted distributed strategies allowing for efficient parallel computations. This thesis studies mixture of experts (ME) models in functional data and large-scale problems, in a heterogeneous scenario. The main objectives are *i)* to deal with situations

in which we are given functional predictors (e.g., time series) to predict a potentially functional response, and *ii)* to learn from distributed data. We first propose in this thesis a new family of ME models, called functional mixture of experts (FME), which includes scalar-on-function FME, function-on-scalar FME, and function-on-function FME. We introduce regularized model estimation approaches via appropriate regularizations that encourage sparse and smooth estimates, while being interpretable. We develop efficient EM algorithms to maximize the corresponding observed-data log-likelihood functions and conduct extensive experimental studies to highlight the performance of the proposed models and algorithms. Then, to scale-up the ME estimation to a potentially distributed data, we develop a distributed learning strategy of ME models. It performs standard inference on local machines to obtain local estimators, then transmits them to a central machine where they are aggregated. Based on minimizing a proposed expected transportation divergence, the local estimators are aggregated to obtain a reduced estimator that is consistent with the global one, *i.e.*, the estimator that could be possibly constructed upon the full dataset. Experimental studies demonstrate the performance of our approach.
Keywords: Mixture of Experts; Functional Data Analysis; Regularized Estimation; LASSO; EM algorithm; MM Algorithm; Distributed Learning; Optimal Transport; Prediction; Clustering.

Keywords: Mixture of Experts, Functional Data Analysis, Regularized Estimation, LASSO, EM algorithm, MM Algorithm, Dantzig Selector, Distributed Learning, Optimal Transport Divergence, Clustering, Classification, Regression.

Acknowledgements

This thesis will never be completed without the support and encouragement from many people to whom I would like to express my deep gratitude. First and foremost, I wish to express the deepest gratitude and appreciation to my supervisor, Professor Faïcel Chamroukhi, who had brought me a great opportunity and constantly supported me to pursue my study. Faïcel always encouraged and gave me the best guidance to keep going through all the challenges from the start to the end of this thesis. Faïcel's explanations are also very clear, I am thankful and feel fortunate to be a student of such wonderful teacher. Without Faïcel's patience and kindness, this work could not have been accomplished.

I would wish to express my gratitude to Professor Agathe Guilloux and Professor Vincent Vandewalle for having accepted to be reviewers of my thesis. I would also wish to express my gratitude to Professor Nicolas Wicker and Doctor Van Ha Hoang for having accepted to be members of the jury. I am very grateful that they took the time to read the thesis and have given the insightful comments.

I am also grateful to thank my friendly colleagues and staffs in Laboratory of Mathematics Nicolas Oresme (LMNO) for all their helps and supports. Especially, I would wish to thank Professor Éric Ricard, Professor Bruno Zanuttini, Madame Anita Foro and Madame Marie Legay for their very kind helps during the time I am in LMNO. I wish also to express my gratitude to the members of my follow-up committee, Professor Mustapha Lebbah and Professor Francesco Amoroso, for helpful suggestions for the progression of my PhD.

I would take this opportunity to thank to all of my professors in University of Quy Nhon and University of Science who gave me many motivations and impacts in the beginning of my academic career. I would also wish to take this opportunity to thank professors Filippo Santambrogio, who kindly helped me to pursue the study in France, François James, my advisor in the PUF master program, and Leo Liberti, my advisor in the Paris-Saclay master program.

Finally, I would love to say thanks to all friends who share many memories with me in France, especially, anh-chi Loc-Phuong, Dung-Trang, Náoh Êl-Huyen, Viet-Y, Thai-Hoa, Thi-Trang, Hung-Trang, Quang-Tram, Son-Hau, Long-Han, Hieu-Thuy, anh Tuyen, Thanh, Nhat, Ban, chi Hong, Huyen, Thu, Ngoc, em Thao Vit, Dat, Hung, Nam, Tin, Thanh, Son, Nhuan. My special thanks are due to my family and my fiancée Thanh Loan.

Caen, November 2022
Pham Nhat Thien

List of Notations

Abbreviations

BIC	Bayesian Information Criterion
ClusErr	Clustering Error
Corr	Correlation
EM	Expectation-Maximization
FDA	Functional Data Analysis
FME	Functional Mixture of Experts
FPCA	Functional Principle Component Analysis
FF-FME	Function-on-Function Functional Mixture of Experts
FS-FME	Function-on-Scalar Functional Mixture of Experts
HME	Hierarchical mixture-of-experts
iFME	Interpretable Functional Mixture of Experts
i.i.d.	Independent and Identically Distributed
KL	Kullback-Leibler
LASSO	Least Absolute Shrinkage and Selection Operator
LS	Least Square
ME	Mixture-of-experts
MISE	Mean Integrated Squared Error
MLE	Maximum Likelihood Estimation
MSE	Mean Squared Error
MM	Minorization–Maximization
PCA	Principle Component Analysis
PDF	Probability Density Function
RI	Rand Index
RPE	Relative Prediction Error
SSE	Sum of Squared Srrors

Notations

$\mathbb{1}_p$	Column vector of length p contains 1's
$k \in [K]$	$k \in \{1, \ldots, K\}$

Operators

$\otimes$	Kronecker product
$\odot$	Hadamard product
$\top$	Vector transpose
$\mathbb{E}$	Expectation
$\mathbb{P}$	Probability
Var	Variance
vec$(\cdot)$	Vectorize a matrix
vech$(\cdot)$	Extract unique elements of a symmetric matrix
$\boldsymbol{I}_{\mathcal{A}}$	Indicator, *i.e.*, equals 1 if $\mathcal{A}$ and equals 0 otherwise

Chapter 1

Introduction

Contents

1.1 Context of study

Nowadays, learning from data is becoming more and more popular. For example, in medicine statistical models have been used to classify patients and detect tumors based on magnetic resonance imaging. In business, sophisticated machine learning algorithms have been used to make business decisions in seconds. In image processing, denoising is no longer done by filters as in classical techniques, but can be automated by machine learning algorithms. And many other applications in various fields such as speech recognition, natural language processing, product recommendation, *etc.* The above applications all have one thing in common: they are all enabled by learning from data. Moreover, in this big data age, data is cheaper and easier to obtain, which requires more efficient methods to deal with data in large-scale. Therefore, statistical models that are efficient and can be learned in parallel fashion are of great interest.

One of the common challenges of modern statistical models is the increasing occurrence of high-dimensional data and, more recently, functional data. High-dimensional data have been considered for many models. The common approach is to impose a penalization, usually the ℓ_1-norm of the parameter vector, when performing model estimation. However, for functional data, classical multivariate models are not appropriate because they ignore the underlying intrinsic nature and structure of the data. Functional data analysis (FDA) (Ramsay and Silverman, 2005; Ferraty and Vieu, 2006) in which the individual data units are assumed to be functions, rather than vectors, offers an adapted framework for dealing with continuously observed data. FDA has been shown to be efficient and flexible in many applications, including regression, classification, and clustering.

For the analysis of heterogeneous data, mixture models (Titterington et al., 1985; McLachlan and Peel., 2000) have become a standard and have shown numerous applications in various fields

including signal processing, medicine, bioinformatics, among many others. Mixtures-of-experts (ME), introduced by (Jacobs et al., 1991), as an extension of mixture models, is a successful and flexible supervised learning architecture for efficiently representing complex nonlinear relationships in observed pairs of heterogeneous data. Since its first introduction 30 years ago, it is still one of the models researchers are most interested in, and has been used in numerous regression, clustering, as well as classification applications in finance, recognition, healthcare, surveillance, *etc.* Some successful applications of ME architecture may be mentioned here include ME for time series prediction (Zeevi et al., 1996; Yümlü et al., 2003), segmentation (Chamroukhi et al., 2013, 2009), ME for social network data (Gormley and Murphy, 2010), for classification of gender and pose of human faces (Gutta et al., 2000), among many others.

Given the success of ME in various domains and the increasing appearance of functional data in real-world applications, the first direction of this thesis is then to develop ME models for functional data. Attention will also be given to the problem regularization and interpretation in models, which is less common than the well-known problem of variable-selection in multivariate models.

However, there is another challenge that limits the application of the ME model. That is the ability to learn from data that are not stored on a single machine. This can be due to the nature of the data, communication issues (such as data in meteorology where they are collected at different stations), or privacy restriction (such as data in finance and medicine). Even when a complete data set is available, it often requires a huge processing resource for handling and running statistical inference methods, this may be evenly more expensive. Such challenges impose new analysis strategies to the modern statistical models. The common approach is use the divide-and-conquer to parallelize the model estimation. There have been many successful attempts in parallelizing the existing learning algorithms and statistical methods. These include the parallelization of stochastic gradient descent(Zinkevich et al., 2010), parallelization of multiple linear regression(Mingxian et al., 1991), of logistic regression (Shofiyah and Sofro, 2018), for penalized regressions (Chen and ge Xie, 2014) parallel K-Means clustering based on MapReduce (Zhao et al., 2009), distributed learning for heterogeneous data via modelintegration (Merugu and Ghosh, 2005), more recently, distributed learning for finite Gaussian mixtures (Zhang and Chen, 2021), among others. Although distributed versions of the EM algorithm have been developed (Nowak, 2003; chen Chen et al., 2013), these approaches have not been considered for ME models, and require the communication of summary statistics and the coordination across local machines at each iteration, leading to high communication cost. So ME models, however, is still outside of this trend. Therefore, the second direction of this thesis is to develop an efficient strategy to estimate ME models in a distributed way.

1.2 Contributions and outline of the thesis

The manuscript is organized as follows. Chapter 2 is dedicated to the state of the art. Chapter 3 presents our first contribution to the modeling of mixtures-of-experts for data with functional predictor and continuous scalar response. Chapter 4 is the second contribution which extends the proposed models to the case of classification and to the cases of functional response. There are three new models proposed in this chapter. Chapter 5 presents our last contribution, in which a distributed learning approach for mixture-of-experts models is proposed. Finally, Chapter 6 is for conclusions and discussions of future work. Technical details related to the mathematical developments of our contributions are provided in Appendices A, B and C.

More particularly, in Chapter 2, we attempt to give a comprehensive overview of the current state of the art of models and estimation methods on the research topics of this thesis. First, we present an overview of the mixture-of-experts models, focusing on the modeling of the expert and gating networks, as well as the inference of the models using maximum likelihood estimation. We then introduce some important concepts of functional data analysis, including the mathematical framework, basis expansions, functional principal component analysis, *etc.* We also provide a review for the models for regression, clustering, as well as classification of functional data. We opt to present in details the estimation methods for the function-on-function and function-on-scalar regression models since they will be used later in our proposed model as extensions.

In Chapter 3, we propose a new family of mixtures-of-experts models, called functional mixtures of experts (FME), to relate a functional predictor to a scalar response. A dedicated EM algorithm for the maximum likelihood parameter estimation is developed for FME model. To deal with potential high-dimensional setting of the proposed FME model, we also consider a Lasso-regularized approach, which consists of a penalized MLE estimation via a hybrid EM-Lasso algorithm. Finally, to obtain a sparse and highly-interpretable regularization of the functional expert and gating parameters, interpretable FME (iFME) model is proposed, which is constructed upon the coefficients of the derivatives of the functional parameters. The resulting model is fitted by regularized MLE via an dedicated EM-iFME algorithm. Extensive experimental studies are then constructed to compare the proposed approaches to the main competitive state of art methods for the subject. The model selection via modified BIC and the implementation details are also discussed. On simulated data, our models outperforms its competitive in both prediction and clustering tasks. The models are then applied to two real-world datasets, on which they also outperforms its competitive on prediction. Specially, the iFME model provides very promising results as it not only produces highly interpretable fits but also high prediction performance. This chapter has leads to the a submission to Journal of Statistics and Computing (Chamroukhi et al., 2022) (under revision).

Then, in Chapter 4, we extend the FME model proposed in Chapter 3 to classification problem, and to the case of functional responses. In particular, in the FME model for classification, the experts will be modeled by the functional multinomial regression model. In the same spirit with Chapter 3, an interpretable model will also be developed.

To deal with functional responses, we propose in Chapter 4 two ME models: function-on-scalar

FME (FS-FME), and function-on-function FME (FF-FME). These models are established based on the framework proposed in previous chapter, adapted for functional responses. Specially, FS-FME and FF-FME are estimated with a roughness penalization on the functional parameters. This results in smoother fits as shown in the numerical experiments on both simulated and real-world data.

In Chapter 5, relying on the divide and conquer principle, we propose a distributed learning approach for ME models. In particular, the proposed approach consists of two steps: a local inference step, where MLE is performed on subsets of data available at local machines, and a aggregation step where the local estimators are aggregated. We focus on developing the aggregation strategy based on minimizing an expected transportation divergence between two ME models. The MM algorithm is used to solve the resulting optimization problem. Numerical experiments on simulated and real-world data are constructed to illustrate the efficiency of the proposed approach. This chapter leads to a paper to be submitted to a journal.

Chapter 2

State of the art

Contents

2.1 Introduction

The goal of this chapter is to provide an overview of the topics of mixture of experts (ME) models and functional data analysis (FDA). The following sections are selective and carefully arranged to introduce the main related concepts of the research topics in this thesis. First, in Section 2.2, the general notions, the modelings and approximation capacity of ME models are presented to provide a first introduction to the research topic. Then, in Section 2.3, we present the estimation of ME models via Expectation-Maximization (EM) algorithm (Dempster et al., 1977), and review some applications of ME models.

In Section 2.4, we present the most basis and important tools of FDA, including the mathematical framework, the basis expansion, data pre-processing methods, and notably the regression models concerning functional data which we will use oftenly through the thesis. In this section, we also review the clustering and classification models for functional data with some illustrative examples. Finally, in preparation for the study in Chapter 5, we present in Section 2.9 some preliminaries on divergence and distance on the space of probability measures.

2.2 Mixture of experts models

In this section, we give a general introduction to a class of probabilistic models known as Mixtures-of-experts (ME). By the end of this section, the reader will have an overview of this research topic, and will be familiarized with the notations regarding ME models that we will use throughout this thesis.

2.2.1 General notions of ME models

Introduced by Jacobs et al. (1991), ME is a successful and flexible supervised learning architecture for efficiently representing complex nonlinear relationships in observed pairs of heterogeneous data. The ME model is based on the principle of divide and conquer, so that the response y is obtained from the soft-association of multiple expert responses, each targeting a homogeneous sub-population of the heterogeneous population, given the input covariates (predictors or features). From the statistical modeling point of view, a ME model is an extension of the finite mixture model (McLachlan and Peel., 2000) which explores the unconditional (mixture) distribution of a given set of features.

Let x and y be the generic notations for the input and output, respectively, we wish to relate together via ME models. In the most general notion, a ME model can be written as

$$\mathrm{ME}(y|x) = \sum_{k=1}^{K} \mathrm{Gate}_k(x)\, \mathrm{Expert}_k(y|x), \tag{2.1}$$

in which $\text{ME}(y|x)$, the distribution of output y given the input x, is modeled as a mixture distribution with input-dependent mixing proportions $\text{Gate}_k(x)$ and conditional mixture components $\text{Expert}_k(y|x)$. ME is therefore a fully conditional mixture model that allows the mixing proportions to be functions of the input. In ME model terminology, $\text{Gate}_k(x)$ is referred to as gating function, and $\text{Expert}_k(y|x)$ is referred to as expert function, while K is the number of experts.

The general notion (2.1) will be referred many times throughout this thesis as a starting point for developing any specific ME model. Note that the generic variables x and y in (2.1), also commonly referred to as covariate (or predictor) and response, respectively, can be of any data type (such as continuous scalars, vectors, binary, categorical, functions, multivariate functions, *etc*) and of arbitrary dimension, leading to many different settings and issues for studies and applications regarding ME models. At the end of this chapter, we will try to provide a list of ME models that have been studied so far, as well as point out the open important questions in this research area.

In the notion of probability theory, a ME model can be explained as follows. Let $(X, Y) \in \mathcal{X} \times \mathcal{Y}$ be a pair random variables which follows some probability model, where $X \in \mathcal{X}$ is the input, and $Y \in \mathcal{Y}$ is the output. The $\mathcal{X} \times \mathcal{Y}$ is a subset of some probability space, *e.g.*, $\mathbb{R}^p \times \mathbb{R}^q$ for some $p, q \in \mathbb{N}^*$. Then, a ME model can be written as the following decomposition of the output variable Y and the input variable X

$$\mathbb{P}(Y|X) = \sum_Z \mathbb{P}(Z|X)\mathbb{P}(Y|X, Z), \tag{2.2}$$

in which, Z is a hidden variable that represents the missing data, takes values in the finite set $\{1, \ldots, K\}$, and indicates which expert was responsible for generating the data point.

The decomposition (2.2) suggests that the data generating process of a ME model is assumed to be as follows: there is a hidden process that specifies the state of Z given X, then the output Y is generated conditional on the values of Z and X. From this decomposition point of view, the functions $\text{Gate}_k(x)$ and $\text{Expert}_k(y|x)$ in (2.1) are thus the ways we model the probabilities $\mathbb{P}(Z|X)$ and $\mathbb{P}(Y|X, Z)$, *i.e.*, how the mixing proportions vary and how the outputs were generated. Because of that, any modeling for the gating function must satisfy the constraint $\sum_{k=1}^{K} \text{Gate}_k(x) = 1$. Thus, depending on the specific parametric forms of the gating function $\text{Gate}_k(x)$ and the expert functions $\text{Expert}_k(y|x)$, we have different specifications for ME models. From the application point of view, the main task for us is to select suitable models for the functions $\text{Gate}_k(x)$ and $\text{Expert}_k(y|x)$, which depends on the problem and the nature of the data, and to develop efficient algorithms for parameter estimations. From theoretical point of view, researchers focus on studying the approximation capacities, convergence rates of the models, as well as addressing many issues regarding model complexity.

Note that while it is possible to have K different forms for the K expert functions $\text{Expert}_k(y|x)$, they are usually modeled by a same parametric family, *e.g.*, Gaussian, t-Student, multinomial distribution, *etc*. Thus, if K is assumed to be known, estimating a ME model consists of estimating the parameters of the gating and expert functions. Figure 2.1 describes the schematic diagram of a K-component ME model.

ME models thus allow one to better capture more complex relationships between y and x in

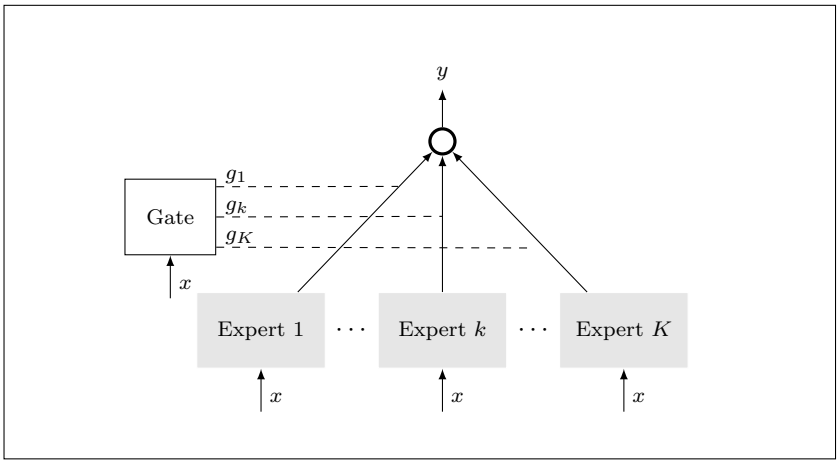


Figure 2.1: Visualization of the architecture of a ME model. Here, the gating network and each of K expert receive x as input, then outputs of the experts are weighted by outputs of the gating network, *i.e.*, $g_1, \ldots, g_K$, to produce the final output y.

heterogeneous situations in nonlinear regression, classification, and clustering the data by associating each expert component to a cluster. In situations where the standard mixture of regression models fails to approximate well the conditional probability distribution of data, ME model may be a good substitute because it can capture a wide variety of data distributions thanks to their universal approximation property well established for almost of popular gating and expert functions. The approximation capacity of ME will be briefly reviewed in Subsection 2.2.4.

Remark 2.2.1. *We also use the terms "gating network" and "expert network" to refer to the set of all gating functions and the set of all expert functions, respectively. In addition, throughout the thesis, we will frequently say "gate k" and "expert k" instead of gating function k and expert function k, respectively.*

So far, x and y represent the generic variables. However, in order to recall the concepts, in the following subsections we will present the modelings of gating and expert functions in the usual vectorial setting, which has been considered in the original proposed papers (Jacobs et al., 1991; Xu et al., 1994b; Jordan and Jacobs, 1994) and many others, although they have been extended to a functional setting that adopts other parametric formulas that we will consider later.

2.2.2 Modeling the gating functions

Let $\{x_i, y_i\}_{i=1}^n$ be a sample of n independently and identically distributed (i.i.d.) data, where $x_i \in \mathbb{R}^p$ is the input and $y_i \in \mathbb{R}^q$ is the output. The most popular models for the function $\text{Gate}_k(x)$ in (2.1) are softmax gating and Gaussian gating function.

■ Softmax gating function

With softmax gating modeling, each function $\text{Gate}_k(x)$ is parameterized by

$$\text{Gate}_k(x_i;\boldsymbol{\alpha}) = \frac{\exp(x_i^\top \alpha_k)}{1+\sum_{k'=1}^{K-1}\exp(x_i^\top \alpha_{k'})}, \tag{2.3}$$

where $\boldsymbol{\alpha} = (\alpha_1^\top,\ldots,\alpha_{K-1}^\top)^\top$, $\alpha_k \in \mathbb{R}^p$ for $k \in [K-1]$, is the unknown parameter vector to be estimated, here and throughout the thesis the superscript $^\top$ denotes the transposition operator. The notation $k \in [K-1]$ means $k = 1,\ldots,K-1$, we will use this convention throughout the thesis. Here, from the constraint $\sum_{k=1}^K \text{Gate}_k(x) = 1$ in the general notion of ME model, one can see that in softmax gating modeling, the Kth gate is constrained to have the form

$$\text{Gate}_K(x;\boldsymbol{\alpha}) = \frac{1}{1+\sum_{k'=1}^{K-1}\exp(x^\top \alpha_{k'})},$$

equivalently, the vector $\boldsymbol{\alpha}_K$ is constrained to be a null vector. The reason for this parameterization is that, without constraining one of the K vectors $\alpha_1,\ldots,\alpha_K$ to a fixed vector, the mixing proportion $\text{Gate}_k(x;\boldsymbol{\alpha})$ can admit an arbitrary number of ways of parameterizing, *e.g.*, $\widetilde{\boldsymbol{\alpha}} = \boldsymbol{\alpha} - \boldsymbol{\gamma}$ for any $\boldsymbol{\gamma} \in \mathbb{R}^{p.K}$. So, constraining the parameter vector of the Kth gate to be null ensures the identification condition for the estimation of the gating network.

■ Gaussian gating function

An frequently used alternative to the softmax gating function is the Gaussian gating function which has the following parametric form

$$\text{Gate}_k(x_i;\boldsymbol{\zeta}) = \frac{\pi_k \phi(x_i;\nu_k,\Sigma_k)}{\sum_{k'}^K \pi_{k'}\phi(x_i;\nu_{k'},\Sigma_{k'})}, \tag{2.4}$$

where

$$\phi(x;\nu,\Sigma) = \det(2\pi\Sigma)^{-1/2}\exp\left[-\frac{1}{2}(x-\nu)^\top\Sigma^{-1}(x-\nu)\right]$$

is the multivariate Gaussian distribution in $\mathbb{R}^p$ with mean $\nu \in \mathbb{R}^p$ and symmetric positive definite covariance matrix $\Sigma \in \mathbb{R}^{p\times p}$. Here, $\pi_k > 0$ for all $k \in [K]$ and $\sum_{k=1}^K \pi_k = 1$, and $\boldsymbol{\zeta} = (\pi_1,\nu_1^\top,\text{vech}^\top\Sigma_1,\ldots,\pi_K,\nu_K^\top,\text{vech}^\top\Sigma_K)^\top$ is the unknown parameter vector of the gating network to be estimated, where $\text{vech}(\cdot)$ is the operator that extracts the unique elements of a symmetric matrix (Henderson and Searle, 1979). Here, one can easily verify the constraint $\sum_{k=1}^K \text{Gate}_k(x_i;\boldsymbol{\zeta}) = 1$ for all x_i.

The modeling (2.4) was first considered by Xu et al. (1994b), since then there have been many using examples in the literature, *e.g.*, Mak and Kung (2000) with application to speaker verification, aki Sato and Ishii (2000) with application to robot dynamics problems using their proposed online EM algorithm, Lima et al. (2007) where Gaussian gating combined with support vector machine

(SVM) experts was applied to problems of nonlinear dynamic system identification, *etc.*

The two softmax and Gaussian gating modelings are shown to be equivalent under some restrictions (Nguyen and Chamroukhi, 2018). In principle, any modeling of the gating network that satisfies the conditions $\text{Gate}_k(x) \geqslant 0$ for all $x \in \mathcal{X}$, $k \in [K]$, and $\sum_{k=1}^{K} \text{Gate}_k(x) = 1$ for all $x \in \mathcal{X}$ can be used as an alternative to (2.3) and (2.4). However, only a few were considered, including Xu et al. (1994a) where the gating network was modeled using the exponential family (the Gaussian gating network is an instance), and Perthame et al. (2016) where Student-t gating function was used in combination with Student-t expert functions to model the air quality on the subways in Paris.

2.2.3 Modeling the expert functions

The modeling of expert functions depends on the problem at hand. For regression problems with continuous responses, the Gaussian expert function is often used, while for classification problems the multinomial expert function is the appropriate choice.

■ Gaussian expert function

When the response y is continuous, $\text{Expert}_k(y|x)$ is usually modeled by the conditional density function of the Gaussian distribution

$$\begin{aligned} \text{Expert}_k(y_i|x_i;\boldsymbol{\theta}_k) &= \phi(y_i; x_i^\top \beta_k, \sigma_k^2) \\ &= (2\pi\sigma_k^2)^{-1/2} \exp\left[-\frac{(y_i - x_i^\top \beta_k)^2}{2\sigma_k^2}\right], \end{aligned} \tag{2.5}$$

where $\boldsymbol{\theta}_k = (\beta_k^\top, \sigma_k^2)^\top$ is the unknown parameter vector of the expert k to be estimated. This modeling is based on the assumption that the error terms ε_i in the linear relationship $y_i = x_i^\top \beta_k + \varepsilon_i$ have normal distribution. Alternatively, for the data containing a group or groups of observations with asymmetric behavior, heavy tails or atypical observations, other assumptions can be put on the error terms, resulting in skew-normal experts, robust t experts, and skew t experts (Chamroukhi, 2015, 2017).

ME models with experts modeled by (2.5) have been widely investigated in many contexts and applied to many regression problems and clustering analysis. We will discuss some notable studies later in this section. For now, we proceed to introduce another important modeling for the experts.

■ Multinomial expert function

When the response y is categorical, *i.e.*, in classification problems, the appropriate choice for the $\text{Expert}_k(y|x)$ in (2.1) is the conditional density function of the categorical distribution[1] given by

$$\begin{aligned}\text{Expert}(y_i|x_i;\boldsymbol{\beta}_k) &= P(y_i|x_i;\boldsymbol{\beta}_k)\\ &= \prod_{g=1}^{G}\left[\frac{\exp(x_i^\top\beta_{kg})}{1+\sum_{g'=1}^{G-1}\exp(x_i^\top\beta_{kg'})}\right]^{y_{ig}},\end{aligned} \tag{2.6}$$

where $\boldsymbol{\beta}_k = (\beta_{k1}^\top,\ldots,\beta_{k,G-1}^\top)^\top$ is the unknown parameter to be estimated, and $y_{ig} = \boldsymbol{I}_{(y_i=g)}$, $g \in [G]$. Here, $\boldsymbol{I}$ denotes the indicator function. Note that, similar to the parameterization of the softmax gating function, the parameter vector responsible for the class G, *i.e.*, β_{kG}, is constrained to be a null vector, this allows the estimation of $\boldsymbol{\beta}_k$ to be identified.

Remark 2.2.2. *The name "multinomial expert" comes from the fact that in machine learning literature, it is common to speak of a "multinomial distribution", although a "categorical distribution" would be more precise. In addition, estimating parameter of the conditional density in (2.6) usually appears in regression analysis literature under the name "multinomial logistic regression", rather than "categorical logistic regression". Moreover, it is also known by a variety of other names such as softmax regression, multiclass logistic regression, maximum entropy classifier, etc.*

In addition to the above mentions, other interesting models for the experts have been considered in the literature include SVM experts (Cao, 2002; Lima et al., 2007), Gaussian processes experts (Tresp, 2001; Jeon and Hwang, 2022; Yang and Ma, 2011). Notably, Jordan and Jacobs (1994) models each expert by another ME model, leading to the so-called hierarchical mixture of experts (HME) models, an important architecture with many applications, as described below.

■ Hierarchical mixture of experts

In HME models, each expert $\text{Expert}_k(y|x)$ in (2.1) is modeled by another ME model, *i.e.*,

$$\text{Expert}_k(y|x;\boldsymbol{\theta}_k) = \sum_{j=1}^{J_k}\text{Gate}_{j|k}(x)\,\text{Expert}_{kj}(y|x), \tag{2.7}$$

where J_k is the number of experts connected to the kth lower-level gating network. Here, the function $\text{Expert}_{kj}(y|x)$ itself can be further modeled by another ME model and so on, resulting in higher level HME models. An illustration of a two-level HME architecture for regression is shown in Figure 2.2.

HME architecture was first introduced in Jordan and Jacobs (1994), since then it has been

[1] occasionally called discrete distribution

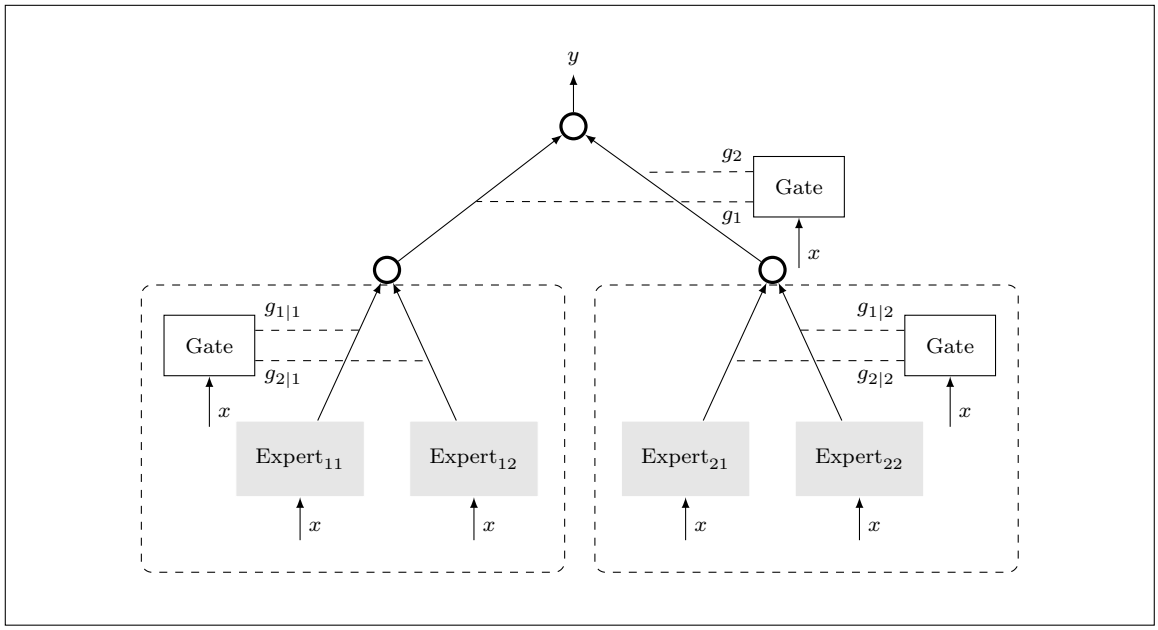


Figure 2.2: Example of a two-level HME model. The two ME (rounded by the two dashed rectangles) are combined with the gate at the top to produce the final response. Each ME model in the dashed rectangles is itself composed of two experts and a gating network.

broadly studied and has found many interesting applications in various fields. In Jordan and Jacobs (1994), the authors illustrated the advantages of HME compared to other approaches such as back propagation network, classification and regression tree (CART), multivariate adaptive regression splines (MARS). In particular, a four-level HME model was used to solve a nonlinear system identification problem, it yielded a relative error better than CART, MARS, and with a running time much shorter than that of the back propagation network. In Ramamurti and Ghosh (1996), the authors presented the use of HME models with multinomial experts for signal classification. In particular, both softmax and Gaussian gating functions were investigated, gathering with multinomial experts, to identify the types of glass in the Glass dataset.

Remark 2.2.3. *The softmax-gated mixture of Gaussian regressions and the softmax-gated mixture of multinomial regressions are the most popular choices for many real-world applications. They and their extensions to functional data will be considered in the thesis.*

2.2.4 Approximation capacity and applications of ME models

The richness of the class of ME models in terms of conditional density approximation capabilities has been recently demonstrated by proving denseness results (Nguyen et al., 2021a, 2019). In this section, we review the theoretical results on the approximation capacity of the ME models. Firstly,

we have an obvious question that: given a continuous function over a compact subset of some Euclidean space, whether we can approximate that function with a ME models? The answer is yes. Moreover, the experts in this case are quiet simple, *i.e.*, linear experts. In particular, consider the ME model in (2.1) in which the gating function $\text{Gate}_k(x)$ is the soft-max function and the experts $\text{Expert}_k(y|x)$ are the linear experts. Then the mean function of this ME model can be written as a function of x as

$$\mathbb{E}[y|x] = \sum_{k=1}^{K} \frac{\exp(\alpha_{k0} + x^\top \alpha_k)}{\sum_{k'}^{K} \exp(\alpha_{k'0} + x^\top \alpha_{k'})} (\beta_{k0} + x^\top \beta_k) := h(x). \tag{2.8}$$

Let $\mathcal{C}(\mathcal{X})$ be the class of continuous functions and $\mathcal{M}(\mathcal{X}) = \{h(x) : h \text{ has form } 2.8\}$ be the class of mean functions of obtained from the mixture of linear experts described above. Then provided $\mathcal{X} \subset \mathbb{R}^p$ is compact, the class $\mathcal{M}(\mathcal{X})$ is dense within the class $\mathcal{C}(\mathcal{X})$ (Nguyen et al., 2016).

Some successful applications of ME architecture may be mentioned here include ME for time series prediction (Zeevi et al., 1996; Yümlü et al., 2003), segmentation (Chamroukhi et al., 2013, 2009), ME for social network data (Gormley and Murphy, 2010), for classification of gender and pose of human faces (Gutta et al., 2000), among many others. For an overview of practical and theoretical aspects of ME modeling, reader is referred to Nguyen and Chamroukhi (2018); Yuksel et al. (2012).

2.3 Estimation of ME models

So far, we have only introduced the ME models by describing their concepts, the modelings for gating and expert functions, and occasionally some applications, but we have not mentioned the approaches for model inference. This section is devoted to presenting a general overview of the estimation for ME models. At the end of this section we provide a list of notable applications of ME models that have been published in the literature.

In this thesis, the ME models will be considered frequently are the softmax gating Gaussian experts, and the softmax gating multinomial experts (and their variants). By softmax gating Gaussian experts we refer the ME model with the gating function given by (2.3) and the expert function given by (2.5). Whereas, by softmax gating multinomial experts, we refer the ME model with the gating function given by (2.3) and the expert function given by (2.6). For simplicity, from now on we will refer to them respectively as ME model with Gaussian regression experts and ME model with multinomial regression experts. In the subsequent parts, we will individually present the maximum likelihood estimation (MLE) of these two models using EM algorithm.

2.3.1 MLE for ME model with Gaussian regression experts

Let $\mathscr{D} = \{(x_i, y_i)\}_{i=1}^n$ be an observed sample of n i.i.d. data pairs (x_i, y_i), where $x_i \in \mathbb{R}^p$ is the input and $y_i \in \mathbb{R}$ is the output. Suppose that the relationship of x_i and y_i follows the ME model

$$f(y_i|x_i;\boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k(x_i;\boldsymbol{\alpha})\phi(y_i; x_i^\top \boldsymbol{\beta}_k, \sigma_k^2),$$

where $\pi_k(x_i;\boldsymbol{\alpha})$ and $\phi(y_i; x_i^\top \boldsymbol{\beta}_k, \sigma_k^2)$ are given as in (2.3) and (2.5), respectively. The objective here is to estimate the unknown parameter $\boldsymbol{\theta} = (\boldsymbol{\alpha}, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_k)$ via MLE. This can be done by maximizing the observed-data log-likelihood function defined as

$$\log L(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log \sum_{k=1}^{K} \pi_k(x_i;\boldsymbol{\alpha})\varphi(y_i; x_i^\top \boldsymbol{\beta}_k, \sigma_k^2) \tag{2.9}$$

using the EM algorithm (Dempster et al., 1977). The EM algorithm starts with an initial solution $\boldsymbol{\theta}^{(0)}$, then alternates the two following steps until convergence, *i.e.*, there is no significant change in the values of the function $\log L(\boldsymbol{\theta})$.

■ **E-step.** For $i = 1, \ldots, n$, calculate the conditional probability memberships $\tau_{ik}^{(s)}$ that the observed pair (x_i, y_i) originates from the kth expert, given the observed data and the current estimate $\boldsymbol{\theta}^{(s)}$, defined by

$$\tau_{ik}^{(s)} = \frac{\pi_k(x_i;\boldsymbol{\alpha}^{(s)})\varphi(y_i; x_i^\top \boldsymbol{\beta}_k^{(s)}, \sigma_k^{2(s)})}{\sum_{k=1}^{K} \pi_k(x_i;\boldsymbol{\alpha}^{(s)})\varphi(y_i; x_i^\top \boldsymbol{\beta}_k^{(s)}, \sigma_k^{2(s)})}.$$

■ **M-step.** Given the memberships $\tau_{ik}^{(s)}$, update the value of the parameter $\boldsymbol{\theta}$ by maximizing w.r.t. $\boldsymbol{\theta}$ the so-called Q-function defined by

$$Q(\boldsymbol{\theta}; \boldsymbol{\theta}^{(s)}) = \sum_{n=1}^{n} \sum_{k=1}^{K} \tau_{ik}^{(s)} \log \left[\pi_k(x_i;\boldsymbol{\alpha})\varphi(y_i; x_i^\top \boldsymbol{\beta}_k, \sigma_k^2)\right]. \tag{2.10}$$

The maximization can be performed separately w.r.t. the gating network parameter $\boldsymbol{\alpha}$ and the expert network parameters $\boldsymbol{\beta}_k$, σ_k as follows.

Updating the gating network parameter $\boldsymbol{\alpha}$ consists of maximizing w.r.t. $\boldsymbol{\alpha}$ the function $Q(\boldsymbol{\alpha}; \boldsymbol{\theta}^{(s)})$ that is the part of (2.10) involved $\boldsymbol{\alpha}$. This is a weighted multinomial logistic regression problem with the weights are the conditional probability memberships $\tau_{ik}^{(s)}$. Since there is no closed-form solution for such problem, one common approach is to use the iteratively reweighted least squares (IRLS) algorithm (Jordan and Jacobs, 1994; Ng and McLachlan, 2004) that starts from an

initial estimate $\boldsymbol{\alpha}^{(0)}$, then updates $\boldsymbol{\alpha}$ iteratively via the following updating formula

$$\boldsymbol{\alpha}^{(t+1)} = \boldsymbol{\alpha}^{(t)} - \Big[\boldsymbol{H}(\boldsymbol{\alpha}; \boldsymbol{\theta}^{(s)})\Big]^{-1}_{\boldsymbol{\alpha}=\boldsymbol{\alpha}^{(t)}} \boldsymbol{g}(\boldsymbol{\alpha}; \boldsymbol{\theta}^{(s)})|_{\boldsymbol{\alpha}=\boldsymbol{\alpha}^{(t)}} \tag{2.11}$$

at each iteration t, where $\boldsymbol{H}(\boldsymbol{\alpha}; \boldsymbol{\theta}^{(s)})$ and $\boldsymbol{g}(\boldsymbol{\alpha}; \boldsymbol{\theta}^{(s)})$ are respectively the Hessian matrix and the gradient vector of $Q(\boldsymbol{\alpha}; \boldsymbol{\theta}^{(s)})$, given in detailed, for example, in the Appendix B.1. We keep running this inner-loop until there is no significant change in $Q(\boldsymbol{\alpha}; \boldsymbol{\theta}^{(s)})$, the obtained solution is then an estimate $\boldsymbol{\alpha}^{(s)}$ for the next EM iteration.

Updating the expert network parameters $\boldsymbol{\beta}_k$ and σ_k consists of solving K independent weighted regression problems with the weights are the conditional probability memberships $\tau_{ik}^{(s)}$. This can be done easily with the following closed-form updating formulas:

$$\boldsymbol{\beta}_k^{(s+1)} = \left[\sum_{i=1}^{n} \tau_{ik}^{(s)} x_i x_i^\top\right]^{-1} \sum_{i=1}^{n} \tau_{ik}^{(s)} y_i x_i$$

$$\sigma_k^{2(s+1)} = \frac{1}{\sum_{i=1}^{n} \tau_{ik}^{(s)}} \sum_{i=1}^{n} \tau_{ik}^{(s)} (y_i - x_i^\top \boldsymbol{\beta}_k^{(s+1)})^2.$$

The EM algorithm increases the value of the log likelihood function $\log L(\boldsymbol{\theta})$ at each iteration, therefore it is guaranteed to converge to at least a local minimum after a finite number of iterations.

2.3.2 MLE for mixture of multinomial regression experts model

ME model with multinomial regression experts is suitable for multiclass classification problems. Let $\mathscr{D} = \{(x_i, y_i)\}_{i=1}^n$ be an observed sample of n i.i.d. data pairs (x_i, y_i), where $x_i \in \mathbb{R}^p$ is the input, and $y_i = (y_{i1}, \ldots, y_{iG}) \in \mathbb{R}^G$ is the categorical output. Suppose that the relationship of x_i and y_i follows the ME model

$$f(y_i|x_i; \boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k(x_i; \boldsymbol{\alpha}) P(y_i|x_i; \boldsymbol{\beta}_k),$$

where $\pi_k(x_i; \boldsymbol{\alpha})$ and $P(y_i|x_i; \boldsymbol{\beta}_k)$ are given as in (2.3) and (2.6), respectively. The objective here is to estimate the unknown parameter $\boldsymbol{\theta} = (\boldsymbol{\alpha}, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_k)$ via MLE. Similarly to the ME model with Gaussian regression experts, this can be done by maximizing the following observed-data log-likelihood function

$$\log L(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log \sum_{k=1}^{K} \pi_k(x_i; \boldsymbol{\alpha}) P(y_i|x_i; \boldsymbol{\beta}_k) \tag{2.12}$$

via the EM algorithm. The **E-step** in this case is analogous to the E-step for the ME model with Gaussian regression experts, with the density function φ is replaced by the density function

$P(y_i|x_i;\boldsymbol{\beta}_k)$, *i.e.*, we calculate the posterior probabilities $\tau_{ik}^{(s)}$ by

$$\tau_{ik}^{(s)} = \frac{\pi_k(x_i;\boldsymbol{\alpha}^{(s)})P(y_i|x_i;\boldsymbol{\beta}_k^{(s)})}{\sum_{k=1}^{K}\pi_k(x_i;\boldsymbol{\alpha}^{(s)})P(y_i|x_i;\boldsymbol{\beta}_k^{(s)})}.$$

In the **M-step** of the EM algorithm, the update for the gating network parameter $\boldsymbol{\alpha}$ is identified with the update for the gating network of the ME model with Gaussian regression experts. This is equivalent to solving a weighted multinomial logistic regression problem, where the weights are the conditional probability memberships $\tau_{ik}^{(s)}$.

However, for each multinomial expert, its parameter vector can no longer be updated in a closed-form as that of the Gaussian experts. It is instead updated using Newton-Raphson method (Fletcher, 1987), which performs an update formula analogous to the update formula of IRLS algorithm in (2.11), but the Hessian matrix and gradient vector are now weighted by the posterior probabilities $\tau_{ik}^{(s)}$ of the current EM iteration. The calculations for these Hessian matrix and gradient vector are provided, for example, in the Appendix B.1. In particular, during the Newton-Raphson loop, the weights $\tau_{ik}^{(s)}$ are fixed, and we keep running this loop until its convergence. The point obtained at convergence is then the update for $\boldsymbol{\beta}_k$ for the next EM iteration.

2.3.3 Regularized MLE for ME model

Alternatively to the MLE, to encourage sparsity in the parameter vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}_k$'s in high-dimensional settings, one may estimate the regularized MLE via maximizing the ℓ_1-regularized observed-data log-likelihood

$$\mathcal{L}(\boldsymbol{\theta}) = \log L(\boldsymbol{\theta}) - \text{Pen}_{\lambda,\chi}(\boldsymbol{\theta}), \tag{2.13}$$

where $\log L(\boldsymbol{\theta})$ is the observed-data log-likelihood of $\boldsymbol{\theta}$ defined in (2.9) or in (2.12), and $\text{Pen}_{\lambda,\chi}(\boldsymbol{\theta})$ is a LASSO regularization term defined by

$$\text{Pen}_{\lambda,\chi}(\boldsymbol{\theta}) = \lambda\sum_{k=1}^{K}\|\boldsymbol{\beta}_k\|_1 + \chi\sum_{k=1}^{K-1}\|\boldsymbol{\alpha}_k\|_1,$$

where $\lambda, \chi \in \mathbb{R}$ are the tuning parameters and $\|\cdot\|_1$ is the ℓ_1-norm operator. The solution to the maximization of (2.13) cannot be obtained in closed form. However, the EM algorithm can be adapted to iteratively maximize (2.13), *e.g.*, see Chamroukhi and Huynh (2019).

The MLE and regularized MLE presented above will be revisited frequently throughout this thesis. In the next chapters, our proposed ME models will oftenly refer to these frameworks for model estimation, the corresponding EM algorithms will also be presented in detailed particularly for each model.

2.4 Functional data analysis

Functional data analysis (FDA) is a branch of statistics that analyzes and provides information about data of types curves, surfaces or anything else varying over a continuum. FDA has roots going back to work by Grenander and Karhunen in the 1940s and 1950s (Grenander, 1950; Müller, 2016). Since then, FDA has been rapidly developed to be a rich subfield of statistics, especially in the last two decades. The term *Functional Data Analysis* was first coined by Ramsay and Silverman (2005).

One of the distinctive features of FDA is its ability to incorporate the correlation within the functional data as well as their derivatives. The key techniques of FDA include data smoothing, dimensionality reduction, functional data clustering, classification, functional linear modeling and forecasting methods. In the subsequent sections, we will review the regression models and the methods of clustering as well as classification for functional data. An excellent review paper on the applications of FDA can be found in Ullah and Finch (2013).

This section aims first to outline some very basic and important concepts of FDA, *e.g.*, the mathematical framework, the basis expansions, and some tools for pre-processing functional data (such as smoothing via penalizing derivatives, curve alignment, *etc*). Secondly, this section aims to review the current state of the art works on FDA, including regression, clustering, and classification for functional data, as well as some related theoretical results. The popular linear models concern functional data will be presented in details in order to prepare for the ME models proposed in the next chapters. For convenience, in this thesis, we use the notation, *e.g.*, $X(t)$ (or sometimes $X(\cdot)$ if there is no ambiguity) to refer to a functional object (more clearly, a function of t), not to a particular value of X at any t. If we want to refer to a particular value, we will make that clear.

2.4.1 Mathematical framework

By *mathematical framework*, we aim at presenting here the general view of the elements (*i.e.*, the functions) that we will work with in FDA, and the basic properties of the space they live in. Firstly, in FDA, each sample element of functional data is considered to be a random function. The random functions can be viewed as random elements taking values in some Hilbert space. The common one is $L^2(\mathcal{T})$, the space of all square integrable functions defined on some interval $\mathcal{T} \subset \mathbb{R}$, and we will mainly working on it.

A function $X(t)$ is said to be square integrable on $\mathcal{T}$, *i.e.*, $X \in L^2(\mathcal{T})$, if

$$\int_{\mathcal{T}} [X(t)]^2 \, dt < \infty.$$

As a Hilbert space, $L^2(\mathcal{T})$ enjoys all properties and theoretical results those have been developed for Hilbert spaces. Let X, Y be two functions in $L^2(\mathcal{T})$, their inner product is defined by

$$\langle X, Y \rangle = \int_{\mathcal{T}} X(t)Y(t)dt.$$

Analogous to the Euclidean space, each function X in $L^2(\mathcal{T})$ has an associated norm, define by $\|X\| = \sqrt{\langle X, X\rangle}$, and the distance between two functions, *e.g.*, X and Y, is also defined as the norm of their difference, *i.e.*,

$$\|X - Y\| = \left[\int_{\mathcal{T}} (X(t) - Y(t))^2 \, dt\right]^{1/2}. \tag{2.14}$$

In addition to the Hilbertian point of view, it is common to view random functions from the applied perspective as stochastic processes, which is more suitable in our context of study in this thesis. In particular, each random function $X(t)$ can be viewed as a collection of random variables $\{X(t)\}_{t\in\mathcal{T}}$, with $\mathcal{T} \subset \mathbb{R}$. From this perspective, we can also see that functional data are of infinite dimensional intrinsically.

Mean and covariance functions. Let X be a random function. Similar to the multivariate case, we have the notions of *mean* and *covariance* for random functions. The mean of X is also a function of t, usually denoted by μ, namely, $\mu(t) := \mathbb{E}[X(t)]$. The covariance of X is defined by

$$c(t, u) = \mathbb{E}\left[(X(t) - \mu(t))(X(u) - \mu(u))\right], \quad \text{for } t, u \in \mathcal{T}.$$

Let $X_1, \ldots, X_n$ are realizations of X. The sample mean and sample covariance functions, respectively, are given by

$$\hat{\mu}(t) = \frac{1}{n}\sum_{i=1}^{N} X_i(t),$$
$$\hat{c}(t, u) = \frac{1}{n-1}\sum_{i=1}^{n} \left(X_i(t) - \hat{\mu}(t)\right)\left(X_i(u) - \hat{\mu}(u)\right).$$

The sample mean and sample covariance functions can be viewed as the estimators of $\mu(t)$ and $c(t, u)$ defined above. It has been shown that $\hat{\mu}$ and $\hat{c}$ converge to μ and c in L_2-norm with convergences rate of $O(n^{-1/2})$ (Jacques and Preda, 2013b; Deville, 1974). These functions play an important role in summary statistics for functional data since they provide a view of the population, similar to the sample mean and covariance in multivariate data analysis. We will revisit these functions oftenly throughout the thesis.

2.4.2 Basis expansions

The very first step in working with functional data is to express them via a basis expansion as

$$X_i(t) \approx \sum_{k=1}^{K} x_{ik}\omega_k(t), \quad i \in [n], \tag{2.15}$$

in which, x_{ik} are the coefficients, and $\omega_k(t)$ are some collection of basis functions. There are some popular collections that are oftenly used in FDA, including B-splines, wavelet, Fourier, Gaussian

and sigmoidal. Figure 2.3 shows some examples of the basis functions collections.

Since we will meet basis expansions many times later, we would mention here some useful conventions on using basis expansions. Firstly, throughout the thesis the expansions as in (2.15) are usually written with an equal sign, *i.e.*, "=". In practice, functional data is often available in form of discretized values. However, through the representation with basis functions, the functional data will become available on the whole domain. Therefore, when the time points differ between the functions, using the basis expansions we can put the functions into a common domain.

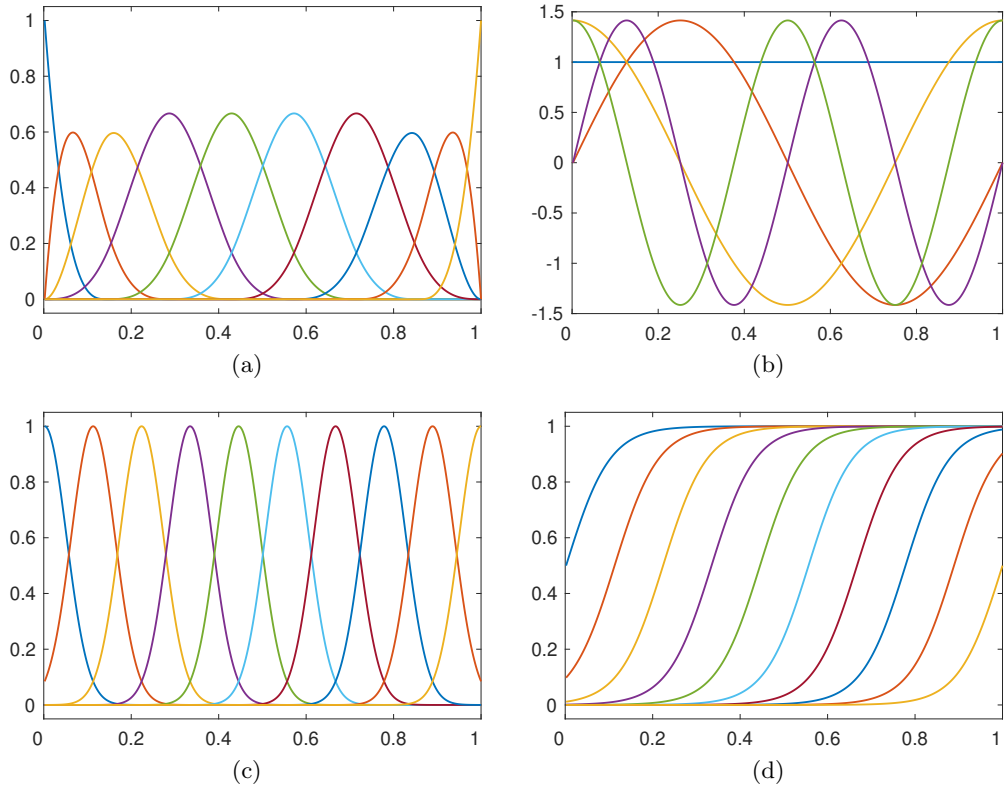


Figure 2.3: Examples of (a) B-splines, (b) Fourier, (c) Gaussian and (d) sigmoidal basis functions

2.4.3 Functional principle component analysis (FPCA)

Analogous to the principal component analysis (PCA) in the classical multivariate analysis, FPCA is an important tool in FDA that helps explaining the variance in the population, via the principal components (PCs), extracting the features, visualizing the data, and is a widely used technique for dimensionality reduction.

The concept of FPCA can be described from the application point of view as follows. Let X be a random function, and $\{X_i\}_{i=1}^n$ be an i.i.d. sample of X. We wish to find a decomposition for

the X_i as

$$X_i(t) = \mu(t) + \sum_{k=1}^{\infty} \xi_{ik} \upsilon_k(t), \tag{2.16}$$

such that the coefficients ξ_{ik} are i.i.d., uncorrelated, and have mean zero; the functions $\mu(t)$ and $\upsilon_k(t)$ are independent of i, $\upsilon_k(t)$ are orthonormal in L^2-space, and importantly X_i can be well approximated by only a small number of the functions $\upsilon_k(t)$. One can see, there are many benefits can be obtained from this decomposition, especially, the dimensionality reduction aspect.

Nowadays, such decomposition in (2.16) is well-known as the Karhunen-Loève expansion of X, and was in fact conceived from 1946 in Karhunen (1946); Loève (1946). Since then, a framework, termed as FPCA, was developed to compute such coefficients and functions as in (2.16) and use them for further data explorations.

Let $c(t, u)$ be the covariance function of X. Consider the following eigenvalue equation

$$\int_{\mathcal{T}} c(t, u)\upsilon(u)du = \lambda\upsilon(t).$$

Denote by $\lambda_1, \lambda_2, \ldots$ the non-increasing ordered eigenvalues of the above equation, and $\upsilon_1, \upsilon_2, \ldots$ their corresponding eigenfunctions. Let $\mu(t)$ be the mean function of X and ξ_{ik} be computed by

$$\xi_{ik} = \int_{\mathcal{T}} (X_i(t) - \mu(t))\upsilon_k(t)dt. \tag{2.17}$$

Then $\mu(t)$, ξ_{ik} and $\upsilon_k(t)$ fulfill the expansion (2.16). In other words, each X_i can be approximated using the mean function, and a linear combination of the eigenfunctions of $c(t, u)$ with coefficients are computed as in (2.17). The eigenfunctions $\upsilon_k(t)$ are called the functional principal components (FPCs). The coefficients ξ_{ik} are called the scores of X_i w.r.t. to the FPCs $\upsilon_k(t)$.

The important properties of the scores ξ_{ik} are that they are independent across i, uncorrelated across k, *i.e.*, $\mathbb{E}[\xi_{ik}] = 0$, $\mathrm{Cov}(\xi_{ik}, \xi_{i\ell}) = 0$ if $k \neq \ell$, moreover, we have

$$\mathrm{Var}[\xi_{ik}] = \lambda_k, \quad \text{and} \quad \mathbb{E}\left[\|X - \mu\|^2\right] = \sum_{k=1}^{\infty} \lambda_k.$$

The latter equation shows that the total variance of X can be decomposed into the sum of the eigenvalues, *i.e.*, variance of its scores. Because λ_k's are arranged in non-increasing order, we can quantify the explanation of the corresponding principal components. We will use FPCA in Chapter 3 as an alternative technique to expand the functions in our models. Figure 2.4 and Figure 2.5 show examples of the estimated eigenfunctions and eigenvalues on the Tecator and Berkeley growth data. In Figure 2.4 we can see that the shape of υ_1 summarizes 98.61% of variability in the population. The subsequent eigenfunctions explains the next important modes of variability. On the other hand, in Figure 2.5, we need three first principal components to explain 98.82% of variability in the data, and the fourth principal component explains only 1.18%. Descriptions of

the Tecator and Berkeley growth data can be found in the Appendix A.2.

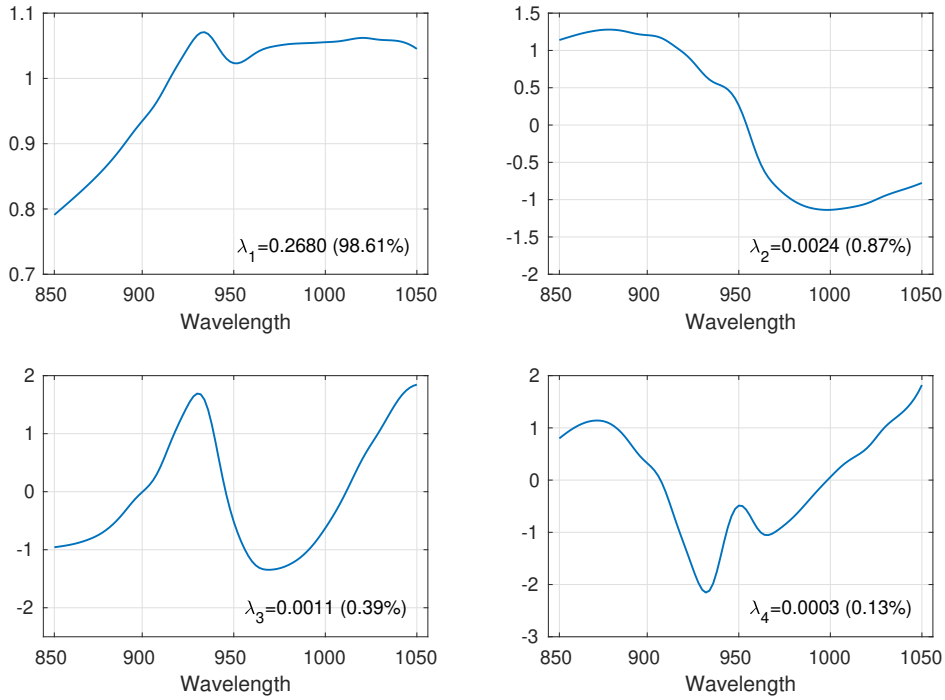


Figure 2.4: The first four estimated eigenfunctions and their associated eigenvalues of the Tecator data.

Applications of FPCA. Several works have used FPCA as the starting point to represent the functional data in their models. By using the scores of the functions corresponding to first few principal components, the construction of the parametric models becomes more parsimonious. In Yao et al. (2010), the authors uses FPCA and proposed a mixture regression model for scalar-on-function regression.

2.5 Scalar-on-function regression

Functional regression models come to deal with situations where one or both of the input and output variables are functions. The coefficient parameters must therefore be appropriately defined for such situations. This section and the next two ones are dedicated to present the three most considered models involving functional data, including: scalar-on-function, function-on-scalar and function-on-function regression models.

Let $\{X_i(t), y_i\}_{i=1}^n$ be a sample of n i.i.d. data pairs where y_i is a real-valued response and $X_i(t)$

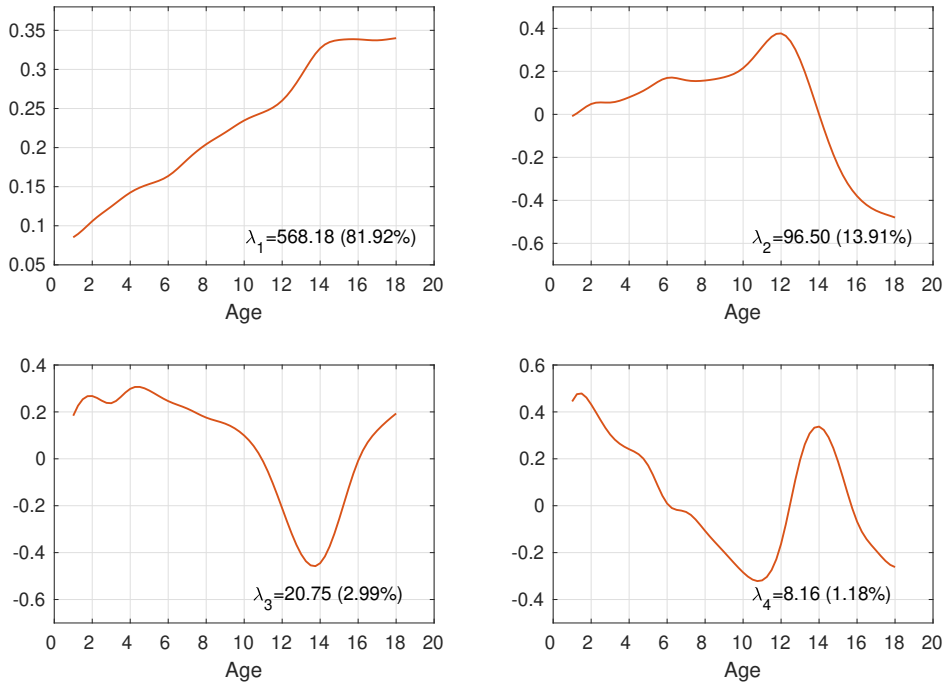


Figure 2.5: The first four estimated eigenfunctions and their associated eigenvalues of the Berkeley growth data.

is a functional predictor with $t \in \mathcal{T} \subset \mathbb{R}$. The scalar-on-function regression model (Müller et al., 2005; Ramsay and Silverman, 2005) is given by

$$y_i = \beta_0 + \int_{\mathcal{T}} X_i(t)\beta(t)dt + \varepsilon_i, \quad i \in [n], \tag{2.18}$$

where β_0 is the unknown intercept and $\beta(t)$ is the unknown functional parameter to be estimated, $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$ are independent Gaussian errors.

The popular approach is to expand the functional parameter $\beta(t)$ using a collection of basis functions $\omega_1(t), \ldots, \omega_K(t)$, *i.e.*,

$$\beta(t) = \sum_{k=1}^{K} b_k \omega(t), \tag{2.19}$$

whee K is generally chosen to be smaller the number of time points at which we observe $X_i(t)$, but sufficiently large to ensure to capture the vary of the functions. Note that, the equation is only approximation, but we use the equality sign "=" to be able to manipulate such expansion without

adding additional error terms. Using the expansion (2.19), the model can be rewritten as

$$\begin{aligned} y_i &= \beta_o + \sum_{k=1}^{K} b_k \int_{\mathcal{T}} X_i(t)\omega_k(t) + \varepsilon_i \\ &= \beta_0 + \sum_{k=1}^{K} b_k x_{ik} + \varepsilon_i \\ &= \beta_0 + \boldsymbol{x}_i \boldsymbol{b} + \varepsilon_i, \quad i \in [n], \end{aligned}$$

where $\boldsymbol{x}_i = (x_[i1], \ldots, x_{iK})^\top$ with $x_{ik} = \int_{cT} X_i(t)\omega_k(t)dt$ is now the new design vector, and $\boldsymbol{b} = (b_1, \ldots, b_K)^\top$ is the unknown parameter to be estimated.

The intercept β_0 and the parameter vector $\boldsymbol{b}$ are then can be estimated by $(\beta_0, \boldsymbol{b}^\top)^\top = (\boldsymbol{X}^\top \boldsymbol{X})^{-1}\boldsymbol{X}^\top \boldsymbol{Y}$, where $\boldsymbol{X}$ and $\boldsymbol{Y}$ denote

$$\boldsymbol{X} = \begin{bmatrix} 1 & x_{11} & \cdots & x_{1K} \\ 1 & x_{21} & \cdots & x_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{nK} \end{bmatrix}, \quad \boldsymbol{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}.$$

Above is the easy-to-use approach for fitting parameter of scalar-on-function regression model. But it is not the only one. Other approaches include functional principal component regression method (Reiss and Ogden, 2007), hybrid method that consists of a FPCA to pre-process predictor functions and penalized splines to model the coefficient function (Goldsmith et al., 2011a). In Yao and Müller (2010), the authors proposed a functional quadratic regression model whose model parameter is estimated using the FPCA approach. Scalar-on-function regression has shown various applications in many fields, including medicine, bioinformatics, *etc.* In the next section we will present the two less-common but still important models in the family of regression models for functional data, they are function-on-scalar, and function-on-function regression models.

2.6 Function-on-scalar regression

Many problems today involve modeling functional responses via scalar explanatory variables, especially in the biological sciences and engineering design. In this section, we present the formulation and estimation methods for the so-called function-on-scalar regression model, which links a functional response to scalar covariates via a linear relationship. This regression model will serve as one of the extensions for the expert regression model that we will consider in Chapter 4.

2.6.1 Model formulation

Suppose that we observe n i.i.d. realizations $\{Y_i(t)\}_{i=1}^n$ of a real-valued smooth function $Y(t)$, $t \in \mathcal{T}$, corresponding to n i.i.d. realizations $\{\mathbf{x}_i\}_{i=1}^n$ of a p-vector valued covariate $\mathbf{x}$. The function-on-scalar regression model is given by

$$Y_i(t) = \beta_0(t) + x_{i1}\beta_1(t) + \ldots + x_{ip}\beta_p(t) + \varepsilon_i(t), \quad i \in [n], \tag{2.20}$$

where $\beta_0(t)$ is the unknown functional intercept and $\beta_1(t), \ldots, \beta_p(t)$, $t \in \mathcal{T}$, are the unknown functional parameters, often referred to as effect functions, to be estimated. Here and throughout the thesis, the notations x_{ij}'s are implicitly understood as the components of the vector $\mathbf{x}_i$, *i.e.*, $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})$. The error function $\varepsilon_i(t)$ is commonly assumed to be independent and has a normal distribution for all $t \in \mathcal{T}$ and for all $i \in [n]$. However, this assumption on the error function could also be replaced by the assumption that $\varepsilon_i(t)$ is a Gaussian process with mean zero and some covariance function $\Sigma_\varepsilon(t, s)$, *i.e.*, in this case we allow the errors to be correlated within individuals, but not between individuals.

The model (2.20) was considered in Faraway (1997) and was applied to an ergonomics design problem. In Faraway (1997), the parameters were estimated pointwisely by the least squares approach. However, it is a pointwise estimation and is only appropriate when the response functions are smooth. When the responses are noisy, a penalty method, as presented in the next subsection, is preferable.

For convenience, let us stack the variables into matrix form as follows

$$\boldsymbol{Y}(t) = \begin{bmatrix} Y_1(t) \\ Y_2(t) \\ \vdots \\ Y_n(t) \end{bmatrix}, \quad \boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix}, \quad \boldsymbol{\beta}(t) = \begin{bmatrix} \beta_1(t) \\ \beta_2(t) \\ \vdots \\ \beta_p(t) \end{bmatrix}, \quad \boldsymbol{\varepsilon}(t) = \begin{bmatrix} \varepsilon_1(t) \\ \varepsilon_2(t) \\ \vdots \\ \varepsilon_n(t) \end{bmatrix}. \tag{2.21}$$

Then, the equations (2.20) can be written in the familiar form as

$$\boldsymbol{Y}(t) = \boldsymbol{X}\boldsymbol{\beta}(t) + \boldsymbol{\varepsilon}(t). \tag{2.22}$$

Note that, in (2.21) and throughout the thesis, in terms of data organization, each curve is aligned as a row of observed values. For example, in practice if the responses are observed in a same grid $t_1, \ldots, t_J$ then such $\boldsymbol{Y}(t)$ will be available as a $n \times J$ matrix. Like that, the matrix multiplication in the equation (2.22) is ensured to be correct in practice.

2.6.2 Penalized least squares estimation

Provided that $\boldsymbol{X}$ has full rank as in the usual regression situation, the least squares estimator of $\boldsymbol{\beta}(t)$ can be obtained pointwisely by

$$\widehat{\boldsymbol{\beta}}_{LS}(t) = (\boldsymbol{X}^\top \boldsymbol{X})^{-1}\boldsymbol{X}^\top \boldsymbol{Y}(t), \quad \forall t \in \mathcal{T}.$$

However, as mentioned earlier, such pointwise least squares estimation may be not suitable for functional data, especially, for noisy data. In this case, the penalized least squares estimation of $\boldsymbol{\beta}(t)$ (*e.g.*, (Kokoszka and Reimherr, 2017, Chapter 5)) defined as follow is more preferred:

$$\widehat{\boldsymbol{\beta}}(t) = \min_{\boldsymbol{\beta}(t)} \left[\int_{\mathcal{T}} \|\boldsymbol{Y}(t) - \boldsymbol{X}\boldsymbol{\beta}(t)\|^2 dt + \sum_{j=1}^{p} \lambda_j \int_{\mathcal{T}} \left[(\mathcal{D}\beta_j)(t)\right]^2 dt \right], \tag{2.23}$$

where λ_j are the penalty constants and $\mathcal{D}$ is some differential operator, for example, the second derivative. Here, $\|\mathbf{f}(t)\|^2 = \sum_{i=1}^{n} f^2(t)$ for all $\mathbf{f}(t) = [f_1(t), \ldots, f_n(t)]^\top$. The following part presents an approach to solve the penalized least squares estimator $\widehat{\boldsymbol{\beta}}(t)$ in (2.23).

Let $\boldsymbol{\omega}(t) = [\omega_1(t), \ldots, \omega_K(t)]^\top$ be a vector of basis functions. Then for each $i \in [n]$ and $j \in [p]$, one can approximate the functions $Y_i(\cdot)$ and $\beta_j(\cdot)$ respectively by

$$Y_i(t) = \sum_{k=1}^{K} r_{ik}\omega_k(t) =: \boldsymbol{r}_i^\top \boldsymbol{\omega}(t), \quad \text{and} \quad \beta_j(t) = \sum_{k=1}^{K} b_{jk}\omega_k(t) =: \boldsymbol{b}_j^\top \boldsymbol{\omega}(t), \tag{2.24}$$

where $\boldsymbol{r}_i$, $\boldsymbol{b}_j \in \mathbb{R}^K$ are the coefficient vectors. Here, K should ensure the tradeoff between smoothness of the functions and complexity of the estimation problem.

By using these basis expansions and denoting $\boldsymbol{R} = [\boldsymbol{r}_1, \ldots, \boldsymbol{r}_n]^\top \in \mathbb{R}^{n\times K}$, $\boldsymbol{B} = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_p]^\top \in \mathbb{R}^{p\times K}$, we can rewrite the equation (2.22) as

$$\boldsymbol{R}\,\boldsymbol{\omega}(t) = \boldsymbol{X}\,\boldsymbol{B}\,\boldsymbol{\omega}(t) + \boldsymbol{\varepsilon}(t).$$

Similarly, the function $(\mathcal{D}\beta_j)(t)$ can also be expressed in terms of $\boldsymbol{B}$ and $\boldsymbol{\omega}(t)$ as

$$(\mathcal{D}\beta_j)(t) = \sum_{k=1}^{K} b_{jk}(\mathcal{D}\omega)(t) =: \boldsymbol{b}_j^\top (\mathcal{D}\boldsymbol{\omega})(t),$$

here $(\mathcal{D}\boldsymbol{\omega})(t)$ obviously denotes the vector obtained by operating $\mathcal{D}$ on the components of $\boldsymbol{\omega}(t)$, *i.e.*, $(\mathcal{D}\boldsymbol{\omega})(t) = [(\mathcal{D}\omega_1)(t), \ldots, (\mathcal{D}\omega_K)(t)]^\top$. Then, the integral in the penalty term can be approximated by

$$\begin{aligned}\int_{\mathcal{T}} \left[(\mathcal{D}\beta_j)(t)\right]^2 dt &= \int_{\mathcal{T}} \boldsymbol{b}_j^\top (\mathcal{D}\boldsymbol{\omega})(t).[(\mathcal{D}\boldsymbol{\omega})(t)]^\top \boldsymbol{b}_j dt \\ &=: \boldsymbol{b}_j^\top \boldsymbol{D}\boldsymbol{b}_j,\end{aligned}$$

where $\boldsymbol{D}$ denotes the remaining cross-product matrix after factorizing $\boldsymbol{b}_j$, that is

$$\boldsymbol{D} = \left[\int_{\mathcal{T}} (\mathcal{D}\omega_k)(t)(\mathcal{D}\omega_\ell)(t)dt\right]_{1\leqslant k,\ell\leqslant K} \in \mathbb{R}^{K\times K}. \tag{2.25}$$

It is worth noting that $\boldsymbol{D}$ is a non-negative definite matrix.

Hence, the objective function in (2.23) can now be rewritten by

$$\int_{\mathcal{T}} \|\boldsymbol{R}\,\boldsymbol{\omega}(t) - \boldsymbol{X}\boldsymbol{B}\,\boldsymbol{\omega}(t)\|^2 dt + \sum_{j=1}^{p} \lambda_j \boldsymbol{b}_j^\top \boldsymbol{D}\boldsymbol{b}_j. \tag{2.26}$$

The coefficient matrix $\boldsymbol{B}$ (hence $\boldsymbol{b}_j$'s) minimizing (2.26) can be obtained by solving a first order differential equation as for ridge regression in multivariate analysis. Particularly, the penalized least squares estimator for the coefficient matrix $\boldsymbol{B}$ can be calculated in a condensed form using the vectorization operator and the Kronecker product operator as follow:

$$\mathrm{vec}(\widehat{\boldsymbol{B}}^\top) = \left(\boldsymbol{U}^\top\boldsymbol{U} + \boldsymbol{\Lambda}\otimes\boldsymbol{D}\right)^{-1}\boldsymbol{U}^\top\mathrm{vec}(\boldsymbol{I}_\omega^{1/2}\boldsymbol{R}^\top), \quad \boldsymbol{U} = \boldsymbol{X}\otimes\boldsymbol{I}_\omega^{1/2}, \tag{2.27}$$

in which, $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1,\ldots,\lambda_p)$ and $\boldsymbol{I}_\omega^{1/2}$ is the square root of the non-negative definite matrix $\boldsymbol{I}_\omega$ defined by

$$\boldsymbol{I}_\omega = \left[\int_{\mathcal{T}} \omega_k(t)\omega_\ell(t)dt\right]_{1\leqslant k,\ell\leqslant K} \in \mathbb{R}^{K\times K}.$$

Note that if the chosen basis is orthonormal then $\boldsymbol{I}_\omega$ is reduced to an identity matrix. The details of the derivation of (2.27) can be found in Appendix A.1.1.

Remark 2.6.1. *The solution for $\boldsymbol{B}$ given in (2.27) will serve in Section 4.4 as the update formula for the experts in our proposed function-on-scalar ME model.*

2.7 Function-on-function regression

This type of regression problem appears in the situation where both the responses and predictors are functions. In this case, the relation between functional response $Y_i(t)$, $t \in \mathcal{T}$, and functional

predictor $X_i(u)$, $u \in \mathcal{U}$, can be modeled by the following fully functional linear model

$$Y_i(t) = \beta_0(t) + \int_{\mathcal{U}} X_i(u)\beta(t,u)du + \varepsilon_i(t), \quad i \in [n],$$

where $\beta_0(t)$, defined on $\mathcal{T}$, is the unknown functional intercept and $\beta(t,u)$, defined on $\mathcal{T} \times \mathcal{U}$, is the unknown bivariate regression coefficient function, often referred to as the *kernel*, to be estimated. Similarly to the function-on-scalar regression case, the error function $\varepsilon_i(t)$ is usually assumed to be independent for all $i \in [n]$, $t \in \mathcal{T}$ and has Gaussian distribution with mean zero and covariance matrix $\mathbf{\Sigma}_\varepsilon$, *i.e.*, $\varepsilon_i(t) \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_\varepsilon)$. From now on we will refer to this model by *function-on-function regression model* (FFRM).

FFRM model was first considered by Ramsay and Silverman (1997), in which the functional parameter $\beta(t,u)$ is estimated by minimizing the integrated sum of squares. Shimokawa et al. (2000) extended it to the case of there are more than one covariate, *i.e.*, functional multiple regression model. Yamanishi and Tanaka (2003) suggested a geographically weighted regression model to explore the functional relationship between the variables. Müller and Yao (2008) proposed an adaptive approach to FFRM where the functional parameters are estimated with regularization. In Matsui et al. (2013), the authors proposed a maximum penalized likelihood approach to estimate $\beta(t,u)$ since the maximum likelihood approach and least squares approach have been shown to potentially produce unstable estimates for the functional parameter $\beta(t,u)$. Recently, Beyaztas and Shang (2020) proposed a partial least squares approach to estimate the parameter of the FFRM model, overcoming the singular matrix problem that occurs when a large number of functional predictors are included in the FFRM.

2.7.1 Model formulation and least squares estimation

Since one can replace the functions X_i and Y_i by their centered versions, the functional intercept $\beta_0(t)$ can be simplified from the model. Therefore, in the remainder we consider the following model

$$Y_i(t) = \int_{\mathcal{U}} X_i(u)\beta(t,u)du + \varepsilon_i(t), \quad i \in [n], \tag{2.28}$$

with assumption that $\mathbb{E}X = 0$ and consequently $\mathbb{E}Y = 0$.

Let $\boldsymbol{\omega}^\star(t,u) = \{\omega_1^\star(t,u), \ldots, \omega_K^\star(t,u)\}$ be a collection of bivariate basis functions defined on $\mathcal{T} \times \mathcal{U}$. Then with K sufficiently large, the bivariate functional parameter $\beta(t,u)$ can be expressed by

$$\beta(t,u) = \sum_{k=1}^{K} b_k \omega_k^\star(t,u), \tag{2.29}$$

with b_k are the coefficients. Therefore, the model (2.28) can be rewritten as

$$\begin{aligned} Y_i(t) &= \int_{\mathcal{U}} X_i(u) \left(\sum_{k=1}^{K} b_k \omega_k^\star(t,u) \right) du + \varepsilon_i(t) \\ &= \sum_{k=1}^{K} b_k \int_{\mathcal{U}} X_i(u) \omega_k^\star(t,u) du + \varepsilon_i(t) \\ &=: \sum_{k=1}^{K} b_k X_{ik}^\star(t) + \varepsilon_i(t), \quad i \in [n], \end{aligned} \tag{2.30}$$

where $X_{ik}^\star(t) := \int_{\mathcal{U}} X_i(u)\omega_k^\star(t,u)du$ are functions of t which serve as new predictors of the model, and b_k are now the unknown coefficients to be estimated. Let us denote

$$\boldsymbol{X}^\star(t) = \begin{bmatrix} X_{11}^\star(t) & X_{12}^\star(t) & \cdots & X_{1K}^\star(t) \\ X_{21}^\star(t) & X_{22}^\star(t) & \cdots & X_{2K}^\star(t) \\ \vdots & \vdots & \ddots & \vdots \\ X_{n1}^\star(t) & X_{n2}^\star(t) & \cdots & X_{nK}^\star(t) \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_K \end{bmatrix}.$$

Then the equations (2.30) can be rewritten simply under matrix form as

$$\boldsymbol{Y}(t) = \boldsymbol{X}^\star(t)\, \boldsymbol{b} + \boldsymbol{\varepsilon}(t), \tag{2.31}$$

where $\boldsymbol{Y}(t)$ and $\boldsymbol{\varepsilon}(t)$ are as in (2.21), and the matrix multiplication is understood formally, *i.e.*, each $X_{ik}^\star(t)$ is treated as an element of the matrix $\boldsymbol{X}^\star(t)$. Such equations (2.22) and (2.31) are the yet simplest and condensed representations for the considered models.

Suppose that the $Y_i(t)$ are observed on a finite grid which does not depend on i, and $X_i^\star(t)$ can also be calculated on the same grid. If we organize all the curves in $\boldsymbol{Y}(t)$ and $\boldsymbol{X}^*(t)$ vertically (to make the equation (2.31) holds in practice) then we can estimate $\boldsymbol{b}$ directly with raw data by

$$\widehat{\boldsymbol{b}} = \left(\boldsymbol{X}^{\star\top}(t)\boldsymbol{X}^\star(t) \right)^{-1} \boldsymbol{X}^{\star\top}(t)\boldsymbol{Y}(t).$$

However, analogously as in functional-on-scalar model, such raw approach is sensitive with noise and potentially produce rough estimate for $\beta(t,u)$. Therefore, here again, we express $Y_i(t)$ and $X_{ik}^\star(t)$ using basis expansions and work on their coefficient vectors as done previously for the functional-on-scalar model. Moreover, a penalization on the derivatives of $\beta(t,u)$ is necessary to produce a smooth estimation for $\beta(t,u)$.

Remark 2.7.1. *The collection of bivariate basis functions $\boldsymbol{\omega}^\star(t,u)$ are generally constructed using two collections of univariate basis functions. In particular, if $\{\omega_k(t),\ k \geqslant 1\}$ and $\{\psi_\ell(u),\ \ell \geqslant 1\}$ are bases in $L^2(\mathcal{T})$ and $L^2(\mathcal{U})$, then $\{\omega_k(t)\psi_\ell(u),\ k,\ell \geqslant 1\}$ is a basis in $L^2(\mathcal{T}\times\mathcal{U})$. The orthonormal property can also be inherited.*

2.7.2 Penalized least squares estimation

Similarly to (2.23), the penalized least squares estimation of the parameter $\beta(t,u)$ is defined as follow

$$\widehat{\boldsymbol{\beta}}(t) = \min_{\boldsymbol{\beta}(t)} \left[\int_{\mathcal{T}} \|\boldsymbol{Y}(t) - \boldsymbol{X}^{\star}(t)\,\boldsymbol{b}\|^2 dt + \mathrm{Pen}_{\lambda}(\beta(t,u)) \right], \tag{2.32}$$

where $\mathrm{Pen}_{\lambda}(\beta(t,u))$ is the penalty term that responsible for the smoothness of $\beta(t,u)$. It it common to penalize the t and u directions separately with a penalty term taking the following form

$$\mathrm{Pen}_{\lambda_t,\lambda_u}(\beta(t,u)) = \lambda_t \iint [(\mathcal{D}_t\beta)(t,u)]^2\, dtdu + \lambda_u \iint [(\mathcal{D}_u\beta)(t,u)]^2\, dtdu, \tag{2.33}$$

where $\mathcal{D}_t$ (resp. $\mathcal{D}_u$) is the derivative operator w.r.t. t (resp. u), and λ_t, λ_u are the smoothness penalty constants. For example, $\mathcal{D}_t$ and $\mathcal{D}_u$ are usually chosen to be the second-order partial derivative operators $\frac{\partial^2}{\partial t^2}$ and $\frac{\partial^2}{\partial u^2}$.

Let $\boldsymbol{\psi}(t) = [\psi_1(t), \ldots, \psi_L(t)]^{\top}$ be a vector of basis functions defined on $\mathcal{T}$. Firstly, we can express $Y_i(t)$ and $X^{\star}_{ik}(t)$ as follows

$$\begin{aligned} Y_i(t) &= \sum_{\ell=1}^{L} r_{i\ell}\psi_{\ell}(t) =: \boldsymbol{r}_i^{\top}\boldsymbol{\psi}(t), \\ X^{\star}_{ik}(t) &= \sum_{\ell=1}^{L} x^{\star}_{ik\ell}\psi_{\ell}(t) =: \boldsymbol{x}^{\star\top}_{ik}\boldsymbol{\psi}(t), \end{aligned}$$

where $\boldsymbol{r}_i \in \mathbb{R}^L$ and $\boldsymbol{x}^{\star}_{ik} \in \mathbb{R}^L$ are the vectors of coefficients. In practice, the coefficients $x^{\star}_{ik\ell}$ can be calculated directly using $X_i(t)$ and the basis responsible for covariate u that construct $\boldsymbol{\omega}^{\star}(t,u)$, see Remark 2.7.1, particularly,

$$\boldsymbol{x}^{\star}_{ik} = \left[\int_{\mathcal{U}} X_i(u)\omega_1(u)du, \ldots, \int_{\mathcal{U}} X_i(u)\omega_L(u)du \right]^{\top} \in \mathbb{R}^L.$$

Let us denote $\boldsymbol{R} = [\boldsymbol{r}_1^{\top}, \ldots, \boldsymbol{r}_n^{\top}]^{\top} \in \mathbb{R}^{n\times L}$, and

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{x}^{\star\top}_{11} & \boldsymbol{x}^{\star\top}_{12} & \cdots & \boldsymbol{x}^{\star\top}_{1K} \\ \boldsymbol{x}^{\star\top}_{21} & \boldsymbol{x}^{\star\top}_{22} & \cdots & \boldsymbol{x}^{\star\top}_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{x}^{\star\top}_{n1} & \boldsymbol{x}^{\star\top}_{n2} & \cdots & \boldsymbol{x}^{\star\top}_{nK} \end{bmatrix} \in \mathbb{R}^{n\times KL}.$$

Then, the model (2.31) can be rewritten in the basis expansion fashion as

$$\boldsymbol{R}\,\boldsymbol{\psi}(t) = \boldsymbol{G}\boldsymbol{b}\,\boldsymbol{\psi}(t) + \boldsymbol{\varepsilon}(t), \tag{2.34}$$

in which, we conventionally treat each vector $\boldsymbol{x}^{\star\top}_{ik}$ as an element when performing the matrix

multiplication. Thus, the function-on-function regression problem (2.28) now has been approximated by the tractable finite dimensional vector-on-vector regression problem (2.34).

Next, analogously to the penalized least squares estimation for function-on-scalar model, the penalty term in (2.32) can also be expressed in terms of the coefficient vector $\boldsymbol{b}$ and derivatives of the bivariate basis functions. In particular, we have

$$\begin{aligned}\iint \left[(\mathcal{D}_t\beta)(t,u)\right]^2 dtdu &= \sum_{k,\ell=1}^{K} b_k b_\ell \iint (\mathcal{D}_t\omega_k)(t,u).(\mathcal{D}_t\omega_\ell)(t,u)dtdu \\ &=: \boldsymbol{b}^\top \boldsymbol{D}_t \boldsymbol{b}, \end{aligned} \tag{2.35}$$

where $\boldsymbol{D}_t$ denotes the $K \times K$ matrix whose entries calculated by $\iint (\mathcal{D}_t\omega_k)(t,u).(\mathcal{D}_t\omega_\ell)(t,u)dtdu$, with $k, \ell \in [K]$. Similarly, the smoothness penalty term corresponding to the u direction can also be expressed by

$$\begin{aligned}\iint \left[(\mathcal{D}_u\beta)(t,u)\right]^2 dtdu &= \sum_{k,\ell=1}^{K} b_k b_\ell \iint (\mathcal{D}_u\omega_k)(t,u).(\mathcal{D}_u\omega_\ell)(t,u)dtdu \\ &=: \boldsymbol{b}^\top \boldsymbol{D}_u \boldsymbol{b}, \end{aligned} \tag{2.36}$$

with $\boldsymbol{D}_u$ is defined correspondingly.

Finally, given the above notations, the penalized least squares estimator of $\boldsymbol{b}$ takes the following formula

$$\widehat{\boldsymbol{b}} = \left(\boldsymbol{V}^\top \boldsymbol{V} + \lambda_t \boldsymbol{D}_t + \lambda_u \boldsymbol{D}_u\right)^{-1} \boldsymbol{V}^\top \mathrm{vec}(\boldsymbol{I}_\psi^{1/2} \boldsymbol{R}^\top), \tag{2.37}$$

in which $\boldsymbol{V}$ is defined by

$$\boldsymbol{V} = \begin{bmatrix} \boldsymbol{I}_\psi^{1/2}\boldsymbol{x}_{11}^\star & \boldsymbol{I}_\psi^{1/2}\boldsymbol{x}_{12}^\star & \cdots & \boldsymbol{I}_\psi^{1/2}\boldsymbol{x}_{1K}^\star \\ \boldsymbol{I}_\psi^{1/2}\boldsymbol{x}_{21}^\star & \boldsymbol{I}_\psi^{1/2}\boldsymbol{x}_{22}^\star & \cdots & \boldsymbol{I}_\psi^{1/2}\boldsymbol{x}_{2K}^\star \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{I}_\psi^{1/2}\boldsymbol{x}_{n1}^\star & \boldsymbol{I}_\psi^{1/2}\boldsymbol{x}_{n2}^\star & \cdots & \boldsymbol{I}_\psi^{1/2}\boldsymbol{x}_{nK}^\star \end{bmatrix} \in \mathbb{R}^{nL \times K},$$

where $\boldsymbol{I}_\psi^{1/2}$ is the square root of the non-negative definite matrix $\boldsymbol{I}_\psi$ defined by

$$\boldsymbol{I}_\psi = \left[\int_{\mathcal{T}} \psi_k(t)\psi_\ell(t)dt\right]_{1 \leqslant k,\ell \leqslant L} \in \mathbb{R}^{L \times L}.$$

The detailed calculation to establish (2.37) can be found in Appendix A.1.2.

2.7.3 Maximum likelihood estimation

Another way to approach the model estimation is via maximum likelihood method. Firstly, we assume that the error functions $\varepsilon_i(t)$ can also be represented by linear combination of basis functions in $\boldsymbol{\psi}(t)$, *i.e.*,

$$\varepsilon_i(t) = \sum_{\ell=1}^{L} \varepsilon_{i\ell}\psi_\ell(t) =: \boldsymbol{\varepsilon}_i^\top \boldsymbol{\psi}(t),$$

with $\boldsymbol{\varepsilon}_i \sim \mathcal{N}_L(\mathbf{0}, \boldsymbol{\Sigma})$, for some unknown symmetric matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{L\times L}$. Then, the equation (2.34) can be rewritten by

$$\boldsymbol{R}\,\boldsymbol{\psi}(t) = \boldsymbol{G}\boldsymbol{b}\,\boldsymbol{\psi}(t) + \boldsymbol{\mathcal{E}}\boldsymbol{\psi}(t),$$

where $\boldsymbol{\mathcal{E}} = (\boldsymbol{\varepsilon}_1^\top, \ldots, \boldsymbol{\varepsilon}_n^\top)^\top \in \mathbb{R}^{n\times L}$ is the matrix of error coefficients. Observe that the matrix $\boldsymbol{\psi}(t)\boldsymbol{\psi}^\top(t)$ is non-singular, therefore the above equation implies

$$\boldsymbol{R} = \boldsymbol{G}\boldsymbol{b} + \boldsymbol{\mathcal{E}}, \tag{2.38}$$

which takes a familiar form as in ordinary multivariate regression if the matrix multiplication is conventionally understood as in (2.34). Alternatively, if we reframe the matrices $\boldsymbol{R}$ and $\boldsymbol{\mathcal{E}}$ into column vectors, equation (2.38) can also be rewritten as

$$\mathrm{vec}(\boldsymbol{R}^\top) = \boldsymbol{H}\boldsymbol{b} + \mathrm{vec}(\boldsymbol{\mathcal{E}}^\top), \tag{2.39}$$

where the matrix $\boldsymbol{H}$, differs from $\boldsymbol{G}$, is formed by the vectors $\boldsymbol{x}_{ik}^\star \in \mathbb{R}^L$ as follow

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{x}_{11}^\star & \boldsymbol{x}_{12}^\star & \cdots & \boldsymbol{x}_{1K}^\star \\ \boldsymbol{x}_{21}^\star & \boldsymbol{x}_{22}^\star & \cdots & \boldsymbol{x}_{2K}^\star \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{x}_{n1}^\star & \boldsymbol{x}_{n2}^\star & \cdots & \boldsymbol{x}_{nK}^\star \end{bmatrix} \in \mathbb{R}^{nL\times K}. \tag{2.40}$$

Note, however, that (2.39) is not the usual linear equation met in linear regression with continuous response, because the noise here may not be i.i.d., and more importantly, the observations are not independent of each other since the matrices are correlated within each L-rows block due to the correlation between the components of $\boldsymbol{r}_i$ and $\boldsymbol{x}_{ik}^\star$.

From the representation (2.38), we see that the model estimation can be viewed as maximum likelihood estimation for the following probability density function

$$f(\boldsymbol{r}_i|\mathbf{x}_i;\boldsymbol{\theta}) = \det(2\pi\boldsymbol{\Sigma})^{1/2} \exp\left[\frac{1}{2}(\boldsymbol{r}_i - \boldsymbol{Z}_i\boldsymbol{b})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{r}_i - \boldsymbol{Z}_i\boldsymbol{b})\right], \tag{2.41}$$

where $\boldsymbol{Z}_i := [\boldsymbol{x}_{i1}^\star, \ldots, \boldsymbol{x}_{iK}^\star] \in \mathbb{R}^{L\times K}$, and $\boldsymbol{\theta} = \{\boldsymbol{b}, \boldsymbol{\Sigma}\}$ is the unknown parameter of the model to be

estimated. The maximum likelihood estimators of $\boldsymbol{b}$ and $\boldsymbol{\Sigma}$ are given by

$$\widehat{\boldsymbol{b}} = \left(\boldsymbol{H}^\top \boldsymbol{H}\right)^{-1} \boldsymbol{H}^\top \mathrm{vec}(\boldsymbol{R}^\top), \tag{2.42}$$

$$\widehat{\boldsymbol{\Sigma}} = \frac{1}{n}\left(\boldsymbol{R} - \boldsymbol{G}\widehat{\boldsymbol{b}}\right)^\top \left(\boldsymbol{R} - \boldsymbol{G}\widehat{\boldsymbol{b}}\right). \tag{2.43}$$

where the matrix $\boldsymbol{H}$ is given in (2.40).

Remark 2.7.2. *The solution for $\boldsymbol{b}$ and $\boldsymbol{\Sigma}$ presented here will serve in Section 4.4 as the update formula for the experts in our proposed function-on-scalar ME model.*

2.8 Clustering and classification of functional data

Clustering and classification are important topics of FDA. The purpose of this section is to review the most important concepts and methods related to clustering and classification of functional data, via illustrative examples, to help the reader have a picture about the fields.

2.8.1 Clustering of functional data

Similarly to the classical multivariate data analysis, clustering functional data concerns an upsupervised learning process that group a set of functional data into homogenous subgroups, *i.e.*, clusters, such that whose members are more similar than across clusters with respect to some metric. The concepts and algorithms developed for clustering multivariate data are therefore can be extended to functional data, certainly with additional considerations regarding to, *e.g.*, the infinite dimension, the approximations of distance measures, *etc.*

There are various algorithms for clustering functional data. They can be categorized into four groups based on their approaches: raw-data clustering, two-stage clustering, non-parametric clustering, and model-based clustering.

Raw-data clustering. This is the most simple approach that uses directly the observed values of the functions as vectorial data for clustering. However, due to the so-called "*curse of dimensionality*", *i.e.*, these raw-data are typically of high-dimension, the techniques in this direction mostly consist of a dimension reduction step followed by an algorithm for clustering vectorial data. The most often used dimension reduction methods includes principal component analysis (PCA) and factor analysis (FA). For a complete review of PCA and FA methods, we would refer reader to Bishop (2006), Chapter 12.

Two-stage clustering. This approach consists of a filtering step followed by a classical clustering algorithm for finite dimensional data. Although it seems to be similar to the raw-data clustering that, their difference is that: rather than using PCA or FA, in the filtering step, we use the tools of FDA to represent the functional data. In particular, the representations via Spline basis (*e.g.*, in Abraham et al. (2003); Rossi et al. (2004)) and FPCA (*e.g.*, in Jacques and Preda (2013a); Peng and Müller (2008)) are the most common choices in this direction. Once the coefficients are obtained, they are fitted to the usual clustering algorithms such as k-means (see, *e.g.*, Hartigan (1975); Hartigan and Wong (1979)) or Self-Organised Map (SOM) (Kohonen and Schroeder, 1995).

Non-parametric clustering. This terminology covers all clustering techniques that involve non-parametric algorithms. Analogously for multivariate data, the techniques in this approach require a dissimilar measure for two objects. The common choice is the measure d_ℓ defined based on the L_2-distance in (2.14), given by

$$d_\ell(X_i, X_j) = \left[\int_{\mathcal{T}} (X_i^{(\ell)}(t) - X_j^{(\ell)}(t))^2 dt\right]^{1/2},$$

where $X_i^{(\ell)}$ denotes the ℓ-derivative of X. Note that, with $\ell = 0$ the measure d_ℓ becomes exactly the distance defined in (2.14), which is equivalent to using the raw-data approach.

In this direction, there are several interesting works, for example, in Ieva et al. (2013) the authors used K-means combing with d_0, d_1, and $(d_0^2 + d_1^2)^{1/2}$ to cluster ECG[2] traces, aim at early detect the heart failure, replacing a traditional clinical observation. In Tarpey and Kinateder (2003), K-means and d_0 were used for clustering Gaussian processes. Moreover, cluster centres are shown to be a linear combinations of the FPCA eigenfunctions. Figure 2.6 shows the result of doing clustering for Tecator data (see Appendix A.2) using d_0, d_1 and d_2 measures combined with K-means algorithm.

Model-based clustering. Let us briefly recall the ideas of the method of model-based clustering for multivariate data. In the view of model-based clustering, the observations $\{\mathbf{x}_i\}_{i=1}^n$ are assumed to be generated according to a K-component mixture distribution. Let $\mathbf{z}_i = (z_{i1}, \ldots, z_{iK})$ be the unknown one-hot vector that encodes the membership of $\mathbf{x}_i$, and the aim here is to estimated $\mathbf{z}_i$'s. This can be done by treating $\mathbf{z}_i$'s as missing data and maximizing the so-called complete-data log-likelihood function given by

$$\begin{aligned}\mathcal{L}(\boldsymbol{\Psi}; \mathbf{x}_1, \ldots, \mathbf{x}_n, \mathbf{z}_1, \ldots, \mathbf{z}_n) &= \sum_{i=1}^n \log \prod_{k=1}^K \Big[\mathbb{P}(z_{ik} = 1) f_k(\mathbf{x}_i; \theta_k)\Big]^{z_{ik}} \\ &= \sum_{i=1}^n \sum_{k=1}^K z_{ik} \log(\pi_k f_k(\mathbf{x}_i; \theta_k)),\end{aligned}$$

[2] electrocardiograph

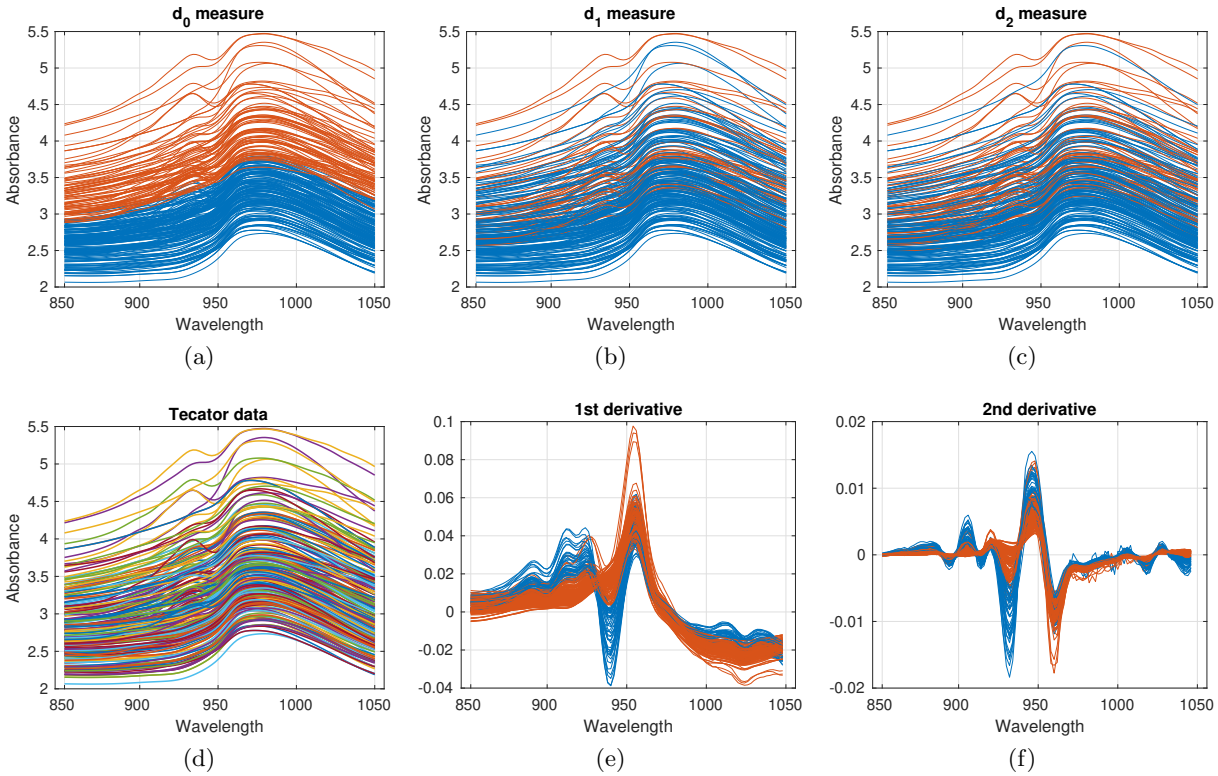


Figure 2.6: Examples of clustering Tecator data using non-parametric technique. Here, the curves are clustered into two groups by K-means using (a) d_0 measure, (b) d_1 measure, and (c) d_2 measure. Bottom panels: (d) Tecator data, (e) the first derivative, and (f) the second derivative curves.

where $\boldsymbol{\Psi} = (\theta_1, \ldots, \theta_K, \pi_1, \ldots, \pi_K)$ is the model parameter to be estimated, $f_k(\mathbf{x}_i|\theta_k)$ is the density corresponding to the kth cluster, parameterized by θ_k, and π_k is the probability that an observation belongs to the kth cluster. The common choice of $f_k(\mathbf{x}_i|\theta_k)$ is the multivariate normal distribution with mean μ_k and variance Σ_k (*i.e.*, $\theta_k = (\mu_k, \Sigma_k)$), for example in Banfield and Raftery (1993), Celeux and Govaert (1995). The inference is usually carried by an iterative procedure such as EM algorithm. Finally, the cluster labels are estimated using the Maximum A Posteriori (MAP) rule

$$\hat{\mathbf{z}}_i = \text{argmax}_{k\in[K]} \frac{\hat{\pi}_k f_k(\mathbf{x}_i; \hat{\theta}_k)}{\sum_{k=1}^{K} \hat{\pi}_k f_k(\mathbf{x}_i; \hat{\theta}_k)}.$$

In this model-based direction for clustering functional data, the techniques are generally consists of two tasks: one performs function expansion (using basis functions or FPCA) and one performs model-based clustering. However, as highlighted in Jacques and Preda (2013b), it differs from the two-stage direction in that it performs the two mentioned tasks simultaneously, whereas in two-stage techniques, clustering is applied once the functions were already "converted" to vectors of coefficients.

The first model-based clustering algorithm for functional data was proposed by James and Sugar (2003), in which the spline expansion coefficients of the curves are assumed to be distributed according to a mixture Gaussian distribution with cluster-specific means μ_k and common variance Σ. Later, Giacofci et al. (2013) proposed using wavelet expansion as an alternative, since the spline expansion is only convenient when the curves are regular but not appropriate for irregular (*e.g.*, peak-like) data. The authors assumes a Gaussian model on the wavelet decomposition of the curves whose parameters are estimated via MLE, and applies the approach on mass spectrometry data, as well as proposes an original application on microarray CGH[3] data. In Chapter 3, via the FME model, we also propose a model-based clustering technique and apply for Canadian weather and DTI[4] data.

2.8.2 Classification of functional data

Similar to regression and clustering for functional data, due to the practical needs, classification for functional data also has a long history of study with many notable applications. Many approaches have been proposed for functional data classification, they are mostly based on functional regression models that treat class labels as responses and the observed functional data as predictors. For example, in Müller et al. (2005), the authors used the PCA in combination with logistic regression model for classification problems, and applied to simulated and medfly data. The generalized linear model has also been extended to functional data (James, 2002; Müller, 2005; Müller et al., 2005). Recently, in Mousavi and Sørensen (2018), the authors applied a wavelet basis expansion in combination with multinomial functional regression model to classify acceleration signals in the problem of lameness diagnosis for horses, and periodogram signals in phoneme data. In their

[3] Comparative Genomic Hybridization. CGH profiles are oftenly used to identify molecular subtypes of cancer
[4] Diffusion Tensor Imaging

proposed model, the parameter vector is estimated via regularized MLE which produces a certain level of sparsity. In Chapter 4, our proposed FME model for classification also use the phoneme data as a benchmark and compare the result with that of Mousavi and Sørensen (2018).

Let $X(t)$, $t \in \mathcal{T}$, be a functional predictor and y be its categorical response, $y \in [G]$. In multinomial functional regression model, the probability that y belongs to category g is modeled by

$$\mathbb{P}(y = g|X(t)) = \frac{\exp\{\beta_{g,0} + \int_{\mathcal{T}} \beta_g(t)X(t)dt\}}{\sum_{g'}^{G} \exp\{\beta_{g',0} + \int_{\mathcal{T}} \beta_{g'}(t)X(t)dt\}}, \tag{2.44}$$

where $\beta_{g,0}$ are the unknown constant intercepts, and $\beta_g(t), t \in \mathcal{T}$, are the unknown coefficient functions to be estimated. Similar to the softmax modeling for the gating function in ME models, model (2.44) is over-parameterized since it can get a same probability by adding a constant to all of the intercepts and coefficient functions. Therefore, one can also constraint, *e.g.*, the parameter responsible for gth class $(\beta_{g,0}, \beta_g(t))$ to be null, which yields the model

$$\mathbb{P}(y = g|X(t)) = \frac{\exp\{\beta_{g,0} + \int_{\mathcal{T}} \beta_g(t)X(t)dt\}}{1 + \sum_{g'}^{G-1} \exp\{\beta_{g',0} + \int_{\mathcal{T}} \beta_{g'}(t)X(t)dt\}}.$$

We will use this parameterization for the functional gating function of the FME model proposed in Chapter 3. Alternatively, over-parameterization in (2.44) can be dealt by the LASSO regularization as can be seen in Mousavi and Sørensen (2018).

Classification of functional data can also be approached using discriminant analysis. The early works include Hall et al. (2001) and James and Hastie (2001a), where linear discrimination analysis (LDA) was applied on the scores obtained from PCA technique, and on the coefficients from spline expansion, respectively. In Ferraty and Vieu (2003), the authors proposed a non-parametric approach for functional data. Later, Chang et al. (2014) developed a kernel-based non-parametric approach, applied to classify the major depressive disorders based on positron emission tomography images. Berlinet et al. (2008) proposed a supervised wavelet-based functional data classification method. Along this line of discriminant analysis, theoretical results have been also developed for linear and quadratic decision boundaries (Delaigle and Hall, 2012, 2013).

2.9 Preliminary on divergence

In order to prepare for Chapter 5 where we propose a distributed learning approach for ME models, in this section we briefly present some preliminaries on general notations of divergence and distance on the space of probability measures.

Let us recall the general notions of divergence and distance defined on the space of probability

measures.

Definition 2.9.1 (Divergence)**.** *Let Θ be a space and $\rho(\cdot,\cdot)$ be a bivariate function defined on Θ. Then $\rho(\cdot,\cdot)$ is called a divergence if $\rho(\theta_1,\theta_2) \geqslant 0$ for all $\theta_1,\theta_2 \in \Theta$, with equality holding if and only if $\theta_1 = \theta_2$.*

Kullback-Leibler (KL) is one of the most used divergences in machine learning to measures the similarity between two probability distributions, or can be interpreted as the information gain from using p instead of q (Cover and T., 2006).

Definition 2.9.2 (KL-divergence)**.** *Let p and q are two probability distributions, KL-divergence between p and q is defined by*

$$\mathrm{KL}\left(p\|q\right) = \int p(u) \log \frac{p(u)}{q(u)} du.$$

In our case, u being $(\mathbf{x}, y)$, the KL-divergence $\mathrm{KL}\left(p\|q\right)$ can be factorized as

$$\begin{aligned}
\mathrm{KL}\left(p(\mathbf{x},y)\|q(\mathbf{x},y)\right) &= \iint p(\mathbf{x},y) \log \frac{p(\mathbf{x},y)}{q(\mathbf{x},y)} d\mathbf{x}dy \\
&= \int p(\mathbf{x}) \log \frac{p(\mathbf{x})}{q(\mathbf{x})} d\mathbf{x} + \int p(\mathbf{x}) \int p(y|\mathbf{x}) \log \frac{p(y|\mathbf{x})}{q(y|\mathbf{x})} d\mathbf{x}dy \\
&= \mathrm{KL}\left(p(\mathbf{x})\|q(\mathbf{x})\right) + \mathbb{E}_{p(\mathbf{x})}\big[\,\mathrm{KL}\left(p(y|\mathbf{x})\|q(y|\mathbf{x})\right)\big], \qquad (2.45)
\end{aligned}$$

where

$$\mathrm{KL}\left(p(y|\mathbf{x})\|q(y|\mathbf{x})\right) := \int p(y|\mathbf{x}) \log \frac{p(y|\mathbf{x})}{q(y|\mathbf{x})} dy. \qquad (2.46)$$

The last term in (2.45) is known as conditional KL-divergence, which is the expected value of the KL-divergence between conditional distributions $p(y|\mathbf{x})$ and $q(y|\mathbf{x})$ where the expectation is taken with respect to $p(\mathbf{x})$.

2.10 Summary

This chapter has introduced the main framework we will working on in the next chapters, *i.e.*, ME models. Notably, the general notion, the modelings, as well as the estimation procedure for gating and expert networks have been presented, these will make the following chapters more consistent and easier to follow. This chapter has also present the necessary tools and concepts for FDA such as data pre-processing, FPCA, functional linear models. Interested reader is referred

to other comprehensive books on functional regression models Kokoszka and Reimherr (2017), Ramsay and Silverman (2005), Ramsay et al. (2011). While the ME models with vectorial setting has been widely studied in literature, their functional versions are still lack of investigations and applications. This thesis therefore proposes, in the subsequent chapters, three new ME models to deal with situations where data are available under functional forms.

Chapter 3

Functional mixtures of experts

Contents

3.1 Introduction

Recall that the general notion of a ME is given by

$$\mathrm{ME}(y|x) = \sum_{k=1}^{K} \mathrm{Gate}_k(x)\, \mathrm{Expert}_k(y|x). \tag{3.1}$$

To the best of our knowledge, ME models have been exclusively studied in multivariate analysis when the inputs are vectors, *i.e.*, $x \in \mathcal{X} = \mathbb{R}^p$. However, in many problems, the predictors and/or the responses are observed from smooth functions. Indeed, in many situations, unlike in predictive and cluster analyses of multivariate and potentially high-dimensional heterogeneous data, which have been studied with the ME modeling in (3.1), the observed data may arise from continuously observed processes, *e.g.*, time series. Thus, a multivariate (vectorial) analysis does not allow one to enough capture the inherent functional structure of the data. In such situations, classical multivariate models are not adapted as they ignore the underlying intrinsic nature and structure of the data. Functional Data Analysis (FDA) (Ramsay and Silverman, 2005; Ferraty and Vieu, 2006) in which the individual data units are assumed to be functions, rather than vectors, offers an adapted framework to deal with continuously observed data, including in regression, classification and clustering. FDA considers the observed data as (discretized) values of smooth functions, rather than multivariate observations represented in the form of "simple" vectors.

The study of functional data has been considered in most of the statistical modeling and inference problems including regression, classification, clustering, functional graphical models (Qiao et al., 2019), among others. In regression, functional linear models have been introduced including penalized functional regression (Élodie Brunel et al., 2016; Goldsmith et al., 2011b) and in particular the FLiRTI approach, a functional linear regression constructed upon interpretable regularization (James et al., 2009), and more generally generalized linear models with functional predictors (Müller et al., 2005; James, 2002), which cover functional logistic regression for classification. In classification, we can also cite functional linear discriminant analysis (James and Hastie, 2001b), and, as a penalized model, Lasso-regularized functional logistic regression (Mousavi and Sørensen, 2017). To deal with heterogeneous functional data, the construction of mixture models with functional data analytic aspects have been introduced for model-based clustering (Liu and Yang, 2009) including Lasso-regularized mixtures for functional data (Devijver, 2017; James and Sugar, 2003; Jacques and Preda, 2014; Chamroukhi and Nguyen, 2019). The resulting functional mixture models are better able to handle functional data structures compared to standard multivariate mixtures.

The problem of clustering and prediction in presence of functional observations from heterogeneous populations, leading to complex distributions, is still however less investigated. In this chapter, we investigate the framework of Mixtures-of-Experts (ME) models, as models of choice in modeling heterogeneity in data for prediction and clustering with vectorial observations, and extend it to the functional data framework. Firstly, we introduce in **Section 3.2** a new family of functional ME models to relate a functional predictor to a scalar response, and develop a dedicated EM algorithm for the maximum-likelihood parameter estimation. Secondly, to deal with potential high-dimensional setting of the introduced FME model, we develop in **Section 3.2.6** a Lasso-regularized approach, which consists of a penalized MLE estimation via a hybrid EM-Lasso algorithm, which integrates an optimized coordinate ascent procedure to efficiently implement the M-Step. Thirdly, we in particular present in **Section 3.3** and extended FME model, which is constructed upon a sparse and highly-interpretable regularization of the functional expert and gating parameters.

The resulting model, abbreviated as iFME, is fitted by regularized MLE via an dedicated EM algorithm. The developed algorithms for the two introduced ME models are applied and evaluated in **Section 3.4** on several simulated scenarios and on real-world data sets, in both clustering and non-linear regression. This chapter is a detailed version of the paper Chamroukhi et al. (2022).

3.2 Functional Mixtures-of-Experts (FME)

We wish to derive and fit new ME models in presence of functional predictors and functional responses. In this chapter, we first consider ME models with a functional predictor $X(\cdot)$ and a real response Y where the pair arises from heterogeneous population composed of unknown K homogeneous sub-populations.

3.2.1 ME with functional predictor and scalar response

Let $\{X_i(t), Y_i\}_{i=1}^n$, be a sample of n i.i.d. data pairs where Y_i is a real-valued response and $X_i(t)$ is a functional predictor with $t \in \mathcal{T} \subset \mathbb{R}$, for example the time in time series. First, to model the conditional relationships between the continuous response Y and the functional predictor $X(\cdot)$, given an expert z, we formulate each expert component $\text{Expert}_z(y|\boldsymbol{x})$ in (3.1) as a functional regression model (cf. Müller et al. (2005), James et al. (2009)). The resulting functional expert regression model for the ith observation takes the following stochastic representation

$$Y_i = \beta_{z_i,0} + \int_{\mathcal{T}} X_i(t)\beta_{z_i}(t)dt + \varepsilon_i, \quad i \in [n], \tag{3.2}$$

where $\beta_{z_i,0}$ is an unknown constant intercept, $\beta_{z_i}(t)$, $t \in \mathcal{T}$ is the function of unknown coefficients of functional expert z_i, and $\varepsilon_i \sim \mathcal{N}(0, \sigma_{z_i}^2)$ are independent Gaussian errors, $z_i \in [K]$ being the unknown label of the expert generating the ith observation. In this context, the response Y is related to the entire trajectory of $X(\cdot)$. Let $\boldsymbol{\beta} = \{\beta_{z,0}, \beta_z(t), t \in \mathcal{T}\}_{z=1}^K$ represent the set of unknown functional parameters for the experts network.

Now consider the modeling of the gating network in the proposed functional ME model. As in the context of ME for vectorial data, different choices are possible to model the gating network function, typically softmax-gated or Gaussian-gated ME (*e.g.*, see Nguyen and Chamroukhi (2018), Xu et al. (1994a), Chamroukhi et al. (2019)). A standard choice as in Jacobs et al. (1991) to model the gating network $\text{Gate}_z(\boldsymbol{x})$ in (3.1) is to use the multinomial logistic (softmax) function as a distribution of the latent variable Z. In this functional data modeling context with $K \geqslant 2$ experts, we use a multinomial logistic function as an extension of the functional logistic regression presented in Mousavi and Sørensen (2018) for linear classification. The resulting functional softmax gating

network then takes the following form

$$\begin{aligned}\pi_z\left(X(t), t \in \mathcal{T}; \boldsymbol{\alpha}\right) &= \mathbb{P}(Z = z | X(t), t \in \mathcal{T}; \boldsymbol{\alpha}) \\ &= \frac{\exp\{\alpha_{z,0} + \int_{\mathcal{T}} X(t)\alpha_z(t)dt\}}{1 + \sum_{z'=1}^{K-1} \exp\{\alpha_{z',0} + \int_{\mathcal{T}} X(t)\alpha_{z'}(t)dt\}},\end{aligned} \tag{3.3}$$

where $\boldsymbol{\alpha} = \{\alpha_{z,0}, \alpha_z(t),\, t \in \mathcal{T}\}_{z=1}^{K}$ is the set of unknown constant intercept coefficients $\alpha_{z,0}$ and functional parameters $\alpha_z(t), t \in \mathcal{T}$ for each expert $z \in [K]$. Note that model (3.3) is equivalent to assuming that each expert z is related to the entire trajectory $X(\cdot)$ via the following functional linear predictor for the gating network

$$\begin{aligned}h_z(X(t), t \in \mathcal{T}; \boldsymbol{\alpha}) &= \log\left\{\frac{\pi_z\left(X(t), t \in \mathcal{T}; \boldsymbol{\alpha}\right)}{\pi_K\left(X(t), t \in \mathcal{T}; \boldsymbol{\alpha}\right)}\right\} \\ &= \alpha_{z,0} + \int_{\mathcal{T}} X(t)\alpha_z(t)dt.\end{aligned} \tag{3.4}$$

The objective is to estimate the functional parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ of the FME model defined by (3.2)-(3.3), from an observed sample. In this setting with functional predictors, this requires estimating a possibly infinite number of coefficients (as many as the number of temporal observations for the predictor). In order to reduce the complexity of the problem, the observed functional predictor can be projected onto a fixed number of basis functions so that we sufficiently capture enough the functional structure of the data, and sufficiently reduce enough the number of coefficients to estimate.

3.2.2 Smoothing representation of the functional experts

Here we consider the case of fixed design, that is, the covariates $X_i(t)$ are non-random functions. We suppose that the $X_i(\cdot)$'s are measured with error at any given time t. Hence, instead of observing directly $X_i(t)$, one has a noisy version of it $U_i(t)$, defined as

$$U_i(t) = X_i(t) + \delta_i(t), \quad i \in [n],$$

where $\delta_i(\cdot) \sim \mathcal{N}(0, \sigma_\delta^2)$ are measurement errors assumed to be independent of the $X_i(\cdot)$'s and the Y_i's. Since the functional predictors $X_i(t)$ are not directly observed, we first construct an approximation of $X_i(t)$ from the noisy predictors $U_i(t)$ by projecting the latter onto a set of continuous basis functions. Let $\boldsymbol{b}_r(t) = [b_1(t), \ldots, b_r(t)]^\top$ be a r-dimensional (B-spline, Fourier, Wavelet) basis, then with r sufficiently large $X_i(t)$ can be represented as

$$X_i(t) = \sum_{j=1}^{r} x_{ij} b_j(t) = \boldsymbol{x}_i^\top \boldsymbol{b}_r(t), \tag{3.5}$$

where $x_{ij} = \int_{\mathcal{T}} X_i(t) b_j(t) dt$ for $j \in [r]$ and $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{ir})^\top$. Since $X_i(t)$ is not observed, the representation coefficients x_{ij}'s are unknown. Hence we propose an unbiased estimator of x_{ij}

defined as

$$\widehat{x}_{ij} := \int_{\mathcal{T}} U_i(t) b_j(t) dt.$$

Thus, an estimate $\widehat{X}_i(t)$ of $X_i(t)$ is given by

$$\widehat{X}_i(t) = \widehat{\boldsymbol{x}}_i^\top \boldsymbol{b}_r(t), \quad i \in [n], \tag{3.6}$$

with $\widehat{\boldsymbol{x}}_i = (\widehat{x}_{i1}, \ldots, \widehat{x}_{ir})^\top$.

Similarly, to represent the regression coefficient functions $\beta_z(\cdot)$, consider a p-dimensional basis $\boldsymbol{b}_p(t) = [b_1(t), b_2(t), \ldots, b_p(t)]^\top$. Then the function $\beta_z(t)$ can be represented as

$$\beta_z(t) = \boldsymbol{\eta}_z^\top \boldsymbol{b}_p(t) \tag{3.7}$$

where $\boldsymbol{\eta}_z = (\eta_{z,1}, \eta_{z,2}, \ldots, \eta_{z,p})^\top$ is the vector of unknown coefficients and the choice of p should ensure the tradeoff between smoothness of the functional predictor and complexity of the estimation problem. We select $r \geqslant p$ to satisfy the identifiability constraint (see for instance Goldsmith et al. (2011b), Ramsay and Silverman (2002)). Furthermore, rather than assuming a perfect fit of $\beta_z(t)$ by $\boldsymbol{b}_p(t)$ as in (3.7), we use for each Gaussian expert regressor z, the following error model as proposed by James et al. (2009) for functional linear regression

$$\beta_z(t) = \boldsymbol{\eta}_z^\top \boldsymbol{b}_p(t) + e(t)$$

where $e(t)$ represents the approximation error of $\beta_z(t)$ by the linear projection (3.7). As we choose $p \gg n$, $|e(t)|$ can be assumed to be small.

3.2.3 Smoothing representation of the functional gating network

Since here we are examining functional predictors, an appropriate representation has also to be given for the gating network (3.3) with functional parameters $\{\alpha_z(t), t \in \mathcal{T}\}_{z=1}^K$. Due to the infinite number of these parameters, we also represent the gating network by a finite set of basis functions similarly as for the experts network. For the representation of the functional predictors $X_i(t)$, $i \in [n]$, we use $\widehat{X}_i(t)$ established in (3.6). The coefficients function $\alpha_z(t)$ is represented similarly as for the β coefficients function of the experts network, by using a q-dimensional basis $\boldsymbol{b}_q(t) = [b_1(t), b_2(t), \ldots, b_q(t)]^\top$, $(q \leqslant r)$ via the projection

$$\alpha_z(t) = \boldsymbol{\zeta}_z^\top \boldsymbol{b}_q(t), \tag{3.8}$$

where $\boldsymbol{\zeta}_z = (\zeta_{z1}, \zeta_{z2}, \ldots, \zeta_{zq})^\top$ is the vector of softmax coefficients function . Note that here we use the same type of basis functions for both representations of β_z and α_z, but one can use different types of bases if needed. Then, by using the representations (3.6) and (3.8) of $X(t)$ and $\alpha_z(t)$, respectively, in the linear predictor $h_z(.)$ defined in (3.4) for $i \in [n]$, the latter is thus approximated

as

$$h_z(U_i(t), t \in \mathcal{T}; \boldsymbol{\alpha}) \quad = \quad \alpha_{z_i,0} + \boldsymbol{\zeta}_{z_i}^\top \mathbf{r}_i, \tag{3.9}$$

where $\mathbf{r}_i = \left[\int_{\mathcal{T}} \boldsymbol{b}_r(t)\boldsymbol{b}_q(t)^\top dt\right]^\top \widehat{\boldsymbol{x}}_i$. Thus, following its definition in (3.3), the functional softmax gating network is approximated as

$$\pi_k(\mathbf{r}_i; \boldsymbol{\xi}) = \frac{\exp\{\alpha_{k,0} + \boldsymbol{\zeta}_k^\top \mathbf{r}_i\}}{1 + \sum_{k'=1}^{K-1} \exp\{\alpha_{k',0} + \boldsymbol{\zeta}_{k'}^\top \mathbf{r}_i\}} \tag{3.10}$$

where $\boldsymbol{\xi} = \left((\alpha_{1,0}, \boldsymbol{\zeta}_1^\top), \ldots, (\alpha_{K-1,0}, \boldsymbol{\zeta}_{K-1}^\top)\right)^\top \in \mathbb{R}^{(q+1)(K-1)}$ is the unknown parameter vector of the functional gating network to be estimated.

3.2.4 The FME model conditional density

We now have appropriate representations for the functional predictors, as well as for both the functional gating network and the functional experts network, involved in the construction of the functional ME (FME) model (3.2)-(3.3). Gathering (3.6) and (3.7), the stochastic representation (3.2) of the FME model can thus be defined as follows,

$$Y_i|u_i(\cdot) \quad = \quad \beta_{z_i,0} + \boldsymbol{\eta}_{z_i}^\top \mathbf{x}_i + \varepsilon_i^\star, \quad i \in [n], \tag{3.11}$$

where $\mathbf{x}_i = \left[\int_{\mathcal{T}} \boldsymbol{b}_r(t)\boldsymbol{b}_p(t)^\top dt\right]^\top \widehat{\boldsymbol{x}}_i$ and $\varepsilon_i^\star = \varepsilon_i + \widehat{\boldsymbol{x}}_i^\top \int_{\mathcal{T}} \boldsymbol{b}_r(t)e(t)dt$. From this stochastic representation under the Gaussian assumption for the error variable ε_i, the conditional density of each approximated functional expert $z_i = k$ is thus given by

$$f(y_i|u_i(\cdot), z_i = k; \boldsymbol{\theta}_k) = \phi(y_i; \beta_{k,0} + \boldsymbol{\eta}_k^\top \mathbf{x}_i, \sigma_k^2), \tag{3.12}$$

where $\phi(.; \mu, v)$ is the Gaussian probability density function (pdf) with mean μ and variance v, $\beta_{k,0} + \boldsymbol{\eta}_k^\top \mathbf{x}_i$ is the mean of the approximated functional regression expert, σ_k^2 its variance, and $\boldsymbol{\theta}_k = (\beta_{k,0}, \boldsymbol{\eta}_k^\top, \sigma_k^2)^\top \in \mathbb{R}^{p+2}$ the unknown parameter vector of expert density k, $k \in [K]$ to be estimated. Finally, combining (3.12) and (3.10) in the ME model (3.1), leads to the the following conditional density defining the FME model,

$$f(y_i|u_i(\cdot); \boldsymbol{\Psi}) = \sum_{k=1}^{K} \pi_k(\mathbf{r}_i; \boldsymbol{\xi}) \phi(y_i; \beta_{k,0} + \boldsymbol{\eta}_k^\top \mathbf{x}_i, \sigma_k^2), \tag{3.13}$$

where $\boldsymbol{\Psi} = (\boldsymbol{\xi}^\top, \boldsymbol{\theta}_1^\top, \ldots, \boldsymbol{\theta}_K^\top)^\top$ is the parameter vector of the model to be estimated.

3.2.5 Maximum likelihood estimation via the EM algorithm

The FME model (3.13) is now defined upon an adapted finite representation of the functional predictors, and its parameter estimation can then be performed given an observed data sample. We

first consider the maximum likelihood estimation framework via the EM algorithm (Dempster et al., 1977; Jacobs et al., 1991) which has many desirable properties including stability and convergence guarantees (eg. see McLachlan and Krishnan (2008), for more details). Note that here we use the term maximum likelihood estimation to not unduly clutter the clarity of the text, while as it will be specified later, we refer to the conditional maximum likelihood estimator.

In practice, the data are available in the form of discretized values of functions. The noisy functional predictors $U_i(t)$ are usually observed at discrete sampling points $t_{i1} < \ldots < t_{im_i}$ with $t_{ij} \in \mathcal{T}$ for $j \in [m_i]$. We suppose that $U_i(t)$ is scaled such that $0 \leqslant t \leqslant 1$ and divide the time period $[0, 1]$ up into a fine grid of m_i points $t_{i1}, \ldots, t_{im_i}$. Thus, in (3.11) we have $\mathbf{x}_i = \left[\sum_{j=1}^{m_i} \boldsymbol{b}_r(t_{ij})\boldsymbol{b}_p(t_{ij})^\top\right]^\top \widehat{\boldsymbol{x}}_i$, $\mathbf{r}_i = \left[\sum_{j=1}^{m_i} \boldsymbol{b}_r(t_{ij})\boldsymbol{b}_q(t_{ij})^\top\right]^\top \widehat{\boldsymbol{x}}_i$, where $\widehat{x}_{ij} = \sum_{j=1}^{m_i} U_i(t_{ij})b_j(t_{ij})$. Note that if we choose $p = q = r$, then $\mathbf{x}_i = \mathbf{r}_i = \widehat{\boldsymbol{x}}_i$. Let $\mathscr{D} = \{(\boldsymbol{u}_1, y_1), \ldots, (\boldsymbol{u}_n, y_n)\}$ be an i.i.d sample of n observed data pairs where $\boldsymbol{u}_i = (u_{i,1}, \ldots, u_{i,m_i})$ is the observed functional predictor for the ith response y_i.

We use $\mathscr{D}$ to estimate the parameter vector $\boldsymbol{\Psi}$ by iteratively maximizing the observed data log-likelihood,

$$\log L(\boldsymbol{\Psi}) = \sum_{i=1}^{n} \log \sum_{k=1}^{K} \pi_k(\mathbf{r}_i; \boldsymbol{\xi})\phi(y_i; \beta_{k,0} + \boldsymbol{\eta}_k^\top \mathbf{x}_i, \sigma_k^2), \tag{3.14}$$

via the EM algorithm. As detailed in Appendix B.1, the EM algorithm for the FME model is implemented as follows. After starting with an initial solution $\boldsymbol{\Psi}^{(0)}$, it alternates, at each iteration s, between the two following steps, until convergence (when there is no longer a significant change in the values of the log-likelihood (3.14)).

■ **E-step.** Calculate the following conditional probability memberships $\tau_{ik}^{(s)}$ (for all $i \in [n]$), that the observed pair (u_i, y_i) originates from the kth expert, given the observed data and the current parameter estimate $\boldsymbol{\Psi}^{(s)}$,

$$\tau_{ik}^{(s)} = \mathbb{P}(Z_i = k | y_i, u_i(\cdot); \boldsymbol{\Psi}^{(s)}) = \frac{\pi_k(\mathbf{r}_i; \boldsymbol{\xi}^{(s)})\phi(y_i; \beta_{k,0}^{(s)} + \mathbf{x}_i^\top \boldsymbol{\eta}_k^{(s)}, \sigma_k^{2(s)})}{f(y_i | u_i(\cdot); \boldsymbol{\Psi}^{(s)})}. \tag{3.15}$$

■ **M-step.** Update the value of the parameter vector $\boldsymbol{\Psi}$ by maximizing the Q-function (B.2) with respect to $\boldsymbol{\Psi}$. The maximization is performed by separate maximizations w.r.t the gating network parameters $\boldsymbol{\xi}$ and, for each expert k, w.r.t the expert network parameters $\boldsymbol{\theta}_k$, for each of the K experts.

Updating the gating network's parameters $\boldsymbol{\xi}$ consists of maximizing w.r.t $\boldsymbol{\xi}$ the part of (B.2) that is a function of $\boldsymbol{\xi}$. Since we use a softmax-gated expert network in (3.10), this maximization problem consists of a weighted multinomial logistic problem for which there is no a closed-form solution. We then use a Newton-Raphson (NR) procedure, which iteratively maximizes (B.3) after starting from an initial parameter vector $\boldsymbol{\zeta}^{(0)}$, by updating, at each NR iteration t, the values of

the parameter vector $\boldsymbol{\xi}$ according to the following updating formula:

$$\boldsymbol{\xi}^{(t+1)} = \boldsymbol{\xi}^{(t)} - \Big[H(\boldsymbol{\xi}; \boldsymbol{\Psi}^{(s)})\Big]^{-1}_{\boldsymbol{\xi}=\boldsymbol{\xi}^{(t)}} g(\boldsymbol{\xi}; \boldsymbol{\Psi}^{(s)})\big|_{\boldsymbol{\xi}=\boldsymbol{\xi}^{(t)}}, \tag{3.16}$$

where $H(\boldsymbol{\xi}; \boldsymbol{\Psi}^{(s)})$ and $g(\boldsymbol{\xi}; \boldsymbol{\Psi}^{(s)})$ are, respectively, the Hessian matrix and the gradient vector of $Q(\boldsymbol{\xi}; \boldsymbol{\Psi}^{(s)})$, and are provided in Appendix B.1. At each NR iteration, the Hessian matrix and gradient vector are evaluated at the current value of $\boldsymbol{\xi}$. We keep updating the gating network parameter $\boldsymbol{\xi}$ according to (3.16) until there is no significant change in $Q(\boldsymbol{\xi}; \boldsymbol{\Psi})$. The maximization then provides $\boldsymbol{\xi}^{(s+1)}$ for the next EM iteration.

Updating the experts network parameters $\boldsymbol{\theta}_k$ consists of solving K independent weighted regression problems where the weights are the conditional expert memberships $\tau_{ik}^{(s)}$ given by (3.15). The updating formulas for the regression parameters $(\beta_{k,0}, \boldsymbol{\eta}_k)$ and the noise variances σ_k^2 for each expert k are straightforward and correspond to weighted versions of those of standard Gaussian linear regression, *i.e.*, weighted ordinary least squares. The updating rules for the experts network parameters are given by the following formulas:

$$\begin{aligned}
\beta_{k,0}^{(s+1)} &= (n_k^{(s)})^{-1} \sum_{i=1}^{n} \tau_{ik}^{(s)} (y_i - \mathbf{x}_i^\top \boldsymbol{\eta}_k^{(s)}); \quad \boldsymbol{\eta}_k^{(s+1)} = \Big[\sum_{i=1}^{n} \tau_{ik}^{(s)} \mathbf{x}_i \mathbf{x}_i^\top\Big]^{-1} \sum_{i=1}^{n} \tau_{ik}^{(s)} (y_i - \beta_{k,0}^{(s+1)}) \mathbf{x}_i, \\
\sigma_k^{2(s+1)} &= (n_k^{(s)})^{-1} \sum_{i=1}^{n} \tau_{ik}^{(s)} \Big[y_i - (\beta_{k,0}^{(s+1)} + \mathbf{x}_i^\top \boldsymbol{\eta}_k^{(s+1)})\Big]^2,
\end{aligned} \tag{3.17}$$

where $n_k^{(s)} = \sum_{i=1}^{n} \tau_{ik}^{(s)}$ represents the expected cardinal number of component k.

This EM algorithm, designed here for the FME that is constructed upon smoothing of the functional data, can be seen as a direct extension of the vectorized version the mixture-of-experts model. While it can hence be expected to provide accurate estimations as in the vector predictors setting, the number of parameters to estimate here in the case of the functional ME can still be high, for example when a big number of basis functions is used to have more accurate approximation of the functional predictors. In that case, it is better to regularize the maximum likelihood estimator in-order to establish a compromise between the quality of fit and complexity.

3.2.6 Regularized maximum likelihood estimation

We rely on the LASSO (Tibshirani, 1996) as a successful procedure to encourage sparse models in high-dimensional linear regression based on an ℓ_1-penalty, and include it in this mixture of experts modeling framework for functional data. The ℓ_1-regularized ME models have demonstrated their performance from a computational point of view (Chamroukhi and Huynh, 2019; Huynh and Chamroukhi, 2019a) and enjoy good theoretical properties (Nguyen et al., 2020, 2021b).

3.2.7 ℓ_1-regularization and the EM-Lasso algorithm

We propose an ℓ_1-regularization of the observed-data log-likelihood (3.18) to be maximized, along with coordinate ascent algorithms to deal with the high-dimensional setting when updating the parameters within the resulting EM-Lasso algorithm. The objective function in this case is given by the following ℓ_1-regularized observed-data log-likelihood,

$$\mathcal{L}(\boldsymbol{\Psi}) = \log L(\boldsymbol{\Psi}) - \text{Pen}_{\lambda,\chi}(\boldsymbol{\Psi}), \tag{3.18}$$

where $\log L(\boldsymbol{\Psi})$ is the observed-data log-likelihood of $\boldsymbol{\Psi}$ defined by (3.14), and $\text{Pen}_{\lambda,\chi}(\boldsymbol{\Psi})$ is a LASSO regularization term encouraging sparsity for the expert and the gating network parameters, defined by

$$\text{Pen}_{\lambda,\chi}(\boldsymbol{\Psi}) = \lambda \sum_{k=1}^{K} \sum_{j=1}^{p} |\eta_{k,j}| + \chi \sum_{k=1}^{K-1} \sum_{j=1}^{q} |\zeta_{k,j}|, \tag{3.19}$$

where λ and χ are positive real values representing tuning parameters. The maximization of (3.18) cannot be performed in a closed form but again the EM algorithm can be adapted to iteratively solve the maximization problem. The resulting algorithm for the FME model, called EM-Lasso, takes the following form, as detailed in Appendix B.2. After starting with an initial solution $\boldsymbol{\Psi}^{(0)}$, it alternates between the two following steps, until convergence (when there is no longer a significant change in the values of the ℓ_1-penalized log-likelihood (3.18)).

■ **E-step.** The E-Step in this EM-Lasso algorithm is unchanged compared to the previously presented EM algorithm, and only requires the computation of the conditional expert memberships $\tau_{ik}^{(s)}$ according to (3.15).

■ **M-step.** In this regularized MLE context, the parameter vector $\boldsymbol{\Psi}$ is now updated by maximizing the regularized Q-function (B.5), *i.e.*, $\boldsymbol{\Psi}^{(s+1)} = \arg\max_{\boldsymbol{\Psi}} \left\{ Q(\boldsymbol{\Psi}; \boldsymbol{\Psi}^{(s)}) - \text{Pen}_{\lambda,\chi}(\boldsymbol{\Psi}) \right\}$. This is performed by separate maximizations w.r.t the gating network parameters $\boldsymbol{\xi}$ and, for each expert k ($k \in [K]$), w.r.t the expert network parameters $\boldsymbol{\theta}_k$.

Updating the gating network parameters. At iteration s of the EM-Lasso algorithm consists of maximizing the following regularized Q-function w.r.t $\boldsymbol{\xi}$,

$$\mathcal{Q}_{\chi}(\boldsymbol{\xi}; \boldsymbol{\Psi}^{(s)}) = Q(\boldsymbol{\xi}; \boldsymbol{\Psi}^{(s)}) - \chi \sum_{k=1}^{K-1} \|\boldsymbol{\zeta}_k\|_1, \tag{3.20}$$

where $Q(\boldsymbol{\xi}; \boldsymbol{\Psi}^{(s)}) = \sum_{i=1}^{n} \left[\sum_{k=1}^{K-1} \tau_{ik}^{(s)} \left(\alpha_{k,0} + \boldsymbol{\zeta}_k^\top \mathbf{r}_i \right) - \log \left(1 + \sum_{k'=1}^{K-1} \exp\{\alpha_{k',0} + \boldsymbol{\zeta}_{k'}^\top \mathbf{r}_i\} \right) \right]$. One can see this is equivalent to solving a weighted regularized multinomial logistic regression problem for which $\mathcal{Q}_{\chi}(\boldsymbol{\xi}; \boldsymbol{\Psi}^{(s)})$ is its penalized log-likelihood, the weights being the conditional probabilities $\tau_{ik}^{(s)}$. There is no closed-form solution for this kind of problem. We then use an iterative optimization algorithm to seek for a maximizer of $\mathcal{Q}_{\chi}(\boldsymbol{\xi}; \boldsymbol{\Psi}^{(s)})$, *i.e.*, an update for the parameters of the gating

network. To be effective when the number of parameters to estimate is high, we propose a coordinate ascent algorithm to update the softmax gating network coefficients in this regularized context.

Coordinate ascent for updating the gating network. The idea of the coordinate ascent algorithm (eg. see Hastie et al. (2015), Huynh and Chamroukhi (2019a)), implemented in our context at the sth EM-Lasso iteration to maximize $\mathcal{Q}_\chi(\boldsymbol{\xi}; \boldsymbol{\Psi}^{(s)})$ at the M-Step, is as follows. The gating function parameter vectors $\boldsymbol{\xi}_k = (\alpha_{k,0}, \boldsymbol{\zeta}_k^\top)^\top$ as components of the whole gating network parameters $\boldsymbol{\xi} = (\boldsymbol{\xi}_1^\top, \ldots, \boldsymbol{\xi}_{K-1}^\top)^\top$, are updated one at a time, while fixing the other gate's parameters to their previous estimates. Furthermore, to update a single gating parameter vector $\boldsymbol{\xi}_k$, we only update its coefficients ξ_{kj} one at a time, while fixing the other coefficients to their previous values. More precisely, for each single gating function k, we partially approximate the smooth part of $\mathcal{Q}_\chi(\boldsymbol{\xi}; \boldsymbol{\Psi}^{(s)})$ with respect to $\boldsymbol{\xi}_k$ at the current EM-Lasso estimate, say $\boldsymbol{\xi}^{(t)}$, then optimize the resulting objective function (with respect to $\boldsymbol{\xi}_k$). This corresponds to solving penalized weighted least squares problems using coordinate ascent. Thus, this results into an inner loop, indexed by m, that cycles over the components of $\boldsymbol{\xi}_k$ and updates them one by one, while the others are kept fixed to their previous values, *i.e.*, $\xi_{kh}^{(m+1)} = \xi_{kh}^{(m)}$ for all $h \neq j$, until the objective function (B.7) is not significantly increased.

The obtained closed form updates for each coefficient ζ_{kj}, $j \in [q]$, and for the intercept $\alpha_{k,0}$, are as follows

$$\zeta_{kj}^{(m+1)} = \frac{\mathcal{S}_\chi\Big(\sum_{i=1}^n w_{ik} \mathrm{r}_{ij}(c_{ik} - \tilde{c}_{ikj}^{(m)})\Big)}{\sum_{i=1}^n w_{ik} \mathrm{r}_{ij}^2} \text{ for } j \in [q], \quad \alpha_{k,0}^{(m+1)} = \frac{\sum_{i=1}^n w_{ik}(c_{ik} - \mathbf{r}_i^\top \boldsymbol{\zeta}_k^{(m+1)})}{\sum_{i=1}^n w_{ik}},$$

where $\tilde{c}_{ikj}^{(m)} = \alpha_{k0}^{(m)} + \mathbf{r}_i^\top \boldsymbol{\zeta}_k^{(m)} - \zeta_{kj}^{(m)} \mathrm{r}_{ij}$ is the fitted value excluding the contribution from the jth component of the ith vector r_{ij} in the design matrix of the gating network and $\mathcal{S}_\chi(\cdot)$ is a soft-thresholding operator defined by $\mathcal{S}_\chi(u) = \mathrm{sign}(u)(|u| - \chi)_+$ and $(v)_+$ a shorthand for $\max\{v, 0\}$. The values $(\alpha_{k,0}^{(m+1)}, \boldsymbol{\zeta}_k^{(m+1)})$ obtained at convergence of the coordinate ascent inner loop for the kth gating function are taken as the fixed values of that gating function, in the procedure of updating the next parameter vector $\boldsymbol{\xi}_{k+1}$. Finally, when all the gating functions have their values updated, *i.e.*, after $K-1$ inner coordinate ascent loops, to avoid numerical instability, we perform a backtracking line search, before actually updating the gating network's parameters for the next EM-Lasso iteration. More precisely, the update is $\boldsymbol{\xi}^{(t+1)} = (1-\nu)\boldsymbol{\xi}^{(t)} + \nu\bar{\boldsymbol{\xi}}^{(t)}$, where $\bar{\boldsymbol{\xi}}^{(t)}$ is the output after $K-1$ inner loops and ν is backtrackingly determined to ensure $\mathcal{Q}_\chi(\boldsymbol{\xi}^{(t+1)}; \boldsymbol{\Psi}^{(s)}) \geqslant \mathcal{Q}_\chi(\boldsymbol{\xi}^{(t)}; \boldsymbol{\Psi}^{(s)})$.

We keep cycling these updated iterates for the parameter vectors $\boldsymbol{\xi}_k$, until convergence of the whole coordinate ascent (CA) procedure inside the M-Step, *i.e.*, when the relative increase in the Lasso-regularized objective $\mathcal{Q}_\chi(\boldsymbol{\xi}; \boldsymbol{\Psi}^{(s)})$ related to the softmax gating network is not significant, e.g., less than 10^{-6}. Then, the next EM-Lasso iteration is calculated with the updated gating network's parameters $\boldsymbol{\xi}^{(s+1)} = (\widetilde{\alpha}_{1,0}, \widetilde{\boldsymbol{\zeta}}_1^\top, \ldots, \widetilde{\alpha}_{K-1,0}, \widetilde{\boldsymbol{\zeta}}_{K-1}^\top)^\top$ where the values $\widetilde{\alpha}_{k,0}$ and $\widetilde{\zeta}_{kj}$ for all $k \in [K-1], j \in [q]$ are those obtained for the $\alpha_{k,0}$'s and the ζ_{kj}'s at convergence of the CA algorithm.

Updating the experts network parameters. The maximization step for updating the expert parameters $\boldsymbol{\theta}_k$ consists of maximizing the function $\mathcal{Q}_\lambda(\boldsymbol{\theta}_k; \boldsymbol{\Psi}^{(s)})$ given by

$$\mathcal{Q}_\lambda(\boldsymbol{\theta}_k; \boldsymbol{\Psi}^{(s)}) \quad = \quad Q(\boldsymbol{\theta}_k; \boldsymbol{\Psi}^{(s)}) - \lambda \|\boldsymbol{\eta}_k\|_1, \tag{3.21}$$

where $Q(\boldsymbol{\theta}_k; \boldsymbol{\Psi}^{(s)}) = -\frac{1}{2\sigma_k^2} \sum_{i=1}^n \tau_{ik}^{(s)} \left(y_i - (\beta_{k,0} + \boldsymbol{\eta}_k^\top \mathbf{x}_i)\right)^2 - \frac{n}{2} \log(2\pi\sigma_k^2)$. This corresponds to solving a weighted LASSO problem where the weights are the conditional experts memberships $\tau_{ik}^{(s)}$. We then solve it by an iterative optimization algorithm similarly to the previous case of updating the gating network parameters. As it can be seen in Appendix B.2.2, updating $(\beta_{k,0}, \boldsymbol{\eta}_k)$ according to (3.21) is obtained by coordinate ascent as follows. For each $j \in [p]$, the closed-form update for η_{kj} at the mth iteration of the coordinate ascent algorithm within the M-Step of EM-Lasso, is given by

$$\eta_{kj}^{(m+1)} = \frac{\mathcal{S}_{\lambda\sigma_k^{2(s)}}\left(\sum_{i=1}^n \tau_{ik}^{(s)} \mathrm{x}_{ij}(y_i - \tilde{y}_{ij}^{(m)})\right)}{\sum_{i=1}^n \tau_{ik}^{(s)} \mathrm{x}_{ij}^2}, \quad \beta_{k,0}^{(m+1)} = \frac{\sum_{i=1}^n \tau_{ik}^{(s)}(y_i - \mathbf{x}_i^\top \boldsymbol{\eta}_k^{(m+1)})}{\sum_{i=1}^n \tau_{ik}^{(s)}}$$

in which $\tilde{y}_{ij}^{(m)} = \beta_{k,0}^{(m)} + \mathbf{x}_i^\top \boldsymbol{\eta}_k^{(m)} - \eta_{kj}^{(m)} \mathrm{x}_{ij}$ is the fitted value excluding the contribution from x_{ij} and $\mathcal{S}_{\lambda\sigma_k^{2(s)}}(\cdot)$ is the soft-thresholding operator. We keep updating the components of $(\beta_{k,0}, \boldsymbol{\eta}_k)$ cyclically until no enough increase in objective function (3.21). Then, once $(\beta_{k,0}, \boldsymbol{\eta}_k)$ are updated while fixing the variance σ_k^2, the latter is then updated straightforwardly as in the case of standard weighted Gaussian regression, and it is update is given by

$$\sigma_k^{2(s+1)} = \frac{\sum_{i=1}^n \tau_{ik}^{(s)} \left(y_i - \beta_{k,0}^{(s+1)} - \mathbf{x}_i^\top \boldsymbol{\eta}_k^{(s+1)}\right)^2}{\sum_{i=1}^n \tau_{ik}^{(s)}},$$

where $(\beta_{k,0}^{(s+1)}, \boldsymbol{\eta}_k^{(s+1)}) = (\widetilde{\beta}_{k,0}, \widetilde{\boldsymbol{\eta}}_k)$ is the solution obtained at convergence of the CA algorithm, which is taken as the update in the next EM-Lasso iteration.

This completes the parameter vector update $\boldsymbol{\Psi}^{(s+1)} = (\boldsymbol{\xi}^{(s+1)}, \boldsymbol{\theta}_1^{(s+1)}, \ldots, \boldsymbol{\theta}_K^{(s+1)})$ of the regularized FME model, where $\boldsymbol{\xi}^{(s+1)}$ and $\boldsymbol{\theta}_k^{(s+1)}, k \in [K]$, are, respectively, the updates of the gating network parameters and the experts network parameters, calculated by the coordinate ascent algorithms.

The EM-Lasso algorithm provides an estimate of the FME parameters with sparsity constraints on the values of the parameter vectors $\boldsymbol{\xi}$ and $\boldsymbol{\theta}_k$, $k \in [K]$. Actually, since here these parameter vectors do not rely directly the original functional inputs, to the output, assuming some of their values is zero is not easily interpretable, compared to the sparsity in vectorial generalized linear models, mixture of regressions and ME models.

From now on, we refer to FME and FME-Lasso, respectively, the FME model fitted by EM algorithm in **Section 3.2.5** and the regularized FME model fitted by EM-Lasso algorithm, in **Section 3.2.7**. In the following section, we introduce a regularization that is interpretable and encourages sparsity in the FME model.

3.3 Interpretable Functional Mixture of Experts (iFME)

In this section, we provide a sparse and, especially highly-interpretable fit, for the coefficient functions $\{\beta_k(t), t \in \mathcal{T}\}$ and $\{\alpha_k(t), t \in \mathcal{T}\}$ representing each of the K functional experts and gating network. We call our approach Interpretable Functional Mixture of Experts (iFME). The presented iFME allows us to control the expert and gating parameter functions while still providing performance as with the standard FME model presented previously.

3.3.1 Motivation

We rely on the methodology of Functional Linear Regression That's Interpretable (FLiRTI) presented in James et al. (2009). The idea of the FLiRTI methodology is as follows. We use variable selection with sparsity assumption on appropriate chosen derivatives of the coefficient function, say $\beta_{z_i}(t)$ here, in the case of the functional expert network, to produce a highly interpretable estimate for the coefficient functions $\beta_{z_i}(t)$. For instance, $\beta_{z_i}^{(0)}(t) = 0$ implies that the predictor $X_i(t)$ has no effect on the response Y_i at t, $\beta_{z_i}^{(1)}(t) = 0$ means that $\beta_{z_i}(t)$ is constant in t, $\beta_{z_i}^{(2)}(t) = 0$ shows that $\beta_{z_i}(t)$ is linear in t, *etc.* Assuming sparsity in higher-order derivatives of $\beta_{z_i}(t)$, for instance when $d = 3$ or $d = 4$, will however give us a less easily interpretable fit. Hence, for example, if one believes that the expert parameter function $\beta_{z_i}(t)$ is exactly zero over a certain region and exactly linear over other region of t, then it is necessary to estimate $\beta_{z_i}(t)$ such that $\beta_{z_i}^{(0)}(t) = 0$ and $\beta_{z_i}^{(2)}(t) = 0$ over those regions, respectively. In this situation, we need to model $\beta_{z_i}(t)$ assuming that its zeroth and second derivatives are sparse. However, with the EM-Lasso method derived above via the Lasso regularization, there is no actually any reason that we could obtain those desired properties, which may result in an estimate for $\beta_{z_i}(t)$ that is rarely exactly zeros (and/or linear), and making the sparsity and coefficient curves hard to interpret. The same situation may occur with the gating parameter functions. To handle this, we describe in what follows the construction of our iFME model that produces flexible-shape and highly-interpretable estimates for the expert and gating coefficient functions, by simultaneously constraining any two of their derivatives to be sparse.

3.3.2 Interpretable sparse regularization

We start by selecting a p-dimensional basis $\boldsymbol{b}_p(t)$ and a q-dimensional basis $\boldsymbol{b}_q(t)$ for approximating the experts and gating networks, respectively. For the expert network, if we divide the time domain into a grid of p evenly spaced points, and let D^d be the dth finite difference operator

defined recursively as

$$\begin{aligned} D\boldsymbol{b}_p(t_j) &= p\left[\boldsymbol{b}_p(t_j) - \boldsymbol{b}_p(t_{j-1})\right], \\ D^2\boldsymbol{b}_p(t_j) &= D\left[D\boldsymbol{b}_p(t_j)\right] = p^2[\boldsymbol{b}_p(t_j) - 2\boldsymbol{b}_p(t_{j-1}) + \boldsymbol{b}_p(t_{j-2})\big], \\ &\vdots \\ D^d\boldsymbol{b}_p(t_j) &= D\left[D^{d-1}\boldsymbol{b}_p(t_j)\right], \end{aligned}$$

then $D^d\boldsymbol{b}_p(t_j)$ provides an approximation for $\boldsymbol{b}_p^{(d)}(t_j) = [b_1^{(d)}(t_j), \ldots, b_p^{(d)}(t_j)]^\top$, $j \in [p]$.

Let

$$\mathbf{A}_p = \left[D^{d_1}\boldsymbol{b}_p(t_1), D^{d_1}\boldsymbol{b}_p(t_2), \ldots, D^{d_1}\boldsymbol{b}_p(t_p), D^{d_2}\boldsymbol{b}_p(t_1), D^{d_2}\boldsymbol{b}_p(t_2), \ldots, D^{d_2}\boldsymbol{b}_p(t_p)\right]^\top \quad (3.22)$$

be the matrix of approximate d_1th and d_2th derivative of the basis $\boldsymbol{b}_p(t)$. We denote $\mathbf{A}_p^{[d_1]}$ the first p rows of $\mathbf{A}_p$ and $\mathbf{A}_p^{[d_2]}$ the remainder, *i.e.*, $\mathbf{A}_p = [\mathbf{A}_p^{[d_1]}\mathbf{A}_p^{[d_2]}]^\top$. One can see such matrix $\mathbf{A}_p$ is in $\mathbb{R}^{2p\times p}$ and $\mathbf{A}_p^{[d_1]}$ is a square invertible matrix. Similarly, let $\mathbf{A}_q = [\mathbf{A}_q^{[d_1]}\mathbf{A}_q^{[d_2]}]^\top \in \mathbb{R}^{2q\times q}$ be the corresponding matrix defined for the gating network.

Now, if we consider the following representation for the expert network coefficient function

$$\boldsymbol{\gamma}_{z_i} = \mathbf{A}_p\boldsymbol{\eta}_{z_i} \quad (3.23)$$

with $\boldsymbol{\gamma}_{z_i} = ({\boldsymbol{\gamma}_{z_i}^{[d_1]}}^\top, {\boldsymbol{\gamma}_{z_i}^{[d_2]}}^\top)^\top$, where $\boldsymbol{\gamma}_{z_i}^{[d_1]} = (\gamma_{1,z_i}^{[d_1]}, \ldots, \gamma_{p,z_i}^{[d_1]})^\top$ and $\boldsymbol{\gamma}_{z_i}^{[d_2]} = (\gamma_{1,z_i}^{[d_2]}, \ldots, \gamma_{p,z_i}^{[d_2]})^\top$, and $\boldsymbol{\eta}_{z_i}$ defined as in relation to $\beta_{z_i}(t)$ in (3.7), then $\boldsymbol{\gamma}_{z_i}^{[d_1]}$ (resp. $\boldsymbol{\gamma}_{z_i}^{[d_2]}$) provides an approximation to $\beta_{z_i}^{(d_1)}(t)$, the d_1th derivative of $\beta_{z_i}(t)$ (resp. $\beta_{z_i}^{(d_2)}(t)$, the d_2th derivative of $\beta_{z_i}(t)$). Hence, enforcing sparsity in $\boldsymbol{\gamma}_{z_i}$ constrains $\beta_{z_i}^{(d_1)}(t)$ and $\beta_{z_i}^{(d_2)}(t)$ to be zero at most time points. Similarly, if we consider the following representation for the gating network coefficient function

$$\boldsymbol{\omega}_{z_i} = \mathbf{A}_q\boldsymbol{\zeta}_{z_i} \quad (3.24)$$

with $\boldsymbol{\omega}_{z_i} = ({\boldsymbol{\omega}_{z_i}^{[d_1]}}^\top, {\boldsymbol{\omega}_{z_i}^{[d_2]}}^\top)^\top$, where $\boldsymbol{\omega}_{z_i}^{[d_1]} = (\omega_{1,z_i}^{[d_1]}, \ldots, \omega_{q,z_i}^{[d_1]})^\top$, $\boldsymbol{\omega}_{z_i}^{[d_2]} = (\omega_{1,z_i}^{[d_2]}, \ldots, \omega_{q,z_i}^{[d_2]})^\top$, and $\boldsymbol{\zeta}_{z_i}$ defined as in relation to $\alpha_{z_i}(t)$ in (3.8), then we can derive the same interpretation for the coefficient vector $\boldsymbol{\omega}_{z_i}$ and the gating parameter function $\alpha_{z_i}(t)$. The constraints (3.23) and (3.24) imply

$$\boldsymbol{\eta}_{z_i} = {\mathbf{A}_p^{[d_1]}}^{-1}\boldsymbol{\gamma}_{z_i}^{[d_1]}, \qquad \boldsymbol{\gamma}_{z_i}^{[d_2]} = \mathbf{A}_p^{[d_2]}{\mathbf{A}_p^{[d_1]}}^{-1}\boldsymbol{\gamma}_{z_i}^{[d_1]},$$

and

$$\boldsymbol{\zeta}_{z_i} = {\mathbf{A}_q^{[d_1]}}^{-1}\boldsymbol{\omega}_{z_i}^{[d_1]}, \qquad \boldsymbol{\omega}_{z_i}^{[d_2]} = \mathbf{A}_q^{[d_2]}{\mathbf{A}_q^{[d_1]}}^{-1}\boldsymbol{\omega}_{z_i}^{[d_1]},$$

respectively.

In fact, one can construct $\mathbf{A}_p$ and $\mathbf{A}_q$ with only one derivative. Then the constraints involved to the d_2th derivative will be eliminated making the estimation easier, but also limiting the flexibility in the shapes of the functions. That is why in this construction and in our experimental studies,

$\mathbf{A}_p$ and $\mathbf{A}_q$ are constructed with multiple derivatives in order to produce curves of $\beta_{z_i}(\cdot)$ and $\alpha_{z_i}(\cdot)$ with such many desired properties.

3.3.3 The iFME model

Combining the stochastic representation of the FME model in (3.11) for the experts model and the linear predictor definition in (3.9), we obtain the following iFME model construction,

$$Y_i|u_i(\cdot) = \beta_{z_i,0} + {\boldsymbol{\gamma}_{z_i}^{[d_1]}}^\top \mathbf{v}_i + \varepsilon_i^\star, \tag{3.25}$$

$$h_{z_i}(u_i(\cdot);\boldsymbol{\alpha}) = \alpha_{z_i,0} + {\boldsymbol{\omega}_{z_i}^{[d_1]}}^\top \mathbf{s}_i, \tag{3.26}$$

subject to

$$\boldsymbol{\gamma}_{z_i}^{[d_2]} = \mathbf{A}_p^{[d_2]} {\mathbf{A}_p^{[d_1]}}^{-1} \boldsymbol{\gamma}_{z_i}^{[d_1]} \quad \text{and} \quad \boldsymbol{\omega}_{z_i}^{[d_2]} = \mathbf{A}_q^{[d_2]} {\mathbf{A}_q^{[d_1]}}^{-1} \boldsymbol{\omega}_{z_i}^{[d_1]}, \tag{3.27}$$

where $\mathbf{v}_i = ({\mathbf{A}_p^{[d_1]}}^{-1})^\top \mathbf{x}_i$ is the new design vector for the experts and $\mathbf{s}_i = ({\mathbf{A}_q^{[d_1]}}^{-1})^\top \mathbf{r}_i$ the new one for the gating network. Hence, from (3.25) and (3.26), the conditional density of each expert and the gating network are now written as

$$f(y_i|u_i(\cdot);\boldsymbol{\psi}_k) = \phi(y_i; \beta_{k,0} + {\boldsymbol{\gamma}_k^{[d_1]}}^\top \mathbf{v}_i, \sigma_k^2) \tag{3.28}$$

and

$$\pi_k(\mathbf{s}_i;\mathbf{w}) = \frac{\exp\left\{\alpha_{k,0} + {\boldsymbol{\omega}_k^{[d_1]}}^\top \mathbf{s}_i\right\}}{1 + \sum_{k'=1}^{K-1} \exp\left\{\alpha_{k',0} + {\boldsymbol{\omega}_{k'}^{[d_1]}}^\top \mathbf{s}_i\right\}}, \tag{3.29}$$

where $\boldsymbol{\psi}_k = (\beta_{k,0}, {\boldsymbol{\gamma}_k^{[d_1]}}^\top, \sigma_k^2)^\top$ is the unknown parameter vector of expert component density k and $\mathbf{w} = \left(\alpha_{1,0}, {\boldsymbol{\omega}_1^{[d_1]}}^\top, \ldots, \alpha_{K-1,0}, {\boldsymbol{\omega}_{K-1}^{[d_1]}}^\top\right)^\top$, with $(\alpha_{K,0}, {\boldsymbol{\omega}_K^{[d_1]}}^\top)^\top$ a null vector, is the unknown parameter vector of the gating network. Finally, gathering (3.28) and (3.29) as for (3.13), the iFME model density is now given by

$$f(y_i|u_i(\cdot);\boldsymbol{\Psi}) = \sum_{k=1}^{K} \pi_k(\mathbf{s}_i;\mathbf{w}) \phi(y_i; \beta_{k,0} + {\boldsymbol{\gamma}_k^{[d_1]}}^\top \mathbf{v}_i, \sigma_k^2), \tag{3.30}$$

where $\boldsymbol{\Psi} = (\mathbf{w}^\top, \boldsymbol{\psi}_1^\top, \ldots, \boldsymbol{\psi}_K^\top)^\top$ is the parameter vector of the model to be estimated. Thus, the iFME model constructed upon the parameter vectors $\boldsymbol{\gamma}_k$'s and $\boldsymbol{\omega}_k$'s, for which the sparsity is assumed to obtain interpretable estimates.

3.3.4 Regularized MLE via the EM-iFME algorithm

In order to fit the iFME model and to maintain the sparsity in $\boldsymbol{\gamma}_k$ and $\boldsymbol{\omega}_k$, the following EM-iFME algorithm is then developed to maximize the penalized log-likelihood function

$$\mathcal{L}(\boldsymbol{\Psi}) = \sum_{i=1}^{n} \log f(y_i|u_i(\cdot);\boldsymbol{\Psi}) + \text{Pen}_{\lambda,\chi}(\boldsymbol{\Psi}) \tag{3.31}$$

with the conditional iFME density $f(y_i|u_i(\cdot);\boldsymbol{\Psi})$ is defined in (3.30) and the new sparse and interpretable regularization term is given by

$$\text{Pen}_{\lambda,\chi}(\boldsymbol{\Psi}) = \lambda \sum_{k=1}^{K} (\|\boldsymbol{\gamma}_k^{[d_1]}\|_1 + \rho\|\boldsymbol{\gamma}_k^{[d_2]}\|_1) + \chi \sum_{k=1}^{K-1} (\|\boldsymbol{\omega}_k^{[d_1]}\|_1 + \varrho\|\boldsymbol{\omega}_k^{[d_2]}\|_1), \tag{3.32}$$

where ρ and ϱ are, respectively, the weights associated to the d_2th derivative of the expert and the gating parameter function. The appearance of the weighting parameters ρ and ϱ, besides the usual regularization parameters λ and χ, is motivated by the fact that one may wish to place a greater emphasis on sparsity in the d_2th derivative than in the d_1th derivative of the parameter functions, or vice versa. In practice, the selection of ρ and ϱ is more about whether they are greater than or less than one, (*i.e.*, the emphasis on d_2th) rather than select an exact value. It is worth noting two points: first, unlike the previous FME-Lasso, in iFME model the regularization operates on the functional derivative $\boldsymbol{\gamma}_k$'s rather than the functional coefficients $\boldsymbol{\eta}_k$'s for the experts, and on the functional derivatives $\boldsymbol{\omega}_k$'s rather than the functional coefficients $\boldsymbol{\zeta}_k$'s for the gating network; and second, maximizing the penalized log-likelihood function (3.31) with penalization in (3.32) in iFME model must be coupled with the constrains (3.27). Follows are the two steps of the proposed EM-iFME algorithm.

■ **E-Step.** The E-Step for the new iFME model calculates for each observation the conditional probability memberships of each expert k

$$\tau_{ik}^{(s)} = \mathbb{P}(Z_i = k|y_i, u_i(\cdot);\boldsymbol{\Psi}^{(s)}) = \frac{\pi_k(\mathbf{s}_i;\mathbf{w}^{(s)})\,\phi(y_i;\beta_{k,0}^{(s)} + \mathbf{v}_i^\top \boldsymbol{\gamma}_k^{[d_1](s)}, \sigma_k^{2(s)})}{f(y_i|u_i(\cdot);\boldsymbol{\Psi}^{(s)})}, \tag{3.33}$$

where $f(y_i|u_i(\cdot);\boldsymbol{\Psi}^{(s)})$ is now calculated according to the iFME density given by (3.30).

■ **M-Step.** The maximization is performed by separate maximizations w.r.t the gating network parameters $\mathbf{w}$ and the experts network parameters $\boldsymbol{\psi}_k$'s.

Updating the gating network parameters. The maximization step for updating the gating network parameters $\mathbf{w} = \left(\alpha_{1,0}, \boldsymbol{\omega}_1^{[d_1]\top}, \ldots, \alpha_{K-1,0}, \boldsymbol{\omega}_{K-1}^{[d_1]\top}\right)^\top$ consists of maximizing the function

$\mathcal{Q}_\chi(\mathbf{w}; \boldsymbol{\Psi}^{(s)})$ given by

$$\mathcal{Q}_\chi(\mathbf{w}; \boldsymbol{\Psi}^{(s)}) = Q(\mathbf{w}; \boldsymbol{\Psi}^{(s)}) - \chi \sum_{k=1}^{K-1} (\|\boldsymbol{\omega}_k^{[d_1]}\|_1 + \varrho \|\boldsymbol{\omega}_k^{[d_2]}\|_1), \tag{3.34}$$

subject to

$$\boldsymbol{\omega}_k^{[d_2]} = \mathbf{A}_q^{[d_2]} {\mathbf{A}_q^{[d_1]}}^{-1} \boldsymbol{\omega}_k^{[d_1]}, \quad \forall k \in [K-1], \tag{3.35}$$

where $Q(\mathbf{w}; \boldsymbol{\Psi}^{(s)}) = \sum_{i=1}^n \left[\sum_{k=1}^{K-1} \tau_{ik}^{(s)} \left(\alpha_{k,0} + {\boldsymbol{\omega}_k^{[d_1]}}^\top \mathbf{s}_i \right) - \log \left(1 + \sum_{k'=1}^{K-1} \exp\{\alpha_{k',0} + {\boldsymbol{\omega}_{k'}^{[d_1]}}^\top \mathbf{s}_i\} \right) \right] \cdot$ This is a constrained version of the weighted regularized multinomial logistic regression problem, where the weights are the conditional probabilities $\tau_{ik}^{(s)}$.

To solve it, in the same spirit as when updating the gating network in the previous EM-Lasso algorithm, we use an outer loop that cycles over the gating function parameters to update them one by one. However, to update a single gating function parameter $\mathbf{w}_k = (\alpha_{k,0}, {\boldsymbol{\omega}_k^{[d_1]}}^\top)^\top$, $k \in [K-1]$, since the maximization problem (3.34) is now coupled with an additional constraint (3.35), rather than using a coordinate ascent algorithm as in EM-Lasso, we use the following alternative approach. For each single gating network k, using a Taylor series expansion, we partially approximate the smooth part of $\mathcal{Q}_\chi(\mathbf{w}; \boldsymbol{\Psi}^{(s)})$ defined in (3.34) w.r.t. $\mathbf{w}_k$ at the current estimate $\mathbf{w}^{(t)}$, then maximize the resulting objective function (w.r.t. $\mathbf{w}_k$), subject to the corresponding constraint (w.r.t. k) in (3.35). It corresponds to solving the following penalized weighted least squares problem with constraints,

$$\begin{aligned} \max_{(\alpha_{k,0},\, \boldsymbol{\omega}_k^{[d_1]}, \boldsymbol{\omega}_k^{[d_2]})} & -\frac{1}{2} \sum_{i=1}^n w_{ik} (c_{ik} - \alpha_{k,0} - \mathbf{s}_i^\top \boldsymbol{\omega}_k^{[d_1]})^2 - \chi(\|\boldsymbol{\omega}_k^{[d_1]}\|_1 + \varrho \|\boldsymbol{\omega}_k^{[d_2]}\|_1) \\ \text{subject to} \quad & \boldsymbol{\omega}_k^{[d_2]} = \mathbf{A}_q^{[d_2]} {\mathbf{A}_q^{[d_1]}}^{-1} \boldsymbol{\omega}_k^{[d_1]}, \end{aligned} \tag{3.36}$$

where $w_{ik} = \pi_k(\mathbf{w}^{(t)}; \mathbf{s}_i) \left(1 - \pi_k(\mathbf{w}^{(t)}; \mathbf{s}_i)\right)$ are the weights and $c_{ik} = \alpha_{k,0}^{(t)} + \mathbf{s}_i^\top {\boldsymbol{\omega}_k^{[d_1]}}^{(t)} + \frac{\tau_{ik}^{(s)} - \pi_k(\mathbf{w}^{(t)}; \mathbf{s}_i)}{w_{ik}}$ are the working responses computed given the current estimate $\mathbf{w}^{(t)}$. This problem can be solved by quadratic programming (see, *e.g.*, Gaines et al. (2018)) or by using the Dantzig selector (Candes et al., 2007), which we opt to use in our experimental studies. The details of using Dantzig selector to solve problem (3.36) are given in Appendix B.3.1.

Therefore, if $(\widetilde{\alpha}_{k,0}, \widetilde{\boldsymbol{\omega}}_k^{[d_1]}, \widetilde{\boldsymbol{\omega}}_k^{[d_2]})$ is an optimal solution to problem (3.36), then $\widetilde{\mathbf{w}}_k = (\widetilde{\alpha}_{k,0}, {\widetilde{\boldsymbol{\omega}}_k^{[d_1]}}^\top)^\top$ is taken as an update for the gating parameter vector $\mathbf{w}_k$. We keep cycling over $k \in [K-1]$ until there is no significant increase in the regularized $Q-$function (3.34).

Updating the experts network parameters. The maximization step for updating the expert parameter vector $\boldsymbol{\psi}_k = (\beta_{k,0}, {\boldsymbol{\gamma}_k^{[d_1]}}^\top, \sigma_k^2)^\top$ consists of solving the following problem:

$$\begin{aligned} \max_{(\beta_{k,0},\, \boldsymbol{\gamma}_k^{[d_1]}, \boldsymbol{\gamma}_k^{[d_2]}, \sigma_k^2)} \quad & \sum_{i=1}^{n} \tau_{ik}^{(s)} \log \phi(y_i; \beta_{k,0} + \mathbf{v}_i^\top \boldsymbol{\gamma}_k^{[d_1]}, \sigma_k^2) - \lambda(\|\boldsymbol{\gamma}_k^{[d_1]}\|_1 + \rho \|\boldsymbol{\gamma}_k^{[d_2]}\|_1) \\ \text{subject to} \quad & \boldsymbol{\gamma}_k^{[d_2]} = \mathbf{A}_p^{[d_2]} {\mathbf{A}_p^{[d_1]}}^{-1} \boldsymbol{\gamma}_k^{[d_1]}. \end{aligned} \tag{3.37}$$

As in the previous EM-Lasso algorithm, we first fix σ_k^2 to its previous estimate and perform the update for $(\beta_{k,0}, \boldsymbol{\gamma}_k^{[d_1]})$, which corresponds to solving a penalized weighted least squares problem with constraints. This is be performed by using the Dantzig selector, in the same manner as previously for solving problem (3.36). The corresponding technical details can be found in Appendix B.3.2.

Once the $(\beta_{k,0}, \boldsymbol{\gamma}_k^{[d_1]})$ are updated, the straightforward update for the variance σ_k^2 is given by the standard estimate of a weighted Gaussian regression. More specifically, let $(\widetilde{\beta}_{k,0}, \widetilde{\boldsymbol{\gamma}}_k^{[d_1]}, \widetilde{\boldsymbol{\gamma}}_k^{[d_2]})$ be the solution to the problem (3.37) (with σ_k^2 fixed to $\sigma_k^{2(s)}$), the updates for expert parameter vector $\boldsymbol{\psi}_k$ are given by

$$\begin{aligned} (\beta_{k,0}^{(s+1)}, {\boldsymbol{\gamma}_k^{[d_1]}}^{(s+1)}) &= (\widetilde{\beta}_{k,0}, \widetilde{\boldsymbol{\gamma}}_k^{[d_1]}), \\ \sigma_k^{2(s+1)} &= \frac{\sum_{i=1}^n \tau_{ik}^{(s)} \left(y_i - \beta_{k,0}^{(s+1)} - \mathbf{v}_i^\top {\boldsymbol{\gamma}_k^{[d_1]}}^{(s+1)} \right)^2}{\sum_{i=1}^n \tau_{ik}^{(s)}}. \end{aligned}$$

Thus, at the end of the M-Step, we obtain a parameter vector update $\boldsymbol{\Psi}^{(s+1)} = (\mathbf{w}^{(s+1)}, \boldsymbol{\psi}_1^{(s+1)}, \ldots, \boldsymbol{\psi}_K^{(s+1)})$ for the next EM iteration, where $\mathbf{w}^{(s+1)}$ and $\boldsymbol{\psi}_k^{(s+1)}$, $k \in [K]$, are, respectively, the updates of the gating network parameters and the experts network parameters, calculated by the two procedures described above. Alternating the E-Step and M-Step until convergence, *i.e.*, when there is no longer a significant change in the values of the penalized log-likelihood (3.31), leads to a penalized maximum likelihood estimate $\widehat{\boldsymbol{\Psi}}$ for $\boldsymbol{\Psi}$.

Estimating the coefficient functions. Finally, once the parameter vector of iFME model has been estimated, the coefficient functions of the gating network $\alpha_k(t)$, $k \in [K-1]$ and the ones of the experts network $\beta_k(t)$, $k \in [K]$, can be reconstructed by the following formulas,

$$\begin{aligned} \widehat{\alpha}_k(t) &= \boldsymbol{b}_q(t)^\top \mathbf{A}_q^{-1} \widehat{\boldsymbol{\omega}}_k^{[d_1]}, \\ \widehat{\beta}_k(t) &= \boldsymbol{b}_p(t)^\top \mathbf{A}_p^{-1} \widehat{\boldsymbol{\gamma}}_k^{[d_1]}, \end{aligned} \tag{3.38}$$

where $\widehat{\boldsymbol{\omega}}_k^{[d_1]}$ and $\widehat{\boldsymbol{\gamma}}_k^{[d_1]}$ are respectively the regularized maximum likelihood estimates for $\boldsymbol{\omega}_k^{[d_1]}$ and $\boldsymbol{\gamma}_k^{[d_1]}$.

3.3.5 Clustering and non-linear regression with FME models

Once the model parameters have been estimated, a soft partition of the data into K clusters, represented by the estimated posterior probabilities $\widehat{\tau}_{ik} = \mathbb{P}(Z_i = k|u_i, y_i; \widehat{\boldsymbol{\Psi}})$, is obtained. A hard partition can also be computed according to the Bayes' optimal allocation rule. That is, by assigning each curve to the component having the highest estimated posterior probability τ_{ik}, defined by (3.15) for FME or by (3.33) for the iFME model, uing the MLE $\widehat{\boldsymbol{\Psi}}$ of $\boldsymbol{\Psi}$:

$$\widehat{z}_i = \arg \max_{1 \leqslant k \leqslant K} \tau_{ik}(\widehat{\boldsymbol{\Psi}}), \quad i \in [n],$$

where $\widehat{z}_i$ denotes the estimated cluster label for the ith curve.

For the aim of functional non-linear regression, the unknown non-linear regression function with functional predictors is given by the following conditional expectation $\widehat{y}|u(\cdot) = \mathbb{E}[Y|U(\cdot); \widehat{\boldsymbol{\Psi}}]$, which is defined by

$$\widehat{y}_i|u_i(\cdot) = \sum_{k=1}^{K} \pi_k(\mathbf{r}_i; \widehat{\boldsymbol{\xi}})(\widehat{\beta}_{k,0} + \widehat{\boldsymbol{\eta}}_k^\top \mathbf{x}_i, \widehat{\sigma}_k^2)$$

for the FME model (3.13), and by

$$\widehat{y}_i|u_i(\cdot) = \sum_{k=1}^{K} \pi_k(\mathbf{s}_i; \widehat{\mathbf{w}})(\widehat{\beta}_{k,0} + {\boldsymbol{\gamma}_k^{[d_1]}}^\top \mathbf{v}_i, \widehat{\sigma}_k^2)$$

for the iFME model (3.30).

3.3.6 Tuning parameters and model selection

In practice, appropriate values of the tuning parameters should be chosen. In using FME, this cover the selection of K, the number of experts, and r, p, and q, the dimensions of B-spline bases used to approximate, respectively, the predictors, the experts, and the gating network functions, although they can be chosen to be equal. For the FME-Lasso, additionally the ℓ_1 penalty constants χ and λ in (3.19) should be chosen. For the iFME model, the tuning parameters include also d_1, d_2, the two derivatives related to the sparsity constraints, and ρ, ϱ, the weights associated to the d_2th derivative of the expert and gating functions (see (3.32)).

The selection of the tuning parameters can be performed by a cross–validation procedure with a grid search scheme to select the best combination. An alternative is to use the well-known BIC criterion (Schwarz, 1978) or, in our context, its extension, called modified BIC (Städler et al., 2010) defined as

$$\text{mBIC} = L(\widehat{\boldsymbol{\Psi}}) - \text{df}(\widehat{\boldsymbol{\Psi}})\frac{\log n}{2}, \tag{3.39}$$

where $\widehat{\boldsymbol{\Psi}}$ is the obtained log-likelihood estimator (for the FME model) or penalized log-likelihood estimator (for the FME-Lasso and iFME models), and the number of degrees of freedom $\text{df}(\widehat{\boldsymbol{\Psi}})$ is

the effective number of parameters of the model, given by

$$\mathrm{df}(\widehat{\boldsymbol{\Psi}}) = \begin{cases} \mathrm{df}(\boldsymbol{\zeta}) + (K-1) + \mathrm{df}(\boldsymbol{\eta}) + K + K & \text{for the FME and FME-Lasso models,} \\ \mathrm{df}(\boldsymbol{\omega}) + (K-1) + \mathrm{df}(\boldsymbol{\gamma}) + K + K & \text{for the iFME model,} \end{cases}$$

in which the quantities $\mathrm{df}(\boldsymbol{\zeta})$, $\mathrm{df}(\boldsymbol{\eta})$, $\mathrm{df}(\boldsymbol{\omega})$ and $\mathrm{df}(\boldsymbol{\gamma})$ are, respectively, the counts for non-zero free coefficients in the vectors $\boldsymbol{\zeta}$, $\boldsymbol{\eta}$, $\boldsymbol{\omega}$, and $\boldsymbol{\gamma}$. Note that, because of the constraints (3.27) for the iFME model, free coefficients in $\boldsymbol{\omega}$ and $\boldsymbol{\gamma}$ consist of only the part related to the d_1 derivative. That is, $\mathrm{df}(\boldsymbol{\zeta}) = \sum_{k=1}^{K-1}\sum_{j=1}^{q} \mathbf{1}_{\{\zeta_{kj}\neq 0\}}$, $\mathrm{df}(\boldsymbol{\eta}) = \sum_{k=1}^{K}\sum_{j=1}^{p} \mathbf{1}_{\{\eta_{kj}\neq 0\}}$, $\mathrm{df}(\boldsymbol{\omega}) = \sum_{k=1}^{K-1}\sum_{j=1}^{q} \mathbf{1}_{\{\omega_{kj}^{[d_1]}\neq 0\}}$ and $\mathrm{df}(\boldsymbol{\gamma}) = \sum_{k=1}^{K}\sum_{j=1}^{p} \mathbf{1}_{\{\gamma_{kj}^{[d_1]}\neq 0\}}$. We apply both the BIC and the modified BIC in our experimental study.

3.4 Experimental studies

We study the performances of the FME, FME-Lasso, and iFME models in regression and clustering problems by considering simulated scenarios and real-world data with functional predictors and scalar responses. The interests of this study consist of the prediction performance as well as the estimation of the functional parameters, *i.e.*, the expert and gating functions in the mixture-of-experts model, along with the clustering partition of the considered heterogeneous data.

3.4.1 Evaluation criteria

We will use the following criteria, for where applicable, to assess and compare the performances of the models and the related algorithms. For regression evaluation, we use the *relative predictions error* (RPE) and the *correlation* (Corr) index to quantify the relationship between the true and the predicted values of the scalar outputs. The RPE is defined by $\mathrm{RPE} = \sum_{i=1}^{n}(y_i - \widehat{y}_i)^2 / \sum_{i=1}^{n} y_i^2$, where y_i and $\widehat{y}_i$ are, respectively, the true and the predicted response of the ith observation in the testing set. For clustering evaluation, we use the *adjusted Rand index* (ARI), and the *clustering error* (ClusErr), to quantify how similar the testing observations are presented in the true partition compared to the predicted partition. To evaluate the parameters estimation performance, we compute the *mean squared error* (MSE) between the true and the estimated functional parameters. The MSE between a true function $g(\cdot)$ and its estimate $\widehat{g}(\cdot)$ is defined by

$$\mathrm{MSE}(\widehat{g}(\cdot)) = \frac{1}{m}\sum_{j=1}^{m}\left(g(t_j) - \widehat{g}(t_j)\right)^2, \tag{3.40}$$

m being the number of time points taken into account. The function $g(\cdot)$ here corresponds to an expert function $\beta(\cdot)$, or a gating function $\alpha(\cdot)$. The parameter functions are reconstructed from

the model parameters using (3.7, 3.8) for both FME and FME-Lasso models, and using (3.38) for iFME model.

The values of these criteria are averaged over N Monte Carlo runs ($N = 100$ for simulation, for the real-world data, see the corresponding section). Note that, the average over N sample replicates of $\text{MSE}(\widehat{g}(\cdot))$ is equivalent to the usual Mean Integrated Squared Error (MISE) $\text{MISE}(\widehat{g}(\cdot)) = \mathbb{E}\left[\int_{\mathcal{T}}(\widehat{g}(t) - g(t))^2 dt\right]$, where the integral here is calculated numerically by a Riemann sum over the grid $t_1, \ldots, t_m$.

3.4.2 Simulation studies

3.4.2.1 Data generating protocol

In the simulated data, the data generation protocol is as follows. We consider a K-component functional mixture of Gaussian experts model that relates a scalar response $y \in \mathbb{R}$ to a univariate functional predictor $X(t), t \in \mathcal{T}$ defined on a domain $\mathcal{T} \subset \mathbb{R}$. Given the model parameters (defined in the next paragraph) $\boldsymbol{\beta} = \{\beta_{k,0}, \beta_k(t), \sigma_k^2\}_{k=1}^K$ and $\boldsymbol{\alpha} = \{\alpha_{k,0}, \alpha_k(t)\}_{k=1}^K$, $t \in \mathcal{T}$, we first construct the functional predictors $X_i(\cdot)$ for $i \in [n]$ using the representation defined in (3.5), with detailed parameterization in (3.42). Then, for each $i \in [n]$, conditional on the functional predictor $X_i(\cdot)$, a hidden categorical random variable $Z_i \in [K]$ is generated following the multinomial distribution $\mathcal{M}\Big(1, \big(\pi_1(X_i(t); \boldsymbol{\alpha}), \ldots, \pi_K(X_i(t); \boldsymbol{\alpha})\big)\Big)$, where $\pi_k(X_i(t); \boldsymbol{\alpha})$ for $k \in [K]$ is given by (3.3). Finally, conditional on $Z_i = z_i$ and $X_i(\cdot)$, the scalar response Y_i is obtained by sampling from the Gaussian distribution with mean $\beta_{z_i,0} + \int_{\mathcal{T}} X_i(t)\beta_{z_i}(t)dt$ and variance $\sigma_{z_i}^2$. The value z_i is then the true cluster label of the predictor $X_i(\cdot)$. This hierarchical generative process can be summarized as follows:

$$\begin{aligned} Y_i | Z_i = z_i, X_i(t) &\sim \mathcal{N}\Big(\beta_{z_i,0} + \int_{\mathcal{T}} X_i(t)\beta_{z_i}(t)dt;\ \sigma_{z_i}^2\Big), \\ Z_i | X_i(t) &\sim \mathcal{M}\Big(1, \big(\pi_1(X_i(t); \boldsymbol{\alpha}), \ldots, \pi_K(X_i(t); \boldsymbol{\alpha})\big)\Big). \end{aligned} \tag{3.41}$$

The true generated data $(X_i(\cdot), Y_i, Z_i)$ are used for evaluating the prediction and clustering performance.

Finally, to mimic real-world data, in our simulations, the generated predictors $X_i(\cdot)$ are contaminated with measurement errors, that means we will not use $X_i(\cdot)$ for analysis, but

$$U_i(t) = X_i(t) + \delta_i(t),$$

with $\delta_i(t)$ is an independent Gaussian noise with mean zero and constant variance σ_δ^2 for all $t \in \mathcal{T}$. We considered different noise levels σ_δ^2 (see Table 3.1).

3.4.2.2 Simulation parameters and experimental protocol

The parameters that were used in the data generating process (3.41) are as follows. We consider $K = 3$ components and a time-domain $\mathcal{T} = [0, 1]$.

Functional experts and gating parameters: The functional experts parameters are given by

$$\begin{aligned}
\beta_1(t) &= \begin{cases} -50(t-0.5)^2+4 & \text{if } 0 \leqslant t < 0.3, \\ 0 & \text{if } 0.3 \leqslant t < 0.7, \\ 50(t-0.5)^2-4 & \text{if } 0.7 \leqslant t \leqslant 1, \end{cases} \\
\beta_2(t) &= -\beta_1(t), \\
\beta_3(t) &= 100(t-0.5)^2 - 10, \quad 0 \leqslant t \leqslant 1, \\
(\beta_{1,0}, \beta_{2,0}, \beta_{3,0})^\top &= (-5, 0, 5)^\top, \\
(\sigma_1^2, \sigma_2^2, \sigma_3^2)^\top &= (5, 5, 5)^\top,
\end{aligned}$$

and the functional gating network parameters are given by

$$\begin{aligned}
\alpha_1(t) &= 80(t-0.5)^2 - 8, \\
\alpha_2(t) &= -\alpha_1(t), \quad \alpha_3(t) = \mathbf{0}, \quad 0 \leqslant t \leqslant 1, \\
(\alpha_{1,0}, \alpha_{2,0}, \alpha_{3,0})^\top &= (-10, -10, 0)^\top.
\end{aligned}$$

Note that to satisfy the identifiability condition (see Jiang and Tanner (1999)), the experts are ordered, for instance $(\beta_{1,0}, \beta_1, \sigma_1^2) \prec \ldots \prec (\beta_{K,0}, \beta_K, \sigma_K^2)$, and the last gating network parameters, $\alpha_{K,0}, \alpha_K(t)$, are initialized, *i.e.*, fixed to zero. Here, "$\prec$" is the lexicographical order on $\mathbb{R}^{p+2}$.

As it can be seen in Figure 3.1, the expert parameter functions $\beta_1(t)$ and $\beta_2(t)$ have a flat region in the interval $0.3 \leqslant t < 0.7$, outside of which they are quadratic, while $\beta_3(t)$ and the gating parameter functions $\alpha_1(t)$, $\alpha_2(t)$ are all quadratic on the whole domain. By considering this network, we can later compare the sparsity in the zeroth and third derivatives of the solutions obtained by the proposed models.

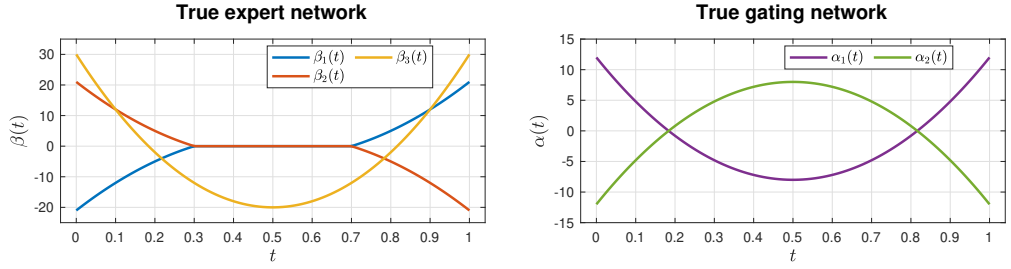


Figure 3.1: The true expert and gating functions used in simulations.

Functional predictors parameters: In this simulation, the functional predictors $X_i(\cdot)$ are constructed using the following formula,

$$X_i(t) = \boldsymbol{x}_i^\top \boldsymbol{b}(t), \quad t \in \mathcal{T}, \tag{3.42}$$

in which $\boldsymbol{b}(t) = [b_1(t), \ldots, b_{10}(t)]^\top$ is a 10-dimensional B-spline basis $\boldsymbol{x}_i \in \mathbb{R}^{10}$ is a coefficient vector defined as $\boldsymbol{x}_i = \boldsymbol{W}\boldsymbol{v}_i$, where $\boldsymbol{W} \in \mathbb{R}^{10\times 10}$ is a matrix of i.i.d random values from the uniform

distribution $\mathcal{U}(0,1)$ and $\boldsymbol{v}_i \in \mathbb{R}^{10}$ is a vector of i.i.d random values from the normal distribution $\mathcal{N}(0,10)$. Here, the matrix $\boldsymbol{W}$ acts as a factor to facilitate the fluctuation of the generated $X_i(\cdot)$.

Noisy functional predictors: Since in real practical situations we do not usually directly observe the true functional predictors $X(\cdot)$'s, but only a noisy and discretized version of them, we thus consider several scenarios with different noise and sampling levels. To this end, in the data generating protocol we consider noisy versions $U_i(t_j) = X_i(t_j) + \delta_i(t_j)$ of the functional predictors values $X_i(t_j)$, where $\delta_i(t_j) \sim \mathcal{N}(0, \sigma_\delta^2)$ is a centred Gaussian noise with variance σ_δ^2, for all $i \in [n]$, $j \in [m]$. We investigate simulated scenarios $S1, \ldots, S4$ with curve length m and the noise level σ_δ^2 of the functional predictors, including two levels of sampling $m \in \{100, 300\}$, and two levels of noise $\sigma_\delta^2 \in \{1, 4\}$. The resulting four considered scenarios of simulated data are presented in Table 3.1. For each considered scenario, we generate $N = 100$ datasets, each dataset contains $n = 200$ pairs of $(U_i(t), Y_i)$, $i \in [n]$. Figure 3.2 displays, for each scenario, 10 randomly taken predictors colored according to their corresponding clusters. For each run, the concerned dataset is randomly split

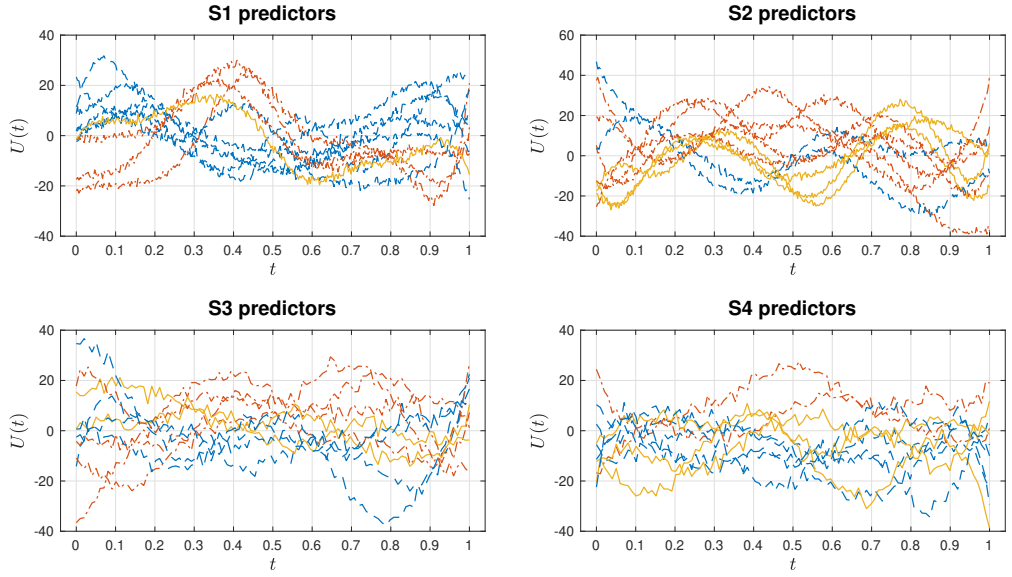


Figure 3.2: 10 randomly taken predictors in scenarios $S1$ (large m and small σ_δ), $S2$ (small m and small σ_δ), $S3$ (large m and large σ_δ), and $S4$ (small m and large σ_δ). Here, the noisy predictors $U_i(\cdot)$ are colored (blue, red, yellow) according to their true cluster labels $Z_i \in \{1, 2, 3\}$.

into a training set and testing set of equal size, the model parameters are estimated using training set, with the tuning parameters selected by maximizing the modified BIC (3.39). The evaluation

Scenario	S1	S2	S3	S4
σ_δ^2	1	1	4	4
m	300	100	300	100

Table 3.1: Simulation scenarios with different noise level σ_δ^2 and curve sampling level m.

criteria are computed on testing set and reported for each model accordingly.

3.4.2.3 Some implementation details

For all scenarios, for all datasets, we implemented the three proposed models with 10 EM runs and with a tolerance of 10^{-6}. For the iFME model, in principle, for each parameter function the two derivatives d_1 and d_2 to be penalized, and the weights for the latter (*i.e.*, ρ and ϱ) can be seen as tuning parameters. However, such an implementation could be computationally expensive in this simulation study with 400 datasets in total and 10 EM runs for each dataset. Therefore, we opted to fix d_1 and d_2 for all implementations ($d_1 = 0$ and $d_2 = 3$ for both expert and gating networks.) and left ρ and ϱ to be selected in some sets of targeted values. The choices of the targeted values were made by the following straightforward arguments. Since $\beta_1(\cdot)$ and $\beta_2(\cdot)$ have zero-valued regions, the weight for their zeroth derivative in penalization term should be large, equivalently, the weight for the third derivative should be small, so ρ is selected in a set of small values: $\rho \in \{10^{-2}, 10^{-3}, 10^{-4}\}$. On the other hand, for $\alpha_1(t)$ and $\alpha_2(t)$, we select $\varrho \in \{10, 10^2, 10^3\}$ as we should emphasize sparsity in their third derivative.

3.4.2.4 Simulation results

Clustering and prediction performances. We report in Table 3.2 the results of regression and clustering tasks on simulated datasets in the four considered scenarios.

	RPE	Corr	ARI	ClusErr
S1 ($m = 300, \sigma_\delta^2 = 1$)				
FME	$.1552_{(.1282)}$	$.9188_{(.0602)}$	$.7852_{(.0934)}$	$.0899_{(.0434)}$
FME-Lasso	$.1390_{(.1074)}$	$.9224_{(.0603)}$	$.7670_{(.1436)}$	$.1112_{(.0891)}$
iFME	$\mathbf{.1334_{(.0977)}}$	$\mathbf{.9287_{(.0504)}}$	$\mathbf{.7997_{(.0796)}}$	$\mathbf{.0792_{(.0340)}}$
S2 ($m = 100, \sigma_\delta^2 = 1$)				
FME	$.1600_{(.1698)}$	$.9164_{(.0712)}$	$\mathbf{.7966_{(.0881)}}$	$.0852_{(.0427)}$
FME-Lasso	$.1656_{(.1316)}$	$.9071_{(.0733)}$	$.7455_{(.1520)}$	$.1236_{(.0961)}$
iFME	$\mathbf{.1540_{(.0950)}}$	$\mathbf{.9192_{(.0476)}}$	$.7955_{(.0822)}$	$\mathbf{.0820_{(.0341)}}$
S3 ($m = 300, \sigma_\delta^2 = 4$)				
FME	$.1724_{(.1807)}$	$.9110_{(.0778)}$	$.7798_{(.0838)}$	$.0918_{(.0415)}$
FME-Lasso	$.1457_{(.1504)}$	$.9197_{(.0785)}$	$.7629_{(.1524)}$	$.1115_{(.0928)}$
iFME	$\mathbf{.1432_{(.0860)}}$	$\mathbf{.9228_{(.0448)}}$	$\mathbf{.7987_{(.0899)}}$	$\mathbf{.0783_{(.0361)}}$
S4 ($m = 100, \sigma_\delta^2 = 4$)				
FME	$.2251_{(.4462)}$	$.9048_{(.0822)}$	$\mathbf{.7816_{(.1040)}}$	$.0927_{(.0497)}$
FME-Lasso	$\mathbf{.1432_{(.1229)}}$	$\mathbf{.9188_{(.0683)}}$	$.7526_{(.1654)}$	$.1215_{(.1055)}$
iFME	$.1639_{(.0978)}$	$.9125_{(.0521)}$	$.7798_{(.0848)}$	$\mathbf{.0864_{(.0325)}}$

Table 3.2: Evaluation criteria of FME, FME-Lasso and iFME models for test data in scenarios $S1, \ldots, S4$. The reported values are the averages of 100 Monte Carlo runs with standard errors in parentheses. The bold values correspond to the best solution.

The mean and standard error of the relative predictions error (RPE) and correlation (Corr) summarize the regression performance, while the mean and standard error of the Rand index (RI), adjusted Rand index (ARI) and clustering error (ClusErr) summarize the clustering performance.

As we can see from Table 3.2, all the models have very good performances on both regression and clustering tasks, with high correlation, RI, ARI, and small RPE and clustering error. The iFME appears to slightly have a better performance than the others in all scenarios. The low standard errors confirm the stability of the algorithms.

Figure 3.3 shows the clustering results obtained by the models with highest values of the modified BIC criterion, on a dataset selected in scenario $S1$. Here, we plotted the responses against the predictors at two specific time points: $t_1 = 0$ and $t_{50} = 0.5$. The highly accurate predictions (in both regression and clustering) can be seen visually through Figure 3.3. This figure also shows that it is difficult to cluster these data according to a few number of time observations, for example in $\mathbb{R}^2$, according to $\{(U_i(t_0), y_i)\}_{i=1}^n$ or $\{(U_i(t_{50}), y_i)\}_{i=1}^n$, which suggests using functional approaches.

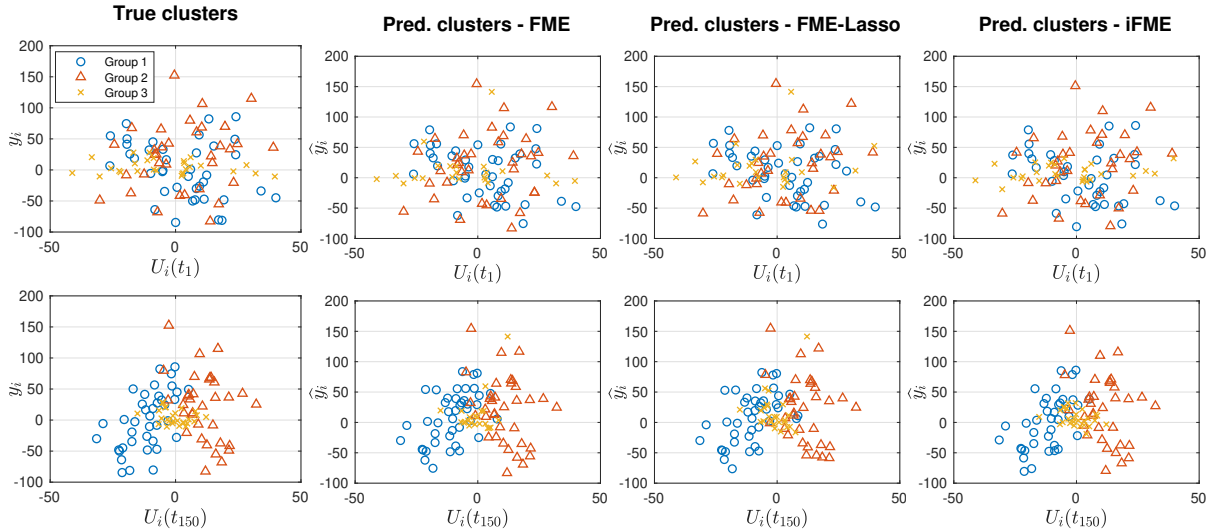


Figure 3.3: Scatter plots of $\widehat{y}_i$ against $U_i(t_1)$ (top panels) and $U_i(t_{150})$ (bottom panels) on a randomly selected dataset. Here, the clustering errors are 5.5%, 4.75% and 5.0% for FME, FME-Lasso and iFME models, respectively.

Comparison with functional regression mixtures (FMR). Finally, we compare our proposed models with the functional mixture regression (FMR) model proposed in Yao et al. (2010). In their approach, the functional predictors are first projected onto its eigenspace, then the obtained new coordinates are fed to the standard mixture regression model to estimate the weights and the coefficients of the $\widehat{\beta}(\cdot)$ in that eigenspace. They performed functional principal component analysis (FPCA) to obtain estimates for the eigenfunctions and the principal component scores (which serve as predictors). The number of relevant FPCA components are chosen automatically for each dataset by selecting the minimum number of components that explain 90% of the total variation

of the predictors. It is noticed that, in their simulation studies, the authors computed the scalar responses by using conditional prediction, *i.e.*, the true y_i were used to determine which cluster the observation belongs to. Then the predicted $\widehat{y}_i$ is calculated as the conditional mean of the density of the corresponding cluster. For comparison with that approach, we also used this strategy to make predictions in our models. We further employed the FMR model with the B-spline functional representation, instead of the functional PCA, the number of B-spline functions is set to be the same as the number of basis functions used in our models. Table 3.3 shows the evaluation criteria corresponding to the considered models evaluated on 100 datasets in scenario $S3$. Here, FMR-PC is the original model of Yao et al. (2010) and FMR-B is the modified one with B-spline bases. As expected, FME, FME-Lasso, and iFME, which are more flexible compared to the the FMR model, allows to capture more complexity in the data, thanks to the predictor-depending mixture weights, and provides clearly better results than the FMR alternatives.

	RPE	Corr	ARI	ClusErr
FME	$.0614_{(.0896)}$	$.9681_{(.0469)}$	$.8523_{(.0934)}$	$.0618_{(.0490)}$
FME-Lasso	$.0145_{(.0254)}$	$.9924_{(.0133)}$	$.9582_{(.0641)}$	$.0174_{(.0325)}$
iFME	$\mathbf{.0139_{(.0086)}}$	$\mathbf{.9928_{(.0046)}}$	$\mathbf{.9594_{(.0430)}}$	$\mathbf{.0151_{(.0166)}}$
FMR-B	$.0460_{(.0399)}$	$.9768_{(.0201)}$	$.7064_{(.1044)}$	$.1124_{(.0434)}$
FMR-PC	$.0191_{(.0331)}$	$.9902_{(.0164)}$	$.8345_{(.0794)}$	$.0612_{(.0361)}$

Table 3.3: Performance comparison of the models for datasets in scenarios $S3$. The reported values are the averages of 100 Monte Carlo samples with standard errors in parentheses.

Parameter estimation performance. To evaluate the parameter estimation performance, we consider the functional parameter functions estimation error as defined by (3.40). This error between the true function and the estimated one, provides an evaluation of how well the proposed models reconstruct the hidden functional gating and expert networks. In this evaluation, we considered scenario S1, i.e., $m = 300$, $\sigma_\delta^2 = 1$. Moreover, in order to provide an idea of the impact of training size on parameter estimation, we run the models with different training sizes (share the same scenario S1) and report the MSE for each function, for each model in Table 3.4. It shows that there are significant improvements, even with small training size, when using iFME model in estimating the gating network, compared to the FME and FME-Lasso model.

Now, in order to evaluate the designed sparsity of zeroth and third derivatives of the reconstructed functions, we compute the MSEs versus their true values, i.e., zeros, on each designed intervals. In particular, we divide the domain $\mathcal{T} = [0, 1]$ into three parts: $\mathcal{T}_1 = [0, 0.3)$, $\mathcal{T}_2 = [0.3, 0.7)$, and $\mathcal{T}_3 = [0.7, 1]$, then for each model the MSEs on the corresponding intervals are reported in Table 3.5.

The reported values show that, as expected, iFME model is better than both FME and FME-Lasso in providing sparse solutions with respect to the derivatives of the parameter functions.

Table 3.6 shows the means and the standard errors of the estimated intercepts and variances.

	$\widehat{\beta}_1(\cdot)$	$\widehat{\beta}_2(\cdot)$	$\widehat{\beta}_3(\cdot)$	$\widehat{\alpha}_1(\cdot)$	$\widehat{\alpha}_2(\cdot)$
Training size: 100					
FME	$11.68_{(56.63)}$	$6.21_{(24.34)}$	$8.57_{(27.60)}$	5.15e+04 (7.23e+04)	5.69e+04 (5.90e+06)
FME-Lasso	$\mathbf{0.53_{(0.19)}}$	$\mathbf{0.66_{(0.55)}}$	$\mathbf{0.62_{(0.79)}}$	$17.06_{(28.68)}$	$19.50_{(37.07)}$
iFME	$1.09_{(0.78)}$	$1.05_{(0.82)}$	$2.44_{(3.06)}$	$\mathbf{5.76_{(7.78)}}$	$\mathbf{4.25_{(3.58)}}$
Training size: 200					
FME	$0.77_{(0.48)}$	$0.61_{(0.30)}$	$\mathbf{0.19_{(0.20)}}$	$8.55_{(11.99)}$	$9.57_{(8.29)}$
FME-Lasso	$\mathbf{0.54_{(0.18)}}$	$0.54_{(0.20)}$	$0.31_{(0.81)}$	$10.71_{(16.63)}$	$12.64_{(16.33)}$
iFME	$0.55_{(0.50)}$	$\mathbf{0.48_{(0.38)}}$	$1.26_{(2.91)}$	$\mathbf{5.34_{(6.21)}}$	$\mathbf{2.81_{(2.82)}}$
Training size: 300					
FME	$0.62_{(0.56)}$	$0.59_{(0.50)}$	$\mathbf{0.16_{(0.18)}}$	$12.58_{(19.83)}$	$16.49_{(30.97)}$
FME-Lasso	$0.60_{(0.52)}$	$\mathbf{0.57_{(0.44)}}$	$0.17_{(0.19)}$	$11.26_{(16.56)}$	$13.15_{(24.80)}$
iFME	$\mathbf{0.56_{(0.43)}}$	$0.59_{(0.50)}$	$0.49_{(0.47)}$	$\mathbf{2.98_{(3.12)}}$	$\mathbf{3.16_{(3.02)}}$
Training size: 400					
FME	$0.54_{(0.23)}$	$0.53_{(0.19)}$	$\mathbf{0.11_{(0.12)}}$	$8.27_{(11.73)}$	$7.08_{(8.53)}$
FME-Lasso	$0.52_{(0.18)}$	$0.51_{(0.16)}$	$0.15_{(0.18)}$	$6.09_{(8.98)}$	$5.14_{(6.51)}$
iFME	$\mathbf{0.34_{(0.19)}}$	$\mathbf{0.38_{(0.25)}}$	$0.74_{(0.62)}$	$\mathbf{3.06_{(4.21)}}$	$\mathbf{2.77_{(3.21)}}$
Training size: 500					
FME	$0.49_{(0.22)}$	$0.50_{(0.28)}$	$\mathbf{0.09_{(0.11)}}$	$5.17_{(7.52)}$	$3.90_{(5.69)}$
FME-Lasso	$0.49_{(0.18)}$	$0.49_{(0.21)}$	$\mathbf{0.09_{(0.11)}}$	$4.62_{(7.28)}$	$7.63_{(16.27)}$
iFME	$\mathbf{0.35_{(0.23)}}$	$\mathbf{0.40_{(0.25)}}$	$0.55_{(0.63)}$	$\mathbf{2.11_{(2.82)}}$	$\mathbf{2.68_{(4.70)}}$

Table 3.4: Average of 100 Monte Carlo runs of MSE between the estimated functions resulted by FME, FME-Lasso and iFME models in S1 scenario.

	on $\mathcal{T}_2$		on $\mathcal{T}_1 \cup \mathcal{T}_3$		on $\mathcal{T}$		
	$\widehat{\beta}_1^{(0)}(\cdot)$	$\widehat{\beta}_2^{(0)}(\cdot)$	$\widehat{\beta}_1^{(3)}(\cdot)$	$\widehat{\beta}_2^{(3)}(\cdot)$	$\widehat{\beta}_3^{(3)}(\cdot)$	$\widehat{\alpha}_1^{(3)}(\cdot)$	$\widehat{\alpha}_2^{(3)}(\cdot)$
Training size: 100							
FME	**0.26** (**0.15**)	0.48 (0.91)	0.15 (0.07)	**0.12** (**0.08**)	3.96e-08 (1.37e-07)	9.24e-06 (3.32e-05)	1.08e-05 (2.62e-05)
FME-Lasso	0.27 (0.19)	**0.26** (**0.16**)	**0.14** (**0.07**)	0.15 (0.09)	**1.64e-09** (**7.64e-09**)	7.37e-09 (2.64e-08)	5.12e-09 (1.70e-08)
iFME	0.40 (0.26)	0.37 (0.30)	0.20 (0.10)	0.18 (0.11)	6.48e-09 (2.44e-08)	**4.05e-09** (**1.33e-08**)	**2.46e-09** (**4.51e-09**)
Training size: 200							
FME	0.43 (0.56)	0.22 (0.09)	0.16 (0.10)	0.14 (0.05)	**2.23e-11** (**2.69e-11**)	**1.64e-10** (**2.33e-10**)	3.83e-10 (8.90e-10)
FME-Lasso	0.21 (0.11)	0.20 (0.09)	0.15 (0.05)	0.15 (0.05)	3.62e-11 (6.04e-11)	2.36e-10 (3.38e-10)	**2.96e-10** (**6.41e-10**)
iFME	**0.18** (**0.08**)	**0.17** (**0.09**)	**0.12** (**0.05**)	**0.12** (**0.05**)	9.95e-10 (2.00e-09)	1.44e-09 (2.72e-09)	8.52e-10 (1.22e-09)
Training size: 300							
FME	0.25 (0.30)	0.22 (0.25)	0.16 (0.10)	0.15 (0.09)	**1.72e-11** (**2.30e-11**)	2.55e-10 (4.61e-10)	1.01e-10 (1.42e-10)
FME-Lasso	0.25 (0.28)	**0.21** (**0.23**)	0.16 (0.09)	**0.15** (**0.08**)	1.73e-11 (2.52e-11)	**1.41e-10** (**2.00e-10**)	**1.64e-10** (**3.77e-10**)
iFME	**0.25** (**0.19**)	0.24 (0.19)	**0.16** (**0.08**)	**0.15** (**0.08**)	5.94e-10 (1.26e-09)	5.41e-10 (7.26e-10)	7.15e-10 (1.05e-09)
Training size: 400							
FME	0.23 (0.12)	0.23 (0.10)	0.16 (0.05)	0.16 (0.06)	**1.62e-11** (**3.53e-11**)	1.29e-10 (3.59e-10)	**8.14e-11** (**1.02e-10**)
FME-Lasso	0.22 (0.11)	0.21 (0.10)	0.16 (0.05)	0.16 (0.05)	1.90e-11 (3.55e-11)	**9.71e-11** (**1.63e-10**)	1.03e-10 (1.54e-10)
iFME	**0.14** (**0.08**)	**0.16** (**0.09**)	**0.08** (**0.03**)	**0.08** (**0.04**)	1.87e-08 (1.42e-08)	1.13e-08 (1.77e-08)	9.91e-09 (1.05e-08)
Training size: 500							
FME	0.22 (0.11)	0.21 (0.10)	0.17 (0.05)	0.17 (0.05)	2.12e-11 (4.66e-11)	8.98e-11 (2.70e-10)	1.03e-10 (2.83e-10)
FME-Lasso	0.21 (0.09)	0.20 (0.09)	0.17 (0.05)	0.17 (0.05)	**1.90e-11** (**3.64e-11**)	**7.02e-11** (**1.44e-10**)	**8.91e-11** (**1.69e-10**)
iFME	**0.14** (**0.07**)	**0.16** (**0.08**)	**0.08** (**0.04**)	**0.08** (**0.04**)	1.55e-08 (1.34e-08)	1.03e-08 (1.58e-08)	8.34e-09 (1.20e-08)

Table 3.5: MSE of the derivatives of reconstructed functions on the corresponding interested intervals, in which, $\mathcal{T}_1 = [0, 0.3)$, $\mathcal{T}_2 = [0.3, 0.7)$, $\mathcal{T}_3 = [0.7, 1]$ and $\mathcal{T} = [0, 1]$. The reported values are the averaged of 100 Monte Carlo samples with standard errors in parentheses.

Note that these coefficients are not considered in the penalization. For the intercepts $\beta_{k,0}$, all the models estimated them very well, while for the intercepts $\alpha_{k,0}$, iFME is slightly better than the others. For the variances, FME-Lasso gave the estimated values closest with the true values on average.

	$\widehat{\beta}_{1,0}$	$\widehat{\beta}_{2,0}$	$\widehat{\beta}_{3,0}$	$\widehat{\alpha}_{1,0}$	$\widehat{\alpha}_{2,0}$	$\widehat{\sigma}_1$	$\widehat{\sigma}_2$	$\widehat{\sigma}_3$
True value	−5	0	5	−10	−10	5	5	5
Training size: 100								
FME	−16.90 (18.91)	−3.18 (3.25)	4.17 (8.87)	−31.37 (30.74)	−25.20 (41.27)	32.38 (40.05)	15.19 (25.15)	21.47 (24.71)
FME-Lasso	−6.48 (9.37)	**−0.07** (**1.95**)	5.51 (4.63)	−12.53 (8.99)	−12.69 (10.99)	6.86 (10.98)	6.73 (8.08)	10.01 (20.60)
iFME	**−4.95** (**1.14**)	0.21 (1.08)	**5.05** (**0.90**)	**−9.44** (**3.85**)	**−9.39** (**3.31**)	**5.70** (**1.46**)	**5.75** (**1.49**)	**4.70** (**1.51**)
Training size: 200								
FME	−4.83 (0.78)	0.14 (0.54)	4.89 (0.38)	−23.77 (26.36)	−16.10 (6.65)	5.77 (1.26)	**5.63** (**0.95**)	5.24 (0.79)
FME-Lasso	**−4.97** (**0.66**)	**0.07** (**0.60**)	4.99 (0.45)	−12.70 (4.45)	−13.72 (4.38)	5.94 (1.23)	5.99 (1.12)	**4.92** (**1.30**)
iFME	−4.85 (0.72)	0.19 (0.68)	**4.99** (**0.42**)	**−9.30** (**3.15**)	**−9.21** (**2.83**)	**5.77** (**0.90**)	5.72 (1.02)	5.28 (1.27)
Training size: 300								
FME	**−4.99** (**0.55**)	−0.06 (0.54)	4.98 (0.26)	−16.82 (13.09)	−17.18 (14.89)	5.92 (1.14)	5.87 (0.94)	**5.34** (**0.84**)
FME-Lasso	−5.03 (0.54)	**−0.03** (**0.53**)	4.98 (0.27)	−13.06 (4.54)	−12.96 (4.96)	6.08 (1.10)	5.99 (1.07)	5.37 (0.89)
iFME	−4.93 (0.61)	0.04 (0.56)	**5.00** (**0.29**)	**−8.96** (**2.48**)	**−8.85** (**2.57**)	**5.54** (**0.75**)	**5.53** (**0.72**)	5.53 (0.87)
Training size: 400								
FME	**−5.02** (**0.43**)	**0.01** (**0.40**)	5.04 (0.28)	−16.37 (9.93)	−15.39 (7.83)	5.99 (1.01)	6.20 (1.09)	5.47 (0.77)
FME-Lasso	−5.03 (0.43)	−0.02 (0.38)	5.02 (0.27)	−13.18 (4.61)	−12.65 (4.57)	5.99 (0.94)	6.13 (1.01)	**5.45** (**0.84**)
iFME	−4.89 (0.45)	0.05 (0.35)	**5.01** (**0.27**)	**−10.17** (**2.79**)	**−9.82** (**2.61**)	**5.08** (**0.60**)	**5.22** (**0.58**)	5.57 (0.85)
Training size: 500								
FME	**−5.00** (**0.41**)	−0.01 (0.38)	**5.01** (**0.25**)	−14.09 (7.26)	−15.42 (9.64)	6.09 (0.98)	6.06 (0.93)	5.44 (0.69)
FME-Lasso	−4.99 (0.40)	**−0.01** (**0.37**)	5.01 (0.26)	−11.35 (3.27)	−12.38 (4.15)	6.13 (0.98)	6.06 (0.90)	**5.44** (**0.68**)
iFME	−4.92 (0.44)	0.06 (0.34)	5.02 (0.27)	**−9.23** (**1.93**)	**−9.61** (**2.68**)	**5.23** (**0.55**)	**5.13** (**0.55**)	5.60 (0.75)

Table 3.6: Intercepts and variances obtained by FME, FME-Lasso and iFME models in scenario $S1$. The reported values are the averages of 100 Monte Carlo samples with standard errors in parentheses.

Table 3.7 shows a comparison between three different initialization strategies for the EM algorithm. In particular, the initialization strategies regards only the gating network parameter vector, since for the expert network a K-means approach is more suitable. With the "rand."

	log likelihood			RPE			Time (s)		
	rand.	zeros	LR	rand.	zeros	LR	rand.	zeros	LR
FME	**−215.52** (**24.77**)	−221.03 (18.51)	−285.23 (69.30)	**0.19** (**0.16**)	0.20 (0.13)	0.38 (0.29)	0.21 (0.50)	**0.16** (**0.12**)	0.52 (1.99)
FME-Lasso	**−235.35** (**21.98**)	−239.47 (18.54)	−244.99 (63.09)	0.14 (0.11)	**0.13** (**0.10**)	0.20 (0.16)	1.13 (1.23)	**1.08** (**2.48**)	4.63 (15.51)
iFME	**−254.65** (**7.83**)	−256.26 (13.76)	−336.87 (80.38)	**0.13** (**0.09**)	**0.13** (**0.09**)	0.35 (0.30)	6.58 (4.45)	4.17 (3.65)	**3.91** (**2.08**)

Table 3.7: Comparison of different initialization strategies. The reported values are the mean and standard error (in parentheses) over 100 Monte Carlo runs.

strategy, all coefficients of the gating network parameter vectors are drawn randomly in $\mathcal{N}(0, 1)$. With the "zeros" strategy, they are initialized as zeros. And with the "LR" we basically perform a logistic regression where the predictors are the design vectors associated with the gating network (*i.e.*, $\mathbf{r}_i$ for FME, FME-Lasso, and $\mathbf{s}_i$ for iFME model), and the responses are the labels resulted by K-means on the gating design matrix.

Finally, to illustrate the selection of the number of expert components using BIC and/or modified BIC in this simulation study, we provide, in Figure 3.4, the plots of these criteria against the number of experts for each model. Here, we implemented the models with all fixed tuning parameters, except K which varies in the set $\{1, \ldots, 6\}$. We can observe that BIC selects the correct number $K = 3$ for both FME-Lasso and iFME, while it selects $K = 4$ for FME. However, the modified BIC selects $K = 4$ for both FME and FME-Lasso, and selects the true number of components $K = 3$ for iFME.

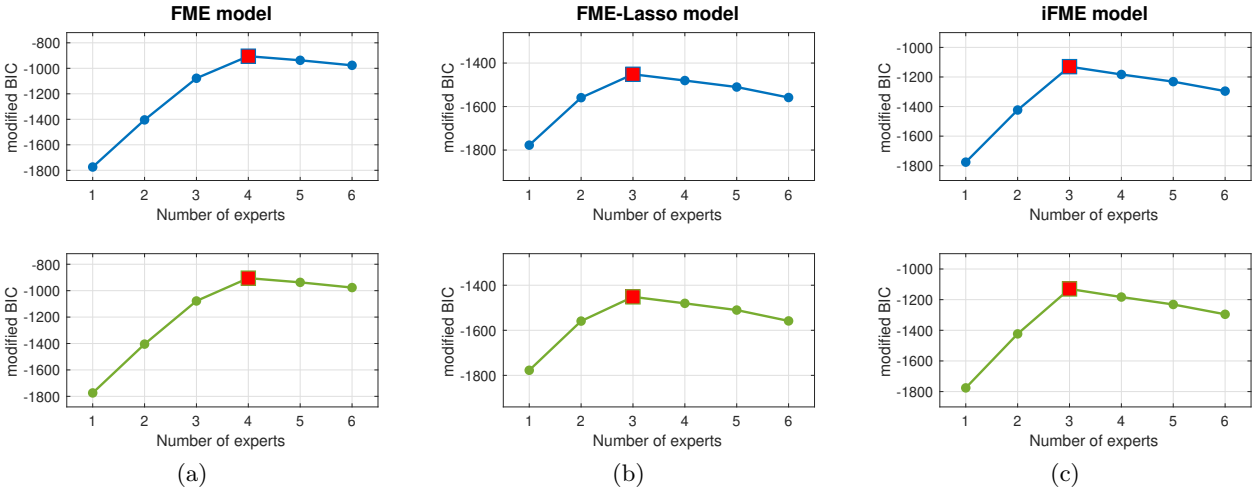


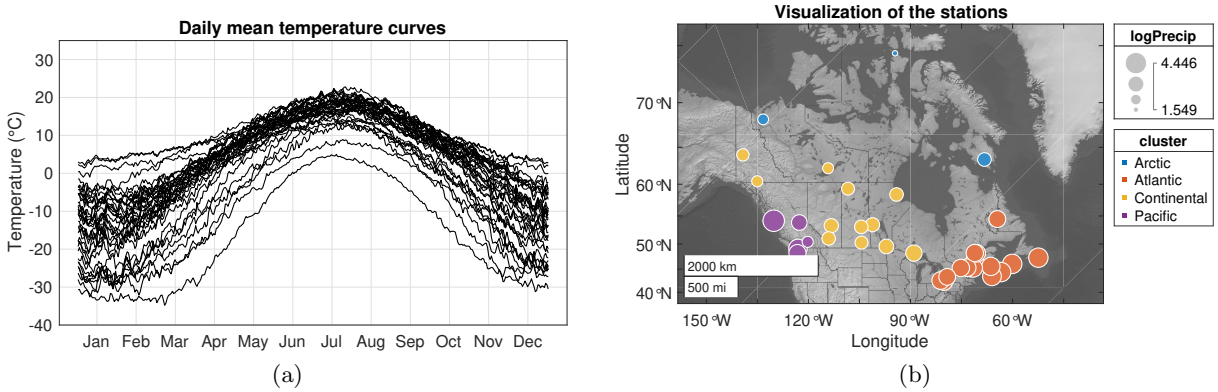
Figure 3.4: Values of BIC (top) and modified BIC (bottom) for (a) FME, (b) FME-Lasso and (c) iFME model versus the number of experts K, fitted on a randomly taken dataset in scenario S1. Here, the square points correspond to the highest values.

3.4.3 Application to real-world data

In this section, we apply the FME, FME-Lasso and iFME models to two well-known real-world datasets, Canadian weather and Diffusion Tensor Imaging (DTI). For each dataset, we perform clustering and investigate the prediction performance, estimate the functional mixture of experts models with different number of experts K and perform the selection of K using modified BIC, and discuss the obtained results.

3.4.3.1 Canadian weather data

Canadian weather is a well-known meteorological data set in FDA. This dataset consists of $m = 365$ daily temperature measurements (averaged over the year 1961 to 1994) at $n = 35$ weather stations in Canada, and their corresponding average annual precipitation (in log scale). The weather stations are located in $K = 4$ climate zones: Atlantic, Pacific, Continental and Arctic (Figure 3.5b). In this dataset, presented in Figure 3.5a, the noisy functional predictors $U_i(\cdot)$ are the curves of 365 averaged daily temperature measurements, the scalar responses Y_i are the corresponding total precipitation at each station i, during the year, for 35 stations. Its station climate zone is taken as a cluster label (the cluster label $Z_i \in \{1, \cdots, 4\}$. The aim here is to use the daily temperature curves (functional



(a) (b)

Figure 3.5: (a) 35 daily mean temperature measurement curves; (b) Geographical visualization of the stations, in which the sizes of the bubbles corresponds to their log of precipitation values and the colors correspond to their climate regions.

predictors) to predict the precipitations (scalar responses) at each station. Moreover, in addition to predicting the precipitation values, we are interested in clustering the temperature curves (therefore the stations), as well as identifying the periods of time of the year that have effect on prediction for each group of curves.

Firstly, in order to assess the prediction performance of the FME, FME-Lasso and iFME models on this dataset, we implement the models by selecting the tuning parameters, including the number of expert components K in the set $\{1, 2, 3, 4, 5, 6\}$, by maximizing the modified BIC criterion, given its performance as shown in the simulation study. We report in Table 3.8 the results in terms of correlations, sum of squared errors (SSE) and relative prediction errors.

	Corr	SSE	RPE
FME	0.690	14.410	0.290
FME-Lasso	0.632	2.011	0.045
iFME	**0.944**	**0.582**	**0.012**

Table 3.8: 7-fold cross-validated correlation (Corr), sum of squared errors (SSE) and relative prediction error (RPE) of predictions on Canadian weather data.

According to the obtained results, iFME provides the best results w.r.t all the criteria. The cross-validated RPE provided by iFME is only of 1.2%, the next is FME-Lasso with 4.5%, while the FME model has the worst RPE value. Note that in James et al. (2009), the authors applied their proposed model to Canadian weather data and obtained a 10-fold cross-validated SSE of 4.77. Clearly, with the smaller cross-validated SSEs, the FME-Lasso and iFME models significantly improve the prediction. Finally, in this cross-validation study, the obtained number of components was $K = 4$ for FME and iFME, while for the FME-Lasso model, the selected one is $K = 5$.

Figure 3.6 shows the obtained results with the FME, FME-Lasso and iFME models, with $K = 4$, and with the derivatives $d_1 = 0$, $d_2 = 3$ for the iFME model. The estimated experts functions and gating functions are presented in the two top panels of the curve, while the clustering for the temperature curves and the stations are shown in the two bottom panels. As we can see, all models provide reasonable clustering for the curves which may be corresponding to different complicated underlying meteorological forecasting mechanisms. Particularly, although not using any spatial information, merely temperature information, the obtained clustering for the stations is also comparable with the original labels of the stations. For example, the FME and FME-Lasso models identify exactly the Arctic stations, while iFME identifies exactly the Pacific stations, and all of the models provide reasonable spatially organized clusters. However, what is interesting here is the shape of the expert and gating functions $\widehat{\alpha}(\cdot)$'s and $\widehat{\beta}(\cdot)$'s obtained by the models. While FME and FME-Lasso gave less interpretable estimations, iFME appears to give, as it can be seen in the two top-right panels, piece-wise zero-valued and possibly quadratic estimated functions, which have a wide range of flat relationship from January to February and from June to September.

Motivated by the above results, on direction of identifying the periods of time of the year that truly have an effect on prediction, we implement the iFME model with $K = 2$, and the derivative levels d_1 and d_2 are set to be the zeroth and the third derivatives. The reason for the choices of d_1 and d_2 is that the penalization on the zeroth derivative would take into account zero ranges in the expert and gating functions, while the penalization on the third derivative, would take into account the smoothness for the changes between the periods of times in the functions. The obtained results are shown in Figure 3.7. As we can see, there are differences in the prediction mechanisms of the models between the northern stations and the southern stations. At southern stations, the obtained $\widehat{\beta}_2(t)$ shows that there is a negative relationship in the spring and a positive relationship in the late fall, but no relationship in the remaining period of the year. This phenomenon is concordant with the result obtained in James et al. (2009), where the authors obtained the same

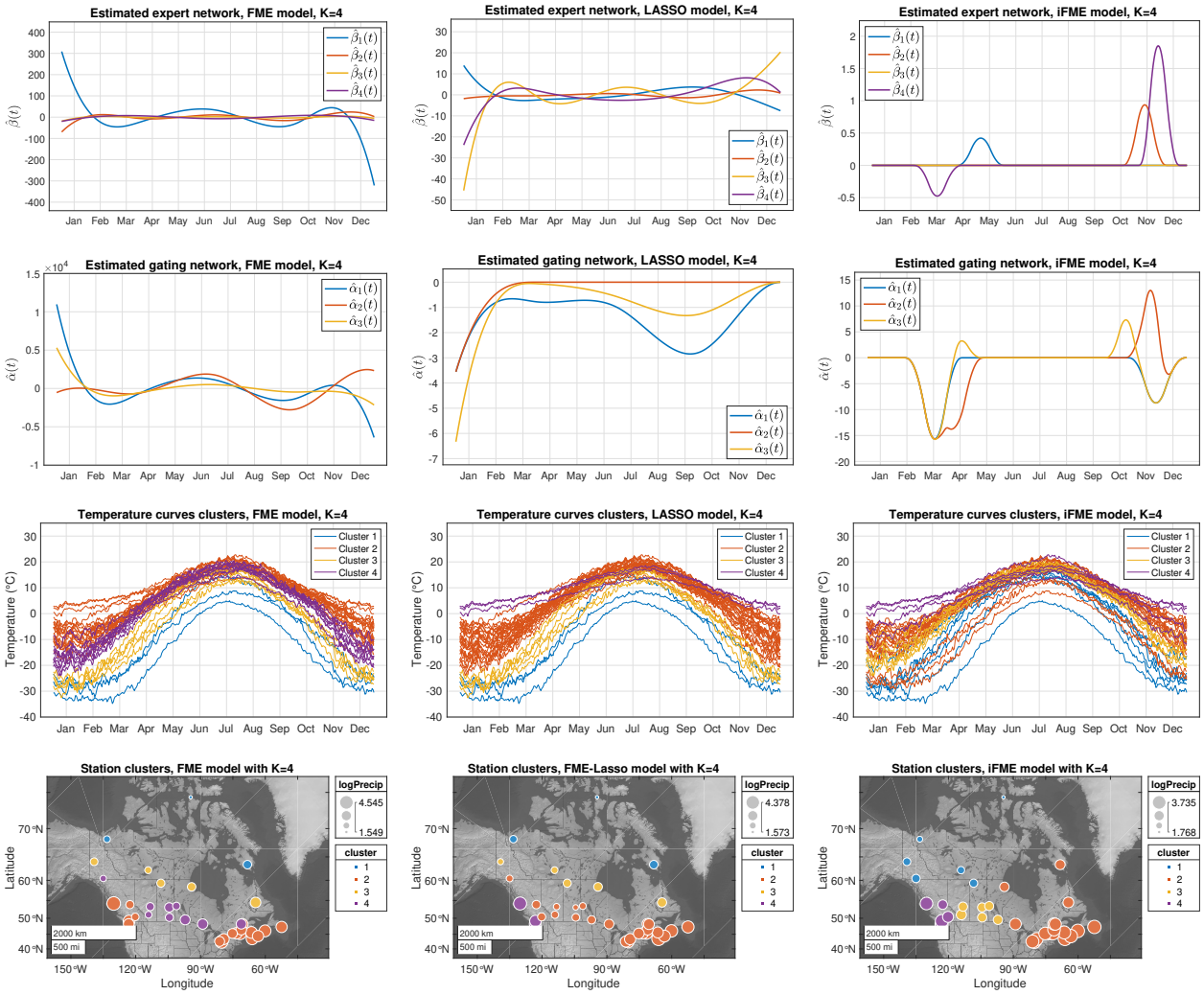


Figure 3.6: Results obtained by FME (left panels), FME-Lasso (middle panels) and iFME (right panels) on Canadian weather data with $K = 4$. For each column, the panels are respectively the estimated functional experts network, estimated functional gating network, estimated clusters of the temperature curves and estimated clusters of the stations.

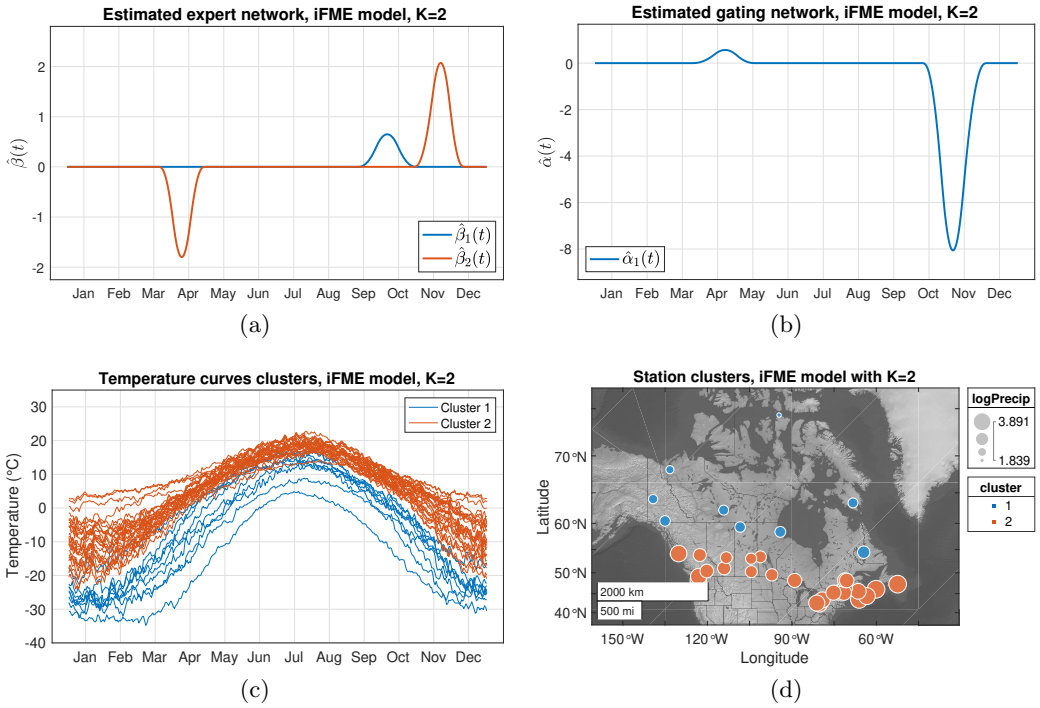


Figure 3.7: Results obtained by iFME model with $K = 2$, $d_1 = 0$, $d_2 = 3$: (a) Estimated functional expert network, (b) Estimated functional gating network, (c) Estimated clusters of the temperature curves, and (d) Estimated clusters of the stations.

relationships in the same periods of time. However, our iFME model additionally suggests that, at the northern stations, the relationship between temperature and precipitation may differ from that of southern stations. This may be explained by the differences in mean temperatures and climatic characteristics between the two regions.

Finally, Figure 3.8 displays the values of modified BIC for varying number of expert component for the proposed models on Canadian weather data. According to these plots, FME-Lasso and iFME select $K = 2$, while FME selects $K = 3$.

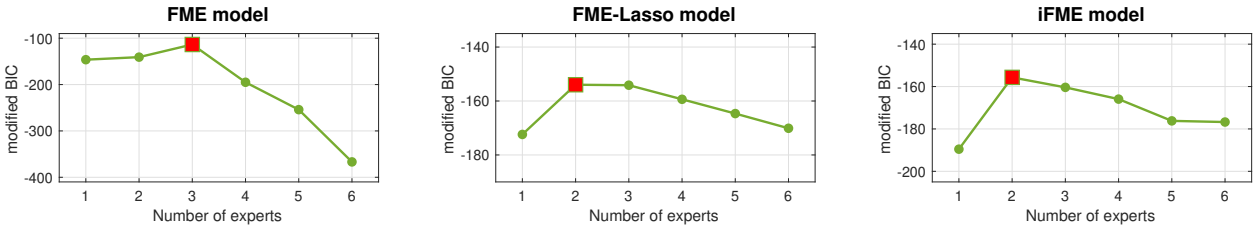


Figure 3.8: Values of modified BIC for (a) FME, (b) FME-Lasso and (c) iFME model, versus the number of experts K, fitted on Canadian weather data. The square points correspond to highest values. Here, iFME is implemented with $d_1 = 0$, $d_2 = 3$ and $\rho = \varrho = 100$.

3.4.3.2 Diffusion tensor imaging data for multiple sclerosis subjects

We now apply our proposed models to the diffusion tensor imaging (DTI) data for subjects with multiple sclerosis (MS), discussed in Goldsmith et al. (2012). The data come from a longitudinal study investigating the cerebral white matter tracts of subjects with multiple sclerosis, recruited from an outpatient neurology clinic and healthy controls. We are interested in the underlying relationship between the fractional anisotropy profile (FAP) from the corpus callosum and the paced auditory serial addition test (PASAT) score, which is a commonly used examination of cognitive function affected by MS. The FAP curves are derived from DTI data, which are obtained by a Magnetic Resonance Imaging (MRI) scanner. Each curve is recorded at 93 locations along the corpus callosum. The PASAT score is the number of correct answers out of 60 questions, and thus ranges from 0 to 60. In our context, the FAP curves serve as the noisy functional predictors $U_i(\cdot)$ and the PASAT scores serve as the scalar responses Y_i. So this dataset consists of $n = 99$ pairs $(U_i(\cdot), Y_i)$, $i \in [n]$, with each $U_i(\cdot)$ contains $m = 93$ fractional anisotropy values. A visualization of the predictors (*i.e.*, the FAP curves) and the responses (*i.e.*, the PASAT Scores) can be seen in Figure 3.9 (top-left panel) and Figure 3.10 (for example, top-right panel).

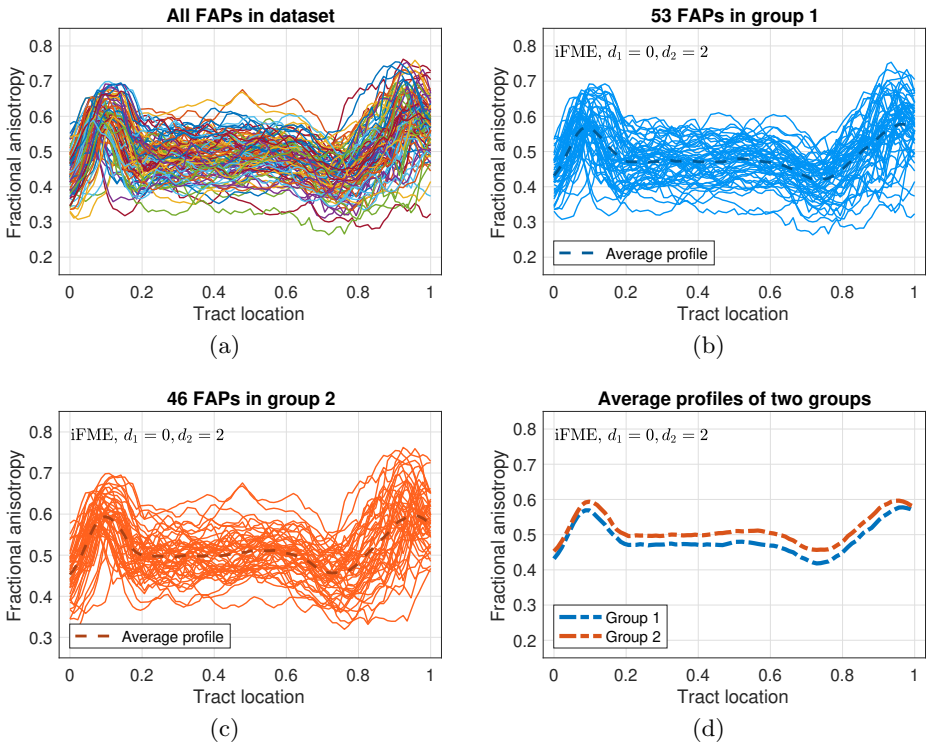


Figure 3.9: DTI data with (a) FAP curves of all subjects, (b) Cluster 1 and (c) Cluster 2, obtained by iFME model with $d_1 = 0$, $d_2 = 2$, and (d) the point-wise average of the curves in each of the two clusters.

In Ciarleglio and Ogden (2016), the authors applied their Wavelet-based functional mixture regression (WBFMR) model with two components to this dataset, and observed that there is one group in which there is no association between the FAP and the PASAT score for those subjects

belonging to it. We accordingly fix $K = 2$ in our models. Figure 3.10 displays the obtained results for each of the three models, and Figure 3.9a shows the functional predictors FAP curves clustered with the iFME model. In this implementation, we tried iFME model with two different combinations of d_1 and d_2: $(d_1, d_2) = (0, 2)$ and $(d_1, d_2) = (0, 3)$. As expected, when d_2 is the second derivative, the reconstructed parameter functions are piecewise zero and linear, while when d_2 is the third derivative, the reconstructed functions have smooth changes along the tract location.

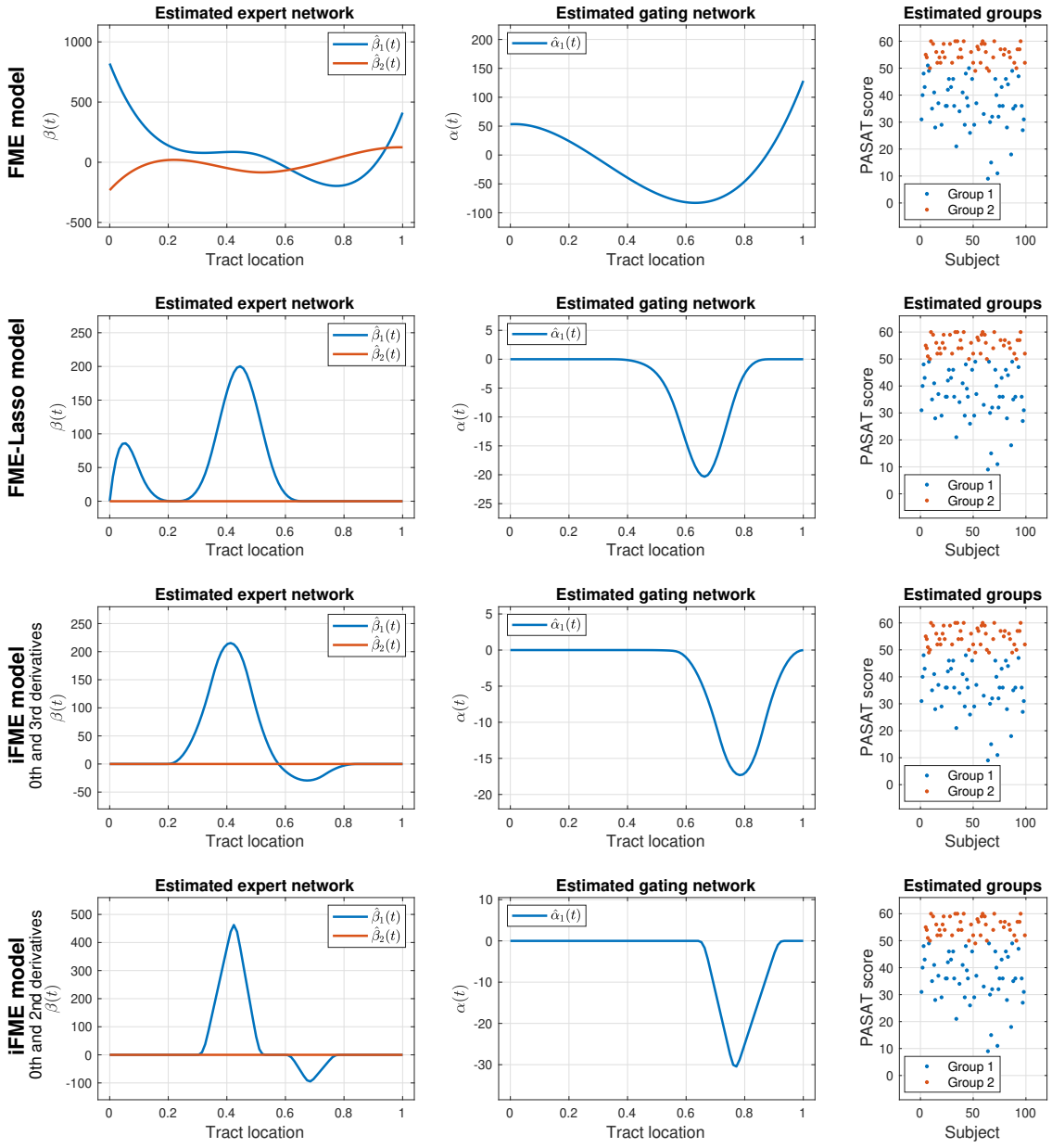


Figure 3.10: The estimated expert and gating coefficient functions, the estimated groups of the PASAT scores, resulted by FME, FME-Lasso and iFME models with $K = 2$ for the DTI dataset. For iFME model, the upper is implemented with penalization on the zeroth and third derivatives, while the lower is with penalized zeroth and second derivatives.

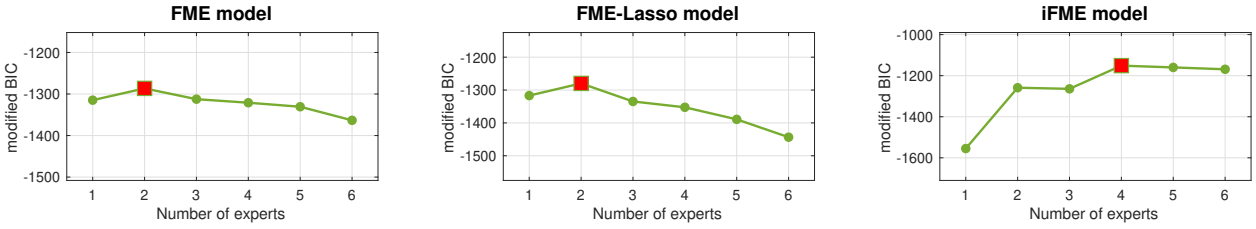
In Figure 3.10, we have the three following observations. First, as it can be observed in Figure 3.10 right-panel, all models give a threshold of 50 that clusters the PASAT scores; This is the same as the threshold observed in Ciarleglio and Ogden (2016). Second, the absolute values of the coefficient functions $\widehat{\beta}_2(t)$'s are significantly smaller than those of $\widehat{\beta}_1(t)$'s; This is again the same with the result obtained by the WBFMR model. Third, when $\widehat{\beta}_2(t)$ is estimated as zero in the FME-Lasso and iFME models, the shape of $\widehat{\beta}_1(t)$ is almost the same as the shape obtained in Ciarleglio and Ogden (2016), particularly, the peak at around the tract location of 0.42. These confirm the underlying relationship between the fractional anisotropy and the cognitive function: higher fractional anisotropy values between the locations about 0.2 to 0.7 results in higher PASAT scores for subjects in Group 1. The clustering of the FAP curves, resulted by the iFME model with $d_1 = 0$, $d_2 = 2$, is shown in Figure 3.9 (b)-(d).

Next, to compare with Ciarleglio and Ogden (2016), we investigate the prediction performance of the proposed models with respect to the leave-one-out cross validated relative prediction errors (CVRPE) defined by CVRPE $= \sum_{i=1}^{n}(y_i - \widehat{y}_i^{(-i)})^2 / \sum_{i=1}^{n} y_i^2$, where y_i is the true score for subject i and $\widehat{y}_i^{(-i)}$ is the score predicted by the model fit on data without subject i. In this implementation, we keep fixing $K = 2$ and select the other tuning parameters by maximizing the modified BIC. The CVRPEs corresponding to the models are provided in Table 3.9. Note that, for comparison, in Ciarleglio and Ogden (2016), the CVRPE of their WBFMR model is 0.0315 and of the wavelet based functional linear model (FLM) is 0.0723.

	CVRPE
FME	0.0273
FME-Lasso	0.0280
iFME (with 0th and 3nd derivatives)	0.0271
iFME (with 0th and 2nd derivatives)	**0.0267**

Table 3.9: CVRPEs of the models on the DTI data.

Finally, we present in Figure 3.11 the selection of the number of experts K with modified BIC. In this case, FME and FME-Lasso select $K = 2$, and iFME selects $K = 4$.



(a) (b) (c)

Figure 3.11: Values of modified BIC for (a) FME, (b) FME-Lasso and (c) iFME, versus the number of experts K, fitted on DTI data. The square points correspond to highest values.

3.5 Summary

The first algorithm for mixtures-of-experts constructed upon functional predictors is presented in this chapter. Beside the classic maximum likelihood parameter estimation, we proposed two other regularized versions that allow sparse and interpretable solutions, by regularizing in particular the derivatives of the underlying functional parameters of the model, after projecting onto a set of continuous basis functions. The performances of the proposed approaches are evaluated in data prediction and clustering via experiments involving simulated and two real-world functional data.

The presented FME models can be extended in different ways. First direct extensions of the modeling framework presented here can be considered with categorical response, to perform supervised classification with functional predictors, or with vector response, to perform multivariate functional regression. Then, it may be interesting to consider the extension of the FME model to setting involving vector (or scalar) predictors and functional responses (Chiou et al., 2004). Another extension, which we intend also to investigate in the future, concerns the case when we observe pairs of functional data, *i.e.*, a sample of n functional data pairs $\{X_i(u), Y_i(t)\}_{i=1}^n$ $t \in \mathcal{T} \subset \mathbb{R}$, $u \in \mathcal{U} \subset \mathbb{R}$, where $Y_i(\cdot)$ is a functional response, explained by a functional predictor $X_i(\cdot)$. The modeling with such FME extension then takes the form $Y_i(t) = \beta_{z_i,0}(t) + \int_t X_i(u)\beta_{z_i}(t,u)du + \varepsilon_i(t)$, to explain the functional response Y by the functional predictor X via the unknown discrete variable z. The particularity with this model is that, for the clustering, as well as for the prediction, we model the relation between Y at any time u and the entire curve of X, or the entire curve of each variable X_{ij} in the case of multivariate functional predictor $\boldsymbol{X}_i$.

Chapter 4

Extensions of functional mixtures of experts

Contents

4.1 Introduction

In this chapter, we have two goals. First, we extend the FME and iFME models proposed in previous chapter to the case of categorical responses, *i.e.*, for multiclass classification problems. Second, we develop ME models to deal with the situations where the responses are functions.

To achieve the first goal, in Section 4.2 we model the experts in FME models by the functional multinomial logistic regression model, and develop the corresponding conditional density functions for the FME and iFME models. The numerical experiments are also presented to illustrate the performance of the extended models.

In Section 4.3, a ME model for *function-on-function* regression is proposed. It can be viewed as an extension of the FME model to the case of functional responses. We refer to it by function-on-function FME model (FF-FME). The model estimation is performed via maximizing a penalized log-likelihood function, with the penalized term is constructed such that estimated kernels are smoothened. An efficient EM-like algorithm is developed to estimate the model parameters. The proposed model is examined on both simulated and real-world data.

Finally, in Section 4.4, to complete the family of FME models, we present a ME model for *function-on-scalar* regression. We refer to this model by function-on-scalar FME model (FS-FME). A penalized MLE is considered, in which the penalization is given to the roughness of the functional parameters. We also develop the corresponding EM-like algorithm for model estimation. The model performance is evaluated on simulated data, which show promising results for other applications.

4.2 FME for classification

In this section, we extend the FME framework for multiclass classification, derive adapted EM-like algorithms to obtain sparse and interpretable fit of the gating and experts network coefficients functions. Let $\{X_i(t), t \in \mathcal{T}; Y_i\}_{i=1}^n$, be a sample of n i.i.d. data pairs where $Y_i \in \{1, \ldots, G\}$ is the class label of a functional predictor $X_i(\cdot)$, G being the number of classes. In this case of functional inputs, a natural choice to model the conditional distribution $\text{Expert}_k(y|x) = \mathbb{P}(Y = y|X_i(\cdot))$ in (2.1) is to use the functional multinomial logistic regression modeling, *e.g.*, see Müller et al. (2005); James (2002), that is

$$P(y_i|X_i(\cdot); \boldsymbol{\beta}_k) = \prod_{g=1}^{G} \left[\frac{\exp\left\{\beta_{kg,0} + \int_{\mathcal{T}} X_i(t)\beta_{kg}(t)dt\right\}}{1 + \sum_{g'=1}^{G-1} \exp\left\{\beta_{kg',0} + \int_{\mathcal{T}} X_i(t)\beta_{kg'}(t)dt\right\}} \right]^{y_{ig}}, \tag{4.1}$$

where $\boldsymbol{\beta}_k$ represents the set of coefficient functions $\{\beta_{kg}(t), t \in \mathcal{T}\}$ and intercepts $\{\beta_{k,0}\}$ for $k \in [K] = \{1, \ldots, K\}$ and $g \in [G]$, and $y_{ig} = \mathbb{I}_{\{y_i=g\}}$.

Similarly, a typical choice for the functional gating network $\text{Gate}_k(x) = \mathbb{P}(Z = k, X(\cdot))$ in (2.1), where $Z \in [K]$ is a hidden within-class clustering label, acting as weights for potential clusters $\{k\}$ in the heterogeneous functional inputs $X(\cdot)$ and which we denote as $\pi_k(X(\cdot))$, is to use a functional softmax function defined by

$$\pi_k(X_i(\cdot); \boldsymbol{\alpha}) = \frac{\exp\{\alpha_{k,0} + \int_{\mathcal{T}} X_i(t)\alpha_k(t)dt\}}{1 + \sum_{k'=1}^{K-1} \exp\{\alpha_{k',0} + \int_{\mathcal{T}} X_i(t)\alpha_{k'}(t)dt\}}, \tag{4.2}$$

with $\boldsymbol{\alpha}$ is composed of the set of coefficient functions $\{\alpha_k(t), t \in \mathcal{T}\}$ and intercepts $\{\alpha_{k,0}\}$ for $k \in [K]$.

Then, from (4.1) and (4.2) given $X_i(\cdot)$, the probability that $Y_i = y_i$, can be modeled by the

following K-component FME model for classification

$$P(y_i|X_i(\cdot);\boldsymbol{\psi}) = \sum_{k=1}^{K} \pi_k(X_i(\cdot);\boldsymbol{\alpha}) P(y_i|X_i(\cdot);\boldsymbol{\beta}_k), \quad \boldsymbol{\psi} = (\boldsymbol{\alpha}, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K). \tag{4.3}$$

where $\boldsymbol{\psi} = (\boldsymbol{\alpha}, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K)$ is the unknown parameter of the model. Here the coefficient functions $\alpha_k(\cdot)$, $\beta_{kg}(\cdot)$ have the roles as the coefficient vectors as in the classic ME model for classification.

4.2.1 Smooth functional representation

In practice, $X_i(\cdot)$ is observed at a finite but large number of points on $\mathcal{T} \subset \mathbb{R}$. In the perspective of parameter estimation, this results in estimating a very large number of coefficients β and α. In order to handle this high-dimensional problem, we consider a usual approach that projects the predictors and coefficient functions onto a family of reduced number of basis functions. Let $\boldsymbol{b}_r(t) = [b_1(t), \ldots, b_r(t)]^\top$ be a r-dimensional basis (B-spline, Wavelet, ...). Then, with $r, p, q \in \mathbb{N}$ sufficiently large, one can approximate $X_i(\cdot)$, $\alpha_k(\cdot)$ and $\beta_{kg}(\cdot)$ respectively by

$$X_i(t) = \boldsymbol{x}_i^\top \boldsymbol{b}_r(t), \quad \alpha_k(t) = \boldsymbol{\zeta}_k^\top \boldsymbol{b}_p(t), \quad \beta_{kg}(t) = \boldsymbol{\eta}_{kg}^\top \boldsymbol{b}_q(t). \tag{4.4}$$

Here, $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{ir})^\top$, with $x_{ij} = \int_{\mathcal{T}} X_i(t) b_j(t) dt$ for $j \in [r]$, is the vector of coefficients of $X_i(\cdot)$ in the basis $\boldsymbol{b}_r(t)$, $\boldsymbol{\zeta}_k = (\zeta_{k,1}, \ldots, \zeta_{k,p})^\top$, and $\boldsymbol{\eta}_{kg} = (\eta_{kg,1}, \ldots, \eta_{kg,q})^\top$ are the unknown coefficient vectors associated with the gating coefficient function $\alpha_k(\cdot)$ and the expert coefficient function $\beta_{kg}(\cdot)$ in the corresponding basis. In our case, we used B-spline bases.

Using the approximation of $X_i(\cdot)$ and $\alpha_k(\cdot)$ in (4.4), the functional softmax gating network (4.2) can be represented by

$$\pi_k(\mathbf{r}_i;\boldsymbol{\xi}) = \frac{\exp\{\alpha_{k,0} + \mathbf{r}_i^\top \boldsymbol{\zeta}_k\}}{1 + \sum_{k'=1}^{K-1} \exp\{\alpha_{k',0} + \mathbf{r}_i^\top \boldsymbol{\zeta}_{k'}\}}, \tag{4.5}$$

where $\mathbf{r}_i = \left[\int_{\mathcal{T}} \boldsymbol{b}_r(t) \boldsymbol{b}_p(t)^\top dt\right]^\top \boldsymbol{x}_i$ is the design vector associated with the gating network and $\boldsymbol{\xi} = ((\alpha_{1,0}, \boldsymbol{\zeta}_1^\top), \ldots, (\alpha_{K-1,0}, \boldsymbol{\zeta}_{K-1}^\top)) \in \mathbb{R}^{(p+1)(K-1)}$ is the unknown parameter vector of the gating network, to be estimated.

In the same manner, using the approximations of $X_i(\cdot)$ and $\beta_{kg}(\cdot)$ in (4.4), the expert conditional distribution (4.1) can be represented by

$$P(y_i|\mathbf{x}_i;\boldsymbol{\theta}_k) = \prod_{g=1}^{G} \left[\frac{\exp\left\{\beta_{kg,0} + \mathbf{x}_i^\top \boldsymbol{\eta}_{kg}\right\}}{1 + \sum_{g'=1}^{G-1} \exp\left\{\beta_{kg',0} + \mathbf{x}_i^\top \boldsymbol{\eta}_{kg'}\right\}}\right]^{y_{ig}}, \tag{4.6}$$

where $\mathbf{x}_i = \left[\int_{\mathcal{T}} \boldsymbol{b}_r(t) \boldsymbol{b}_q(t)^\top dt\right]^\top \boldsymbol{x}_i$ is the design vector associated with the expert network, and $\boldsymbol{\theta}_k = (\boldsymbol{\theta}_{k1}^\top, \ldots, \boldsymbol{\theta}_{k,G-1}^\top)^\top$, with $\boldsymbol{\theta}_{kg} = (\beta_{kg,0}, \boldsymbol{\eta}_{kg}^\top)^\top \in \mathbb{R}^{q+1}$ for $g \in [G-1]$, is the unknown parameter vector to be estimated of the expert distribution k.

Finally, combining (4.5) and (4.6), the conditional distribution $P(y_i|X_i(\cdot);\boldsymbol{\psi})$ in (4.3) can be

rewritten as

$$P(y_i|X_i(\cdot);\boldsymbol{\Psi}) = \sum_{k=1}^{K} \pi_k(\mathbf{r}_i;\boldsymbol{\xi}) P(y_i|\mathbf{x}_i;\boldsymbol{\theta}_k),$$

where $\boldsymbol{\Psi} = (\boldsymbol{\xi}^\top, \boldsymbol{\theta}_1^\top, \ldots, \boldsymbol{\theta}_K^\top)^\top$ is the unknown parameter vector of the model.

4.2.2 Parameter estimation via EM algorithm

A maximum likelihood estimate (MLE) $\widehat{\boldsymbol{\Psi}}$ of $\boldsymbol{\Psi}$ can be obtained by using the EM algorithm for ME model for classification with vector data as in Chen et al. (1999). We will refer to this approach as FME-EM.

Similarly to the FME model, to encourage sparsity in the model parameters $\boldsymbol{\Psi}$, one can perform penalized MLE by using the EM-Lasso algorithm as in Huynh and Chamroukhi (2019b). We refer to this approach as FME-EM-Lasso.

4.2.3 Interpretable FME for classification

Although fitting the FME model via EM-Lasso can accommodate sparsity in the parameters, it unfortunately does not ensure the reconstructed coefficient functions $\widehat{\alpha}_k(\cdot)$ and $\widehat{\beta}_{kg}(\cdot)$ are sparse and enjoy easy interpretable sparsity.

To obtain interpretable and sparse fits for the coefficient functions, we simultaneously estimate the model parameters while constraining some targeted derivatives of the coefficient functions to be zero (Chamroukhi et al., 2022). The construction of the interpretable FME model which we will fit with an adapted EM algorithm, is as follows.

First, in order to calculate the derivative of the gating coefficient functions $\alpha_k(\cdot)$, let $\mathbf{A}_p$ be the matrix of approximate d_1th and d_2th derivative of $\boldsymbol{b}_p(t)$, defined as in (3.22) by

$$\mathbf{A}_p \;=\; [\mathbf{A}_p^{[d_1]}\mathbf{A}_p^{[d_2]}]^\top = \left[D^{d_1}\boldsymbol{b}_p(t_1), \ldots, D^{d_1}\boldsymbol{b}_p(t_p), D^{d_2}\boldsymbol{b}_p(t_1), \ldots, D^{d_2}\boldsymbol{b}_p(t_p)\right]^\top,$$

where D^d is the dth finite difference operator. Here $\mathbf{A}_p^{[d_j]}$ is a square invertible matrix and $\mathbf{A}_p \in \mathbb{R}^{2p\times p}$. Similarly, to calculate the derivatives of the expert coefficient functions $\beta_{kg}(\cdot)$, let $\mathbf{A}_q = [\mathbf{A}_q^{[d_1]}\mathbf{A}_q^{[d_2]}]^\top \in \mathbb{R}^{2q\times q}$ be the corresponding matrix defined for the $\boldsymbol{b}_q(t)$'s.

Now, if we define $\boldsymbol{\omega}_k = \mathbf{A}_p\boldsymbol{\zeta}_k$ and denote $\boldsymbol{\omega}_k = ({\boldsymbol{\omega}_k^{[d_1]}}^\top, {\boldsymbol{\omega}_k^{[d_2]}}^\top)^\top$, then $\boldsymbol{\omega}_k^{[d_1]}$ and $\boldsymbol{\omega}_k^{[d_2]}$ provide approximations to the d_1 and the d_2 derivatives of the coefficient function $\alpha_k(\cdot)$, respectively, which we denote as $\alpha_k^{(d_1)}(\cdot)$ and $\alpha_k^{(d_2)}(\cdot)$. Therefore, enforcing sparsity in $\boldsymbol{\omega}_k$ will constrain $\alpha_k^{(d_1)}(\cdot)$ and $\alpha_k^{(d_2)}(\cdot)$ to be zero at most of time points. Similarly, if we define $\boldsymbol{\gamma}_{kg} = \mathbf{A}_q\boldsymbol{\zeta}_k$ and denote by $\boldsymbol{\gamma}_{kg} = ({\boldsymbol{\gamma}_{kg}^{[d_1]}}^\top, {\boldsymbol{\gamma}_{kg}^{[d_2]}}^\top)^\top$, then we can derive the same regularization for the coefficient functions $\beta_{kg}(\cdot)$.

From the definitions of $\boldsymbol{\omega}_k$ and $\boldsymbol{\gamma}_{kg}$ we can easily get the following relations:

$$\begin{cases} \boldsymbol{\zeta}_k = \mathbf{A}_p^{[d_1]^{-1}} \boldsymbol{\omega}_k^{[d_1]} \text{ and } \boldsymbol{\omega}_k^{[d_2]} = \mathbf{A}_p^{[d_2]} \mathbf{A}_p^{[d_1]^{-1}} \boldsymbol{\omega}_k^{[d_1]} & \text{(4.7a)} \\ \boldsymbol{\eta}_{kg} = \mathbf{A}_q^{[d_1]^{-1}} \boldsymbol{\gamma}_{kg}^{[d_1]} \text{ and } \boldsymbol{\gamma}_{kg}^{[d_2]} = \mathbf{A}_q^{[d_2]} \mathbf{A}_q^{[d_1]^{-1}} \boldsymbol{\gamma}_{kg}^{[d_1]}. & \text{(4.7b)} \end{cases}$$

Plugging the relation (4.7a) into (4.5) one gets the following new representation for $\pi_k(\mathbf{r}_i; \boldsymbol{\xi})$

$$\pi_k(\mathbf{s}_i; \mathbf{w}) = \frac{\exp\left\{\alpha_{k,0} + \mathbf{s}_i^\top \boldsymbol{\omega}_k^{[d_1]}\right\}}{1 + \sum_{k'=1}^{K-1} \exp\left\{\alpha_{k',0} + \mathbf{s}_i^\top \boldsymbol{\omega}_{k'}^{[d_1]}\right\}}, \tag{4.8}$$

where $\mathbf{s}_i = (\mathbf{A}_p^{[d_1]^{-1}})^\top \mathbf{r}_i$ is now the new design vector and $\mathbf{w} = (\alpha_{1,0}, \boldsymbol{\omega}_1^{[d_1]^\top}, \ldots, \alpha_{K-1,0}, \boldsymbol{\omega}_{K-1}^{[d_1]^\top})^\top$, with $(\alpha_{K,0}, \boldsymbol{\omega}_K^{[d_1]^\top})^\top$ a null vector, is the unknown parameter vector of the gating network.

Similarly, plugging (4.7b) into (4.6) one obtains the new representation for $P(y_i|\mathbf{x}_i; \boldsymbol{\theta}_k)$:

$$P(y_i|\mathbf{v}_i; \boldsymbol{\Gamma}_k) = \prod_{g=1}^{G} \left[\frac{\exp\left\{\beta_{kg,0} + \mathbf{v}_i^\top \boldsymbol{\gamma}_{kg}^{[d_1]}\right\}}{1 + \sum_{g'=1}^{G-1} \exp\left\{\beta_{kg',0} + \mathbf{v}_i^\top \boldsymbol{\gamma}_{k'g}^{[d_1]}\right\}} \right]^{y_{ig}}, \tag{4.9}$$

in which, $\mathbf{v}_i = (\mathbf{A}_q^{[d_1]^{-1}})^\top \mathbf{x}_i$ is now the new design vector and $\boldsymbol{\Gamma}_k = (\beta_{kg,0}, \boldsymbol{\gamma}_{k'g}^{[d_1]^\top})^\top$ is the unknown parameter vector of the expert network. Finally, gathering the gating network (4.8) and the expert network (4.9), the iFME model for classification is given by

$$P(y_i|X_i(\cdot); \boldsymbol{\Upsilon}) = \sum_{k=1}^{K} \pi_k(\mathbf{s}_i; \mathbf{w}) P(y_i|\mathbf{v}_i; \boldsymbol{\Gamma}_k), \tag{4.10}$$

where $\boldsymbol{\Upsilon} = (\mathbf{w}^\top, \boldsymbol{\Gamma}_1^\top, \ldots, \boldsymbol{\Gamma}_K^\top)^\top$ is the unknown parameter vector to be estimated.

We perform penalized MLE by penalizing the ML via a Lasso penalization on the derivative coefficients $\boldsymbol{\omega}_k$'s and $\boldsymbol{\gamma}_{kg}$'s of the form $\text{Pen}_{\chi,\lambda}(\boldsymbol{\Upsilon}) = \chi \sum_{k=1}^{K-1} \|\boldsymbol{\omega}_k\|_1 + \lambda \sum_{k=1}^{K} \sum_{g=1}^{G-1} \|\boldsymbol{\gamma}_{kg}\|_1$, with χ and λ regularization constants. The estimation is performed by using an adaptation to this classification context of the EM algorithm developed in Chamroukhi et al. (2022). The only difference resides in the maximization with respect to the expert network parameters $\boldsymbol{\Gamma}_k$.

4.2.4 Numerical experiments

■ Experiments on simulated data

We conducted experiments by considering a 3-class classification problem (*i.e.*, $G = 3$) with a $K = 2$-component FME model. In particular, we generate $n = 1000$ functions $X_i(\cdot)$ using $X_i(t) = \boldsymbol{x}_i^\top \boldsymbol{b}_r(t)$, with $\boldsymbol{x}_i = \boldsymbol{W}\boldsymbol{v}_i$, $\boldsymbol{W} \in \mathbb{R}^{8\times 8}$ is a matrix of i.i.d. random values from the uniform distribution $\mathcal{U}(0,1)$, $\boldsymbol{v}_i \in \mathbb{R}^8$ is a vector of i.i.d. random values from the normal distribution $\mathcal{N}(1,10)$, and $\boldsymbol{b}_r(t)$ is the 8-dimensional B-spline basis defined on $[0,1]$. Then, Y_i is generated conditional on $X_i(\cdot)$ and the true coefficient functions. The time domain is divided into a grid of

$m = 100$ evenly spaced points. To mimic real-world data, for each $i \in [n]$, we add to $X_i(t_j), j \in [m]$, measurement error $\delta_i(t_j) \sim \mathcal{N}(0, \sigma_\delta^2)$, with σ_δ^2 the noise level. We consider two situations: $\sigma_\delta^2 = 1$ and $\sigma_\delta^2 = 5$. We compare our proposed models, including FME fitted by EM, FME fitted by EM-Lasso, and iFME-EM model, with the classic functional multinomial logistic regression (FMLR) model that receives the coefficient vectors $\boldsymbol{x}_i$'s as inputs.

Each dataset is split into training and testing set of equal sizes. Parameters are fits on training set and the correct classification rates are evaluated on testing set. This procedure is repeated 100 times.

The classification results obtained with the described algorithms FME-EM, FME-EM-Lasso and iFME-EM, as well as with functional multinomial logistic regression (FMLR), are given in Table 4.1 and show higher classification performance of the iFME-EM approach.

Model	Correct classification rate	
	Noise level: $\sigma_\delta^2 = 1$	Noise level: $\sigma_\delta^2 = 5$
FME-EM	$.8560_{(.0199)}$	$.8474_{(.0196)}$
FME-EM-Lasso	$.9332_{(.0104)}$	$.9178_{(.0142)}$
iFME-EM	$\mathbf{.9346_{(.0108)}}$	$\mathbf{.9219_{(.0127)}}$
FMLR	$.7951_{(.0249)}$	$.7922_{(.0270)}$

Table 4.1: Correct classification rates obtained on testing data. The reported values are averages on 100 samples with standard errors in parentheses.

■ Application to phonemes data

We then applied the two algorithms allowing for sparsity (FME-EM-Lasso and iFME-EM) to the well-known phoneme data (Hastie et al., 1995). The data consists of $n = 1000$ log-periodogram recordings of length 256 each, used here as the univariate functional predictors, of five phonemes transcribed (*i.e.*, the corresponding class labels): "*sh*" as in *she*, "*dcl*" as in *dark*, "*iy*" as the vowel in *she*, "*aa*" as the vowel in dark, and "*ao*" as the first vowel in *water*.

The obtained averaged correct classification rate for the two approaches are more than 0.94 in mean, which are competitive with the correct rate of 0.929 resulted by the wavelet-based multinomial functional regression (MFR) model (Mousavi and Sørensen, 2017). Figure 4.1 shows the estimated coefficient functions for the gating network $\widehat{\alpha}_k(t)$ (left) and those for the expert network $\widehat{\beta}_{kg}(t)$ as functions of sampling time t, obtained by FME-EM-Lasso (top) and the iFME-EM (bottom). Here the iFME-EM is fitted with constraints on the zero and the second derivatives of the coefficients functions. The results show clearly sparse and piece-wise-linear gating and experts functions when using the iFME-EM approach.

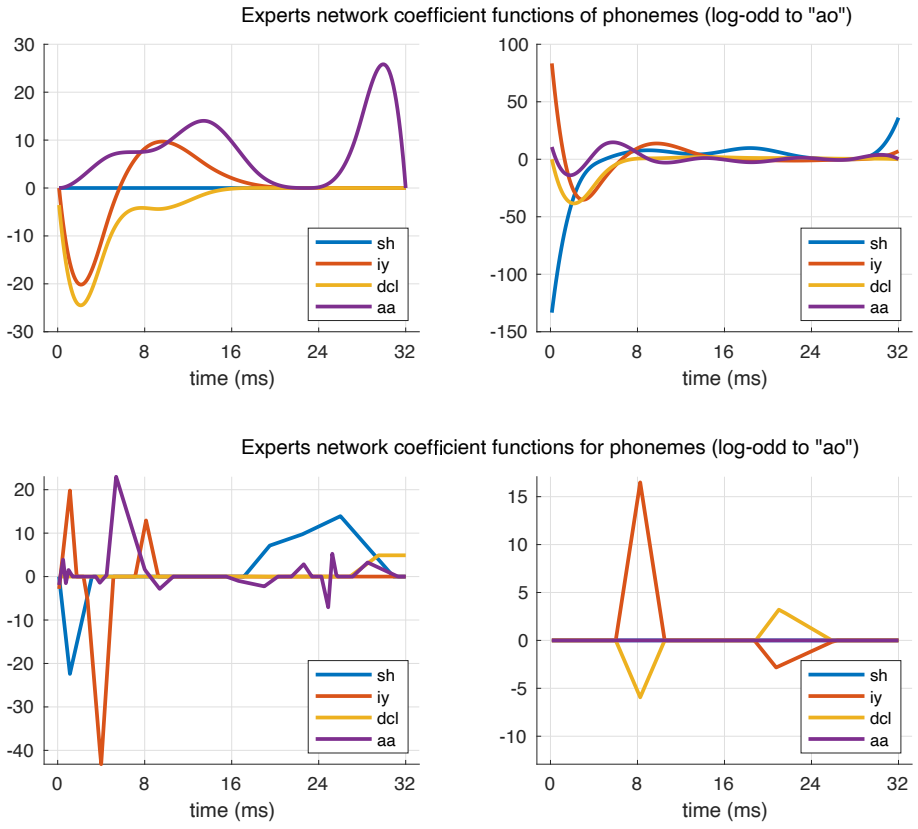


Figure 4.1: Results of (top) FME-EM-Lasso and (bottom) iFME-EM on phoneme data.

4.3 FME for function-on-function regression

In this section, we extend the FME model proposed in **Chapter 3** to the case where the responses are functions, *i.e.*, to relate input and output when they are both functions. In particular, we assume that in the population, the relationship between the functional output and the functional input is governed by some kernels that vary among the subpopulations of the input space. We refer to this proposed model as function-on-function FME model (FF-FME).

This model can also be viewed as an extension of the mixture model for functional regression proposed in Devijver (2015), where the author considered a mixture of functional regression models to deal with function-on-function regression problems. In Devijver (2015), the model parameters are estimated using MLE and penalized MLE approaches, and the performance was evaluated on both simulated and real functional datasets. In our proposed FF-FME model, similar to the way the mixture of experts regression model extends the mixture model, we allow the mixing proportions to vary across predictors via a functional gating network, so that the model can potentially capture a more complex structure in the population.

This section begins with the construction of the FF-FME model. Then the model estimation via EM algorithm is presented. Finally, the numerical experiments on both simulated and real functional data are presented to illustrate the use of the proposed model.

4.3.1 The FF-FME model

Let $\{X_i(u), Y_i(t)\}_{i=1}^n$ be a sample of n i.i.d. data pairs where $Y_i(t)$, $t \in \mathcal{T}$, is a functional response, and $X_i(u)$, $u \in \mathcal{U}$, is a functional predictor, where $\mathcal{T}$ and $\mathcal{U}$ are some intervals on $\mathbb{R}$. An example of such context is: $Y_i(t)$ is the demand of electricity, and $X_i(u)$ is the temperature during the day at a certain house or city. In this case, both the response and the predictor are time series and $\mathcal{U}$ and $\mathcal{T}$ represent the time, *e.g.*, from 0h to 24h.

Firstly, we formulate each expert component $\text{Expert}_k(y|x)$ in (2.1) by a fully functional linear model as in (2.28). In particular, the regression model for the ith observation takes the following stochastic representation

$$Y_i(t) = \beta_{z_i,0}(t) + \int_{\mathcal{U}} X_i(u)\beta_{z_i}(t,u)du + \varepsilon_i(t), \quad i \in [n], \tag{4.11}$$

where $\beta_{z_i,0}(t)$, defined on $\mathcal{T}$, is the unknown functional intercept, $\beta_{z_i}(t,u)$, defined on $\mathcal{T}\times\mathcal{U}$, is the unknown kernel (*i.e.*, unknown bivariate function) associated with expert z_i to be estimated, and $\varepsilon_i(t)$ is an independent random error function follows a normal distribution with mean zero and covariance function $\Sigma_{z_i}(t,u)$. Here $z_i \in [K]$ denotes the unknown label of the expert responsible for the ith observation. However, the appearance of $\beta_{z_i,0}(t)$ in the model can be simplified by assuming that the functional predictors and responses are all centered (*e.g.*, as mentioned in Section 2.7). Therefore, for simplicity, from now on we take this assumption and denote by

$\boldsymbol{\beta} = \{\beta_1(\cdot,*), \ldots, \beta_K(\cdot,*)\}$ the set of K unknown kernels of the expert network to be estimated.

Modeling the functional gating network

The modeling for the functional gating network is unchanged compared to the FME model. That is, we also model the gating network by the functional logistic regression model for linear classification (Mousavi and Sørensen, 2017), with the note that the intercept $\alpha_{z,0}$ is now simplified. The resulting functional softmax gating network is then given by

$$\pi_k\left(X_i(u), u \in \mathcal{U}; \boldsymbol{\alpha}\right) = \frac{\exp\{\int_{\mathcal{U}} X_i(u)\alpha_k(u)du\}}{1 + \sum_{k'=1}^{K-1} \exp\{\int_{\mathcal{U}} X_i(u)\alpha_{k'}(u)du\}}, \tag{4.12}$$

where $\alpha_k(u)$ is the unknown functional parameter of gating network, and $\boldsymbol{\alpha} = \{\alpha_k(u),\, u \in \mathcal{U}\}_{k=1}^{K-1}$ is the set of unknown functional parameters $\alpha_k(u)$, $u \in \mathcal{U}$ to be estimated. Again, the function $\alpha_K(u)$ is necessarily zero to satisfy the identifiability condition.

Analogous to the construction for the finite representation of the functional gating network as in (3.10) or in (4.5), here we can also obtain such representation for (4.12). In particular, let $\boldsymbol{\omega}(u) = [\omega_1(u), \ldots, \omega_q(u)]^\top$ be a vector of basis functions defined on $\mathcal{U}$. Then with q sufficiently large, we can approximate $\alpha_k(u)$ by

$$\alpha_k(u) = \boldsymbol{\zeta}_k^\top \boldsymbol{\omega}(u), \tag{4.13}$$

where $\boldsymbol{\zeta}_k \in \mathbb{R}^q$ is the unknown vector of coefficients of the functional gating parameter $\alpha_k(u)$. Substituting (4.13) into (4.12) we obtain a new representation for the gating network:

$$\pi_k\left(\mathbf{r}_i; \boldsymbol{\xi}\right) = \frac{\exp(\boldsymbol{\zeta}_k^\top \mathbf{r}_i)}{1 + \sum_{k'=1}^{K-1} \exp(\boldsymbol{\zeta}_k^\top \mathbf{r}_i)}, \tag{4.14}$$

where $\mathbf{r}_i = \left(\int_U X_i(u)\omega_1(u)du, \ldots, \int_U X_i(u)\omega_q(u)du\right)^\top \in \mathbb{R}^q$ is the new design vector, and $\boldsymbol{\xi} = (\boldsymbol{\zeta}_1^\top, \ldots, \boldsymbol{\zeta}_{K-1}^\top)^\top \in \mathbb{R}^{q(K-1)}$ is now the unknown parameter vector to be estimated.

Modeling the functional expert network

It is known that in general there is no notion of probability density for functional data (Delaigle and Hall, 2010), so in particular in our case it is not obvious to write down a conditional density function for the functional responses $Y_i(t)$ as in (3.13). However, as can be seen in Section 2.7, using projections onto appropriate bases, and assuming a normal distribution for the coefficients of the error functions, we can also introduce the conditional density function for the FF-FME model analogously to (3.13) as follows.

Let $\boldsymbol{\omega}^\star(t,u) = [\omega_1^\star(t,u), \ldots, \omega_M^\star(t,u)]^\top$ be a vector of bivariate M basis functions defined on $\mathcal{T} \times \mathcal{U}$. Then with M sufficiently large, the bivariate functional parameter $\beta_k(t,u)$ can be expressed

by

$$\beta_k(t,u) = \sum_{m=1}^{M} b_{km}\omega_m^\star(t,u) =: \boldsymbol{b}_k^\top \boldsymbol{\omega}^\star(t,u), \tag{4.15}$$

where $\boldsymbol{b}_k = (b_{k1},\ldots,b_{kM})^\top \in \mathbb{R}^M$ is the vector of unknown coefficients associated with the expert kth, to be estimated. Recall that, analogous to the equations related to basis expansions stated throughout the thesis, the equation (4.15) is only approximation, but we use the equality sign "=" to be able to manipulate such expansion without adding additional error terms.

The stochastic representation (4.11), with the intercept has been simplified as discussed earlier and with $z_i = k$, can be rewritten as

$$\begin{aligned} Y_i(t) &= \int_{\mathcal{U}} X_i(u) \left(\sum_{m=1}^{M} b_{km}\omega_m^\star(t,u) \right) du + \varepsilon_i(t) \\ &= \sum_{m=1}^{M} b_{km} \int_{\mathcal{U}} X_i(u)\omega_m^\star(t,u)du + \varepsilon_i(t) \\ &= \sum_{m=1}^{M} b_{km} X_{im}^\star(t) + \varepsilon_i(t), \quad i \in [n], \end{aligned} \tag{4.16}$$

where $X_{im}^\star(t) := \int_{\mathcal{U}} X_i(u)\omega_m^\star(t,u)du$ are functions of t which now serve as new predictors.

Let $\boldsymbol{\omega}(t) = [\omega_1(t),\ldots,\omega_q(t)]^\top$ be a vector of basis functions defined on $\mathcal{T}$. Then with q sufficiently large, the functions $Y_i(t)$, $X_{im}^\star(t)$ and $\varepsilon_i(t)$ can also be represented by

$$Y_i(t) = \sum_{\ell=1}^{q} y_{i\ell}\omega_\ell(t) =: \boldsymbol{y}_i^\top \boldsymbol{\omega}(t), \tag{4.17a}$$

$$X_{im}^\star(t) = \sum_{\ell=1}^{q} x_{im\ell}^\star \omega_\ell(t) =: \boldsymbol{x}_{im}^{\star\top} \boldsymbol{\omega}(t), \tag{4.17b}$$

$$\varepsilon_i(t) = \sum_{\ell=1}^{q} \varepsilon_{i\ell}\omega_\ell(t) =: \boldsymbol{\epsilon}_i^\top \boldsymbol{\omega}(t), \tag{4.17c}$$

respectively, where $\boldsymbol{y}_i$, $\boldsymbol{x}_{im}^\star$ and $\boldsymbol{\epsilon}_i$ belong to $\mathbb{R}^q$ are the coefficient vectors, and $\boldsymbol{\varepsilon}_i \sim \mathcal{N}_q(\mathbf{0}, \boldsymbol{\Sigma})$ for some unknown symmetric matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{q\times q}$.

Note that the matrix $\boldsymbol{\omega}(t)\boldsymbol{\omega}^\top(t)$ is non-singular, therefore by substituting equations (4.17) into (4.16) and simplifying $\boldsymbol{\omega}(t)$, we obtain a new stochastic representation for the model:

$$\begin{aligned} \boldsymbol{y}_i &= \sum_{m=1}^{M} b_{km}\boldsymbol{x}_{im}^\star + \boldsymbol{\epsilon}_i \\ &= \boldsymbol{X}_i^{\star\top}\boldsymbol{b}_k + \boldsymbol{\epsilon}_i, \quad i \in [n], \end{aligned}$$

where $\boldsymbol{X}_i^\star$ denotes the matrix $[\boldsymbol{x}_{i1}^\star,\ldots,\boldsymbol{x}_{iM}^\star]^\top \in \mathbb{R}^{M\times q}$. From this representation and the Gaussian assumption for the error $\boldsymbol{\varepsilon}_i$, the conditional density of each approximated functional expert $z_i = k$

is then given by

$$f(\boldsymbol{y}_i|X_i(\cdot), z_i = k; \boldsymbol{\theta}_k) = \phi(\boldsymbol{y}_i; \boldsymbol{X}_i^{\star\top}\boldsymbol{b}_k, \boldsymbol{\Sigma}_k), \tag{4.18}$$

where $\phi(\cdot; \boldsymbol{\mu}, \boldsymbol{\nu})$ is the pdf of the q-dimensional Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\nu}$, and $\boldsymbol{\theta}_k = (\boldsymbol{b}_k^\top, \text{vech}^\top \boldsymbol{\Sigma}_k)^\top$ is the unknown parameter vector of the expert k, $k \in [K]$, to be estimated. Here, vech$(\cdot)$ is the operator that extracts the unique elements of a symmetric matrix (Henderson and Searle, 1979).

Conditional density function of the FF-FME model

Finally, combining the gating network (4.14) and the expert density (4.18) leads to the following conditional density function for our FF-FME model

$$f(\boldsymbol{y}_i|X_i(u); \boldsymbol{\Psi}) = \sum_{k=1}^{K} \pi_k(\mathbf{r}_i; \boldsymbol{\xi})\phi(\boldsymbol{y}_i; \boldsymbol{X}_i^{\star\top}\boldsymbol{b}_k, \boldsymbol{\Sigma}_k), \tag{4.19}$$

where $\boldsymbol{\Psi} = (\boldsymbol{\xi}^\top, \boldsymbol{\theta}_1^\top, \ldots, \boldsymbol{\theta}_K^\top)^\top$ is the parameter vector of the model to be estimated.

There are some remarks on the constructions of the design vectors in practice. First, although the formulas are written using the functions $X_i(u)$ and $Y_i(t)$, in practice they are only available in form of discretized values. However, for simplicity we use the same notations to indicate the noisy version. Second, as seen in the previous models, all the involving integrals here can be calculated numerically by a Riemann sum over the grid where data was sampled. Finally, in practice the design vectors $\boldsymbol{x}_{im}^\star$ can be calculated directly from the predictors $X_i(t)$ and the basis responsible for covariate u that construct $\boldsymbol{\omega}^\star(t, u)$, as we have seen in Section 2.7. For example, if the bivariate basis $\boldsymbol{\omega}^\star(t, u)$ is constructed by two B-spline bases $\boldsymbol{\omega}(t)$ and $\boldsymbol{\omega}(u)$, where $\boldsymbol{\omega}(u) = [\omega_1(u), \ldots, \omega_q(u)]^\top$, then $\boldsymbol{x}_{im}^\star$ is simply

$$\boldsymbol{x}_{im}^\star = \left[\int_{\mathcal{U}} X_i(u)\omega_1(u)du, \ldots, \int_{\mathcal{U}} X_i(u)\omega_q(u)du\right]^\top \in \mathbb{R}^q.$$

4.3.2 Penalized maximum likelihood estimation via EM algorithm

The FF-FME model (4.19) is now defined upon a tractable finite representation of the functional predictors and responses. In this subsection, we consider the MLE of the model via the EM algorithm (Dempster et al., 1977).

With the maximum likelihood estimation method, the observed-data log-likelihood function for FF-FME model to be maximized is given by

$$\log L(\boldsymbol{\Psi}) = \sum_{i=1}^{n} \log \sum_{k=1}^{K} \pi_k(\mathbf{r}_i; \boldsymbol{\xi})\phi(\boldsymbol{y}_i; \boldsymbol{X}_i^{\star\top}\boldsymbol{b}_k, \boldsymbol{\Sigma}_k),$$

where $\boldsymbol{\Psi} = (\boldsymbol{\xi}^\top, \boldsymbol{\theta}_1^\top, \ldots, \boldsymbol{\theta}_K^\top)^\top$. However, to encourage the smoothness of the bivariate functions

$\beta_k(t,u)$, we propose to consider the following penalized log-likelihood function

$$\mathcal{L}(\boldsymbol{\Psi}) = \log L(\boldsymbol{\Psi}) - \text{Pen}_\chi(\boldsymbol{\Psi}) - \text{Pen}_\lambda(\boldsymbol{\Psi}).$$

In the above definition, $\text{Pen}_\chi(\boldsymbol{\Psi})$ is the penalty term with respect to the gating network, defined by

$$\text{Pen}_\chi(\boldsymbol{\Psi}) = \chi \sum_{k=1}^{K-1} \|\boldsymbol{\xi}_k\|_1,$$

and $\text{Pen}_\lambda(\boldsymbol{\Psi})$ is a roughness penalty term for the expert network, defined by

$$\text{Pen}_\lambda(\boldsymbol{\Psi}) = \sum_{k=1}^{K} \left[\lambda_t \iint \left[(\mathcal{D}_t\beta_k)(t,u)\right]^2 dtdu + \lambda_u \iint \left[(\mathcal{D}_u\beta_k)(t,u)\right]^2 dtdu \right], \tag{4.20}$$

where $\mathcal{D}_t$ and $\mathcal{D}_u$ are the second-order partial derivatives with respect to t and u directions, and χ, λ_t, λ_u are the tuning parameters. If $\chi = \lambda_t = \lambda_u = 0$, the penalty term does not contribute to $\mathcal{L}(\boldsymbol{\Psi})$, we end up with a maximum likelihood estimation. Note that, the dependence of $\text{Pen}_\lambda(\boldsymbol{\Psi})$ on the model parameter $\boldsymbol{\Psi}$ is understood implicitly, since the bivariate functions $(\mathcal{D}_t\beta_k)(t,u)$ and $(\mathcal{D}_u\beta_k)(t,u)$ can be expressed in terms of the coefficient vector $\boldsymbol{b}_k$ as we have seen in (2.35) and (2.36). The two steps of the EM algorithm for FF-FME model are performed as follows.

E-step. The conditional probability memberships $\tau_{ik}^{(s)}$ that the observed pair $(X_i(u), Y_i(t))$ originates from the kth expert at EM iteration sth is calculated by

$$\tau_{ik}^{(s)} = \mathbb{P}(z_i = k | X_i(u), Y_i(t); \widehat{\boldsymbol{\Psi}}^{(s)}) = \frac{\pi_k(\mathbf{r}_i; \widehat{\boldsymbol{\xi}}^{(s)}) \phi(\boldsymbol{y}_i; \boldsymbol{X}_i^{\star\top} \widehat{\boldsymbol{b}}_k^{(s)}, \widehat{\boldsymbol{\Sigma}}_k^{(s)})}{f(\boldsymbol{y}_i | X_i(*); \widehat{\boldsymbol{\Psi}}^{(s)})}, \tag{4.21}$$

where $\widehat{\boldsymbol{\Psi}}^{(s)} = (\widehat{\boldsymbol{\xi}}^{(s)\top}, \widehat{\boldsymbol{\theta}}_1^{(s)\top}, \ldots, \widehat{\boldsymbol{\theta}}_K^{(s)\top})^\top$ is the estimate of $\boldsymbol{\Psi}$ at iteration sth of the EM algorithm.

M-step. The maximization is performed by separately with respect to the gating network parameter $\boldsymbol{\xi}$ and the expert network parameters $\boldsymbol{\theta}_k$ for each of K experts.

First, the maximization with respect to $\boldsymbol{\xi}$ for such a functional gating function has been done earlier in this thesis, for example in (3.16) via the MLE, or in (3.20) via the regularized MLE. Therefore, we refer to the *updating gating network parameters* part given in Appendix B.1 for the maximization with respect to $\boldsymbol{\xi}$, with a minor difference that in FF-FME model we have simplified the intercept $\alpha_{k,0}$.

The maximization with respect to $\boldsymbol{\theta}_k$, on the other hand, consists of solving a weighted function-on-function regression problem, where the weights are the conditional expert memberships $\tau_{ik}^{(s)}$ given in (4.21). More specifically, if penalized MLE is considered, then by expressing the penalized log-likelihood function $\mathcal{L}(\boldsymbol{\Psi})$ in terms of the parameters of the expert kth, *i.e.*, $\boldsymbol{b}_k$ and $\boldsymbol{\Sigma}_k$, the maximization with respect to $\boldsymbol{b}_k$ can be recognized as a penalized least square problem. Therefore,

we can take advantage from the solution formula for the function-on-function regression model presented in Subsection 2.7.2.

In particular, let us denote $\boldsymbol{Y} = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n]^\top$, $\boldsymbol{\tau}_k^{(s)} = (\tau_{k1}^{(s)}, \ldots, \tau_{kn}^{(s)})^\top$. The update formulas for the parameters of expert k for the next EM iteration are then given by

$$\widehat{\boldsymbol{b}}_k^{(s+1)} = \left(\boldsymbol{V}_\tau^\top \boldsymbol{V}_\tau + \lambda_t \boldsymbol{D}_t + \lambda_u \boldsymbol{D}_u\right)^{-1} \boldsymbol{V}_\tau^\top \mathrm{vec}(\boldsymbol{I}_\omega^{1/2} \boldsymbol{Y}_\tau^\top), \tag{4.22}$$

$$\widehat{\boldsymbol{\Sigma}}_k^{(s+1)} = \frac{1}{\sum_{i=1}^n \tau_{ik}^{(s)}} \left(\boldsymbol{Y}_\tau - \boldsymbol{G}_\tau \widehat{\boldsymbol{b}}_k^{(s+1)}\right)^\top \left(\boldsymbol{Y}_\tau - \boldsymbol{G}_\tau \widehat{\boldsymbol{b}}_k^{(s+1)}\right). \tag{4.23}$$

Follows is the definitions of the matrices appear in the above formulas. First, $\boldsymbol{D}_t$ and $\boldsymbol{D}_t$ are the matrices related to the penalty term, defined similarly as in (2.35) and (2.36). Particularly, if we consider $\mathcal{D}_t$ and $\mathcal{D}_u$ the second-order partial derivatives, which is the case in our experimental studies later, $\boldsymbol{D}_t$ and $\boldsymbol{D}_u$ take the following forms

$$\boldsymbol{D}_t = \left[\iint \frac{\partial^2 \omega_g}{\partial t^2}(t,u).\frac{\partial^2 \omega_h}{\partial t^2}(t,u)dtdu\right]_{1\leqslant g,h\leqslant M} \in \mathbb{R}^{M\times M},$$

$$\boldsymbol{D}_u = \left[\iint \frac{\partial^2 \omega_g}{\partial u^2}(t,u).\frac{\partial^2 \omega_h}{\partial u^2}(t,u)dtdu\right]_{1\leqslant g,h\leqslant M} \in \mathbb{R}^{M\times M}.$$

Next, $\boldsymbol{I}_\omega^{1/2}$ is the principal square root of the matrix $\boldsymbol{I}_\omega$ given by

$$\boldsymbol{I}_\omega = \left[\int_{\mathcal{T}} \omega_g(t)\omega_h(t)dt\right]_{1\leqslant g,h\leqslant q} \in \mathbb{R}^{q\times q}.$$

The matrix $\boldsymbol{Y}_\tau$ denotes the weighted version of $\boldsymbol{Y}$, *i.e.*, $\boldsymbol{Y}_\tau = \sqrt{\boldsymbol{\tau}_k^{(s)}} \odot \boldsymbol{Y}$, where the square root is componentwise and $\odot$ is the Hadamard product operator. The matrix $\boldsymbol{V}_\tau$ is the weighted design matrix given by

$$\boldsymbol{V}_\tau = \left(\sqrt{\boldsymbol{\tau}_k^{(s)}} \otimes \mathbb{1}_q\right) \odot \begin{bmatrix} \boldsymbol{I}_\omega^{1/2}\boldsymbol{x}_{11}^\star & \boldsymbol{I}_\omega^{1/2}\boldsymbol{x}_{12}^\star & \cdots & \boldsymbol{I}_\omega^{1/2}\boldsymbol{x}_{1M}^\star \\ \boldsymbol{I}_\omega^{1/2}\boldsymbol{x}_{21}^\star & \boldsymbol{I}_\omega^{1/2}\boldsymbol{x}_{22}^\star & \cdots & \boldsymbol{I}_\omega^{1/2}\boldsymbol{x}_{2M}^\star \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{I}_\omega^{1/2}\boldsymbol{x}_{n1}^\star & \boldsymbol{I}_\omega^{1/2}\boldsymbol{x}_{n2}^\star & \cdots & \boldsymbol{I}_\omega^{1/2}\boldsymbol{x}_{nM}^\star \end{bmatrix} \in \mathbb{R}^{nq\times M},$$

where $\otimes$ is the Kronecker product operator. Finally, $\boldsymbol{G}_\tau$ is the weighted design matrix given by

$$\boldsymbol{G}_\tau = \sqrt{\boldsymbol{\tau}_k^{(s)}} \odot \begin{bmatrix} \boldsymbol{x}_{11}^{\star\top} & \boldsymbol{x}_{12}^{\star\top} & \cdots & \boldsymbol{x}_{1M}^{\star\top} \\ \boldsymbol{x}_{21}^{\star\top} & \boldsymbol{x}_{22}^{\star\top} & \cdots & \boldsymbol{x}_{2M}^{\star\top} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{x}_{n1}^{\star\top} & \boldsymbol{x}_{n2}^{\star\top} & \cdots & \boldsymbol{x}_{nM}^{\star\top} \end{bmatrix} \in \mathbb{R}^{n\times Mq},$$

and the matrix multiplication $\boldsymbol{G}_\tau \widehat{\boldsymbol{b}}_k^{(s+1)}$ in equation (4.23) is performed by treating each $\boldsymbol{x}_{ij}^{\star\top}$, $i \in [n], j \in [M]$, as an element, the same way as discussed in (2.34).

We can see, the right hand sides of (4.22) and (4.23) depend on the current iteration of the EM algorithm implicitly via the posterior probabilities $\boldsymbol{\tau}_k^{(s)}$. Finally, if MLE is considered, then (4.22) can also be used to update the parameter $\boldsymbol{b}_k$ by simply setting $\lambda_t = \lambda_u = 0$.

4.3.3 Numerical experiments

In this subsection, we illustrate the use of the proposed FF-FME model on both simulated and real-world data to show its practical utility. For model evaluation, we compute the MISE between the true and the predicted functional responses, which quantifies how well our model was used for predicting. To evaluate the approximation performance, we also compute the MISE between the true and the estimated functional gating networks, as well as between the true and the estimated functional expert networks. Here, the MISE for bivariate functions is simply defined as follow

$$\text{MISE}(\widehat{\beta}(t,u)) = \mathbb{E}\left[\int_{\mathcal{T}}\int_{\mathcal{U}}(\widehat{\beta}(t,u) - \beta^{\star}(t,u))^2 dtdu\right],$$

where $\beta^{\star}(t,u)$ and $\widehat{\beta}(t,u)$ respectively represents the true and the estimated functions. Moreover, we use ARI to evaluate the clustering performance of the model.

■ Experiment on simulated data

The data generating process is as follows. Firstly, we fix two kernel functions $\beta_1(t,u)$, $\beta_2(t,u)$ and a gating function $\alpha_1(t)$ those to be estimated. The domains for both t and u are fixed to be the interval $[0,1]$. Next, we generate the predictors $X_i(u)$ using the formula

$$X_i(u) = \boldsymbol{x}_i^{\top}\boldsymbol{\omega}(u), \quad t \in [0,1],$$

in which $\boldsymbol{\omega}(u)$ is a 5-dimensional B-spline basis, and $\boldsymbol{x}_i \in \mathbb{R}^5$ is the coefficient vector follows the normal distributed $\mathcal{N}_5(0,1)$. Then, the functional responses $Y_i(t)$ are constructed using the usual data generating protocol for ME models. In particular, for each $i \in [n]$, sample Z_i from on the multinomial distribution $\mathcal{M}\Big(1, \big(\pi_1(X_i(t);\boldsymbol{\alpha}), \ldots, \pi_K(X_i(t);\boldsymbol{\alpha})\big)\Big)$, then conditional on $Z_i = z_i$, the response $Y_i(t)$ is generated by

$$Y_i(t_j) = \int_0^1 X_i(u)\beta_{z_i}(t_j,u)du + \varepsilon_i(t_j), \quad i \in [n],$$

where $\varepsilon_i(t_j) \sim \mathcal{N}(0,1)$ at all time points t_j. Furthermore, to mimic real-world data, we contaminate the predictors $X_i(u)$ with measurement noises before performing model estimation. That means we will not use the $X_i(u)$, but

$$\widetilde{X}_i(u) = X_i(u) + \delta_i(u),$$

for analysis. Here, $\delta_i(u)$ is an independent Gaussian noise, and for all u_j, $\delta_i(u_j)$ follows the distribution $\mathcal{N}(0, 0.04)$.

We generated 200 pairs $(U_i(u), Y_i(t))$ for training and another sample of 200 pairs for evaluation.

For illustration, Figure 4.2 displays some randomly taken predictors $\widetilde{X}_i(u)$ and their responses $Y_i(t)$ that we have just generated.

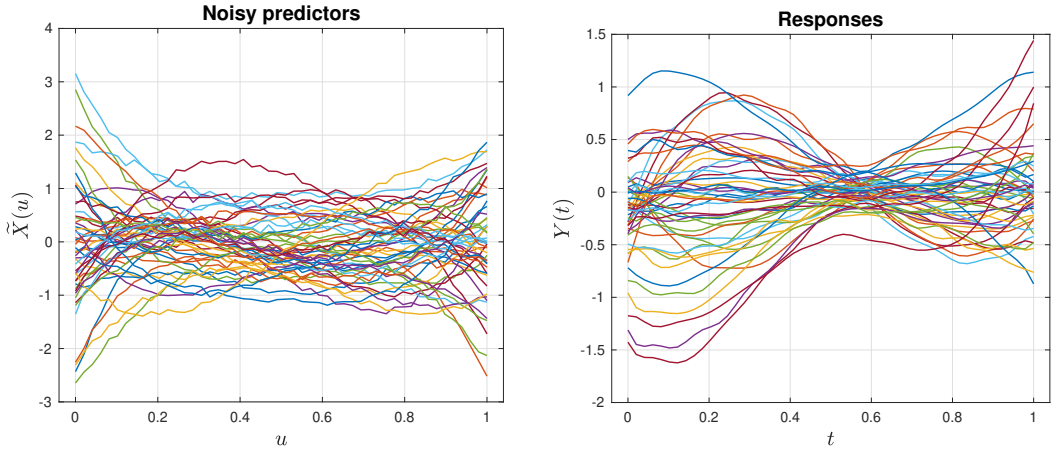


Figure 4.2: 50 randomly taken predictors and their responses.

Results. Figure 4.3 shows the true and the estimated kernels obtained by FF-FME model. We can see, given the generated dataset, the estimated kernels approximate very well the true ones. The results estimated without roughness penalization fit almost exactly the true kernels. MISEs for each of $\widehat{\beta}_1(t,u)$ and $\widehat{\beta}_2(t,u)$ are 0.02 and 0.01, respectively. When estimating with roughness penalization, the resulting kernels clearly less wiggle than the the ones obtained without roughness penalization. Here the simulated true kernels are smooth and the noise level in data is quite small, so the estimated ones in both cases have more chance to be smooth. However, in real-world applications, when the data are more noisy, and the underlying true kernels may not be smooth, the proposed roughness penalization is promising.

The true and estimated functional gating networks are also shown in Figure 4.4 for illustration. The MISE of the predicted functional responses $\widehat{Y}_i(t)$ on testing set is 0.3928. Finally, the ARI between the true labels and the estimated labels using the maximum a posteriori (MAP) rule is 0.7822.

■ Application to Berkeley growth data

In this part, we apply the proposed FF-FME approach to the Berkeley growth data, which is a well-known data originally published in Tuddenham and Snyder (1954) and have broadly been analyzed in the literature since then. The data consists of the height records for 39 boys and 54 girls from age 1 to 18. In original data, the measurements were taken quarterly from ages 1 to 2, annually from 2 to 8, and semiannually from 8 till 18. Therefore, as in many related studies, we performed a simple interpolation such that the trajectories are all available quarterly from age 1 to 18.

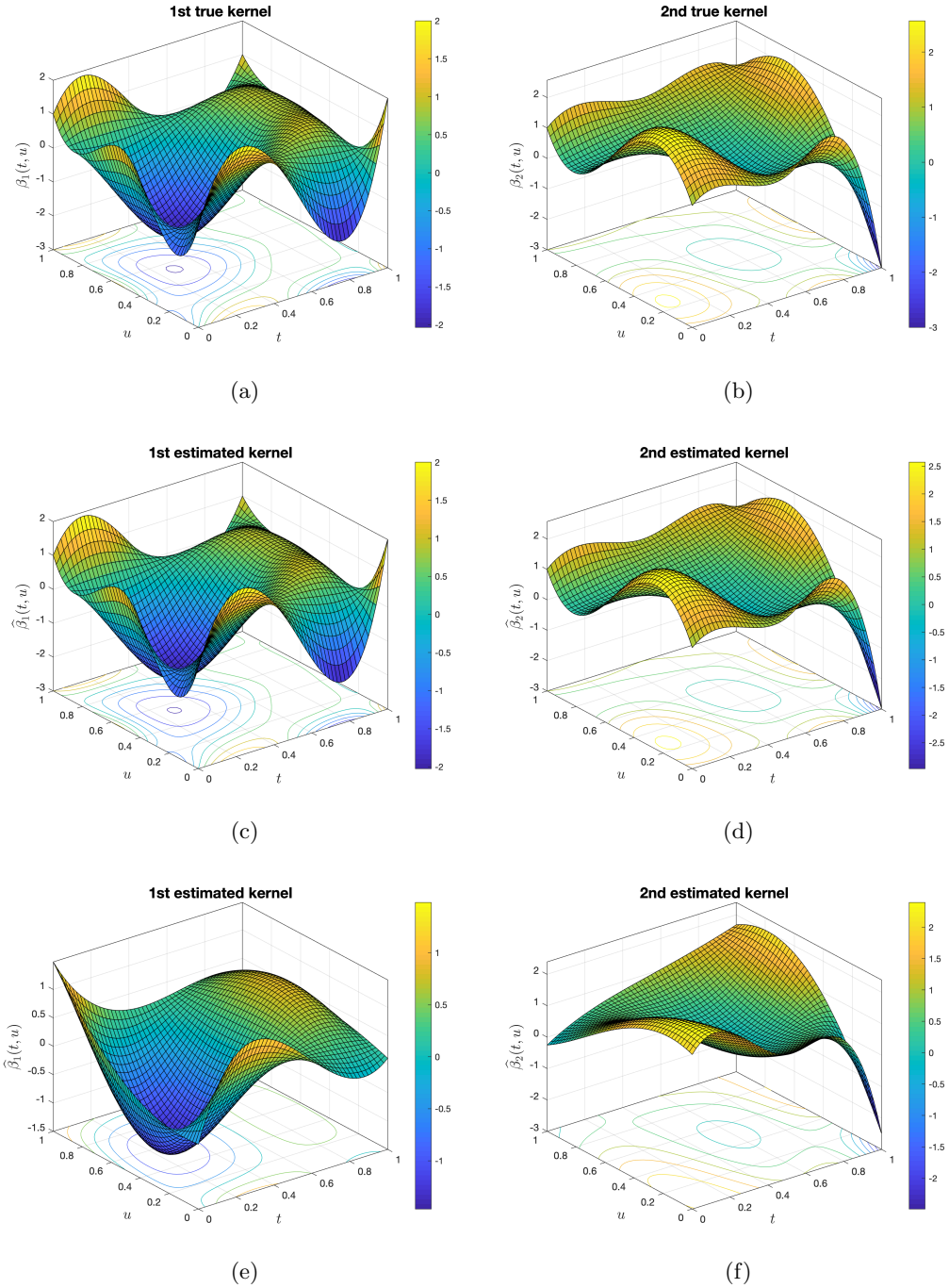


Figure 4.3: Panels (a) and (b) are the true kernels $\beta_1(t,u)$ and $\beta_2(t,u)$. Panels (c) and (d) are the estimated kernels $\widehat{\beta}_1(t,u)$ and $\widehat{\beta}_2(t,u)$, estimated without roughness penalization. Panels (e) and (f) are the estimated kernels $\widehat{\beta}_1(t,u)$ and $\widehat{\beta}_2(t,u)$, estimated with roughness penalty constant $\lambda_t = \lambda_u = 0.001$.

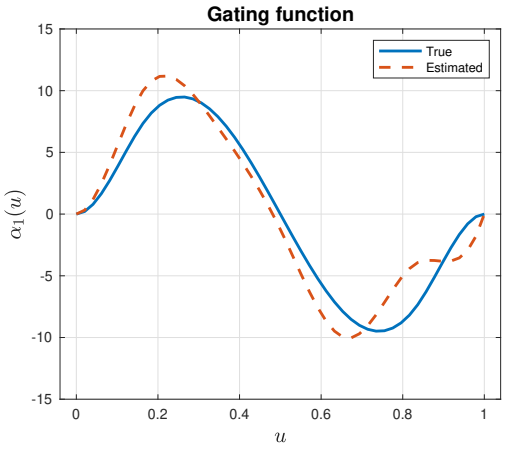
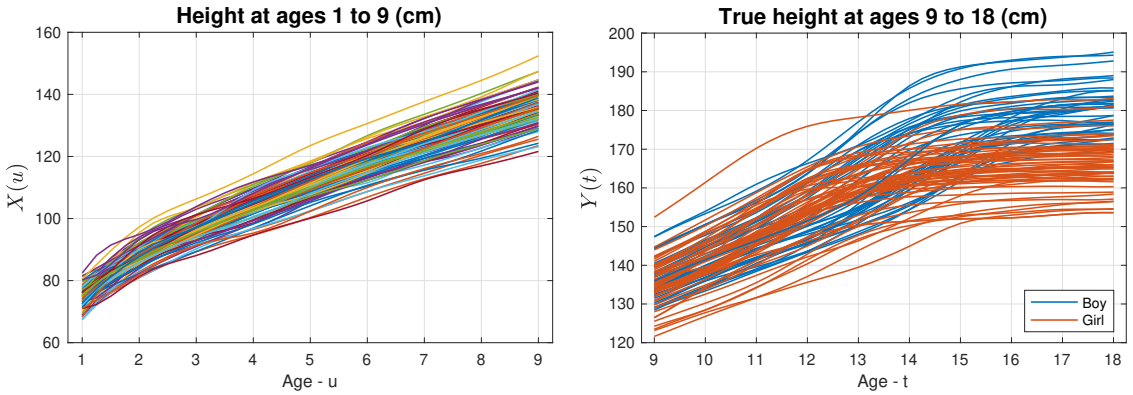


Figure 4.4: The true gating function $\alpha_1(t)$ and its estimated $\widehat{\alpha}_1(t)$.

In Ramsay et al. (1995), the authors investigate the height acceleration to reveal the dynamics of human growth. In their study, they predicted the height at age 18 based on the dynamic pattern till age 9, since it is known that the growth patterns of boys and girls during their pubertal spurts differ significantly in terms of magnitude and timing. The height trajectories of the two groups and their heights at 18 can be found in the appendix (Figure A.2).

In this experiment for FF-FME model, we study from a different perspective by examining the dependence of the heights at age 9 to 18 (functional response) on the heights at age 1 to 9 (functional predictor), *i.e.*, before the pubertal spurts. Figure 4.5 displays our functional predictors $X_i(u)$ and functional responses $Y_i(t)$, where $u \in [1, 9]$ and $t \in [9, 18]$.



(a) (b)

Figure 4.5: The functional predictors and the true functional responses from Berkeley data.

Implementation details and results. Firstly, we split the data in to training and testing sets with 83 and 10 pairs, respectively. Next, we centerize the functional predictors and responses using the mean functions $\mu_X(u)$, $\mu_Y(t)$ computed on the training set. The centered data on training

set are then fed into our algorithm. Here, we used $K = 2$, a tolerance of 10^{-8} and five random initializations for the EM algorithm.

For simplicity, the penalty constant χ for the functional gating function is fixed to a constant. Then, we implement the FF-FME model with four different values for the roughness penalty constant to see the effect of them on the shapes of the estimated kernels. In particular, $\lambda_t = \lambda_u = \gamma$, with $\gamma \in \{0, 10^{-7}, 10^{-5}, 10^{-2}\}$ were used as illustrations. Figure 4.6 shows the corresponding results. As we can see, the larger roughness penalty constant, the smoother kernels were obtained. The shapes of $\widehat{\beta}_1(t, u)$ and $\widehat{\beta}_2(t, u)$ can give us some interpretations about the relationship between the heights *before* and *after* age 9. The kernels estimated without roughness penalization suggest that, for example, for boys, the height at age 18 is highly related to the height at age 6, whereas for girls it is at age 3. Further deeper interpretations could be made about the growth patterns, but that is beyond the scope of this experiment. The estimated gating function, for example when $\lambda_t = \lambda_u = 0$, is shown in Figure 4.7a.

Finally, we evaluate the prediction performance by compute the MISE between the true and the predicted responses on the testing set. Given $X_i(u)$, the predicted response $\widehat{Y}_i(t)$ is computed using the estimated kernels $\widehat{\beta}_1(t, u)$, $\widehat{\beta}_2(t, u)$ and the estimated gating parameter $\widehat{\boldsymbol{\alpha}}$ as

$$\widehat{Y}_i(t) = \mathbb{E}\big[Y_i(t)|X_i(u)\big] = \mu_Y(t) + \sum_{k=1}^{K} \pi_k(X_i(u), \widehat{\boldsymbol{\alpha}}) \int_{\mathcal{U}} X_i(u)\widehat{\beta}_k(t, u)du, \quad t \in \mathcal{T}.$$

Note that, in the notation $\mathbb{E}\left[Y_i(t)|X_i(u)\right]$, the independent variables t and u were written to clarify the defined domains of Y and X, although it is more correctly to write $\mathbb{E}\left[Y_i(t)|X_i(*)\right]$.

Figure 4.7b show the predicted responses obtained by the model estimated with $\lambda_t = \lambda_u = 0$. If we compare Figure 4.7b and Figure 4.5b, we can see that the predicted responses agree with the true responses about the average heights of two groups, *i.e.*, higher for boys and lower for girls. Moreover, the roots of the MISEs for training and testing data are 1.64 and 1.54, respectively. Here, the root of MISE is reported because it is directly proportional to the unit of the height (cm). This means that, on testing set, given the heights at ages 1 to 9, the average error of predictions for heights at ages 9 to 18 is 1.54 cm.

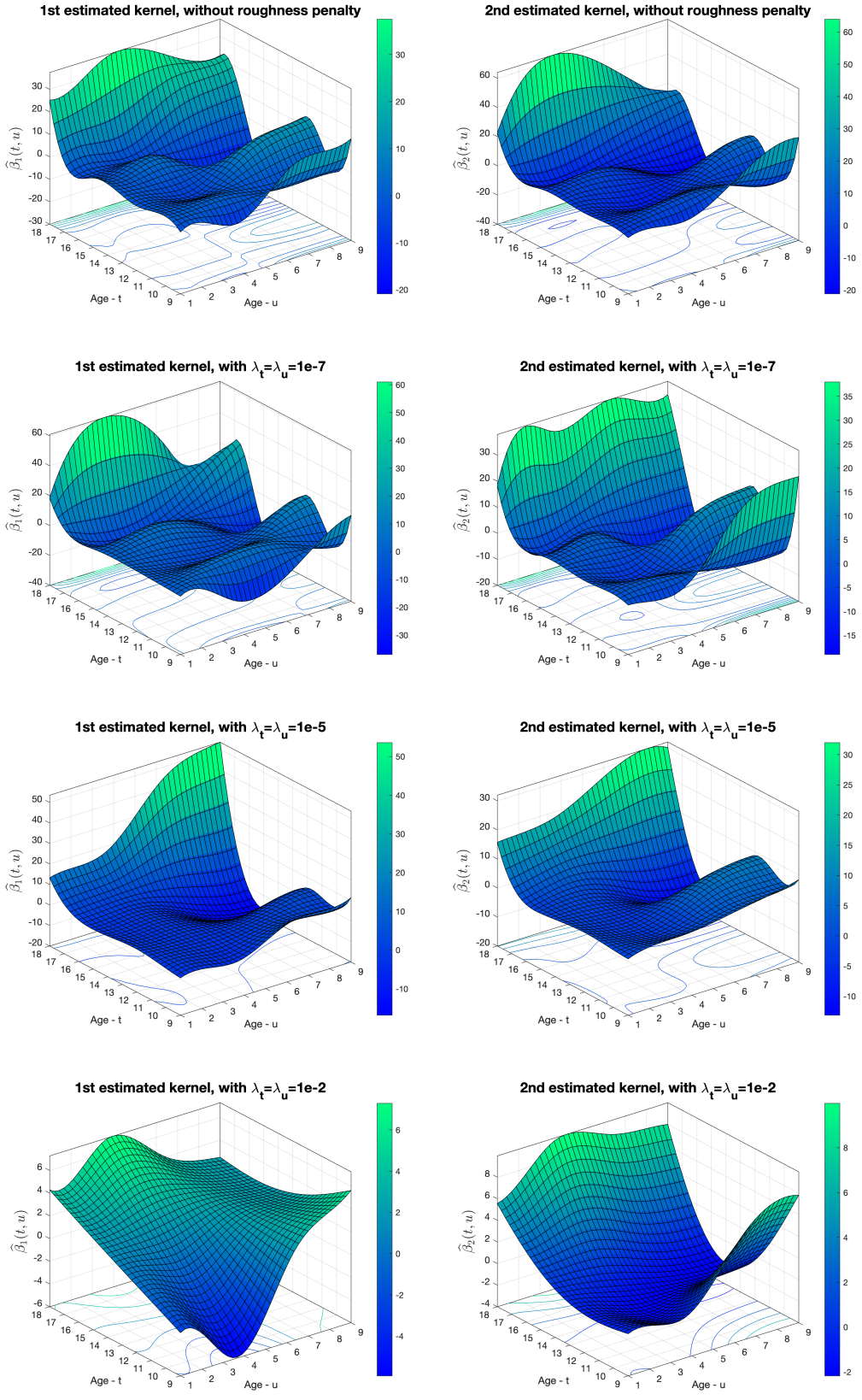
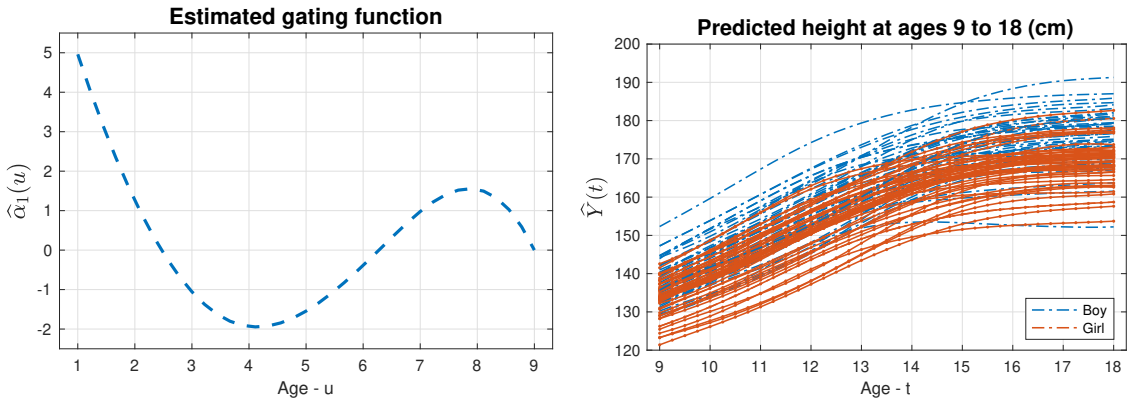


Figure 4.6: Estimated kernels $\widehat{\beta}_1(t,u)$ and $\widehat{\beta}_2(t,u)$ resulted by FF-FME model for Berkeley data.



(a) (b)

Figure 4.7: (a) Estimated functional gating function resulted by FF-FME model and (b) the functional responses predicted by FF-FME model, colored in their predicted cluster.

4.4 FME for function-on-scalar regression

To complete the family of FME models, this sections extends the FME model proposed in **Chapter 3** to the case where the responses are functions and the predictors are scalars. In particular, we consider a function-on-scalar FME (FS-FME) in which the responses are univariate functions, *e.g.*, curves or time series, and we wish to relate them with the explanatory variables via a mixture of experts model. The dedicated algorithm is developed, via maximizing log-likelihood function, to estimate the expert and gating networks. Numerical experiments are also presented to illustrate the usage of our proposed model.

4.4.1 The FS-FME model

Let $\{\mathbf{x}_i, Y_i(t)\}_{i=1}^n$ be a sample of n i.i.d. data pairs where $\mathbf{x}_i \in \mathbb{R}^p$ is the input and $Y_i(t)$ is a functional response with $t \in \mathcal{T} \in \mathbb{R}$. For each $k \in [K]$, we formulate the expert component $\text{Expert}_k(y|x)$ in (2.1) by the function-on-scalar linear model as in (2.20). In particular, the regression model for the ith observation takes the following stochastic representation

$$Y_i(t) = \beta_{z_i,0}(t) + \sum_{j=1}^{p} x_{ip}\beta_{z_i,j}(t) + \varepsilon_i(t), \quad i \in [n], \tag{4.24}$$

where $\varepsilon_i(t)$ is an independent random error function follows a normal distribution with mean zero and covariance function $\Sigma_{z_i}(t,s)$, and $z_i \in [K]$ denotes the unknown label of the expert responsible for the ith observation.

For simplicity, we can simplify the functional intercept $\beta_{z_i,0}$ by assuming that the number one was already incorporated into the predictor vector $\mathbf{x}_i$. Then conditional on $z_i = k$ the above

equations can be rewritten under matrix form simply as

$$Y_i(t) = \mathbf{x}_i^\top \boldsymbol{\beta}_k(t) + \varepsilon_i(t), \quad i \in [n], \tag{4.25}$$

where $\boldsymbol{\beta}_k(t) = [\beta_{k1}, \ldots, \beta_{kp}]^\top$ is the vector of functional parameters to be estimated.

Modeling the gating network

The gating function in this case can take the same form as one of those in the vectorial setting, *e.g.*, softmax gating function as in (2.3) or Gaussian gating function as in (2.4). Here, we opt to model the gating network for FS-FME model using softmax modeling

$$\pi_k(\mathbf{x}_i; \boldsymbol{\alpha}) = \frac{\exp(\mathbf{x}_i^\top \alpha_k)}{1 + \sum_{k'=1}^{K-1} \exp(\mathbf{x}_i^\top \alpha_{k'})}, \tag{4.26}$$

where $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{K-1})^\top \in \mathbb{R}^{p(K-1)}$ is the gating parameter vector to be estimated.

Modeling the expert network

Analogously to the modeling of expert network in the FF-FME model, we assume that the responses $Y_i(t)$, the parameters $\beta_{kj}(t)$, and the error functions $\varepsilon_i(t)$ can be represented by a collection of basis functions $\boldsymbol{\psi}(t) = [\psi_1(t), \ldots, \psi_q(t)]^\top$ as

$$Y_i(t) = \boldsymbol{y}_i^\top \boldsymbol{\psi}(t)^\top, \quad \beta_{kj}(t) = \boldsymbol{b}_{kj}^\top \boldsymbol{\psi}(t), \quad \text{and } \varepsilon_i(t) = \boldsymbol{\epsilon}_i^\top \boldsymbol{\psi}(t),$$

where $\boldsymbol{y}_i$, $\boldsymbol{b}_{kj}$, $\boldsymbol{\epsilon}_i \in \mathbb{R}^q$ are the corresponding coefficient vectors, and $\boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{z_i})$. Then, the equations in (4.25) implies

$$\boldsymbol{y}_i = \mathbf{x}_i^\top \boldsymbol{B}_k + \boldsymbol{\epsilon}_i^\top, \quad i \in [n], \tag{4.27}$$

where $\boldsymbol{B}_k = [\boldsymbol{b}_{k1}, \ldots, \boldsymbol{b}_{kp}]^\top \in \mathbb{R}^{p \times q}$ is now the unknown coefficient matrix associated with the expert kth to be estimated.

From (4.27), the model for a coefficient vector $\boldsymbol{y}_i$ of a functional response $Y_i(t)$ given $\mathbf{x}_i$ can be expressed as a probability density function as follows

$$f(\boldsymbol{y}_i | \mathbf{x}_i, z_i = k; \boldsymbol{\theta}_k) = \phi(\boldsymbol{y}_i; \mathbf{x}_i^\top \boldsymbol{B}_k, \boldsymbol{\Sigma}_k), \tag{4.28}$$

where $\phi(\cdot; \boldsymbol{\mu}, \boldsymbol{\nu})$ is the pdf of the q-dimensional Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\nu}$, and $\boldsymbol{\theta}_k = (\text{vec}(\boldsymbol{B}_k^\top)^\top, \text{vech}^\top \boldsymbol{\Sigma}_k)^\top$ is the unknown parameter vector of the expert k, $k \in [K]$, to be estimated.

Conditional density function of the FS-FME model

Gathering the gating function (4.26) and the expert density (4.28) we obtain the following conditional density function for the FS-FME model:

$$f(\boldsymbol{y}_i|\mathbf{x}_i;\boldsymbol{\Psi}) = \sum_{k=1}^{K} \pi_k(\mathbf{x}_i;\boldsymbol{\alpha})\phi(\boldsymbol{y}_i;\mathbf{x}_i^\top \boldsymbol{B}_k, \boldsymbol{\Sigma}_k), \tag{4.29}$$

where $\boldsymbol{\Psi} = (\boldsymbol{\alpha}^\top, \boldsymbol{\theta}_1^\top, \ldots, \boldsymbol{\theta}_K^\top)^\top$ is the parameter vector of the model.

4.4.2 Penalized maximum likelihood estimation via EM algorithm

The maximum likelihood estimation for FS-FME model can be obtained via the EM algorithm in the same manner as in the FF-FME model considered in previous section. In addition, to obtain smooth fits for the functional parameters $\beta_{kj}(t)$ we impose a roughness penalty term to the log-likelihood function and consider the following penalized log-likelihood function

$$\mathcal{L}(\boldsymbol{\Psi}) = \log L(\boldsymbol{\Psi}) - \text{Pen}_\lambda(\boldsymbol{\Psi}), \tag{4.30}$$

where $\log L(\boldsymbol{\Psi})$ is the observed-data log-likelihood of $\boldsymbol{\Psi}$ and $\text{Pen}_\lambda(\boldsymbol{\Psi})$ is a roughness penalty term that encouraging smoothness for the functional experts defined by

$$\text{Pen}_\lambda(\boldsymbol{\Psi}) = \sum_{k=1}^{K}\sum_{j=1}^{p} \lambda_{kj} \int_{\mathcal{T}} \left[(\mathcal{D}\beta_{kj})(t)\right]^2 dt, \tag{4.31}$$

where $\mathcal{D}$ is some differential operator, for example, the second derivative. Here, similarly to the FF-FME model, the dependence of the penalty term of $\boldsymbol{\Psi}$ is implicitly. The two steps of the EM algorithm for FS-FME model are performed as follows.

E-step. We calculate the conditional probability memberships $\tau_{ik}^{(s)}$ given by

$$\tau_{ik}^{(s)} = \mathbb{P}(z_i = k|\mathbf{x}_i, \boldsymbol{y}_i; \widehat{\boldsymbol{\Psi}}^{(s)}) = \frac{\pi_k(\mathbf{x}_i;\widehat{\boldsymbol{\alpha}}^{(s)})\phi(\boldsymbol{y}_i;\mathbf{x}_i^\top \widehat{\boldsymbol{B}}_k^{(s)}, \widehat{\boldsymbol{\Sigma}}_k^{(s)})}{f(\boldsymbol{y}_i|\mathbf{x}_i;\widehat{\boldsymbol{\Psi}}^{(s)})}, \tag{4.32}$$

where $\widehat{\boldsymbol{\Psi}}^{(s)} = (\widehat{\boldsymbol{\xi}}^{(s)\top}, \widehat{\boldsymbol{\theta}}_1^{(s)\top}, \ldots, \widehat{\boldsymbol{\theta}}_K^{(s)\top})^\top$ is the estimate of $\boldsymbol{\Psi}$ at the iteration sth of the EM algorithm.

M-step. In the M-step, the gating parameter and the expert parameters will be estimated separately as we have seen many times before. Firstly, for the gating network, similarly to the FF-FME model, the maximization with respect to $\boldsymbol{\alpha}$ can be done as in (3.16) via the MLE, or in (3.20) via the regularized MLE. Therefore, we again refer to the *updating gating network parameters* part given in Appendix B.1 for the maximization with respect to $\boldsymbol{\alpha}$.

For each functional expert k, because of the stochastic representation (4.24), we can recognize that the maximization with respect to $\boldsymbol{\theta}_k$ consists of solving a weighted function-on-scalar regression

problem, where the weights are the current conditional expert memberships $\tau_{ik}^{(s)}$ given in (4.32). Therefore, we can take advantage from the solution formula for the function-on-scalar regression model presented in Subsection 2.6.2. In particular, let us denote $\boldsymbol{Y} = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n]^\top$, $\boldsymbol{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]^\top$, $\boldsymbol{\tau}_k^{(s)} = (\tau_{k1}^{(s)}, \ldots, \tau_{kn}^{(s)})^\top$, the update formula for the parameter vector of expert kth for the next EM iteration is given by

$$\operatorname{vec}(\widehat{\boldsymbol{B}}_k^\top)^{(s+1)} = \left(\boldsymbol{U}_\tau^\top \boldsymbol{U}_\tau + \boldsymbol{\Lambda}_k \otimes \boldsymbol{D}\right)^{-1} \boldsymbol{U}_\tau^\top \operatorname{vec}(\boldsymbol{I}_\omega^{1/2} \boldsymbol{Y}_\tau^\top), \quad \boldsymbol{U}_\tau = \boldsymbol{X}_\tau \otimes \boldsymbol{I}_\omega^{1/2}, \tag{4.33}$$

$$\widehat{\boldsymbol{\Sigma}}_k^{(s+1)} = \frac{1}{\sum_{i=1}^n \tau_{ik}^{(s)}} \left(\boldsymbol{Y}_\tau - \boldsymbol{X}_\tau \widehat{\boldsymbol{B}}_k^{(s+1)}\right)^\top \left(\boldsymbol{Y}_\tau - \boldsymbol{X}_\tau \widehat{\boldsymbol{B}}_k^{(s+1)}\right), \tag{4.34}$$

where

$$\begin{aligned}
\boldsymbol{\Lambda}_k &= \operatorname{diag}(\lambda_{k1}, \ldots, \lambda_{kp}) \in \mathbb{R}^{p \times p}, \\
\boldsymbol{D} &= \left[\int_{\mathcal{T}} (\mathcal{D}\omega_g)(t)(\mathcal{D}\omega_h)(t)dt\right]_{g,h} \in \mathbb{R}^{q \times q}, \\
\boldsymbol{I}_\omega^{1/2}\boldsymbol{I}_\omega^{1/2} &= \left[\int_{\mathcal{T}} \omega_g(t)\omega_h(t)dt\right]_{g,h} \in \mathbb{R}^{q \times q}, \\
\boldsymbol{X}_\tau &= \sqrt{\boldsymbol{\tau}_k^{(s)}} \odot \boldsymbol{X} \in \mathbb{R}^{n \times p}, \\
\boldsymbol{Y}_\tau &= \sqrt{\boldsymbol{\tau}_k^{(s)}} \odot \boldsymbol{Y} \in \mathbb{R}^{n \times q}
\end{aligned}$$

where the square root is understood componentwise, $\otimes$ is the Kronecker product operator and $\odot$ is the Hadamard product operator. Similarly to the update formulas in FF-FME model, the dependence of the right hand side of (4.33) on the sth iteration of the EM algorithm is implicit via $\boldsymbol{\tau}_k^{(s)}$.

4.4.3 Numerical experiments

Data generating process

In this part, we illustrate the usage of the proposed FS-FME model on simulated data, to relate functional responses to scalar predictors. The data generating process is follows. First, we fix a *true* functional expert network that consists of the functions $\beta_{kj}(t)$ as can be seen in Figure 4.8. Here $K = 3$, $p = 3$, and we intentionally created rough curves so that we can see the effect of the smooth penalization on the estimation.

Next, we generated 200 predictors $\mathbf{x}_i$ with whose components follow $\mathcal{N}(0, 1)$, $\mathcal{N}(1, 1)$, $\mathcal{N}(2, 1)$ with equal proportions. The functional responses $Y_i(t)$ are then generated by

$$\begin{aligned}
Y_i(t)|Z_i = z_i, \mathbf{x}_i &= \mathbf{x}_i^\top \boldsymbol{\beta}_{z_i}(t) + \varepsilon_i(t), \\
Z_i|\mathbf{x}_i &\sim \mathcal{M}\Big(1, \big(\pi_1(\mathbf{x}_i; \boldsymbol{\alpha}), \ldots, \pi_K(\mathbf{x}_i; \boldsymbol{\alpha})\big)\Big).
\end{aligned}$$

We can see, the wiggles in Y_i come from both the expert functions and the error functions. Figure 4.8

shows the generated $Y_i(t)$ colored according to their true labels.

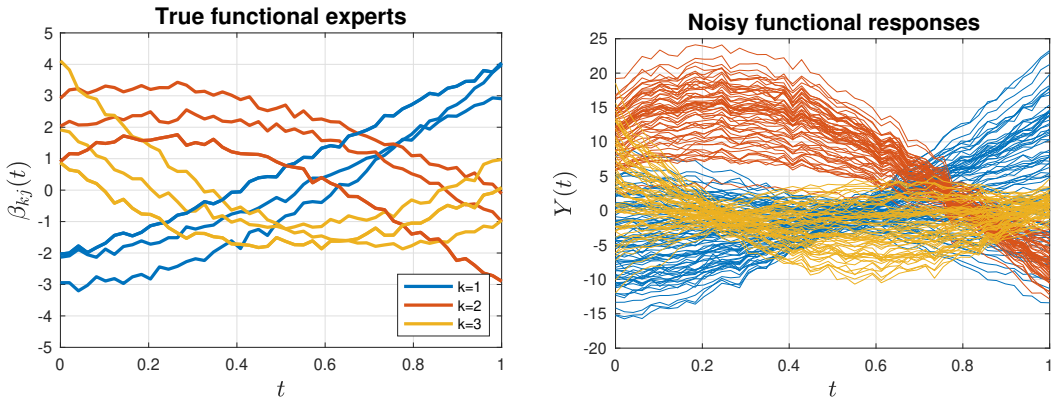


Figure 4.8: The true functional experts and the noisy functional responses in the simulated data.

Implementation and results

We split the dataset into training and testing sets with equal proportions, estimate the functional experts using the training set, and report the evaluation metrics on the testing set. Specially, we implement the FS-FME model with different roughness penalty constants to illustrate the effect of the smooth penalization on the estimation. In this study, we set the operator $\mathcal{D}$ to the second derivative.

To evaluate the estimation of the functional experts, we compute the following quantity

$$\text{MISE}(\widehat{\boldsymbol{\beta}}^{\lambda}) = \frac{1}{Kp}\sum_{k=1}^{K}\sum_{j=1}^{p}\int_{\mathcal{T}}(\widehat{\beta}_{kj}^{\lambda}(t) - \beta_{kj}(t))^2 dt,$$

where $\beta_{kj}(t)$ is the true parameter function, and $\widehat{\beta}_{kj}^{\lambda}(t)$ is the parameter function estimated with penalty constant λ. To evaluate the prediction, we compute the MISE between the true and the predicted responses in the testing set.

Figure 4.9 shows the estimated expert functions $\widehat{\beta}_{kj}(t)$ with three different penalty constants: $\lambda_1 = 0$, $\lambda_2 = 0.001$, $\lambda_3 = 1$, and the corresponding predicted responses $\widehat{Y}(t)$ for the predictors in the testing set. As we can see, the larger λ results in the smoother for the estimated functions. With $\lambda = 1$, the estimated experts tends to straight lines. The corresponding values for $\text{MISE}(\widehat{\boldsymbol{\beta}}^{\lambda})$ are 0.0151, 0.0107, and 0.1607. The prediction on testing set can be seen visually in Figure 4.9. The MISEs in prediction on the testing set are 24.6612, 24.4827, and 25.4218, respectively for the three estimated models.

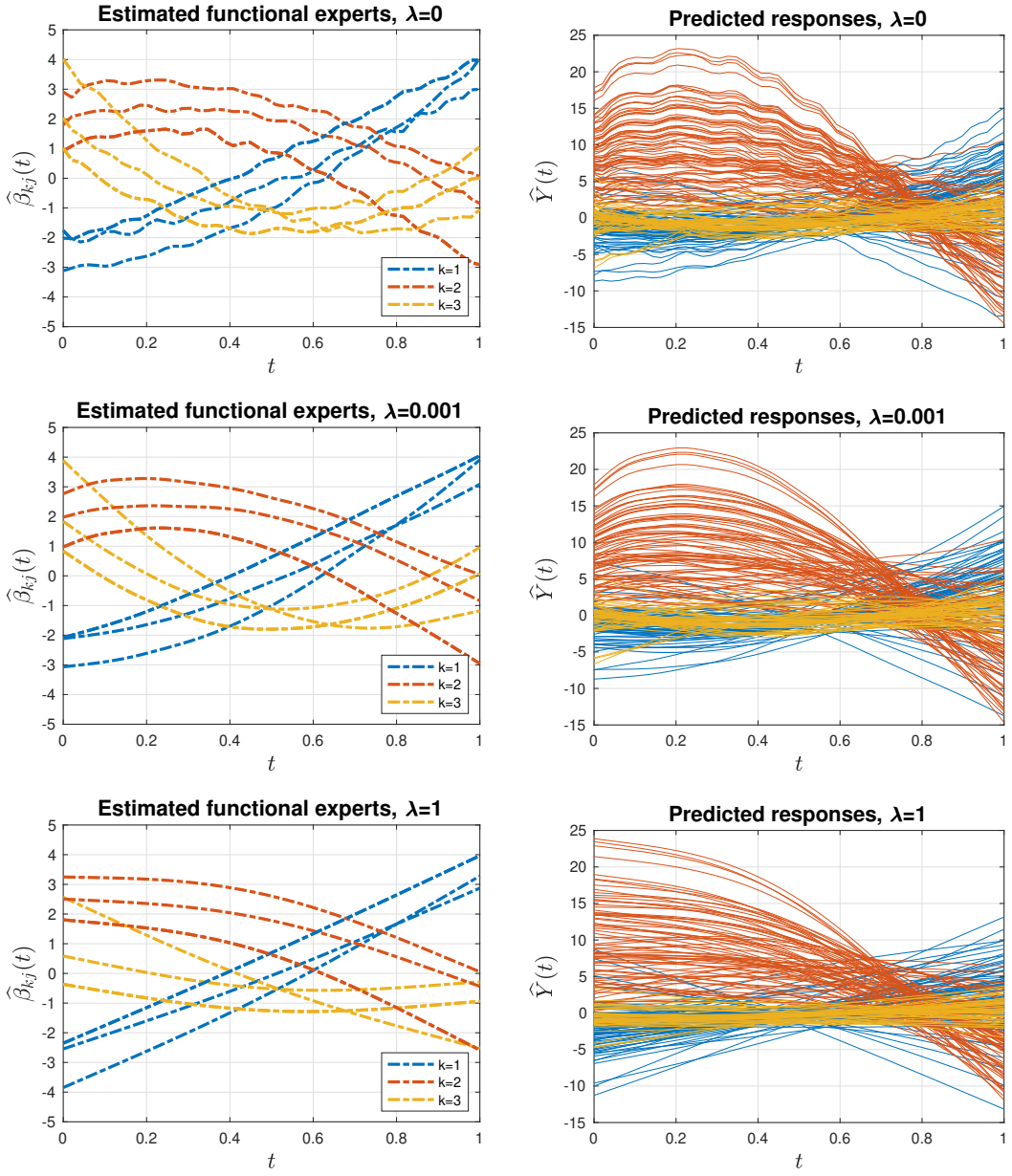


Figure 4.9: The estimated functional experts resulted by different roughness penalty constants, and the predicted responses for the predictors in the testing set.

4.5 Summary

In this chapter, we extended the FME, FME-Lasso and iFME models proposed in **Chapter 3** to the case of categorical responses, *i.e.*, for multiclass classification problems. The corresponding EM algorithm were also developed for the extended models. Simulation study was constructed to illustrate the performance of the models. We also applied the models to the phoneme data and obtained promising results in terms of both classification correct rate and accuracy of parameter fitting.

The first time ME models are developed to deal with regression problems with *functional responses*. In particular, we proposed FS-ME and FF-ME models to model the relationship between functional responses and vetorial predictors (FS-ME model)/ functional predictors (FF-ME model) in heterogeneous populations. The dedicated algorithms were presented for parameter estimations. These models have many potential applications as illustrated in the experimental studies subsections, especially the effect of roughness penalization on producing smooth fits for the underlying true kernels.

There are many other extensions can be considered. For example, we can develop interpretable variants for the FF-FME and FS-FME models. In particular, the functional gating network in the FF-FME model can be defined upon the finite representation of the gating function's derivatives. Like that, the estimated kernels can be truly sparse and get interpretable shapes as seen in the iFME models.

Chapter 5

Distributed learning for mixtures of experts

Contents

5.1 Introduction

Many modern applications nowadays, including regression, classification as well as clustering, must deal with datasets that cannot be stored on a single machine. This may be due to the nature of the data, communication issues such as with data in meteorology where they are collected at different

stations, or by privacy restriction such as with data in finance and medicine. Even if a single full dataset is available, it often requires a huge processing resource for handling and running statistical inference methods, this may be evenly more expensive.

Such restrictions may prevent one from obtaining global estimators for the models assumed on the data. Parallelizing or sequentializing the existing learning algorithms and statistical methods is therefore an active research area. Most of works are based on a two-step divide and conquer procedure: • Local inference (or local computation) and • Aggregation. In the first step, standard inference or computation is performed on subsets of data available at local machines, then in the second step, the local results are collected to a central machine where they are aggregated.

Although relying on the same divide and conquer principle, these attempts can be categorized into two groups: one parallelizes the model computation and one parallelizes the model inference. Parallelizing the model computation means the estimator for the model parameter is maintained and updated on a central machine, while the computations for that update procedure are distributed and carried out on local machines. However, not all the computations of the existing algorithms can be parallelized effectively, by the nature of the formulas, or by the high cost of communication across the machines during the computations. Therefore, parallelizing the model inference is a preferred approach. It performs standard inference on local machines to obtain local estimators, then transmits them to a central machine where they are aggregated together, based on some efficient strategies, to produce an overall estimator for the model parameter.

There are many successful attempts in this direction of parallelizing the existing learning algorithms and statistical methods. Those may be mentioned here include parallelizing stochastic gradient descent(Zinkevich et al., 2010), parallelizing multiple linear regression(Mingxian et al., 1991), parallel K-Means clustering based on MapReduce (Zhao et al., 2009), distributed learning for heterogeneous data via model integration (Merugu and Ghosh, 2005), split-and-conquer approach for penalized regressions (Chen and ge Xie, 2014), for logistic regression (Shofiyah and Sofro, 2018), distributed learning for finite Gaussian mixtures (Zhang and Chen, 2021), among others.

To make this chapter self-complete, let us briefly recall the notion of the ME framework. The main idea of ME models themselves is also the divide-and-conquer principle, in which a complex problem is divided into smaller simple sub-problems, then each one can be solved by a special expert. A ME model can be written in general notation as

$$f(y|x) = \sum_{k=1}^{K} \text{Gate}_k(x)\, \text{Expert}_k(y|x),$$

in which $f(y|x)$, the distribution of response y given the covariate x, is modeled as a mixture distribution with covariate-dependent mixing proportions $\text{Gate}_k(x)$ and conditional mixture components $\text{Expert}_k(y|x)$. In ME model terminology, $\text{Gate}_k(x)$ is referred to as gating function, and $\text{Expert}_k(y|x)$ is referred to as expert function, while $K \in \mathbb{N}$ is the number of experts. ME is therefore a fully conditional mixture model that allows the mixing proportions to be functions of the covariates.

The most popular models for the function $\text{Gate}_k(x)$ are the softmax gating function and the

Gaussian gating function, while for the conditional mixture components $\text{Expert}_k(y|x)$, there are many choices depend on the considered problem. When the response y is continuous and assumed to be Gaussian given the input x, $\text{Expert}_k(y|x)$ can be modeled by the conditional density function of Gaussian distribution. When the response y is binary, resp. categorical, the suitable choice for $\text{Expert}_k(y|x)$ is the conditional density function of the binary, resp. multinomial, logistic regression distribution. In addition to these popular ones, many other models for the conditional mixture components have been studied (Chamroukhi, 2015). In situations where standard mixture of regression models fails to approximate the conditional probability distribution of data, ME model may be a good replacement as it can capture a wide variety of data distributions thanks to their universal approximation property well established for almost of popular gating and experts functions (Nguyen et al., 2019, 2021a).

Some successful applications of ME architecture may be mentioned here include ME for time series prediction (Zeevi et al., 1996; Yümlü et al., 2003), segmentation (Chamroukhi et al., 2013, 2009), ME for social network data (Gormley and Murphy, 2010), for classification of gender and pose of human faces (Gutta et al., 2000), *etc.* For an overview of practical and theoretical aspects of ME modeling, reader is referred to Nguyen and Chamroukhi (2018); Yuksel et al. (2012).

In this chapter, we particularly introduce a distributed learning approach for the softmax-gated mixtures of Gaussian experts model because of its popularity in applications. Let $(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$ be a random observation from some probability model, where $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^d$ ($d \in \mathbb{N}$) is the input and $y \in \mathcal{Y} \subset \mathbb{R}$ is the output. A softmax-gated mixture of Gaussian experts model links y and $\mathbf{x}$ via the following conditional density function

$$f(y|\mathbf{x};\boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k(\mathbf{x};\boldsymbol{\alpha})\varphi(y;\mathbf{x}^\top\boldsymbol{\beta}_k,\sigma_k^2), \tag{5.1}$$

where $\boldsymbol{\theta} = (\boldsymbol{\alpha}^\top, \boldsymbol{\beta}_1^\top, \ldots, \boldsymbol{\beta}_K^\top, \sigma_1^2, \ldots, \sigma_K^2)^\top$ is the parameter vector of the model. Here, $\pi_k(\mathbf{x};\boldsymbol{\alpha})$ is the softmax gating function given by

$$\pi_k(\mathbf{x};\boldsymbol{\alpha}) = \frac{\exp(\mathbf{x}^\top\boldsymbol{\alpha}_k)}{1+\sum_{k'=1}^{K-1}\exp(\mathbf{x}^\top\boldsymbol{\alpha}_{k'})},$$

where $\boldsymbol{\alpha} = (\boldsymbol{\alpha}_1^\top, \ldots, \boldsymbol{\alpha}_{K-1}^\top)^\top$, $\boldsymbol{\alpha}_k \in \mathbb{R}^d$ for $k \in [K-1]$, is the parameter vector of the gating network; $\varphi(\cdot;\mu,\sigma^2)$ is the conditional Gaussian density function with mean μ and variance σ^2, and $(\boldsymbol{\beta}_k^\top, \sigma_k^2)^\top \in \mathbb{R}^{d+1}$ is the parameter vector of the expert k for $k \in [K]$.

Such setting has been exclusively studied in multivariate analysis. However, in practice when the dataset (based on it we estimate the model parameter) is large, the computation for parameter estimation may be costly or evenly infeasible. Therefore, a distributed learning approach for this ME model is particularly interesting and will be developed in this chapter.

The rest of this chapter is organized as follows. In Section 5.2 we formulate the problem of aggregating the local estimates of a ME model and we will propose a numerical strategy for solving it in Section 5.3. Section 5.5 is dedicated for numerical study and Section 5.6 is for summary and

discussion.

5.2 Aggregating distributed ME models

5.2.1 Problem setting

Let $\mathscr{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$ be a random sample of N (with $N \in \mathbb{N}$ supposed to be large) independently identically distributed (i.i.d.) data pairs from a ME model $f(\cdot|\mathbf{x}, \boldsymbol{\theta}^*)$ as in (5.1), where $(\mathbf{x}_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, $\mathcal{X} \subset \mathbb{R}^d$, $\mathcal{Y} \subset \mathbb{R}^q$, and $\boldsymbol{\theta}^*$ is the true parameter vector. For simpler notation, we will denote by f^* the true conditional density function $f(\cdot|\mathbf{x}, \boldsymbol{\theta}^*)$. Note that $\mathbf{x}_i$, y_i denote observed data, whereas $\mathbf{x}$, y denote the generic variables.

With the assumption that N is so large that it is difficult to estimate $\boldsymbol{\theta}^*$ on a single computer within a reasonable amount of time, a natural approach is to use the divide-and-conquer principle to estimate $\boldsymbol{\theta}^*$. Suppose that $\mathscr{D}$ is randomly divided into M disjoint subsets $\mathscr{D}_1, \ldots, \mathscr{D}_M$ and stored on M local machines. Let $f(\cdot|\mathbf{x}, \widehat{\boldsymbol{\theta}}_m)$, abbreviated simply by $\widehat{f}_m$, be the conditional density function (with parameter vector $\widehat{\boldsymbol{\theta}}_m$) of the ME model estimated based on sample $\mathscr{D}_m$, namely,

$$\widehat{f}_m := f(\cdot|\mathbf{x}, \widehat{\boldsymbol{\theta}}_m) = \sum_{k=1}^K \pi_k(\mathbf{x}; \widehat{\boldsymbol{\alpha}}^{(m)})\, \varphi(\cdot|\mathbf{x}; \widehat{\boldsymbol{\beta}}_k^{(m)}), \tag{5.2}$$

where $\widehat{\boldsymbol{\theta}}_m = (\widehat{\boldsymbol{\alpha}}^{(m)}, \widehat{\boldsymbol{\beta}}_1^{(m)}, \ldots, \widehat{\boldsymbol{\beta}}_K^{(m)})$ obtained by maximizing the observed-data log-likelihood function (2.9) with the data in $\mathscr{D}_m$.

The question is how to approximate the true global density f^* from the local densities $\widehat{f}_m$, and more importantly, how to aggregate the local estimators $\widehat{\boldsymbol{\theta}}_m$ to produce a single aggregated estimator for the true parameter vector $\boldsymbol{\theta}^*$. Here and after, we use "local density" and "local estimator" to refer to the conditional density function $\widehat{f}_m$ and the parameter vector $\widehat{\boldsymbol{\theta}}_m$, respectively, of the ME model estimated at local machine m based on $\mathscr{D}_m$, $m \in [M]$. Besides, for the reader's convenience, from now on the local models and local parameters will be denoted with hat notations.

5.2.2 Aggregation strategies

Let N_m be the sample size of the subset $\mathscr{D}_m$, *i.e.*, $\sum_{m=1}^M N_m = N$. A natural strategy to approximate the true density f^* is to use the weighted average density $\bar{f}^W$ defined by

$$\bar{f}^W = \sum_{m=1}^M \lambda_m \widehat{f}_m, \tag{5.3}$$

where λ_m denote the sample proportions $\frac{N_m}{N}$ those sum to one, and $\widehat{f}_m$ are the local densities defined in (5.2). This average density $\bar{f}^W$ fits our intuition about an average model, and in fact, approximates very well the true density f^*. However, there are two issues with this aggregating strategy. First, it only gives us an expression to approximate the conditional probability values, *i.e.*, for each $\mathbf{x} \in \mathcal{X}$ we have $\bar{f}^W(y|\mathbf{x}) \approx f^*(y|\mathbf{x})$, it does not give us the approximation expression for the true parameter $\boldsymbol{\theta}^*$. In other words, it is not clear how to obtain a "weighted average" approximation for $\boldsymbol{\theta}^*$ from the local estimators $\widehat{\boldsymbol{\theta}}_m$'s. Second, since each $\widehat{f}_m$ is a mixture of K components, we can express the density $\bar{f}^W$ in (5.3) as

$$\begin{aligned}\bar{f}^W &= \sum_{m=1}^{M} \lambda_m \sum_{k=1}^{K} \pi_k(\mathbf{x}; \widehat{\boldsymbol{\alpha}}^{(m)})\, \varphi(\cdot|\mathbf{x}; \widehat{\boldsymbol{\beta}}_k^{(m)}) \\ &= \sum_{m=1}^{M} \sum_{k=1}^{K} \lambda_m \pi_k(\mathbf{x}; \widehat{\boldsymbol{\alpha}}^{(m)})\, \varphi(\cdot|\mathbf{x}; \widehat{\boldsymbol{\beta}}_k^{(m)}). \end{aligned} \tag{5.4}$$

One can see that $\sum_{m=1}^{M}\sum_{k=1}^{K} \lambda_m \pi_k(\mathbf{x}; \widehat{\boldsymbol{\alpha}}^{(m)}) = 1$ for all $\mathbf{x} \in \mathcal{X}$, so $\bar{f}^W$ can be viewed as a mixture model with MK components in which the component densities are $\varphi(\cdot|\mathbf{x}; \widehat{\boldsymbol{\beta}}_k^{(m)})$ and the gating functions are $\lambda_m \pi_k(\mathbf{x}; \widehat{\boldsymbol{\alpha}}^{(m)})$, for $k \in [K]$, $m \in [M]$. Therefore, although this MK-component mixture approximates well the true density f^*, it has a wrong number of components, *i.e.*, MK instead of K, this will make the clustering results useless or difficult to interpret. Hence, this aggregating approach is not what we are aiming for.

Let $\mathcal{M}_K$ denotes the space of all K-component ME models as in (5.1). Two other common strategies to aggregate the local densities $\widehat{f}_1, \ldots, \widehat{f}_M$ are via

$$\bar{f}^B := f(y|\mathbf{x}; \bar{\boldsymbol{\theta}}^B) = \operatorname*{arg\,inf}_{g \in \mathcal{M}_K} \sum_{m=1}^{M} \lambda_m \rho(\widehat{f}_m, g), \tag{5.5}$$

and

$$\bar{f}^R := f(y|\mathbf{x}; \bar{\boldsymbol{\theta}}^R) = \operatorname*{arg\,inf}_{g \in \mathcal{M}_K} \rho(\bar{f}^W, g), \tag{5.6}$$

where $\rho(\cdot, \cdot)$ is some divergence defined on the space of finite mixture distributions. The solution $\bar{f}^B$ and $\bar{f}^R$ are often known as barycenter solution and reduction solution, respectively, with their interpretations are given as follows.

In the case of $\bar{f}^B$, we are finding a K-component ME model g that can be viewed as a barycenter the local models $\widehat{f}_1, \ldots, \widehat{f}_M$ with respect to the weights $\lambda_1, \ldots, \lambda_M$ and the divergence $\rho(\cdot, \cdot)$. Whereas, in the case of $\bar{f}^R$, we are finding a K-component ME model g that is closest with the weighted average density $\bar{f}^W$ defined in (5.3), with respect to the divergence $\rho(\cdot, \cdot)$. Because we are searching for solutions in $\mathcal{M}_K$, both $\bar{f}^B$ and $\bar{f}^R$ solve the problem of wrong number of components in $\bar{f}^W$.

Both solutions are desirable, and in fact, can be shown to be connected under specific choices of $\rho(\cdot, \cdot)$. Indeed, for any bivariate function $\rho(\cdot, \cdot)$ that is linear in the first argument, $\bar{f}^B$ and $\bar{f}^R$ are identical. However, in this chapter we prefer the reduction solution $\bar{f}^R$ for many reasons. First,

as we have already remarked, the MK-component mixture $\bar{f}^W$ is a good approximation to the true density f^*, so it makes more sense to find a K-component ME model that approximates $\bar{f}^W$, which we already know is good. Second, the barycenter approach may lead to a counter-intuitive solution under some specific $\rho(\cdot,\cdot)$, *e.g.*, as already shown in Zhang and Chen (2021) for the case of univariate Gaussian mixture models and the 2-Wasserstein divergence with Euclidean ground distance. Finally, as we can see, given the same divergence $\rho(\cdot,\cdot)$, the computation in (5.5) will be more expensive than that of (5.6), especially in our large data context.

Thus, our objective in the next sections is to develop an efficient algorithm to find the reduction estimator $\bar{\boldsymbol{\theta}}^R$. We shall denote the components of the reduction estimator by $\bar{\boldsymbol{\theta}}^R = (\bar{\boldsymbol{\alpha}}^R, \bar{\boldsymbol{\beta}}_1^R, \ldots, \bar{\boldsymbol{\beta}}_K^R)$. A description of our considered problem can be seen in Figure 5.1, in which, the gating and expert parameters are explicitly indicated for each component of the local and the reduced models.

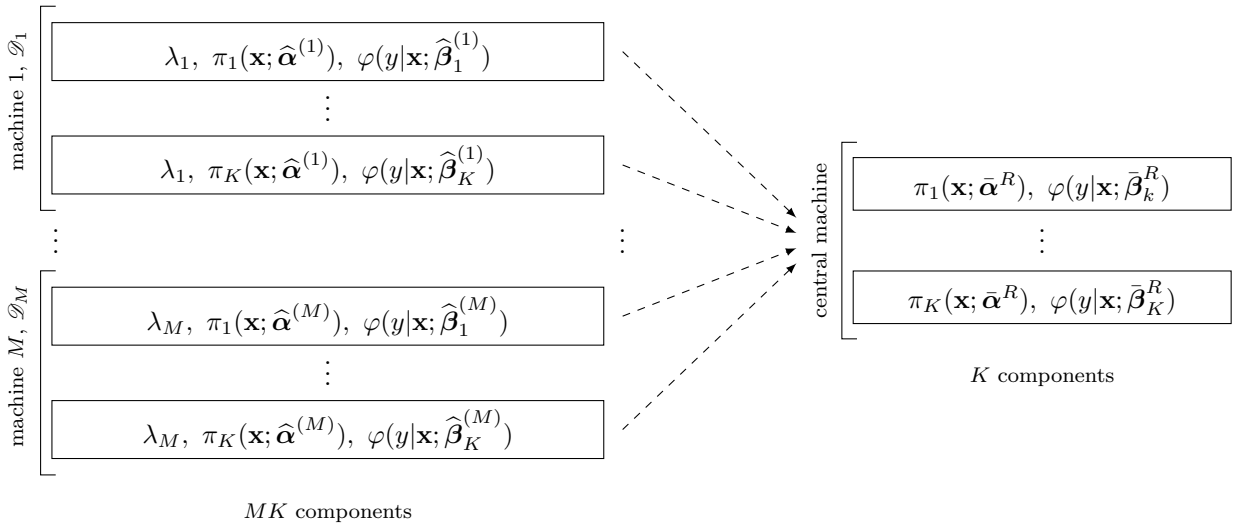


Figure 5.1: Description of the considered aggregation problem. Left panels are the components of the local ME models with their corresponding estimated parameters. Right panels are the components of the aggregated K-component ME model, *i.e.*, the desired solution. The unknowns are therefore $\bar{\boldsymbol{\alpha}}^R$, $\bar{\boldsymbol{\beta}}_1^R, \ldots, \bar{\boldsymbol{\beta}}_K^R$.

5.2.3 Some remarks and notation conventions

Recall that, as we can see in equation (5.4), the weighted average mixture $\bar{f}^W$ has totally MK terms of the form $\lambda_m \pi_k(\mathbf{x};\widehat{\boldsymbol{\alpha}}^{(m)})\varphi(\cdot|\mathbf{x};\widehat{\boldsymbol{\beta}}_k^{(m)})$, for $m \in [M]$, $k \in [K]$. However, due to the presence of the weights λ_m and the fact that the local gating parameters $\widehat{\boldsymbol{\alpha}}^{(1)}, \ldots, \widehat{\boldsymbol{\alpha}}^{(M)}$ are not yet integrated into a single gating parameter vector, $\bar{f}^W$ do not has the standard form of a ME model as in (5.1). For mathematical convenience, we will incorporate the weights λ_m into the local gating functions $\pi_k(\mathbf{x};\widehat{\boldsymbol{\alpha}}^{(m)})$ and write them simply by $\{\widehat{\pi}_\ell(\mathbf{x})\}_{\ell\in[MK]}$, which will be also referred to as gating functions. The corresponding experts in $\bar{f}^W$ will be abbreviated by $\{\widehat{\varphi}_\ell(y|\mathbf{x})\}_{\ell\in[MK]}$. Similarly, for $g \in \mathcal{M}_K$, we will abbreviate its K gating functions by $\{\pi_k(\mathbf{x})\}_{k\in[K]}$, and its K experts by

$\{\varphi_k(y|\mathbf{x})\}_{k\in[K]}$. We then have

$$\bar{f}^W = \sum_{\ell=1}^{MK} \widehat{\pi}_\ell(\mathbf{x})\widehat{\varphi}_\ell(\cdot|\mathbf{x}) \quad \text{and} \quad g = \sum_{k=1}^{K} \pi_k(\mathbf{x})\varphi_k(\cdot|\mathbf{x}), \tag{5.7}$$

for $g \in \mathcal{M}_K$. Here, the dependence of the functions on the parameters is understood implicitly. Table 5.1 summarizes our notations for this chapter.

Symbol	Explanation
$\boldsymbol{\theta}$	Parameter vector of the model
$\mathbf{x}_i, y_i$	Training data. Without subscript they refer to generic variables
K, M	Number of components of the true model, number of local machines
$\mathbb{U}$	The interval $[0,1] \subset \mathbb{R}$
$\bar{f}^W$, $\bar{f}^B$, $\bar{f}^R$	Weighted average, barycenter and reduction densities
$\{\pi_k(\mathbf{x})\}_{k\in[K]}$	Gating functions $\pi_k(\mathbf{x};\boldsymbol{\alpha})$, $k \in [K]$, with unknown $\boldsymbol{\alpha}$
$\{\varphi_k(\cdot\|\mathbf{x})\}_{k\in[K]}$	Expert functions $\varphi(\cdot\|\mathbf{x};\boldsymbol{\beta}_k)$, $k \in [K]$, with unknown $\boldsymbol{\beta}_k$
$\{\widehat{\pi}_\ell(\mathbf{x})\}_{\ell\in[MK]}$	Gating functions $\pi_k(\mathbf{x};\widehat{\boldsymbol{\alpha}}^{(m)})$, $m \in [M]$, $k \in [K]$
$\{\widehat{\varphi}_\ell(\cdot\|\mathbf{x})\}_{\ell\in[MK]}$	Expert functions $\varphi(\cdot\|\mathbf{x};\widehat{\boldsymbol{\beta}}_k^{(m)})$, $m \in [M]$, $k \in [K]$
$\mathcal{T}_c(h,g)$	Transportation divergence between mixtures h and g
$\mathcal{T}_c(g)$	Abbreviation of $\mathcal{T}_c(\bar{f}^W, g)$
$\mathcal{R}_c(g)$	Relaxation of $\mathcal{T}_c(g)$ from the constraints involving $\boldsymbol{\pi}$
$\mathcal{S}_c(g, g^{(t)})$	Majorant function of $\mathcal{R}_c(g)$ at $g^{(t)}$

Table 5.1: Summary of notation

5.3 Optimal transport approach for the reduction strategy

Let us substitute (5.3) into (5.6) and write the problem more explicitly as

$$\bar{f}^R = \underset{g\in\mathcal{M}_K}{\arg\inf}\ \rho\left(\sum_{m=1}^{M} \lambda_m \widehat{f}_m, g\right), \tag{5.8}$$

that is, we are seeking for a K-component ME model g of form as in (5.1) that is closest to the MK-component mixture $\bar{f}^W = \sum_{m=1}^{M} \lambda_m \widehat{f}_m$ with respect to some divergence $\rho(\cdot,\cdot)$. In general, an analytical solution is difficult to obtain for such reduction problem, especially in the context of ME models where the conditional density function is made up of several different gating and expert functions. Therefore, the aim of this section is to propose a framework to find a numerical solution

for the the reduction model $\bar{f}^R$.

We observe that, firstly, the solution depends heavily on the choice of the divergence $\rho(\cdot,\cdot)$, which measures the *goodness* of a candidate model $g \in \mathcal{M}_K$. A good choice of ρ should take into account the dissimilarity between the experts, as well as the weights incorporated in the gating functions between the mixtures. Then, the best candidate mixture g should minimize the divergence from the large mixture $\bar{f}^W$. Secondly, compute the divergence between two mixtures is difficult, while it would be easier to compute the divergence between two component densities, which may has explicit formula in some specific cases. Therefore, borrowing the idea in Zhang and Chen (2021) that finds a reduction estimator for finite Gaussian mixtures based on minimizing transportation divergence between the large starting mixture and the desired mixture, we establish a framework for solving a reduction estimator for ME models, which is generally formalized as in (5.8). Particularly, in the case $\varphi(y|\mathbf{x};\boldsymbol{\beta})$ being Gaussian experts, if conditionally on one single observation of $\mathbf{x}$, problem (5.8) can be viewed as the problem of finding reduction estimator for finite mixtures of 1D Gaussian distributions.

5.3.1 Expected transportation divergence

Let $h = \sum_{\ell=1}^{L} \widehat{\pi}_\ell(\mathbf{x})\widehat{\varphi}_\ell(y|\mathbf{x})$, $L \in \mathbb{N}$, and $g = \sum_{k=1}^{K} \pi_k(\mathbf{x})\varphi_k(y|\mathbf{x})$ be two mixture models of L and K components, respectively. Here, h will play the role as the mixture $\bar{f}^W$ in problem (5.6). We wish to define a divergence that measures the dissimilarity between the two mixtures h and g. Let $\boldsymbol{\Phi}$ denotes the family of the component conditional densities $\varphi(\cdot|\mathbf{x};\boldsymbol{\beta}) =: \varphi(\cdot|\mathbf{x})$. Let $\widehat{\boldsymbol{\pi}}$ and $\boldsymbol{\pi}$ be, respectively, the column vectors of gating functions $\widehat{\pi}_\ell(\cdot)$'s and $\pi_k(\cdot)$'s. For $\mathbf{x} \in \mathcal{X}$, we denote

$$\boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \boldsymbol{\pi}) := \boldsymbol{\Pi}\left(\widehat{\boldsymbol{\pi}}(\mathbf{x}), \boldsymbol{\pi}(\mathbf{x})\right) = \left\{ \boldsymbol{P} \in \mathbb{U}^{L\times K} \colon \boldsymbol{P}\mathbb{1}_K = \widehat{\boldsymbol{\pi}}(\mathbf{x}), \boldsymbol{P}^\top \mathbb{1}_L = \boldsymbol{\pi}(\mathbf{x}) \right\},$$

where $\mathbb{U}$ denotes the interval $[0,1]$, $\mathbb{1}_K$ and $\mathbb{1}_K$ are vectors of all ones. In other words, for $\mathbf{x} \in \mathcal{X}$, $\boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \boldsymbol{\pi})$ denotes the set of all matrices $\boldsymbol{P}$ of size $L \times K$, with entries $P_{\ell k} \in [0,1]$, satisfying the marginal constraints

$$\sum_{k=1}^{K} P_{\ell k} = \widehat{\pi}_\ell(\mathbf{x}), \quad \text{for all } \ell \in [L], \tag{5.9a}$$

$$\sum_{\ell=1}^{L} P_{\ell k} = \pi_k(\mathbf{x}), \quad \text{for all } k \in [K]. \tag{5.9b}$$

Given $\mathbf{x} \in \mathcal{X}$, one can think of such matrix $\boldsymbol{P} \in \boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \boldsymbol{\pi})$ as a plan that transports an amount of material distributed according to $\widehat{\boldsymbol{\pi}}(\mathbf{x})$ to distributed according to $\boldsymbol{\pi}(\mathbf{x})$, for short, we will say "transports $\widehat{\boldsymbol{\pi}}(\mathbf{x})$ to $\boldsymbol{\pi}(\mathbf{x})$". Suppose that the transportation is costing for each unit of material. We are then interested in the optimal way, *i.e.*, with a minimum cost, to transports $\widehat{\boldsymbol{\pi}}(\mathbf{x})$ to $\boldsymbol{\pi}(\mathbf{x})$. The optimal plan is clearly depends on $\mathbf{x}$. We will write $\boldsymbol{P}(\mathbf{x})$ to indicate such optimal plan. Figure 5.2 gives an example of transportation plans.

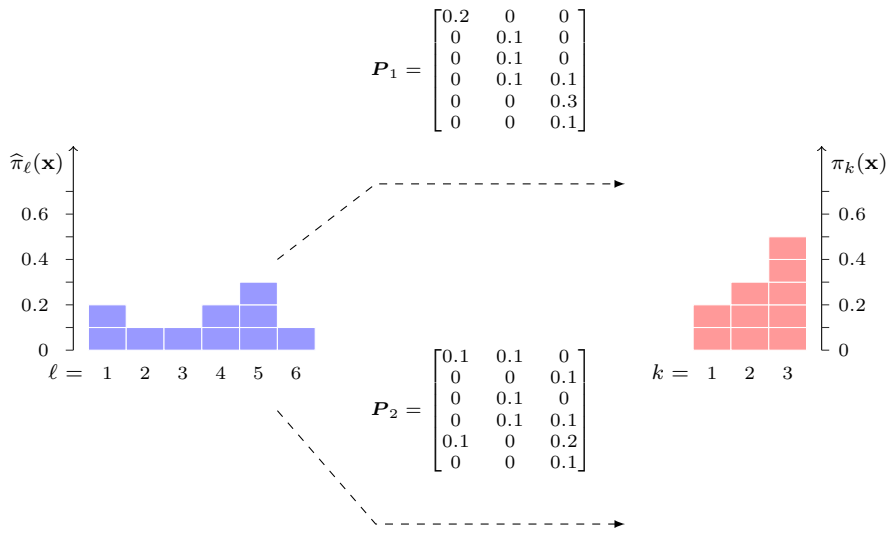


Figure 5.2: Example of transportation plans with $L = 6$ and $K = 3$. Given $\mathbf{x} \in \mathcal{X}$, the vectors $\widehat{\boldsymbol{\pi}}(\mathbf{x})$ and $\boldsymbol{\pi}(\mathbf{x})$ can be viewed as the distributions of the weights (sum to one). Here, for example, the weights are distributed as in the left and right panels. $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$ are two valid plans those transport $\widehat{\boldsymbol{\pi}}(\mathbf{x})$ in two different ways to $\boldsymbol{\pi}(\mathbf{x})$.

Definition 5.3.1 (Expected transportation divergence)**.** *Given the above notations, the expected transportation divergence between two mixtures h and g is defined by*

$$\mathcal{T}_c(h, g) = \mathbb{E}\left[\inf_{\boldsymbol{P} \in \boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \boldsymbol{\pi})} \sum_{\ell=1}^{L} \sum_{k=1}^{K} P_{\ell k}\, c\left(\widehat{\varphi}_\ell(\cdot|\mathbf{x}), \varphi_k(\cdot|\mathbf{x})\right)\right] =: \mathbb{E}\Big[\mathcal{T}_c(h, g, \mathbf{x})\Big], \tag{5.10}$$

where $c(\cdot, \cdot)$ is a real-valued bivariate function defined on $\boldsymbol{\Phi} \times \boldsymbol{\Phi}$ satisfies $c(\varphi_1, \varphi_2) \geqslant 0$ for all $\varphi_1, \varphi_2 \in \boldsymbol{\Phi}$, and the equality holds if and only if $\varphi_1 = \varphi_2$.

Here, the expectation is taken with respect to $\mathbf{x}$ and the function c is often referred to as the cost function. Assume that, for a given $\mathbf{x} \in \mathcal{X}$, $\boldsymbol{P}^*(\mathbf{x})$ is an optimal solution to the optimization problem inside the expectation operator in (5.10). Then the value of $\mathcal{T}_c(h, g, \mathbf{x})$ can be interpreted as the optimal total cost of transporting $\widehat{\boldsymbol{\pi}}(\mathbf{x})$ to $\boldsymbol{\pi}(\mathbf{x})$ with the unit cost of transportation is proportional to the values $c\left(\widehat{\varphi}_\ell(\cdot|\mathbf{x}), \varphi_k(\cdot|\mathbf{x})\right)$'s. This interpretation is common in optimal transportation literature under the name Kantorovich formulation, *e.g.*, Oberman and Ruan (2015). The value $\mathcal{T}_c(h, g)$ is therefore defined as the expected value of these optimal transportation costs.

The cost function c must be chosen such that it not only reflects the dissimilarity between the conditional densities $\widehat{\varphi}_\ell(\cdot|\mathbf{x})$ and $\varphi_k(\cdot|\mathbf{x})$, but also has to be easy to calculate. Later we will see that c being KL-divergence is a suitable choice for both the cases when the experts are Gaussian regression models (*i.e.*, for regression problems), and when the experts are logistic regression models (*i.e.*, for classification problems).

Thus, by considering $\rho(\cdot,\cdot)$ being the expected transportation divergence $\mathcal{T}_c(\cdot,\cdot)$, problem (5.6) becomes

$$\begin{aligned}\bar{f}^R &= \arg\inf_{g\in\mathcal{M}_K} \mathcal{T}_c(\bar{f}^W, g) \\ &= \arg\inf_{g\in\mathcal{M}_K} \left\{ \mathbb{E}\left[\inf_{\boldsymbol{P}\in\boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}},\boldsymbol{\pi})} \sum_{\ell=1}^{MK}\sum_{k=1}^{K} P_{\ell k}\; c\left(\widehat{\varphi}_\ell(\cdot|\mathbf{x}), \varphi_k(\cdot|\mathbf{x})\right)\right]\right\}. \end{aligned} \tag{5.11}$$

From now on, we will use $\mathcal{T}_c(g)$ to refer to $\mathcal{T}_c(\bar{f}^W, g)$, since our objective now is to minimize the transportation divergence $\mathcal{T}_c(\bar{f}^W, g)$ as a function of $g \in \mathcal{M}_K$.

Problem (5.11) appears to involve two optimizations: one over $\boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \boldsymbol{\pi})$ and one over $\mathcal{M}_K$. However, we will show that the constraint $\boldsymbol{P} \in \boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \boldsymbol{\pi})$ in fact can be relaxed to $\boldsymbol{P} \in \boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \cdot)$. This means we only need $\boldsymbol{P}$ to satisfy (5.9a), the constraints (5.9b) are redundant. In other word, instead of searching, for each $\mathbf{x}$, a plan $\boldsymbol{P}$ that satisfies $\boldsymbol{\pi}(\mathbf{x})$, we can move $\boldsymbol{\pi}$, the gating functions make up g, to match $\boldsymbol{P}$.

Indeed, for $\bar{f}^W$ and g as in (5.7), let us define $\mathcal{R}_c(\bar{f}^W, g)$ as a function of g as follow

$$\mathcal{R}_c(\bar{f}^W, g) = \mathbb{E}\left[\inf_{\boldsymbol{P}\in\boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}},\cdot)} \sum_{\ell=1}^{MK}\sum_{k=1}^{K} P_{\ell k}\; c\left(\widehat{\varphi}_\ell(\cdot|\mathbf{x}), \varphi_k(\cdot|\mathbf{x})\right)\right] =: \mathbb{E}\Big[\mathcal{R}_c(\bar{f}^W, g, \mathbf{x})\Big]. \tag{5.12}$$

Here, the $MK \times K$ matrix $\boldsymbol{P}$ is now free of the constraints involving the gating network $\boldsymbol{\pi}$ of g, namely, for all $\mathbf{x} \in \mathcal{X}$, the equality $\boldsymbol{P}^\top \mathbb{1}_{MK} = \boldsymbol{\pi}(\mathbf{x})$ need not to be hold. Then, the following proposition will claim that solving $\bar{f}^R = \arg\inf_{g\in\mathcal{M}_K} \mathcal{T}_c(\bar{f}^W, g)$ can actually be reduced to solving $\bar{f}^R = \arg\inf_{g\in\mathcal{M}_K} \mathcal{R}_c(\bar{f}^W, g)$, which is much easier.

Before presenting the proposition, we define $\mathcal{P}(\bar{f}^W, g, \mathbf{x})$ a function of $g \in \mathcal{M}_K$ and $\mathbf{x} \in \mathcal{X}$ as follow

$$\mathcal{P}(\bar{f}^W, g, \mathbf{x}) = \arg\inf_{\boldsymbol{P}\in\boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}},\cdot)} \sum_{\ell=1}^{MK}\sum_{k=1}^{K} \left[P_{\ell k}\; c\left(\widehat{\varphi}_\ell(\cdot|\mathbf{x}), \varphi_k(\cdot|\mathbf{x})\right)\right]. \tag{5.13}$$

This function returns the optimal plan for the optimization problem inside the expectation operator in the definition of $\mathcal{R}_c(\bar{f}^W, g)$. For simplicity of notation, similarly to $\mathcal{T}_c(g)$, from now on we let the dependence of the functions on $\bar{f}^W$ in the background, *i.e.*, $\mathcal{R}_c(\bar{f}^W, g)$, $\mathcal{R}_c(\bar{f}^W, g, \mathbf{x})$ and $\mathcal{P}(\bar{f}^W, g, \mathbf{x})$ will be written by $\mathcal{R}_c(g)$, $\mathcal{R}_c(g, \mathbf{x})$ and $\mathcal{P}(g, \mathbf{x})$, respectively.

Proposition 5.3.1. *Let $\mathcal{T}_c(g), \mathcal{R}_c(g)$ and $\mathcal{P}(g, \mathbf{x})$ defined as above. Then*

$$\inf_{g \in \mathcal{M}_K} \mathcal{T}_c(g) = \inf_{g \in \mathcal{M}_K} \mathcal{R}_c(g).$$

The reduction solution is hence given by

$$\bar{f}^R = \arg\inf_{g \in \mathcal{M}_K} \mathcal{R}_c(g), \tag{5.14}$$

and the gating functions of $\bar{f}^R$ can be calculated by

$$\pi_k(\mathbf{x}) = \sum_{\ell=1}^{MK} \mathcal{P}_{\ell k}(\bar{f}^R, \mathbf{x}), \quad \forall \mathbf{x} \in \mathcal{X}, \tag{5.15}$$

where $\mathcal{P}_{\ell k}(\bar{f}^R, \mathbf{x})$ denotes the entry (ℓ, k) of $\mathcal{P}(\bar{f}^R, \mathbf{x})$.

The proof of Proposition 5.3.1 is given in Appendix C.1.1. It is important to note here that the equation (5.15) only gives us an expression for computing the values of the gating network for each $\mathbf{x} \in \mathcal{X}$, it does not yet have the parametric form as in the standard model (5.1). Thus, to obtain the desired gating parameter $\bar{\boldsymbol{\alpha}}^R$ as described in Figure 5.1, we need to perform an additional estimation step as in equation (5.21), which will become clearer later in Subsection 5.4.2.

Thus, thanks to Proposition 5.3.1, our objective now is to minimize $\mathcal{R}_c(g)$ with respect to g. Replacing $\mathcal{T}_c(g)$ by $R_c(g)$, problem (5.11) becomes

$$\begin{aligned} \bar{f}^R &= \arg\inf_{g \in \mathcal{M}_K} \mathcal{R}_c(g) \\ &= \arg\inf_{g \in \mathcal{M}_K} \left\{ \mathbb{E}\left[\inf_{\boldsymbol{P} \in \boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \cdot)} \sum_{\ell=1}^{MK} \sum_{k=1}^{K} P_{\ell k}\, c\left(\widehat{\varphi}_\ell(\cdot|\mathbf{x}), \varphi_k(\cdot|\mathbf{x})\right) \right] \right\}. \end{aligned} \tag{5.16}$$

This optimization can be done with the help of majorization-minimization (MM) algorithm (*e.g.*, Lange (2016)). We defer the numerical approach for solving the problem (5.16) in Section 5.4, now we will show that the problem is well-posed and state the conditions for the consistency of the reduction estimator.

5.3.2 Well-posedness and consistency of the reduction estimator

We will make the following standard assumptions.

- A1 The dataset $\mathscr{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$ is an i.i.d. sample from the K-component ME model $f(y|\mathbf{x}, \boldsymbol{\theta}^*)$, in which the parameters are ordered and initialized.
- A2 The cost function $c(\cdot, \cdot)$ is continuous in both arguments, and $c(\varphi_1, \varphi_2) \to 0$ if and only if $\varphi_1 \to \varphi_2$ in distribution.

A3 The cost function $c(\cdot,\cdot)$ is convex in the second arguments. This property holds for most of popular divergences such as Kullback Liebler, Hellinger distance, ..., see *e.g.*, Dragomir (2013).

■ **Well-posedness.** First, we observe that for any $\mathbf{x} \in \mathcal{X}$, the minimization problem inside the expectation operator in (5.16) is in fact a linear programming (LP) problem. Moreover, for $\mathbf{x} \in \mathcal{X}$, we see that $\mathbf{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \cdot)$ is a nonempty set, and the sum $\sum_{\ell=1}^{MK}\sum_{k=1}^{K} P_{\ell k} c\left(\bar{\varphi}_\ell(\cdot|\mathbf{x}), \varphi_k(\cdot|\mathbf{x})\right)$ is bounded below by zero for all $\boldsymbol{P} \in \mathbf{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \cdot)$. Therefore, this LP problem has a global minimizer (*e.g.*, see Boyd and Vandenberghe (2004)). Hence, for all $g \in \mathcal{M}_K$, there exists non-negative finite expectation for the random quantity $\mathcal{R}_c(g, \mathbf{x})$.

Now, for a mixture $g \in \mathcal{M}_K$ as in (5.7), we let $\mathcal{C}(\boldsymbol{P}, \boldsymbol{\varphi})$ be a function of the transport plan with respect to the component densities of g, given by

$$\mathcal{C}(\boldsymbol{P}, \boldsymbol{\varphi}) = \sum_{\ell=1}^{MK}\sum_{k=1}^{K} P_{\ell k}\ c\left(\widehat{\varphi}_\ell(\cdot|\mathbf{x}), \varphi_k(\cdot|\mathbf{x})\right),$$

where $\boldsymbol{P} \in \mathbb{U}^{MK\times K}$, $\mathbb{U} = [0,1] \subset \mathbb{R}$, and $\boldsymbol{\varphi} = (\varphi_1(\cdot|\mathbf{x}), \ldots, \varphi_K(\cdot|\mathbf{x})) \in \mathbf{\Phi}^K$. Here, we can think of the family $\mathbf{\Phi}$ as the space of parameter vectors that make up the density $\varphi(\cdot|\mathbf{x})$ (typically $\mathbb{R}^\kappa$, for some $\kappa \in \mathbb{N}^*$). Then, the assumptions A2 and A3 on the continuity and convexity of the cost function c immediately lead to the continuity and convexity in $\boldsymbol{\varphi}$ of the function $\mathcal{C}(\boldsymbol{P}, \boldsymbol{\varphi})$. Moreover, $\mathcal{C}(\boldsymbol{P}, \boldsymbol{\varphi})$ as an affine function is obviously continuous and convex in $\boldsymbol{P}$.

Next, for $\mathbf{x} \in \mathcal{X}$, let $\mathcal{I}(g, \mathbf{x})$ be a function of g and $\mathbf{x}$ defined by

$$\mathcal{I}(g, \mathbf{x}) = \inf_{\boldsymbol{P} \in \mathbf{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \cdot)} \mathcal{C}(\boldsymbol{P}, \boldsymbol{\varphi}),$$

i.e., the optimal cost given g and $\mathbf{x}$ inside the expectation operator in (5.16). We see that, the value of $\mathcal{I}(g, \mathbf{x})$ depends on the mixture g only through the component densities $\boldsymbol{\varphi}$, but the gating functions. Then, the objective function of the problem (5.16) can be written as $\mathcal{R}_c(g) = \mathbb{E}\left[\mathcal{I}(g, \mathbf{x})\right]$. We have the following proposition.

Proposition 5.3.2. *Given the notations as above. Assume in addition that there exists $\Delta \in \mathbb{R}_+$ such that $\mathcal{I}(g, \mathbf{x}) \leqslant \Delta$ for all $g \in \mathcal{M}_K$, $\mathbf{x} \in \mathcal{X}$. Then $\mathcal{R}_c(g)$ is continuous and convex as a function of $g \in \mathcal{M}_K$. It follows that the problem (5.16) has a global solution.*

The proof of the Proposition 5.3.2 is given in Appendix C.1.2. Note that the condition on the boundedness of the optimal transportation costs is common, and the solution to the problem (5.16) is not necessarily unique.

■ **Consistency.** The reduction estimator $\bar{\boldsymbol{\theta}}^R$ has a desired property that it is a consistent estimator of the true parameter $\boldsymbol{\theta}^*$ as soon as the local estimators are consistent estimators of $\boldsymbol{\theta}^*$. We have the following proposition.

Proposition 5.3.3. *Let $\bar{\boldsymbol{\theta}}^R$ be the parameter of the reduction density $\bar{f}^R$ defined in (5.6) with ρ being the expected transportation divergence $\mathcal{T}_c$. Suppose assumptions A1-A2 are satisfied. Then $\bar{\boldsymbol{\theta}}^R$ is a consistent estimator of $\boldsymbol{\theta}^*$.*

The assumption that the parameters are ordered and initialized in Assumption A1 is necessary because under it the ME models are identifiable (Jiang and Tanner, 1999). The proof of Proposition 5.3.3 is given in Appendix C.1.3.

5.4 An MM algorithm for constructing the reduction estimator

5.4.1 The main algorithm

We see that, the optimization problem (5.16) is accompanied by the objective function $\mathcal{R}_c(g)$ defined upon another optimization, with respect to the transportation plan $\boldsymbol{P}$, that makes the approach such as gradient descent hard to apply directly. In this section, we present the approach of using the MM algorithm, which requires the definition of a so-called majorant function, to solve the problem (5.16). We will apply it to the two common specifications of ME models: softmax-gated mixtures Gaussian regression and softmax-gated mixtures logistic regression models.

MM algorithm (Lange, 2004) is an iterative procedure that consists of alternating between two steps: *Majorize* the objective function at the current iterate with a majorant function, and *Minimize* the majorant function to define the next iterate. The sequence of minimizers of the majorant functions is guaranteed to converge to a stationary point of the objective function. Employing MM algorithm to our problem means that we will starts from an initial model $g^{(0)} \in \mathcal{M}_K$, then at each iteration tth, find a majorant function for $\mathcal{R}_c(g)$ at $g^{(t)}$, and minimize it to obtain the next model $g^{(t+1)}$. The generated sequence $(g^{(t)})_{t\geqslant 1}$ satisfies $\mathcal{R}_c(g^{(0)}) \geqslant \mathcal{R}_c(g^{(1)}) \geqslant \dots$ We keep running until convergence is reached, *i.e.*, there is no significant change in the value of $\mathcal{R}_c(g)$. The model obtained at convergence is then supposed to be the desired model $\bar{f}^R$. Note that, for a general objective function, MM algorithm only guarantees to converge to a stationary point, not a global or local minimum, so multiple initializations are often required in order to obtain high quality solution (Nguyen, 2016).

The following proposition gives us a majorant function for $\mathcal{R}_c(g)$.

Proposition 5.4.1. *Let $g^{(t)}$ be the model obtained at MM iteration t-th. Let*

$$\mathcal{S}_c(g, g^{(t)}) = \mathbb{E}\left[\sum_{\ell=1}^{MK}\sum_{k=1}^{K} \mathcal{P}_{\ell k}(g^{(t)}, \mathbf{x})\ c(\widehat{\varphi}_\ell(\cdot|\mathbf{x}), \varphi_k(\cdot|\mathbf{x}))\right], \tag{5.17}$$

where $\mathcal{P}_{\ell k}(g^{(t)}, \mathbf{x})$ is given by

$$\mathcal{P}_{\ell k}(g^{(t)}, \mathbf{x}) = \begin{cases} \widehat{\pi}_\ell(\mathbf{x}) & \text{if } k = \underset{k' \in [K]}{\arg\inf}\ c(\widehat{\varphi}_\ell(\cdot|\mathbf{x}), \varphi_{k'}^{(t)}(\cdot|\mathbf{x})) \\ 0 & \text{otherwise.} \end{cases} \tag{5.18}$$

Then $\mathcal{S}_c(g, g^{(t)})$ is a majorant function of $\mathcal{R}_c(g)$ at $g^{(t)}$.

The proof of Proposition 5.4.1 is in Appendix C.1.4. Here we see that, the plan with entries $\mathcal{P}_{\ell k}(g^{(t)}, \mathbf{x})$ given in (5.18) is an optimal plan for the optimization problem (5.13) when $g = g^{(t)}$ (this will be clear in the proof of the proposition). Note that, similarly to $\mathcal{R}_c(g)$, the function $\mathcal{S}_c(g, g^{(t)})$ varies only through the parameters of the conditional densities $\varphi_k(\cdot|\mathbf{x})$'s.

Then, the next step is to minimize $\mathcal{S}_c(g, g^{(t)})$, with respect to g, to obtain a next point $g^{(t+1)}$ for the MM algorithm. As we can see, the optimization for $\mathcal{S}_c(g, g^{(t)})$ can be performed separately. At each iteration tth, the parameter vector of the expert k can be optimized via

$$\varphi_k^{(t+1)}(\cdot|\mathbf{x}) = \underset{\varphi \in \mathbf{\Phi}}{\arg\inf}\left\{\mathbb{E}\left[\sum_{\ell=1}^{MK} \mathcal{P}_{\ell k}(g^{(t)}, \mathbf{x})\ c(\widehat{\varphi}_\ell(\cdot|\mathbf{x}), \varphi(\cdot|\mathbf{x}))\right]\right\}. \tag{5.19}$$

According to each specification of the experts and the cost function $c(\cdot, \cdot)$, problem (5.19) has a certain form that is supposed to be easy to solve, as we will see later for the cases of the Gaussian and logistic regression experts, with c is the KL-divergence.

Hence, our algorithm will alternate between two following steps until convergence:

- calculating majorant function $\mathcal{S}_c(g, g^{(t)})$ as in (5.17), which in fact only involves updating $\mathcal{P}_{\ell k}(g^{(t)}, \mathbf{x})$ according to (5.18);
- updating the experts parameters by solving (5.19).

At the convergence, the parameters of all the experts are optimized, so it remains to update the parameter of the gating functions. To do that, we rely on the equation (5.15), in which the theoretical solution $\bar{f}^R$ is replaced by the model g^* obtained at the convergence of the MM algorithm. This procedure of updating the gating parameter will be presented in Subsection 5.4.2.

As we can see, up to this point all the optimizations carried on the central machine are involving the expectations with respect to $\mathbf{x}$. In practice, the distribution of $\mathbf{x}$ is unknown. Therefore, our approach is to create a small sample $\mathscr{D}_S = \{(\mathbf{x}_s, y_s)\}_{s=1}^S$ of size S by randomly drawing from the full dataset $\mathscr{D}$. This $\mathscr{D}_S$ acts as a supporting sample for the calculations on the central machine. Then, the expectation $\mathbb{E}_{\mu(\mathbf{x})}[\,\cdot\,]$ will be understood as the empirical expectation calculated based

on the empirical distribution $\mu(\cdot) = S^{-1}\sum_{s=1}^{S}\delta_{\mathbf{x}_s}(\cdot)$. The size of $\mathscr{D}_S$ is also an issue for further consideration. However, let us assume that $\mathscr{D}_S$ has a sufficiently large size to characterize the distribution of the variable $\mathbf{x}$. We will denote by $\boldsymbol{X}_S$ the set of all observations $\mathbf{x}_s$ in the supporting sample $\mathscr{D}_S$, and where there is no ambiguity we also use $\boldsymbol{X}_S$ to denote the matrix with each row corresponds to one row vector $\mathbf{x}_s^\top$.

The Algorithm 5.1 gives a pseudo-code that summarizes our main algorithm of aggregating ME models.

Algorithm 5.1 Aggregating ME local estimators

Input:
- local gating parameters $\widehat{\boldsymbol{\alpha}}^{(m)}$, $m \in [M]$;
- local expert parameters $\widehat{\boldsymbol{\beta}}_k^{(m)}$, $m \in [M]$, $k \in [K]$;
- sample proportions λ_m, $m \in [M]$.

Output: Reduction estimator $\bar{\boldsymbol{\theta}}^R = (\bar{\boldsymbol{\alpha}}^R, \bar{\boldsymbol{\beta}}_1^R, \ldots, \bar{\boldsymbol{\beta}}_K^R)$.

Initialize a ME model $g^{(0)}$, *i.e.*, initialize $\boldsymbol{\theta}^{(0)} = (\boldsymbol{\alpha}^{(0)}, \boldsymbol{\beta}_1^{(0)}, \ldots, \boldsymbol{\beta}_K^{(0)})$. Assign $t \longleftarrow 0$
Sampling a supporting sample $\mathscr{D}_S$ from $\mathscr{D}$
repeat
  for $k \in [K]$ **do**
    for $\ell \in [MK]$ **do**
      For all $\mathbf{x} \in \mathscr{D}_S$, compute $\mathcal{P}_{\ell k}(g^{(t)}, \mathbf{x})$ according to (5.18)
    end for
    Update expert parameters by solving (5.19). Depending on weather the expert is Gaussian or logistic regression model, update formula takes the form (5.23) or (5.23), respectively
  end for
  Assign $t \longleftarrow t + 1$
until the objective function (5.17) converges.
Estimate the gating parameter according to (5.21).

5.4.2 Updating the gating network parameter

As mentioned earlier, although the equation (5.15) gives a formula to compute the mixing weights for every covariate $\mathbf{x}$, to obtain the final desired ME model as described in Figure 5.1, it is necessary to find the parameter $\bar{\boldsymbol{\alpha}}^R$ for the gating functions $\pi_k(\cdot, \bar{\boldsymbol{\alpha}}^R)$. Therefore, we propose to estimate $\bar{\boldsymbol{\alpha}}^R$ via solving a softmax regression problem in which the predictors are $\boldsymbol{X}_S$ and the responses are computed according to (5.15). In particular, let $\boldsymbol{A} = [a_{sk}]$ be a matrix in $\mathbb{R}^{S\times K}$ defined by

$$a_{sk} = \sum_{\ell=1}^{MK} \mathcal{P}_{\ell k}(g^*, \mathbf{x}_s), \tag{5.20}$$

where g^* is the model obtained at convergence of the MM algorithm, $\mathbf{x}_s \in \mathscr{D}_S$, and $\mathcal{P}_{\ell k}(g^*, \mathbf{x}_s)$ is computed according to (5.18). Then $\bar{\boldsymbol{\alpha}}^R$ can be estimated via the maximum likelihood approach

$$\begin{aligned}\bar{\boldsymbol{\alpha}}^R &= \arg\max_{\boldsymbol{\alpha}} \sum_{s=1}^{S}\sum_{k=1}^{K} a_{sk} \log \pi_k(\mathbf{x}_s; \boldsymbol{\alpha}) \\ &= \arg\max_{\boldsymbol{\alpha}} \sum_{s=1}^{S} \left[\sum_{k=1}^{K-1} a_{sk} \boldsymbol{\alpha}_k^\top \mathbf{x}_s - \log\left(1 + \sum_{k'=1}^{K-1} \exp\{\boldsymbol{\alpha}_{k'}^\top \mathbf{x}_s\}\right)\right]. \end{aligned} \tag{5.21}$$

This problem can be efficiently solved by the Newton-Raphson procedure, for example as used for updating the gating network in the M-Step of the EM algorithm in Section 2.3.1. Here, we note that the parameter vector is also initialized, *i.e.*, the Kth gate's parameter is constrained to be zero, this is necessary for proving the consistency of $\bar{\boldsymbol{\theta}}^R$.

Now, we are left with the question of how to obtain the update for the expert parameter $\boldsymbol{\beta}_k^{(t+1)}$ as the solution of (5.19). In the next subsections, we will present the update formulas for the two cases of the Gaussian regression expert and logistic regression expert when the cost function $c(\cdot, \cdot)$ is KL-divergence.

5.4.3 Updating the Gaussian regression experts parameter

Now, we consider the case of $\varphi(\cdot|\mathbf{x})$ being Gaussian regression experts, *i.e.*, $\varphi(\cdot|\mathbf{x}) = \varphi(\cdot|\mathbf{x}^\top\boldsymbol{\beta}; \sigma^2)$, where $\boldsymbol{\beta} \in \mathbb{R}^d$ and $\sigma^2 \in \mathbb{R}_+$ are the parameters of the expert. The KL-divergence between two Gaussian regression experts takes the following form

$$\mathrm{KL}\left(\varphi_1(\cdot|\mathbf{x}) \| \varphi_2(\cdot|\mathbf{x})\right) = \frac{1}{2}\left(\log\frac{\sigma_2^2}{\sigma_1^2} + \frac{\sigma_1^2}{\sigma_2^2} + \frac{(\boldsymbol{\beta}_2 - \boldsymbol{\beta}_1)^\top \mathbf{x}\mathbf{x}^\top (\boldsymbol{\beta}_2 - \boldsymbol{\beta}_1)}{\sigma_2^2} - 1\right).$$

Therefore, with the experts being Gaussian and the cost function $c(\cdot, \cdot)$ being $\mathrm{KL}(\cdot\|\cdot)$, the objective function of the problem (5.19) can be written as a function of $\boldsymbol{\beta}$ and σ^2 as

$$\mathcal{K}(\boldsymbol{\beta}, \sigma^2) := \mathbb{E}\left[\sum_{\ell=1}^{MK} \mathcal{P}_{\ell k}^{(t)}(\mathbf{x}) \frac{1}{2}\left(\log\frac{\sigma^2}{\widehat{\sigma}_\ell^2} + \frac{\widehat{\sigma}_\ell^2}{\sigma^2} + \frac{(\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}_\ell)^\top \mathbf{x}\mathbf{x}^\top (\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}_\ell)}{\sigma^2} - 1\right)\right], \tag{5.22}$$

where $\mathcal{P}_{\ell k}^{(t)}(\mathbf{x})$ denotes $\mathcal{P}_{\ell k}(g^{(t)}, \mathbf{x})$, which is calculated by (5.18) with $c(\widehat{\varphi}_\ell(\cdot|\mathbf{x}), \varphi_{k'}^{(t)}(\cdot|\mathbf{x}))$ is now replaced by $\mathrm{KL}(\widehat{\varphi}_\ell(\cdot|\mathbf{x}) \| \varphi_{k'}^{(t)}(\cdot|\mathbf{x}))$. Then, (5.19) can be simply rewritten by

$$(\boldsymbol{\beta}_k^{(t+1)}, \sigma_k^{2(t+1)}) = \inf_{(\boldsymbol{\beta}, \sigma^2) \in \mathbb{R}^d \times \mathbb{R}_+} \mathcal{K}(\boldsymbol{\beta}, \sigma^2).$$

The following proposition gives a solution to this problem, *i.e.*, the updating formula for the parameters of the Gaussian regression experts.

Proposition 5.4.2. *Let the notations be as defined above. If $c(\cdot,\cdot)$ is chosen to be $\mathrm{KL}(\cdot\|\cdot)$, and $\varphi_k^{(t+1)}(\cdot|\mathbf{x})$ in (5.19) are Gaussian regression experts, then the update formulas for their parameters are given by*

$$\boldsymbol{\beta}_k^{(t+1)} = (\boldsymbol{X}_S^\top \boldsymbol{D}_k^{(t)} \boldsymbol{X}_S)^{-1} \boldsymbol{X}_S^\top \boldsymbol{X}_S \odot (\boldsymbol{W}_k^{(t)\top} \widehat{\boldsymbol{B}}^\top) \mathbb{1}_p, \tag{5.23a}$$

$$\sigma_k^{2(t+1)} = \frac{1}{\mathrm{trace}(\boldsymbol{D}_k^{(t)})} \left(\widehat{\boldsymbol{\Sigma}}^\top \boldsymbol{W}_k^{(t)} \mathbb{1}_S + \mathbb{1}_{MK}^\top \boldsymbol{W}_k^{(t)} \odot (\boldsymbol{\beta}^{(t+1)} - \widehat{\boldsymbol{B}})^\top \boldsymbol{X}_S^\top \boldsymbol{X}_S (\boldsymbol{\beta}^{(t+1)} - \widehat{\boldsymbol{B}}) \mathbb{1}_{MK} \right), \tag{5.23b}$$

in which, the matrices $\boldsymbol{D}_k^{(t)}$, $\boldsymbol{W}_k^{(t)}$ and $\widehat{\boldsymbol{B}}$ are given in (C.6)-(C.8).

The detailed calculations are given in Appendix C.1.5.

5.4.4 Updating the logistic regression experts parameter

When the problem under consideration is classification, it is common to use logistic regression model to model the component densities. For simplicity, we consider here the case of binary responses, *i.e.*, $y \in \{0, 1\}$. The conditional density in this case takes the following form

$$\varphi(y|\mathbf{x};\boldsymbol{\beta}) = \left[\frac{\exp(\mathbf{x}^\top\boldsymbol{\beta})}{1+\exp(\mathbf{x}^\top\boldsymbol{\beta})}\right]^y \left[\frac{1}{1+\exp(\mathbf{x}^\top\boldsymbol{\beta})}\right]^{1-y},$$

where $\boldsymbol{\beta} \in \mathbb{R}^d$ is the parameter vector. Let $\varphi_1(y|\mathbf{x};\boldsymbol{\beta}_1)$ and $\varphi_1(y|\mathbf{x};\boldsymbol{\beta}_2)$ be the conditional densities of two binary logistic regression experts, then the KL-divergence between them is given by

$$\begin{aligned}\mathrm{KL}\left(\varphi_1(\cdot|\mathbf{x})\|\varphi_2(\cdot|\mathbf{x})\right) = &\frac{\exp(\mathbf{x}^\top\boldsymbol{\beta}_1)}{1+\exp(\mathbf{x}^\top\boldsymbol{\beta}_1)}\left(\log\frac{\exp(\mathbf{x}^\top\boldsymbol{\beta}_1)}{1+\exp(\mathbf{x}^\top\boldsymbol{\beta}_1)} - \log\frac{\exp(\mathbf{x}^\top\boldsymbol{\beta}_2)}{1+\exp(\mathbf{x}^\top\boldsymbol{\beta}_2)}\right) \\ &+ \frac{1}{1+\exp(\mathbf{x}^\top\boldsymbol{\beta}_1)}\left(\log\frac{1}{1+\exp(\mathbf{x}^\top\boldsymbol{\beta}_1)} - \log\frac{1}{1+\exp(\mathbf{x}^\top\boldsymbol{\beta}_2)}\right). \end{aligned} \tag{5.24}$$

Analogously to the case of Gaussian regression experts, substituting the expression of KL-divergence between two logistic experts into (5.19), the objective function can be rewritten as a function of $\boldsymbol{\beta}$ by

$$\mathcal{K}_{Bi}(\boldsymbol{\beta}) := \mathbb{E}\left[\sum_{\ell=1}^{MK} \mathcal{P}_{\ell k}^{(t)}(\mathbf{x}) \left(\log\left(1+\exp(\mathbf{x}^\top\boldsymbol{\beta})\right) - \mathbf{x}^\top\boldsymbol{\beta}\frac{\exp(\mathbf{x}^\top\widehat{\boldsymbol{\beta}}_\ell)}{1+\exp(\mathbf{x}^\top\widehat{\boldsymbol{\beta}}_\ell)} + \mathrm{const}\right)\right], \tag{5.25}$$

where $\mathcal{P}_{\ell k}^{(t)}(\mathbf{x})$, similarly to the case of Gaussian experts, is calculated by (5.18) with $c(\widehat{\varphi}_\ell(\cdot|\mathbf{x}), \varphi_{k'}^{(t)}(\cdot|\mathbf{x}))$ is now replaced by $\mathrm{KL}(\widehat{\varphi}_\ell(\cdot|\mathbf{x})\|\varphi_{k'}^{(t)}(\cdot|\mathbf{x}))$ given in (5.24).

Proposition 5.4.3. *Let the notations be as defined above. If $c(\cdot,\cdot)$ is chosen to be $\mathrm{KL}(\cdot\|\cdot)$, and $\varphi_k^{(t+1)}(\cdot|\mathbf{x})$ in (5.19) are logistic regression experts, then the update formulas for their parameters are given by*

$$\boldsymbol{\beta}_k^{(t+1)} = (\boldsymbol{X}_S^\top \boldsymbol{X}_S)^{-1}\boldsymbol{X}_S^\top(\log(\boldsymbol{V}_k^{(t)}) - \log(1-\boldsymbol{V}_k^{(t)})), \quad \boldsymbol{V}_k = \boldsymbol{D}_k^{(t)^{-1}}\boldsymbol{W}_k^{(t)\top}\boldsymbol{U}^\top, \tag{5.26a}$$

in which, the matrices $\boldsymbol{D}_k^{(t)}$ and $\boldsymbol{W}_k^{(t)}$ the same with those in Proposition 5.4.2, and $\boldsymbol{U}$ is given in (C.11).

The detailed calculations can be found in Appendix C.1.6.

5.5 Experimental study

5.5.1 Models in comparison

In this section, we illustrate the effectiveness of our proposed approach on simulated and real-world datasets of various sizes. In particular, we compare the performances of our reduction estimator $\bar{\boldsymbol{\theta}}^R$ and the following estimators:

- Global estimator, denoted by $\boldsymbol{\theta}^G$. It is the MLE of the ME model estimated based on the full dataset on the central machine.

- Middle estimator, denoted by $\bar{\boldsymbol{\theta}}^{Mid}$. It is the parameter of the local density (among M local densities) that gives the smallest sum of transportation divergences, weighted by λ_m, with the other local densities, *i.e.*,

$$\bar{f}^{Mid} = f(y|\mathbf{x};\bar{\boldsymbol{\theta}}^{Mid}) = \inf_{g\in\{\widehat{f}_1,\ldots,\widehat{f}_M\}} \sum_{m=1}^{M} \lambda_m \mathcal{T}_c(\widehat{f}_m, g).$$

- Weighted average estimator, denoted by $\bar{\boldsymbol{\theta}}^W$. It is an ad-hoc estimator defined as the pointwise weighted average of the local estimators, *i.e.*,

$$\bar{\boldsymbol{\theta}}^W = \sum_{m=1}^{M} \lambda_m \widehat{\boldsymbol{\theta}}_m.$$

  One can see this estimator is sensitive with label switching and non-i.i.d. data partitions, *i.e.*, when there are differences between the partitions in terms of sample sizes of in terms of the equilibrium between clusters. However, when the partitions are i.i.d., have equal sizes,

and the estimated parameters are well-ordered, *i.e.*, satisfy the identification condition for ME model (Jiang and Tanner, 1999), $\bar{\boldsymbol{\theta}}^W$ is a potential candidate.

5.5.2 Evaluation metrics

We compare the estimators in terms of the following metrics

- Transportation distance, defined in (5.10) with c being KL divergence, between the true mixture and the estimated mixture. This metric is used for simulated data where we know the true mixture.
- Log likelihood value of the estimated parameter evaluated on the testing set.
- Mean squared error (MSE) between the true parameter and the estimated parameter, used for simulated data.
- Relative prediction error (RPE) on testing set, defined by $\text{RPE} = \sum_{i=1}^{n}(y_i - \widehat{y}_i)^2)/\sum_{i=1}^{n} y_i^2$, where y_i and $\widehat{y}_i$ are, respectively, the true and the predicted responses of the ith observation in the testing set.
- Adjusted Rand Index (ARI) between the true clustering in the testing set and the clustering predicted by the estimated mixture. ARI is used for simulated data where we know prior labels of the clusters.
- Learning time, *i.e.*, the time consumed to obtain the estimator.

5.5.3 Simulated data

Data generating process

In this simulation, we fixed $K = 4$ and $d = 20$. We considered datasets of different sizes, from moderate to very large, to illustrate the statistical performance of the estimators. The datasets were generated as follows. Firstly we specified a *true mixture* of K components, which to be estimated, and the centers $\mathbf{x}^{(k)}$ of the clusters $k \in [K]$. Then for each $N \in \{100000, 500000, 1000000\}$, we generated 100 random dataset $\mathscr{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ by drawing, for each k, N/K points from a multivariate Gaussian distribution with mean $\mathbf{x}^{(k)}$ and covariance matrix defined by $\Sigma_{uv} = \frac{1}{4}^{|u-v|}$, for $u, v \in [d]$. Finally, the responses y_i are generated by the standard generating process for ME model, *i.e.*,

$$
\begin{aligned}
y_i | Z_i = k, \mathbf{x}_i \;&\sim\; \begin{cases} \mathcal{N}\Big(\mathbf{x}_i^\top \boldsymbol{\beta}_k^*;\; {\sigma^*}_k^2\Big) & \text{if regression problem,} \\ \mathcal{M}\Big(1, \big(\pi_1(\mathbf{x}_i; \boldsymbol{\beta}_k^*), \ldots, \pi_G(\mathbf{x}_i; \boldsymbol{\beta}_k^*)\big)\Big) & \text{if classification problem} \end{cases} \\
Z_i | \mathbf{x}_i \;&\sim\; \mathcal{M}\Big(1, \big(\pi_1(\mathbf{x}_i; \boldsymbol{\alpha}^*), \ldots, \pi_K(\mathbf{x}_i; \boldsymbol{\alpha}^*)\big)\Big),
\end{aligned}
$$

where $\mathcal{M}(1, \boldsymbol{\pi})$ is the multinomial distribution with parameter vector $\boldsymbol{\pi}$. Here, the true parameter vector of the model to be estimated is $\boldsymbol{\theta}^* := ({\boldsymbol{\alpha}^*}^\top, {\boldsymbol{\beta}_1^*}^\top, \ldots, {\boldsymbol{\beta}_K^*}^\top, {\sigma_1^*}^2, \ldots, {\sigma_K^*}^2)^\top$, if regression

problem, or $\boldsymbol{\theta}^* := (\boldsymbol{\alpha}^{*\top}, \boldsymbol{\beta}_1^{*\top}, \ldots, \boldsymbol{\beta}_K^{*\top})^\top$, if classification problem. In this simulation study, we opted to consider the regression problem. To have more fairly data generating process, these true parameter vectors and the cluster centers are also selected randomly from the set of integers in the interval $[-5, 5]$. For each of the N's considered above, we also generated 100 corresponding testing sets such that the ratio of training-testing split is $80\% - 20\%$, more precisely, the testing sets will have 50k, 125k and 500k observations, respectively. It is worth mentioning that each of the datasets with one million observations consumes roundly 1.6 GB of memory. The estimators are computed using training sets and the metrics (except the learning time) are evaluated on testing sets.

Implementation details and results

For the distributed approaches, *i.e.*, the reduction, middle and weighted average estimators, we consider four settings of M, respectively, 4, 16, 64 and 128 machines. The data are distributed equally to the local machines, and the size of the supporting set is taken to be equal to the size of local data, *i.e.*, $S = N_m = N/M$ for all $m \in [M]$.

The Global estimator is obtained by running the EM algorithm, on a single computer. The learning time for each of them is therefore the total time of running the corresponding algorithm. On the other hand, for the reduction estimator, since the local inferences can be performed parallelly on multiple machines, the learning time is recorded as the sum of the maximum local time and the aggregating time, *i.e.*, the time consumed by the Algorithm 5.1. Similarly, the learning time of Middle and Weighted average estimator is the sum of the maximum local time and the time of the corresponding calculations to obtain them. All EM algorithms involving the global and local inferences are run with five random initializations. Finally, for each of the obtained estimators, we compute the transportation distance form the true model, the log-likelihood, MSE, RPE and ARI on the corresponding testing sets. model evaluation metrics described in Subsection 5.5.2.

Figure 5.3 shows the box plots of 100 Monte Carlo runs of the evaluation metrics when $N = 100000$. For each panel, *e.g.*, the transport distance, the x-axis has two rows. The lower one represents the number of machines used for training, while the upper one represents the leaning approaches in comparison. As we can see, the prediction and clustering performance of our reduction estimator is as good as that of the global estimator when M is 4 or 16 machines. When M is 64 or 128 machines, the errors of the reduction estimator are slightly higher than the global estimator, but still better than the middle and weighted average estimators. In terms of transportation distance, we can see that with the reduction estimator, the obtained ME model is as close as the ME model formed by the global estimator, even when $M = 128$ machines. Finally, in terms of learning time, the distributed approach requires much less time than the global approach, for example 25 times on average when 4 machines are used.

Figure 5.4 and Figure 5.5 show the results when N is 500k and one million, respectively. Here, the evaluation metrics behave similarly to those in Figure 5.3, except in the MSE panels where the reduction estimator appears to be much worse than the middle and weighted average estimators when $M = 128$. However, given the magnitudes of the MSE in these cases, the difference is actually not too large. This behavior can also be explained by the fact that for N of 500k and one million,

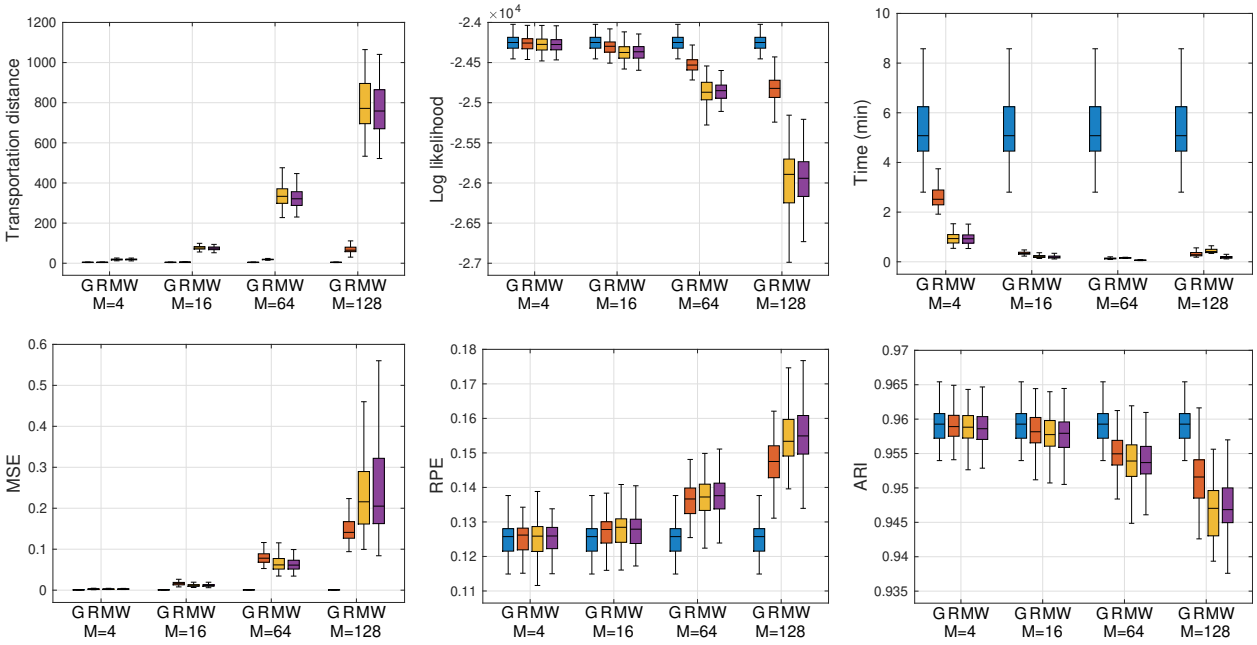


Figure 5.3: Performance comparison of the Global ME (G), Reduction (R), Middle (M) and Weighted average (W) estimator on datasets of size N=100000 with different number of machines.

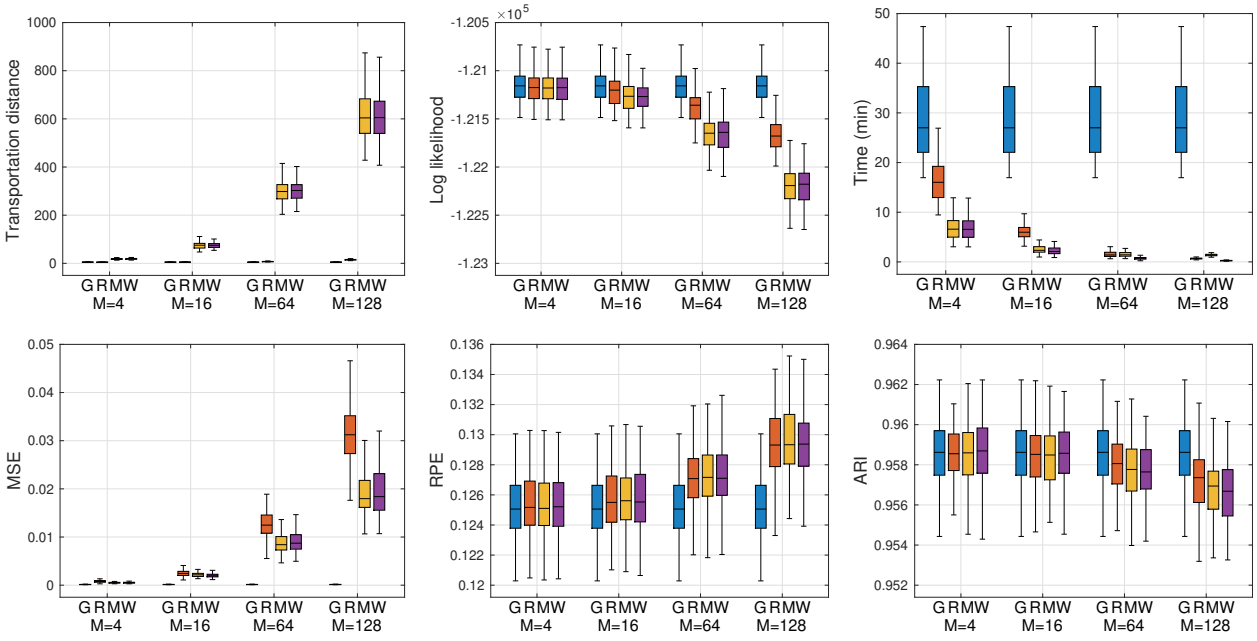


Figure 5.4: Same interpretation as of Figure 5.3 but with $N = 500000$.

even with $M = 128$, the local sizes N_m are approximately 3900 and 7800 observations, respectively, which are still large sample sizes for estimating the model parameters.

In Figure 5.6, we compare the evaluation metrics of the estimators across sample sizes when 128 machines were used. With this large number of machines, the learning time is significantly reduced

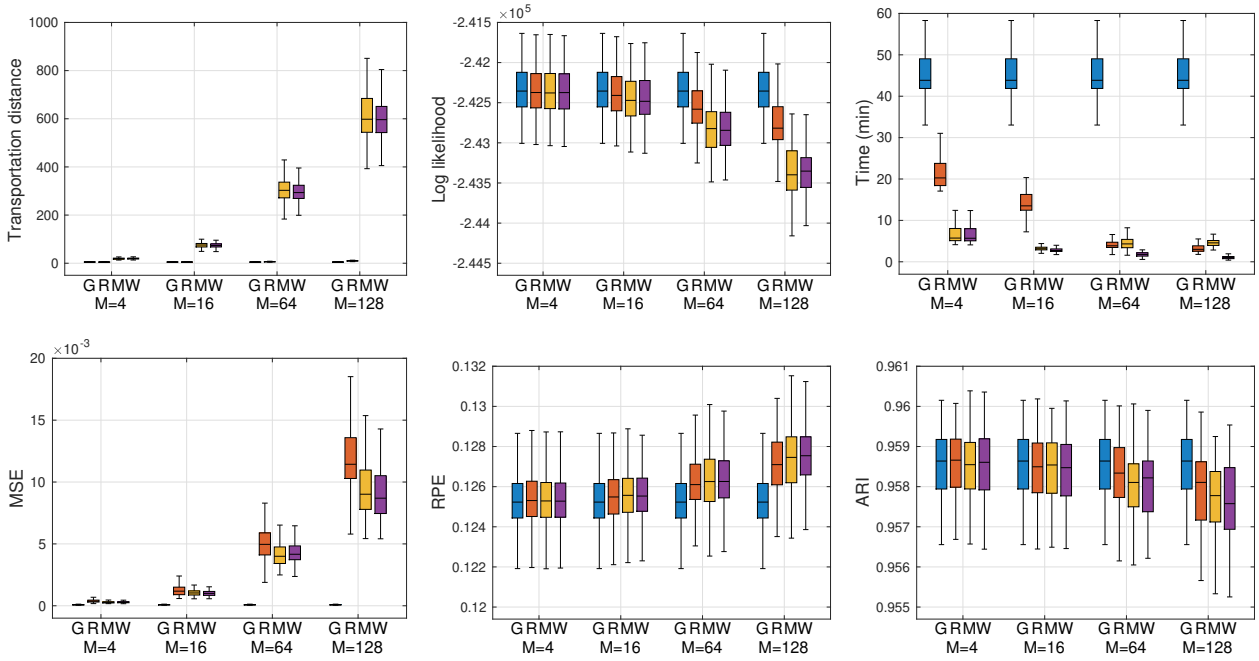


Figure 5.5: Same interpretation as of Figure 5.3 but with $N = 1000000$.

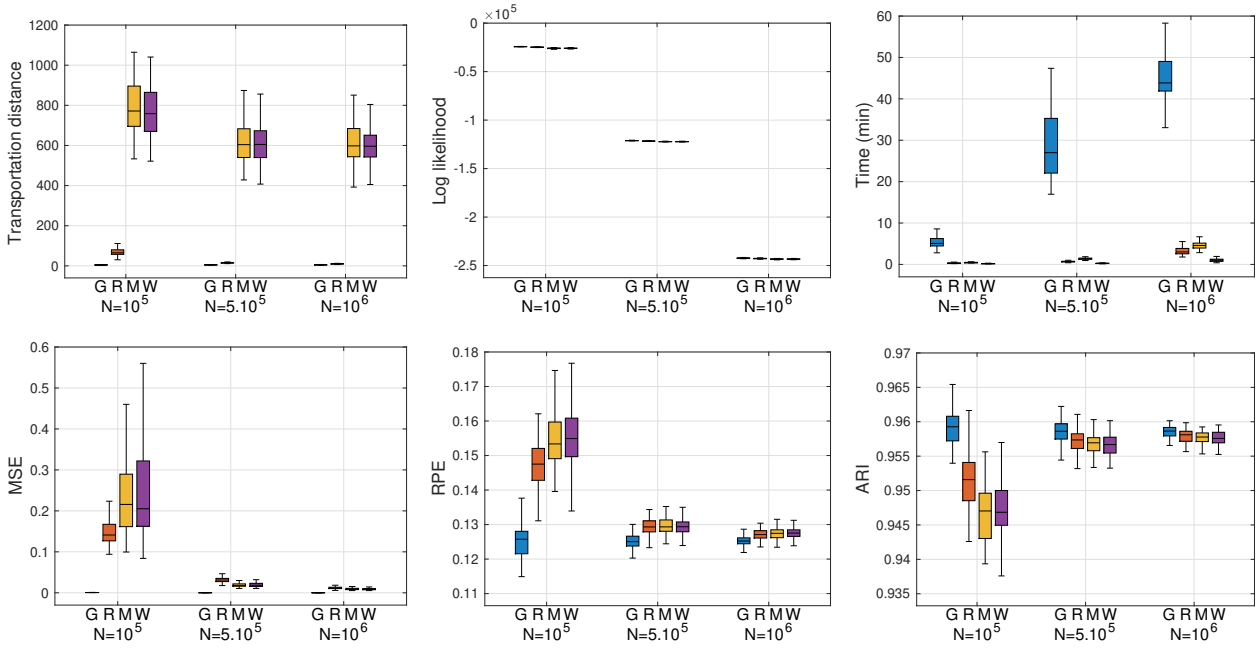


Figure 5.6: Performance comparison of the Global (G), versus Reduction (R), Middle (M) and Weighted average (W) estimator using 128 machines on different sizes of datasets.

when using the distributed approaches, especially for datasets with one million observations. The prediction and clustering performance metrics such as MSE, RPE and ARI behave as expected across estimators and across sample sizes.

5.5.4 Application to MMASH dataset

Data description and pre-processing

In this study, we investigate the prediction performance of all the learning approaches (*i.e.*, global, distributed, median and weighted averaged) to the Multilevel Monitoring of Activity and Sleep in Healthy people (MMASH) dataset. This dataset consists of 24 hours of continuous inter-beat interval data (IBI), triaxial accelerometer data, sleep quality, physical activity and psychological characteristics of 22 healthy young males. During the 24-hours data recording, participants wore two devices that continuously recorded heartbeats and beat-to-beat interval (via a heart rate monitor), and actigraphy information such as accelerometer data, sleep quality and physical activity (via an Actigraph device). The data was recorded on a second-by-second basis for each subject. More details about the experiment setup of MMASH are provided in Rossi et al. (2020), Oyeleye et al. (2022).

While MMASH is a rich dataset to assess many relations between physical, psychological and physiological characteristics, in this study we consider the problem of predicting the upcoming heart rate (HR) given the most recent records from them. In particular, we applied the data pre-processing procedure as in Oyeleye et al. (2022) to extract a *HR series dataset* that contains HR series of all participants in the MMASH dataset. Then, for each participant, his/her HR series is reframed into a set of predictor-response data pairs, *i.e.*, into supervised setting. More specifically, we will use the last HRs during 300 seconds, to predict the 5-minutes-ahead HR. Each predictor can be viewed as a sample of 300 discretized values $U(t_j)$, $j \in [300]$, of some HR function $X(t)$, and the 5-minutes-ahead HR (to be predicted) is its scalar response. Therefore, the FME model proposed in Chapter 3 will be the choice in this context.

The recording times vary among the participants, so the number of pairs $(X_i(t), y_i)$ extracted from their HR series are different from each other. However, the differences are not large, and the average is about 66000 pairs for each participant. In total, our dataset contains 1453000 data pairs $(U_i(t), y_i)$. The dataset is split into training and testing data with a ratio of 80%-20% (to ensure the balance, this split is also made w.r.t. each participant). The size of our training set is $N = 1162400$, and the size of testing set is 290600. Figure 5.7 shows some randomly taken predictors $U_i(t)$, as well as their smooth versions using a 50-dimensional B-spline basis.

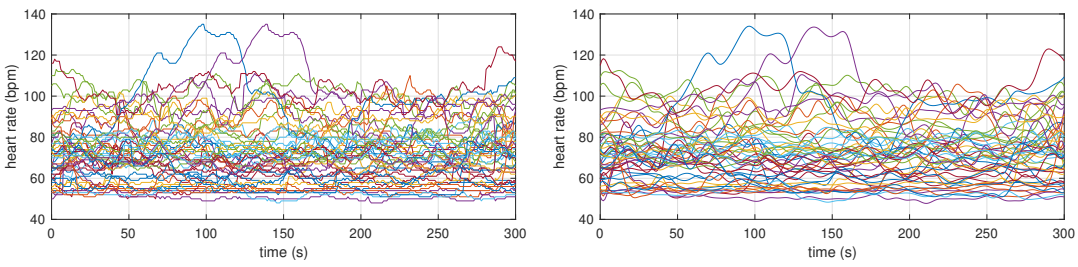


Figure 5.7: (a) 50 randomly taken HR curves and (b) their smooth versions using a 50-dimensional B-spline basis.

Model evaluation, implementation details and results

Beside the RPE metric as used before in many regression problems throughout the thesis, we use the root mean square error (RMSE) value to evaluate the HR prediction since it is directly proportional to the unit of the predicted values. In addition, to compare with the related work in Oyeleye et al. (2022), the scatter index (SI), which is simply the RMSE divided by the average value of the observed value, was also computed. It is commonly understood that SI $< 10\%$ is a good model, and SI $< 5\%$ is a very good model.

Now, we describe how we integrate our distributed leaning approach into the FME model proposed in Chapter 3, it is quite simple as follows. First, the functional predictors, the functional expert and the gating networks are represented using some appropriate basses (here we used B-spline). Then the conditional density function of the model can be written in a vectorial fashion as in (3.13). Note that, if we use a same basis for the functional expert and gating networks (which is our case here), the design vectors $\mathbf{r}_i$ and $\mathbf{x}_i$ are identical. Therefore, the model density (3.13) is now taking the same formula as the density (5.1), which is our starting point for developing the distributed learning approach. This means we can employ the Algorithm 5.1 to estimate the parameter vector $\boldsymbol{\Psi}$ in (3.13) of the FME model in a distributed manner.

The RPE, RMSE, SI and the learning time of each approach are given in Table 5.2. We can see, the distributed approach significantly reduced the learning time while other prediction metrics are not too different. The estimated functional expert and gating functions are shown in Figure 5.8.

	RPE	RMSE	SI	Learning time (min)
Global estimator	1.03%	8.58	9.01%	57.32
Reduction estimator, $M = 64$	1.45%	9.46	9.54%	7.84
Reduction estimator, $M = 128$	1.45%	9.49	9.58%	6.32

Table 5.2: Comparison of Global estimator and Reduction estimator on MMASH data.

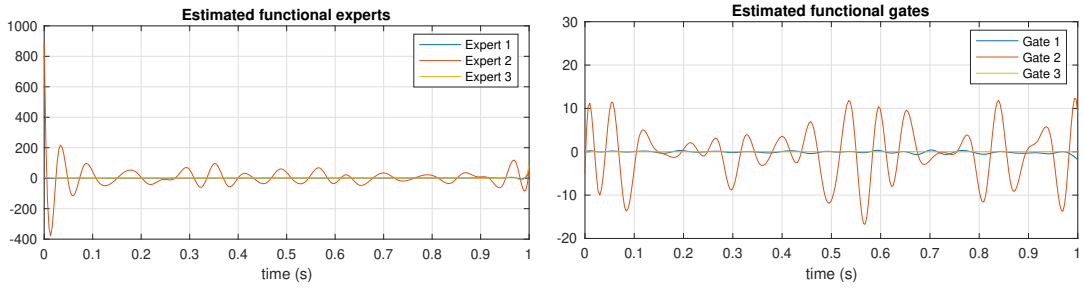


Figure 5.8: Estimated functional expert and gating networks of the FME model applied on MMASH data.

5.6 Summary

In this chapter, a distributed learning approach for ME models was proposed. In particular, the learning approach is based on minimizing the expected transportation divergence between two ME models: one is the large ME model that consists of all local mixtures obtained from the local machines, and one is the targeted K-component ME model. An algorithm based on MM principle was also proposed to solve the resulting minimization problem.

The proposed distributed learning approach was successfully applied to simulated and real-world datasets. This distributed approach was shown to have a leaning time much less than the global approach, but still give results as good as those based on training with full dataset. Specially, we applied the proposed distributed learning approach to the FME model for functional data proposed in Chapter 3 with promising remarks: the integration is easy and the prediction performance is as good as the global approach, while the learning time has been much improved.

Chapter 6

Conclusion and future directions

Contents

6.1 Conclusion

This thesis was written regarding to the ME models for functional data and a distributed learning approach for ME models. The goal of developing efficient ME models for functional data was achieved through the introduction of a new class of ME models, referred to by functional mixtures-of-experts (FME), whose members including

i. FME for regression, *i.e.*, scalar-on-function FME,

ii. FME for classification, also a scalar-on-function FME,

iii. FF-FME for regression, *i.e.*, function-on-function FME,

iv. FS-FME for regression, *i.e.*, function-on-scalar FME.

In addition, for scalar-on-function FME models, we have developed their *interpretable* variants: iFME for regression and iFME for classification, which were able to produce highly-interpretable fits for the functional expert and gating networks. The goal of developing a distributed learning approach for ME models was also achieved through the proposed reduction strategy and an efficient developed MM algorithm whose performance was demonstrated via application to simulated and real-world data.

More specifically, in Chapter 3, we proposed a ME for regression problems that takes functional data as inputs and continuous scalars as outputs. We refer to it by FME. Model estimation was performed using MLE and regularized MLE, which results in the FME-Lasso model that imposes a certain degree of sparsity on the estimated parameters. Furthermore, to obtain truly sparse fits for the functional expert and gating networks, we proposed an interpretable version of the FME

model, referred to by iFME model, to produce fits that are not only sparse itself, but also sparse with respect to their derivatives. The proposed FME, FME-Lasso, and iFME models for regression were applied to simulated and real-world datasets. The implementation of the algorithms, model selection, as well as the interpretation of the results were also discussed in detail. Matlab codes for the proposed FME model (as well as FME-Lasso and iFME models) will be available at the following link: https://github.com/fchamroukhi/FunME.

In Chapter 4, we extended the FME, FME-Lasso, and iFME models to the classification problems. In this case, we modeled the experts by the functional multinomial regression models. The resulting models were also outperform the current state of the art methods as the case of regression in the previous chapter. The corresponding EM algorithms were also developed for model estimations in the classification case. Furthermore, the FME model was extended to the case of functional responses via the introduction of two new models: FF-FME and FS-FME. Specially, the proposed models were estimated using penalized MLE that penalizes the roughness (via appropriate derivative operators) and results in smooth fits for the functional parameters. The corresponding EM-like algorithms were also developed for model estimations. Numerical experiments were shown to illustrate the application of the models.

Finally, in Chapter 5, a distributed learning approach was proposed for ME models to deal with the situation where the data is so large that it is difficult to estimate the model parameter on a single computer, within a reasonable amount of time. The approach is based on the divide-and-conquer principle. In particular, the local estimators (of ME models on local machines) are aggregated by minimizing an expected transportation divergence between two ME models. The MM algorithm was also developed particularly for Gaussian and logistic regression experts. The experiments on simulated and real data verified the efficiency of our approach.

6.2 Future directions

Given the promising results of our proposals and the motivation from a methodological point of view, there are many further interesting extensions that can be considered in future work. First, with the FME models for regression and classification, it is natural to extend them to the multiple case, *i.e.*, for each expert/gate k there are more than one functional parameter. This can also be applied to extend the iFME models.

Second, an interpretable approach to the FF-FME and FS-FME models can be explored. In particular, as we can see in Chapter 4, the models were able to generate smooth fits for the functional parameters via roughness penalties with second-order (partial) derivatives, they do not ensure highly sparse fits as, for example, in Figure 3.7. Therefore, according to our idea, a representation upon the coefficients of the derivatives of the functional parameter can be established for the model. In this way, constraining sparsity on the resulting parameter vectors can lead to truly sparse fits

for the experts and gating networks.

Regarding the distributed learning approach for ME models, some interesting considerations can also be made. For example, in high-dimensional setting, the local parameter vectors are oftenly estimated with regularization, so the update formula such as the one in (5.23) must be adapted to preserve (at some level) the sparsity in the local estimators. This can be approached using a majority voting method as in Chen and ge Xie (2014). Moreover, the proposed distributed learning approach can be applied to the iFME, FF-FME, and FS-FME models.

Although numerous extensive experiments were constructed throughout the thesis to demonstrate the performance of the proposed models, real-world data experiment for the FS-FME model unfortunately was not included. However, we would like to mention that, given the performance on the simulated data, we strongly believe that FS-FME model can also produce good results when applied to real-world data. Numerical experiment for the distributed learning approach in classification problems was also not provided, it therefore is one of our future works. Finally, although the proposed models and the proposed learning approach work quite well, the theoretical results, however, need further investigation.

Appendix A

A.1 Calculations for the State of the art chapter

A.1.1 Penalized LS estimation of function-on-scalar regression

This appendix presents the detailed calculations to obtain the penalized least squares estimator (2.27). The problem under consideration now is

$$\underset{\boldsymbol{B}\in\mathbb{R}^{p\times K}}{\text{minimize}} \left\{ \int_{\mathcal{T}} \|\boldsymbol{R}\,\boldsymbol{\omega}(t) - \boldsymbol{X}\boldsymbol{B}\,\boldsymbol{\omega}(t)\|^2 dt + \sum_{j=1}^{p} \lambda_j \boldsymbol{b}_j^\top \boldsymbol{D}\boldsymbol{b}_j \right\}, \tag{A.1}$$

where $\boldsymbol{B} = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_p]^\top \in \mathbb{R}^{p\times K}$ is the coefficient matrix to be estimated; $\boldsymbol{R} \in \mathbb{R}^{n\times K}$ and $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ are the response and design matrix, $\boldsymbol{\omega}(t) = [\omega_1(t), \ldots, \omega_K(t)]^\top$, and finally $\boldsymbol{D} \in \mathbb{R}^{K\times K}$ is given in (2.25).

The first term of the objective function in (A.1) can be reformulated as

$$\begin{aligned}
\int_{\mathcal{T}} \|\boldsymbol{R}\,\boldsymbol{\omega}(t) - \boldsymbol{X}\boldsymbol{B}\,\boldsymbol{\omega}(t)\|^2 dt &= \int_{\mathcal{T}} \sum_{i=1}^{n} (\boldsymbol{r}_i^\top \boldsymbol{\omega}(t) - \boldsymbol{x}_i^\top \boldsymbol{B}\boldsymbol{\omega}(t))^2 dt \\
&= \sum_{i=1}^{n} \int_{\mathcal{T}} (\boldsymbol{r}_i^\top \boldsymbol{\omega}(t) - \boldsymbol{x}_i^\top \boldsymbol{B}\boldsymbol{\omega}(t))^2 dt \\
&= \sum_{i=1}^{n} \int_{\mathcal{T}} (\boldsymbol{r}_i^\top - \boldsymbol{x}_i^\top \boldsymbol{B})\boldsymbol{\omega}(t)\,[\boldsymbol{\omega}(t)]^\top (\boldsymbol{r}_i^\top - \boldsymbol{x}_i^\top \boldsymbol{B})^\top dt \\
&= \sum_{i=1}^{n} (\boldsymbol{r}_i^\top - \boldsymbol{x}_i^\top \boldsymbol{B}) \left[\int_{\mathcal{T}} \boldsymbol{\omega}(t)\,[\boldsymbol{\omega}(t)]^\top dt \right] (\boldsymbol{r}_i^\top - \boldsymbol{x}_i^\top \boldsymbol{B})^\top.
\end{aligned} \tag{A.2}$$

Let

$$\boldsymbol{I}_\omega := \left[\int_{\mathcal{T}} \omega_k(t)\omega_\ell(t) dt \right]_{1\leqslant k,\ell\leqslant K} \in \mathbb{R}^{K\times K},$$

then $\boldsymbol{I}_\omega$ is a non-negative define matrix, hence there exits $\boldsymbol{I}_\omega^{1/2} \in \mathbb{R}^{K\times K}$ such that $\boldsymbol{I}_\omega = \boldsymbol{I}_\omega^{1/2}\boldsymbol{I}_\omega^{1/2}$.

The equation (A.2) can thus be continued as

$$\begin{aligned}\int_{\mathcal{T}} \|\boldsymbol{R}\,\boldsymbol{\omega}(t) - \boldsymbol{X}\boldsymbol{B}\,\boldsymbol{\omega}(t)\|^2 dt &= \sum_{i=1}^{n} (\boldsymbol{r}_i^\top - \boldsymbol{x}_i^\top \boldsymbol{B}) \boldsymbol{I}_\omega^{1/2} \boldsymbol{I}_\omega^{1/2} (\boldsymbol{r}_i^\top - \boldsymbol{x}_i^\top \boldsymbol{B})^\top \\ &= \|\boldsymbol{R}\boldsymbol{I}_\omega^{1/2} - \boldsymbol{X}\boldsymbol{B}\boldsymbol{I}_\omega^{1/2}\|^2 \qquad \text{(A.3)} \\ &= \|\operatorname{vec}(\boldsymbol{I}_\omega^{1/2}\boldsymbol{R}^\top) - (\boldsymbol{X} \otimes \boldsymbol{I}_\omega^{1/2}) \operatorname{vec}(\boldsymbol{B}^\top)\|^2, \qquad \text{(A.4)}\end{aligned}$$

in which the last equality can be verified by a simple vectorization step that uses the definition of the matrices.

The second term of the objective function in (A.1) can be written in terms of $\operatorname{vec}(\boldsymbol{B}^\top)$ as

$$\sum_{j=1}^{p} \lambda_j \boldsymbol{b}_j^\top \boldsymbol{D}\boldsymbol{b}_j = [\operatorname{vec}(\boldsymbol{B}^\top)]^\top \boldsymbol{\Lambda} \otimes \boldsymbol{D} \operatorname{vec}(\boldsymbol{B}^\top), \qquad \text{(A.5)}$$

where $\boldsymbol{\Lambda} = \operatorname{diag}(\lambda_1, \ldots, \lambda_p)$. Gathering (A.4) and (A.5), the problem (A.1) is now reformulated to

$$\underset{\operatorname{vec}(\boldsymbol{B}^\top) \in \mathbb{R}^{pK}}{\text{minimize}} \left\{ \|\operatorname{vec}(\boldsymbol{I}_\omega^{1/2}\boldsymbol{R}^\top) - (\boldsymbol{X} \otimes \boldsymbol{I}_\omega^{1/2}) \operatorname{vec}(\boldsymbol{B}^\top)\|^2 + [\operatorname{vec}(\boldsymbol{B}^\top)]^\top \boldsymbol{\Lambda} \otimes \boldsymbol{D} \operatorname{vec}(\boldsymbol{B}^\top) \right\}, \qquad \text{(A.6)}$$

which takes the same form as the usual ridge regression. Therefore, by calculating the first order derivative with respect to $\operatorname{vec}(\boldsymbol{B}^\top)$ and setting it to zero, the solution to (A.6) is established as given in (2.27).

A.1.2 Penalized LS estimation of function-on-function regression

This appendix presents the detailed calculations to obtain the penalized least squares estimator (2.37) for the function-on-function regression problem. First, by plugging (2.33) into (2.32) and using the vectorization representation of the penalty term, the problem under consideration is now rewritten as

$$\underset{\boldsymbol{b} \in \mathbb{R}^K}{\text{minimize}} \left\{ \int_{\mathcal{T}} \|\boldsymbol{Y}(t) - \boldsymbol{X}^\star(t)\,\boldsymbol{b}\|^2 dt + \lambda_t \boldsymbol{b}^\top \boldsymbol{D}_t \boldsymbol{b} + \lambda_u \boldsymbol{b}^\top \boldsymbol{D}_u \boldsymbol{b} \right\}, \qquad \text{(A.7)}$$

where $\boldsymbol{Y}(t)$ and $\boldsymbol{X}^\star(t)$ are defined as in (2.31).

From the relations (2.31) and (2.34), the first term of the objective function in (A.7) can be expressed as

$$\begin{aligned}\int_{\mathcal{T}} \|\boldsymbol{Y}(t) - \boldsymbol{X}^\star(t)\,\boldsymbol{b}\|^2 dt &= \int_{\mathcal{T}} \|\boldsymbol{R}\,\boldsymbol{\omega}(t) - \boldsymbol{G}\boldsymbol{b}\,\boldsymbol{\omega}(t)\|^2 dt \\ &= \|\boldsymbol{R}\boldsymbol{I}_\omega^{1/2} - \boldsymbol{G}\boldsymbol{b}\boldsymbol{I}_\omega^{1/2}\|^2 \qquad \text{(A.8)}\end{aligned}$$

with the second equality was obtained using the same manner as in (A.2) and (A.3).

Let $\boldsymbol{V}$ be defined by

$$\boldsymbol{V} = \begin{bmatrix} \boldsymbol{I}_\psi^{1/2}\boldsymbol{x}_{11}^\star & \boldsymbol{I}_\psi^{1/2}\boldsymbol{x}_{12}^\star & \cdots & \boldsymbol{I}_\psi^{1/2}\boldsymbol{x}_{1K}^\star \\ \boldsymbol{I}_\psi^{1/2}\boldsymbol{x}_{21}^\star & \boldsymbol{I}_\psi^{1/2}\boldsymbol{x}_{22}^\star & \cdots & \boldsymbol{I}_\psi^{1/2}\boldsymbol{x}_{2K}^\star \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{I}_\psi^{1/2}\boldsymbol{x}_{n1}^\star & \boldsymbol{I}_\psi^{1/2}\boldsymbol{x}_{n2}^\star & \cdots & \boldsymbol{I}_\psi^{1/2}\boldsymbol{x}_{nK}^\star \end{bmatrix} \in \mathbb{R}^{nL\times K},$$

where $\boldsymbol{I}_\psi^{1/2}$ is the square root of the non-negative definite matrix $\boldsymbol{I}_\psi$ defined by

$$\boldsymbol{I}_\psi = \left[\int_{\mathcal{T}} \psi_k(t)\psi_\ell(t)dt\right]_{1\leqslant k,\ell\leqslant L} \in \mathbb{R}^{L\times L}.$$

Then (A.8) can be reformulated by

$$\|\boldsymbol{R}\boldsymbol{I}_\omega^{1/2} - \boldsymbol{G}\boldsymbol{b}\boldsymbol{I}_\omega^{1/2}\|^2 = \|\operatorname{vec}(\boldsymbol{I}_\omega^{1/2}\boldsymbol{R}^\top) - \boldsymbol{V}\boldsymbol{b}\|^2, \tag{A.9}$$

which can be verified simply by the definitions of the matrices.

Thus, the problem (A.7) can thus be rewritten by

$$\underset{\boldsymbol{b}\in\mathbb{R}^K}{\text{minimize}} \left\{\|\operatorname{vec}(\boldsymbol{I}_\omega^{1/2}\boldsymbol{R}^\top) - \boldsymbol{V}\boldsymbol{b}\|^2 + \lambda_t\boldsymbol{b}^\top\boldsymbol{D}_t\boldsymbol{b} + \lambda_u\boldsymbol{b}^\top\boldsymbol{D}_u\boldsymbol{b}\right\}. \tag{A.10}$$

By taking the first derivative of the objective function and setting it to zero, the solution is obtained as given in (2.37).

A.2 Descriptions of data

Tecator data

Tecator is a well-known data in FDA. This dataset consists of 215 spectrometric curves and the corresponding Water, Fat and Protein content of meat samples. Each spectrometric curve corresponds to the absorbance measured at 100 wavelengths (100 discretization points from 850mm to 1050mm). The curves are split according to Ferraty and Vieu (2006) into two classes: with small (less than 20) and large fat content obtained by an analytical chemical processing.

Berkeley growth data

Berkeley growth data was originally published in Tuddenham and Snyder (1954) and was analyzed by many studies of functional data since then. The data consists of the height records for 39 boys and 54 girls. In original data, the measurements were taken quarterly from ages 1 to 2,

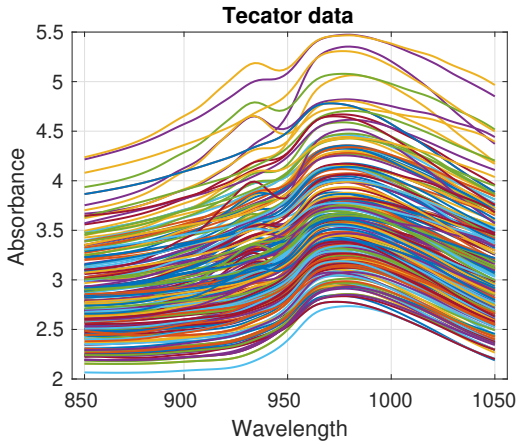


Figure A.1: Spectrometric curves in the Tecator data.

annually from 2 to 8, and semiannually from 8 till 18. Therefore, it is common to perform a simple interpolation such that the trajectories are available quarterly from age 1 to 18. Figure A.2 display the height trajectories of the two groups.

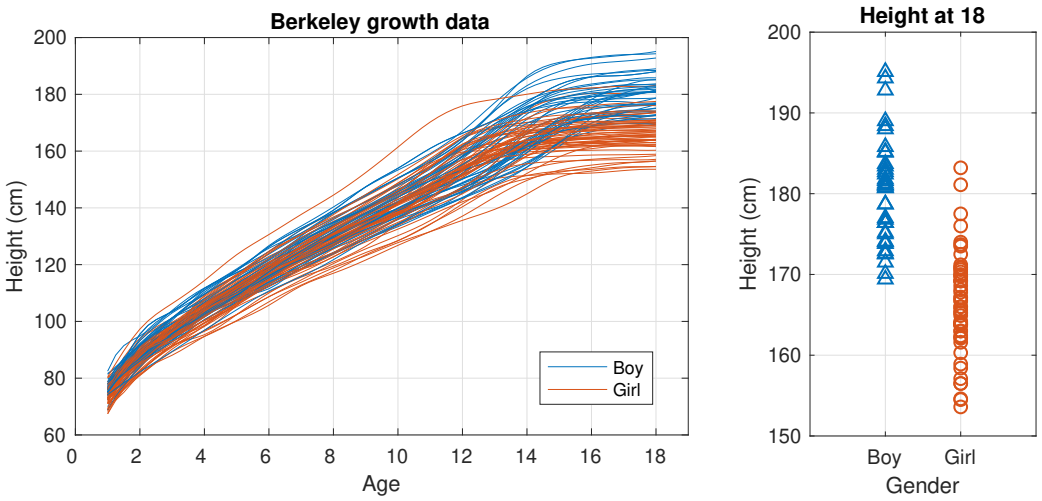


Figure A.2: Left panel: height trajectories from age 1 to 18 of 39 boys and 54 girls. Right panel: height at age 18 of the two groups.

Appendix B

B.1 EM for the FME model

The FME model can be fitted by iteratively maximizing the observed-data log-likelihood (3.14) iteratively via the EM algorithm. For FME, the EM takes the following form. The complete-data log-likelihood upon which the EM principle is constructed is defined by

$$\log L_c(\boldsymbol{\Psi}) = \sum_{i=1}^{n}\sum_{k=1}^{K} Z_{ik} \log \left[\pi_k(\mathbf{r}_i;\boldsymbol{\xi})\phi(y_i;\beta_{k,0}+\boldsymbol{\eta}_k^{\top}\mathbf{x}_i,\sigma_k^2)\right], \tag{B.1}$$

Z_{ik} being an indicator binary-valued variable such that $Z_{ik} = 1$ if $Z_i = k$ (*i.e.*, if the ith pair $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ is generated from the kth expert component) and $Z_{ik} = 0$, otherwise.

E-step This step computes at each EM iteration s the expectation of the complete-data log-likelihood (B.1), given the observed data $\mathcal{D}$, and the current parameter vector $\boldsymbol{\Psi}^{(s)}$:

$$\begin{aligned} Q(\boldsymbol{\Psi};\boldsymbol{\Psi}^{(s)}) &= \mathbb{E}\left[\log L_c(\boldsymbol{\Psi})|\mathcal{D};\boldsymbol{\Psi}^{(s)}\right] \\ &= \sum_{i=1}^{n}\sum_{k=1}^{K} \tau_{ik}^{(s)} \log \left[\pi_k(\mathbf{r}_i;\boldsymbol{\xi})\phi(y_i;\beta_{k,0}+\boldsymbol{\eta}_k^{\top}\mathbf{x}_i,\sigma_k^2)\right], \end{aligned} \tag{B.2}$$

where $\tau_{ik}^{(s)} = \mathbb{P}(Z_i = k|y_i, u_i(\cdot);\boldsymbol{\Psi}^{(s)})$ is the conditional probability that the observed pair $(u_i(\cdot), y_i)$ is generated by the kth expert. This step therefore only requires the computation of the conditional probabilities $\tau_{ik}^{(s)}$ as defined by (3.15).

M-step This step updates the value of the parameter vector $\boldsymbol{\Psi}$ by maximizing the Q-function (B.2) with respect to $\boldsymbol{\Psi}$, that is $\boldsymbol{\Psi}^{(s+1)} = \arg\max_{\boldsymbol{\Psi}} Q(\boldsymbol{\Psi};\boldsymbol{\Psi}^{(s)})$, via separate maximizations w.r.t the gating network parameters, and the experts network parameters as follows.

Updating the gating network parameters Updating the the gating network's parameters $\boldsymbol{\xi}$ consists of maximizing w.r.t $\boldsymbol{\xi}$ the following function

$$\begin{aligned} Q(\boldsymbol{\xi};\boldsymbol{\Psi}^{(s)}) &= \sum_{i=1}^{n}\sum_{k=1}^{K}\tau_{ik}^{(s)}\log\pi_k(\mathbf{r}_i;\boldsymbol{\xi}) \\ &= \sum_{i=1}^{n}\left[\sum_{k=1}^{K-1}\tau_{ik}^{(s)}(\alpha_{k,0}+\boldsymbol{\zeta}_k^{\top}\mathbf{r}_i) - \log\left(1+\sum_{k'=1}^{K-1}\exp\{\alpha_{k',0}+\boldsymbol{\zeta}_{k'}^{\top}\mathbf{r}_i\}\right)\right]\cdot \end{aligned} \tag{B.3}$$

This consists of a weighted multinomial logistic problem for which there is no closed-form solution. This can be performed by the Newton-Raphson (NR) algorithm which iteratively maximizes (B.3) according to the procedure (3.16).

Let us denote by $\boldsymbol{\xi}_1,\ldots,\boldsymbol{\xi}_{K-1}$ the parameter vectors $(\alpha_{1,0},\boldsymbol{\zeta}_1^{\top})^{\top},\ldots,(\alpha_{K-1,0},\boldsymbol{\zeta}_{K-1}^{\top})^{\top}$. Since there are $K-1$ parameter vectors to be estimated, the Hessian matrix $H(\boldsymbol{\xi};\boldsymbol{\Psi}^{(s)})$ is a block-matrix, consists of $(K-1)\times(K-1)$ blocks, in which each block $H_{k\ell}(\boldsymbol{\xi};\boldsymbol{\Psi}^{(s)})$, for $k,\ell\in[K-1]$, is given by:

$$H_{k\ell}(\boldsymbol{\xi};\boldsymbol{\Psi}^{(s)}) = \frac{\partial^2 Q(\boldsymbol{\xi};\boldsymbol{\Psi}^{(s)})}{\partial\boldsymbol{\xi}_k\partial\boldsymbol{\xi}_{\ell}^{\top}} = -\sum_{i=1}^{n}\pi_k(\mathbf{r}_i;\boldsymbol{\xi}^{(t)})\left[\delta_{k\ell}-\pi_l(\mathbf{r}_i;\boldsymbol{\xi}^{(t)})\right]\mathbf{r}_i\mathbf{r}_i^{\top},$$

where $\delta_{k\ell}$ is the Kronecker symbol ($\delta_{k\ell}=1$ if $k=\ell$, 0 otherwise). The gradient vector consists of $K-1$ gradients corresponding to the vectors $\boldsymbol{\xi}_k$, for $k\in[K-1]$, and is given by

$$g(\boldsymbol{\xi};\boldsymbol{\Psi}^{(s)}) = \frac{\partial Q(\boldsymbol{\xi};\boldsymbol{\Psi}^{(s)})}{\partial\boldsymbol{\xi}} = \left[g_1(\boldsymbol{\xi}^{(t)}),\ldots,g_{K-1}(\boldsymbol{\xi}^{(t)})\right]^{\top},$$

where, for $k\in[K-1]$, $g_k(\boldsymbol{\xi}^{(t)}) = \frac{\partial Q(\boldsymbol{\xi};\boldsymbol{\Psi}^{(s)})}{\partial\boldsymbol{\xi}_k} = \sum_{i=1}^{n}\left[\tau_{ik}^{(s)}-\pi_k(\boldsymbol{\xi}^{(t)};\mathbf{r}_i)\right]\mathbf{r}_i^{\top}$.

Updating the experts network parameters Updating the experts network's parameters $\boldsymbol{\theta}_k$ consists of maximizing the function $Q(\boldsymbol{\theta}_k;\boldsymbol{\Psi}^{(s)})$ given by

$$\begin{aligned} Q(\boldsymbol{\theta}_k;\boldsymbol{\Psi}^{(s)}) &= \sum_{i=1}^{n}\tau_{ik}^{(s)}\log\phi(y_i;\beta_{k,0}+\boldsymbol{\eta}_k^{\top}\mathbf{x}_i,\sigma_k^2) \\ &= -\frac{1}{2\sigma_k^2}\sum_{i=1}^{n}\tau_{ik}^{(s)}\left[y_i-(\beta_{k,0}+\boldsymbol{\eta}_k^{\top}\mathbf{x}_i)\right]^2 - \frac{n}{2}\log(2\pi\sigma_k^2)\cdot \end{aligned}$$

Thus, updating $\boldsymbol{\theta}_k=(\beta_{k,0},\boldsymbol{\eta}_k^{\top},\sigma_k^2)^{\top}$, consists of a weighted Gaussian regression problem where the weights are the conditional memberships $\tau_{ik}^{(s)}$, and the updates are given by (3.17).

B.2 EM-Lasso for ℓ_1-regularized MLE of the FME model

The EM-Lasso algorithm for the maximization of (3.18) firstly requires the construction of the penalized complete-data log-likelihood

$$\mathcal{L}_c(\boldsymbol{\Psi}) = \log L_c(\boldsymbol{\Psi}) - \text{Pen}_{\lambda,\chi}(\boldsymbol{\Psi}) \tag{B.4}$$

where $\log L_c(\boldsymbol{\Psi})$ is the non-regularized complete-data log-likelihood log-likelihood defined by (B.1). Thus, the EM algorithm for the FME model is implemented as follows. After starting with an initial solution $\boldsymbol{\Psi}^{(0)}$, it alternates between the two following steps, until convergence (when there is no longer a significant change in the values of the penalized log-likelihood (3.18)).

E-step. This step computes the expectation of the complete-data log-likelihood (B.4), given the observed data $\mathcal{D}$, using the current parameter vector $\boldsymbol{\Psi}^{(s)}$:

$$\mathcal{Q}_{\lambda,\chi}(\boldsymbol{\Psi};\boldsymbol{\Psi}^{(s)}) \quad = \quad \mathbb{E}\left[\mathcal{L}_c(\boldsymbol{\Psi})|\mathcal{D};\boldsymbol{\Psi}^{(s)}\right] = Q(\boldsymbol{\Psi};\boldsymbol{\Psi}^{(s)}) - \text{Pen}_{\lambda,\chi}(\boldsymbol{\Psi}), \tag{B.5}$$

which only requires the computation of the posterior probabilities of component membership $\tau_{ik}^{(s)}$ ($i \in [n]$), for each of the K experts as defined by (3.15).

M-step. This step updates the value of the parameter vector $\boldsymbol{\Psi}$ by maximizing the Q-function (B.5) with respect to $\boldsymbol{\Psi}$, that is, by computing the parameter vector update

$$\boldsymbol{\Psi}^{(s+1)} = \arg\max_{\boldsymbol{\Psi}} \mathcal{Q}_{\lambda,\chi}(\boldsymbol{\Psi};\boldsymbol{\Psi}^{(s)}). \tag{B.6}$$

The maximization is performed by separate maximizations w.r.t the gating network parameters and the experts network parameters.

B.2.1 Updating the gating network parameters

Updating the gating network parameters at the sth EM iteration consists of maximizing the following function

$$\mathcal{Q}_{\chi}(\boldsymbol{\xi};\boldsymbol{\Psi}^{(s)}) \quad = \quad Q(\boldsymbol{\xi};\boldsymbol{\Psi}^{(s)}) - \chi \sum_{k=1}^{K-1} \|\boldsymbol{\zeta}_k\|_1,$$

with

$$Q(\boldsymbol{\xi};\boldsymbol{\Psi}^{(s)}) = \quad \sum_{i=1}^{n} \left[\sum_{k=1}^{K-1} \tau_{ik}^{(s)} \left(\alpha_{k,0} + \boldsymbol{\zeta}_k^\top \mathbf{r}_i\right) - \log\left(1 + \sum_{k'=1}^{K-1} \exp\{\alpha_{k',0} + \boldsymbol{\zeta}_{k'}^\top \mathbf{r}_i\}\right)\right]$$

where $\boldsymbol{\xi} = (\alpha_{1,0}, \boldsymbol{\zeta}_1^\top, \ldots, \alpha_{K-1}, \boldsymbol{\zeta}_{K-1}^\top)^\top \in \mathbb{R}^{(q+1)(K-1)}$ is the gating network parameter vector and $\boldsymbol{\Psi}^{(s)}$ is the current estimation of model's parameters. One can see this is equivalent to solving

a weighted regularized multinomial logistic problem for which $\mathcal{Q}_\chi(\boldsymbol{\xi};\boldsymbol{\Psi}^{(s)})$ is its penalized log-likelihood. There is no closed-form solution for this kind of problem. We then use an iterative optimization algorithm to seek for a maximizer of $\mathcal{Q}_\chi(\boldsymbol{\xi};\boldsymbol{\Psi}^{(s)})$, *i.e.*, an update for the parameters of the gating network. The idea is to update only a single gate at a time, while fixing other gate's parameters to their previous estimates. Again, to update that single gate, we only update one component at a time, while fixing the other components to their previous values. This procedure for updating the gating network parameters is supported by the methodology of coordinate ascent algorithm: if the objective function consists of a concave, differentiable function and a sum of concave functions then the maximizer can be achieved by iteratively maximizing with respect to each coordinate direction at a time.

Coordinate ascent for updating the gating network. Suppose at the sth EM iteration, we wish to update the gates one by one such that it maximizes $\mathcal{Q}_\chi(\boldsymbol{\xi};\boldsymbol{\Psi}^{(s)})$. To do that, we create an outer loop, indexed by t, which cycles over the gates. For each single gate, say gate k, we partially approximate the smooth part of $\mathcal{Q}_\chi(\boldsymbol{\xi};\boldsymbol{\Psi}^{(s)})$ with respect to $(\alpha_{k,0},\boldsymbol{\zeta}_k)$ at $\boldsymbol{\xi}^{(t)}$, then optimize the obtained objective function (with respect to $(\alpha_{k,0},\boldsymbol{\zeta}_k)$) by solving a penalized weighted least square problem using coordinate ascent algorithm. Note that $\boldsymbol{\xi}^{(t)}$ denotes the current value of $\boldsymbol{\xi}$ at the iteration tth of the outer loop, while $\boldsymbol{\xi}^{(s)}$ is the value of $\boldsymbol{\xi}$ before entering the outer loop.

In particular, using Taylor expansion, one has a quadratic approximation for smooth part of $\mathcal{Q}_\chi(\boldsymbol{\xi};\boldsymbol{\Psi}^{(s)})$ with respect to $(\alpha_{k,0},\boldsymbol{\zeta}_k)$ at $\boldsymbol{\xi}^{(t)}$ given by

$$l_k(\alpha_{k,0},\boldsymbol{\zeta}_k) = -\frac{1}{2}\sum_{i=1}^{n} w_{ik}(c_{ik} - \alpha_{k,0} - \mathbf{r}_i^\top\boldsymbol{\zeta}_k)^2 + C(\boldsymbol{\xi}^{(t)}),$$

where

$$\begin{aligned} w_{ik} &= \pi_k(\boldsymbol{\xi}^{(t)};\mathbf{r}_i)\left[1-\pi_k(\boldsymbol{\xi}^{(t)};\mathbf{r}_i)\right], \quad \text{(weights)} \\ c_{ik} &= \alpha_{k,0}^{(t)} + \mathbf{r}_i^\top\boldsymbol{\zeta}_k^{(t)} + \frac{\tau_{ik}^{(s)} - \pi_k(\boldsymbol{\xi}^{(t)};\mathbf{r}_i)}{w_{ik}}, \quad \text{(working response)} \end{aligned}$$

and $C(\boldsymbol{\xi}^{(t)})$ is a function of $\boldsymbol{\xi}^{(t)}$. After calculating the partial quadratic approximation $l_k(\alpha_{k,0},\boldsymbol{\zeta}_k)$ about the current estimator $\boldsymbol{\xi}^{(t)}$, we then solve the following penalized weighted least square problem

$$\max_{(\alpha_{k,0},\boldsymbol{\zeta}_k)} l_k(\alpha_{k,0},\boldsymbol{\zeta}_k) - \chi\|\boldsymbol{\zeta}_k\|_1, \quad \chi > 0, \tag{B.7}$$

to obtain an update for the parameters of gate k.

As mentioned above, this problem could be solved by coordinate ascent algorithm. This means we will create an inner loop, indexed by m, cycles over the components of $(\alpha_{k,0},\boldsymbol{\zeta}_k)$ and update them one by one until the objective function of (B.7) does not gain any significant increase. For each $j \in [q]$, using the soft-thresholding operator (see Hastie et al. (2015), sec. 5.4), one can obtain

the closed form update for $\boldsymbol{\zeta}_{kj}$ as follows

$$\zeta_{kj}^{(m+1)} = \frac{\mathcal{S}_\chi\Big(\sum_{i=1}^n w_{ik}\mathrm{r}_{ij}(c_{ik} - \tilde{c}_{ikj}^{(m)})\Big)}{\sum_{i=1}^n w_{ik}\mathrm{r}_{ij}^2},$$

in which $\tilde{c}_{ikj}^{(m)} = \alpha_{k0}^{(m)} + \mathbf{r}_i^\top \boldsymbol{\zeta}_k^{(m)} - \zeta_{kj}^{(m)}\mathrm{r}_{ij}$ is the fitted value excluding the contribution from r_{ij}, $\mathcal{S}_\chi(\cdot)$ is a soft-thresholding operator defined by $\mathcal{S}_\chi(u) = \mathrm{sign}(u)(|u| - \chi)_+$ and $(v)_+$ a shorthand for $\max\{v, 0\}$. Note that at each iteration of the inner loop, only one component is updated while the others are kept to their previous values, that means $\zeta_{kh}^{(m+1)} = \zeta_{kh}^{(m)}$ for all $h \neq j$. For $\alpha_{k,0}$, the closed-form update is given by

$$\alpha_{k,0}^{(m+1)} = \frac{\sum_{i=1}^n w_{ik}(c_{ik} - \mathbf{r}_i^\top \boldsymbol{\zeta}_k^{(m+1)})}{\sum_{i=1}^n w_{ik}}.$$

Once the inner loop converges, the new values of $(\alpha_{k,0}, \boldsymbol{\zeta}_k)$ are used for the updating procedure of the next gate. When all the gates have their new values, *i.e.*, after $K-1$ inner loops, we perform a backtracking line search before actually updating the gating network's parameters for the next t-indexed iteration. More precisely, the update is $\boldsymbol{\xi}^{(t+1)} = (1-\nu)\boldsymbol{\xi}^{(t)} + \nu\bar{\boldsymbol{\xi}}^{(t)}$ where $\bar{\boldsymbol{\xi}}^{(t)}$ is the output after $K-1$ inner loops and ν is backtrackingly determined to ensure $\mathcal{Q}_\chi(\boldsymbol{\xi}^{(t+1)}; \boldsymbol{\Psi}^{(s)}) > \mathcal{Q}_\chi(\boldsymbol{\xi}^{(t)}; \boldsymbol{\Psi}^{(s)})$.

We keep running the t-indexed loop until convergence, *i.e.*, when there is no significant relative variation in $\mathcal{Q}_\chi(\boldsymbol{\xi}; \boldsymbol{\Psi}^{(s)})$. Once $\alpha_{k,0}$ and ζ_{kj} reach their optimal values $\widetilde{\alpha}_{k,0}$ and $\widetilde{\zeta}_{kj}$ for all $k \in [K-1], j \in [q]$, the update for gating network's parameters is then $\boldsymbol{\xi}^{(s+1)} = (\widetilde{\alpha}_{1,0}, \widetilde{\boldsymbol{\zeta}}_1^\top, \ldots, \widetilde{\alpha}_{K-1,0}, \boldsymbol{\zeta}_{K-1}^\top)^\top$.

B.2.2 Updating the experts network parameters

The maximization step for updating the expert parameters $\boldsymbol{\theta}_k$ consists of maximizing the function $\mathcal{Q}_\lambda(\boldsymbol{\theta}_k; \boldsymbol{\Psi}^{(s)})$ given by

$$\mathcal{Q}_\lambda(\boldsymbol{\theta}_k; \boldsymbol{\Psi}^{(s)}) \;=\; Q(\boldsymbol{\theta}_k; \boldsymbol{\Psi}^{(s)}) - \lambda\|\boldsymbol{\eta}_k\|_1,$$

with

$$Q(\boldsymbol{\theta}_k; \boldsymbol{\Psi}^{(s)}) = -\frac{1}{2\sigma_k^2}\sum_{i=1}^n \tau_{ik}^{(s)}\Big[y_i - (\beta_{k,0} + \boldsymbol{\eta}_k^\top \mathbf{x}_i)\Big]^2 - \frac{n}{2}\log(2\pi\sigma_k^2),$$

where $\boldsymbol{\theta}_k = (\beta_{k,0}, \boldsymbol{\eta}_k^\top, \sigma_k^2)^\top \in \mathbb{R}^{p+2}$ is the unknown vector and $\boldsymbol{\Psi}^{(s)}$ is the current estimation of model's parameters. There is no closed-form solution for this optimization problem, we then solve it by an iterative optimization algorithm similarly to updating the gating network parameters. We first perform the update for $(\beta_{k,0}, \boldsymbol{\eta}_k)$ while fixing σ_k^2. This corresponds to solving a weighted LASSO problem where the weights are the the posterior experts memberships $\tau_{ik}^{(s)}$. Once $(\beta_{k,0}, \boldsymbol{\eta}_k)$ has new value, the variance σ_k^2 is updated straightforwardly by the standard update of a weighted

Gaussian regression.

More specifically, when σ_k^2, the variance of expert k, is fixed to $\sigma_k^{2(s)}$, updating $(\beta_{k,0}, \boldsymbol{\eta}_k)$ consists of solving the following weighted LASSO problem:

$$\max_{(\beta_{k,0},\boldsymbol{\eta}_k)} -\frac{1}{2\sigma_k^{2(s)}}\sum_{i=1}^{n}\tau_{ik}^{(s)}\left[y_i-(\beta_{k,0}+\boldsymbol{\eta}_k^\top\mathbf{x}_i)\right]^2-\frac{n}{2}\log(2\pi\sigma_k^{2(s)})-\lambda\sum_{j=1}^{q}|\boldsymbol{\eta}_{kj}|,\quad \lambda>0, \tag{B.8}$$

which can be solved by coordinate ascent algorithm. For each $j \in [p]$, the closed-form update for $\boldsymbol{\eta}_{kj}$ is given by

$$\eta_{kj}^{(m+1)}=\frac{\mathcal{S}_{\lambda\sigma_k^{2(s)}}\left(\sum_{i=1}^{n}\tau_{ik}^{(s)}\mathrm{x}_{ij}(y_i-\tilde{y}_{ij}^{(m)})\right)}{\sum_{i=1}^{n}\tau_{ik}^{(s)}\mathrm{x}_{ij}^2},$$

in which $\tilde{y}_{ij}^{(m)}=\beta_{k,0}^{(m)}+\mathbf{x}_i^\top\boldsymbol{\eta}_k^{(m)}-\eta_{kj}^{(m)}\mathrm{x}_{ij}$ is the fitted value excluding the contribution from x_{ij} and $\mathcal{S}_\chi(\cdot)$ is the soft-thresholding operator (see Hastie et al. (2015), sec. 5.4). Here m denotes the mth iteration of the coordinate ascent algorithm. The update for $\beta_{k,0}$ is

$$\beta_{k,0}^{(m+1)}=\frac{\sum_{i=1}^{n}\tau_{ik}^{(s)}(y_i-\mathbf{x}_i^\top\boldsymbol{\eta}_k^{(m+1)})}{\sum_{i=1}^{n}\tau_{ik}^{(s)}}.$$

We keep updating the components of $(\beta_{k,0}, \boldsymbol{\eta}_k)$ cyclically until the change in objective function of (B.8) is small enough. So, the update for $(\beta_{k,0}, \boldsymbol{\eta}_k)$ in this EM iteration is then $(\beta_{k,0}^{(s+1)}, \boldsymbol{\eta}_k^{(s+1)}) = (\beta_{k,0}^*, \boldsymbol{\eta}_k^*)$ where the latter is the optimal solution of (B.8). Finally, the update for σ_k^2 is given by

$$\sigma^{2}{}_k^{(s+1)}=\frac{\sum_{i=1}^{n}\tau_{ik}^{(s)}(y_i-\beta_{k,0}^{(s+1)}-\mathbf{x}_i^\top\boldsymbol{\eta}_k^{(s+1)})^2}{\sum_{i=1}^{n}\tau_{ik}^{(s)}}.$$

Hence, the update for the value of parameters vector $\boldsymbol{\Psi}$ at M-step, *i.e.*, the solution to problem (B.6), is $\boldsymbol{\Psi}^{(s+1)} = (\boldsymbol{\xi}^{(s+1)}, \boldsymbol{\theta}_1^{(s+1)}, \ldots, \boldsymbol{\theta}_K^{(s+1)})$ where $\boldsymbol{\xi}^{(s+1)}$ and $\boldsymbol{\theta}_k^{(s+1)}, k \in [K]$, are solved by maximizing $\mathcal{Q}_\chi(\boldsymbol{\xi}; \boldsymbol{\Psi}^{(s)})$ and $\mathcal{Q}_\lambda(\boldsymbol{\theta}_k; \boldsymbol{\Psi}^{(s)})$, respectively, using the algorithms described above. The EM algorithm monotonically increases (3.18). Furthermore, the sequence of parameter estimates generated by the EM algorithm converges toward a local maximum of the log-likelihood function (Dempster et al., 1977; McLachlan and Krishnan, 2008; Wu, 1983).

B.3 EM-iFME for updating iFME model parameters

B.3.1 Updating the gating network parameters

This section presents the using of Dantzig selector to solve problem (3.36). Let us simplify the subscript k in the notations and rewrite the problem under matrix form as follows

$$\begin{aligned} \max_{\widetilde{\boldsymbol{\omega}}\in\mathbb{R}^{2q+1}} \quad & -\frac{1}{2}\|\mathbf{c}_{\boldsymbol{w}} - \mathbf{X}_{\boldsymbol{w}}\widetilde{\boldsymbol{\omega}}\|_2^2 - \chi\|\Omega\widetilde{\boldsymbol{\omega}}\|_1 \\ \text{subject to} \quad & \mathbf{A}\widetilde{\boldsymbol{\omega}} = \mathbf{0}_q, \end{aligned} \tag{B.9}$$

where $\widetilde{\boldsymbol{\omega}} = (\alpha_0,\ {\boldsymbol{\omega}^{[d_1]}}^\top, {\boldsymbol{\omega}^{[d_2]}}^\top)^\top$ is the unknown coefficients vector, $\mathbf{c}_{\boldsymbol{w}} = (\sqrt{w_1}c_1, \ldots, \sqrt{w_n}c_n)^\top$ is the weighted working response vector, $\mathbf{X}_{\boldsymbol{w}} = [\sqrt{\boldsymbol{w}}|\mathbf{S}_{\boldsymbol{w}}|\mathbf{0}_{n\times q}] \in \mathbb{R}^{2q+1}$ is the weighted design matrix, $\Omega = \text{diag}(0, \mathbf{1}_q^\top, \varrho\mathbf{1}_q^\top)$ is the diagonal weighting matrix and $\mathbf{A} = [\mathbf{0}_q|\mathbf{A}_q^{[d_2]}{\mathbf{A}_q^{[d_1]}}^{-1}| - \boldsymbol{I}_q]$ is the constraints matrix. Here, $\sqrt{\boldsymbol{w}} = (\sqrt{w_1}, \ldots, \sqrt{w_n})^\top$, $\mathbf{S}_{\boldsymbol{w}} = [\sqrt{w_1}\mathbf{s}_1, \ldots, \sqrt{w_n}\mathbf{s}_n]^\top$, with $\mathbf{s}_i$ are the design vectors (see (3.27)), $\mathbf{0}_{n\times q} \in \mathbb{R}^{n\times q}$ contains 0's, $\mathbf{0}_q \in \mathbb{R}^q$ contains 0's, $\mathbf{1}_q \in \mathbb{R}^q$ contains 1's and $\boldsymbol{I}_q$ is the identity matrix in $\mathbb{R}^{q\times q}$. This problem can be viewed as the problem of finding a sparse solution via Lasso estimate for a linear regression model with constraints. Therefore, we can solve it alternatively by Dantzig selector estimate as the solution to the following problem

$$\begin{aligned} \max_{\widetilde{\boldsymbol{\omega}}\in\mathbb{R}^{2q+1}} \quad & -\|\Omega\widetilde{\boldsymbol{\omega}}\|_1 \\ \text{subject to} \quad & \begin{cases} |\mathbf{X}_{\boldsymbol{w}}^\top(\mathbf{c}_{\boldsymbol{w}} - \mathbf{X}_{\boldsymbol{w}}\widetilde{\boldsymbol{\omega}})| \leqslant \chi\mathbf{1}_{2q+1}, \\ \mathbf{A}\widetilde{\boldsymbol{\omega}} = \mathbf{0}_q, \end{cases} \end{aligned}$$

where the absolute value operator is understood componentwise. By decomposing $\widetilde{\boldsymbol{\omega}}$ into its positive and negative parts, $\widetilde{\boldsymbol{\omega}} = \widetilde{\boldsymbol{\omega}}_+ - \widetilde{\boldsymbol{\omega}}_-$, the above problem becomes

$$\begin{aligned} \max_{(\widetilde{\boldsymbol{\omega}}_+,\widetilde{\boldsymbol{\omega}}_-)\in\mathbb{R}^{4q+2}} \quad & -[0, \mathbf{1}_q^\top, \varrho\mathbf{1}_q^\top, 0, \mathbf{1}_q^\top, \varrho\mathbf{1}_q^\top] \begin{bmatrix} \widetilde{\boldsymbol{\omega}}_+ \\ \widetilde{\boldsymbol{\omega}}_- \end{bmatrix} \\ \text{subject to} \quad & \begin{cases} \begin{bmatrix} \mathbf{X}_{\boldsymbol{w}}^\top\mathbf{X}_{\boldsymbol{w}} & -\mathbf{X}_{\boldsymbol{w}}^\top\mathbf{X}_{\boldsymbol{w}} \\ -\mathbf{X}_{\boldsymbol{w}}^\top\mathbf{X}_{\boldsymbol{w}} & \mathbf{X}_{\boldsymbol{w}}^\top\mathbf{X}_{\boldsymbol{w}} \end{bmatrix} \begin{bmatrix} \widetilde{\boldsymbol{\omega}}_+ \\ \widetilde{\boldsymbol{\omega}}_- \end{bmatrix} \leqslant \begin{bmatrix} \chi + \mathbf{X}_{\boldsymbol{w}}^\top\mathbf{c}_{\boldsymbol{w}} \\ \chi - \mathbf{X}_{\boldsymbol{w}}^\top\mathbf{c}_{\boldsymbol{w}} \end{bmatrix}, \\ \begin{bmatrix} \mathbf{A} & -\mathbf{A} \end{bmatrix} \begin{bmatrix} \widetilde{\boldsymbol{\omega}}_+ \\ \widetilde{\boldsymbol{\omega}}_- \end{bmatrix} = \mathbf{0}_{4q+2}, \\ \widetilde{\boldsymbol{\omega}}_+ \geqslant \mathbf{0}_{2q+1}, \quad \widetilde{\boldsymbol{\omega}}_- \geqslant \mathbf{0}_{2q+1}, \end{cases} \end{aligned} \tag{B.10}$$

which is a standard linear program with $4q + 2$ variables, therefore can be easily solved by available toolboxes for linear programming, e.g., Matlab's linprog function. Finally, the solution $\widetilde{\boldsymbol{\omega}}^* = (\alpha_0^*,\ {\boldsymbol{\omega}^{[d_1]*}}^\top, {\boldsymbol{\omega}^{[d_2]*}}^\top)^\top$ to the original problem could be retrieved by the relation $\widetilde{\boldsymbol{\omega}}^* = \widetilde{\boldsymbol{\omega}}_+^* - \widetilde{\boldsymbol{\omega}}_-^*$ where $(\widetilde{\boldsymbol{\omega}}_+^{*\top},\ \widetilde{\boldsymbol{\omega}}_-^{*\top})^\top$ is the solution of (B.10).

B.3.2 Updating the expert network parameters

This section presents how to solve problem (3.37). Firstly, we fix σ_k^2 to its previous estimate and perform an update for $(\beta_{k,0}, \boldsymbol{\gamma}_k^{[d_1]})$, it corresponds to solving a penalized weighted least square problem with constraints. From now on, let us simplify the subscript k in the notations and rewrite the problem under matrix form as follows

$$\begin{aligned} \max_{\widetilde{\boldsymbol{\gamma}} \in \mathbb{R}^{2p+1}} \quad & -\frac{1}{2}\|\mathbf{y}_{\boldsymbol{\sigma\tau}} - \mathbf{X}_{\boldsymbol{\sigma\tau}}\widetilde{\boldsymbol{\gamma}}\|_2^2 - \lambda\|\Lambda\widetilde{\boldsymbol{\gamma}}\|_1 - \frac{n_k}{2}\log(2\pi\sigma^2) \\ \text{subject to} \quad & \mathbf{A}\widetilde{\boldsymbol{\gamma}} = \mathbf{0}_p, \end{aligned}$$

where $\widetilde{\boldsymbol{\gamma}} = (\beta_0,\ \boldsymbol{\gamma}^{[d_1]\top}, \boldsymbol{\gamma}^{[d_2]\top})^\top$ is the unknown coefficients vector, $\mathbf{y}_{\boldsymbol{\sigma\tau}} = (\sigma\sqrt{\tau_1}y_1, \ldots, \sigma\sqrt{\tau_n}y_n)^\top \in \mathbb{R}^n$ is the weighted response vector, $\mathbf{X}_{\boldsymbol{\sigma\tau}} = \boldsymbol{\sigma} \odot [\sqrt{\boldsymbol{\tau}}|\mathbf{V}_{\boldsymbol{\tau}}|\mathbf{0}_{n\times p}] \in \mathbb{R}^{n\times(2p+1)}$ is the weighted design matrix, $\Lambda = \text{diag}(0, \mathbf{1}_p^\top, \rho\mathbf{1}_p^\top)$ is the diagonal weighting matrix, $n_k = \sum_{i=1}^n \tau_{ik}$ and $\mathbf{A} = [\mathbf{0}_p|\mathbf{A}_p^{[d_2]}{\mathbf{A}_p^{[d_1]}}^{-1}| - \boldsymbol{I}_p] \in \mathbb{R}^{p\times(2p+1)}$ is the constraints matrix. Here, $\sqrt{\boldsymbol{\tau}} = (\sqrt{\tau_1}, \ldots, \sqrt{\tau_n})^\top \in \mathbb{R}^n$, $\mathbf{V}_{\boldsymbol{\tau}} = [\sqrt{\tau_1}\mathbf{v}_1, \ldots, \sqrt{\tau_n}\mathbf{v}_n]^\top \in \mathbb{R}^{n\times p}$, with $\mathbf{v}_i$ are the design vectors (see (3.27)), $\mathbf{0}_{n\times p} \in \mathbb{R}^{n\times p}$ contains 0's, $\mathbf{0}_p \in \mathbb{R}^p$ contains 0's, $\mathbf{1}_p \in \mathbb{R}^p$ contains 1's and $\boldsymbol{I}_p$ is the identity matrix in $\mathbb{R}^{p\times p}$. As the last term in the objective function is independent of $\widetilde{\boldsymbol{\gamma}}$, this problem is similar to the problem (B.9) and then can be solved by Dantzig selector estimate as the solution to the following problem

$$\begin{aligned} \max_{\widetilde{\boldsymbol{\gamma}} \in \mathbb{R}^{2p+1}} \quad & -\|\Lambda\widetilde{\boldsymbol{\gamma}}\|_1 \\ \text{subject to} \quad & \begin{cases} |\mathbf{X}_{\boldsymbol{\sigma\tau}}^\top(\mathbf{y}_{\boldsymbol{\sigma\tau}} - \mathbf{X}_{\boldsymbol{\sigma\tau}}\widetilde{\boldsymbol{\gamma}})| \leqslant \lambda\mathbf{1}_{2p+1}, \\ \mathbf{A}\widetilde{\boldsymbol{\gamma}} = \mathbf{0}_p. \end{cases} \end{aligned}$$

Similarly to the problem in the gating network update, by decomposing $\widetilde{\boldsymbol{\gamma}}$ into its positive and negative parts, $\widetilde{\boldsymbol{\gamma}} = \widetilde{\boldsymbol{\gamma}}_+ - \widetilde{\boldsymbol{\gamma}}_-$, the above problem becomes

$$\begin{aligned} \max_{(\widetilde{\boldsymbol{\gamma}}_+, \widetilde{\boldsymbol{\gamma}}_-) \in \mathbb{R}^{4p+2}} \quad & -[0, \mathbf{1}_p^\top, \rho\mathbf{1}_p^\top, 0, \mathbf{1}_p^\top, \varrho\mathbf{1}_p^\top] \begin{bmatrix} \widetilde{\boldsymbol{\gamma}}_+ \\ \widetilde{\boldsymbol{\gamma}}_- \end{bmatrix} \\ \text{subject to} \quad & \begin{cases} \begin{bmatrix} \mathbf{X}_{\boldsymbol{\sigma\tau}}^\top\mathbf{X}_{\boldsymbol{\sigma\tau}} & -\mathbf{X}_{\boldsymbol{\sigma\tau}}^\top\mathbf{X}_{\boldsymbol{\sigma\tau}} \\ -\mathbf{X}_{\boldsymbol{\sigma\tau}}^\top\mathbf{X}_{\boldsymbol{\sigma\tau}} & \mathbf{X}_{\boldsymbol{\sigma\tau}}^\top\mathbf{X}_{\boldsymbol{\sigma\tau}} \end{bmatrix} \begin{bmatrix} \widetilde{\boldsymbol{\gamma}}_+ \\ \widetilde{\boldsymbol{\gamma}}_- \end{bmatrix} \leqslant \begin{bmatrix} \lambda + \mathbf{X}_{\boldsymbol{\sigma\tau}}^\top\mathbf{y}_{\boldsymbol{\sigma\tau}} \\ \lambda - \mathbf{X}_{\boldsymbol{\sigma\tau}}^\top\mathbf{y}_{\boldsymbol{\sigma\tau}} \end{bmatrix}, \\ \begin{bmatrix} \mathbf{A} & -\mathbf{A} \end{bmatrix} \begin{bmatrix} \widetilde{\boldsymbol{\gamma}}_+ \\ \widetilde{\boldsymbol{\gamma}}_- \end{bmatrix} = \mathbf{0}_{4p+2}, \\ \widetilde{\boldsymbol{\gamma}}_+ \geqslant \mathbf{0}_{2p+1}, \quad \widetilde{\boldsymbol{\gamma}}_- \geqslant \mathbf{0}_{2p+1}, \end{cases} \end{aligned} \tag{B.11}$$

which is a standard linear program with $4q+2$ variables and can be solved similarly to the gating network case. The solution $\widetilde{\boldsymbol{\gamma}}^* = (\beta_0^*,\ \boldsymbol{\gamma}^{[d_1]*\top}, \boldsymbol{\gamma}^{[d_2]*\top})^\top$ to the original problem is retrieved by the relation $\widetilde{\boldsymbol{\gamma}}^* = \widetilde{\boldsymbol{\gamma}}_+^* - \widetilde{\boldsymbol{\gamma}}_-^*$ where $(\widetilde{\boldsymbol{\gamma}}_+^{*\top},\ \widetilde{\boldsymbol{\gamma}}_-^{*\top})^\top$ is the solution of (B.11).

Finally, the update for σ_k^2 is

$$\sigma_k^{2(s+1)} = \frac{\sum_{i=1}^n \tau_{ik}^{(s)}(y_i - \beta_{k,0}^{(s+1)} - \mathbf{v}_i^\top \boldsymbol{\gamma}_k^{[d_1]^{(s+1)}})^2}{\sum_{i=1}^n \tau_{ik}^{(s)}},$$

in which $\beta_{k,0}^{(s+1)}, \boldsymbol{\gamma}_k^{[d_1]^{(s+1)}}$ are the new updates for $\beta_{k,0}$ and $\boldsymbol{\gamma}_k^{[d_1]}$.

Appendix C

C.1 Proofs

C.1.1 Proof of Proposition 5.3.1

First, by definition, $\mathcal{R}_c(g)$ is obtained by relaxing the constraint in $\mathcal{T}_c(g)$ from $\boldsymbol{P} \in \boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \boldsymbol{\pi})$ to $\boldsymbol{P} \in \boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \cdot)$, therefore the inequality

$$\inf_{g \in \mathcal{M}_K} \mathcal{T}_c(g) \geqslant \inf_{g \in \mathcal{M}_K} \mathcal{R}_c(g)$$

is trivial.

The inequality in the opposite direction is also true. Indeed, let $g^\star = \arg\inf_{g \in \mathcal{M}_K} \mathcal{R}_c(g)$ be a minimizer of $\mathcal{R}_c(g)$, *i.e.*, we have $\mathcal{R}_c(g^\star) = \inf_{g \in \mathcal{M}_K} \mathcal{R}_c(g)$. Then, in the remainder we will show that $\inf_{g \in \mathcal{M}_K} \mathcal{T}_c(g) \leqslant \mathcal{R}_c(g^\star)$. We denote the gating and expert functions of $g^\star$ by $\{\pi^\star(\mathbf{x}), \varphi_k^\star(\cdot|\mathbf{x})\}$, $k \in [K]$.

According to (5.15), the gating functions are calculated by

$$\pi_k^\star(\mathbf{x}) = \sum_{\ell=1}^{MK} \mathcal{P}_{\ell k}(g^\star, \mathbf{x}), \quad \forall \mathbf{x} \in \mathcal{X}, \tag{C.1}$$

where $\mathcal{P}_{\ell k}(g^\star, \mathbf{x})$ denotes the entry (ℓ, k) of $\mathcal{P}(g^\star, \mathbf{x})$ as defined in (5.13). The relation (C.1) means that the matrix $\mathcal{P}(g^\star, \mathbf{x})$ is belonging to $\boldsymbol{\Pi}_{\mathbf{x}}(\cdot, \boldsymbol{\pi}^\star)$. On the other hand, by its definition $\mathcal{P}(g^\star, \mathbf{x})$ is obviously belonging to $\boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \cdot)$. Hence, $\mathcal{P}(g^\star, \mathbf{x}) \in \boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \boldsymbol{\pi}^\star)$. Note that, this holds for all $\mathbf{x} \in \mathcal{X}$. Therefore, taking expectation we have

$$\begin{aligned}
\inf_{g \in \mathcal{M}_K} \mathcal{T}_c(g) &= \inf_{g \in \mathcal{M}_K} \left\{ \mathbb{E}\left[\inf_{\boldsymbol{P} \in \boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \boldsymbol{\pi})} \sum_{\ell=1}^{KM} \sum_{k=1}^{K} P_{\ell k}\ c\left(\widehat{\varphi}_\ell(\cdot|\mathbf{x}), \varphi_k(\cdot|\mathbf{x})\right) \right] \right\} \\
&\leqslant \mathbb{E}\left[\inf_{\boldsymbol{P} \in \boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \boldsymbol{\pi}^\star)} \sum_{\ell=1}^{L} \sum_{k=1}^{K} \mathcal{P}_{\ell k}(g^\star, \mathbf{x})\ c\left(\widehat{\varphi}_\ell(\cdot|\mathbf{x}), \varphi_k^\star(\cdot|\mathbf{x})\right) \right] \\
&= \mathcal{R}_c(g^\star),
\end{aligned}$$

which completes the proof. □

C.1.2 Proof of Proposition 5.3.2

Continuity. Firstly, we show that, conditional on $\mathbf{x}$, $\mathcal{I}(g, \mathbf{x})$ is continuous in g. Indeed, because $\mathcal{C}(\boldsymbol{P}, \boldsymbol{\varphi})$ is a continuous function, and $\boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \cdot)$ is a compact set for all $\mathbf{x} \in \mathcal{X}$, the function defined by $\boldsymbol{\varphi} \mapsto \inf_{\boldsymbol{P} \in \boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \cdot)} \mathcal{C}(\boldsymbol{P}, \boldsymbol{\varphi})$ is continuous, moreover, the infimum is attained. By definition, $\mathcal{I}(g, \mathbf{x})$ depends on g only through $\boldsymbol{\varphi}$, it follows that $\mathcal{I}(g, \mathbf{x})$ is also continuous in g.

Then, since $\mathcal{I}(g, \mathbf{x})$ is bounded for all g and $\mathbf{x}$ by the assumption, from the Lebesgue's dominated convergence theorem we have that: for all sequence $(g^{(t)})_{t \geqslant 1}$ such that $g^{(t)} \to g$,

$$\lim_{t \to \infty} \int_{\mathcal{X}} \mathcal{I}(g^{(t)}, \mathbf{x}) d\mu(\mathbf{x}) = \int_{\mathcal{X}} \mathcal{I}(g, \mathbf{x}) d\mu(\mathbf{x}) = \mathcal{R}_c(g),$$

that is $\mathcal{R}_c(g)$ is continuous.

Convexity. We observe that $\boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \cdot)$ is a convex set for all $\mathbf{x} \in \mathcal{X}$ (it can be verified easily by definition). It follows that $\mathcal{I}(g, \mathbf{x})$ is convex in g because of the facts that $\mathcal{C}(\boldsymbol{P}, \boldsymbol{\varphi})$ is convex and taking infimum over a convex set preserves convexity. Then, let $g_1, g_2 \in \mathcal{M}_K$ and $\lambda_1, \lambda_2 \in [0, 1]$ such that $\lambda_1 + \lambda_2 = 1$, one have

$$\begin{aligned} \mathcal{R}_c(\lambda_1 g_1 + \lambda_2 g_2) &= \int_{\mathcal{X}} \mathcal{I}(\lambda_1 g_1 + \lambda_2 g_2, \mathbf{x}) d\mu(x) \\ &\leqslant \int_{\mathcal{X}} \left(\lambda_1 \mathcal{I}(g_1, \mathbf{x}) + \lambda_2 \mathcal{I}(g_2, \mathbf{x})\right) d\mu(x) \\ &= \lambda_1 \mathcal{R}_c(g_1) + \lambda_2 \mathcal{R}_c(g_2), \end{aligned}$$

which is the definition for the convexity of $\mathcal{R}_c(g)$. □

C.1.3 Proof of Proposition 5.3.3

For each $m \in [M]$, the assumption that the local estimator $\widehat{\boldsymbol{\theta}}_m$ is consistent implies that its expert parameters $\widehat{\boldsymbol{\beta}}_1^{(m)}, \ldots, \widehat{\boldsymbol{\beta}}_K^{(m)}$ must be ordered and its gating parameter $\widehat{\boldsymbol{\alpha}}^{(m)}$ must be initialized as those of $\boldsymbol{\theta}^*$. Therefore, their weighted average $\bar{\boldsymbol{\theta}}^W = \sum_{m=1}^M \lambda_m \widehat{\boldsymbol{\theta}}_m$ is also a consistent estimator of $\boldsymbol{\theta}^*$, *i.e.*, we have $\bar{\boldsymbol{\theta}}^W \to \boldsymbol{\theta}^*$ in probability. It follows that the expected transportation divergence between the two corresponding ME models, $\bar{f}^W$ and f^*, will tend to zero almost surely as $N \to \infty$, *i.e.*, we have $\mathcal{T}(\bar{f}^W, f^*) \to 0$.

On the other hand, by definition, the ME model corresponding to the reduction estimator $\bar{\boldsymbol{\theta}}^R$ is the one, among all $g \in \mathcal{M}_K$, minimizes the expected transportation divergence from the true model f^*. Therefore, we have

$$\mathcal{T}(\bar{f}^R, f^*) \leqslant \mathcal{T}(\bar{f}^W, f^*) \to 0 \tag{C.2}$$

almost surely. As a consequence, $\bar{\boldsymbol{\beta}}_k^R \to \boldsymbol{\beta}_k^*$ in probability, because any possibility that it does not

hold will lead to a contradiction to (C.2). Indeed, suppose that there exists $k \in [K]$ such that the expert parameter $\bar{\boldsymbol{\beta}}_k^R$ does not tend to $\boldsymbol{\beta}_k^*$ at some event. Then due to the property of the cost function $c(\cdot, \cdot)$, the value $\mathcal{T}(\bar{f}^R, f^*)$ must be bounded from below by some positive constant, which obviously contradicts the fact (C.2).

We are left to show that $\bar{\boldsymbol{\alpha}}^R$ also converge to $\boldsymbol{\alpha}^*$ in probability. The estimation of $\bar{\boldsymbol{\alpha}}^R$ via MLE in (5.21) makes itself a consistent estimator for the parameter of the logistic regression model with predictors $\mathbf{x}_s \in \boldsymbol{X}_S$ and responses a_{sk} given in (5.20). Since $\widehat{\boldsymbol{\alpha}}^{(m)} \xrightarrow{p} \boldsymbol{\alpha}^*$ and the fact that the gating function is continuous with respect to the parameter, for all $\mathbf{x}_s \in \boldsymbol{X}_S$ we have $\pi_k(\mathbf{x}_s; \widehat{\boldsymbol{\alpha}}^{(m)}) \xrightarrow{p} \pi_k(\mathbf{x}_s; \boldsymbol{\alpha}^*)$ by the Slusky theorem. It follows

$$\sum_{m=1}^{M} \lambda_m \pi_k(\mathbf{x}_s; \widehat{\boldsymbol{\alpha}}^{(m)}) \xrightarrow{p} \pi_k(\mathbf{x}_s; \boldsymbol{\alpha}^*), \quad \forall \mathbf{x}_s \in \boldsymbol{X}_S. \tag{C.3}$$

Moreover, because at the convergence of the MM algorithm, the obtained experts parameters are supposed to be the theoretical solution $\bar{\boldsymbol{\beta}}_k^R$, the matrix $\mathcal{P}_{\ell k}(g^*, \mathbf{x})$ in (5.20) can be written by

$$\mathcal{P}_{\ell k}(g^*, \mathbf{x}_s) = \begin{cases} \widehat{\pi}_\ell(\mathbf{x}_s) & \text{if } k = \underset{k' \in [K]}{\arg\inf}\ c(\widehat{\varphi}_\ell(\cdot|\mathbf{x}_s), \varphi_{k'}^R(\cdot|\mathbf{x}_s)) \\ 0 & \text{otherwise,} \end{cases} \tag{C.4}$$

where $\varphi_{k'}^R(\cdot|\mathbf{x}_s)$ denotes $\varphi_{k'}(\cdot|\mathbf{x}_s; \bar{\boldsymbol{\beta}}_k^R)$, the kth expert density of the reduced ME model. Therefore, by noting the convention about the connection between $\lambda_m \pi_k(\mathbf{x}_s; \widehat{\boldsymbol{\alpha}}^{(m)})$ and $\widehat{\pi}_\ell(\mathbf{x}_s)$ (discussed at the beginning of the Subsection 5.2.3), we can see that the responses a_{sk} are in fact the approximations of $\pi_k(\mathbf{x}_s; \boldsymbol{\alpha}^*)$. This means $\bar{\boldsymbol{\alpha}}^R$ is also a consistent estimation of $\boldsymbol{\alpha}^*$. □

C.1.4 Proof of Proposition 5.4.1

We will use the definition to prove that $\mathcal{S}_c(g, g^{(t)})$ given in (5.17) is a majorant function of $\mathcal{R}_c(g)$ given in (5.12) at $g^{(t)}$. That is, we will show that $\mathcal{S}_c(g, g^{(t)}) \geqslant \mathcal{R}_c(g)$ and the equality holds when $g = g^{(t)}$.

First, for all $g \in \mathcal{M}_K$ and $\mathbf{x} \in \mathcal{X}$, from the definition of $\mathcal{P}_{\ell k}(g^{(t)}, \mathbf{x})$ we see that

$$\sum_{k=1}^{K} \mathcal{P}_{\ell k}(g^{(t)}, \mathbf{x}) = \widehat{\pi}_\ell(\mathbf{x}), \quad \forall \ell \in [MK].$$

This means the transportation plan $\boldsymbol{P} = [\mathcal{P}_{\ell k}(g^{(t)}, \mathbf{x})]_{\ell k}$ is a member of $\boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}}, \cdot)$. Therefore, by the definition of $\mathcal{S}_c(g, g^{(t)})$ and $\mathcal{R}_c(g)$ we have $\mathcal{S}_c(g, g^{(t)}) \geqslant \mathcal{R}_c(g)$.

Finally, at $g = g^{(t)}$, by definition the functions $\mathcal{S}_c(g, g^{(t)})$ and $\mathcal{R}_c(g)$ are respectively given by

$$\mathcal{R}_c(g^{(t)}) = \mathbb{E}\left[\inf_{\boldsymbol{P}\in\boldsymbol{\Pi}_{\mathbf{x}}(\widehat{\boldsymbol{\pi}},\cdot)} \sum_{\ell=1}^{MK}\sum_{k=1}^{K} P_{\ell k}\; c\left(\widehat{\varphi}_\ell(\cdot|\mathbf{x}), \varphi_k^{(t)}(\cdot|\mathbf{x})\right)\right], \tag{C.5}$$

$$\mathcal{S}_c(g^{(t)}, g^{(t)}) = \mathbb{E}\left[\sum_{\ell=1}^{MK}\sum_{k=1}^{K} \mathcal{P}_{\ell k}(g^{(t)}, \mathbf{x})\; c(\widehat{\varphi}_\ell(\cdot|\mathbf{x}), \varphi_k^{(t)}(\cdot|\mathbf{x}))\right].$$

Obviously, the transportation $\boldsymbol{P} = [\mathcal{P}_{\ell k}(g^{(t)}, \mathbf{x})]_{\ell k}$ with $\mathcal{P}_{\ell k}(g^{(t)}, \mathbf{x})$ given in (5.18) is a minimizer of the minimization in (C.5). This means that the two quantities above are equal. □

C.1.5 Proof of Proposition 5.4.2

To see this, it is necessary to minimize the function $\mathcal{K}(\boldsymbol{\beta}, \sigma^2)$ given in (5.22). Firstly, we observe that $\mathcal{K}(\boldsymbol{\beta}, \sigma^2)$ is continuously differentiable and convex with respect to $\boldsymbol{\beta}$, so any stationary point is a global minimizer of $\mathcal{K}(\boldsymbol{\beta}, \sigma^2)$ as a function of $\boldsymbol{\beta}$. On the other hand, $\mathcal{K}(\boldsymbol{\beta}, \sigma^2)$ is coercive with respect to σ^2, then it has a global minimizer that satisfies the first-order optimality condition.

The partial derivatives of $\mathcal{K}(\boldsymbol{\beta}, \sigma^2)$ with respect to $\boldsymbol{\beta}$ and σ^2 are given by

$$\frac{\partial \mathcal{K}}{\partial \boldsymbol{\beta}} = \mathbb{E}\left[\sum_{\ell=1}^{MK} \mathcal{P}_{\ell k}^{(t)}(\mathbf{x}) \frac{\mathbf{x}\mathbf{x}^\top(\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}_\ell)}{\sigma^2}\right],$$

$$\frac{\partial \mathcal{K}}{\partial \sigma^2} = \mathbb{E}\left[\sum_{\ell=1}^{MK} \mathcal{P}_{\ell k}^{(t)}(\mathbf{x}) \left(\frac{\sigma^2 - \widehat{\sigma}_\ell^2 - (\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}_\ell)^\top \mathbf{x}\mathbf{x}^\top(\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}_\ell)}{2\sigma^4}\right)\right].$$

Replacing the expectation operator by the empirical expectation and setting $\frac{\partial \mathcal{K}}{\partial \boldsymbol{\beta}}$ to $\mathbf{0}$ one gets

$$\frac{1}{S}\sum_{s=1}^{S}\sum_{\ell=1}^{MK} \mathcal{P}_{\ell k}^{(t)}(\mathbf{x}_s)\mathbf{x}_s\mathbf{x}_s^\top\boldsymbol{\beta} = \frac{1}{S}\sum_{s=1}^{S}\sum_{\ell=1}^{MK} \mathcal{P}_{\ell k}^{(t)}(\mathbf{x}_s)\mathbf{x}_s\mathbf{x}_s^\top\widehat{\boldsymbol{\beta}}_\ell$$

$$\Leftrightarrow \quad \sum_{s=1}^{S}\mathbf{x}_s\left(\sum_{\ell=1}^{MK} \mathcal{P}_{\ell k}^{(t)}(\mathbf{x}_s)\right)\mathbf{x}_s^\top\boldsymbol{\beta} = \sum_{s=1}^{S}\mathbf{x}_s\mathbf{x}_s^\top\sum_{\ell=1}^{MK} \mathcal{P}_{\ell k}^{(t)}(\mathbf{x}_s)\widehat{\boldsymbol{\beta}}_\ell.$$

Let us denote

$$\widehat{\boldsymbol{B}} = \left[\widehat{\boldsymbol{\beta}}_1, \ldots, \widehat{\boldsymbol{\beta}}_{MK}\right] \in \mathbb{R}^{p\times MK}, \tag{C.6}$$

$$\boldsymbol{D}_k^{(t)} = \operatorname{diag}\left(\sum_{\ell=1}^{MK} \mathcal{P}_{\ell k}^{(t)}(\mathbf{x}_1), \ldots, \sum_{\ell=1}^{MK} \mathcal{P}_{\ell k}^{(t)}(\mathbf{x}_S)\right) \in \mathbb{U}^{S\times S}, \tag{C.7}$$

$$\boldsymbol{W}_k^{(t)} = \begin{bmatrix} \mathcal{P}_{1k}^{(t)}(\mathbf{x}_1) & \mathcal{P}_{1k}^{(t)}(\mathbf{x}_2) & \ldots & \mathcal{P}_{1k}^{(t)}(\mathbf{x}_S) \\ \vdots & \vdots & \ddots & \\ \mathcal{P}_{MK,k}^{(t)}(\mathbf{x}_1) & \mathcal{P}_{MK,k}^{(t)}(\mathbf{x}_2) & \ldots & \mathcal{P}_{MK,k}^{(t)}(\mathbf{x}_S) \end{bmatrix} \in \mathbb{U}^{MK\times S}, \tag{C.8}$$

where $\mathbb{U} = [0, 1] \subset \mathbb{R}$. Then the above equation can be written under matrix form as

$$\boldsymbol{X}_S^\top \boldsymbol{D}_k^{(t)} \boldsymbol{X}_S \boldsymbol{\beta} = \boldsymbol{X}_S^\top \boldsymbol{X}_S \odot (\boldsymbol{W}_k^\top \widehat{\boldsymbol{B}}^\top) \mathbb{1}_p,$$

where $\odot$ denotes the Hadamard product. This follows the root for $\boldsymbol{\beta}$ is given by

$$\boldsymbol{\beta} = (\boldsymbol{X}_S^\top \boldsymbol{D}_k^{(t)} \boldsymbol{X}_S)^{-1} \boldsymbol{X}_S^\top \boldsymbol{X}_S \odot (\boldsymbol{W}_k^{(t)\top} \widehat{\boldsymbol{B}}^\top) \mathbb{1}_p.$$

Similarly, setting $\frac{\partial \mathcal{K}}{\partial \sigma^2}$ to 0 one gets

$$\sum_{s=1}^{S} \sum_{\ell=1}^{MK} \mathcal{P}_{\ell k}^{(t)}(\mathbf{x}_s) \sigma^2 = \sum_{s=1}^{S} \sum_{\ell=1}^{MK} \mathcal{P}_{\ell k}^{(t)}(\mathbf{x}_s) \left(\widehat{\sigma}_\ell^2 + (\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}_\ell)^\top \mathbf{x}\mathbf{x}^\top (\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}_\ell) \right),$$

which can be written under matrix form as

$$\operatorname{trace}(\boldsymbol{D}_k^{(t)}) \sigma^2 = \widehat{\boldsymbol{\Sigma}}^\top \boldsymbol{W}_k^{(t)} \mathbb{1}_S + \mathbb{1}_{MK}^\top \boldsymbol{W}_k^{(t)} \odot (\boldsymbol{\beta} - \widehat{\boldsymbol{B}})^\top \boldsymbol{X}_S^\top \boldsymbol{X}_S (\boldsymbol{\beta} - \widehat{\boldsymbol{B}}) \mathbb{1}_{MK},$$

where $\widehat{\boldsymbol{\Sigma}} = (\widehat{\sigma}_1^2, \ldots, \widehat{\sigma}_{MK}^2)^\top \in \mathbb{R}_+^{MK}$. The conclusion is followed immediately. □

C.1.6 Proof of Proposition 5.4.3

To see the update formula for the parameter vector of the logistic regression expert, it is necessary to minimize the function $\mathcal{K}_{Bi}(\boldsymbol{\beta})$ given in (5.25). The first order derivative of $\mathcal{K}_{Bi}(\boldsymbol{\beta})$ is given by

$$\frac{\partial \mathcal{K}}{\partial \boldsymbol{\beta}} = \mathbb{E}\left[\sum_{\ell=1}^{MK} \mathcal{P}_{\ell k}^{(t)}(\mathbf{x}) \left(\frac{\mathbf{x} \exp(\mathbf{x}^\top \boldsymbol{\beta})}{1 + \exp(\mathbf{x}^\top \boldsymbol{\beta})} - \frac{\mathbf{x} \exp(\mathbf{x}^\top \widehat{\boldsymbol{\beta}}_\ell)}{1 + \exp(\mathbf{x}^\top \widehat{\boldsymbol{\beta}}_\ell)} \right) \right].$$

Replacing the expectation operator by the empirical expectation and setting $\frac{\partial \mathcal{K}}{\partial \boldsymbol{\beta}}$ to $\mathbf{0}$ we obtain the following equation

$$\sum_{s=1}^{S} \sum_{\ell=1}^{MK} \mathcal{P}_{\ell k}^{(t)}(\mathbf{x}_s) \frac{\mathbf{x}_s \exp(\mathbf{x}_s^\top \boldsymbol{\beta})}{1 + \exp(\mathbf{x}_s^\top \boldsymbol{\beta})} = \sum_{s=1}^{S} \sum_{\ell=1}^{MK} \mathcal{P}_{\ell k}^{(t)}(\mathbf{x}_s) \frac{\mathbf{x}_s \exp(\mathbf{x}_s^\top \widehat{\boldsymbol{\beta}}_\ell)}{1 + \exp(\mathbf{x}_s^\top \widehat{\boldsymbol{\beta}}_\ell)}. \tag{C.9}$$

Let us denote

$$\boldsymbol{E} = \left[\frac{\exp(\mathbf{x}_1^\top \boldsymbol{\beta})}{1 + \exp(\mathbf{x}_1^\top \boldsymbol{\beta})}, \ldots, \frac{\exp(\mathbf{x}_S^\top \boldsymbol{\beta})}{1 + \exp(\mathbf{x}_S^\top \boldsymbol{\beta})} \right]^\top \in \mathbb{U}^S, \tag{C.10}$$

$$\boldsymbol{U} = \begin{bmatrix} \frac{\exp(\mathbf{x}_1^\top \widehat{\boldsymbol{\beta}}_1)}{1+\exp(\mathbf{x}_1^\top \widehat{\boldsymbol{\beta}}_1)} & \frac{\exp(\mathbf{x}_1^\top \widehat{\boldsymbol{\beta}}_2)}{1+\exp(\mathbf{x}_1^\top \widehat{\boldsymbol{\beta}}_2)} & \cdots & \frac{\exp(\mathbf{x}_1^\top \widehat{\boldsymbol{\beta}}_{MK})}{1+\exp(\mathbf{x}_1^\top \widehat{\boldsymbol{\beta}}_{MK})} \\ \vdots & \vdots & \ddots & \\ \frac{\exp(\mathbf{x}_S^\top \widehat{\boldsymbol{\beta}}_1)}{1+\exp(\mathbf{x}_S^\top \widehat{\boldsymbol{\beta}}_1)} & \frac{\exp(\mathbf{x}_S^\top \widehat{\boldsymbol{\beta}}_2)}{1+\exp(\mathbf{x}_S^\top \widehat{\boldsymbol{\beta}}_2)} & \cdots & \frac{\exp(\mathbf{x}_S^\top \widehat{\boldsymbol{\beta}}_{MK})}{1+\exp(\mathbf{x}_S^\top \widehat{\boldsymbol{\beta}}_{MK})} \end{bmatrix} \in \mathbb{U}^{S \times MK}. \tag{C.11}$$

Then the equation (C.9) can be rewritten under matrix form as

$$\boldsymbol{X}_S^\top \boldsymbol{D}_k^{(t)} \boldsymbol{E} \mathbb{1}_S = \boldsymbol{X}_S^\top \boldsymbol{W}_k^{(t)\top} \boldsymbol{U}^\top \mathbb{1}_S.$$

We see that, the above equation holds if $\boldsymbol{E} = {\boldsymbol{D}_k^{(t)}}^{-1} \boldsymbol{W}_k^{(t)\top} \boldsymbol{U}^\top$, or equivalently, $\boldsymbol{X}_S \boldsymbol{\beta} = \log(\boldsymbol{V}_k^{(t)}) - \log(1 - \boldsymbol{V}_k^{(t)})$ with $\boldsymbol{V}_k := {\boldsymbol{D}_k^{(t)}}^{-1} \boldsymbol{W}_k^{(t)\top} \boldsymbol{U}^\top$. Therefore, we obtain $\boldsymbol{\beta} = (\boldsymbol{X}_S^\top \boldsymbol{X}_S)^{-1} \boldsymbol{X}_S^\top (\log(\boldsymbol{V}_k^{(t)}) - \log(1 - \boldsymbol{V}_k^{(t)}))$.

List of Publications and Communications

Papers and communications

- Chamroukhi F., Pham, T. N., Hoang, V. H., and McLachlan, G. J. (2022). Functional-mixtures-of-experts. Under revision, *Journal of Statistics and Computing*. *ArXiv preprint arXiv:2202.02249*. https://arxiv.org/pdf/2202.02249.pdf.
- Chamroukhi F., Pham, T. N. (2022). Distributed learning approach for mixture-of-experts models. To be submitted to a journal.
- Chamroukhi F., Pham, T. N., Hoang, V. H., and McLachlan, G. J. (2021). Functional-mixtures-of-experts with functional predictors. In *The 14th International Conference CMStatistics 2021*, London, UK.
- Pham, T. N., Chamroukhi F. (2022). Functional-mixtures-of-experts for classification. *ArXiv preprint arXiv:2202.13934*. https://arxiv.org/pdf/2202.13934.pdf.
- Pham, T. N. (July 2022). Functional-mixtures of expert model for classification. Talk at The 53èmes Journées de Statistique.
- Pham, T. N. (September 20). Mixtures of Experts with functional predictors and applications. Poster at Journée Thématique Rouen INSA.
- Pham, T. N. (November 2019). Quantile regression in large-scale data sets. Talk at LMNO Lab.

Other scientific activities

- Participation to "Working Group on Model-Based Clustering Summer Session", Vienne, 14-20 July 2019.
- Research visit at LIPN, University Paris-Nord, October-November 2020.
- Participation to "Conference: Math & IA", 9 March 2021.
- Participation to "MiMo 2021: Workshop on Mixture Models", Rouen, 8-9 April 2021.
- Participation to conference MHC2021, 2-4 June 2021
- Participation to "Third Conference on Statistics and Data Science", 28-30 October 2021.

Bibliography

Abraham, C., Cornillon, P., Matzner-Løber, E., and Molinari, N. (2003). Unsupervised curve clustering using b-splines. *Scandinavian Journal of Statistics*, 30:581–595.

aki Sato, M. and Ishii, S. (2000). On-line em algorithm for the normalized gaussian network. *Neural Computation*, 12:407–432.

Banfield, J. D. and Raftery, A. E. (1993). Model-based gaussian and non-gaussian clustering. *Biometrics*, 49:803–821.

Berlinet, A., Biau, G., and Rouviere, L. (2008). Functional supervised classification with wavelets. In *Annales de l'ISUP*, volume 52, page 19.

Beyaztas, U. and Shang, H. L. (2020). On function-on-function regression: Partial least squares approach. *Environmental and ecological statistics*, 27(1):95–114.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer Verlag, U K.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.

Candes, E., Tao, T., et al. (2007). The dantzig selector: Statistical estimation when p is much larger than n. *Annals of statistics*, 35(6):2313–2351.

Cao, L. (2002). Support vector machines experts for time series forecasting 3 lijuan cao. In *inproceedings*.

Celeux, G. and Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern Recognition*, 28(5):781–793.

Chamroukhi, F. (2015). Non-normal mixtures of experts. *ArXiv*, abs/1506.06707.

Chamroukhi, F. (2017). Skew t mixture of experts. *Neurocomputing - Elsevier*, 266:390–408.

Chamroukhi, F. and Huynh, B. T. (2019). Regularized Maximum Likelihood Estimation and Feature Selection in Mixtures-of-Experts Models. *Journal de la Société Française de Statistique*, 160(1):57–85.

Chamroukhi, F., Lecocq, F., and Nguyen, H. D. (2019). Regularized estimation and feature selection in mixtures of gaussian-gated experts models. In *Research School on Statistics and Data Science*,

pages 42–56. Springer.

Chamroukhi, F. and Nguyen, H. D. (2019). Model-based clustering and classification of functional data. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 9(4):e1298.

Chamroukhi, F., Pham, T. N., Hoang, V. H., and McLachlan, G. J. (2022). Functional mixtures-of-experts. *ArXiv preprint arXiv:2202.02249*.

Chamroukhi, F., Samé, A., Govaert, G., and Aknin, P. (2009). Time series modeling by a regression approach based on a latent process. *Neural Networks*, 22(5-6):593–602.

Chamroukhi, F., Trabelsi, D., Mohammed, S., Oukhellou, L., and Amirat, Y. (2013). Joint segmentation of multivariate time series with hidden process regression for human activity recognition. *Neurocomputing*, 120:633–644.

Chang, C., Chen, Y., and Ogden, R. (2014). Functional data classification: a wavelet approach. *Computational Statistics*, 29:1497–1513.

Chen, K., Xu, L., and Chi, H. (1999). Improved learning algorithms for mixture of experts in multiclass classification. *Neural Networks*, 12(9):1229–1252.

Chen, X. and ge Xie, M. (2014). A split-and-conquer approach for analysis of extraordinarily large data. *Statistica Sinica*, 24:1655–1684.

chen Chen, W., Ostrouchov, G., Pugmire, D., Prabhat, and Wehner, M. (2013). A parallel em algorithm for model-based clustering applied to the exploration of large spatio-temporal data. *Technometrics*, 55(4):513–523.

Chiou, J.-M., Müller, H.-G., and Wang, J.-L. (2004). Functional response models. *Statistica Sinica*, 14(3):675–693.

Ciarleglio, A. and Ogden, R. T. (2016). Wavelet-based scalar-on-function finite mixture regression models. *Computational Statistics & Data Analysis*, 93:86–96.

Cover, T. M. and T., J. A. (2006). *Elements of Information Theory*. Wiley Interscience, second edition edition.

Delaigle, A. and Hall, P. (2010). Defining probability density for a distribution of random functions. *Annals of Statistics*, 38:1171–1193.

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

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

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of The Royal Statistical Society, B*, 39(1):1–38.

Devijver, E. (2015). *High-dimensional mixture regression models, application to functional data*. Ph.D. thesis, Université Paris-Sud.

Devijver, E. (2017). Model-based clustering for high-dimensional data. Application to functional data. *Advances in Data Analysis and Classification*, 11:243–279.

Deville (1974). Méthodes statistiques et numériques de l'analyse harmonique. In *Annales de l'INSEE*, volume 15, pages 3–101.

Dragomir, S. (2013). A generalization of f-divergence measure to convex functions defined on linear spaces. *Communications in Mathematical Analysis*, 15.

Faraway, J. J. (1997). Regression analysis for a functional response. *Technometrics*, 39(3):254–261.

Ferraty, F. and Vieu, P. (2003). Curves discrimination: a nonparametric functional approach. *Computational Statistics & Data Analysis*, 44(1):161–173. Special Issue in Honour of Stan Azen: a Birthday Celebration.

Ferraty, F. and Vieu, P. (2006). *Nonparametric functional data analysis : theory and practice*. Springer series in statistics.

Fletcher, R. (1987). Practical methods of optimization. *Pattern Recognition Letters*, 18:859–872.

Gaines, B. R., Kim, J., and Zhou, H. (2018). Algorithms for fitting the constrained lasso. *Journal of Computational and Graphical Statistics*, 27(4):861–871. PMID: 30618485.

Giacofci, M., Lambert-Lacroix, S., Marot, G., and Picard, F. (2013). Wavelet-based clustering for mixed-effects functional models in high dimension. *Biometrics*, 69.

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., Bobb, J., Crainiceanu, C. M., Caffo, B., and Reich, D. (2011b). 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.

Gormley, I. C. and Murphy, T. B. (2010). A mixture of experts latent position cluster model for social network data. *Statistical Methodology*, 7:385–405.

Grenander, U. (1950). Stochastic processes and statistical inference. *Arkiv för Matematik*, 1(3):195–277.

Gutta, S., Huang, J. R., Phillips, P. J., and Wechsler, H. (2000). Mixture of experts for classification of gender, ethnic origin, and pose of human faces. *IEEE transactions on neural networks*, 11 4:948–60.

Hall, P., Poskitt, D. S., and Presnell, B. (2001). A functional data—analytic approach to signal discrimination. *Technometrics*, 43(1):1–9.

Hartigan, J. (1975). *Clustering Algorithms*. Out-of-print Books on demand. Wiley.

Hartigan, J. A. and Wong, M. A. (1979). A K-means clustering algorithm. *Applied Statistics*,

28:100–108.

Hastie, T., Buja, A., and Tibshirani, R. (1995). Penalized Discriminant Analysis. *Annals of Statistics*, 23:73–102.

Hastie, T., Tibshirani, R., and Wainwright, M. (2015). *Statistical learning with sparsity: the lasso and generalizations*. Chapman and Hall/CRC.

Henderson, H. V. and Searle, S. R. (1979). Vec and vech operators for matrices, with some uses in jacobians and multivariate statistics. *Canadian Journal of Statistics*, 7(1):65–81.

Huynh, T. and Chamroukhi, F. (2019a). Estimation and feature selection in mixtures of generalized linear experts models. *arXiv*.

Huynh, T. and Chamroukhi, F. (2019b). Estimation and feature selection in mixtures of generalized linear experts models. *arXiv*.

Ieva, F., Paganoni, A. M., Pigoli, D., and Vitelli, V. (2013). Multivariate functional clustering for the morphological analysis of electrocardiograph curves. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 62.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., and Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87.

Jacques, J. and Preda, C. (2013a). Funclust: A curves clustering method using functional random variables density approximation. *Neurocomputing*, 112:164–171.

Jacques, J. and Preda, C. (2013b). Functional data clustering: A survey. *Advances in Data Analysis and Classification*, 8:231–255.

Jacques, J. and Preda, C. (2014). Model-based clustering for multivariate functional data. *Computational Statistics & Data Analysis*, 71:92–106.

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

James, G. M. and Hastie, T. J. (2001a). Functional linear discriminant analysis for irregularly sampled curves. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(3):533–550.

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

James, G. M. and Sugar, C. (2003). Clustering for sparsely sampled functional data. *Journal of the American Statistical Association*, 98(462).

James, G. M., Wang, J., and Zhu, J. (2009). Functional linear regression that's interpretable. *Annals of Statistics*, 37(5A):2083–2108.

Jeon, Y. and Hwang, G. (2022). Bayesian mixture of gaussian processes for data association problem. *Pattern Recognition*, 127:108592.

Jiang, W. and Tanner, M. (1999). On the identifiability of mixtures-of-experts. *Neural Networks*, 12(9):1253–1258.

Jordan, M. I. and Jacobs, R. A. (1994). Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6:181–214.

Karhunen, K. (1946). *Zur Spektraltheorie stochastischer Prozesse*, volume 7 of *Annales Academiae scientiarum Fennicae. Series A. 1, Mathematica-physica*. Annales Academiae Scientiarum Fennicae.

Kohonen, T. and Schroeder, M. (1995). *Self-Organizing Maps*. Springer-Verlag.

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

Lange, K. (2004). The mm algorithm. In *Optimization*, pages 119–136. Springer.

Lange, K. (2016). *MM optimization algorithms*. SIAM.

Lima, C. A., Coelho, A. L., and Von Zuben, F. J. (2007). Hybridizing mixtures of experts with support vector machines: Investigation into nonlinear dynamic systems identification. *Information Sciences*, 177(10):2049–2074. Including Special Issue on Hybrid Intelligent Systems.

Liu, X. and Yang, M. (2009). Simultaneous curve registration and clustering for functional data. *Computational Statistics and Data Analysis*, 53(4):1361–1376.

Loève, M. (1946). Fonctions aléatoires à décomposition orthogonale exponentielle. *La Revue Scientifique*, 84.

Mak, M.-W. and Kung, S.-Y. (2000). Estimation of elliptical basis function parameters by the em algorithm with application to speaker verification. *IEEE Transactions on Neural Networks*, 11(4):961–969.

Matsui, H., Kawano, S., and Konishi, S. (2013). Regularized functional regression modeling for functional response and predictors. *Journal of Math-for-Industry*, 1:17–25.

McLachlan, G. J. and Krishnan, T. (2008). *The EM algorithm and extensions*. New York: Wiley, second edition.

McLachlan, G. J. and Peel., D. (2000). *Finite mixture models*. New York: Wiley.

Merugu, S. and Ghosh, J. (2005). A distributed learning framework for heterogeneous data sources. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 208–217.

Mingxian, X., Miller, J. J., and Wegman, E. J. (1991). Parallelizing multiple linear regression for speed and redundancy: an empirical study. *Journal of Statistical Computation and Simulation*, 39(4):205–214.

Mousavi, S. and Sørensen, H. (2017). Multinomial functional regression with wavelets and lasso penalization. *Econometrics and Statistics*, 150-166.

Mousavi, S. N. and Sørensen, H. (2018). Functional logistic regression: a comparison of three methods. *Journal of Statistical Computation and Simulation*, 88(2):250–268.

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

Müller, H.-G. (2016). Functional data analysis and random objects. *The Annals of Statistics*, 44(5):1867–1887.

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

Müller, H.-G. and Yao, F. (2008). Functional additive models. *Journal of the American Statistical Association*, 103(484):1534–1544.

Ng, S.-K. and McLachlan, G. J. (2004). Using the em algorithm to train neural networks: misconceptions and a new algorithm for multiclass classification. *IEEE Transactions on Neural Networks*, 15(3):738–749.

Nguyen, H. D. (2016). An introduction to mm algorithms for machine learning and statistical estimation. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 7.

Nguyen, H. D. and Chamroukhi, F. (2018). Practical and theoretical aspects of mixture-of-experts modeling: An overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, pages e1246–n/a.

Nguyen, H. D., Chamroukhi, F., and Forbes, F. (2019). Approximation results regarding the multiple-output Gaussian gated mixture of linear experts model. *Neurocomputing*, 366:208–214.

Nguyen, H. D., Lloyd-Jones, L. R., and McLachlan, G. J. (2016). A universal approximation theorem for mixture-of-experts models. *Neural Computation*, 28(12):2585–2593.

Nguyen, H. D., Nguyen, T., Chamroukhi, F., and McLachlan, G. J. (2021a). Approximations of conditional probability density functions in Lebesgue spaces via mixture of experts models. *Journal of Statistical Distributions and Applications*, 8(1):13.

Nguyen, T., Nguyen, H. D., Chamroukhi, F., and Forbes, F. (2021b). A non-asymptotic penalization criterion for model selection in mixture of experts models. *arXiv preprint arXiv:2104.02640*.

Nguyen, T., Nguyen, H. D., Chamroukhi, F., and McLachlan, G. J. (2020). An l_1-oracle inequality for the lasso in mixture-of-experts regression models. *arXiv preprint arXiv:2009.10622*.

Nowak, R. (2003). Distributed em algorithms for density estimation and clustering in sensor networks. *IEEE Transactions on Signal Processing*, 51(8):2245–2253.

Oberman, A. M. and Ruan, Y. (2015). An efficient linear programming method for optimal transportation. *arXiv: Numerical Analysis*.

Oyeleye, M., Chen, T., Titarenko, S., and Antoniou, G. (2022). A predictive analysis of heart rates

using machine learning techniques. *International Journal of Environmental Research and Public Health*, 19(4).

Peng, J. and Müller, H.-G. (2008). Distance-based clustering of sparsely observed stochastic processes, with applications to online auctions. *The Annals of Applied Statistics*, 2(3):1056–1077.

Perthame, É., Forbes, F., Olivier, B., and Deleforge, A. (2016). Non linear robust regression in high dimension. In *inproceedings*.

Qiao, X., Guo, S., and James, G. M. (2019). Functional graphical models. *Journal of the American Statistical Association*, 114(525):211–222.

Ramamurti, V. and Ghosh, J. (1996). Advances in using hierarchical mixture of experts for signal classification. In *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, volume 6, pages 3569–3572 vol. 6.

Ramsay, J., Bock, R., and Gasser, T. (1995). Comparison of height acceleration curves in the fels, zurich, and berkeley growth data. *Annals of Human Biology*, 22(5):413–426.

Ramsay, J., Ramsay, T., and Sangalli, L. (2011). Spatial functional data analysis. In Ferraty, F., editor, *Recent Advances in Functional Data Analysis and Related Topics*, pages 269–275. Springer.

Ramsay, J. O. and Silverman, B. W. (1997). *Functional Data Analysis*. Springer Series in Statistics. Springer.

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

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

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.

Rossi, A., Da Pozzo, E., Menicagli, D., Tremolanti, C., Priami, C., Sîrbu, A., Clifton, D. A., Martini, C., and Morelli, D. (2020). A public dataset of 24-h multi-levels psycho-physiological responses in young healthy adults. *Data*, 5(4).

Rossi, F., Conan-Guez, B., and Golli, A. (2004). Clustering functional data with the som algorithm. *Proceedings of ESANN 2004*, page 305–312.

Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464.

Shimokawa, M., Mizuta, M., and Sato, Y. (2000). An expansion of functional regression analysis (in japanese). *Ouyou toukeigaku*, 29(1):27–39.

Shofiyah, F. and Sofro, A. (2018). Split and conquer method in penalized logistic regression with lasso (application on credit scoring data). *Journal of Physics: Conference Series*, 1108:012107.

Städler, N., Bühlmann, P., and Van De Geer, S. (2010). ℓ1-penalization for mixture regression models. *Test*, 19(2):209–256.

Tarpey, T. and Kinateder, K. K. J. (2003). Clustering functional data. *Journal of Classification*, 20:093–114.

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

Titterington, D., Smith, A., and Makov, U. (1985). *Statistical Analysis of Finite Mixture Distributions*. John Wiley & Sons.

Tresp, V. (2001). Mixtures of gaussian processes. *Advances in Neural Information Processing Systems*, 13.

Tuddenham, R. D. and Snyder, M. (1954). Physical growth of california boys and girls from birth to eighteen years. *Publications in child development. University of California, Berkeley*, 1 2:183–364.

Ullah, S. and Finch, C. (2013). Applications of functional data analysis: A systematic review. *BMC medical research methodology*, 13:43.

Wu, C. F. J. (1983). On the convergence properties of the em algorithm. *The Annals of Statistics*, 11(1):95–103.

Xu, L., Jordan, M., and Hinton, G. E. (1994a). An alternative model for mixtures of experts. *Advances in neural information processing systems*, 7:633–640.

Xu, L., Jordan, M. I., and Hinton, G. E. (1994b). An alternative model for mixtures of experts. In *NIPS*.

Yamanishi, Y. and Tanaka, Y. (2003). 8. functional data analysis geographically weighted functional multiple regression analysis: A numerical investigation. *Journal of the Japanese Society of Computational Statistics*, 15(2):307–317.

Yang, Y. and Ma, J. (2011). An efficient em approach to parameter learning of the mixture of gaussian processes. In *Proceedings of the 8th International Conference on Advances in Neural Networks - Volume Part II*, ISNN'11, page 165–174, Berlin, Heidelberg. Springer-Verlag.

Yao, F., Fu, Y., and Lee, T. C. M. (2010). Functional mixture regression. *Biostatistics*, 12(2):341–353.

Yao, F. and Müller, H.-G. (2010). Functional quadratic regression. *Biometrika*, 97(1):49–64.

Yuksel, S. E., Wilson, J. N., and Gader, P. D. (2012). Twenty years of mixture of experts. *IEEE Trans. Neural Netw. Learning Syst.*, 23(8):1177–1193.

Yümlü, M. S., Gürgen, F. S., and Okay, N. (2003). Financial time series prediction using mixture of experts. In *ISCIS*.

Zeevi, A. J., Meir, R., and Adler, R. J. (1996). Time series prediction using mixtures of experts.

In *Proceedings of the 9th International Conference on Neural Information Processing Systems*, NIPS'96, page 309–315, Cambridge, MA, USA. MIT Press.

Zhang, Q. and Chen, J. (2021). Distributed learning of finite gaussian mixtures. *Proceedings of the 3rd International Conference on Statistics: Theory and Applications*.

Zhao, W., Ma, H., and He, Q. (2009). Parallel k-means clustering based on mapreduce. In Jaatun, M. G., Zhao, G., and Rong, C., editors, *Cloud Computing*, pages 674–679, Berlin, Heidelberg. Springer Berlin Heidelberg.

Zinkevich, M., Weimer, M., Li, L., and Smola, A. (2010). Parallelized stochastic gradient descent. In Lafferty, J., Williams, C., Shawe-Taylor, J., Zemel, R., and Culotta, A., editors, *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc.

Élodie Brunel, Mas, A., and Roche, A. (2016). Non-asymptotic adaptive prediction in functional linear models. *Journal of Multivariate Analysis*, 143:208–232.

List of Figures

List of Tables

List of Algorithms

Modeling and Learning with Mixtures of Experts for Functional Data and Distributed Data

Abstract: Mixture of experts (ME) models are popular in statistics and machine learning and have been studied for high-dimensional vectorial data and that are centralized. However, in many problems, we observe time series data and applying the existing ME models directly to these raw data may limit the performance because they ignore the correlation between variables, an intrinsic nature of functional data, and therefore do not adequately capture the inherent functional structure of the data. In many applications the data may also be not available at a centralized mode, and there is therefore a need to develop adapted distributed strategies allowing for efficient parallel computations. This thesis studies mixture of experts (ME) models in functional data and large-scale problems, in a heterogeneous scenario. The main objectives are *i)* to deal with situations in which we are given functional predictors (e.g., time series) to predict a potentially functional response, and *ii)* to learn from distributed data. We first propose in this thesis a new family of ME models, called functional mixture of experts (FME), which includes scalar-on-function FME, function-on-scalar FME, and function-on-function FME. We introduce regularized model estimation approaches via appropriate regularizations that encourage sparse and smooth estimates, while being interpretable. We develop efficient EM algorithms to maximize the corresponding observed-data log-likelihood functions and conduct extensive experimental studies to highlight the performance of the proposed models and algorithms. Then, to scale-up the ME estimation to a potentially distributed data, we develop a distributed learning strategy of ME models. It performs standard inference on local machines to obtain local estimators, then transmits them to a central machine where they are aggregated. Based on minimizing a proposed expected transportation divergence, the local estimators are aggregated to obtain a reduced estimator that is consistent with the global one, *i.e.*, the estimator that could be possibly constructed upon the full dataset. Experimental studies demonstrate the performance of our approach.

Keywords: Mixture of Experts; Functional Data Analysis; Regularized Estimation; LASSO; EM algorithm; MM Algorithm; Distributed Learning; Optimal Transport; Prediction; Clustering.

***

Modélisation et Apprentissage avec des Mélanges d'Experts pour des Données Fonctionnelles et des Données Distribuées

Résumé: Les modèles de mélange d'experts (ME) sont populaires en statistique et en apprentissage automatique et ont été étudiés pour des données vectorielles de haute dimension et qui sont centralisées. Cependant, dans de nombreux problèmes, nous observons des données de séries temporelles et l'application des modèles ME existants directement à ces données brutes peut limiter les performances car ils ignorent la corrélation entre les variables, une prropriété intrinsèque à ces données fonctionnelles, et ne capturent donc pas adéquatement la structure fonctionnelle inhérente à ces données. Dans de nombreuses applications, les données peuvent également ne pas être disponibles en mode centralisé, et il est donc nécessaire de développer des stratégies distribuées adaptées permettant des calculs parallèles efficaces. Cette thèse étudie les modèles de mélange d'experts (ME) pour les données fonctionnelles et les problèmes à grande échelle, dans un scénario hétérogène. Les objectifs principaux sont *i)* d'adresser des situations dans lesquelles on cherche à prédire une réponse potentiellement fonctionnelle à partir d'un prédicteur fonctionnel (par exemple, des séries temporelles), et *ii)* d'apprendre à partir de données distribuées. Dans cette thèse, nous proposons premièrement une nouvelle famille de modèles ME, appelée mélanges d'experts fonctionnels (FME), qui inclut le FME scalaire-sur-fonction, le FME fonction-sur-scalaire, et le FME fonction-sur-fonction. Nous introduisons des approches d'estimation régularisée via des régularisations appropriées qui encouragent des solutions parsimonieuses et des modèles lisses, tout en étant interprétables. Nous développons des algorithmes EM efficaces pour maximiser les fonctions de log-vraisemblance pour des données observées et menons des études expérimentales approfondies pour mettre en évidence les performances des modèles et algorithmes proposés. Ensuite, pour étendre l'estimation des mélanges d'experts ME au cas des données potentiellement distribuées, nous développons une stratégie d'apprentissage distribué des modèles ME. Elle effectue une inférence standard sur des machines locales pour obtenir des estimateurs locaux qui sont agrégés au niveau central. En se basant sur la minimisation d'une divergence de transport espérée proposée, les estimateurs locaux sont agrégés pour obtenir un estimateur réduit qui est consistant avec l'estimateur global, *i.e.*, l'estimateur qui pourrait éventuellement être construit à partir de l'ensemble complet de données dans la version non-distribuée. Des études expérimentales démontrent la performance de notre approche.

Mots-clés: Mélanges d'Experts; Analyse de Données Fonctionnelles; Techniques de régularisation; LASSO; Algorithmes EM; Algorithmes MM; Apprentissage Distribué; Transport Optimal; Prédiction; Clustering.