12-2016

Group transformation and identification with kernel methods and big data mixed logistic regression

Chao Pan
Purdue University

Graduate School Form
30 Updated 12/26/2015

PURDUE UNIVERSITY
GRADUATE SCHOOL
Thesis/Dissertation Acceptance

This is to certify that the thesis/dissertation prepared

By Chao Pan

Entitled

GROUP TRANSFORMATION AND IDENTIFICATION WITH KERNEL METHODS AND BIG DATA MIXED LOGISTIC REGRESSION

For the degree of Doctor of Philosophy

Is approved by the final examining committee:

Michael Y. Zhu
Chair

Mark D. Ward

Michael Levine

Rebecca W. Doerge

To the best of my knowledge and as understood by the student in the Thesis/Dissertation Agreement, Publication Delay, and Certification Disclaimer (Graduate School Form 32), this thesis/dissertation adheres to the provisions of Purdue University's "Policy of Integrity in Research" and the use of copyright material.

Approved by Major Professor(s): Michael Y. Zhu

Approved by: Jun Xie 9/15/2016

Head of the Departmental Graduate Program Date

GROUP TRANSFORMATION AND IDENTIFICATION WITH KERNEL METHODS AND BIG DATA MIXED LOGISTIC REGRESSION

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Chao Pan

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

December 2016

Purdue University

West Lafayette, Indiana

To Zhu Qun

ACKNOWLEDGMENTS

I would like to express the deepest appreciation to my advisor, Professor Michael Yu Zhu. He continually and convincingly conveyed a spirit of adventure in regard to research and scholarship, and an excellency in regard of mentoring. Without his guidance and persistent help this dissertation would not have been possible.

I would like to thank my committee members, Professor Rebecca W. Doerge, Professor Michael Levine and Professor Mark Daniel Ward, who demonstrated to me that hard working and a proactive attitude are crucial for a successful research career.

A special thank you to Professor Jun Xie, who provided me much guidance during the time of struggle when I first started at Purdue and when I decided to change my research direction.

In addition, I am grateful for the help of many others in the department. Especially, I thank Professor Hao Zhang for his leadership, and Doug Crabill for his excellent work in providing computing resources to the department. I am grateful for the work of Marian Cannova, Nicole Cox, Linda Foster, Ce-Ce Furtner, Anna Hook, Aaron Kosdrosky, Shaun Ponder, Alicia Schragg, and Jesse Wallenfang for keeping the department running so smoothly.

TABLE OF CONTENTS

Page

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

ABSTRACT

Pan, Chao PhD, Purdue University, December 2016. Group Transformation and Identification with Kernel Methods and Big Data Mixed Logistic Regression. Major Professor: Michael Yu Zhu.

Exploratory Data Analysis (EDA) is a crucial step in the life cycle of data analysis. Exploring data with effective methods would reveal main characteristics of data and provides guidance for model building. The goal of this thesis is to develop effective and efficient methods for data exploration in the regression setting.

First, we propose to use optimal group transformations as a general approach for exploring the relationship between predictor variables $\boldsymbol{X}$ and the response Y. This approach can be considered an automatic procedure to identify the best characteristic of $P(Y|\boldsymbol{X})$ under which the relationship between Y and $\boldsymbol{X}$ can be fully explored. The emphasis on using group transformations allows the approach to recover true group structures among the predictors. We also develop kernel methods for estimating the optimal group transformations based on cross-covariance and conditional covariance operators. The statistical consistency of the estimates has been established. We refer to the proposed framework and approach as the Optimal Kernel Group Transformation (OKGT) method.

Secondly, we define the true additive group structure for OKGT when the response transformation is known, and further develop an effective penalized kernel regression method for its identification. The procedure uses a novel penalty we propose to control the complexity of additive group structures. This method is referred to as the Additive Group Structure Identification (AGSI). We also establish the selection consistency for AGSI.

Finally, we construct the Hierarchical Mixed Logistic Regression Model (HMLRM) and propose to use it for exploring heterogeneity in big data. By explicitly modeling the hidden layer, we individualize the calculation of the probability that a sample belongs to a subpopulation. While estimating the model parameters by EM algorithm, the separability of the parameter space is exploited. In order to apply HMLRM on big data, we design a distributed algorithm for model estimation which is implemented in Apache Spark.

1. INTRODUCTION

Exploratory Data Analysis (EDA) is a statistical approach to analyze data sets from different perspectives and summarize their main characteristics. In the life cycle of statistical analysis, EDA is usually performed after data collection and before statistical modelling. The goal of performing EDA is to form a first impression of data and to get an idea what can be done to data. It is stated in John Tukey's Exploratory Data Analysis [1] that "It is important to understand what you CAN DO before you learn to measure how WELL you seem to have DONE it".

In order to perform effective EDA, we need to rely on proper techniques and tools. One topic of this thesis is to show that the combination of additive model and kernel methods can be effectively used for data exploration.

Additive model and its generalized version (see [2], [3], [4]) are often used in non-parametric regression analysis. They are more general than linear models to explore nonlinearity in datasets when there is no or limited knowledge about data. With a simplified additive structure, (generalized) additive model is less affected by curse of dimensionality and hence its model estimation is more efficient. Because additive model applies one dimensional smoothers, it is more interpretable than the results obtained from more general non-parametric regression models. This is especially important for EDA since its purpose it to make sense of data for model building and statistical inference. However, it has to be admitted that by disregarding any possible interaction between variables, additive model may not be sufficient to fully explore data. In this thesis, we will extend (generalized) additive model by proposing the notion of *group structure* to accommodate low dimensional interactions.

Kernel methods have been popular over the last two decades and witnessed great achievement in both theory and applications (see [5], [6], [7], [8]). The fundamental idea of kernel methods is that instead of reducing data dimensionality, samples are

mapped to a higher dimensional space (sometimes even an infinite dimensional space) in the hope that some nonlinear features of data will emerge. In machine learning, this is called feature mapping and the space that data is mapped to is called feature space.

In kernel methods, a feature space is usually a Reproducing Kernel Hilbert Space (RKHS). The theory of RKHSs has been well developed (see [9]). One of the most important properties of RKHSs is the reproducing property, which is helpful to reformulate a statistical problem to facilitate its theoretical analysis. RKHSs are especially useful for empirical risk minimization because of the celebrated representer theorem, which shows that a function that minimize a regularized empirical risk over an RKHS can be represented as a linear combination of the kernel functions evaluated at the samples (see [10], [11], [12] for details). Thus by using kernel methods, model estimation in an infinite dimensional space can be formulated as a equivalent finite dimensional problem.

Kernel methods are flexible tools for data exploration. Many traditional statistical methods have their kernalized versions, for example, kenrel PCA (see [13]), kernel CCA (see [14]), kernel dimension reduction (see [15], [16]), and kernel test of independence (see [17]). So kernels can be combined with classical statistical methods to expand the horizon of data exploration. Since different RKHSs include functions possessing different properties, such as degree of smoothness and integrability, choosing an RKHS can be considered as leveraging some prior knowledge for analyzing data. Kernel methods can also unify heterogeneous types of data so that the same statistical method can be applied. This is because we have the freedom to design different kernels for different data types, such as string kernels for text classification (see [18], [19]), match kernels over image patches (see [20], [21]), and kernel function for clinical data (see [22]).

In Chapter 2, we propose the Optimal Kernel Group Transformation (OKGT) framework for effective data exploration when regression is used. OKGT combines the idea of additive structure with groups and kernel methods in a unified framework.

This overcomes the restriction of additive model and takes advantage of the flexibility of RKHSs. In Chapter 3, we answer the question of how to identify the additive group structure for OKGT when the response transformation is known. This is achieved by proposing a novel penalty which controls the group structure complexity in solving OKGT. Solving OKGT by using this novel penalty is called Additive Group Structure Identification (AGSI).

The second topic of this thesis is to develop and implement methods for exploring big data. The information and communication technology evolution has been driven by Moore's Law for the past half a century. We are now marching towards the era of the Internet of Things (IoT). Along with the wide adoption of mobile devices that are connected through communication networks are the huge amount of data. They are generated from the communication among devices, for example the browsing information your cell phone sent to Google's servers, and the interaction between human and the devices, for example the shopping records you left behind in Target's server. More importantly, the speed of data generation and the complexity of data structure is unparalleled in human history. How can we use this humongous amount of complex data that is growing at an accelerating speed?

When the size of data increases, we would expect data's structure becomes complex. So a "big" model is necessary for revealing complex structural information from data. As heterogeneity and anomaly are common in big data, being able to identify hierarchical and clustering structures and detect anomalies is fundamental for exploring big data. A model for big data exploration should be "big" enough to accomplish these tasks.

In Chapter 4, we propose using Hierarchical Mixed Logistic Regression Model (HMLRM) for exploring large data sets with categorical response. We also implemented this model on Apache Spark which is one of the most popular big data computing platforms these days.

2. OPTIMAL KERNEL GROUP TRANSFORMATION

Regression analysis is a statistical technique for studying the relationship between a response variable Y and a predictor vector $\boldsymbol{X}$ based on a sample of Y and $\boldsymbol{X}$. The relationship between Y and $\boldsymbol{X}$ can be fully characterized by the conditional distribution of Y given $\boldsymbol{X}$, which is denoted as $P(Y|\boldsymbol{X})$. Therefore, the general goal of regression analysis is to infer about $P(Y|\boldsymbol{X})$ as much as possible with the given sample, which we refer to as the exploratory regression analysis. However, many commonly used regression methods only focus on some features of $P(Y|\boldsymbol{X})$ instead of the full conditional distribution. For example, ordinary least squares regression analysis focuses on the conditional expectation $\mathbb{E}[Y|\boldsymbol{X}]$, and quantile regression analysis targets the conditional median or other quantiles of the response.

Regression methods that focus on particular features of $P(Y|\boldsymbol{X})$ suffer from some limitations. Firstly, the majority of those methods such as linear regression relies on strong model assumptions, and departure from the model assumptions may render those methods ineffective. Secondly, focusing only on the feature of interest while neglecting other aspects of $P(Y|\boldsymbol{X})$ may make the regression analysis inefficient. Thirdly, those methods cannot be used to fully explore and capture the dependence of Y on $\boldsymbol{X}$ in the conditional distribution $P(Y|\boldsymbol{X})$. For example, suppose $Y = 2X_1 + X_2\epsilon$, where (X_1, X_2) and ϵ are independent and ϵ has mean zero and variance one. Under this model, ordinary least squares regression analysis can only capture X_1, and the estimate of the coefficient of X_1 is not efficient.

There exists some effort to directly estimate the conditional distribution $P(Y|\boldsymbol{X})$ using nonparametric methods, which is commonly referred to as conditional density estimation in the literature. Rosenblatt [23] introduced conditional density estimation in 1969. For conditional density estimation, Fan et al. [24] proposed to use local polynomial regression in 1996, and recently Sujiyama et al. (2010) [25] proposed to

use least-squares density ratio estimation. Conditional density estimation may be useful for some specific application, but generally it is not practical or feasible especially when $\boldsymbol{X}$ is multidimensional. It is known that density estimation is challenging when the dimensionality of $\boldsymbol{X}$ is higher than five, and conditional density estimation can be even more difficult. To ensure sufficient accuracy of the density estimator, an extremely large number of data points is required (see [26]). Even when the conditional density can be accurately estimated as a function of Y and $\boldsymbol{X}$, the dependence of Y on $\boldsymbol{X}$ cannot be easily interpreted. Another approach that can potentially overcome the limitations of the two types of approaches discussed above is to first apply transformation to Y and $\boldsymbol{X}$ and then study the relationship between the transformed Y and $\boldsymbol{X}$. Box and Cox (1964) proposed a family of power transformations (called Box-Cox transformations now) and used them to transform the response Y so that after transformation, the assumptions of linear model, normality and homoscedasticity become appropriate. Later on, Box-Cox transformations were applied to both Y and $\boldsymbol{X}$, and then regression analysis was conducted for the transformed response and predictor variables (see [27]). This extension can accommodate nonliear relationship between the transformed Y and $\boldsymbol{X}$. Although Box-Cox transformations work well in many applications, the power transformations can become too restrictive.

Breiman and Friedman (1985) [3] considered applying general non-parametric transformations to Y and $\boldsymbol{X}$ and further developed the Alternate Conditional Expectation (ACE) algorithm to compute the optimal transformations. Let Y be the response and $\boldsymbol{X} = (X_1, \ldots, X_p)$ be the predictors. Let $g(Y), f_1(X_1), \ldots, f_p(X_p)$ be the transformations of Y and $X_1, \ldots, X_p$, respectively. The optimal transformations are the solutions to the following minimization problem.

$$\begin{aligned} \min_{\substack{g \in L^2(P_Y), \\ f_j \in L^2(P_{X_j})}} \quad & \widetilde{e}^2 = \mathbb{E}[\{g(Y) - \sum_{j=1}^{p} f_j(X_j)\}^2], \\ \text{s.t.} \quad & \mathbb{E}[g(Y)] = \mathbb{E}[f_j(X_j)] = 0; \\ & \mathbb{E}[g^2(Y)] = 1, \mathbb{E}[f_j^2(X_j)] < \infty. \end{aligned} \tag{2.1}$$

Here, P_Y and P_{X_j} denote the marginal distributions of Y and X_j, respectively, and $L^2(P)$ denotes the class of square integrable functions under the measure P. The target function $\widetilde{e}^2$ can be interpreted as the mean square error of regressing $g(Y)$ against $f_j(X_j)$'s. Notice that in the regression, the transformations are applied to the predictors individually, and then the transformed response is regressed against the sum of the transformed predictors. We refer to such a framework as the *optimal univariate transformation framework*. Under some regularity conditions, Breiman and Friedman showed that the optimal transformations exist, and their estimates are asymptotically consistent. Burman [28] proposed to estimate the optimal transformations using B-splines and showed that the resulting estimates are consistent.

The reason we believe optimal transformation can be an effective and efficient approach to investigating the relationship between Y and $\boldsymbol{X}$ is two-fold. Firstly, compared to regression methods based on pre-specified features of $P(Y|\boldsymbol{X})$, the optimal transformation approach does not need to pre-specify a particular feature of $P(Y|\boldsymbol{X})$. As a result, finding the optimal transformations can be considered an automatic procedure to find the best feature under which the relationship between Y and $\boldsymbol{X}$ can be best explored. Secondly, compared with the conditional density estimation approach, finding optimal transformations essentially solves a regression problem, which is numerically less challenging and can lead to more interpretable results.

The optimal univariate transformation framework discussed above has one limitation, that is, it only applies transformation to individual variable. Optimal univariate transformations may be computationally easy to calculate, but from the view point of exploring the relationship between Y and $\boldsymbol{X}$, it can become a disadvantage. When predictors interact with each other, optimal univariate transformations are not able to capture the interactions, and much information about Y and $\boldsymbol{X}$ will be lost. In many applications, predictors are naturally divided into different categories or groups, and they affect the response in groups. In such an application, optimal univariate transformations ignore the group information. In practice, sometimes, the group information is hidden. The optimal univariate transformation framework does not

provide the capacity to recover the group structure of the predictors. The recovery of such group structures not only helps understand the dependence of Y on $\boldsymbol{X}$ but also leads to models with higher prediction power as will be shown later.

To overcome the limitation of the optimal univariate transformation framework, in this chapter, we propose a new framework called the optimal group transformation framework as a general approach for exploring the relationship between Y and $\boldsymbol{X}$. The framework is described as follows. First, the predictors $X_1, \ldots, X_p$ are partitioned into d groups denoted as $\mathbf{X}_1, \ldots, \mathbf{X}_d$. Then, let $g(Y), f_1(\mathbf{X}_1), \ldots, f_d(\mathbf{X}_d)$ be the transformations of Y and $\mathbf{X}_1, \ldots, \mathbf{X}_d$, respectively. The optimal group transformations are the solutions to the following minimization problem.

$$\begin{aligned} \min_{\substack{g\in L^2(P_Y),\\ f_\ell\in L^2(P_{\mathbf{X}_\ell})}} \quad & \widetilde{e}^2 = \mathbb{E}[\{g(Y) - \sum_{\ell=1}^{d} f_\ell(\mathbf{X}_\ell)\}^2], \\ \text{s.t.} \quad & \mathbb{E}\{g(Y)\} = \mathbb{E}\{f_\ell(\mathbf{X}_\ell)\} = 0; \\ & \mathbb{E}\{g^2(Y)\} = 1, \mathbb{E}\{f_\ell^2(\mathbf{X}_\ell)\} < \infty. \end{aligned} \tag{2.2}$$

Here, P_Y denotes the marginal distribution of Y and $P_{\boldsymbol{X}_\ell}$ denotes the joint distribution of all variables in $\boldsymbol{X}_\ell$. It is clear that the original problem (2.1) is a special case of the group version (2.2) with $d = p$. The other extreme case is when $d = 1$ in which the optimization problem (2.2) is equivalent to the maximum correlation problem in [3].

To solve Problem (2.2) and calculate the optimal group transformations, we propose to use Reproducing Kernel Hilbert Space (RKHS)-based methods (or kernel methods) and use cross-covariance and conditional covariance operators developed for kernel methods (see [29], [30], and [16]). The reason of choosing kernel methods over B-splines is due to a number of advantages kernel methods provide for fitting multivariate nonparametric functions, which are discussed in details by [5]. In addition, cross-covariance operators and conditional covariance operators between RKHSs defined via the expectation and covariance of random variables characterize the distributions and conditional distributions of the involved random variables. By using

conditional covariance operator, we can transform the original functional optimization problem to be a functional eigen problem, which can allow simple theoretical analysis and numerical solution.

Given a sample of Y and $\boldsymbol{X}$, the functional eigen problem can further be reduced to a finite rank eigen problem, and the empirical cross-covariance and conditional covariance operators can be estimated by Gram matrices calculated from the kernel functions and the data. Applying matrix eigen value and vector decomposition, we obtain the estimates of the optimal group transformations. Because our proposed approach uses kernel methods, we refer to it as the Optimal Kernel Group Transformation (OKGT) method.

In this chapter, we further show that the OKGT estimates are statistically consistent, that is, they converges to their population counterparts. When the group structure of the predictors are not given a priori, we further propose to apply the OKGT method to randomly generated partitions of the predictors, and then select the partitions that achieve top performance in model fitting after transformation. The optimal kernel group transformations can also be used to generate graphics visualizing the dependence of Y on $\boldsymbol{X}$. Through simulation study and real data applications, we show that the OKGT method is flexible and powerful for exploring the relationship between Y and $\boldsymbol{X}$. We believe the proposed framework, particularly the OKGT method, is a significant contribution to high dimensional regression and useful for data exploration.

The rest of the chapter is organized as follows. In Section 2.1, we introduce various RKHSs, define cross-covariance and conditional covariance operators, and convert the optimal group optimization problem to a functional eigen problem; and we further derive the estimates of the optimal kernel group transformations. The theoretical properties of the estimates are given in Section 2.2. The proofs of the theoretical properties are also included. We report the experimental results based on simulation study and real data applications in Section 2.3 and 2.4. Section 2.5 summarizes this chapter.

2.1 Methods

In this section, we present the development of OKGT. First, we introduce RKHS and direct sum RKHS, and rewrite Problem (2.2) based on those RKHSs. Then, we use covariance and conditional covariance operators on RKHSs to convert the optimal transformation problem to an eigen problem and obtain the optimal transformations at the population level. Lastly, we give an algorithm to obtain the estimates of the optimal transformations under a given sample.

2.1.1 Optimal Kernel Group Transformation

Let $\mathcal{Y}$ be the compact support of Y, and $\mathcal{X}_\ell$ the compact support of the ℓ-th group of predictors $\boldsymbol{X}_\ell$ for $\ell = 1, \ldots, d$. Let $\mathcal{H}_\mathcal{Y}$ and $\mathcal{H}_{\mathcal{X}_\ell}$ denote the RKHSs with domains $\mathcal{Y}$ and $\mathcal{X}_\ell$ and kernels $k_\mathcal{Y}$ and $k_{\mathcal{X}_\ell}$, respectively. It is always assumed that the kernels are positive and satisfy

$$\mathbb{E}_Y[k_\mathcal{Y}(Y, Y)] < \infty \quad \text{and} \quad \mathbb{E}_{\mathbf{X}_\ell}[k_{\mathcal{X}_\ell}(\mathbf{X}_\ell, \mathbf{X}_\ell)] < \infty. \tag{2.3}$$

As pointed out in [30], the assumptions in (2.3) guarantee that $\mathcal{H}_\mathcal{Y}$ and $\mathcal{H}_{\mathcal{X}_\ell}$ are continuously included in $L^2(P_Y)$ and $L^2(P_{X_\ell})$, respectively.

We search for the optimal transformations of Y and $\mathcal{X}_\ell$ in $\mathcal{H}_\mathcal{Y}$ and $\mathcal{H}_{\mathcal{X}_\ell}$ instead of the function space $L^2(P)$. Therefore, the original optimal group transformation problem (2.2) needs to be rewritten as follows.

$$\begin{aligned} \min_{\substack{g \in \mathcal{H}_\mathcal{Y}, \\ f_\ell \in \mathcal{H}_{\mathcal{X}_\ell}}} \quad & e^2 = \mathbb{E}[\{g(Y) - \sum_{\ell=1}^{d} f_\ell(\mathbf{X}_\ell)\}^2], \\ \text{s.t.} \quad & \mathbb{E}[g(Y)] = \mathbb{E}[f_\ell(\mathbf{X}_\ell)] = 0; \\ & \mathbb{E}[g^2(Y)] = 1, \mathbb{E}[f_\ell^2(\mathbf{X}_\ell)] < \infty. \end{aligned} \tag{2.4}$$

Similar to [3], to ensure the existence of the optimal transformations, the following assumption needs to be imposed.

Assumption 2.1.1 *The only set of functions satisfying the constraints in (2.4) such that* $g(Y) + \sum_{\ell=1}^{d} f_\ell(\mathbf{X}_\ell) = 0$ *a.s. are individually zero a.s.*

The optimization problem (2.4) appears to search separately for the transformation g and the individual transformations f_ℓ for $\ell = 1, \cdots, d$. Due to the fact that the target function e^2 only involves $\sum_{\ell=1}^{d} f_\ell(\mathbf{X}_\ell)$, which is an additive sum of f_ℓ's, (2.4) can indeed be solved equivalently in $\mathcal{H}_\mathcal{Y}$ and the direct sum space consisting of $\mathcal{H}_{\mathcal{X}_\ell}$'s, which is defined as

$$\mathcal{H}_\mathcal{X}^+ = \oplus_{\ell=1}^{d} \mathcal{H}_{\mathcal{X}_\ell} := \left\{ f = \sum_{\ell=1}^{d} f_\ell \mid f_\ell \in \mathcal{H}_{\mathcal{X}_\ell}, \ell = 1, \cdots, d \right\}.$$

It can be proved[1] that $\mathcal{H}_\mathcal{X}^+$ is also a RKHS with the corresponding kernel $\sum_{\ell=1}^{d} k_{\mathcal{X}_\ell}$.

Therefore, Problem (2.4), which minimizes the target function w.r.t. each individual function, can be considered as a minimization problem over just $\mathcal{H}_\mathcal{Y}$ and $\mathcal{H}_\mathcal{X}^+$ subject to the same constraints. To solve Problem (2.4) at the population level, one approach is to apply kernel basis expansion methods. In order to simplify and facilitate the theoretical analysis, we resort to covariance and conditional covariance operators and use them to convert the original problem (2.4) to an equivalent eigen problem.

Suppose U and W are two random variables or vectors. Let $\mathcal{H}_\mathcal{U}$ and $\mathcal{H}_\mathcal{W}$ be two RKHSs associated with U and W, respectively. The *cross-covariance operators* $R_{WU} : \mathcal{H}_\mathcal{U} \to \mathcal{H}_\mathcal{W}$ is a mapping from $\mathcal{H}_\mathcal{U}$ to $\mathcal{H}_\mathcal{W}$ such that

$$\begin{aligned} &\langle g, R_{WU} f \rangle_{\mathcal{H}_\mathcal{W}} \\ &\quad = \mathbb{E}_{WU} \left[(f(U) - \mathbb{E}_U[f(U)])(g(W) - \mathbb{E}_W[g(W)]) \right] \\ &\quad = \mathrm{Cov}\left(f(U), g(W) \right) \end{aligned} \tag{2.5}$$

holds for all $f \in \mathcal{H}_\mathcal{U}$ and $g \in \mathcal{H}_\mathcal{W}$ (also see [29], [30]). Riesz's representation theorem guarantees the existence and uniqueness of R_{WU} and it is bounded. The cross-covariance operator R_{WU} contains all the information regarding the dependence of U and W that can be characterized by the functions in the RKHSs. If W is the same as U, $\mathcal{H}_\mathcal{W}$ becomes R_{WW} (or R_{UU}), which is a positive self-adjoint operator and called the *covariance operator*.

[1] See Section 1.4.1 in [31].

In the optimal transformation problem (2.4), $\mathcal{H}_{\mathcal{Y}}$ and $\mathcal{H}_{\mathcal{X}}^{+}$ play the roles of $\mathcal{H}_{\mathcal{W}}$ and $\mathcal{H}_{\mathcal{U}}$ respectively. And the cross-covariance operator $\mathbf{R}_{Y\boldsymbol{X}} : \mathcal{H}_{\mathcal{X}}^{+} \to \mathcal{H}_{\mathcal{Y}}$ can be defined through

$$\langle g, \mathbf{R}_{Y\boldsymbol{X}} f \rangle_{\mathcal{H}_{\mathcal{Y}}} = \mathbb{E}_{Y\boldsymbol{X}} \left[\left(\sum_{\ell=1}^{d} f_\ell(X_\ell) - \mathbb{E}_{\boldsymbol{X}} \left[\sum_{\ell=1}^{d} f_\ell(X_\ell) \right] \right) (g(Y) - \mathbb{E}_Y [g(Y)]) \right].$$

Following the definition of (2.5), the operators $R_{Y\boldsymbol{X}_\ell}$, $R_{\boldsymbol{X}_\ell \boldsymbol{X}_j}$ and $\mathbf{R}_{\boldsymbol{X}\boldsymbol{X}}$ can be similarly defined. Because $\mathcal{H}_{\mathcal{X}}^{+}$ is a direct sum space of $\mathcal{H}_{\mathcal{X}_\ell}$'s, $\mathbf{R}_{Y\boldsymbol{X}}$ and $\mathbf{R}_{\boldsymbol{X}\boldsymbol{X}}$ can be decomposed in terms of $R_{Y\boldsymbol{X}_\ell}$ and $R_{\boldsymbol{X}_\ell \boldsymbol{X}_j}$ with $\ell, j = 1, 2, \ldots, d$. In particular, for f and $f' \in \mathcal{H}_{\mathcal{X}}$ and $g \in \mathcal{H}_{\mathcal{Y}}$,

$$\langle g, \mathbf{R}_{Y\boldsymbol{X}} f \rangle_{\mathcal{H}_{\mathcal{Y}}} = \sum_{\ell=1}^{d} \langle g, R_{Y\boldsymbol{X}_\ell} f_\ell \rangle_{\mathcal{H}_{\mathcal{Y}}}, \tag{2.6}$$

$$\langle f', \mathbf{R}_{\boldsymbol{X}\boldsymbol{X}} f \rangle_{\mathcal{H}_{\mathcal{X}}} = \sum_{\ell=1}^{d} \sum_{j=1}^{d} \left\langle f'_\ell, R_{\boldsymbol{X}_\ell \boldsymbol{X}_j} f_j \right\rangle_{\mathcal{H}_{\mathcal{X}_\ell}}. \tag{2.7}$$

Due to the above decompositions, we can define the matrix representations for the additive cross-covariance and covariance operators $\mathbf{R}_{Y\boldsymbol{X}}$ and $\mathbf{R}_{\boldsymbol{X}\boldsymbol{X}}$ as follows.

$$\mathbf{R}_{Y\boldsymbol{X}} = \begin{bmatrix} R_{Y\boldsymbol{X}_1} & R_{Y\boldsymbol{X}_2} & \cdots & R_{Y\boldsymbol{X}_d} \end{bmatrix}, \tag{2.8}$$

and

$$\mathbf{R}_{\boldsymbol{X}\boldsymbol{X}} = \begin{bmatrix} R_{\boldsymbol{X}_1\boldsymbol{X}_1} & R_{\boldsymbol{X}_1\boldsymbol{X}_2} & \cdots & R_{\boldsymbol{X}_1\boldsymbol{X}_d} \\ R_{\boldsymbol{X}_2\boldsymbol{X}_1} & R_{\boldsymbol{X}_2\boldsymbol{X}_2} & \cdots & R_{\boldsymbol{X}_2\boldsymbol{X}_d} \\ \vdots & \vdots & \ddots & \vdots \\ R_{\boldsymbol{X}_d\boldsymbol{X}_1} & R_{\boldsymbol{X}_d\boldsymbol{X}_2} & \cdots & R_{\boldsymbol{X}_d\boldsymbol{X}_d} \end{bmatrix}. \tag{2.9}$$

These matrix representations admit the usual matrix operations (see [32]), which will facilitate the estimation procedure for the operators in Section 2.1.2.

To convert the optimal group transformation problem (2.4) to an eigen problem, we need to introduce and use another type of operators called the *conditional covariance operator*. Following [16], the conditional covariance operator for W given

U, which are equipped with the corresponding RKHSs as discussed earlier, is defined through the cross-covariance and covariance operators as

$$R_{WW|U} := R_{WW} - R_{WU}R_{UU}^{-1}R_{UW}.$$

Proposition 2 in [16] shows that for any $g \in \mathcal{H}_{\mathcal{W}}$,

$$\langle g, R_{WW|U}g\rangle_{\mathcal{H}_{\mathcal{W}}} = \inf_{f\in H_{\mathcal{U}}} \mathbb{E}_{WU}\left|(g(W) - \mathbb{E}_W[g(W)]) - (f(U) - \mathbb{E}_U[f(U)])\right|^2. \tag{2.10}$$

Again by replacing $\mathcal{W}$ and $\mathcal{U}$ with $\mathcal{H}_{\mathcal{Y}}$ and ${\mathcal{H}_{\mathcal{X}}}^+$, the conditional covariance operator $\mathbf{R}_{YY|\boldsymbol{X}}$ is defined as

$$\mathbf{R}_{YY|\boldsymbol{X}} := R_{YY} - \mathbf{R}_{Y\boldsymbol{X}}\mathbf{R}_{\boldsymbol{X}\boldsymbol{X}}^{-1}\mathbf{R}_{\boldsymbol{X}Y}. \tag{2.11}$$

Similar to Proposition 2 in [16], we have the following proposition.

Proposition 2.1.1 *For any $g \in \mathcal{H}_{\mathcal{Y}}$,*

$$\langle g, \mathbf{R}_{YY|X}g\rangle_{\mathcal{H}_{\mathcal{Y}}} = \inf_{f\in\mathcal{H}_{\mathcal{X}}^+} \mathbb{E}_{YX}|\left(g(Y) - \mathbb{E}_Y[g(Y)]\right) - \left(f(X) - \mathbb{E}_X[f(X)]\right)|^2, \tag{2.12}$$

where $\mathcal{H}_{\mathcal{X}}^+$ is the direct sum RKHS defined in (2.1.1).

Proposition 2.1.1 contributes a key step towards converting the optimization problem (2.4) to an equivalent eigen problem. To solve Problem (2.4), a two-step approach can be taken. In the first step, the target function is minimized with respect to f. Then in the second step, the resulting target function is further minimized with respect to g. With the help of Proposition 2.1.1, the second step becomes an eigen problem involving the conditional covariance operator $\mathbf{R}_{YY|\boldsymbol{X}}$. We state this result as another proposition.

Proposition 2.1.2 *The optimization problem (2.4) is equivalent to*

$$\begin{aligned} \min_{g\in\mathcal{H}_{\mathcal{Y}}} \quad & \langle g, \mathbf{R}_{YY|\boldsymbol{X}}g\rangle_{\mathcal{H}_{\mathcal{Y}}}, \\ s.t. \quad & \langle g, R_{YY}g\rangle_{\mathcal{H}_{\mathcal{Y}}} = 1. \end{aligned} \tag{2.13}$$

Plugging the expression of $\mathbf{R}_{YY|\boldsymbol{X}}$ in (2.11), the minimization problem (2.13) becomes the following generalized eigen problem,

$$\begin{aligned} \max_{g\in\mathcal{H}_\mathcal{Y}} \quad & \langle g, \mathbf{R}_{YX}\mathbf{R}_{\boldsymbol{XX}}^{-1}\mathbf{R}_{\boldsymbol{X}Y} g\rangle_{\mathcal{H}_\mathcal{Y}}, \\ \text{s.t.} \quad & \langle g, R_{YY} g\rangle_{\mathcal{H}_\mathcal{Y}} = 1. \end{aligned} \tag{2.14}$$

Further defining $\varphi := R_{YY}^{1/2} g$, the generalized eigen problem (2.14) can be rewritten as

$$\begin{aligned} \max_{\varphi\in\mathcal{H}_\mathcal{Y}} \quad & \langle \varphi, R_{YY}^{-1/2}\mathbf{R}_{Y\boldsymbol{X}}\mathbf{R}_{\boldsymbol{XX}}^{-1}\mathbf{R}_{\boldsymbol{X}Y} R_{YY}^{-1/2}\varphi\rangle_{\mathcal{H}_\mathcal{Y}}, \\ \text{s.t.} \quad & ||\varphi||_{\mathcal{H}_\mathcal{Y}}^2 = 1. \end{aligned} \tag{2.15}$$

It is not difficult to see that the solution of Problem (2.15), denoted as φ^*, is a unit eigenfunction of $R_{YY}^{-1/2}\mathbf{R}_{Y\boldsymbol{X}}\mathbf{R}_{\boldsymbol{XX}}^{-1}\mathbf{R}_{\boldsymbol{X}Y} R_{YY}^{-1/2}$ corresponding to its largest eigen value. By denoting the largest eigen value as λ_1 and the minimum of the target function in (2.13) as e^{2*}, we have $\lambda_1 = 1 - e^{2*}$.

After having obtained φ^*, the optimal transformations of Y and X_ℓ's are given by the inverse mappings $g^* = R_{YY}^{-1/2}\varphi^*$ and $f^* = \mathbf{R}_{\boldsymbol{XX}}^{-1}\mathbf{R}_{\boldsymbol{X}Y} g^*$. Note that $f^* = f_1^* + f_2^* + \cdots + f_d^*$ is a function in $\mathcal{H}_\mathcal{X}^+$. Using the matrix representations (2.8) and (2.9) of $\mathbf{R}_{Y\boldsymbol{X}}$ and $\mathbf{R}_{\boldsymbol{XX}}$, we can obtain the individual optimal transforms f_ℓ^* for $\ell = 1, \ldots, d$.

Remark It is proved in [29] that in general, a cross-covariance operator $R_{WU} : \mathcal{H}_\mathcal{U} \to \mathcal{H}_\mathcal{W}$ admits the decomposition $R_{WU} = R_{WV}^{1/2} V_{WU} R_{UU}^{1/2}$, where $V_{WU} : \mathcal{H}_\mathcal{U} \to \mathcal{H}_\mathcal{W}$ is a unique bounded operator such that $\|V_{WU}\| \leq 1$ and $V_{WU} = Q_W V_{WU} Q_U$ [2]. Based on the above decomposition, the conditional covariance operator can be rewritten as $R_{WW|U} := R_{WW} - R_{WW}^{1/2} V_{WU} V_{UW} R_{WW}^{1/2}$. In our case, we denote the counterpart of V_{YX} by $\mathbf{V}_{Y\boldsymbol{X}} : \mathcal{H}_\mathcal{X}^+ \to \mathcal{H}_\mathcal{Y}$ Then, we have

$$\mathbf{V}_{Y\boldsymbol{X}}\mathbf{V}_{\boldsymbol{X}Y} = R_{YY}^{-1/2}\mathbf{R}_{Y\boldsymbol{X}}\mathbf{R}_{\boldsymbol{XX}}^{-1}\mathbf{R}_{\boldsymbol{X}Y} R_{YY}^{-1/2}$$

Clearly, $\mathbf{V}_{Y\boldsymbol{X}}\mathbf{V}_{\boldsymbol{X}Y}$ is self-adjoint. Assuming it is compact, the existence of optimal transformations g^* and f^* in RKHSs is guaranteed by the spectral theorem. We will

[2] $Q_U : \mathcal{H}_\mathcal{U} \to \overline{\mathcal{R}(R_{UU})}$ and $Q_W : \mathcal{H}_\mathcal{W} \to \overline{\mathcal{R}(R_{WW})}$ are two orthogonal projections.

show later in Section 2.2 that $\mathbf{V}_{Y\boldsymbol{X}}$ plays an important role in deriving the theoretical results.

2.1.2 Estimation Method

In Section 2.1.1, we have shown that the optimal group transformations can be obtained by solving an equivalent eigen problem involving covariance and conditional covariance operators. In this section, we focus on estimating the optimal transformations for a finite sample. We first derive the empirical covariance and conditional covariance operators, and further use them to define the empirical version of the eigen problem. With proper regularization, the empirical eigen problem can be solved to produce estimates of the optimal group transformations.

Let $\{y_i, \boldsymbol{x}_{i1}, \ldots, \boldsymbol{x}_{id}\}_{i=1}^n$ be i.i.d. samples[3]. We use the cross covariance operator R_{YX_ℓ} as an example to show how to derive the empirical operators.

Let $\tilde{k}_{\mathcal{Y}}(\cdot, y_i) = k_{\mathcal{Y}}(\cdot, y_i) - n^{-1}\sum_{s=1}^n k_{\mathcal{Y}}(\cdot, y_s)$ and $\tilde{k}_{\mathcal{X}_\ell} = k_{\mathcal{X}}(\cdot, \boldsymbol{x}_i) - n^{-1}\sum_{s=1}^n k_{\mathcal{X}}(\cdot, \boldsymbol{x}_s)$ be the centered feature mappings of the observed data. We define $\widetilde{\mathcal{H}}_{\mathcal{Y}}$ and $\widetilde{\mathcal{H}}_{\mathcal{X}_\ell}$ to be the spaces spanned by $\{\tilde{k}_{\mathcal{Y}}(\cdot, y_i)\}_{i=1}^n$ and $\{\tilde{k}_{\mathcal{X}}(\cdot, \boldsymbol{x}_i)\}_{i=1}^n$, respectively. For any $g \in \mathcal{H}_{\mathcal{Y}}$ and $f \in \mathcal{H}_{\mathcal{X}_\ell}$, we can write $g = \sum_{i=1}^n \beta_i \tilde{k}_{\mathcal{Y}}(\cdot, y_i) + g^\perp$ and $f_\ell = \sum_{i=1}^n \alpha_i^\ell \tilde{k}_{\mathcal{X}_\ell}(\cdot, \boldsymbol{x}_i) + f_\ell^\perp$, where $g^\perp$ and $f_\ell^\perp$ are the functions that belong to the orthogonal complements of $\widetilde{\mathcal{H}}_{\mathcal{Y}}$ and $\widetilde{\mathcal{H}}_{\mathcal{X}_\ell}$ in $\mathcal{H}_{\mathcal{Y}}$ and $\mathcal{H}_{\mathcal{X}_\ell}$, respectively. This construction ensures the zero mean constrains in (2.4). By using the reproducing property of RKHSs and Riesz representation theorem, the empirical version of the operator $R_{Y\boldsymbol{X}_\ell}$, denoted as $\widehat{R}^{(n)}_{Y\boldsymbol{X}_\ell}$, is given by

$$\begin{aligned}
\left\langle g, \widehat{R}^{(n)}_{Y\mathbf{X}_\ell} f_\ell \right\rangle_{\mathcal{H}_{\mathcal{Y}}} &= \widehat{\mathrm{Cov}}(g(Y), f_\ell(\mathbf{X}_\ell)) \\
&= \frac{1}{n}\sum_{i=1}^n \left\langle g, \tilde{k}_{\mathcal{Y}}(\cdot, y_i) \right\rangle_{\mathcal{H}_{\mathcal{Y}}} \left\langle f_\ell, \tilde{k}_{\mathcal{X}}(\cdot, \boldsymbol{x}_i) \right\rangle_{\mathcal{H}_{\mathcal{X}_\ell}} \\
&= \sum_{k=1}^n \sum_{i=1}^n \sum_{j=1}^n \beta_i \tilde{k}_{\mathcal{Y}}(y_i, y_k) \tilde{k}_{\mathcal{X}_\ell}(\boldsymbol{x}_{j\ell}, \boldsymbol{x}_{k\ell}) \alpha_j^\ell,
\end{aligned} \tag{2.16}$$

[3]Here we use the group representation. Each $\boldsymbol{x}_{i\ell}$ denotes the i^{th} observation for the ℓ^{th} group of predictor variables. So it can be a scalar value or a vector depending on the pre-specified group structure.

where $\widehat{\mathrm{Cov}}(g(Y), f_\ell(\boldsymbol{X}_\ell))$ is the sample covariance of $g(Y)$ and $f_\ell(\boldsymbol{X}_\ell)$. Therefore, the sample covariance is of finite rank and can be represented by Gram matrices.

We define $G_\mathcal{Y}$ to be the Gram matrix of the kernel $k_\mathcal{Y}$ for Y as $(G_\mathcal{Y})_{ij} = k_\mathcal{Y}(y_i, y_j)$. Let $\mathbf{1}_n = (1, \ldots, 1)^T$, then the centered Gram matrix is given by

$$K_\mathcal{Y} = \left(I_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right) G_\mathcal{Y} \left(I_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right).$$

Similarly, we can derive the centered Gram matrix $K_{\mathcal{X}_\ell}$ for $\boldsymbol{X}_\ell$'s. By applying the representer theorem, the empirical operator in (2.16) admits the following matrix representation,

$$\left\langle g, \widehat{R}^{(n)}_{Y\mathbf{X}_\ell} f_\ell \right\rangle_{\mathcal{H}_\mathcal{Y}} = \boldsymbol{\beta}^T K_\mathcal{Y} K_{\mathcal{X}_\ell} \boldsymbol{\alpha}^\ell.$$

Therefore, the finite rank operator $\widehat{R}^{(n)}_{Y\boldsymbol{X}_\ell}$ can be estimated by $K_\mathcal{Y} K_{\mathcal{X}_\ell}$. Similarly, we estimate $\widehat{R}^{(n)}_{YY}$ and $\widehat{R}_{\boldsymbol{X}_\ell \boldsymbol{X}_k}$ by $K_\mathcal{Y} K_\mathcal{Y}$ and $K_{\mathcal{X}_\ell} K_{\mathcal{X}_j}$ respectively. Using the operator matrix representation gives the following estimates of the additive operators as block matrices,

$$\widehat{\mathbf{R}}^{(n)}_{\boldsymbol{X}\boldsymbol{X}} = \begin{bmatrix} K_{\mathcal{X}_1} K_{\mathcal{X}_1} & K_{\mathcal{X}_1} K_{\mathcal{X}_2} & \cdots & K_{\mathcal{X}_1} K_{\mathcal{X}_d} \\ K_{\mathcal{X}_2} K_{\mathcal{X}_1} & K_{\mathcal{X}_2} K_{\mathcal{X}_2} & \cdots & K_{\mathcal{X}_2} K_{\mathcal{X}_d} \\ \vdots & \vdots & \ddots & \vdots \\ K_{\mathcal{X}_d} K_{\mathcal{X}_1} & K_{\mathcal{X}_d} K_{\mathcal{X}_2} & \cdots & K_{\mathcal{X}_d} K_{\mathcal{X}_d} \end{bmatrix}$$

and

$$\widehat{\mathbf{R}}^{(n)}_{Y\boldsymbol{X}} = \left(K_\mathcal{Y} K_{\mathcal{X}_1}, K_\mathcal{Y} K_{\mathcal{X}_2}, \ldots, K_\mathcal{Y} K_{\mathcal{X}_d}\right).$$

By using the estimates of the operators defined above, the empirical version of the equivalent eigen problem of OKGT in (2.15) can now be written as

$$\max_{\varphi \in \mathcal{H}_\mathcal{Y}} \left\langle \varphi, \left(\widehat{R}^{(n)}_{YY} + \epsilon_n I\right)^{-1/2} \widehat{\mathbf{R}}^{(n)}_{Y\boldsymbol{X}} \left(\widehat{\mathbf{R}}^{(n)}_{\boldsymbol{X}\boldsymbol{X}} + \epsilon_n I\right)^{-1} \widehat{\mathbf{R}}^{(n)}_{\boldsymbol{X}Y} \widehat{\mathbf{R}}^{(n)}_{\boldsymbol{X}Y} \left(\widehat{R}^{(n)}_{YY} + \epsilon_n I\right)^{-1/2} \varphi \right\rangle_{\mathcal{H}_\mathcal{Y}}$$

$$s.t. \left\|\varphi\right\|_{\mathcal{H}_\mathcal{Y}} = 1. \tag{2.17}$$

Note that the regularization term $\epsilon_n I$ is needed above, which would enable matrix inversion and avoid trivial solution. A detailed discussion can be found in [33].

In the spirit of the decomposition of a cross-covariance operator mentioned in the remark at the end of Section 2.1.1, we simplify the notations by defining

$$\widehat{\mathbf{V}}_{Y\boldsymbol{X}}^{(n)} = \left(\widehat{R}_{YY}^{(n)} + \epsilon_n I\right)^{-1/2} \widehat{\mathbf{R}}_{Y\boldsymbol{X}}^{(n)} \left(\widehat{\mathbf{R}}_{\boldsymbol{XX}}^{(n)} + \epsilon_n I\right)^{-1/2}.$$

So the product of the matrices in (2.17) becomes $\widehat{\mathbf{V}}_{Y\boldsymbol{X}}^{(n)} \widehat{\mathbf{V}}_{\boldsymbol{X}Y}^{(n)}$ in the following discussion.

Let $\widehat{\varphi}^*$ be the unit eigen vector of $\widehat{\mathbf{V}}_{Y\boldsymbol{X}}^{(n)} \widehat{\mathbf{V}}_{\boldsymbol{X}Y}^{(n)}$ corresponding to its largest eigen value. Then the empirical estimates of the optimal transformations are given by

$$\hat{g}^* = \left(\widehat{R}_{YY}^{(n)} + \epsilon_n I\right)^{-1/2} \hat{\varphi}^*, \tag{2.18}$$

$$\hat{f}^* = \left(\widehat{\mathbf{R}}_{\boldsymbol{XX}}^{(n)} + \epsilon_n I\right)^{-1} \widehat{\mathbf{R}}_{\boldsymbol{X}Y}^{(n)} \hat{g}^*. \tag{2.19}$$

Thanks to the additive structure, the numerical estimate of $\hat{f}^*$ is in the form of a column stack of $\hat{f}_\ell^*$, $\ell = 1, \ldots, d$, which are the estimates of the optimal transformations for individual groups.

2.1.3 Speeding Up Estimation for Large Sample

One limitation of using kernel methods is that it does not scale well when the sample size is large. However, there are methods developed to overcome this difficulty. Sparse greedy matrix approximation [34] uses a variant of matching pursuit algorithm with probabilistic speedup. Low-rank kernel representations [35] uses a known factorization technique to approximate a given kernel matrix by a low rank matrix, which will be used in training instead of the original kernel matrix. Nyström method for low rank matrix approximation [36] uses random samples of a kernel matrix's rows and columns to construct the low rank matrix. All of these techniques can be incorporated in OKGT to speed up its estimation for large sample size.

2.1.4 Additive Group Structure Identification and Graphics

We call the method developed in Sections 2.1.1 and 2.1.2 the Optimal Kernel Group Transformation (OKGT). Note that different group partitions may yield different fitting results, which further lead to different model interpretation. Ideally, the underlying group structure is given before the OKGT method is applied. However, the underlying structure may be unknown in practice. Therefore, it is essential to have a procedure to detect a suitable group structure which can well approximate the underlying true structure and yield meaningful interpretations. An optimal procedure to find the true underlying group structure should take a number of factors into consideration, such as a proper definition of the discrepancy measure between two group structures and the selection of group size and group numbers. The development of such an optimal procedure will be discussed in Chapter 3.

In this chapter, we use an intuitive approach, which is the random partition method, for group structure detection and use $R^2 = \lambda_1$, the largest eigen value of $\widehat{\mathbf{V}}^{(n)}_{Y\boldsymbol{X}}\widehat{\mathbf{V}}^{(n)}_{\boldsymbol{X}Y}$, as the criterion to identify a suitable group structure. We prefer a structure that maximizes R^2 among all partitions and at the same time has small group sizes. A model with relatively small group sizes can alleviate the curse of dimensionality and enhance the interpretability of the fitting results. Due to this reason, we suggest that each group contains no more than four variables. Though we will develop an additive group structure identification method in Chapter 3, random partition can still be used to quickly explore data and serve as a benchmark for comparing with a principled method.

Once a proper group structure is detected and optimal transformations are found by applying the OKGT method, graphical tools can be used to explore the relationship between the variables. Two examples include the plot of transformed response against the original response, and the marginal plots of transformed response against each transformed group of predictors. When a certain group contains two variables, 3-D plots can be employed to visualize the relationship between the response, the

transformed group of variables, and each of the variables in the group. We believe all these plots will provide more insights in revealing the relationships between the predictors and the response, resulting in meaningful interpretations. More illustrations on the afore-mentioned plots and graphs are given in the synthetic data and real data examples in Section 2.3 and 2.4.

2.2 Theoretical Properties of OKGT

In this section, we show that the estimates of the optimal kernel group transformations produced by the OKGT method are consistent in L^2 norm. The regularization parameter ϵ_n is assumed to decay to zero and the main idea of the proof follows [30].

First, we will show that the empirical eigen function obtained by solving Problem (2.17) converges to its population counterpart in (2.15) under the RKHS norm.

This section is divided into two subsections. The first subsection includes the main theorems. The supporting lemmas are collected and presented in the second subsection if readers are interested in the details of the proof.

2.2.1 Main Results

The proof of the theorems relies on the assumption that $\mathbf{V}_{Y\mathbf{X}}$ is compact, which may not hold in general (see [30]). If $\mathbf{V}_{Y\mathbf{X}}$ is not compact, the solution of the population version of optimal transformation problem may not exist in RKHSs. A sufficient condition for $\mathbf{V}_{Y\mathbf{X}}$ being compact is given in [30], which is restated here.

Assumption 2.2.1 *Let $(\mathcal{X}, \mathcal{B}_{\mathcal{X}}, \mu_{\mathcal{X}})$ and $(\mathcal{Y}, \mathcal{B}_{\mathcal{Y}}, \mu_{\mathcal{Y}})$ be two probability spaces. Let $p_{\mathbf{X}Y}$, $p_{\mathbf{X}}$, and p_Y be the density functions. If*

$$\int\int \frac{p_{\mathbf{X}Y}(\mathbf{x}, y)^2}{p_{\mathbf{X}}(\mathbf{x}) p_Y(y)} d\mu_{\mathcal{X}} d\mu_{\mathcal{Y}} < \infty,$$

then the operator $\mathbf{V}_{Y\mathbf{X}} : \mathcal{H}_{\mathcal{X}} \to \mathcal{H}_{\mathcal{Y}}$ is Hilbert-Schmidt, which implies the compactness of $\mathbf{V}_{Y\mathbf{X}}$.

Then, the following theorem shows that the largest eigen function converges in probability.

Theorem 2.2.1 *Assume* $\mathbf{V}_{Y\mathbf{X}}$ *is compact. Let* $\hat{\varphi}^*$ *be an eigen function corresponding to the largest eigen value of* $\widehat{\mathbf{V}}_{Y\mathbf{X}}^{(n)}\widehat{\mathbf{V}}_{\mathbf{X}Y}^{(n)}$. *Then, as* $n \to \infty$, *there exists a sequence* $\epsilon_n \to 0$ *such that,*

$$\left|\langle \hat{\varphi}^*, \varphi^* \rangle_{\mathcal{H}_\mathcal{Y}}\right| \xrightarrow{P} 1,$$

where φ^* *is an eigen function corresponding to the largest eigen value of* $\mathbf{V}_{Y\mathbf{X}}\mathbf{V}_{\mathbf{X}Y}$.

Proof Denote $A = \mathbf{V}_{Y\boldsymbol{X}}\mathbf{V}_{\boldsymbol{X}Y}$. Because A is positive and compact, the spectrum theorem gives the following decomposition:

$$A = \sum_{i=1}^{\infty} \lambda_i \varphi_i \langle \varphi_i, \cdot \rangle_{\mathcal{H}_\mathcal{Y}},$$

where $\lambda_1 > \lambda_2 \geq \cdots \geq 0$ are the eigen-values and $\{\varphi_i\}_i$ are the corresponding eigen-vectors. Note here we assume that the eigen-vector corresponding to the largest eigen-value is unique. Though the dimension of the eigen-space corresponding to the largest eigen-value may be higher than one, OKGT only requires the existence of one such eigen-vector as the optimal transformation.

Let $\hat{\varphi}^*$ be the eigen-vector corresponding to the largest eigen-value of $A_n = \widehat{\mathbf{V}}_{Y\boldsymbol{X}}^{(n)}\widehat{\mathbf{V}}_{\boldsymbol{X}Y}^{(n)}$, then

$$\begin{aligned}
\langle \hat{\varphi}^*, A\hat{\varphi}^* \rangle_{\mathcal{H}_\mathcal{Y}} &= \lambda_1 \langle \hat{\varphi}^*, \varphi_1 \rangle_{\mathcal{H}_\mathcal{Y}}^2 + \sum_{i=2}^{\infty} \lambda_2 \langle \hat{\varphi}^*, \varphi_i \rangle_{\mathcal{H}_\mathcal{Y}}^2 \\
&= \lambda_1 \langle \hat{\varphi}^*, \varphi_1 \rangle_{\mathcal{H}_\mathcal{Y}}^2 + \lambda_2 \left(1 - \langle \hat{\varphi}^*, \varphi_1 \rangle_{\mathcal{H}_\mathcal{Y}}^2\right).
\end{aligned}$$

On the other hand,

$$\begin{aligned}
\left|\langle \hat{\varphi}^*, A\hat{\varphi}^* \rangle_{\mathcal{H}_\mathcal{Y}} - \langle \varphi_1, A\varphi_1 \rangle_{\mathcal{H}_\mathcal{Y}}\right| &\leq \left|\langle \hat{\varphi}^*, A\hat{\varphi}^* \rangle_{\mathcal{H}_\mathcal{Y}} - \langle \hat{\varphi}^*, A_n\hat{\varphi}^* \rangle_{\mathcal{H}_\mathcal{Y}}\right| + \left|\langle \hat{\varphi}^*, A_n\hat{\varphi}^* \rangle_{\mathcal{H}_\mathcal{Y}} - \langle \varphi_1, A\varphi_1 \rangle_{\mathcal{H}_\mathcal{Y}}\right| \\
&\leq \|A - A_n\| + |\|A_n\| - \|A\|| \to 0.
\end{aligned}$$

This implies that $\langle \hat{\varphi}^*, A\hat{\varphi}^* \rangle_{\mathcal{H}_\mathcal{Y}} \to \langle \varphi_1, A\varphi_1 \rangle_{\mathcal{H}_\mathcal{Y}}$. So $\langle \hat{\varphi}^*, \varphi_1 \rangle_{\mathcal{H}_\mathcal{Y}} \to 1$, equivalently $\|\hat{\varphi}^* - \varphi_1\|_{\mathcal{H}_\mathcal{Y}} \to 1$. ■

The next theorem further establishes the consistency of the estimated optimal transformations in L^2 norm.

Theorem 2.2.2 *Assume that φ^* is in the range of R_{YY}, and $\mathbf{V}_{Y\mathbf{X}}$ is compact. Then, as $n \to \infty$, there exists a sequence $\epsilon_n \to 0$ such that*

$$\|\hat{g}^* - g^*\|_{L^2_{P_Y}} \xrightarrow{P} 0 \qquad \textit{and} \qquad \left\|\hat{f}^* - f^*\right\|_{L^2_{P_{\mathbf{X}}}} \xrightarrow{P} 0.$$

where g^* and f^* are obtained by solving Problem (2.4) and $\hat{g}^*$ and $\hat{f}^*$ are given by (2.18) and (2.19).

Proof Without loss of generality, we assume $\hat{\varphi}^{*(n)} \to \varphi_k^*$ in $\mathcal{H}_\mathcal{Y}$. As $\hat{g}^* = \left(\widehat{R}_{YY}^{(n)} + \epsilon_n I\right)^{-\frac{1}{2}} \hat{\varphi}^*$ and $g^* = R_{YY}^{-1/2}\varphi^*$, we have

$$\|\hat{g}^* - g^*\|^2_{L^2_{P_Y}} = \left\|R_{YY}^{1/2}\left(\hat{g}^* - g^*\right)\right\|^2_{\mathcal{H}_\mathcal{Y}} = \left\|R_{YY}^{1/2}\hat{g}^* - \varphi^*\right\|^2_{\mathcal{H}_\mathcal{Y}}.$$

The fact that $\left\|R_{YY}^{1/2}\hat{g}^* - \varphi^*\right\|^2_{\mathcal{H}_\mathcal{Y}} \xrightarrow{P} 0$ follows the proof of Theorem 2 in [30].

Similarly, as $\hat{f}^* = \left(\widehat{\mathbf{R}}_{\boldsymbol{XX}}^{(n)} + \epsilon_n I\right)^{-1} \widehat{\mathbf{R}}_{\boldsymbol{X}Y}^{(n)}\hat{g}^*$ and $f^* = \left(\mathbf{R}_{\boldsymbol{XX}}\right)^{-1}\mathbf{R}_{\boldsymbol{X}Y}g^*$, the same result holds for $\left\|\hat{f}^* - f^*\right\|_{L^2_{P_{\mathbf{X}}}}$. ■

2.2.2 Supporting Lemmas

This subsection collects the lemmas that are needed to prove Theorem 2.2.1 and 2.2.2 in the previous subsection.

Lemma 2.2.1 *As $n \to \infty$,*

$$\begin{aligned}
\left\|\widehat{\mathbf{R}}_{Y\boldsymbol{X}}^{(n)} - \mathbf{R}_{Y\boldsymbol{X}}\right\|_{HS} &= \mathcal{O}_p\left(dn^{-1/2}\right), \\
\left\|\widehat{R}_{YY}^{(n)} - R_{YY}\right\|_{HS} &= \mathcal{O}_p\left(n^{-1/2}\right), \\
\left\|\widehat{\mathbf{R}}_{\boldsymbol{XX}}^{(n)} - \mathbf{R}_{\boldsymbol{XX}}\right\|_{HS} &= \mathcal{O}\left(d^2n^{-1/2}\right).
\end{aligned}$$

Proof Throughout the proof, we use the following definition of the norm of the product of two functions in a product space:

$$\|fg\|_{\mathcal{H}_f\otimes\mathcal{H}_g} := \|f \otimes g\|_{\mathcal{H}_f\otimes\mathcal{H}_g} = \|f\|_{\mathcal{H}_f}\|g\|_{\mathcal{H}_g}.$$

We prove the first one, and the rest can be proved similarly.

Write $F = \sum_{\ell=1}^{d} F^{(\ell)}$ where $F^{(\ell)} = k_{\mathcal{X}_\ell}(\cdot, \mathbf{X}_\ell) - \mathbb{E}\left[k_{\mathcal{X}_\ell}(\cdot, \mathbf{X}_\ell)\right]$, $G = k_{\mathcal{Y}}(\cdot, Y) - \mathbb{E}\left[k_{\mathcal{Y}}(\cdot, Y)\right]$, $F_i = \sum_{\ell=1}^{d} F_i^{(\ell)}$ where $F_i^{(\ell)} = k_{\mathcal{X}_\ell}(\cdot, \mathbf{X}_{\ell i}) - \mathbb{E}\left[k_{\mathcal{X}_\ell}(\cdot, \mathbf{X}_\ell)\right]$, $G_i = k_{\mathcal{Y}}(\cdot, Y_i) - \mathbb{E}\left[k_{\mathcal{Y}}(\cdot, Y)\right]$ for $i = 1, 2, \ldots, n$, and $\mathcal{F} = \mathcal{H}_{\mathcal{X}}^{+} \otimes \mathcal{H}_{\mathcal{Y}}$. Then, $F, F_1, \ldots, F_n$ are i.i.d. random elements in $\mathcal{H}_{\mathcal{X}}^{+}$, and a similar fact holds for $G, G_1, \ldots, G_n$.

Then,

$$\begin{aligned}\left\|\widehat{\mathbf{R}}_{Y\boldsymbol{X}}^{(n)} - \mathbf{R}_{Y\boldsymbol{X}}\right\|_{HS}^2 &= \left\|\frac{1}{n}\sum_{i=1}^{n}\left(F_i - \frac{1}{n}\sum_{j=1}^{n} F_j\right)\left(G_i - \frac{1}{n}\sum_{j=1}^{n} G_j\right) - \mathbb{E}[FG]\right\|_{\mathcal{F}}^2 \\ &= \left\|\frac{1}{n}\sum_{i=1}^{n} F_i G_i - \mathbb{E}[FG] - \left(2 - \frac{1}{n}\right)\left(\frac{1}{n}\sum_{i=1}^{n} F_i\right)\left(\frac{1}{n}\sum_{i=1}^{n} G_i\right)\right\|_{\mathcal{F}}^2,\end{aligned}$$

which provides the following bound

$$\left\|\widehat{\mathbf{R}}_{Y\boldsymbol{X}}^{(n)} - \mathbf{R}_{Y\boldsymbol{X}}\right\|_{HS} \leq \left\|\frac{1}{n}\sum_{i=1}^{n} F_i G_i - \mathbb{E}[FG]\right\|_{\mathcal{F}} + 2\left\|\left(\frac{1}{n}\sum_{i=1}^{n} F_i\right)\left(\frac{1}{n}\sum_{i=1}^{n} G_i\right)\right\|_{\mathcal{F}}. \tag{2.20}$$

Let $Z_i^{(\ell)} = \left(F_i^{(\ell)} G_i - \mathbb{E}[F^{(\ell)} G]\right)$ and $Z_i = \sum_{\ell=1}^{d} Z_i^{(\ell)} = F_i G_i - \mathbb{E}[FG]$, we have,

$$\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n} Z_i\right\|_{\mathcal{F}}^2 = \frac{1}{n}\mathbb{E}\left\|Z_1\right\|_{\mathcal{F}}^2 = \frac{1}{n}\mathbb{E}\left\|\sum_{\ell=1}^{d} Z_1^{(\ell)}\right\|_{\mathcal{F}}^2 \leq \frac{d^2}{n}\max_{\ell}\mathbb{E}\left\|Z_1^{(j)}\right\|_{\mathcal{F}}^2 = O(d^2/n), \tag{2.21}$$

where the last equality is due to $\max_\ell \mathbb{E}\left\|Z_1^{(j)}\right\|_{\mathcal{F}}^2 < \infty$.

$$\begin{aligned}\mathbb{E}\left\|\left(\frac{1}{n}\sum_{i=1}^{n} F_i\right)\left(\frac{1}{n}\sum_{i=1}^{n} G_i\right)\right\|_{\mathcal{F}} &= \mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n} F_i\right\|_{\mathcal{H}_{\mathcal{X}}^{+}}\left\|\frac{1}{n}\sum_{i=1}^{n} G_i\right\|_{\mathcal{H}_{\mathcal{Y}}} \\ &\leq \left(\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n} F_i\right\|_{\mathcal{H}_{\mathcal{X}}^{+}}^2\right)^{1/2}\left(\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n} G_i\right\|_{\mathcal{H}_{\mathcal{Y}}}^2\right)^{1/2}\end{aligned}$$

In the similar way to (2.21), we have

$$\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}F_i\right\|^2_{\mathcal{H}_{\mathcal{X}}^+} = O(d^2/n),$$
$$\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}G_i\right\|^2_{\mathcal{H}_{\mathcal{Y}}} = O(1/n),$$

which give us

$$\mathbb{E}\left\|\left(\frac{1}{n}\sum_{i=1}^{n}F_i\right)\left(\frac{1}{n}\sum_{i=1}^{n}G_i\right)\right\|_{\mathcal{F}} = \mathcal{O}\left(d/\sqrt{n}\right).$$

From (2.20), we have $\mathbb{E}\left\|\widehat{\mathbf{R}}^{(n)}_{Y\mathbf{X}} - \mathbf{R}_{Y\mathbf{X}}\right\|_{HS} = \mathcal{O}\left(d/\sqrt{n}\right)$ and using Chebyshev's inequality completes the proof. ■

With Lemma 2.2.1 and the fact that

$$\left\|\widehat{\mathbf{R}}^{(n)}_{Y\boldsymbol{X}}\right\| - \|\mathbf{R}_{Y\boldsymbol{X}}\| \leq \left\|\widehat{\mathbf{R}}^{(n)}_{Y\boldsymbol{X}} - \mathbf{R}_{Y\boldsymbol{X}}\right\| \leq \left\|\widehat{\mathbf{R}}^{(n)}_{Y\boldsymbol{X}} - \mathbf{R}_{Y\boldsymbol{X}}\right\|_{HS},$$

we have that

$$\left\|\widehat{\mathbf{R}}^{(n)}_{Y\boldsymbol{X}}\right\| = \|\mathbf{R}_{Y\boldsymbol{X}}\| + \mathcal{O}_p\left(dn^{-1/2}\right).$$

Similarly, we can obtain the following results.

$$\left\|\widehat{R}^{(n)}_{YY}\right\| = \|R_{YY}\| + \mathcal{O}_p\left(n^{-1/2}\right),$$
$$\left\|\widehat{\mathbf{R}}^{(n)}_{\boldsymbol{X}\boldsymbol{X}}\right\| = \|\mathbf{R}_{\boldsymbol{X}\boldsymbol{X}}\| + \mathcal{O}_p\left(d^2n^{-1/2}\right).$$

Lemma 2.2.2

$$\left\|\left(\widehat{R}^{(n)}_{YY} + \epsilon_n I\right)^{-1/2} - \left(R_{YY} + \epsilon_n I\right)^{-1/2}\right\| = \mathcal{O}_p\left(\epsilon_n^{-2}n^{-1/2}\right).$$

Proof Due to the equality $A^{-1/2}-B^{-1/2} = A^{-1/2}(B^{3/2}-A^{3/2})B^{-3/2}+(A-B)B^{-3/2}$, we have

$$
\begin{aligned}
&\left\|\left(\widehat{\mathbf{R}}_{YY}^{(n)}+\epsilon_n I\right)^{-1/2}-(\mathbf{R}_{YY}+\epsilon_n I)^{-1/2}\right\| \\
=&\left\|\left\{\left(\widehat{\mathbf{R}}_{YY}^{(n)}+\epsilon_n I\right)^{-1/2}\left[(\mathbf{R}_{YY}+\epsilon_n I)^{3/2}-\left(\widehat{\mathbf{R}}_{YY}^{(n)}+\epsilon_n I\right)^{3/2}\right]+\left(\widehat{\mathbf{R}}_{YY}^{(n)}-\mathbf{R}_{YY}\right)\right\}(\mathbf{R}_{YY}+\epsilon_n I)^{-3/2}\right\| \\
\leq&\left\|\left(\widehat{\mathbf{R}}_{YY}^{(n)}+\epsilon_n I\right)^{-1/2}\left[(\mathbf{R}_{YY}+\epsilon_n I)^{3/2}-\left(\widehat{\mathbf{R}}_{YY}^{(n)}+\epsilon_n I\right)^{3/2}\right]+\left(\widehat{\mathbf{R}}_{YY}^{(n)}-\mathbf{R}_{YY}\right)\right\|\left\|(\mathbf{R}_{YY}+\epsilon_n I)^{-3/2}\right\| \\
\leq&\left(\left\|\left(\widehat{\mathbf{R}}_{YY}^{(n)}+\epsilon_n I\right)^{-1/2}\left[(\mathbf{R}_{YY}+\epsilon_n I)^{3/2}-\left(\widehat{\mathbf{R}}_{YY}^{(n)}+\epsilon_n I\right)^{3/2}\right]\right\|+\left\|\widehat{\mathbf{R}}_{YY}^{(n)}-\mathbf{R}_{YY}\right\|\right)\left\|(\mathbf{R}_{YY}+\epsilon_n I)^{-3/2}\right\| \\
\leq&\left(\left\|\left(\widehat{\mathbf{R}}_{YY}^{(n)}+\epsilon_n I\right)^{-1/2}\right\|\left\|\left[(\mathbf{R}_{YY}+\epsilon_n I)^{3/2}-\left(\widehat{\mathbf{R}}_{YY}^{(n)}+\epsilon_n I\right)^{3/2}\right]\right\|+\left\|\widehat{\mathbf{R}}_{YY}^{(n)}-\mathbf{R}_{YY}\right\|\right)\left\|(\mathbf{R}_{YY}+\epsilon_n I)^{-3/2}\right\| \\
\leq&\left(\left\|\left(\widehat{\mathbf{R}}_{YY}^{(n)}+\epsilon_n I\right)^{-1/2}\right\|3\max\left\{\|\mathbf{R}_{YY}+\epsilon_n I\|^{3/2},\left\|\widehat{\mathbf{R}}_{YY}^{(n)}+\epsilon_n I\right\|^{3/2}\right\}\left\|\widehat{\mathbf{R}}_{YY}^{(n)}-\mathbf{R}_{YY}\right\|+\left\|\widehat{\mathbf{R}}_{YY}^{(n)}-\mathbf{R}_{YY}\right\|\right)\left\|(\mathbf{R}_{YY}+\epsilon_n I)^{-3/2}\right\| \\
\leq&\left(\frac{3}{\sqrt{\epsilon_n}}\max\left\{\|\mathbf{R}_{YY}+\epsilon_n I\|^{3/2},\left\|\widehat{\mathbf{R}}_{YY}^{(n)}+\epsilon_n I\right\|^{3/2}\right\}\left\|\widehat{\mathbf{R}}_{YY}^{(n)}-\mathbf{R}_{YY}\right\|+\left\|\widehat{\mathbf{R}}_{YY}^{(n)}-\mathbf{R}_{YY}\right\|\right)\left\|(\mathbf{R}_{YY}+\epsilon_n I)^{-3/2}\right\| \\
=&\left(\frac{3}{\sqrt{\epsilon_n}}\max\left\{\|\mathbf{R}_{YY}+\epsilon_n I\|^{3/2},\left\|\widehat{\mathbf{R}}_{YY}^{(n)}+\epsilon_n I\right\|^{3/2}\right\}+1\right)\left\|\widehat{\mathbf{R}}_{YY}^{(n)}-\mathbf{R}_{YY}\right\|\epsilon_n^{-3/2} \\
=&\mathcal{O}\left(\epsilon_n^{-2}n^{-1/2}\right).
\end{aligned}
$$

The fourth inequality holds due to Lemma 8 in [30]. ■

Lemma 2.2.3

$$\left\|\left(\widehat{\mathbf{R}}_{\mathbf{XX}}^{(n)}+\epsilon_n I\right)^{-1}-\left((\mathbf{R}_{\mathbf{XX}}+\epsilon_n I)^{-1}\right)\right\|=\mathcal{O}_p\left(d^2\epsilon_n^{-2}n^{-1/2}\right).$$

Before we state and prove the following Lemmas, we first define the following two operators:

$$
\begin{aligned}
\mathbf{V}_{Y\boldsymbol{X}}^{(\epsilon)} &:= (\mathbf{R}_{YY}+\epsilon_n I)^{-1/2}\mathbf{R}_{Y\boldsymbol{X}}(\mathbf{R}_{\boldsymbol{XX}}+\epsilon_n I)^{-1/2}, \\
\mathbf{V}_{\boldsymbol{X}Y}^{(\epsilon)} &:= (\mathbf{R}_{\boldsymbol{XX}}+\epsilon_n I)^{-1/2}\mathbf{R}_{\boldsymbol{X}Y}(\mathbf{R}_{YY}+\epsilon_n I)^{-1/2}.
\end{aligned}
$$

Then, we have

$$\mathbf{V}_{Y\boldsymbol{X}}^{(\epsilon)}\mathbf{V}_{\boldsymbol{X}Y}^{(\epsilon)} = (R_{YY}+\epsilon_n I)^{-1/2}\mathbf{R}_{Y\boldsymbol{X}}(\mathbf{R}_{\boldsymbol{XX}}+\epsilon_n I)^{-1}\mathbf{R}_{\boldsymbol{X}Y}(R_{YY}+\epsilon_n I)^{-1/2}.$$

Lemma 2.2.4 *When $d=\mathcal{O}\left(n^{1/4}\right)$, we have that for a sequence $\epsilon_n\to 0$ as $n\to\infty$,*

$$\left\|\widehat{\mathbf{V}}_{Y\boldsymbol{X}}^{(n)}\widehat{\mathbf{V}}_{\boldsymbol{X}Y}^{(n)}-\mathbf{V}_{Y\boldsymbol{X}}^{(\epsilon)}\mathbf{V}_{\boldsymbol{X}Y}^{(\epsilon)}\right\|=\mathcal{O}_p\left(d^2\epsilon_n^{-7/2}n^{-1/2}\right).$$

Proof Let $a = (R_{YY} + \epsilon_n I)^{-1/2}$, $b = \mathbf{R}_{Y\boldsymbol{X}}$, $H = (\mathbf{R}_{\boldsymbol{XX}} + \epsilon_n I)^{-1}$, $a_n = (\widehat{R}^{(n)}_{YY} + \epsilon_n I)^{-1/2}$, $b_n = \widehat{\mathbf{R}}^{(n)}_{YX}$, $H_n = (\widehat{\mathbf{R}}^{(n)}_{\boldsymbol{XX}} + \epsilon_n I)^{-1}$ and A^* represents the adjoint operator of A.

$$\begin{aligned}
&\|a_n b_n H_n b_n^* a_n\| - \|abHb^*a\| \\
\leq& \|(a_n - a)b_n H_n b_n^*(a_n - a)\| + 2\|(a_n - a)b_n H_n b_n^* a\| + \|a(b_n H_n b_n^* - bHb^*)a\| \\
\triangleq& S_1 + S_2 + S_3
\end{aligned}$$

From Lemma 2.2.1 to Lemma 2.2.3, we have the following inequalities for S_1, S_2 and S_3.

$$\begin{aligned}
S_1 &\leq \|a_n - a\|^2 \|b_n\|^2 \|H_n\| \\
&= \mathcal{O}_p\left(\left(\epsilon_n^{-2} n^{-1/2}\right)^2 \cdot \epsilon_n^{-1}\right) \\
&= \mathcal{O}_p\left(\epsilon_n^{-5} n^{-1}\right).
\end{aligned} \tag{2.22}$$

$$\begin{aligned}
S_2 &\leq \|a_n - a\| \|b_n\|^2 \|H_n\| \|a\| \\
&= \mathcal{O}_p\left(\epsilon_n^{-2} n^{-1/2} \cdot \epsilon_n^{-1} \cdot \epsilon_n^{-1/2}\right) \\
&= \mathcal{O}_p\left(\epsilon_n^{-7/2} n^{-1/2}\right).
\end{aligned} \tag{2.23}$$

$$\begin{aligned}
S_3 &\leq \|a\|^2 \|b_n H_n b_n^* - bHb^*\| \\
&\leq \|a\|^2 \left(\|(b_n - b)H_n(b_n - b)^*\| + 2\|b_n H_n b^*\| + \|b(H_n - H)b^*\|\right) \\
&\leq \|a\|^2 \left(\|(b_n - b)\|^2 \|H_n\| + 2\|b_n\| \|H_n\| \|b\| + \|b\|^2 \|H_n - H\|\right) \\
&\leq \mathcal{O}_p\left(d^2 \epsilon_n^{-3} n^{-1/2}\right)
\end{aligned} \tag{2.24}$$

Then Lemma 2.2.4 follows by combining (2.22) - (2.24). ■

Lemma 2.2.5 *Assume* $\mathbf{V}_{Y\boldsymbol{X}}$ *is compact. Then, as* $n \to \infty$*, for a sequence* $\epsilon_n \to 0$*,*

$$\left\|\mathbf{V}^{(\epsilon)}_{Y\boldsymbol{X}} \mathbf{V}^{(\epsilon)}_{\boldsymbol{X}Y} - \mathbf{V}_{Y\boldsymbol{X}} \mathbf{V}_{\boldsymbol{X}Y}\right\| \xrightarrow{P} 0.$$

Proof From Lemma 7 in [30], we have $\left\|\mathbf{V}^{(\epsilon)}_{Y\mathbf{X}} - \mathbf{V}_{Y\mathbf{X}}\right\|_{op} \xrightarrow{P} 0$ as $n \to \infty$. Similarly, we have $\left\|\mathbf{V}^{(\epsilon)}_{\mathbf{X}Y} - \mathbf{V}_{\mathbf{X}Y}\right\|_{op} \xrightarrow{P} 0$. Then,

$$\begin{aligned}
\left\|\mathbf{V}^{(\epsilon)}_{Y\mathbf{X}} \mathbf{V}^{(\epsilon)}_{\mathbf{X}Y} - \mathbf{V}_{Y\mathbf{X}} \mathbf{V}_{\mathbf{X}Y}\right\| &= \left\|\left(\mathbf{V}^{(\epsilon)}_{Y\mathbf{X}} - \mathbf{V}_{Y\mathbf{X}}\right)\mathbf{V}^{(\epsilon)}_{\mathbf{X}Y} + \mathbf{V}_{Y\mathbf{X}}\left(\mathbf{V}^{(\epsilon)}_{\mathbf{X}Y} - \mathbf{V}_{\mathbf{X}Y}\right)\right\| \\
&\leq \left\|\mathbf{V}^{(\epsilon)}_{Y\mathbf{X}} - \mathbf{V}_{Y\mathbf{X}}\right\| \left\|\mathbf{V}^{(\epsilon)}_{\mathbf{X}Y}\right\| + \|\mathbf{V}_{Y\mathbf{X}}\| \left\|\mathbf{V}^{(\epsilon)}_{\mathbf{X}Y} - \mathbf{V}_{\mathbf{X}Y}\right\|
\end{aligned}$$

The result follows as $\left\|\mathbf{V}_{\mathbf{X}Y}^{(\epsilon)}\right\| \leq 1$ and $(\mathbf{V}_{Y\mathbf{X}}) \leq 1$. ■

2.3 Simulation Study

In this section, we evaluate the effectiveness of the proposed OKGT method using synthetic data sets. We use R^2 as the performance measure for OKGT. In our first simulation example, we show the effectiveness of OKGT in recovering the true function structure from the data generated from a given model. In our second experiment, we demonstrate the gain by using a proper group structure for OKGT.

2.3.1 Effectiveness on Synthetic Data

In this experiment, we apply OKGT on synthetic data simulated from a model with known group structure. In particular, we assume the following function as the true model

$$Y = \ln\left(4 + \sin(2\pi X_1) + |X_2| + X_3^2 + X_4^3 + X_5 + X_6 * X_7 + 0.1\epsilon\right) \tag{2.25}$$

where the variables X_1 to X_5 each forms an univariate group and X_6 and X_7 form a bivariate group through their product. The predictor variables $X_j, j = 1, \ldots, 7$, are independent and identically distributed as Unif$(-1, 1)$. The error term ϵ is standard normal. We assume the true structure is known, that is X_1 to X_5 each forms a univariate group and (X_6, X_7) is a bivariate group, and expect our algorithm to recover the functional forms for those groups, especially for the interaction between X_6 and X_7.

We use Laplace kernel $k(x, y) = \exp\{-\sigma \|x - y\|\}$ with a fixed bandwidth $\sigma = 0.5$ for all the groups. The regularization parameter ϵ_n for estimating the optimal transformations is set at 0.01. We generate one set of data with sample size 500 from model (2.25) and apply OKGT, which results in an R^2 value equal to 0.909. Figure 2.1 shows the univariate transformations for the variables X_1 to X_5. Figure 2.2 shows the bivariate transformation for the variables X_6 and X_7 as a group.

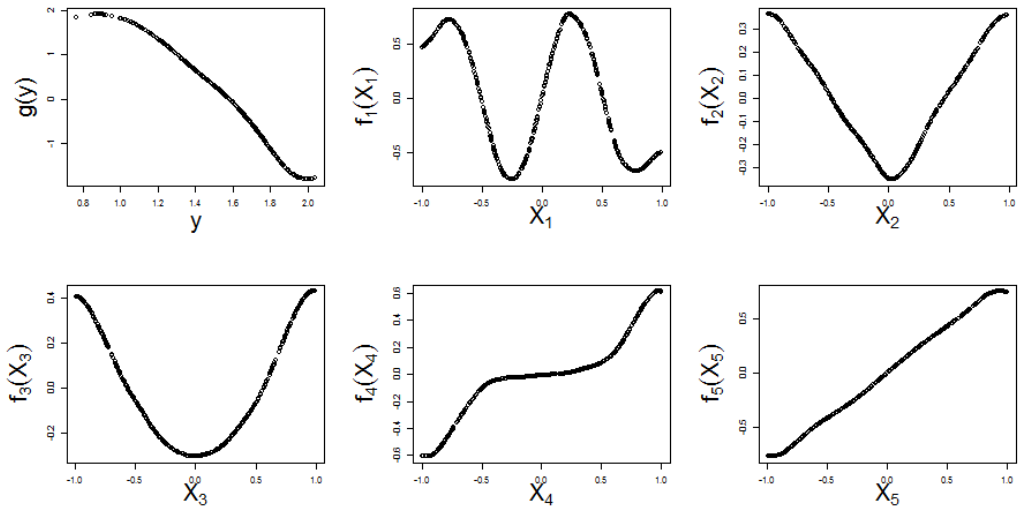

Fig. 2.1. Optimal transformations of the variables from X_1 to X_5 in model (2.25) by applying OKGT.

From Figures 2.1 and 2.2, we can see that OKGT successfully recovers all the function forms of the univariate variables from X_1 to X_5. It also clearly reveals the interaction between X_6 and X_7 as $f_6(X_6, X_7) = X_6 * X_7$.

We also apply the additive univariate transformation where each predictor forms its own group. The resulting R^2 equals to 0.8368, which is lower than that from the group structure mentioned above. Furthermore, the univariate transformations of X_6 and X_7 fail to capture their interaction.

2.3.2 Impact of Group Structure

In this experiment, we investigate the effect of different group structures on the model fitting for OKGT. It can be conjectured that a fully nonparametric model, i.e. $d = 1$ in OKGT, would not give a good fit in terms of R^2 because of the limited sample size and complex function space where the algorithm searches a solution. Besides, applying a fully nonparametric model usually produces a result that is hard

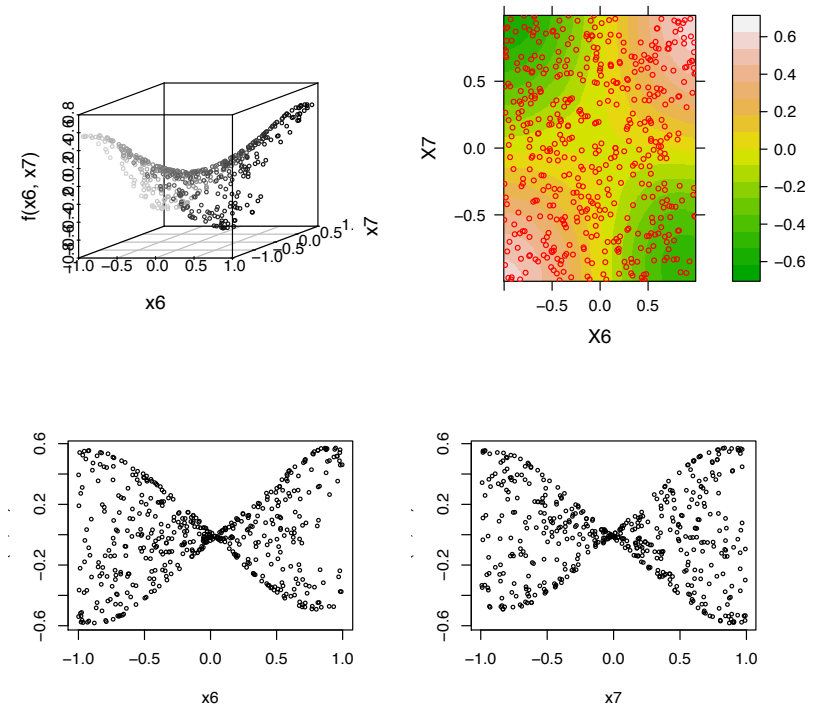

Fig. 2.2. Optimal transformation of the grouped variables X_6 and X_7 in model (2.25) by applying OKGT. Top-left: 3-D scatter plot. Top-right: Smoothed contour plot with data points. Bottom-left: 2-D projection of X_6 versus $f_6(X_6, X_7)$. Bottom-right: 2-D projection of X_7 versus $f_6(X_6, X_7)$.

to interpret. On the other hand, imposing a fully additive structure, i.e. $d = p$, may be too restrictive to eliminate any possible interaction between predictor variables and would cause excessive information loss. Thus, a compromise is needed to balance fitting efficiency and interpretability of the result.

The setting of the experiment is given as follows. We generate 500 i.i.d. observations from the model:

$$Y = \left(5 + \sin\left(X_1 X_2\right) + |X_3 X_4| + X_5^{X_6} + (X_7 - X_8)_+ + \frac{X_9}{X_{10} + 0.1} + 0.1\epsilon\right)^2. \quad (2.26)$$

where the values of X_1 through X_{10} are sampled from Unif$(0, 2)$, the error ϵ is standard normal, and $(a)_+$ denotes the maximum of 0 and a. In each simulation, we apply OKGT under each of the following six group structures:

1) $g(Y) \leftarrow f(X_1, X_2, \cdots, X_{10})$;
2) $g(Y) \leftarrow f_1(X_1, \cdots, X_5) + f_2(X_6, \cdots, X_{10}))$;
3) $g(Y) \leftarrow f_1(X_1, \cdots, X_4) + f_2(X_5, \ldots, X_8) + f_3(X_9, X_{10})$;
4) $g(Y) \leftarrow f_1(X_1, X_2, X_3) + f_2(X_4, X_5, X_6) + \cdots + f_4(X_{10})$;
5) $g(Y) \leftarrow f_1(X_1, X_2) + f_2(X_3, X_4) + \cdots + f_5(X_9, X_{10})$;
6) $g(Y) \leftarrow f_1(X_1) + f_2(X_2) + \cdots + f_{10}(X_{10})$.

If a group structure is able to include interacting variables in the same group, we call it a *correct* group structure, otherwise it is called an *incorrect* group structure. If any further partition renders an existing group structure incorrect, we call the structure as the *intrinsic* group structure. Note that any combinations of groups from a correct group structure A will yield another correct group structure B, we call B as an *inherited* group structure from A. According to the definitions, the group structures 2, 4, 6 are incorrect and the group structures 1, 3 and 5 are correct ones. Group structure 5 is the intrinsic group structure and group structures 1 and 3 are inherited from group structure 5.

In simulation, the Laplace kernel is used with bandwidth being 0.5, and the regularization parameter ϵ_n is set to 0.01. By repeating this procedure 100 times, we obtain Figure 2.3 which shows the side-by-side boxplots of the R^2 from the six group structures after applying OKGT.

From Figure 2.3, we noticed that for OKGT with the correct group structures (except for group structure 1), the average R^2 is larger than those with the incorrect group structures. Group structures 3 and 5 achieve the maximum average of $R^2 \approx 0.94$, with correct structure specifications. While group 1 represents the most flexible structure which in theory can accommodate any model, its fitting result ($R^2 \approx 0.90$) is not as good as that of other correct group structures. With incorrect group structure 6, which assumes a fully additive structure, the fitting result of OKGT is the worst

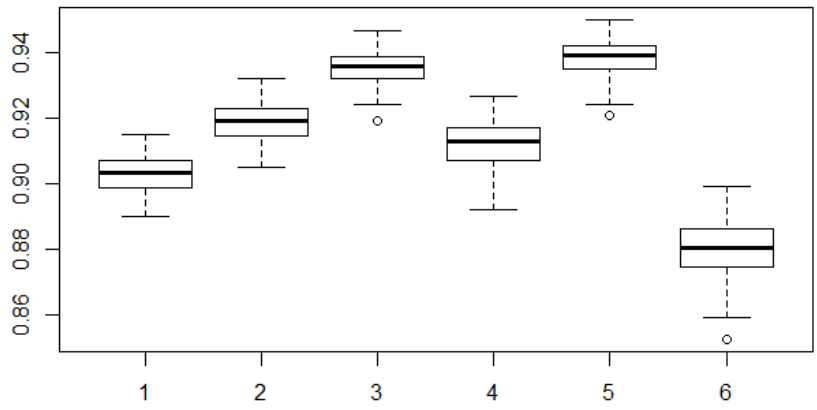

Fig. 2.3. Boxplots of R^2 for different number of groups when applying OKGT on the sample from model (2.26) under six different group structures.

($R^2 \approx 0.88$). It is interesting to observe that though group structures 2 and 4 are incorrect, their average R^2's are even higher the that from group structure 1.

This phenomenon demonstrates that, with limited observations, the grouping effect (different R^2) is the result of the interplay of three factors: 1) group structure specification; 2) group size; and 3) the number of groups. For group structure specification, it is expected that the fitting result from a correct group structure is generally better than that from an incorrect group structure; For group size, estimating a single function which contains many variables suffers from the curse of dimensionality, thus the fitting efficiency is generally lower than estimating a single function with less variables. The number of groups determines the number of functions to be estimated. Estimating more functions will accumulate the losses on fitting efficiency.

Assume that all different group structures are correct, then there will be a trade-off between group size and the number of groups on fitting efficiency. For example, when $d = 1$, though function size is small, we need to estimate a function containing all variables and the estimation efficiency is low for large p due to the curse of dimensionality. It could be even worse than the case where the group structure is

incorrect. For the fully additive structure $d = p$, though group size is small, we need to estimate p functions in total and fitting efficiency will decrease with the increase of the number of functions to be estimated. Therefore, a balance between the group size and the number of groups is preferable.

It can also been seen in Figure 2.3 that the result from group structure 3 is almost as good as that of the intrinsic structure (group structure 5). This indicates that under finite sample, a more flexible group structure can approximate the intrinsic structure without too much information loss measured by R^2. This justifies the generalization of optimal transformation by using a group structure.

2.4 Real Data Applications

In this section, We compare the performance of OKGT with different group structures on two real datasets, the SkillCraft1 Master data from UCI Machine Learning Repository[4] and the glioblastoma multiforme data from the TCGA Data Coordinating Center[5].

2.4.1 SkillCraft1 Master data

In this experiment, we apply OKGT on the SkillCraft1 Master data set. This is a video game telemetry data from real-time strategy (RTS) games and was originally used in [37] to explore the development of expertise. The study of the development of RTS expertise is of interest because the knowledge learned can be applied in other domains. The data was collected from 3395 Star Craft 2 players ranging over 7 levels of expertise from novices to full-time professionals. The levels are coded by the leagues in which they compete, and are coded from 1 to 8 as ordinal data. For each player, a replay file recorded all the commands issued in the game and the data of game related variables are calculated from the replay file. Some game related vari-

[4] `http://archive.ics.uci.edu/ml/`

[5] `https://tcga-data.nci.nih.gov/tcga/`

ables include action per minute, number of unique hotkeys used per timestamp, and number right-clicks on minimap per timestamp. Our goal is to find some interesting relationships between game related predictor variables and the league index as the response variable.

Before applying our method, we randomly select a subsample of size 500. There are 19 predictor variables and we only use the 15 game related variables in this experiment. By using all of the 15 predictors and imposing a fully additive structure ($d = p$), the resulting R^2 is 0.8454. A single group structure ($d = 1$) results in an R^2 of 0.6666. The first eight plots in Figure 2.4 show the transformations of the response and the seven predictors with the large variance $\mathrm{Var}(f_\ell(X_\ell))$. The last plot in the figure shows a scatter plot of $\hat{g}^*(y_i)$ and $\sum_{\ell=1}^{15} \hat{f}_\ell^*(x_{i\ell})$. The red lines are the loess smoothing curves. The last plot shows a fairly linear relationship between the transformed response and the sum of all transformed predictors, indicating an overall good fit using OKGT with the additive structure.

From Figure 2.4, the transformations are highly nonlinear for both the response and the seven predictors. This graphics can provide some meaningful interpretations. For example, the transformation of the response variable LeagueIndex shows a S-shaped pattern, indicating that the acquittance of skill is not linear. The improvement of skill from level 1 to 2 and from 7 to 8 are more significant than at the other levels. The transformation of APM shows a similar pattern as that of the response. Between 100 to 150, APM is roughly independent of the skill levels, whereas in the lower and higher range some linear pattern is shown. The transformation of GapBetweenPACs shows an overall decreasing pattern. However, the curve drops dramatically before 30 milliseconds but decreases slowly after that. The overall decreasing pattern indicates that players with higher LeagueIndex generally have lower GapBetweenPACs. This transformation can be interpreted as follows. For different players with large GapBetweenPACs, their skills will not change so rapidly. Once they reach the level with GapBetweenPACs as small as 30 milliseconds, it will require

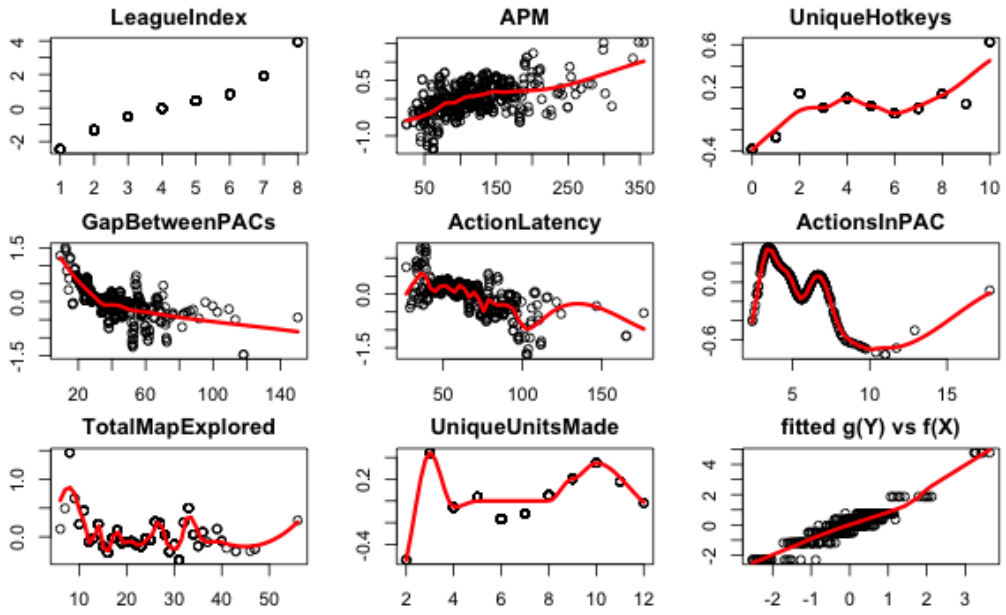

Fig. 2.4. Application of OKGT on SkillCraft1 data. First eight figures are transformation of the response and seven variables presenting large fitted norm. The last figure is scatter plot of $\hat{g}^*(y_i)$ and $\sum_{j=1}^{15} \hat{f}_\ell^*(\mathbf{x}_{i\ell})$ by OKGT with all variables. The red curves are the loess smoothing applied on the transformations.

a huge improvement in skills to achieve a small saving in time between PACs. Thus, LeagueIndex drops dramatically within that range.

The predictor variables can be partitioned into different groups according to different types of skill in gaming. Based on this observation, we partition the predictors into the following seven groups.

- APM
- SelectByHotkeys AssignToHotkeys UniqueHotkeys
- MinimapAttacks MinimapRightClicks
- NumberOfPACs GapBetweenPACs ActionLatency ActionsInPAC
- TotalMapExplored
- WorkersMade UniqueUnitsMade ComplexUnitsMade
- ComplexAbilitiesUsed

Applying OKGT with the group structure defined above, we can achieve an R^2 of 0.8675, which is higher than the R^2 from fitting the fully additive structure. This improvement clearly supports the advantages provided by useful grouping.

2.4.2 TCGA glioblastoma multiforme data

In this example, we consider modeling the survival time of patients with glioblastoma, which is the first cancer studied by The Cancer Genome Atlas (TCGA). The dataset we use contains the expression levels of 12042 genes and the survival time (length) of 400 Glioblastoma patients. A smaller sample (206 patients) was considered in [38]. We are interested in identifying important genes associated with patients' survival time and in investigating their relationship, to improve our understanding of the underlying biology of gliomas.

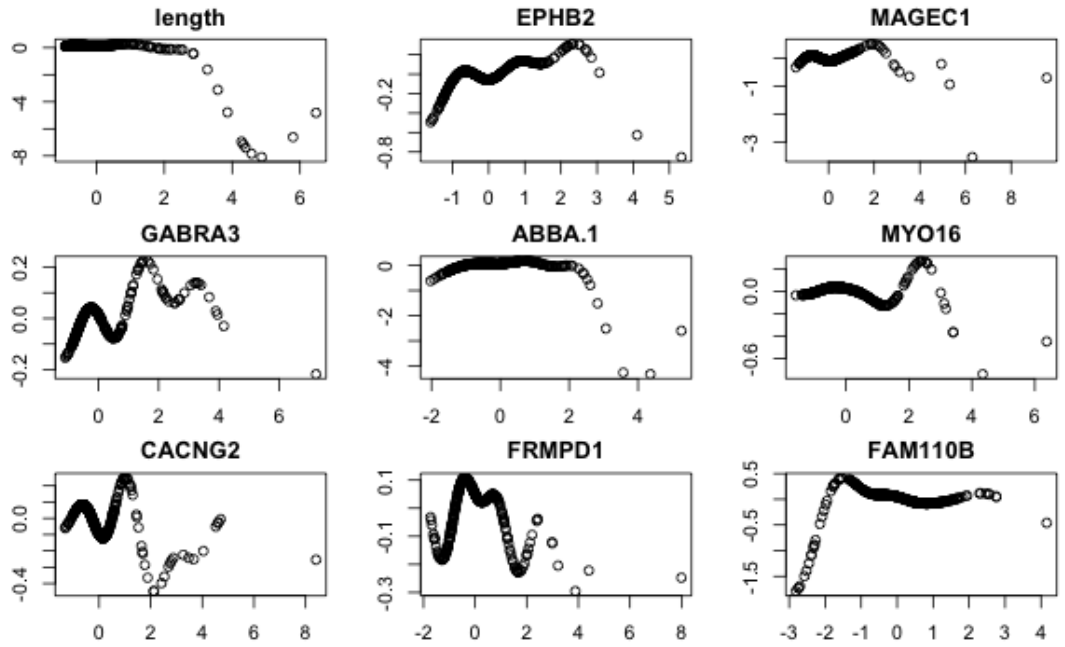

Fig. 2.5. Application of OKGT on TCGA glioblastoma data. Transformations of the response and the first eight variables after fitting 30 top ranked variables.

We first apply OKGT to the response and each gene as a predictor marginally and then rank all the genes according to their R^2 values in descending order. We keep the top 30 genes with the largest R^2 values. By imposing a fully additive structure ($d = 30$) on the 30 retained predictors and preforming OKGT, the resulting R^2 is 0.8510. If a single group structure ($d = 1$) is used, the R^2 is 0.7142.

To compare the effect of different group structures, we conduct the following experiment. The 30 retained gene predictors are randomly partitioned into a fixed number of groups of equal size. The number of groups (d) in this experiment is set at 15, 10, 5, 3, and 2, which is corresponding to having 2, 3, 6, 10, and 15 predictors in each group. The variables in each group are randomly assigned. By applying OKGT under a random group structure using the original data, we can obtain an estimate of R^2. The boxplots in Figure 2.6 are based on 100 simulations at each fixed number of groups.

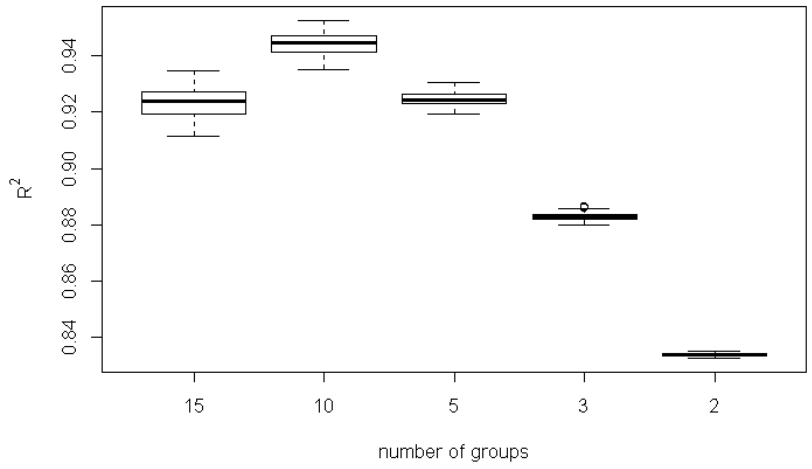

Fig. 2.6. Boxplots of R^2 for different number of groups when applying OKGT on the TCGA glioblastoma data with top ranked 30 genes.

From Figure 2.6, we notice that even with random grouping, OKGT can easily achieve a higher R^2 compared with that with the fully additive structure $d = p$. This experiment further supports that grouping can be advantageous, due to the trade-off on fitting efficiency between function complexity and group size.

2.5 Summary

In this chapter, we have developed an effective kernel method called OKGT for achieving the general goal of multivariate regression analysis, which is to explore the relationship of a response variable Y and a predictor vector X. In simulation study and real data applications, the OKGT method outperforms the optimal univariate transformation method (i.e. ACE) as well as multivariate nonparametric regression. The reason for OKGT's excellent performance is because it can either take advantage of existing group structures of the predictor variables or it can be used to recover the hidden group structure. The use of cross-covariance and conditional covariance operators and their empirical counterparts much simplifies both the theoretical and numerical analysis of the OKGT method, demonstrating their power for high dimensional data exploration.

There are three immediate directions to further improve the OKGT method. First, a more effective and efficient procedure is needed for the OKGT method to detect intrinsic group structure among the predictor variables. We have developed an procedure for additive group structure identification, which will be reported in Chapter 3. Second, after the optimal kernel group transformations are estimated, how to use graphics to reliably infer the relationship between Y and $\boldsymbol{X}$ needs to be further studied. Third, when the dimension of $\boldsymbol{X}$ is high or extremely high, penalization will probably be needed to make the OKGT method stable and effective in exploring Y and $\boldsymbol{X}$.

3. ADDITIVE GROUP STRUCTURE IDENTIFICATION

In the previous chapter, we proposed the Optimal Kernel Group Transformation (OKGT) method for exploring the relationship between Y and $\boldsymbol{X}$. The method considers an additive structure of groups of predictor variables instead of an additive structure of individual predictor variables. It was observed from the numerical studies that changing the additive group structure resulted in different estimation performance of OKGT. So using a proper additive group structure for OKGT is crucial for effective data exploration.

In this chapter, we develop a general framework to simultaneously identify the optimal additive group structure and fit the nonparametric regression functions for each group of predictor variables, using kernel methods. The main idea is to add an additional penalty that controls the complexity of additive group structures to the usual penalized risk function (see Equation (3.21)). This new penalty function is motivated by the complexity measures of Reproducing Kernel Hilbert Spaces (RKHSs), and it penalizes more complex structures and favors the true structures. We further develop two algorithms, one of which uses exhaustive search and the other employs a backward stepwise search, for identifying true additive group structures under the small p and large p scenarios, respectively. Extensive simulation study and real data applications show that our proposed method can successfully recover the true additive group structures in a variety of models.

The most similar work to ours is [39] which introduced a novel set of constraints on the weight vectors for Projection Pursuit Regression (PPR) so that partitioning of $\boldsymbol{X}$ is enforced while the linearity is retained. Since there is no linear constraint imposed on each group of predictor variables, our method is more general than that in [39]. In order to divide the predictor variables into groups, [39] also relies on some prior knowledge. By introducing the novel penalty for the complexity of group

structure, our method provides a more principled way for additive group structure identification.

The rest of the chapter is organized as follows. In Section 3.1, we formalize the problem of Additive Group Structure Identification (AGSI) for nonparametric regression. A brief review of covering number will be given, which provides the motivation for our proposed penalty. Section 3.2 provides the implementation details and algorithms. We provide the theory of selection consistency for AGSI in Section 3.3. The experimental results based on simulation studies and real data applications are reported in Section 3.4 and 3.5. Section 3.6 summarizes this chapter with some discussion.

3.1 Methodology

In this section, we formally define the problem of Additive Group Structure Identification. This will require the definitions of some basic but important concepts. First, the concept of additive group structure is rigorously defined and its implication in L^2 space and Reproducing Kernel Hilbert Spaces (RKHSs) are formalized. Second, the idea of controlling the complexity of an additive group structure is concretized through the discussion of various function space capacity measures. Finally, the finite sample version of AGSI is formulated along with the discussion on its estimation.

3.1.1 Additive Group Structures

Throughout this chapter, we assume that the transformation function for the response is known and given as $h(Y)$. That is the optimal response transformation g^* in OKGT is assumed to be known. Currently, we rely on this assumption to establish the selection consistency. Without loss of generality, we will simply use Y instead of $h(Y)$ in the following discussion.

We consider an additive model with a group structure G. A group structure for non-parametric regression is defined as a particular partition of the indices of the predictor variables $\boldsymbol{X} = \{X_1, \ldots, X_p\}$.

Definition 3.1.1 *Let $\boldsymbol{I} = \{1, \ldots, p\}$ be the set of indices of the predictor variables in $\boldsymbol{X}$ and $G := \{\boldsymbol{u}_i\}_{i=1}^d$ be a particular partition of $\boldsymbol{I}$, that is, $\boldsymbol{u}_i \cap \boldsymbol{u}_j = \emptyset$ if $i \neq j$ and $\bigcup_{i=1}^d \boldsymbol{u}_i = \boldsymbol{I}$. We refer G as a **group structure** and each $\boldsymbol{u}$ as a **group**. The collection of all possible group structures, that is all possible partitions of $\boldsymbol{I}$, is denoted as $\mathcal{G}$.*

If there exists a group structure G such that

$$\mathbb{E}[Y|\boldsymbol{X} = \boldsymbol{x}] = f_1(x_i; i \in \boldsymbol{u}_1) + \ldots + f_d(x_i; i \in \boldsymbol{u}_d),$$

we say that $\mathbb{E}[Y|\boldsymbol{X} = \boldsymbol{x}]$ admits the **additive group structure** G. Obviously, the usual additive model is a special case with the additive group structure consisting of only univariate groups, i.e. $G = \{(1), \ldots, (p)\}$.

Consider the following model $Y = 2+3X_1+1/(1+X_2^2+X_3^2)+\arcsin((X_4+X_5)/2)+\epsilon$, where ϵ is the error independent of $\boldsymbol{X}$ and has 0 mean. According to Definition 3.1.1, this model admits the additive group structure $G_0 = \{(1), (2,3), (4,5)\}$. Let $G_1 = \{(1,2,3), (4,5)\}$ and $G_2 = \{(1,4,5), (2,3)\}$. The model can also be said to admit the additive group structures G_1 and G_2. However, there exists a major difference between G_0, G_1 and G_2. While the groups in G_0 cannot be further divided into subgroups, both G_1 and G_2 contain groups that can be further split. We characterize this difference by defining the following partial order between different group structures.

Definition 3.1.2 *Let G and G' be two additive group structures for the predictor variables in $\boldsymbol{X}$. If for every group $\boldsymbol{u} \in G$ there is a group $\boldsymbol{v} \in G'$ such that $\boldsymbol{u} \subseteq \boldsymbol{v}$, then G is called a **sub group structure** of G'. This relation is denoted as $G \leq G'$.*

Definition 3.1.3 *If G is a sub group structure of G', then G' is called a **super group structure** of G, which is denoted as $G' \geq G$.*

In the previous example, G_0 is a sub group structure of either G_1 or G_2 (equivalently, both G_1 and G_2 are super group structures of G). However, the order between G_1 and G_2 is not defined.

The partial order defined for group structures in $\mathcal{G}$ also carries over to the functions spaces used in additive non-parametric regression. To facilitate our discussion, we need to formalize the following technical setup.

Let $L^2_{P_{\boldsymbol{X}_{\boldsymbol{u}}}}\left([0,1]^{|\boldsymbol{u}|}\right) := \{f \in L^2(\mathcal{X}) \mid \int_{\mathcal{X}_{\boldsymbol{u}}} |f(x)|^2 \, dP_{\boldsymbol{X}_{\boldsymbol{u}}} < \infty\}$ be a space of square integrable (w.r.t. distribution $P_{\boldsymbol{X}_{\boldsymbol{u}}}$) functions defined on a $|\boldsymbol{u}|$-dimensional unit cube, where $|\boldsymbol{u}|$ is the number of indices in group $\boldsymbol{u}$. If $f_{\boldsymbol{u}} \in L^2_{P_{\boldsymbol{X}_{\boldsymbol{u}}}}([0,1]^{|\boldsymbol{u}|})$ for every $\boldsymbol{u} \in G$, then the additive function $f = \sum_{\boldsymbol{u}\in G} f_{\boldsymbol{u}}(\boldsymbol{x}_{\boldsymbol{u}})$ (as a model for $\mathbb{E}[Y|X = x]$) is a member of the direct sum function space defined as $L^2_{P_{\boldsymbol{X}}}([0,1]^G) := \oplus_{\boldsymbol{u}\in G} L^2_{P_{\boldsymbol{X}_{\boldsymbol{u}}}}([0,1]^{|\boldsymbol{u}|})$. If $|\boldsymbol{u}| = p$, then $L^2_{P_{\boldsymbol{X}_{\boldsymbol{u}}}}([0,1]^{|\boldsymbol{u}|}) = L^2_{P_{\boldsymbol{X}}}([0,1]^p)$ and f is a fully non-parametric function. For notational convenience, $L^2_{P_{\boldsymbol{X}_{\boldsymbol{u}}}}([0,1]^{|\boldsymbol{u}|})$ and $L^2_{P_{\boldsymbol{X}}}([0,1]^G)$ will be simplified as $L^2_{\boldsymbol{u}}$ and L^2_G respectively whenever no confusion is raised.

With the function spaces defined above, we have the following two theorems which describes the relationship between group structures and their induced function spaces.

Theorem 3.1.1 *Let G_1 and G_2 be two additive group structures. If $G_1 \leq G_2$, then $L^2_{G_1} \subseteq L^2_{G_2}$.*

Proof Since $f \in L^2_{G_1}$, we have $f = \sum_{\boldsymbol{u}\in G_1} f_{\boldsymbol{u}}(\boldsymbol{x}_{\boldsymbol{u}})$.

If $G_1 \cap G_2 \neq \emptyset$, then for each $\boldsymbol{u} \in G_1 \cap G_2$, it is true that $f_{\boldsymbol{u}} \in L^2_{G_2}$.

If $\boldsymbol{u} \notin G_1 \cap G_2$ and $\boldsymbol{u} \in G_1 \backslash G_2$, because $G_1 \leq G_2$, there exists $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k \in G_2 \backslash G_1$ for some $k < |G_2|$ such that $\boldsymbol{v} := \boldsymbol{u} \cup \boldsymbol{u}_1 \cup \cdots \cup \boldsymbol{u}_k \in G_2$. Since

$$L^2([0,1]^{|\boldsymbol{u}|}) \oplus L^2([0,1]^{|\boldsymbol{u}_1|}) \oplus \cdots \oplus L^2([0,1]^{|\boldsymbol{u}_k|}) \subseteq L^2([0,1]^{|\boldsymbol{v}|}), \tag{3.1}$$

by induction, we have the desired result.

The sub-additivity in (3.1) is true because for two groups $\boldsymbol{u}$ and $\boldsymbol{v}$ in a group structure G, we have

$$\begin{aligned}
&\int (f_{\boldsymbol{u}}(\boldsymbol{x_u}) + f_{\boldsymbol{v}}(\boldsymbol{x_v}))^2 p(\boldsymbol{x_u}, \boldsymbol{x_v}) d\boldsymbol{x_u} d\boldsymbol{x_v} \\
=&\int f_{\boldsymbol{u}}^2(\boldsymbol{x_u}) p(\boldsymbol{x_u}, \boldsymbol{x_v}) d\boldsymbol{x_u} d\boldsymbol{x_v} + \int f_{\boldsymbol{v}}^2(\boldsymbol{x_v}) p(\boldsymbol{x_u}, \boldsymbol{x_v}) d\boldsymbol{x_u} d\boldsymbol{x_v} \\
&\qquad + 2\int f_{\boldsymbol{u}}(\boldsymbol{x_u}) f_{\boldsymbol{v}}(\boldsymbol{x_v}) p(\boldsymbol{x_u}, \boldsymbol{x_v}) d\boldsymbol{x_u} d\boldsymbol{x_v} \\
\leq&\int f_{\boldsymbol{u}}^2(\boldsymbol{x_u}) p(\boldsymbol{x_u}, \boldsymbol{x_v}) d\boldsymbol{x_u} d\boldsymbol{x_v} + \int f_{\boldsymbol{v}}^2(\boldsymbol{x_v}) p(\boldsymbol{x_u}, \boldsymbol{x_v}) d\boldsymbol{x_u} d\boldsymbol{x_v} \\
&\qquad + 2\left(\int f_{\boldsymbol{u}}^2(\boldsymbol{x_u}) p(\boldsymbol{x_u}, \boldsymbol{x_v}) d\boldsymbol{x_u} d\boldsymbol{x_v}\right)^{1/2} \cdot \left(\int f_{\boldsymbol{v}}^2(\boldsymbol{x_v}) p(\boldsymbol{x_u}, \boldsymbol{x_v}) d\boldsymbol{x_u} d\boldsymbol{x_v}\right)^{1/2} \\
<&\infty
\end{aligned}$$

The second to the last inequality is due to Holder's inequality with $p = q = 2$. ■

Theorem 3.1.2 *Let $\{X_1, \ldots, X_p\}$ be a set of independent covariates and G_1 and G_2 are two group structures. If $G_1 \leq G_2$ and $G_1 \neq G_2$, then $L^2_{G_1} \subset L^2_{G_2}$.*

Proof The subsetting part is already shown in Theorem 3.1.1, we further need to show the proper part (i.e. strict subset).

For $\boldsymbol{u}, \boldsymbol{v} \in G_1$, $\boldsymbol{u}, \boldsymbol{v} \notin G_2$, $\boldsymbol{u} \cup \boldsymbol{v} \in G_2$, we need to show that there is a function $h(\boldsymbol{x_u}, \boldsymbol{x_v}) \in L^2([0,1]^{|\boldsymbol{u}\cup\boldsymbol{v}|})$ which does not belong to $L^2_{\boldsymbol{u}} \oplus L^2_{\boldsymbol{v}}$. That is

$$\inf_{\substack{f \in L^2_{\boldsymbol{u}} \\ g \in L^2_{\boldsymbol{v}}}} \int \left(h(\boldsymbol{x_u}, \boldsymbol{x_v}) - f(\boldsymbol{x_u}) - g(\boldsymbol{x_v})\right)^2 p(\boldsymbol{x_u}, \boldsymbol{x_v}) d\boldsymbol{x_u} d\boldsymbol{x_v} > 0 \tag{3.2}$$

Define the following functional of f and g as

$$F(f, g) := \int \left(h(\boldsymbol{x_u}, \boldsymbol{x_v}) - f(\boldsymbol{x_u}) - g(\boldsymbol{x_v})\right)^2 p(\boldsymbol{x_u}, \boldsymbol{x_v}) d\boldsymbol{x_u} d\boldsymbol{x_v} \tag{3.3}$$

Let $\delta(\boldsymbol{x_u})$ be the Gâteaux's derivative at $\boldsymbol{x_u}$, then

$$F(f_{\boldsymbol{u}} + t\delta_{\boldsymbol{u}}, g_{\boldsymbol{v}}) - F(f_{\boldsymbol{u}}, g_{\boldsymbol{v}}) = \int \left(2tf\delta + t^2\delta^2 - 2th\delta + 2tg\delta\right) p_{\boldsymbol{uv}} d\boldsymbol{x_u} d\boldsymbol{x_v}$$

At minimum, we have

$$\begin{aligned}
&\lim_{t\to 0}\frac{F(f_{\boldsymbol{u}}+t\delta_{\boldsymbol{u}},g_{\boldsymbol{v}})-F(f_{\boldsymbol{u}},g_{\boldsymbol{v}})}{t}\\
&=\lim_{t\to 0}\int\left(2f\delta+t\delta^2-2h\delta+2g\delta\right)p_{\boldsymbol{uv}}d\boldsymbol{x}_{\boldsymbol{u}}d\boldsymbol{x}_{\boldsymbol{v}}\\
&=\int\left(2f\delta-2h\delta+2g\delta\right)p_{\boldsymbol{uv}}d\boldsymbol{x}_{\boldsymbol{u}}d\boldsymbol{x}_{\boldsymbol{v}}\\
&=0
\end{aligned}\tag{3.4}$$

Since (3.4) holds for all $\delta\in L^2_{\boldsymbol{u}}$, then we have

$$\int\left(f+g-h\right)p_{\boldsymbol{uv}}d\boldsymbol{x}_{\boldsymbol{v}}=0\tag{3.5}$$

By symmetry, we also have the following identity.

$$\int\left(f+g-h\right)p_{\boldsymbol{uv}}d\boldsymbol{x}_{\boldsymbol{u}}=0\tag{3.6}$$

Since h is given, we set $C_1=\int hp_{\boldsymbol{uv}}dv$ and $C_2=\int hp_{\boldsymbol{uv}}du$.

Solving (3.5) for f we have,

$$f=\frac{\int hp_{\boldsymbol{uv}}d\boldsymbol{x}_{\boldsymbol{v}}-\int gp_{\boldsymbol{uv}}d\boldsymbol{x}v}{\int p_{\boldsymbol{uv}}d\boldsymbol{x}_{\boldsymbol{v}}}\tag{3.7}$$

Plug (3.7) into (3.6), we have

$$\begin{aligned}
&\int\frac{\int hp_{\boldsymbol{uv}}d\boldsymbol{x}_{\boldsymbol{v}}-\int gp_{\boldsymbol{uv}}d\boldsymbol{x}_{\boldsymbol{v}}}{\int p_{\boldsymbol{uv}}d\boldsymbol{x}_{\boldsymbol{v}}}p_{\boldsymbol{uv}}d\boldsymbol{x}_{\boldsymbol{u}}+g\int p_{\boldsymbol{uv}}d\boldsymbol{x}_{\boldsymbol{u}}-\int hp_{\boldsymbol{uv}}d\boldsymbol{x}_{\boldsymbol{u}}=0\\
\Leftrightarrow\quad & g\int p_{\boldsymbol{uv}}d\boldsymbol{x}_{\boldsymbol{u}}-\int\frac{\int gp_{\boldsymbol{uv}}d\boldsymbol{x}_{\boldsymbol{v}}}{\int p_{\boldsymbol{uv}}d\boldsymbol{x}_{\boldsymbol{v}}}p_{\boldsymbol{uv}}d\boldsymbol{x}_{\boldsymbol{u}}=\int hp_{\boldsymbol{uv}}d\boldsymbol{x}_{\boldsymbol{u}}-\int\frac{\int hp_{\boldsymbol{uv}}d\boldsymbol{x}_{\boldsymbol{v}}}{\int p_{\boldsymbol{uv}}d\boldsymbol{x}_{\boldsymbol{v}}}p_{\boldsymbol{uv}}d\boldsymbol{x}_{\boldsymbol{u}}
\end{aligned}\tag{3.8}$$

Since $\boldsymbol{X}_{\boldsymbol{u}}\perp\boldsymbol{X}_{\boldsymbol{v}}$, we have $p_{\boldsymbol{uv}}=p_{\boldsymbol{u}}p_{\boldsymbol{v}}$. Then, identity (3.8) is equivalent to

$$g-\int gp_{\boldsymbol{v}}d\boldsymbol{x}_{\boldsymbol{v}}=\int hp_{\boldsymbol{u}}d\boldsymbol{x}_{\boldsymbol{u}}-\int\int hp_{\boldsymbol{v}}d\boldsymbol{x}_{\boldsymbol{v}}p_{\boldsymbol{u}}d\boldsymbol{x}_{\boldsymbol{u}}.$$

This is a Fredholm integral equation, with solution

$$\begin{cases}f=\int hp_{\boldsymbol{v}}d\boldsymbol{x}_{\boldsymbol{v}}-C\\ g=\int hp_{\boldsymbol{u}}d\boldsymbol{x}_{\boldsymbol{u}}+C\end{cases}\tag{3.9}$$

where C is any constant.

To this end, the minimum approximation error in (3.2) achieves 0 when the following identity is true almost surely.

$$h = \int hp_{\boldsymbol{v}} d\boldsymbol{x}_{\boldsymbol{v}} + \int hp_{\boldsymbol{u}} d\boldsymbol{x}_{\boldsymbol{u}}$$

A counter example is given by $h(\boldsymbol{x}_{\boldsymbol{u}}, \boldsymbol{x}_{\boldsymbol{v}}) = \sin(\boldsymbol{x}_{\boldsymbol{u}} + \boldsymbol{x}_{\boldsymbol{v}})$ which does not assume the above decomposition. So $L^2_{\boldsymbol{u}} \oplus L^2_{\boldsymbol{v}}$ is a proper subspace of $h(\boldsymbol{x}_{\boldsymbol{u}}, \boldsymbol{x}_{\boldsymbol{v}}) \in L^2([0,1]^{|\boldsymbol{u}\cup\boldsymbol{v}|})$.

Thus the theorem is proved. ■

From Theorem 3.1.1 and Theorem 3.1.2, we know that different group structures could induce function spaces of different sizes. For non-parametric regression, the size of the function spaces in which the estimated regression function is searched is critical. Usually, an unbiased estimate is desired which requires the function space to be large enough so that the true model is included. Thus, the group structure, from which the function space is induced, plays an important role for additive non-parametric regression. To emphasize the importance, we formally define a specific type of group structures which is desirable for non-parametric estimation.

Definition 3.1.4 *Let $f(\boldsymbol{x}) = \mathbb{E}[Y|\boldsymbol{X} = \boldsymbol{x}]$ be a non-parametric regression model. Given a group structure G for the predictor variables in $\boldsymbol{X}$, if there is a function $f_G \in L^2_G$ such that $f_G = f$ almost surely, then G is called an **amiable group structure** for the model f.*

In the example we discussed previously, G_0, G_1 and G_2 are all amiable group structures. So we know that an amiable group structure for a given model may not be unique. This non-uniqueness property is formalized in the following theorem.

Theorem 3.1.3 *Suppose G is an amiable group structure for $f(\boldsymbol{x}) = \mathbb{E}[Y|\boldsymbol{X} = \boldsymbol{x}]$. If there is a second group structure G' such that $G \leq G'$, then G' is also amiable for f.*

Proof Combine Definition 3.1.3 and Theorem 3.1.1. ■

The set of all amiable group structure for a given model is denoted as $\mathcal{G}^a$. As a consequence of Theorem 3.1.3 and the partial order of group structures defined in Definition 3.1.2 and Definition 3.1.3, $\mathcal{G}^a$ is a fully ordered subset of $\mathcal{G}$. This leads to the following theorem which shows that there is the best group structure for a given model.

Theorem 3.1.4 *For any two group structures in $\mathcal{G}^a$ with the order defined according to Definition 3.1.2 and Definition 3.1.3, there is a unique minimal group structure $G^* \in \mathcal{G}^a$ such that $G^* \leq G$ for all $G \in \mathcal{G}^a$.*

Proof Since the partial order is defined any subset of group structures in $\mathcal{G}^a$, the existence of G^* is the result of Zorn's Lemma. The uniqueness is due to the fact that $\mathcal{G}^a$ is a finite set. ■

Definition 3.1.5 *Let $\boldsymbol{X} = \{X_1, \ldots, X_p\}$ and the model $f(\boldsymbol{x}) = \mathbb{E}[Y|\boldsymbol{X} = \boldsymbol{x}] \in L^2_{\mathcal{X}}$. Then the minimal group structure G^* of $\boldsymbol{X}$ is called the **true group structure** for f.*

From the perspective of statistical modelling, the true group structure G^* represents an achievement in the greatest dimension reduction for the relationship between Y and $\boldsymbol{X}$. In the previous example, we have G_0 being the true group structure. If $G^* = G_0$ is known, one only needs to estimate one univariate and two bivariate nonparametric regression functions. While G_1 and G_2 are both amiable, they both require fitting a three-dimensional non-parametric regression functions. This is both computationally and statistically inefficient. In general, the true additive group structure can help much mitigate the curse of dimensionality while improving both efficiency and interpretability of high dimensional nonparametric regression analysis.

3.1.2 Kernel Methods for Non-parametric Regression

From the above discussion, we can see that a group structure plays an important role in additive non-parametric regression. Our desire for estimation efficiency and unbiasedness is fulfilled by choosing the true group structure G^*.

When the true group structure is known, another consideration for non-parametric regression is how to choose the class of functions for estimation. There have been many methods proposed for non-parametric regression such as Nadaraya-Watson kernel smoothing estimator named after [40] and [41], regression spline and smoothing spline estimators detailed in [42]. In this chapter, we focus on using kernel methods for solving non-parametric regression.

The general formulation[1] of kernel based methods for non-parametric regression is given by

$$\hat{f}_\lambda = \arg\min_{f\in\mathcal{H}} \left(\widehat{\mathcal{R}}(f) + \lambda \mathcal{P}(f) \right). \tag{3.10}$$

Here, $\mathcal{H}$ is an RKHS, $\mathcal{P}$ is some penalty functional (a popular choice is $\mathcal{P}(f) = \|f\|^2_{\mathcal{H}}$ in SVMs), $\lambda > 0$ is a regularization parameter controlling the trade-off between the model fitting and the RKHS-norm penalty, and $\widehat{\mathcal{R}}(f)$ is the empirical risk of f defined as

$$\widehat{\mathcal{R}}(f) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}\left(y_i, f(\boldsymbol{x}_i)\right),$$

where $\mathcal{L}$ is a loss function measuring the goodness of fit using f as the model on the data. This is the sample version of the following population risk

$$\mathcal{R}(f) = \int_{\mathcal{Y},\boldsymbol{\mathcal{X}}} \mathcal{L}\left(y, f(\boldsymbol{x})\right) dP_{Y\boldsymbol{X}}(y, \boldsymbol{x}). \tag{3.11}$$

One of the most common loss functions in regression setting is the quadratic loss $\mathcal{L}(y,t) = (y-t)^2$. It has been well established (see [7], [11]) that the solution

[1] In fact, there is a more general formulation where the observations of the response variable Y is also transformed by some monotone function as $g(y)$. One example of this more general formulation is given in [43]. However, in this chapter it is assumed that the response is not transformed or the response transformation is known.

of Problem (3.10) exists and is unique. More importantly, due to the celebrated representer theorem, the solution assumes the following general form

$$\hat{f}_\lambda = \sum_{i=1}^{n} \alpha_i K(x_i, \cdot),$$

where $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$ are the estimated coefficients.

It is well known that when $\mathcal{L}$ is the quadratic loss the minimizer of Equation (3.11) is given by the conditional expectation

$$f_{Y|X}(\boldsymbol{x}) = \int_{\mathcal{Y}} y dP_{Y|\boldsymbol{X}}(y).$$

which is the target that we want to recover for non-parametric regression.

While using an RKHS, denoted as $\mathcal{H}$, as our model space in a kernel based method, it is possible that our target function $f_{Y|X} \notin \mathcal{H}$. That is, we may have the following population optimal solution in $\mathcal{H}$,

$$f_{\mathcal{H}} = \arg\min_{f \in \mathcal{H}} \int_{\mathcal{Y}\times\boldsymbol{\mathcal{X}}} \mathcal{L}\left(y, f(\boldsymbol{x})\right) dP_{Y\boldsymbol{X}}(y, \boldsymbol{x}),$$

and $f_{\mathcal{H}} \neq f_{Y|X}$.

In this case, the population risk $\mathcal{R}(\hat{f}_\lambda)$ of a empirical solution $\hat{f}_\lambda$ in RKHS can be decomposed as follows when the quadratic loss is used.

$$\begin{aligned}
\mathcal{R}(\hat{f}_\lambda) &= \int_{\mathcal{Y}\times\boldsymbol{\mathcal{X}}} (y - f_{Y|X})^2 dP_{Y\boldsymbol{X}}(y, \boldsymbol{x}) + \int_{\boldsymbol{\mathcal{X}}} (f_{Y|X} - \hat{f}_\lambda)^2 dP_{\boldsymbol{X}}(\boldsymbol{x}) \\
&= \int_{\mathcal{Y}\times\boldsymbol{\mathcal{X}}} (y - f_{Y|X})^2 dP_{Y\boldsymbol{X}}(y, \boldsymbol{x}) + \\
&\qquad \int_{\boldsymbol{\mathcal{X}}} (f_{Y|X} - f_{\mathcal{H}})^2 dP_{\boldsymbol{X}}(\boldsymbol{x}) + \int_{\boldsymbol{\mathcal{X}}} (f_{\mathcal{H}} - \hat{f}_\lambda)^2 dP_{\boldsymbol{X}}(\boldsymbol{x}),
\end{aligned}$$

where the second term in the first identity is the source of error contributed by using the functions in $\mathcal{H}$ to approximate our target $f_{Y|X}$. Whenever $f_{Y|X} \notin \mathcal{H}$, it is strictly positive. In Chapter 2, we have shown that $\int_{\boldsymbol{\mathcal{X}}} (f_{\mathcal{H}} - \hat{f}_\lambda)^2 dP_{\boldsymbol{X}}(\boldsymbol{x})$ converges to zero for OKGT. This is also true when the response transformation is known. In this chapter, we assume that our target $f_{Y|X} \in \mathcal{H}$. As a consequence, we have $\int_{\boldsymbol{\mathcal{X}}} (f_{Y|X} - f_{\mathcal{H}})^2 dP_{\boldsymbol{X}}(\boldsymbol{x})$ also vanishes.

Assuming $f_{Y|X} \in \mathcal{H}$ may seem to over-simplify our problem. However, we can choose a kernel K which is universal so that the corresponding RKHS is dense in L^2 space. The following definition of a *universal kernel* is the same as that in [44].

Definition 3.1.6 *A kernel K is universal for $\mathcal{C}([0,1])$ if its induced RKHS $\mathcal{H}_K$ is dense in $\mathcal{C}([0,1])$ in uniform norm, that is, for all $f \in \mathcal{C}([0,1])$ and $\epsilon > 0$ there is $g \in \mathcal{H}_K$ such that $\|f - g\|_\infty < \epsilon$.*

Then, the following Lemma shows that a universal kernel for the space of continuous functions is also universal for L^2 space (see Theorem 3 in [45]).

Lemma 3.1.1 *Let K be a kernel. If K is universal for $\mathcal{C}([0,1])$ in uniform norm, then K is also universal for $L^2([0,1])$ in L_2 norm.*

If a kernel K is chosen to be universal, the induced RKHS $\mathcal{H}_K$ is rich enough such that using $\mathcal{H}_K$ as an approximation of L^2 space incurs no loss. Two examples of universal kernel are Gaussian and Laplace.

3.1.3 RKHS for Additive Non-parametric Regression

Given an additive group structure G, the non-parametric regression tries to find a regression function for each group of predictor variables. By using kernel methods, each regression function is estimated from a RKHS. Let $(K_j, \mathcal{H}_j)$ be the kernel and the corresponding RKHS for the j-th group $\boldsymbol{u}_j$, then we are essentially finding the solution for the additive non-parametric regression problem from the following direct sum RKHS

$$\mathcal{H}_G := \left\{ f = \sum_{\boldsymbol{u} \in G} f_{\boldsymbol{u}} \middle| f_{\boldsymbol{u}} \in \mathcal{H}_{\boldsymbol{u}} \text{ induced by some kernel } K_{\boldsymbol{u}} \right\}. \tag{3.12}$$

By considering $\mathcal{H}_G$ as the model space for our non-parametric regression problem, Problem (3.10) can be re-written as

$$\hat{f}_{\lambda,G} = \arg\min_{f \in \mathcal{H}_G} \left(\widehat{\mathcal{R}}(f) + \lambda \mathcal{P}(f) \right), \tag{3.13}$$

where we explicitly add the group structure G as an subscript of the minimizer $\hat{f}$.

If the direct sum RKHS $\mathcal{H}_G$ is dense in L^2_G, then it is guaranteed that there is no loss incurred by using a $f \in \mathcal{H}_G$ to approximate our target function. The following two Lemmas show that universality is preserved under direct sum and direct product operations between two kernels.

Lemma 3.1.2 *If K_1 and K_2 are two universal kernels for $L^2([0,1]^{p_1})$ and $L^2([0,1]^{p_2})$ respectively, then $K_1 \oplus K_2$ is universal for $L^2([0,1]^{p_1}) \oplus L^2([0,1]^{p_2})$.*

This leads to the following Lemma which shows that $\mathcal{H}_G$ for a given group structure G is dense in L^2_G with the individual kernels are universal.

Lemma 3.1.3 *Let G be a group structure where each group $\boldsymbol{u} \in G$ is equipped with a kernel $K_{\boldsymbol{u}}$. If $K_{\boldsymbol{u}}$ is universal for all $\boldsymbol{u} \in G$, then $K_G := \sum_{\boldsymbol{u} \in G} K_{\boldsymbol{u}}$ is universal for L^2_G.*

In this chapter, we use Gaussian kernel for all $K_{\boldsymbol{u}}$'s. That is, $K_{\boldsymbol{u}}(\boldsymbol{x}_{\boldsymbol{u}}, \boldsymbol{x}'_{\boldsymbol{u}}) = \exp\{-\sigma \left\|\boldsymbol{x}_{\boldsymbol{u}} - \boldsymbol{x}'_{\boldsymbol{u}}\right\|^2\}$ for all $\boldsymbol{u} \in G$ with common value for the parameter σ.

3.1.4 Complexity of Group Structure

As we discussed in the previous sections, the true additive group structure G^* represents the finest additive structure for the relationship between Y and $\boldsymbol{X}$. This parsimonious representation would lead to the highest possible efficiency in both statistical estimation and numerical computation. However, in practice the true group structure G^* is not known and needs to be learned from data.

The idea of searching for G^* from data is based on the intuition that the hypothesis space $\mathcal{H}_{G^*}$ induced by the true group structure is smaller than any other function space induced by an amiable group structure. Using $\mathcal{H}_{G^*}$ for kernel non-parametric regression achieves the fastest[2] rate of the risk vanishing to zeros. So with large enough sample size, the true group structure G^* as the minimizer of the risk (with

[2]This is in probability.

other things fixed) will stands out. The reason for this intuition to be true is in two-fold. First, when the hypothesis space induced by a non-amiable group structure is used for non-parametric regression, there is always positive approximation error due to the usage of a function space which does not include the true target function. So non-parametric regression with a non-amiable group structure will eventually be out-performed (in terms of total risk) by that with an amiable group structure. Second, though the approximation error disappears when amiable group structures are used, their estimation errors (since estimation is a finite sample problem) have different convergence rate. This is because different amiable group structure induce hypothesis spaces with different complexity. While the most general additive group structure $\{(1, \ldots, p)\}$ induces the largest function space for non-parametric estimation, it is the most complex hence its estimation is the least efficient due to its slow convergence rate. On the other hand, the true group structure G^* induces a function space which is the least complex which enjoys the fastest convergence rate while being used for non-parametric regression. Therefore, when the sample size is large enough, G^* will most likely stand out as a more optimal solution for non-parametric regression if we compare its estimation risk with that of the other amiable group structure. Actually, this is the idea we follow in proving the model selection consistency for our method later in this chapter.

Based on the above reasoning, the concept of complexity of an additive group structure plays a critical role in learning the true group structure G^* for additive non-parametric regression. Since a group structure G affects the performance of non-parametric through the hypothesis space $\mathcal{H}_G$ it induces, we can use the complexity of the function space as a proxy to measure the complexity of a group structure. By considering this general notation of complexity of additive group structures, our kernel non-parametric regression problem is represented as

$$\left(\hat{f}_{\lambda,\mu}, \widehat{G}\right) = \arg \min_{\substack{G \in \mathcal{G} \\ f \in \mathcal{H}_G}} \left(\widehat{\mathcal{R}}(f) + \lambda \mathcal{P}(f) + \mu \boldsymbol{\mathcal{C}}(G)\right). \tag{3.14}$$

By comparing with Problem (3.13), here we added one more additive term $\mu\boldsymbol{\mathcal{C}}(G)$ in our minimization target. This is our main contribution to non-parametric regression. There are two components in this newly added term. The first component is the complexity measure $\boldsymbol{\mathcal{C}}(G)$ which quantifies the complexity of the additive group structure G. We require $\mathcal{P}$ to be defined for all group structures $G \in \mathcal{G}$ but monotonically increasing in the domain of all amiable group structure $G \in \mathcal{G}^a$ with respect to the partial order defined in Definition 3.1.2 and 3.1.3. That is, if we only consider the group structures in $\mathcal{G}^a$, the complexity measure of the true group structure $\boldsymbol{\mathcal{C}}(G^*)$ is the smallest, and the most general group structure $\{(1, \ldots, p)\}$ has the largest complexity measure. The second component of our novel term is the tuning parameter μ, which controls the trade-off between the usual regularized loss and the complexity measure of the group structure. By introducing this third term, the regularized risk of non-parametric regression depends on the underlying group structure.

While we have considered the impact of group structure on non-parametric regression by using a general complexity regularization term, it is still not clear how the complexity looks like. We can neither perform analysis nor solve a finite sample problem by using the formulation of (3.14).

A number of different types of complexity (or capacity) measures have been proposed and studied for RKHSs. Some examples include entropy (see [46]), VC dimensions (see [47]), Rademacher complexity (see [48]), and covering numbers (see [7], [46]). We will use the results on covering number to design a practically convenient penalty for AGSI and nonparametric regression.

First we define ϵ-cover for a set.

Definition 3.1.7 *An ϵ-cover of a set $\mathcal{S} \subset \mathcal{F}$ is a set of elements in $\mathcal{F}$ such that the union of the ϵ-balls around these elements contains $\mathcal{S}$.*

Then, we formally define the covering number for a general function space as follows, which is similar to the definition in [7].

Definition 3.1.8 *Let $\mathcal{X}$ be a Banach space. For $\epsilon > 0$, the ϵ-covering number of $\mathcal{X}$ with respect to some metric d, denoted as $\mathcal{N}(\epsilon, \mathcal{X}, d)$, is the smallest number of an ϵ-cover of $\mathcal{X}$ using the metric d.*

It is also useful to define the covering number for an operator.

Definition 3.1.9 *Let $\mathcal{X}$ and $\mathcal{Y}$ be two Banach spaces and $\mathcal{B}_{\mathcal{X}}$ be the unit ball in $\mathcal{X}$. For $\epsilon > 0$, the ϵ-covering number of an operator $T : \mathcal{X} \to \mathcal{Y}$ is defined as*

$$\mathcal{N}(\epsilon, T) := \mathcal{N}(\epsilon, T(\mathcal{B}_{\mathcal{X}}), d_{\mathcal{Y}}),$$

where $d_{\mathcal{Y}}$ is the metric of Banach space $\mathcal{Y}$.

It is well known (see [49], [50]) that an RKHS $\mathcal{H}_K$ can be embedded into the space of continuous functions $\mathcal{C}(\mathcal{X})$, and we denoted the inclusion as $I_K : \mathcal{H}_K \to \mathcal{C}(\mathcal{X})$. So for an r-ball in $\mathcal{H}_K$ defined as $\mathcal{H}_{K,r} := \left\{ f \in \mathcal{H}_K \mid \|f\|_{\mathcal{H}_K} \leq r \right\}$, its inclusion $I(\mathcal{H}_{K,r})$ is a subset of $\mathcal{C}(\mathcal{X})$. One way to describe the complexity of an RKHS $\mathcal{H}_K$ is through the complexity of $\overline{I(\mathcal{H}_{K,r})}$, the closure of $I(\mathcal{H}_{K,r})$, for a given value of r.

In [50], an upper bound for $\mathcal{N}\left(\epsilon, \overline{I(\mathcal{H}_{K,r})}, d_\infty\right)$ is given which depends on the regularity of the kernel function K. Here the metric d_∞ denotes the usual sup-norm. This result is stated in the following theorem.

Theorem 3.1.5 *Let K be be a convolution kernel, i.e. $K(x,t) = k(x-t)$, on $[0,1]^p$ and $\mathcal{H}_K$ be the associated RKHS. If the Fourier transform of k decays exponentially, then the covering number of the r-ball, $\mathcal{N}\left(\epsilon, \overline{I(\mathcal{H}_{K,r})}, d_\infty\right)$, satisfies*

$$\ln \mathcal{N}\left(\epsilon, \overline{I(\mathcal{H}_{K,r})}, d_\infty\right) \leq C_{k,p} \left(\ln \frac{r}{\epsilon}\right)^{p+1} \tag{3.15}$$

where $C_{k,p}$ is a constant depending on the kernel function k and dimension p.

In particular, when K is a Gaussian kernel, it is shown in [46] that the following upper bound holds for the covering number.

Proposition 3.1.1 *Let* $K(x,y) = \exp\left\{-\frac{\|x-y\|^2}{\sigma^2}\right\}$ *be a Gaussian kernel with* $\sigma > 0$ *and* $x, y \in \mathbb{R}^p$. *Then for* $0 < \epsilon < r/2$, *there holds*

$$\ln \mathcal{N}\left(\epsilon, \overline{I(\mathcal{H}_{K,r})}, d_\infty\right) \leq \left(3\ln\frac{r}{\epsilon} + \frac{54p}{\sigma^2} + 6\right)^p \times \left((6p+1)\ln\frac{r}{\epsilon} + \frac{90p^2}{\sigma^2} + 11p + 3\right). \tag{3.16}$$

In particular, when $0 < \epsilon < r\exp\{-\frac{90p^2}{\sigma^2} - 11p - 3\}$, *we have*

$$\ln \mathcal{N}\left(\epsilon, \overline{I(\mathcal{H}_{K,r})}, d_\infty\right) \leq 4^p(6p+2)\left(\ln\frac{r}{\epsilon}\right)^{p+1}.$$

Both upper bounds given by (3.15) and (3.16) have the dimension p of the input space as a power term. The upper bounds also depends on the radius r of the RKHS ball and the radius ϵ of the covering balls. The value of r restrict the size of RKHS under consideration. When we choose the popular penalty $\mathcal{P}(f) = \|f\|_{\mathcal{H}}^2$ in Problem (3.14), the parameter λ also controls (indirectly) the size of RKHS for model fitting. In this case, r and λ could be related. The value of ϵ represents the measuring granularity for characterizing the complexity for RKHSs. According to [51], the growth rate of $\mathcal{N}(\epsilon, I_K)$ or its logarithmic version can be viewed as a measure of the complexity of RKHS. Note that r is the length determined by the RKHS norm while ϵ is the length determined by the norm of the embedding space (in our case, it is the space of continuous function $\mathcal{C}(\mathcal{X})$).

The upper bounds depend on r and ϵ through $\ln(r/\epsilon)$. When $\epsilon \to 0$ with r either fixed or determined by the value of the other parameter λ, $(\ln(r/\epsilon))^{p+1}$ becomes the dominating factor in the upper bounds and the other terms are negligible. So we parameterized the upper bound by α^{p+1} where α is a tuning parameter taking the place of $\ln(r/\epsilon)$. In theory, the choice of α depends on the usual tuning parameter λ and the embedding space. In practice, we choose its value via cross-validation.

In our additive kernel non-parametric regression problem (3.14), the hypothesis space $\mathcal{H}_G$ is an direct sum RKHS given in (3.12). In order to take the advantage of the additive structure in constructing the complexity measure for RKHSs, we rely on the following result.

Lemma 3.1.4 *Let $S, T : \mathcal{F}_1 \to \mathcal{F}_2$ be operators in Banach spaces and $\epsilon_1, \epsilon_2 > 0$. Then we have*

$$\mathcal{N}(\epsilon_1 + \epsilon_2, T + S) \leq \mathcal{N}(\epsilon_1, S) \cdot \mathcal{N}(\epsilon_2, T).$$

In order to apply Lemma 3.1.4, we need some technical preparation. For each $\mathcal{H}_{\boldsymbol{u}}$ used in the construction of the direct sum RKHS (3.12), we define its extension in $\mathcal{H}_G$ as

$$\widetilde{\mathcal{H}}_{\boldsymbol{u}} := \{f : [0,1]^p \to \mathbb{R}, \boldsymbol{x} \to f_{\boldsymbol{u}}(\boldsymbol{x}_{\boldsymbol{u}})\}, \tag{3.17}$$

so we have $\widetilde{\mathcal{H}}_{\boldsymbol{u}} \subset \mathcal{H}_G$. In other words, each function in $\widetilde{\mathcal{H}}_{\boldsymbol{u}}$ is the sum of a function in $\mathcal{H}_{\boldsymbol{u}}$ and the zero function. We denote the extension operator as $J_{\boldsymbol{u}} : \mathcal{H}_{\boldsymbol{u}} \to \widetilde{\mathcal{H}}_{\boldsymbol{u}}$. Then, the inclusion operator I can be naturally applied on $\widetilde{\mathcal{H}}_{\boldsymbol{u}}$. Since the extension $J_{\boldsymbol{u}}$ is a bijection, they have the same complexity in terms of covering numbers. That is

$$\mathcal{N}\left(\epsilon, \overline{I_K(\mathcal{H}_{\boldsymbol{u}})}, d_\infty\right) = \mathcal{N}\left(\epsilon, \overline{I_K(\widetilde{\mathcal{H}}_{\boldsymbol{u}})}, d_\infty\right). \tag{3.18}$$

Proof Since $\tilde{f}$ is equivalent to $(f_{\boldsymbol{u}}, \mathbf{0})$ for each $\tilde{f} \in \tilde{\mathcal{H}}_{\boldsymbol{u}}$, the $\mathbf{0}$ component does not contribute additional complexity. Besides, d_∞ is the sup-norm, then the metric does not change. So the covering numbers are the same. ■

Then, we have the following upper bound for the covering number of direct sum RKHS, which is the result of the application of Lemma 3.1.4.

Lemma 3.1.5 *Let G be an additive group structure and $\mathcal{H}_G$ be the induced direct sum RKHS defined in (3.12). Then, we have the following inequality relating the covering number of $\mathcal{H}_G$ and the covering numbers of $\mathcal{H}_{\boldsymbol{u}}$*

$$\ln \mathcal{N}(\epsilon, I_G, d_\infty) \leq \sum_{\boldsymbol{u} \in G} \ln \mathcal{N}\left(\frac{\epsilon}{|G|}, I_{\boldsymbol{u}}, d_\infty\right), \tag{3.19}$$

where $|G|$ denotes the number of groups in G.

Proof Due to Lemma 3.1.4, we have $\mathcal{N}(\epsilon, I_G, d_\infty) \leq \Pi_{\boldsymbol{u}\in G}\mathcal{N}\left(\frac{\epsilon}{|G|}, \overline{I(\widetilde{\mathcal{H}}_{\boldsymbol{u}})}, d_\infty\right) = \Pi_{\boldsymbol{u}\in G}\mathcal{N}\left(\frac{\epsilon}{|G|}, I_{\boldsymbol{u}}, d_\infty\right)$. Then, taking log on both sides gives the desired result. ■

By applying Theorem 3.1.5, Lemma 3.1.5 and the argument on using α as a tuning parameter, we have

$$\ln \mathcal{N}(\epsilon, I_G, d_\infty) = \mathcal{O}\left(\sum_{\boldsymbol{u}\in G} \alpha(\epsilon)^{|\boldsymbol{u}|+1}\right), \tag{3.20}$$

where we explicitly indicate the dependency of α on ϵ and use the same α for all groups. Now we could use the rate in (3.20) as the explicit expression of the complexity measure $\boldsymbol{\mathcal{C}}(G)$ in Problem (3.14). Recall that there is another tuning parameter μ in Problem (3.14) which controls the effect of the complexity of group structure has on the penalized risk. By factoring out the common 1 in the exponent for all groups and combining it with μ, we could further simplify the penalty's expression. Thus, we have the following explicit formulation for Additive Group Structure Identification which simultaneously solves the non-parametric regression problem.

$$\left(\hat{f}_{\lambda,\mu}, \widehat{G}\right) = \arg\min_{\substack{G\in\mathcal{G}\\ f\in\mathcal{H}_G}} \left(\sum_{i=1}^{n} (y_i - f(\boldsymbol{x}_i))^2 + \lambda \|f\|_{\mathcal{H}_G}^2 + \mu \sum_{\boldsymbol{u}\in G} \alpha(\epsilon)^{|\boldsymbol{u}|}\right). \tag{3.21}$$

3.1.5 Estimation

In this chapter, we assume that the value of λ is pre-specified. In practice, this parameter can be tuned separately. If the values of μ and α are also given, Problem (3.21) can be solved by following a two-step procedure.

First, for a given group structure G, the functions $f_{\boldsymbol{u}}$ in the problem

$$\min_{\substack{f_{\boldsymbol{u}}\in\mathcal{H}_{\boldsymbol{u}}\\ \boldsymbol{u}\in G}} \widehat{\mathcal{R}}_G^{\lambda} = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \sum_{\boldsymbol{u}\in G} f_{\boldsymbol{u}}(\boldsymbol{x}_{\boldsymbol{u},i})\right)^2 + \lambda \left\|\sum_{\boldsymbol{u}\in G} f_{\boldsymbol{u}}\right\|_{\mathcal{H}_G}^2 \tag{3.22}$$

can be estimated by applying OKGT with g being identity transformation or kernel ridge regression (KRR) [52] where a Gaussian kernel is used for each group $\boldsymbol{u} \in G$. We use $\widetilde{\mathcal{R}}_G^{\lambda}$ to denote the minimum value of the target function in (3.22).

Second, together with the new penalty, $\widetilde{\mathcal{R}}_G^{\lambda}$ is minimized over the space of additive group structures $\mathcal{G}$ to find the optimal group structure $\widehat{G}$. That is,

$$\widehat{G} = \arg\min_{G \in \mathcal{G}} \left\{ \widetilde{\mathcal{R}}_G^{\lambda} + \mu \sum_{\boldsymbol{u} \in G} \alpha^{|\boldsymbol{u}|} \right\}. \tag{3.23}$$

The two-step procedure above is expected to identify the true structure, that is, $\widehat{G} = G_0$. Recall that all possible group structures are classified into three categories, the true structure (G_0), amiable structures, and non-amiable structures. If G is non-amiable structure, then $\widetilde{\mathcal{R}}_G^{\lambda}$ is expected to be large, because G is a wrong structure and will result in bias in model fitting. If G is amiable, though $\widetilde{\mathcal{R}}_G^{\lambda}$ is expected to be small, the complexity penalty of G is larger than that for G_0. As a consequence, only G_0 can simultaneously achieve a small $\widetilde{\mathcal{R}}_{G_0}^{\lambda}$ and a relatively small complexity penalty. Therefore, when the sample size is large enough, we expect $\widehat{G} = G_0$ with high probability.

If the values of the turning parameters μ and α are not given, cross-validation can be used to select proper values for them. The discussion on this account will be deferred to Section 3.2.

3.2 Algorithm

In this section, an exhaustive search algorithm is introduced for group structure selection. This algorithm works when the number of predictor variables is relatively small or the set of candidate group structures is restricted. In addition, we will also propose a stepwise algorithm for the case when the number of predictor variables is relatively large.

3.2.1 Exhaustive Search

When the number of predictor variables in a nonparametric model is small and the values of the tuning parameters are given, we can afford to select the optimal group

structure by solving Problem (3.21) for each possible group structure. For example, for six predictor variables there are 203 possible group structures[3].

Though the brute force employed by the exhaustive algorithm looks intimidating, it will be used to effectively show that adding the complexity penalty is necessary to consistently select the true group structure. Besides, the exhaustive search algorithm is an easy and effective approach for small models. The detailed implementation is shown in Algorithm 1.

Algorithm 1 Exhaustive Search Algorithm for AGSI

1: **Input:** $\{y_i, \boldsymbol{x}_i\}_{i=1}^n$, μ, α.
2: fix $K_{\boldsymbol{u}}$ to be Gaussian for for each group;
3: **for** G in $\mathcal{G}$ **do**
4: $\quad \widetilde{\mathcal{R}}_G \leftarrow$ fit OKGT with fixed response or KRR;
5: $\quad \widetilde{\mathcal{R}}_G^{\text{pen}} \leftarrow \widetilde{\mathcal{R}}_G + \mu \sum_{\boldsymbol{u} \in G} \alpha^{|\boldsymbol{u}|}$;
6: **end for**
7: $\widehat{G} \leftarrow \arg\min_{G \in \mathcal{G}} \widetilde{\mathcal{R}}_G^{\text{pen}}$;

In Algorithm 1, we assume that μ and α are given. However in practice, μ and α need to be estimated from data. Algorithm 2 is created for this purpose, where a validation step is added to learn the optimal tuning parameters.

Algorithm 2 assumes that a data set is large enough to be split into training and testing sets. For each (μ, α) value pair, a group structure is selected by running the same procedure as Algorithm 1. The estimated nonparametric functions are also returned from the training phase. Then, the estimated functions are used on the test data to calculate the prediction error. The optimal tuning parameter values are chosen to be the ones with the lowest prediction error. In general, cross-validation (e.g. 10-fold) can be used to select the tuning parameters.

[3]The number of group structures for a fixed number of predictor variables is given by the Bell number.

Algorithm 2 Exhaustive Search with Validation Algorithm for AGSI

Input: $\{y_i, \boldsymbol{x}_i\}_{i=1}^n$, (μ, α) value grid.
fix $K_{\boldsymbol{u}}$ to be Gaussian for each group;
split data into train set $\mathcal{T}$ and validation set $\mathcal{V}$;
for (μ, α) in the value grid **do**
    for $G \in \mathcal{G}$ **do**
        $\widetilde{\mathcal{R}}_G, \hat{f}_G \leftarrow$ fit KRR on $\mathcal{T}$;
        $\widetilde{\mathcal{R}}_G^{\text{pen},\mu,\alpha} \leftarrow \widetilde{\mathcal{R}}_G + \mu \sum_{\boldsymbol{u} \in G} \alpha^{|\boldsymbol{u}|}$;
    end for
    $\widehat{G}^{\mu,\alpha} \leftarrow \arg\min_{G \in \mathcal{G}} \widetilde{\mathcal{R}}_G^{\text{pen},\mu,\alpha}$;
    $\hat{y}^{\mathcal{V}} \leftarrow \hat{f}_{\widehat{G}^{\mu,\alpha}}(\boldsymbol{x}^{\mathcal{V}})$;
    $e^2_{\widehat{G}^{\mu,\alpha}} \leftarrow \left\| y^{\mathcal{V}} - \hat{y}^{\mathcal{V}} \right\|^2$;
end for
$\mu^*, \alpha^* \leftarrow \arg\min_{\mu,\alpha} e^2_{\widehat{G}^{\mu,\alpha}}$;
$G^* \leftarrow \widehat{G}^{\mu^*,\alpha^*}$;

3.2.2 Stepwise Approach

The exhaustive search algorithms are suitable for small models. When a model contains large number of predictor variables, the computation cost can be prohibitively high. In order to address the question of estimation and parameter selection for a large model, we propose a backward stepwise algorithm. The base procedure for estimation is illustrated in Algorithm 3.

At the beginning of the algorithm, a fully non-parametric regression with complexity penalty is estimated. During each while loop, one predictor variable is separated, which either forms a new univariate group or joins one of the other groups. Each newly created group structure is used to fit a non-parametric regression. The group structure is updated whenever a better fit is achieved. The iteration continues until all predictor variables are tested. When the tuning parameters are not given, cross validation can be used to learn the optimal values from data.

Algorithm 3 Basic Backward Stepwise Algorithm for AGSI

Input: $\{y_i, \boldsymbol{x}_i\}_{i=1}^n$, μ, α.
fix kernel K to be Gaussian with parameter σ;
$\mathcal{J} \leftarrow \{1, \ldots, p\}$;
$\boldsymbol{u} \leftarrow (1, \ldots, p)$; $G \leftarrow \{\boldsymbol{u}\}$;
$\widetilde{\mathcal{R}}_G^{\text{pen}} \leftarrow$ fit KRR under G with complexity penalty;
while $\mathcal{J} \neq \emptyset$ **do**
    for j in $\mathcal{J}$ **do**
        $\boldsymbol{u}' \leftarrow \boldsymbol{u} \setminus \{j\}$;
        $G' \leftarrow G \setminus \{\boldsymbol{u}\} \cup \{\boldsymbol{u}', (j)\}$;
        $\widetilde{\mathcal{R}}_{G'}^{\text{pen}} \leftarrow$ fit KRR under G' with complexity penalty;
        if $\widetilde{\mathcal{R}}_{G'}^{\text{pen}} < \widetilde{\mathcal{R}}_G^{\text{pen}}$ **then**
            $\widetilde{\mathcal{R}}_G^{\text{pen}} \leftarrow \widetilde{\mathcal{R}}_{G'}^{\text{pen}}$; $G^* \leftarrow G'$;
        end if
        for $\boldsymbol{u}_\ell$ in $G \setminus \{\boldsymbol{u}\}$ **do**
            $\boldsymbol{u}'_\ell \leftarrow \boldsymbol{u}_\ell \cup \{j\}$;
            $G' \leftarrow G \setminus \{\boldsymbol{u}, \boldsymbol{u}_\ell\} \cup \{\boldsymbol{u}', \boldsymbol{u}'_\ell\}$;
            $\widetilde{\mathcal{R}}_{G'}^{\text{pen}} \leftarrow$ fit KRR under G' with complexity penalty;
            if $\widetilde{\mathcal{R}}_{G'}^{\text{pen}} < \widetilde{\mathcal{R}}_G^{\text{pen}}$ **then**
                $\widetilde{\mathcal{R}}_G^{\text{pen}} \leftarrow \widetilde{\mathcal{R}}_{G'}^{\text{pen}}$; $G^* \leftarrow G'$;
            end if
        end for
        $\mathcal{J} \leftarrow \mathcal{J} \setminus \{j\}$; $\boldsymbol{u} \leftarrow \boldsymbol{u} \setminus \{j\}$;
    end for
end while
return G^*;

3.3 Theoretical Properties of AGSI

In this section, we will show that by adding the penalty $\mu \sum_{\boldsymbol{u} \in G} \alpha^{|\boldsymbol{u}|}$ in (3.21) ensures group structure selection consistency for a group additive model. The mean-

ing of the selection consistency is that the probability that solving Problem (3.21) does not results in the true group structure approaches zero when the sample size increases.

3.3.1 Main Results

As we discussed before, when a non-amiable group structure is used, the solution of a usual kernel non-parametric regression problem has a non-zero bias. While all amiable group structures give unbiased estimates, using the true group structure will enjoy the fastest rate of convergence. Thus, the new complexity penalty is used to filter out all amiable group structures with slow convergence rate. We provide the selection consistency theory in this section and the proof follows this idea.

In order to facilitate the proof, we adopt the following notations:

- $\mathcal{R}_g(f_G) := \mathbb{E}\left[\left(g(Y) - \sum_{\boldsymbol{u}\in G} f_{\boldsymbol{u}}(\boldsymbol{X}_{\boldsymbol{u}})\right)^2\right]$ denotes the population risk for some function $f_G \in \mathcal{H}_G$. The subscript G in f_G indicates the associated group structure.
- $\widehat{\mathcal{R}}_g(f_G) := \frac{1}{n}\sum_{i=1}^{n}\left(g(y_i) - \sum_{\boldsymbol{u}\in G} f_{\boldsymbol{u}}(\boldsymbol{x}_{\boldsymbol{u},i})\right)^2$ denotes the empirical risk for some function $f_G \in \mathcal{H}_G$.

First, we need to show that for all amiable group structures $G \in \mathcal{G}_A$, the optimized empirical risk $\widehat{\mathcal{R}}_g(\hat{f}_G)$ converges in probability to the optimal population risk $\mathcal{R}_g(f^*_{G^*})$ which is achieved by the true group structure. Here $\hat{f}_G$ denotes the minimizer of the empirical risk when group structure G is used and $f^*_{G^*}$ denotes the minimizer of the population risk when the true group structure is used. The result is given by Theorem 3.3.1 with an upper bound for the convergence rate.

Theorem 3.3.1 *Let G^* be the true group structure and $G \in \mathcal{G}^a$ be an amiable group structure. The associated direct sum RKHS are denoted as $\mathcal{H}_{G^*}$ and $\mathcal{H}_G$, respectively.*

Assume the optimal transformation for the response is known and given by g. If $\hat{f}_G^{\lambda} \in \mathcal{H}_G$ is the optimal solution of Problem (3.22), then for any $\epsilon > 0$, we have

$$\begin{aligned}
&P\left(\left|\widehat{\mathcal{R}}_g(\hat{f}_G) - \mathcal{R}_g(f_{G^*}^*)\right| > \epsilon\right) \\
\leq & 12n \cdot \exp\left\{\sum_{\boldsymbol{u}\in G} \ln \mathcal{N}\left(\frac{\epsilon}{12|G|}, \mathcal{H}_{\boldsymbol{u}}, \ell_\infty\right) - \frac{\epsilon^2 n}{144}\right\} + \\
& 12n \cdot \exp\left\{\sum_{\boldsymbol{u}\in G} \ln \mathcal{N}\left(\frac{\epsilon}{12|G|}, \mathcal{H}_{\boldsymbol{u}}, \ell_\infty\right) - n\left(\frac{\epsilon}{24} - \frac{\lambda_n \left\|f_{G^*}^*\right\|^2}{12}\right)^2\right\}. \qquad (3.24)
\end{aligned}$$

Proof Since the following inequality holds,

$$\left|\widehat{\mathcal{R}}_g(\hat{f}_G) - \mathcal{R}_g(f_{G^*}^*)\right| \leq \left|\widehat{\mathcal{R}}_g(\hat{f}_G) - \mathcal{R}_g(\hat{f}_G)\right| + \left|\mathcal{R}_g(\hat{f}_G) - \mathcal{R}_g(f_{G^*}^*)\right|, \qquad (3.25)$$

the upper bound for the desired deviation can be derived from the upper bounds of the two terms on RHS in the inequality.

The upper bound for the first term can be derived by using the uniform convergence bound in [53] (also see Lemma 12.38 in [12]). So we have the following probabilistic upper bound for the first term. For all $n > \frac{8}{\epsilon^2}$,

$$\begin{aligned}
&P\left(\left|\widehat{\mathcal{R}}_g(\hat{f}_G) - \mathcal{R}_g(\hat{f}_G)\right| > \frac{\epsilon}{2}\right) \\
\leq & 12n \cdot \mathbb{E}\left[\mathcal{N}\left(\frac{\epsilon}{12}, \mathcal{H}_G, \ell_\infty^{X'}\right)\right] \cdot \exp\left\{-\frac{\epsilon^2 n}{144}\right\} \\
\leq & 12n \cdot \exp\left\{\ln \mathcal{N}^{(n)}\left(\frac{\epsilon}{12}, \mathcal{H}_G, \ell_\infty\right) - \frac{\epsilon^2 n}{144}\right\} \\
\leq & 12n \cdot \exp\left\{\ln \mathcal{N}\left(\frac{\epsilon}{12}, \mathcal{H}_G, \ell_\infty\right) - \frac{\epsilon^2 n}{144}\right\}, \qquad (3.26)
\end{aligned}$$

where $\ell_\infty^{X'}$ denotes the sup-norm of function $f \in \mathcal{F}$ restricted to the sample $X' = \{x_1', \ldots, x_n'\}$ which is independent of the sample $X = \{x_1, \ldots, x_n\}$ used for estimation and $\mathcal{N}^{(n)}(\epsilon, \mathcal{H}, \ell_\infty)$ is called the *ϵ-growth function* of the space $\mathcal{H}$ which is defined as

$$\mathcal{N}^{(n)}(\epsilon, \mathcal{H}, \ell_\infty) := \sup_{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathcal{X}} \mathcal{N}\left(\epsilon, \mathcal{H}, \ell_\infty^X\right).$$

The second inequality is due to the fact that $\mathbb{E}\left[\mathcal{N}\left(\epsilon, \mathcal{H}, \ell_\infty^{X'}\right)\right] \leq \mathcal{N}^{(n)}(\epsilon, \mathcal{H})$.

The upper bound for the second term in 3.25 can be derived by repeatedly applying the same uniform convergence bound. Due to Lemma 3.3.1 in Section 3.3.2, we have for all $\epsilon > 0$ and all $n > 2/\epsilon^2$,

$$\begin{aligned}
&P\left(\left|\mathcal{R}_g(\hat{f}_G) - \mathcal{R}_g(f^*_{G^*})\right| > \frac{\epsilon}{2}\right) \\
\leq& 12n \cdot \ln \mathbb{E}\left[\mathcal{N}\left(\frac{\epsilon^2}{12}, \mathcal{H}_G, \ell_\infty^{X'}\right)\right] \cdot \exp\left\{-n\left(\frac{\epsilon}{24} - \frac{\lambda_n \left\|f^*_{G^*}\right\|^2}{12}\right)^2\right\} \\
\leq& 12n \cdot \exp\left\{\ln \mathcal{N}^{(n)}\left(\frac{\epsilon}{12}, \mathcal{H}_G, \ell_\infty\right) - n\left(\frac{\epsilon}{24} - \frac{\lambda_n \left\|f^*_{G^*}\right\|^2}{12}\right)^2\right\} \\
\leq& 12n \cdot \exp\left\{\ln \mathcal{N}\left(\frac{\epsilon}{12}, \mathcal{H}_G, \ell_\infty\right) - n\left(\frac{\epsilon}{24} - \frac{\lambda_n \left\|f^*_{G^*}\right\|^2}{12}\right)^2\right\}. \qquad (3.27)
\end{aligned}$$

By plugging the upper bounds (3.26) and (3.27) in (3.25), we have

$$\begin{aligned}
&P\left(\left|\widehat{\mathcal{R}}_g(\hat{f}_G) - \mathcal{R}_g(f^*_{G^*})\right| > \epsilon\right) \\
\leq& 12n \cdot \exp\left\{\ln \mathcal{N}\left(\frac{\epsilon}{12}, \mathcal{H}_G, \ell_\infty\right) - \frac{\epsilon^2 n}{144}\right\} + \\
&\qquad 12n \cdot \exp\left\{\ln \mathcal{N}\left(\frac{\epsilon}{12}, \mathcal{H}_G, \ell_\infty\right) - n\left(\frac{\epsilon}{24} - \frac{\lambda_n \left\|f^*_{G^*}\right\|^2}{12}\right)^2\right\} \qquad (3.28)
\end{aligned}$$

By using Lemma 3.3.2, we can bound the covering number for $\mathcal{H}_G$ from above and obtain the following inequality.

$$\begin{aligned}
&P\left(\left|\widehat{\mathcal{R}}_g(\hat{f}_G) - \mathcal{R}_g(f^*_{G^*})\right| > \epsilon\right) \\
\leq& 12n \cdot \exp\left\{\sum_{\boldsymbol{u} \in G} \ln \mathcal{N}\left(\frac{\epsilon}{12|G|}, \mathcal{H}_{\boldsymbol{u}}, \ell_\infty\right) - \frac{\epsilon^2 n}{144}\right\} + \\
&\qquad 12n \cdot \exp\left\{\sum_{\boldsymbol{u} \in G} \ln \mathcal{N}\left(\frac{\epsilon}{12|G|}, \mathcal{H}_{\boldsymbol{u}}, \ell_\infty\right) - n\left(\frac{\epsilon}{24} - \frac{\lambda_n \left\|f^*_{G^*}\right\|^2}{12}\right)^2\right\}.
\end{aligned}$$

■

As a result of Theorem 3.3.1, we can construct a Bonferroni type union upper bound for all amiable group structures in $\mathcal{G}^a$. This is stated in the following proposition.

Proposition 3.3.1 *Let $\mathcal{G}^a$ be the set of all amiable group structures and g is the true transformation of the response. For any $\epsilon > 0$ and $n > 2/\epsilon^2$, we have*

$$P\left(\left|\widehat{\mathcal{R}}_g(\hat{f}_G^{\lambda}) - \mathcal{R}_g(f_{G^*}^*)\right| > \epsilon\right) \leq$$
$$12n\left|\mathcal{G}^a\right| \cdot \left[\exp\left\{\max_{G\in\mathcal{G}^a} \ln \mathcal{N}\left(\frac{\epsilon}{12}, \mathcal{H}_G, \ell_\infty\right) - \frac{\epsilon^2 n}{144}\right\} + \right.$$
$$\left. \exp\left\{\max_{G\in\mathcal{G}^a} \ln \mathcal{N}\left(\frac{\epsilon}{12}, \mathcal{H}_G, \ell_\infty\right) - n\left(\frac{\epsilon}{24} - \frac{\lambda\left\|f_{G^*}^*\right\|^2}{12}\right)^2\right\}\right] \quad (3.29)$$

Proof Denote $\mathcal{D}_{G,\epsilon}^{(n)} = \left\{(\boldsymbol{x}_i, y_i)_{i=1}^n \in \mathcal{X}\times\mathcal{Y} \middle| \left|\widehat{\mathcal{R}}(\hat{f}_G, g) - \mathcal{R}(f_{G^*}^*, g)\right| > \epsilon\right\}$, then we have

$$P\left(\bigcup_{G\in\mathcal{G}^a} \mathcal{D}_{G,\epsilon}\right) \leq \sum_{G\in\mathcal{G}^a} P\left(\mathcal{D}_{G,\epsilon}\right)$$
$$\leq \left|\mathcal{G}^a\right| 12n \exp\left\{\max_{G\in\mathcal{G}^a} \ln \mathcal{N}\left(\frac{\epsilon}{12}, \mathcal{H}_G, \ell_\infty\right) - \frac{\epsilon^2 n}{144}\right\} +$$
$$\left|\mathcal{G}^a\right| 12n \exp\left\{\max_{G\in\mathcal{G}^a} \ln \mathcal{N}\left(\frac{\epsilon}{12}, \mathcal{H}_G, \ell_\infty\right) - n\left(\frac{\epsilon}{24} - \frac{\lambda\left\|f_{G^*}^*\right\|^2}{12}\right)^2\right\}$$

where the second inequality is due to the proof of Theorem 3.3.1. ■

Theorem 3.3.1 and Proposition 3.3.1 show that the solution of Problem (3.22) is asymptotically consistent for an amiable group structure. This is true even without considering the capacity of the function spaces.

The second step towards proving AGSI's selection consistency is to show that for a non-amiable group structure $G' \in \mathcal{G} \setminus \mathcal{G}^a$, its minimum empirical risk (3.22) is not consistent to the true population risk. The idea of the proof is based on the bias-variance decomposition of the population risk.

Theorem 3.3.2 *For an non-amiable group structure $G' \in \mathcal{G} \setminus \mathcal{G}^a$, the empirical risk does not converge in probability to the true optimal risk. In particular, there is a constant $C > 0$ such that*

$$\left|\widehat{\mathcal{R}}_g(\hat{f}_{G'}^{\lambda}) - \mathcal{R}_g(f_{G^*}^*)\right| \to C,$$

with rate $\mathcal{O}_p\left(12n\exp\left\{48|G'|\left(\frac{R|G'|}{\epsilon}\right)^2\ln^2\left(\frac{4|G'|eRn}{\epsilon}\right)\right\}\right)$.

Proof We start with the following triangle inequality

$$\left|\widehat{\mathcal{R}}_g(\hat{f}_{G'})-\mathcal{R}_g(f^*_{G^*})\right|\leq\left|\widehat{\mathcal{R}}_g(\hat{f}_{G'})-\mathcal{R}_g(\hat{f}_{G'})\right|+\left|\mathcal{R}_g(\hat{f}_{G'})-\mathcal{R}_g(f^*_{G^*})\right|. \tag{3.30}$$

The first term on the RHS can be bounded by using the same uniform convergence bound (12.135) in [12]. For any $\epsilon>0$ and all $n>2/\epsilon^2$,

$$\begin{aligned}\mathbb{P}\left(\left|\widehat{\mathcal{R}}_g(\hat{f}_{G'})-\mathcal{R}_g(\hat{f}_{G'})\right|>\epsilon\right)&\leq 12n\cdot\mathbb{E}\left[\mathcal{N}\left(\frac{\epsilon}{6},\mathcal{H}_{G'},\ell_\infty^{X'}\right)\right]\exp\left\{-\frac{\epsilon^2 n}{36}\right\}\\&\leq 12n\cdot\exp\left\{\ln\mathcal{N}\left(\frac{\epsilon}{6},\mathcal{H}_{G'},\ell_\infty\right)-\frac{\epsilon^2 n}{36}\right\}.\end{aligned} \tag{3.31}$$

In order to derive an upper bound for the second term, we first decompose each risk into bias and variance. According to [54], the risk of the empirical estimate of $\hat{f}_{G'}$ can be decomposed as

$$\begin{aligned}\mathcal{R}_g(f_{G'})&=\int_{\mathcal{X}\times\mathcal{Y}}\left(g(y)-\hat{f}_{G'}(x)\right)^2 dP_{XY}\\&=\int_{\mathcal{X}\times\mathcal{Y}}\left(g(y)-f_{Y|X}(x)\right)^2 dP_{XY}+\int_{\mathcal{X}\times\mathcal{Y}}\left(f_{X|Y}(x)-\hat{f}_{G'}(x)\right)^2 dP_{XY},\end{aligned} \tag{3.32}$$

where $f_{X|Y}(x):=\int_{\mathcal{Y}}g(y)dP_{Y|X}$ is the optimal regression function.

By assuming $f_{X|Y}(x)=f^*_{G^*}$ (this is the assumption we use throughout this chapter), we have

$$\left|\mathcal{R}_g(\hat{f}_{G'})-\mathcal{R}_g(f^*_{G^*})\right|=\int_{\mathcal{X}\times\mathcal{Y}}\left(f^*_{G^*}(x)-\hat{f}_{G'}(x)\right)^2 dP_{XY} \tag{3.33}$$

According to Theorem 2.1 in [55], we have the following decompositions for the two function on the RHS of (3.33):

$$\begin{aligned}f^*_{G^*}&=\sum_{\boldsymbol{u}\subseteq\{1,\ldots,p\}}f^*_{G^*,\boldsymbol{u}}\quad\text{with}\quad f^*_{G^*,\boldsymbol{u}}:=\sum_{\boldsymbol{v}\subseteq\boldsymbol{u}}(-1)^{|\boldsymbol{u}|-|\boldsymbol{v}|}\,\mathrm{P}_{\{1,\ldots,p\}\setminus\boldsymbol{v}}(f^*_{G^*}),\\\hat{f}_{G'}&=\sum_{\boldsymbol{u}\subseteq\{1,\ldots,p\}}\widehat{f}_{G',\boldsymbol{u}}\quad\text{with}\quad\widehat{f}_{G',\boldsymbol{u}}:=\sum_{\boldsymbol{v}\subseteq\boldsymbol{u}}(-1)^{|\boldsymbol{u}|-|\boldsymbol{v}|}\,\mathrm{P}_{\{1,\ldots,p\}\setminus\boldsymbol{v}}(\widehat{f}_{G'}).\end{aligned}$$

Since G' is an non-amiable group structure, there is at least one subset[4] of $\boldsymbol{u} \subseteq \{1,\ldots,p\}$ such that $f^*_{G^*,\boldsymbol{u}} \neq \hat{f}_{G',\boldsymbol{u}}$. Let $C = \min_{\boldsymbol{u}\subseteq\{1,\ldots,p\}} \int_{\mathcal{X}\times\mathcal{Y}} \left(f^*_{G^*,\boldsymbol{u}} - \hat{f}_{G',\boldsymbol{u}}\right)^2 dP_{XY} > 0$ and denote $\boldsymbol{u}^c = \{1,\ldots,p\} \setminus \boldsymbol{u}$, then we have

$$\begin{aligned}
&\int_{\mathcal{X}\times\mathcal{Y}} \left(f^*_{G^*}(\boldsymbol{x}) - \hat{f}_{G'}(\boldsymbol{x})\right)^2 dP_{XY} \\
=&\int_{\mathcal{X}_{\boldsymbol{u}}\times\mathcal{Y}} \left(f^*_{G^*,\boldsymbol{u}}(\boldsymbol{x}_{\boldsymbol{u}}) - \hat{f}_{G',\boldsymbol{u}}(\boldsymbol{x}_{\boldsymbol{u}})\right)^2 dP_{X_{\boldsymbol{u}}Y} + \int_{\mathcal{X}_{\boldsymbol{u}^c}\times\mathcal{Y}} \left(f^*_{G^*,\boldsymbol{u}^c}(\boldsymbol{x}_{\boldsymbol{u}^c}) - \hat{f}_{G',\boldsymbol{u}^c}(\boldsymbol{x}_{\boldsymbol{u}^c})\right)^2 dP_{X_{\boldsymbol{u}^c}Y} \\
\geq& C + \int_{\mathcal{X}\times\mathcal{Y}} \left(f^*_{G^*,\boldsymbol{u}^c}(\boldsymbol{x}_{\boldsymbol{u}^c}) - \hat{f}_{G',\boldsymbol{u}^c}(\boldsymbol{x}_{\boldsymbol{u}^c})\right)^2 dP_{XY} \\
>& 0. \qquad (3.34)
\end{aligned}$$

where the first equality is due to the orthogonality possessed by a direct sum Hilbert space.

By using (3.30), (3.31), (3.33) and (3.34), we can obtain

$$P\left(\left|\widehat{\mathcal{R}}_g(\hat{f}_{G'}) - \mathcal{R}_g(f^*_{G^*})\right| > \epsilon + C\right) \leq 12n \cdot \exp\left\{\ln \mathcal{N}\left(\frac{\epsilon}{6}, \mathcal{H}_{G'}, \ell_\infty\right) - \frac{\epsilon^2 n}{36}\right\} \qquad (3.35)$$

■

By combining Theorem 3.3.2 and Proposition 3.3.1, we eventually achieve the following selection consistency result for AGSI.

Theorem 3.3.3 *Let $\lambda n \to 0$. By choosing a proper tuning parameter $\mu > 0$ for the capacity penalty , the group structure $\widehat{G}$ that minimizes (3.23) is consistent.*

Proof According to Theorem 3.3.2, by choosing $\epsilon < C$, an agreeable group structure will be chosen with high probability.

For an amiable group structure, let $\epsilon_1 = \left|\widehat{\mathcal{R}}_g(\hat{f}^{\lambda}_G) - \mathcal{R}_g(f^*_{G^*})\right|$ and $\epsilon_2 = \mu\boldsymbol{\mathcal{C}}(G) - \mu\boldsymbol{\mathcal{C}}(G^*)$. Since $\boldsymbol{\mathcal{C}}(G) > \boldsymbol{\mathcal{C}}(G^*)$ when G is not the true group structure, we have $\epsilon_2 > 0$. Because ϵ_1 converges to 0 in probability. Thus the true group structure G^* will be picked with high probability if Problem (3.21) is solved. ■

[4]If G' is amiable, then a subset $\boldsymbol{u}$ of G' always assumes an additive structure. So there is no error between $f^*_{G^*}$ and $\hat{f}_{G'}$ after such a decomposition.

3.3.2 Supporting Lemmas

This section includes the lemmas (with proof) that are used to prove the Theorems and Proposition in the previous section.

Lemma 3.3.1 *For all $\epsilon > 0$ and all $n > 2/\epsilon^2$,*

$$P\left(\left|\mathcal{R}_g\left(\hat{f}_G\right) - \mathcal{R}_g\left(f^*_{G^*}\right)\right| > \frac{\epsilon}{2}\right) \leq$$
$$12n \cdot \ln \mathbb{E}\left[\mathcal{N}\left(\frac{\epsilon^2}{12}, \mathcal{H}_G, \ell_\infty^{X'}\right)\right] \exp\left\{-n\left(\frac{\epsilon}{24} - \frac{\lambda_n \left\|f^*_{G^*}\right\|^2}{12}\right)^2\right\}$$

Proof Due to the uniform convergence bound (12.135) in [12], given $\widehat{f}_G$,, we have for all $\epsilon > 0$ and all $n \geq 2/\epsilon^2$,

$$P\left(\left|\widehat{\mathcal{R}}_g\left(\hat{f}_G\right) - \mathcal{R}_g\left(\hat{f}_G\right)\right| > \epsilon\right) \leq 12n \cdot \mathbb{E}\left[\mathcal{N}\left(\frac{\epsilon^2}{12}, \mathcal{H}_G, \ell_\infty^{X'}\right)\right] \exp\left\{-\frac{\epsilon^2 n}{36}\right\}.$$

By setting $\delta = 12n \cdot \mathbb{E}\left[\mathcal{N}\left(\frac{\epsilon^2}{12}, \mathcal{H}_G, \ell_\infty^{X'}\right)\right] \exp\left\{-\frac{\epsilon^2 n}{36}\right\}$ and solve for ϵ, we have

$$\epsilon = 6n^{-1/2}\left(\ln 12n + \ln \mathbb{E}\left[\mathcal{N}\left(\frac{\epsilon^2}{12}, \mathcal{H}_G, \ell_\infty^{X'}\right)\right] - \ln \delta\right)^{1/2}.$$

Equivalently with probability at least $1 - \delta$,

$$\left|\widehat{\mathcal{R}}_g\left(\hat{f}_G\right) - \mathcal{R}_g\left(\hat{f}_G\right)\right| \leq 6n^{-1/2}\left(\ln 12n + \ln \mathbb{E}\left[\mathcal{N}\left(\frac{\epsilon^2}{12}, \mathcal{H}_G, \ell_\infty^{X'}\right)\right] - \ln \delta\right)^{1/2}.$$

Due to the symmetry of the above bound, we have with probability at least $1-\delta$,

$$\begin{aligned}
\mathcal{R}_g\left(\hat{f}_G\right) &\leq \widehat{\mathcal{R}}_g\left(\hat{f}_G\right) + 6n^{-1/2}\left(\ln 12n + \ln \mathbb{E}\left[\mathcal{N}\left(\frac{\epsilon^2}{12}, \mathcal{H}_G, \ell_\infty^{X'}\right)\right] - \ln \delta\right)^{1/2} \\
&\leq \widehat{\mathcal{R}}_g\left(\hat{f}_G\right) + \lambda\left\|\hat{f}_G\right\|^2 + 6n^{-1/2}\left(\ln 12n + \ln \mathbb{E}\left[\mathcal{N}\left(\frac{\epsilon^2}{12}, \mathcal{H}_G, \ell_\infty^{X'}\right)\right] - \ln \delta\right)^{1/2} \\
&\leq \widehat{\mathcal{R}}_g\left(f^*_{G^*}\right) + \lambda\left\|f^*_{G^*}\right\|^2 + 6n^{-1/2}\left(\ln 12n + \ln \mathbb{E}\left[\mathcal{N}\left(\frac{\epsilon^2}{12}, \mathcal{H}_G, \ell_\infty^{X'}\right)\right] - \ln \delta\right)^{1/2} \\
&\leq \mathcal{R}_g\left(f^*_{G^*}\right) + \lambda\left\|f^*_{G^*}\right\|^2 + 12n^{-1/2}\left(\ln 12n + \ln \mathbb{E}\left[\mathcal{N}\left(\frac{\epsilon^2}{12}, \mathcal{H}_G, \ell_\infty^{X'}\right)\right] - \ln \delta\right)^{1/2}
\end{aligned}$$

where the third inequality is due to the definition of $\hat{f}_G$ as the minimizer of the empirical problem. We applied the uniform convergence bound twice, one for the first inequality and the other for the last inequality.

Since it is always true that $\mathcal{R}_g\left(f^*_{G^*}\right) \leq \mathcal{R}_g\left(\hat{f}_G\right)$, we have the symmetric upper bound with probability $1-\delta$,

$$\left|\mathcal{R}_g\left(\hat{f}_G\right) - \mathcal{R}_g\left(f^*_{G^*}\right)\right| \leq \lambda \left\|f^*_{G^*}\right\|^2 + 12n^{-1/2}\left(\ln 12n + \ln \mathbb{E}\left[\mathcal{N}\left(\frac{\epsilon^2}{12}, \mathcal{H}_G, \ell_\infty^{X'}\right)\right] - \ln\delta\right)^{1/2}.$$

By setting $\lambda \left\|f^*_{G^*}\right\|^2 + 12n^{-1/2}\left(\ln 12n + \ln \mathbb{E}\left[\mathcal{N}\left(\frac{\epsilon^2}{12}, \mathcal{H}_G, \ell_\infty^{X'}\right)\right] - \ln\delta\right)^{1/2} = \epsilon/2$ and solve for δ, we have

$$\delta = 12n \cdot \ln \mathbb{E}\left[\mathcal{N}\left(\frac{\epsilon^2}{12}, \mathcal{H}_G, \ell_\infty^{X'}\right)\right] \exp\left\{-n\left(\frac{\epsilon}{24} - \frac{\lambda_n \left\|f^*_{G^*}\right\|^2}{12}\right)^2\right\}$$

Thus the bound for the second term is for all $\epsilon > 0$ and all $n > 2/\epsilon^2$,

$$P\left(\left|\mathcal{R}_g\left(\hat{f}_G\right) - \mathcal{R}_g\left(f^*_{G^*}\right)\right| > \frac{\epsilon}{2}\right) \leq 12n \cdot \ln \mathbb{E}\left[\mathcal{N}\left(\frac{\epsilon^2}{12}, \mathcal{H}_G, \ell_\infty^{X'}\right)\right] \exp\left\{-n\left(\frac{\epsilon}{24} - \frac{\lambda_n \left\|f^*_{G^*}\right\|^2}{12}\right)^2\right\}$$

■

The following Lemma is taken from Lemma 1 in [56], which shows the relationship between the covering number of the direct sum of two operators and the covering numbers of the individual operators.

Lemma 3.3.2 *Let $S, T : \mathcal{B}_1 \to \mathcal{B}_2$ be operators in real Banach spaces and $\epsilon, \delta > 0$. Then,*

$$\mathcal{N}\left(\epsilon + \delta, T + S\right) \leq \mathcal{N}\left(\epsilon, T\right) \cdot \mathcal{N}\left(\delta, S\right).$$

3.4 Simulation Study

In this section, we evaluate the performance of AGSI for nonparametric regression using synthetic data. In the first simulation experiment, we show that our method can indeed identify the true group structure with properly chosen tuning parameters. In

the second experiment, we demonstrate the procedure of tuning parameter selection by augmenting a validation step in exhaustive search. In the third experiment, we show the performance of the backward stepwise procedure.

3.4.1 Effectiveness of Exhaustive Search

In this study, we apply AGSI for nonparametric regression on several selected models to show that the exhaustive search method has the ability to identify the true group structure if the tuning parameters are chosen properly. The size of each model is restricted to be small so that the exhaustive algorithms can be applied. Specifically, the five models listed in Table 3.1 along with their true group structure are used.

Table 3.1.
Selected models for the simulation study using the exhaustive search method and the corresponding additive group structures.

ID	Model	True Group Structure
M1	$y = 2x_1 + x_2^2 + x_3^3 + \sin(\pi x_4) + \log(x_5 + 5) + \lvert x_6\rvert + \epsilon$	$\{(1), (2), (3), (4), (5), (6)\}$
M2	$y = \frac{1}{1+x_1^2} + \arcsin\left(\frac{x_2+x_3}{2}\right) + \arctan\left((x_4 + x_5 + x_6)^3\right) + \epsilon$	$\{(1), (2,3), (4,5,6)\}$
M3	$y = \arcsin\left(\frac{x_1+x_3}{2}\right) + \frac{1}{1+x_2^2} + \arctan\left((x_4 + x_5 + x_6)^3\right) + \epsilon$	$\{(1,3), (2), (4,5,6)\}$
M4	$y = x_1 \cdot x_2 + \sin((x_3 + x_4) \cdot \pi) + \log(x_5 \cdot x_6 + 10) + \epsilon$	$\{(1,2), (3,4), (5,6)\}$
M5	$y = \exp\left\{\sqrt{x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2}\right\} + \epsilon$	$\{(1,2,3,4,5,6)\}$

The observations of $\boldsymbol{X}$ are independently stimulated. The distributions of $\boldsymbol{X}$ are standard normal in M1, Uniform$(-1, 1)$ in M2 and M3, and Uniform$(0, 2)$ in M4 and M5. The noise ϵ is i.i.d. normal with mean 0 and standard deviation 0.01. During each simulation, a data set of size 500 is generated which is used to estimate $\widehat{G}$ as a solution in Problem (3.23) for each (μ, α) pair in a provided grid. In constructing

the grid, the values of μ are chosen to be equally spaced from 1e$-$10 to 1/64 on the log-scale and the values of α are the integers from 1 to 10 inclusive. So there are 50 different (μ, α) pairs in the grid. The simulation is performed 100 times for each model.

We are interested in knowing if the true group structure can be identified frequently in each model setting. If there are (μ, α) pairs for each model that its true group structure can be often identified, this means that our novel penalty has the potential to identify true group structures. The frequencies that the true group structures are identified are calculated for each (μ, α) pair under each model setting. In Table 3.2, we report the maximum frequency that the true group structure is identified and the corresponding pair of tuning parameters under each model setting[5]. It can be seen in Table 3.2 that the true group structures can be successfully identified when the tuning parameters are properly chosen.

Table 3.2.
Maximum frequencies that the true group structures are identified for the five selected models using exhaustive search algorithm without tuning parameter selection.

Model	Max freq.	μ	α
M1	100	1.2500e-06	10
M2	97	1.2500e-06	8
M3	97	1.2500e-06	9
M4	100	1.2500e-06	7
M5	100	1.2500e-06	1

[5]There are ties for a model that different pairs of (μ, α) result in the same maximum frequency. In such a case, the pair of parameter values corresponding to the maximum frequency is randomly chosen and reported in Table 3.2.

The complete results are reported in Table 3.3 which are also visualized using the 3D surface plots in Figure 3.1. While considering all selected values for the tuning parameters, we can see that Model 1 performs the best. Within a wide range of parameter values ($\mu \geq 1.25\text{e}{-6}$ and $\alpha \geq 3$), there are records of correct selection of the true group structure. When the values of the parameters become larger, the true group structure can be identified 100 percentage of time. This is mostly due to the fact that large values of the capacity penalty prefers a simple model, and the fully additive structure is the simplest among all group structures. Model 2 and 3 perform comparably well. Within the middle range of μ (between 1.2500e-06 and 1.3975e-04), the true group structures are identified with high frequencies. Model 4 performs well in the similar range as that for Model 2 and 3. It also shows good performance at some higher value of μ and α. In Model 5, the true group structure is most often identified when α is towards the lower end of the range. This is due to the fact that small penalty favors larger model, and Model 5 has the "largest" group structure.

3.4.2 Tuning Parameters for Exhaustive Search

In this simulation study, we assume that the tuning parameters are not known and need to be learned from data. We simulate data by using the same five models in Table 3.1 and then apply Algorithm 2 to estimate the group structure and the transformation functions.

During each simulation, two independent data sets of size 500 are generated, one for training and the other for validation. For each (μ, α) pair, Problem (3.21) is solved by applying the exhaustive search algorithm to obtain the best estimate of the group structure $\widehat{G}$ and the corresponding transformations $\hat{f}_{\boldsymbol{u}}$'s. Then, the estimated model is applied on the validation set to calculate the goodness of fit measure R^2. The model with the largest value of R^2 is retained as the best and its $\widehat{G}$ is considered as the estimated true group structure. The simulation is repeated 100 times for each model in Table 3.1.

Table 3.3.
Frequencies that the true group structures are selected under different parameter pairs for the six models.

μ	α	M1	M2	M3	M4	M5
1.0000e-10	1.00	0	0	0	0	100
1.0000e-10	2.00	0	0	0	0	100
1.0000e-10	3.00	0	0	0	0	100
1.0000e-10	4.00	0	0	0	0	99
1.0000e-10	5.00	0	0	0	0	10
1.0000e-10	6.00	0	0	0	0	0
1.0000e-10	7.00	0	0	0	0	0
1.0000e-10	8.00	0	0	0	0	0
1.0000e-10	9.00	0	0	0	0	0
1.0000e-10	10.00	0	0	0	0	0
1.1180e-08	1.00	0	0	0	0	100
1.1180e-08	2.00	0	0	0	0	98
1.1180e-08	3.00	0	0	0	0	0
1.1180e-08	4.00	0	0	0	0	0
1.1180e-08	5.00	0	0	0	0	0
1.1180e-08	6.00	0	0	0	0	0
1.1180e-08	7.00	0	0	0	0	0
1.1180e-08	8.00	0	0	0	1	0
1.1180e-08	9.00	0	0	0	77	0
1.1180e-08	10.00	0	0	0	92	0
1.2500e-06	1.00	0	0	0	0	100
1.2500e-06	2.00	0	0	0	0	0
1.2500e-06	3.00	14	0	0	84	0
1.2500e-06	4.00	81	3	4	99	0
1.2500e-06	5.00	90	77	77	99	0
1.2500e-06	6.00	94	92	90	99	0
1.2500e-06	7.00	96	96	95	100	0
1.2500e-06	8.00	98	97	96	100	0
1.2500e-06	9.00	98	97	97	100	0
1.2500e-06	10.00	100	97	97	100	0
1.3975e-04	1.00	0	0	0	0	100
1.3975e-04	2.00	0	95	93	100	0
1.3975e-04	3.00	100	95	92	90	0
1.3975e-04	4.00	100	28	23	9	0
1.3975e-04	5.00	100	13	12	3	0
1.3975e-04	6.00	100	5	7	3	0
1.3975e-04	7.00	100	0	0	2	0
1.3975e-04	8.00	100	0	0	0	0
1.3975e-04	9.00	100	0	0	0	0
1.3975e-04	10.00	100	0	0	0	0
1.5625e-02	1.00	0	0	0	0	100
1.5625e-02	2.00	0	0	0	100	0
1.5625e-02	3.00	100	0	0	0	0
1.5625e-02	4.00	100	0	0	0	0
1.5625e-02	5.00	100	0	0	0	0
1.5625e-02	6.00	100	0	0	0	0
1.5625e-02	7.00	100	0	0	0	0
1.5625e-02	8.00	100	0	0	0	0
1.5625e-02	9.00	100	0	0	0	0
1.5625e-02	10.00	100	0	0	0	0

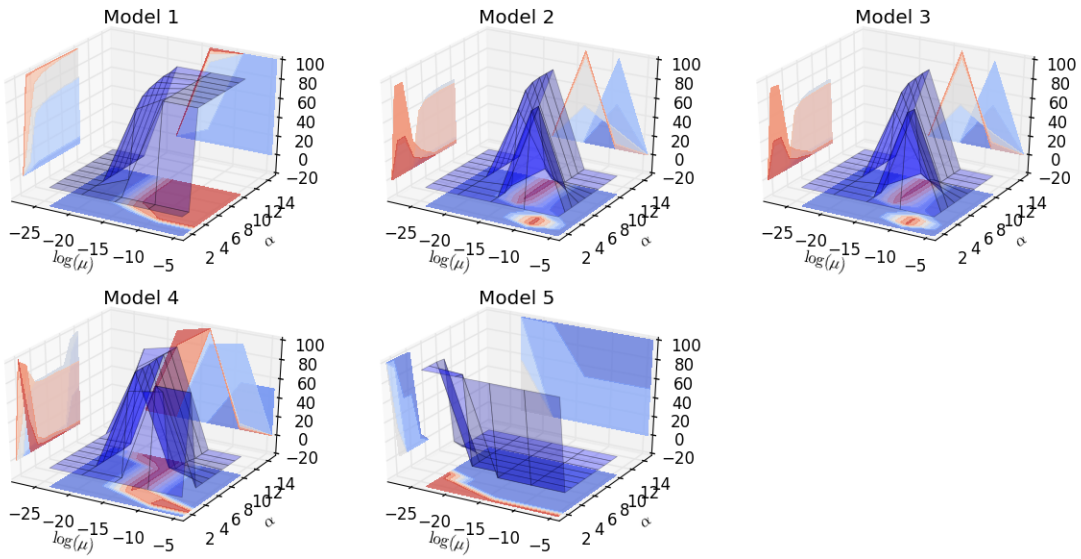

Fig. 3.1. The 3D surface of the frequencies (out of 100) that the true group structures are identified for the five chosen models in Table 3.1 over the entire parameters grid. Given a (μ, α) pair, the penalized goodness of fit is calculated for all group structures. We recorded each time the true group structure is identified. The values of μ are reported in log-scale. Each surface plot is accompanied with three contour plots as the 2D projections of the surface to enhance the effect of the visualization.

The maximum frequency that the true group structure is identified is reported for each model in Table 3.4[6]. We can see from Table 3.4 that the optimal tuning parameters are quite close to those in Table 3.2. Except for Model 1, the frequencies of identifying true group structure are very close to 100. This indicates that simply adding a validation step can help to select the optimal values for the tuning parameter. In practice, a more general cross validation (CV) procedure can be used. The deteriorated performance of Model 1 might be caused by the estimation method (Kernel Ridge Regression to solve Problem (3.22)) used in the algorithm. It is also affected by the choice for the value of the third turning parameter λ.

[6]There are ties for a model that multiple pairs of (μ, α) result in the same maximum frequency. In such a case, the pair of parameter values corresponding to the maximum frequency is randomly chosen and reported in Table 3.4.

Table 3.4.
Maximum frequencies that the true group structures are identified for the five chosen models using exhaustive search algorithm with tuning parameter selection.

Model	Max freq.	μ	α
M1	59	1.2500e-06	4
M2	89	1.2500e-06	7
M3	89	1.2500e-06	7
M4	99	1.2500e-06	4
M5	100	1.2500e-06	1

The complete results are visualized by 3D surface plots in Figure 3.2. Again, we can see that except Model 1, the performance of the other models is quite similar to that in the previous simulation study.

3.4.3 Stepwise Approach

When the number of predictor variables is large, using the exhaustive search algorithms is not practical. Instead, a stepwise algorithm would be a more reasonable choice in terms of computational cost. In this simulation study, we show the performance of our backward algorithm. We apply Algorithm 3 on the data simulated from the same models listed in Table 3.1.

In order to select the values for the tuning parameters, we use a training data set to estimated the OKGT model which is then used to calculate the prediction error on an independent validation set. We use the same grid values for μ and α.

During each simulation, the backward algorithm is applied on the training data to estimate the group structure and the corresponding transformation functions for each (μ, α) pair. Then, the estimated model is used to calculated the fitting error

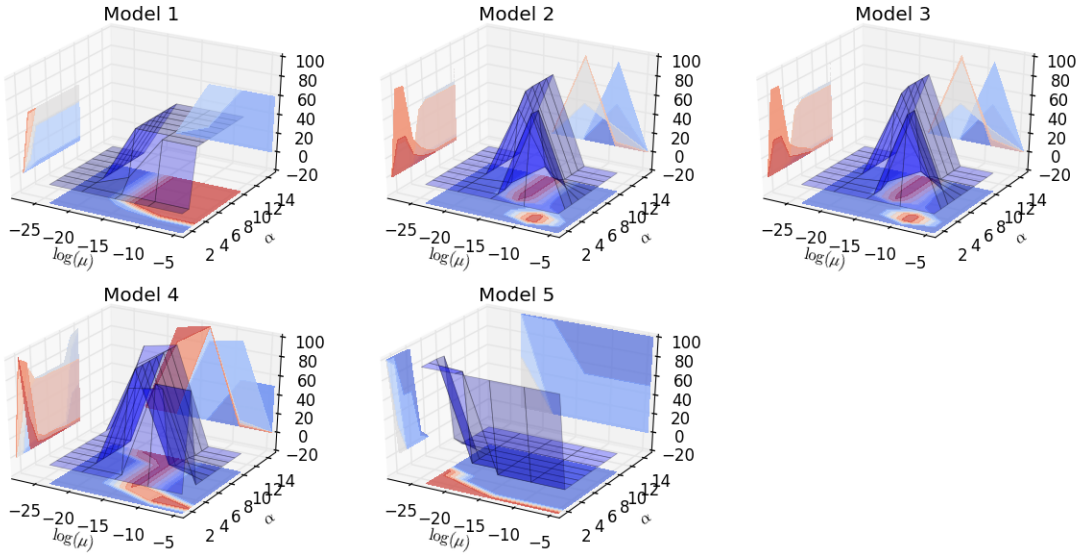

Fig. 3.2. The 3D surfaces of the frequencies (out of 100) that the true group structures are identified for the five chosen models in Table 3.1 over the entire parameter grids. The training procedure uses a separate validation data set to select the optimal tuning parameters (μ, α). The values of μ are reported in log-scale. Each surface plot is accompanied with three contour plots as the 2D projections of the surface to enhance the effect of the visualization.

on the validation data. The model which results in the smallest validation error is retained as the best model. This simulation procedure is repeated 100 time for each model and we record the frequencies that the true group structures are recovered.

The complete results are visualized by the 3D surface plots in Figure 3.3. Model 1 and 5 can be successfully identified almost all the time within some range of the tuning parameters. Though the frequencies for Model 2 and 3 are not as high as those obtained from applying the exhaustive search algorithms, the true models can still be selected most of time. The performance of the backward algorithm on Model 4 was not satisfying. Since the backward algorithm search in a greedy fashion for the best direction to update during each iteration, it is possible that the true group

structures were never visited. Nevertheless, further research is needed to have a deep understanding of the role played by the complexity penalty in stepwise algorithms.

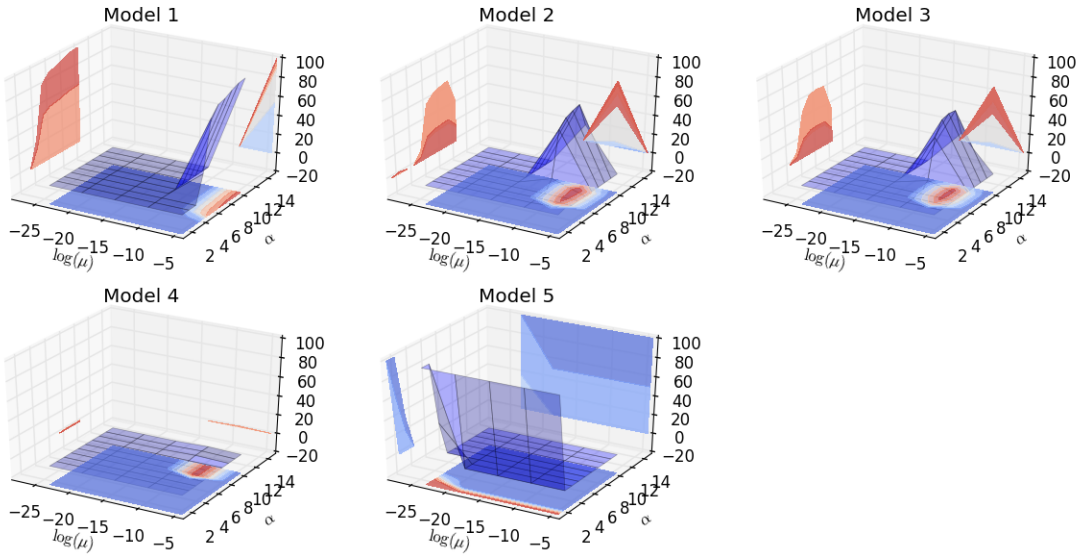

Fig. 3.3. The 3D surfaces of the frequencies (out of 100) that the true group structures are identified for the five chosen models in Table 3.1 over the entire parameter grids. The training uses the backward stepwise algorithm and the procedure uses a separate validation data set to select the optimal tuning parameters (μ, α). The values of μ are reported in log-scale. Each surface plot is accompanied with three contour plots as the 2D projections of the surface to enhance the effect of the visualization.

3.5 Real Data Applications

In this section, two real data sets will be used to show how AGSI for nonparametric regression can be used in data analysis. They are both available from the UCI machine learning repository.

3.5.1 Boston Housing Data

In this study, we apply AGSI for nonparametric regression on the famous housing data concerning housing values in suburbs of Boston. The data set includes thirteen predictor variables about community and property related attributes and median value of owner-occupied homes as the response. The sample size is 506. Our goal is to identify a possible group structure for the predictor variables in terms of their effect on the housing values.

Since the number of possible group structures for thirteen variables is large, we use the backward algorithm for identifying the true group structure. In order to select the proper tuning parameters μ and α for the capacity penalty, a 10-fold CV is implemented. The data set is (almost) equally divided into 10 subsets. Each time, Algorithm 3 is applied on 9 subsets, then the estimated functions are used to calculate the prediction error on the left-out piece. The values of the tuning parameters are chosen to be the pair corresponding to the smallest average prediction error.

The average prediction errors are plotted for each value pairs of the tuning parameters (where μ and α are arranged in increasing order) in Figure 3.4. The values of the average prediction error on the y-axis are log-scaled. The minimum value of the prediction error is 3.75203, which is corresponding to the four parameter pairs (1.0e-10, 2), (1.12e-8, 2), (1.25e-6, 2), and (1.3975e-4, 2).

In this study, a small set of eight group structures were identified. They all achieved relatively low prediction errors. Among the eight group structures, the group structure $\{(1,6),(2,11),(3),(4,9),(5,8),(7,13),(10,12)\}$ achieved lowest average prediction error. So we use it for a detailed investigation.The nonparametric functions for each group based on the whole data set. Because the groups contains no more than two variables, the estimated functions can be visualized. Selected estimation results are shown in Figure 3.5.

It is interesting to see some patterns emerging in the plots. The first plot show the function of the average number of rooms per dwelling and per capita crime rate

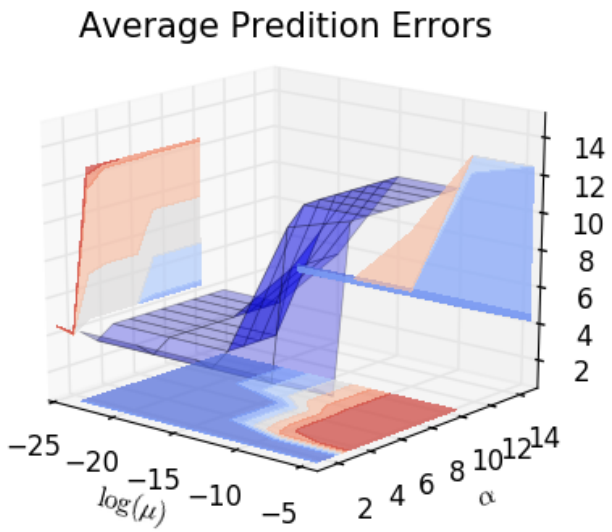

Fig. 3.4. The results of applying the backward step-wise algorithm on Boston Housing data with 10-fold CV. The 3D surfaces shows the average validation error over the entire grid of (μ, α) pairs. The surface plot is accompanied with three contour plots as the 2D projections of the surface to enhance the effect of the visualization.

by town. It shows the value of houses increase as there are more rooms and decreases as the crime rate increases. However, at the low end of the crime rate, smaller sized houses (4 or 5 rooms) seem to be preferred than a house with around 6 rooms. The second plot (top-right) shows that there is a changing point in terms of how house value is related to the size of non-retail business in the area. The value initially drops when the percentage of non-retail business is small, then increases at around 8%. The increase in the value might be due to the high demand of housing from the employees of those business. The third plot (bottom-left) reveals an interesting pattern of nitric oxides concentration and weighted distances to five Boston employment centers. There are two changing points regarding how people value a house in terms of its distance from working. The value reaches the minimum at around 1.5 and the maximum at around 3.5. This might because houses next to the working places are

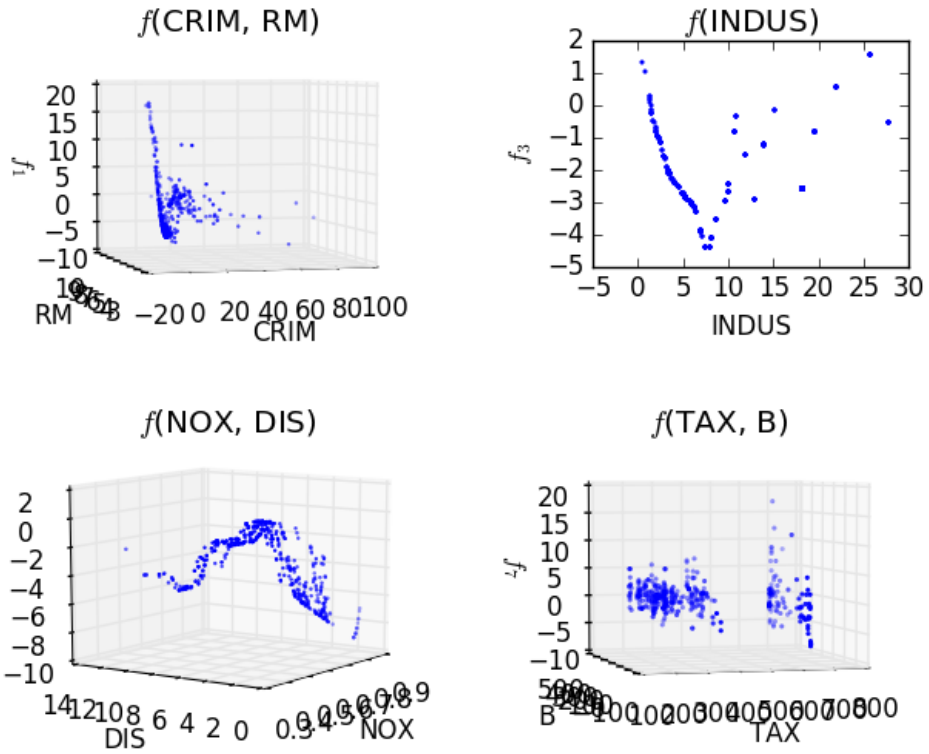

Fig. 3.5. Estimated transformation functions for selected groups in the chosen group structure $\{(1,6),(2,11),(3),(4,9),(5,8),(7,13),(10,12)\}$. Top-left: group $(1,6)$, top-right: group (3), bottom-left: group $(5,8)$, bottom-right: group $(10,12)$.

appealing to those who do not have cars. The houses in the moderate distance are preferred by those who have more mobility but do not want to live too far from work. The concentration of nitric oxides affects a house value negatively when it is close to the employment centers but not much when the houses are remote.

3.5.2 Communities and Crime Data

In the second application, we choose to apply our method on the communities and crime data also available on UCI repository. There are two versions of the data set, normalized and unnormalized. We use the unnormalized version for our study. The

data set combines socio-economic data from the '90 Census, law enforcement data from the 1990 Law Enforcement Management and Admin Stats survey, and crime data from the 1995 FBI UCR.

There are 2215 samples and 147 variables in the data set with missing values. Besides four variables for identification purpose, there are 122 predictor variables and 18 crime attributes that could be treated as response. We choose Number of Murders in 1995 to be our response in this study and investigate its relationship between the predictor variables. While missing values are encountered, they are simply removed.

Because of the large number of predictor variables in the data set. A preliminary screening procedure is applied to reduce the number of variables and select the most related predictors for our goal. Since OKGT can be used to determine the dependency between two variables, we fit OKGT for each of the 122 predictor against the response to obtain the estimate of the dependence measure R^2. Then, the predictor variables are kept if its corresponding $R^2 > 0.99$. This indicates that the marginal dependence between the selected predictor and the response are very high.

After screening, the number of predictor variables is reduced to 23. Some of them include Total Requests for Police Per 100K Population, Number of People in Homeless Shelters, Per Capita Income for People with Asian Heritage, and Land Area. Then the backward algorithm was ran on the reduced data set. Because of the missing value issue, the sample size of the reduced data set is reduced to 343. The optimal group structure is determined by a simple one-fold validation.

The procedure selected the fully additive group structure as the optimal one. We reported the estimated results for the four selected groups in Figure 3.6. They show highly nonlinear relationship between each predictor variable and the number of murders. The first plot shows that the effect of Median Family Income is almost zero until it reaches the high end where murders drop dramatically. The second plot shows an interesting pattern for Total Requests for Police per Police Officer. As the number of requests increases, the number of murders initially decreases slowly. One reason for this is that increasing requests cause more presence of police in the area

which is helpful to control crimes. However, murders increase quickly as the number of requests enters the high range. An explanation for this is that the surging number of requests for police is due to the low security and high murder rate in the area.

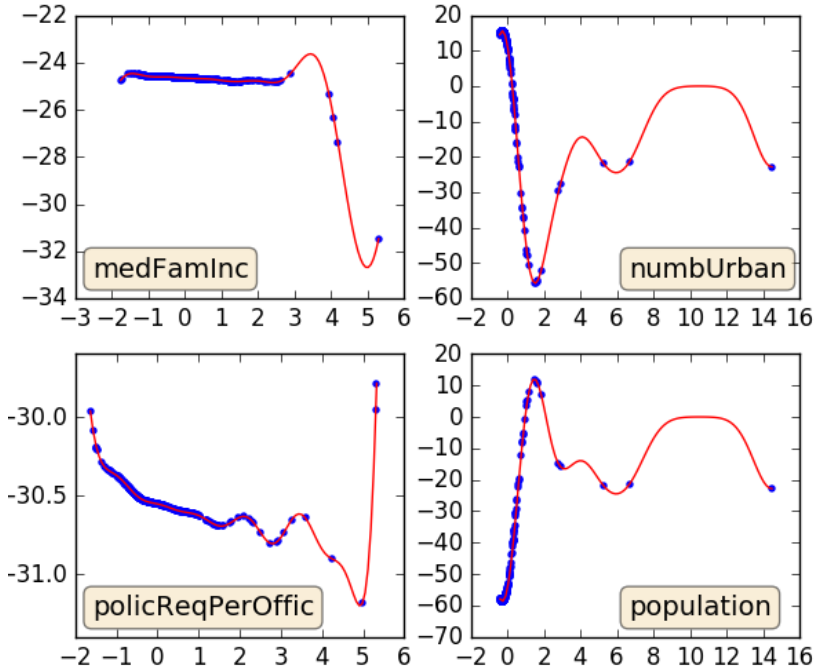

Fig. 3.6. Selected results for the communities and crime data where the number of murders is the response. The blue dots are the transformed observation of the predictor variable. The red line is the estimated function.

3.6 Summary

In this chapter, we have developed an effective method for general nonparametric regression analysis. By imposing an additive group structure, we achieve the goal of both preserving important interactions between the predictor variables and reducing the dimensionality of the problem for efficient estimation. In order to identify the true additive group structure, we proposed a novel complexity penalty on additive group structures and incorporated it into the penalized kernel regression method.

Simulation studies and real data applications demonstrate the effectiveness of our proposed method.

There are three main directions for future research. First, the theoretical properties of the proposed method, including selection consistency, need to be rigorously established. Second, our penalty is based on the covering number of RKHSs. It is of interest to know if there exist other more effective penalty. Third, it is noticed that the current backward stepwise algorithm may become unstable and fail to achieve the potential in identifying the true additive group structure as shown by the exhaustive algorithm. It is of great interest to further improve upon the current algorithm so that the proposed method can be applied in general high dimensional nonparametric regression.

4. HIERARCHICAL MIXED LOGISTIC REGRESSION MODEL AND ITS SPARK IMPLEMENTATION

In Chapter 2 and 3, we developed a new general framework for data exploration in regression setting. Our contribution includes proposing the concept of group structure which greatly generalizes the usual additive model and developing a method for identifying the optimal group structure for data. Though this framework has been shown useful for EDA, the methods were suitable for exploring data on a single machine. Since kernel methods usually have difficulty in handling large scale problems, especially when sample size is large, new methods are needed for exploring large datasets.

In this chapter, we are going to build and implement a model for exploring big data. When the sample size of data increases, we would expect the structure of data becomes more complex. So a big model that can accommodate these complex structures is needed. As heterogeneity is common in big data (see [57]), being able to identify hierarchical and clustering structures is fundamental in exploring big data. Hodas & Lerman (2013) [58] gives an example that aggregated exposure response obscures heterogeneous behavior in a study of social epidemics. A model for big data exploration should be helpful for detecting heterogeneity. After that, a more fine-grained data exploration or model building can be carried out targeting each subpopulation.

We choose to use Hierarchical Mixed Logistic Regression Model (HMLRM) for exploring data with categorical response. The reason that HMLRM is useful for EDA is in two-fold. First, logistic regression model is a simple yet effective statistical model. Its statistical property is well studied and results are easy to interpret. Second, a hidden variable is introduced in the model to accommodate possible heterogeneity.

By using this hidden layer, we can explicitly model the probabilities that a sample belongs to different subpopulations.

When sample size increases, estimating the model becomes more difficult. In order to apply HMLRM on a large dataset, we will implement the model in Apache Spark, one of the most popular distributed computing platform. Spark uses Resilient Distributed Dataset (RDD) as data abstraction and an in-memory computing model. Spark allows data to be persisted in memory, hence it enables faster computation. These features make Spark more efficient than Hadoop in data analysis.

The rest of the chapter is organized as follows. Section 4.1 discusses the construction of HMLRM and the details of its estimation using EM algorithm. Section 4.2 discusses the consideration of implementing HMLRM in Spark. We report the results of some simulation studies in Section 4.3. Section 4.4 concludes this chapter.

4.1 Hierarchical Mixed Logistic Regression Model

In this section, we introduce the construction of the Hierarchical Mixed Logistic Regression Model (HMLRM) and discuss the details of using EM algorithm for model estimation.

4.1.1 Notations

In order to facilitate the discussion, we first fix some notations.

We denote each observed response[1] as a K-vector, where $K \in \mathbb{N}$ is the number of category that an observation belongs to. So we have $K = 2$ for binary and binomial response variables and $K > 2$ for multi-class and multinomial response variables. Furthermore, if the response is a binary or multi-class Bernoulli[2] random variable, each observation is coded as a binary vector of size K with all but one zeros. For

[1] In machine learning literature, especially supervised learning, an observed response is called a label.

[2] A multi-class Bernoulli variables is a generalization of a Bernoulli (or binary) random variable which indicates the membership to one of more than two categories.

example, in the famous MNIST handwriting digit dataset[3] each observation of the response is a digit from 0 to 9. If $y_i = 1$, the corresponding coding is given by the 10-vector $(0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0)^\top$. If the response is a binomial or multinomial random variable, each observation is coded as a K-vector where each element being the number of occurrence in the corresponding category and the sum of the elements equals the total number of trials.

Since we are dealing with a mixture model, one of our model assumptions is that an observation belongs to one of several hidden subpopulations. For example, people who provided the handwriting digits in the MNIST dataset maybe divided into multiple categories according to their writing habits. This is a reasonable structural assumption for a medium to large sized dataset. Since such kind of information cannot usually be obtained from observational study, they have to be imputed from the data itself. Though one can build up such a hierarchical model with more than one hidden layers of this kind, we restrict our attention to the mixed model with only one hidden layer. Further extension can be straightforward.

We introduce the hidden variable $Z_i \sim \text{MBern}\,(\pi_{1,i}, \ldots, \pi_{C,i})$ to denote the hidden membership of the i-th sample, where MBern denotes a multi-class Bernoulli distribution, $C \in \mathbb{N}$ denotes the fixed number of groups in the hidden layer and each $\pi_{c,i}$ for $c = 1, \ldots, C$ is the probability that the sample belongs to the c-th hidden subpopulation. All Z_i's for $i = 1, \ldots, n$ are assumed to be i.i.d. .

When $C = 2$, MBern becomes a Bernoulli distribution. For notational simplicity, we will often use the vector $\boldsymbol{\pi}_i = (\pi_{1,i} \quad \ldots \quad \pi_{C,i})^\top$ to represent the collection of the probabilities for one sample. The value of C is either given based on the prior knowledge or tuned as a hyper-parameter. Note that the current parameterization $\boldsymbol{\pi}_i$ is redundant because of the implicit constraint $\sum_{c=1}^{C} \pi_{c,i} = 1$. This will cause

[3]The MNIST database of handwritten digits has a training set of 60,000 examples, and a test set of 10,000 examples. It is a subset of a larger set available from NIST. The digits have been size-normalized and centered in a fixed-size image. The dataset is available for download at `http://yann.lecun.com/exdb/mnist/`.

identifiability issue and we will solve this problem by choosing one hidden group as a base group.

For each sample, conditional on that it belongs to the c-th hidden group, the distribution of its response $\{Y_i|Z_i = c\} \sim \text{MBern}\left(p_{1,i|c}, \ldots, p_{K,i|c}\right)$. Here, each $p_{k,i|c}$ for $k = 1, \ldots, K$ denotes the probability that the i-th sample fall into the k-th response category if it is a member of the c-th subpopulation. In the MNIST example, $p_{k=1,i|c}$ could be interpreted as the probability of being digit 1 if the person belongs to the writing habit group c. Being in a different hidden group would results in a different conditional probability for being in the same response category.

Note that there is also over-parameterization in the above notation due to the same type of constraint $\sum_{k=1}^{K} p_{k,i|c} = 1$. While there is only one such constraint in the hidden layer, there are in total C constraints in the observed layer, one for each hidden group. For convenience, we will often denote the vector of the conditional probabilities as $\boldsymbol{p}_{c,i} = (p_{1,i|c} \quad \ldots \quad p_{K,i|c})^\top$.

We use $\boldsymbol{\theta}$ to denote the collection of all parameters in the model. More notation will be introduced in the following discussion if needed.

4.1.2 Model Construction

In this section, we formally introduce the Hierarchical Mixed Logistic Regression Model (HMLRM) with a detailed description of its construction. There are two layers in our model. The first layer determines the subpopulation that a sample belongs to, and the second layer determines the response category for the sample. We refer to the first layer as the hidden layer, and the second layer as the observed layer.

The key step in constructing HMLRM is to impose a model for the hidden membership probabilities $\pi_{c,i}$ for all[4] $c = 1, \ldots, C$ and the conditional observing probabilities $p_{k,i|c}$ for all[5] $k = 1, \ldots, K$ and $c = 1, \ldots, C$. This modelling step is necessary to allow

[4]Though one of them will be redundant and treated as a base group.

[5]Though one $p_{k|c}$ for a given c is redundant and will be treated as a base group.

those probabilities to depend on some predictor variables. We impose the simple models[6] for those probabilities, which are based on sigmoid[7] and softmax[8] functions.

When there are exactly two response categories (which is the case for Bernoulli and Binomial response variables), the following sigmoid function is used to model the probability of being in one category,

$$p = \text{sigmoid}(z) = \frac{1}{1+e^{-z}}. \tag{4.1}$$

So the probability of being in the other category is given by $1-p = \frac{e^{-z}}{1+e^{-z}}$. Thus the probabilities add up to one.

When a response has $K > 2$ categories, the following softmax[9] function is used for modeling the probability of being in one of the first $K-1$ categories,

$$p_k = \frac{e^{z_k}}{1+\sum_{k=1}^{K-1} e^{z_k}}. \tag{4.2}$$

So we have $p_K = \frac{1}{1+\sum_{k=1}^{K-1} e^{z_k}}$ if the sum-one constraint has to be satisfied.

The exponent z (4.1) (or z_k in (4.2)) is a linear combination of the predictor variables. If we denote the predictor variables as $\boldsymbol{X} = (X_1, \ldots, X_p)$ and its observation as $\mathbf{x} = (x_1 \quad \ldots \quad x_p)^\top$, then we have $z = \boldsymbol{\beta}^\top \mathbf{x}$ where $\boldsymbol{\beta} = (\beta_1 \quad \ldots \quad \beta_p)^\top$ is the vector of the regression coefficients. Usually, we add a constant one at the beginning of each observation to accommodate the intercept[10]. This would give use the observation vector $\mathbf{x} = (1 \quad x_1 \quad \ldots \quad x_p)^\top$. Correspondingly, the coefficient vector becomes $\boldsymbol{\beta} = (\beta_0 \quad \beta_1 \quad \ldots \quad \beta_p)^\top$ where β_0 denotes the intercept.

Thus, the full model specification with the sigmoid function is given by

$$p = \frac{1}{1+e^{-\boldsymbol{\beta}^\top \boldsymbol{x}}} \quad \text{and} \quad 1-p = \frac{e^{-\boldsymbol{\beta}^\top \boldsymbol{x}}}{1+e^{-\boldsymbol{\beta}^\top \boldsymbol{x}}}. \tag{4.3}$$

[6]There are other options such as probit.

[7]The sigmoid function is also called "logistic" function.

[8]The multi-class generalization of the sigmoid function.

[9]The softmax function used here is slightly modified. The standard softmax function is given by $\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$ for $j = 1, \ldots, K$. In this chapter, we impose the restriction $z_K \equiv 0$ for the K-th group.

[10]An intercept is sometimes called a "bias" term in the machine learning language.

Similarly, the full model with the softmax function is written as

$$p_k = \frac{e^{\boldsymbol{\beta}_k^\top \boldsymbol{x}}}{1 + \sum_{k=1}^{K-1} e^{\boldsymbol{\beta}_k^\top \boldsymbol{x}}} \quad \text{and} \quad p_K = \frac{1}{1 + \sum_{k=1}^{K-1} e^{\boldsymbol{\beta}_k^\top \boldsymbol{x}}}. \tag{4.4}$$

Now, we can see clearly how the sum-one constraints are accounted. By using the sigmoid function, there is only one set of coefficients $\boldsymbol{\beta}$ which parameterizes the Bernoulli probability. Equivalently, $\boldsymbol{\beta}$ can be considered as the coefficients for the choosing probability p and $\boldsymbol{\beta} \equiv \mathbf{0}$ for $1 - p$. Thus, the group with probability $1 - p$ is the base group. This is more obvious when the softmax function is used, where the K-th group serves as the base group and its "coefficient vector" $\boldsymbol{\beta}$ is always zero. This model specification allows us to get around the identifiability issue in estimation for HMLRM. The base group is also called the *reference* or *baseline level*.

The sigmoid function in (4.1) is closely related to the logit function

$$\text{logit}(p) = \ln\left(\frac{p}{1-p}\right), \quad p \in (0, 1),$$

which is often used as a link function in a generalized linear model with binary response, i.e. $\text{logit}(p) = z = \boldsymbol{\beta}_k^\top \boldsymbol{x}$. The sigmoid function and the logit function are inverse to each other, that is

$$\text{logit}^{-1}(z) = \text{sigmoid}(z).$$

In addition, sigmoid function itself is a valid cumulative distribution function (CDF)[11]. It is used to describe binary logit models for discrete choice modelling in [59], where the utility of choosing one alternative can be modeled as a linear function of predictor variables (the systemic component) plus an error term following Gumbel distribution. The difference of two independent Gambel errors from two alternatives follows logistic distribution the sigmoid function as the CDF. So discrete choice modelling sees a logistic regression model from a different angle, where the randomness is from the difference of utilities by choosing different alternatives.

By using the sigmoid (or softmax) function, we are allowed to model the hidden layer explicitly, which gives raise "hierarchical" in the name of the model.

[11]Sometimes, a PDF is also called probability distribution function.

The probabilities $\pi_{c,i}$ and $p_{k,i|c}$ in the hidden and observed layers depend on two sets of predictor variables. This is especially helpful when there are different factors that influence the hidden layer and observed layer respectively. For example, for a cancer patient, the hidden layer can be used to model the probability that which stage the patient is at. This probability could depend on his/her medical examination results. Conditional on the hidden layer, the observed layer can be used to model the effect of different medical treatments on the patient. In particular, $p_{k,i|c}$ could be the probability of remission which depends on different treatment levels and his/her genetic attributes. Another example is given in [60] where the model is used to handle over-dispersion problem in the data from a study in evolutionary biology reported in [61]. The study was interested in knowing if the three species of adult Tribolium beetles have developed the ability to avoid eating eggs of their own species.

When there are multiple categories in both hidden and observed layers, the complete construction of HMLRM is formally given by

$$P(Y_i = y_i|\mathbf{x}_i^{(h)}, \mathbf{x}_i^{(o)}, \mathbf{A}, \mathbf{B}_1, \ldots, \mathbf{B}_C) = \sum_{c=1}^{C} \pi_{c,i} \cdot p_{y_i|c}, \tag{4.5}$$

where

$$\pi_{c,i} = \begin{cases} \frac{\exp\{\boldsymbol{\alpha}_c^\top \mathbf{x}_i^{(h)}\}}{1+\sum_{c=1}^{C-1} \exp\{\boldsymbol{\alpha}_c^\top \mathbf{x}_i^{(h)}\}} & \text{if } c \neq C \\ \frac{1}{1+\sum_{c=1}^{C-1} \exp\{\boldsymbol{\alpha}_c^\top \mathbf{x}_i^{(h)}\}} & \text{if } c = C \end{cases} \tag{4.6}$$

$$p_{k|c} = \begin{cases} \frac{\exp\{\boldsymbol{\beta}_{k|c}^\top \mathbf{x}_i^{(o)}\}}{1+\sum_{k=1}^{K-1} \exp\{\boldsymbol{\beta}_{k|c}^\top \mathbf{x}_i^{(o)}\}} & \text{if } k \neq K \\ \frac{1}{1+\sum_{k=1}^{K-1} \exp\{\boldsymbol{\beta}_{k|c}^\top \mathbf{x}_i^{(o)}\}} & \text{if } k = K \end{cases} \tag{4.7}$$

where we use the new notation $p_{y_i|c} := P(Y_i = y_i|\boldsymbol{p}_{c,i})$. We also introduced some new notations to denote various subsets of observations and parameters. The predictor variable vectors for the hidden and observed layers are denoted as $\boldsymbol{X}^{(h)}$ and $\boldsymbol{X}^{(o)}$. Their dimensions are denoted as $d^{(h)}$ and $d^{(o)}$ respectively. Their observations are denoted as $\mathbf{x}^{(h)}$ and $\mathbf{x}^{(o)}$ respectively. Each of the C hidden groups (except the C-th

group) possess a vector of regression coefficients, denoted as $\boldsymbol{\alpha}_c$. They are collectively represented by the matrix $\mathbf{A} := (\boldsymbol{\alpha}_c \quad \ldots \quad \boldsymbol{\alpha}_{C-1})$. In the c-th subpopulation, each of the K response categories (except the K-th category) possesses a vector of regression coefficients, denoted as $\boldsymbol{\beta}_{k|c}$. They are collectively represented by the matrix $\mathbf{B}_c := (\boldsymbol{\beta}_{1|c} \quad \ldots \quad \boldsymbol{\beta}_{K-1|c})$.

In HMLRM, the observed layer coefficient matrix $\mathbf{B}$ is a random variable following a discrete distribution taking one of the C possible values in its support $\{\mathbf{B}_1, \ldots, \mathbf{B}_C\}$. This distribution is determined by the hidden layer probabilities $\pi_{c,i}$'s.

The value of $\pi_{c,i}$ depends on the predictor vector $\boldsymbol{X}^{(h)}$. This extension allows us to individualize the hidden group membership for each observation. Note that possible overlapping between the predictor vectors $\boldsymbol{X}^{(h)}$ and $\boldsymbol{X}^{(o)}$ is allowed in HMLRM. In the extreme case, they can be identical. We will use $\mathbf{X}^{(h)} = (\boldsymbol{x}_1^{(h)} \quad \ldots \quad \boldsymbol{x}_n^{(h)})^\top$ and $\mathbf{X}^{(o)} = (\boldsymbol{x}_1^{(o)} \quad \ldots \quad \boldsymbol{x}_n^{(o)})^\top$ to denote the data matrices for the hidden and observed layers' predictor variables respectively[12].

Intuitively speaking, HMLRM (4.5) says that the probability of observing a positive response is according to one of the C logit curves. Which logit curve is the right one depends on the vector of the hidden covariates $\mathbf{x}^{(h)}$ through the hidden probabilities in $\boldsymbol{\pi}(\mathbf{A}, \mathbf{x}^{(h)})$.

We can also make a sense of HMLRM from the perspective of data generation. Let $\mathbf{x}_i$'s be i.i.d. observations. For the i-th observation, its hidden group membership is decided according to the distribution MBern($\boldsymbol{\pi}$). Subsequently, the corresponding observed response y_i is generated from a Binomial distribution with parameters m_c and $\boldsymbol{p}_c$. This is the exact scheme we will be following to sample data in our simulation study.

[12]As a convention in this chapter, a capital letter in italic (e.g. X or $\boldsymbol{X}$) denotes a random variable or a random vector. A capital letter in roman (not italic, e.g. $\mathbf{X}$) denotes a matrix.

4.1.3 Estimation using EM Algorithm

When the number of hidden groups C is known, we can use the EM algorithm [62] to obtain maximum likelihood estimates for the parameters in HMLRM (4.5). If the value of C is unknown, we need to pick a value by either relying on some prior knowledge (e.g. domain specific knowledge) or a data driven approach (such as cross validation). In this section, we assume C is known and discuss the details of the estimation method.

If the values of the hidden variables $\{Z_i\}_i$ were observable, we have the following joint probability

$$P(Y_i = \mathbf{y}_i, Z_i = z_i|\boldsymbol{\theta}) = P(Z_i = z_i|\boldsymbol{\pi})P(Y_i = \mathbf{y}_i|\boldsymbol{p}_{z_i}).$$

Let $\mathbf{z}$ be the "observed" vector of hidden memberships and $\mathbf{Y}$ be the observed response matrix. Then the log-likelihood of the dataset consisting of $\mathbf{Y}$ and $\mathbf{z}$ is written as

$$\begin{aligned}
\boldsymbol{\ell}(\boldsymbol{\theta}|\mathbf{Y}, \mathbf{z}) &= \ln\{\Pi_{i=1}^n P(Y_i = \mathbf{y}_i, Z_i = z_i|\boldsymbol{\theta})\} \\
&= \ln\left\{\Pi_{i=1}^n P(Z_i = z_i|\boldsymbol{\pi}) \cdot P(Y_i = \mathbf{y}_i|\boldsymbol{p}_{z_i})\right\} \\
&= \sum_{i=1}^n \ln P(Z_i = z_i|\boldsymbol{\pi}) + \sum_{i=1}^n \ln P(Y_i = \mathbf{y}_i|\boldsymbol{p}_{z_i}) \\
&= \sum_{i=1}^n \ln \pi_{z_i} + \sum_{i=1}^n \ln p_{y_i|z_i}. \qquad (4.8)
\end{aligned}$$

By using the model specified by (4.5) - (4.7) and the observations for the predictor variables $\mathbf{X}^{(h)}$ and $\mathbf{X}^{(o)}$, the complete data log-likelihood can be written as

$$\boldsymbol{\ell}\left(\boldsymbol{\theta}|\mathbf{X}^{(h)},\mathbf{X}^{(o)},\mathbf{Y},\mathbf{z}\right)=\sum_{i=1}^{n}\ln\pi_{z_i}+\sum_{i=1}^{n}\ln p_{y_i|z_i}$$

where

$$\pi_{z_i}=\begin{cases}\frac{\exp\{\boldsymbol{\beta}_{z_i}^{\top}\mathbf{x}_i^{(h)}\}}{1+\sum_{c=1}^{C-1}\exp\{\boldsymbol{\beta}_c^{\top}\mathbf{x}_i^{(h)}\}} & \text{if } z_i\neq C\\ \frac{1}{1+\sum_{c=1}^{C-1}\exp\{\boldsymbol{\beta}_c^{\top}\mathbf{x}_i^{(h)}\}} & \text{if } z_i=C\end{cases}, \tag{4.9}$$

$$p_{y_i|z_i}=\begin{cases}\frac{\exp\{\boldsymbol{\beta}_{y_i|z_i}^{\top}\mathbf{x}_i^{(o)}\}}{1+\sum_{k=1}^{K-1}\exp\{\boldsymbol{\beta}_{k|c}^{\top}\mathbf{x}_i^{(o)}\}} & \text{if } y_i\neq K\\ \frac{1}{1+\sum_{k=1}^{K-1}\exp\{\boldsymbol{\beta}_{k|c}^{\top}\mathbf{x}_i^{(o)}\}} & \text{if } y_i=K\end{cases}.$$

In practice the values of z_i's are never observed. In order to evaluate the complete likelihood function (4.9) from observed data, we take the expectation w.r.t. the conditional distribution of the hidden membership variable, i.e. $P(Z|Y,\boldsymbol{\theta}^{(t)})$, where $\boldsymbol{\theta}^{(t)}$ denotes the some parameter estimates. In EM algorithm, it is the parameter estimates from the iteration t. We denote the resulting expected complete data log-likelihood as $\mathcal{Q}$ whose expression is given below.

$$\begin{aligned}\mathcal{Q}(\boldsymbol{\theta};\boldsymbol{\theta}^{(t)})&=\mathbb{E}_{\boldsymbol{Z}|Y,\boldsymbol{\theta}^{(t)}}\left[\boldsymbol{\ell}\left(\boldsymbol{\theta}|\mathbf{Y},\mathbf{X},\boldsymbol{Z}\right)\right]\\&=\sum_{c=1}^{C}\sum_{i=1}^{n}\left\{P(Z_i=c|\mathbf{y}_i,\boldsymbol{\theta}^{(t)})\cdot\left[\ln\pi_{c,i}+\ln p_{y_i|c}\right]\right\}\\&=\sum_{c=1}^{C}\sum_{i=1}^{n}\left\{P(Z_i=c|\mathbf{y}_i,\boldsymbol{\theta}^{(t)})\cdot\ln\pi_{c,i}\right\}+\sum_{c=1}^{C}\sum_{i=1}^{n}\left\{P(Z_i=c|\mathbf{y}_i,\boldsymbol{\theta}^{(t)})\cdot\ln p_{y_i|c}\right\}\\&=\sum_{c=1}^{C}\sum_{i=1}^{n}\left(q_{c,i}^{(t)}\cdot\ln\pi_{c,i}\right)+\sum_{c=1}^{C}\sum_{i=1}^{n}\left(q_{c,i}^{(t)}\cdot\ln p_{y_i|c}\right)\end{aligned} \tag{4.10}$$

where we define the true notation for the weights $q_{c,i}^{(t)}:=P(Z_i=c|y_i,\boldsymbol{\theta}^{(t)})$ which is the posterior probability that an observation i belongs to the c-th hidden group given its observed category y_i. Note that its value depends on the existing parameter

estimates and thus is a constant. To evaluate $q_{c,i}^{(t)}$, we apply the Bayes rule for each observation as

$$q_{c,i}^{(t)} = \frac{\pi_{c,i}^{(t)} \cdot p_{y_i|c}^{(t)}}{\sum_{c=1}^{C} \left(\pi_{c,i}^{(t)} \cdot p_{y_i|c}^{(t)} \right)}, \tag{4.11}$$

which also guarantees $\sum_{c=1}^{C} q_{c,i}^{(t)} = 1$. Now we can evaluate $q_{c,i}^{(t)}$ for $i = 1, \ldots, n$ immediately after each EM update.

After the $\mathcal{Q}$-function (4.10) is constructed, it is maximized w.r.t. $\boldsymbol{\theta}$ to update the parameter estimates. It is worth pointing out that the probabilities, $\pi_{c,i}$'s and $p_{y_i|c}$'s, in Equation (4.10) belong to two separate additive components. We will take advantage of this structure to improve the estimation efficiency by optimizing each component separately.

To this end, we can summarize the EM algorithm for estimating HMLRM as follows. Given the values of the parameter estimates from the previous iteration, denote as $\boldsymbol{\alpha}_c^{(t)}$ for $c = 1, \ldots, C-1$ and $\boldsymbol{\beta}_{k|c}^{(t)}$ for $k = 1, \ldots, K-1$ and $c = 1, \ldots, C$, solve the following two steps iteratively:

- **E-step** Evaluate the posterior probabilities $q_{c,i}^{(t)}$ in Equation (4.11) for all observations $i \in \{1, \ldots, n\}$. Then, construct the $\mathcal{Q}$-function (4.10).

- **M-step** Update the parameter estimates by solving the following maximization problem:

$$\left\{ \mathbf{A}^{(t+1)}, \mathbf{B}_1^{(t+1)}, \ldots, \mathbf{B}_C^{(t+1)} \right\} = \underset{\mathbf{A}, \mathbf{B}_1, \ldots, \mathbf{B}_C}{\arg\max}\, \mathcal{Q}(\boldsymbol{\theta}; \boldsymbol{\theta}^{(t)}) \tag{4.12}$$

Remark The hidden layer parameter matrix $\mathbf{A}$ and the observed parameter matrices $\mathbf{B}_c$ for $c = 1, \ldots, C$ belong to two different additive terms. So the optimization problem (4.12) can be simplified. This fact will be exploited in the following derivation of its implementation.

4.1.4 Implementation

In this section, we discuss the implementation of the EM algorithm to estimate HMLRM. The algorithm will be presented after the discussion. The algorithm discussed in this section is suitable for small to medium sized data set. When the sample size is large, a parallelized version of the algorithm is more efficient. The discussion of parallelizing the estimation for HMLRM is deferred to the next section.

Separability of Parameters

In Equation (4.10), we noticed that the hidden layer probabilities $\pi_{c,i}$'s and observed layer probabilities $p_{k,i|c}$'s belong to two different additive components. Consequently, the estimation of hidden layer parameter matrix $\mathbf{A}$ and observed layer parameter matrices $\mathbf{B}_c$'s can be separated. This fact can be utilized to simplify the optimization problem (4.12) in the M-step. To simplify the notations, we rewrite Equation (4.10) in the following form

$$\mathcal{Q}(\boldsymbol{\theta};\boldsymbol{\theta}^{(t)}) = \mathcal{Q}^{(h)}(\boldsymbol{\pi};\boldsymbol{\theta}^{(t)}) + \mathcal{Q}^{(o)}(\mathbf{p}_1,\ldots,\mathbf{p}_C;\boldsymbol{\theta}^{(t)}),$$

where

$$\mathcal{Q}^{(h)}(\boldsymbol{\pi};\boldsymbol{\theta}^{(t)}) = \sum_{i=1}^{n}\sum_{c=1}^{C} q_{c,i}^{(t)} \ln \pi_c, \tag{4.13}$$

$$\mathcal{Q}^{(o)}(\mathbf{p}_1,\ldots,\mathbf{p}_C;\boldsymbol{\theta}^{(t)}) = \sum_{i=1}^{n}\sum_{c=1}^{C} q_{c,i}^{(t)} \ln p_{y_i|c} \tag{4.14}$$

Now instead of optimizing (4.12), we can optimize (4.13) and (4.14) separately, which should be easier since each problem searches solution in a lower dimensional space.

In order to proceed, we use the following assumption HMLRM to simplify the subsequent analysis.

Assumption 4.1.1 *In our hierarchical mixed logistic regression models, the response variable Y follows* $\mathrm{MBern}(p_1,\ldots,p_K)$ *distribution. That means that an observation of*

Y is the result of a single trial which indicates its membership in one of the K allowed categories. The probability that it belongs to category k equals p_k for $k = 1, \ldots, K$.

From now on, each observation of the response variable Y is denoted as a binary vector $\mathbf{y}_i$ which is of size K and has zeros for all but one of the numbers. For example, if there are 10 possible categories and one of Y's observation results in the second category, then we use the notation $\mathbf{y}_i = (0 \quad 1 \quad 0 \quad \ldots \quad 0)^\top$ as a vector of length 10 where all of the numbers are zeros except the second one.

Because of Assumption 4.1.1, the probabilities in the second additive component (4.14) can be written by explicitly incorporating the logistic models for π_c and $p_{k|c}$ as

$$p_{y_i|c} = \mathbf{p}_c^\top \mathbf{y}_i = \frac{e^{(\mathbf{B}_c \mathbf{y}_i)^\top \mathbf{x}_i}}{\sum_{k=1}^{K} e^{\boldsymbol{\beta}_{k|c}^\top \mathbf{x}_i}}. \tag{4.15}$$

It is constructed as the column bind of the multi-class logistic regression coefficients conditioned on the c-th hidden group. Recall that we have chosen to use $\boldsymbol{\beta}_{K|c} \equiv \mathbf{0}$ for all $c = 1, \ldots, C$ to ensure identifiability of the model. Consequently, the second additive component (4.14) can be further written in the following nicer form

$$\mathcal{Q}^{(o)}(\mathbf{p}_1, \ldots, \mathbf{p}_C; \boldsymbol{\theta}^{(t)}) = \sum_{c=1}^{C} \sum_{i=1}^{n} q_{c,i}^{(t)} \ln(\mathbf{p}_c^\top \mathbf{y}_i). \tag{4.16}$$

If we look at (4.15) and (4.16) closely, we can find that each parameter matrix $\mathbf{B}_c$ belongs to a different summand where is summation is over $c = 1, \ldots, C$. So we can further decompose $\mathcal{Q}^{(o)}$ as

$$\mathcal{Q}^{(o)}(\mathbf{p}_1, \ldots, \mathbf{p}_C; \boldsymbol{\theta}^{(t)}) = \sum_{c=1}^{C} \mathcal{Q}_c^{(o)} \left(\mathbf{B}_c; \boldsymbol{\theta}^{(t)} \right) \tag{4.17}$$

where

$$\mathcal{Q}_c^{(o)} \left(\mathbf{B}_c; \boldsymbol{\theta}^{(t)} \right) = \sum_{i=1}^{n} q_{c,i}^{(t)} \ln(\mathbf{p}_c^\top \mathbf{y}_i) = \sum_{i=1}^{n} q_{c,i}^{(t)} \frac{e^{(\mathbf{B}_c \mathbf{y}_i)^\top \mathbf{x}_i}}{\sum_{k=1}^{K} e^{\boldsymbol{\beta}_{k|c}^\top \mathbf{x}_i}}. \tag{4.18}$$

Thus maximizing $\mathcal{Q}^{(o)}$ is equivalent to maximizing each component $\mathcal{Q}_c^{(o)}$ individually. That is,

$$\widehat{\mathbf{B}}_c = \arg\max_{\mathbf{B}_c} \mathcal{Q}_c^{(o)} \left(\mathbf{B}_c; \boldsymbol{\theta}^{(t)} \right).$$

Note that this maximization problem is almost the same as that for a multi-class logistic regression model described in [63], which can be solved using Newton-Raphson method. The only difference is that for a regular multi-class logistic regression, the place of $q_{c,i}^{(t)}$ is occupied by either 0 or 1 indicating the membership of the observed response.

By taking advantage of the separability of the parameter space, we can update the original EM algorithm as follows.

- **E-step** Evaluate the posterior probabilities $q_{c,i}^{(t)}$ in Equation (4.11) for all observations $i \in \{1, \ldots, n\}$.

- **M-step** Update the estimates of the parameters by solving the following maximization problems separately:

$$\widehat{\mathbf{A}}^{(t+1)} = \arg\max_{\mathbf{A}} \mathcal{Q}^{(h)}(\mathbf{A}; \boldsymbol{\theta}^{(t)}), \tag{4.19}$$

$$\widehat{\mathbf{B}}_c^{(t+1)} = \arg\max_{\mathbf{B}_c} \mathcal{Q}_c^{(o)}\left(\mathbf{B}_c; \boldsymbol{\theta}^{(t)}\right), \qquad \text{for } c = 1, \ldots, C. \tag{4.20}$$

Newton Method for Optimization

Each maximization subproblems in (4.19) and (4.20) can be solved by using Newton's method (see [64]). In order to improve the computation efficiency, we derive the explicit formulae for the gradient and Hessian of each target function. The general update rule in the Newton's method is as follows[13]

$$x^{(\tau+1)} = x^{(\tau)} - \mathbf{H}^{-1}(x^{(\tau)})\mathbf{g}(x^{(\tau)}).$$

We would like to obtain the explicit form of $\mathbf{g}$ and $\mathbf{H}$ for each optimization subproblem.

[13]We intentionally use τ to denote the index of an iteration in the application of Newton's method for solving a logistic regression subproblem. Each logistic regression subproblem is a component of the t-th iteration of the EM algorithm.

In the hidden layer, each hidden membership probability $\pi_{c,i}$ depends on the parameter vector $\boldsymbol{\alpha}_c$ through the softmax function. The gradient and Hessian of $\mathcal{Q}^{(h)}(\mathbf{A};\boldsymbol{\theta}^{(t)})$ w.r.t. $\boldsymbol{\alpha}_c$ are given below

$$\mathbf{g}_c^{(h)} = \nabla_{\boldsymbol{\alpha}_c}\mathcal{Q}^{(h)} = \sum_{i=1}^{n}\left(q_{c,i}^{(t)} - \pi_{c,i}\right)\mathbf{x}_i^{(h)} \quad \Rightarrow \quad \mathbb{g}^{(h)} = \begin{bmatrix} \mathbf{g}_1^{(h)} \\ \mathbf{g}_2^{(h)} \\ \vdots \\ \mathbf{g}_{C-1}^{(h)} \end{bmatrix}, \tag{4.21}$$

$$\mathbf{H}_{cc'}^{(h)} = \nabla_{\boldsymbol{\beta}_{c'}}\mathbf{g}_c^{(h)} = -\sum_{i=1}^{n}\pi_{c,i}\left(1_{\{c'=c\}} - \pi_{c',i}\right)\mathbf{x}_i^{(h)}\mathbf{x}_i^{(h)\top}$$

$$\Rightarrow \mathbb{H}^{(h)} = \begin{bmatrix} \mathbf{H}_{11}^{(h)} & \mathbf{H}_{12}^{(h)} & \cdots & \mathbf{H}_{1(C-1)}^{(h)} \\ \mathbf{H}_{21}^{(h)} & \mathbf{H}_{22}^{(h)} & \cdots & \mathbf{H}_{2(C-1)}^{(h)} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{H}_{(C-1)1}^{(h)} & \mathbf{H}_{(C-1)2}^{(h)} & \cdots & \mathbf{H}_{(C-1)(C-1)}^{(h)} \end{bmatrix}. \tag{4.22}$$

In order to arrange the values of the gradient in a meaningful way, we choose to vertically stack together all components (one for each hidden group) to create the gradient vector, denoted as $\mathbb{g}^{(h)}$. The length of the gradient vector is $(C-1)\cdot(d^{(h)}+1)$.

Proof We first derive the gradient for the following function

$$\mathcal{Q}_{\pi}(\boldsymbol{\pi};\boldsymbol{\theta}^{(t)}) = \sum_{i=1}^{n}\sum_{c=1}^{C} q_{c,i}^{(t)} \ln \pi_{c,i},$$

where

$$\pi_{c,i} = \begin{cases} \frac{\exp\{\boldsymbol{\beta}_c^{\top}\mathbf{x}_i^{(h)}\}}{1+\sum_{c=1}^{C-1}\exp\{\boldsymbol{\beta}_c^{\top}\mathbf{x}_i^{(h)}\}} & \text{if } c \neq 1 \\ \frac{1}{1+\sum_{c=1}^{C-1}\exp\{\boldsymbol{\beta}_c^{\top}\mathbf{x}_i^{(h)}\}} & \text{if } c = C \end{cases}.$$

Since we have

$$\frac{\partial \mathcal{Q}_{\pi}}{\partial \pi_{c,i}} = \frac{q_{c,i}^{(t)}}{\pi_{c,i}},$$

$$\frac{\partial \pi_{c,i}}{\partial \boldsymbol{\alpha}_{c'}} = \begin{cases} \pi_{c,i}(1-\pi_{c,i})\mathbf{x}_i^{(h)} & \text{if } c' = c \\ -\pi_{c,i}\pi_{c',i}\mathbf{x}_i^{(h)} & \text{if } c' \neq c \end{cases},$$

then by chain rule we have

$$\begin{aligned}
\frac{\partial \mathcal{Q}_\pi}{\partial \boldsymbol{\alpha}_{c'}} &= \sum_{i=1}^{n} \left(\frac{q_{c',i}^{(t)}}{\pi_{c',i}} \pi_{c',i}(1-\pi_{c',i}) \mathbf{x}_i^{(h)} \right) - \sum_{c \neq c'} \sum_{i=1}^{n} \left(\frac{q_{c,i}^{(t)}}{\pi_{c,i}} \pi_{c,i} \pi_{c',i} \mathbf{x}_i^{(h)} \right) \\
&= \sum_{i=1}^{n} \left(q_{c',i}^{(t)} (1-\pi_{c',i}) \mathbf{x}_i^{(h)} \right) - \sum_{c \neq c'} \sum_{i=1}^{n} \left(q_{c,i}^{(t)} \pi_{c',i} \mathbf{x}_i^{(h)} \right) \\
&= \sum_{i=1}^{n} \left(q_{c',i}^{(t)} (1-\pi_{c',i}) \mathbf{x}_i^{(h)} - \sum_{c \neq c'} q_{c,i}^{(t)} \pi_{c',i} \mathbf{x}_i^{(h)} \right) \\
&= \sum_{i=1}^{n} \left[q_{c',i}^{(t)} (1-\pi_{c',i}) - \left(\sum_{c \neq c'} q_{c,i}^{(t)} \right) \pi_{c',i} \right] \mathbf{x}_i^{(h)} \\
&= \sum_{i=1}^{n} \left(q_{c',i}^{(t)} - \pi_{c',i} \right) \mathbf{x}_i^{(h)},
\end{aligned}$$

and the Hessian

$$\begin{aligned}
\frac{\partial}{\partial \boldsymbol{\alpha}_{c'}} \left(\frac{\partial \mathcal{Q}_\pi}{\partial \boldsymbol{\alpha}_c} \right) &= \frac{\partial}{\partial \boldsymbol{\alpha}_{c'}} \sum_{i=1}^{n} \left(q_{c,i}^{(t)} - \pi_{c,i} \right) \mathbf{x}_i^{(h)} \\
&= -\sum_{i=1}^{n} \left(\frac{\partial \pi_{c,i}}{\partial \boldsymbol{\alpha}_{c'}} \right) \mathbf{x}_i^{(h)\top} \\
&= -\sum_{i=1}^{n} \pi_{c,i} \left(1_{c'=c} - \pi_{c,i} \right) \mathbf{x}_i^{(h)} \mathbf{x}_i^{(h)\top}.
\end{aligned}$$

■

In the observed layer, there are C maximization subproblems (one for each hidden group) in each M-step. Each subproblem requires the gradient and the Hessian for carrying out Newton update. They can be derived in a similar way as those for the

hidden layer. Conditional on a given hidden group $c \in \{1, \ldots, C\}$, the gradient and the Hessian of $\mathcal{Q}_c^{(o)}\left(\mathbf{B}_c; \boldsymbol{\theta}^{(t)}\right)$ w.r.t. the parameters in $\mathbf{B}_c$ are

$$\mathbf{g}_{k|c}^{(o)} = \sum_{i=1}^{n} q_{c,i}^{(t)} \left((\mathbf{y}_i)_k - p_{k,i|c}\right) \mathbf{x}_i^{(o)} \quad \Rightarrow \quad \mathbb{g}_c^{(o)} = \begin{bmatrix} \mathbf{g}_{1|c}^{(h)} \\ \mathbf{g}_{2|c}^{(h)} \\ \vdots \\ \mathbf{g}_{K-1|c}^{(h)} \end{bmatrix}, \tag{4.23}$$

$$\mathbf{H}_{kk'|c}^{(o)} = -\sum_{i=1}^{n} q_{c,i}^{(t)} p_{k,i|c} \left(1_{\{k'=k\}} - p_{k,i|c}\right) \mathbf{x}_i^{(o)} \mathbf{x}_i^{(o)\top}$$

$$\Rightarrow \mathbb{H}_c^{(o)} = \begin{bmatrix} \mathbf{H}_{11|c}^{(o)} & \mathbf{H}_{12|c}^{(o)} & \cdots & \mathbf{H}_{1(K-1)|c}^{(o)} \\ \mathbf{H}_{21|c}^{(o)} & \mathbf{H}_{22|c}^{(o)} & \cdots & \mathbf{H}_{2(K-1)|c}^{(o)} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{H}_{(K-1)1|c}^{(o)} & \mathbf{H}_{(K-1)2|c}^{(o)} & \cdots & \mathbf{H}_{(K-1)(K-1)|c}^{(o)} \end{bmatrix}. \tag{4.24}$$

Proof Now, we derive the gradients and Hessian for the observed layer optimization target.

$$\mathcal{Q}_{\mathbf{B}_c}\left(\mathbf{B}_c; \boldsymbol{\theta}^{(t)}\right) = \sum_{i=1}^{n} q_{c,i}^{(t)} \ln(\mathbf{p}_{c,i}^{\top} \mathbf{y}_i),$$

where

$$\mathbf{p}_{c,i}^{\top} \mathbf{y}_i = \frac{e^{(\mathbf{B}_c \mathbf{y}_i)^{\top} \mathbf{x}_i}}{\sum_{k=0}^{K} e^{\boldsymbol{\beta}_{k|c}^{\top} \mathbf{x}_i}}.$$

Let $\mathcal{Q}_{\mathbf{B}_c}^{(i)} := q_{c,i}^{(t)} \ln(\mathbf{p}_c^{\top} \mathbf{y}_i)$ and $p_{y_i|c} := \mathbf{p}_{c,i}^{\top} \mathbf{y}_i$. We have

$$\frac{\partial \mathcal{Q}_{\mathbf{B}_c}^{(i)}}{\partial p_{y_i|c}} = \frac{q_{c,i}^{(t)}}{p_{y_i|c}},$$

$$\frac{\partial p_{y_i|c}}{\partial \boldsymbol{\beta}_{k|c}} = \begin{cases} p_{k,i|c}(1 - p_{k,i|c}) \mathbf{x}_i^{(o)} & \text{if } (\mathbf{y}_i)_k = 1 \\ -p_{k,i|c} p_{y_i|c} \mathbf{x}_i^{(o)} & \text{if } (\mathbf{y}_i)_k = 0 \end{cases}.$$

Then by chain rule, we have

$$\begin{aligned}\frac{\partial \mathcal{Q}_{\mathbf{B}_c}}{\partial \boldsymbol{\beta}_{k|c}} &= \sum_{i=1}^{n} \frac{\partial \mathcal{Q}^{(i)}_{\mathbf{B}_c}}{\partial \boldsymbol{\beta}_{k|c}} \\ &= \sum_{i=1}^{n} \left\{ q^{(t)}_{c,i} \left[1_{\{(\mathbf{y}_i)_k=1\}} \cdot (1 - p_{k,i|c}) - 1_{\{(\mathbf{y}_i)_k\neq 1\}} \cdot p_{k,i|c} \right] \right\} \mathbf{x}_i^{(o)} \\ &= \sum_{i=1}^{n} q^{(t)}_{c,i} \left(1_{\{(\mathbf{y}_i)_k=1\}} - p_{k,i|c} \right) \mathbf{x}_i^{(o)},\end{aligned}$$

and the Hessian

$$\begin{aligned}\frac{\partial}{\partial \boldsymbol{\beta}_{k'|c}} \left(\frac{\partial \mathcal{Q}_{\mathbf{B}_c}}{\partial \boldsymbol{\beta}_{k|c}} \right) &= \frac{\partial}{\partial \boldsymbol{\beta}_{k'|c}} \sum_{i=1}^{n} q^{(t)}_{c,i} \left(1_{\{(\mathbf{y}_i)_k=1\}} - p_{k,i|c} \right) \mathbf{x}_i^{(o)\top} \\ &= \sum_{i=1}^{n} q^{(t)}_{c,i} \left(-\frac{\partial p_{k,i|c}}{\partial \boldsymbol{\beta}_{k',i|c}} \right) \mathbf{x}_i^{(o)\top} \\ &= -\sum_{i=1}^{n} q^{(t)}_{c,i} p_{k,i|c} \left(1_{\{k'=k\}} - p_{k,i|c} \right) \mathbf{x}_i^{(o)} \mathbf{x}_i^{(o)\top}\end{aligned}$$

■

In the above gradient, $(\mathbf{y}_i)_k$ is the k-th number in the observed response vector $\mathbf{y}_i$. Since $\mathbf{y}_i$ is a binary vector, the value of $(\mathbf{y}_i)_k$ is either 1 or 0 depending on if the observed response belongs to the k-th category. We also follow the practice in the hidden layer to stack the components together to build the gradient vector $\mathbb{g}_c^{(o)}$ and the Hessian matrix $\mathbb{H}_c^{(o)}$ for the subproblem.

So for each subproblem component (in both hidden and observed layers) in the M-step, the update rules used in Newton's method for solving (4.19) and (4.20) are given by

$$\mathbb{\alpha}_c^{(\tau+1)} = \mathbb{\alpha}^{(\tau)} - \left[\mathbb{H}^{(h)} \right]^{-1} \mathbb{g}^{(h)}, \tag{4.25}$$

$$\mathbb{\beta}_c^{(\tau+1)} = \mathbb{\beta}_c^{(\tau)} - \left[\mathbb{H}_c^{(o)} \right]^{-1} \mathbb{g}_c^{(o)}, \quad \text{for } c = 1, \ldots, C. \tag{4.26}$$

Usual optimization packages minimize a target function by default. So in order to use those packages to solve our problem, we have to transform the maximization problems into minimization by negate the target functions. Correspondingly, the gradients and Hessians will also be negated.

Matrix Representation

In this section we will describe the estimation for HMLRM with Newton's method in pure matrix notations. In particular, we will arrange the data in matrices which will be suitable to evaluate the gradients and Hessians. This approach is suitable for small to medium sized data sets[14]. With the estimation implemented using matrix, vectorization will be used for efficient computation. This is especially useful when interpreted languages, such as Python and R, are used for data analysis.

Recall that under Assumption 4.1.1 an observed response $\mathbf{y}_i$ is coded as a binary vector of length K where all but one numbers are zeros. Using this coding scheme, all of the response observations can be combined in a n-by-K matrix, which is denoted as $\mathbf{Y}$. This matrix is shown on the LHS in (4.27)[15]. Each row of $\mathbf{Y}$ is an observed response vector. Each column of $\mathbf{Y}$ is corresponding to one category and each number in the column indicates if the observation falls into this category. The construction of the matrix $\mathbf{Y}$ is redundant because using $K-1$ columns is enough to preserve all the information. In this regard, we can choose to drop the first column. By

[14]The definition of data size varies under different considerations. It depends on the factors such as the measurement of size (e.g. bytes, number of records), the stage of data analysis (e.g. raw data for ETL, post-processed data for model fitting), and area of study (this affects the structure of data). Here, we use the number of records as the measure of size, and consider a data set with number of records in the order of hundreds of thousands as being medium sized.

[15]The numbers in the matrix $\mathbf{Y}$ are made up solely for the illustration purpose.

vertically stacking all the remaining columns, we obtain a long column vector $\mathbf{y}$ of length $(K-1)n$, which is shown on the RHS in (4.27).

$$\mathbf{Y} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 & \cdots & 0 \\ 1 & 0 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 1 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \vdots & & \vdots \end{bmatrix} \quad \Rightarrow \quad \mathrm{y} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ \vdots \\ \hline 1 \\ 0 \\ 0 \\ 0 \\ \vdots \\ \hline 0 \\ 0 \\ 0 \\ 1 \\ \vdots \\ \hline \vdots \\ \hline 0 \\ 0 \\ 1 \\ 0 \\ \vdots \\ \hline \vdots \end{bmatrix}. \tag{4.27}$$

The original matrices of the predictor variables (for both hidden and observed layers) are shown on the LHS in (4.28) and (4.29), where each row is an observed vector of the predictor variables. We added ones as the first column in both matrices to accommodate the intercepts. Then, we make $C-1$ copies of $\mathbf{X}^{(h)}$ and $K-1$ copies

of $\mathbf{X}^{(o)}$ which are stacked diagonally to construct the block matrix shown on the RHS in (4.28) and (4.29).

$$\mathbf{X}^{(h)} = \begin{bmatrix} 1 & x_{11}^{(h)} & x_{12}^{(h)} & \cdots & x_{1d^{(h)}}^{(h)} \\ 1 & x_{21}^{(h)} & x_{22}^{(h)} & \cdots & x_{2d^{(h)}}^{(h)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{i1}^{(h)} & x_{i2}^{(h)} & \cdots & x_{id^{(h)}}^{(h)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{n1}^{(h)} & x_{n2}^{(h)} & \cdots & x_{nd^{(h)}}^{(h)} \end{bmatrix} \quad \Rightarrow \quad \mathbb{X}^{(h)} = \underbrace{\begin{bmatrix} \mathbf{X}^{(h)} & & & \\ & \mathbf{X}^{(h)} & & \\ & & \ddots & \\ & & & \mathbf{X}^{(h)} \end{bmatrix}}_{C-1 \text{ copies}} \tag{4.28}$$

$$\mathbf{X}^{(o)} = \begin{bmatrix} 1 & x_{11}^{(o)} & x_{12}^{(o)} & \cdots & x_{1d^{(o)}}^{(o)} \\ 1 & x_{21}^{(o)} & x_{22}^{(o)} & \cdots & x_{2d^{(o)}}^{(o)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{i1}^{(o)} & x_{i2}^{(o)} & \cdots & x_{id^{(o)}}^{(o)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{n1}^{(o)} & x_{n2}^{(o)} & \cdots & x_{nd^{(o)}}^{(o)} \end{bmatrix} \quad \Rightarrow \quad \mathbb{X}^{(o)} = \underbrace{\begin{bmatrix} \mathbf{X}^{(o)} & & & \\ & \mathbf{X}^{(o)} & & \\ & & \ddots & \\ & & & \mathbf{X}^{(o)} \end{bmatrix}}_{K-1 \text{ copies}} \tag{4.29}$$

So far we have heavily relied on the usage of the hidden layer probabilities π_c and observed layer probabilities $p_{k|c}$. Though they are not the true parameters for HMLRM, using them could greatly relieve the pressure to introduce more notations. In this section, we will keep using them to construct related matrices.

There is a vector of C hidden layer probabilities $\{\pi_{c,i}\}_{c=1}^{C}$ corresponding to each observation, one for belonging to each hidden group. So for a dataset with n observations, there are nC such probabilities, which can be arranged into a matrix. This matrix, denoted as $\mathbf{\Pi}$, is shown on the LHS in (4.30). Because of the constraint $\sum_{c=1}^{C} \pi_{c,i} = 1$ for all $i \in \{1, \ldots, n\}$, we can safely remove the first column of $\mathbf{\Pi}$. The

remaining columns can be stacked vertically which results in a long vector $\mathbb{\pi}$ shown on the RHS in (4.30).

$$\mathbf{\Pi} = \begin{bmatrix} \pi_{1,1} & \cdots & \pi_{C-1,1} & \cancel{\pi_{C,1}} \\ \pi_{1,2} & \cdots & \pi_{C-1,2} & \cancel{\pi_{C,2}} \\ \vdots & & \vdots & \vdots \\ \pi_{1,i} & \cdots & \pi_{C-1,i} & \cancel{\pi_{C,i}} \\ \vdots & & \vdots & \vdots \\ \pi_{1,n} & \cdots & \pi_{C-1,n} & \cancel{\pi_{C,n}} \end{bmatrix} \quad \Rightarrow \quad \mathbb{\pi} = \begin{bmatrix} \pi_{1,1} \\ \pi_{1,2} \\ \vdots \\ \underline{\pi_{1,n}} \\ \vdots \\ \vdots \\ \underline{\phantom{\pi_{1,n}}} \\ \pi_{C-1,1} \\ \pi_{C-1,2} \\ \vdots \\ \pi_{C-1,n} \end{bmatrix}. \tag{4.30}$$

If all the observations belong to the c-th hidden group, the observed layer probabilities can be summarized by the matrix $\mathbf{P}_c$ shown in (4.31). Each row in $\mathbf{P}_c$ is corresponding to one observation, which contains the probabilities that it belongs to the respective categories indexed by the columns. Since the probabilities on each row are constrained such that they must add up to one, we could eliminate the first column without any information loss. By vertically stacking all the remaining columns, we obtain the long vector $\mathbb{p}_c$ of length $(K-1)n$, which is shown on the RHS in (4.31).

$$\mathbf{P}_c = \begin{bmatrix} p_{1,1|c} & \cdots & p_{K-1,1|c} & \cancel{p_{K,1|c}} \\ p_{1,2|c} & \cdots & p_{K-1,2|c} & \cancel{p_{K,2|c}} \\ \vdots & & \vdots & \vdots \\ \vdots & & \vdots & \vdots \\ p_{1,n|c} & \cdots & p_{K-1,n|c} & \cancel{p_{K,n|c}} \end{bmatrix} \quad \Rightarrow \quad \mathbb{p}_c = \begin{bmatrix} p_{2,1|c} \\ p_{2,2|c} \\ \vdots \\ \underline{p_{2,n|c}} \\ \vdots \\ \vdots \\ \underline{\phantom{p_{2,n|c}}} \\ p_{K-1,1|c} \\ p_{K-1,2|c} \\ \vdots \\ p_{K-1,n|c} \end{bmatrix}. \tag{4.31}$$

Applying Newton's method to solve a logistic regression is equivalent to solving an Iteratively Re-weighted Least Square (IRLS) problem [65]. The weight matrix used in a IRLS iteration is part of the Hessian used in Newton's iteration. For both hidden and observed layers' subproblems, the weight matrices can be constructed by using $\mathbf{\Pi}$ and $\mathbf{P}_c$'s.

For the subproblem in the hidden layer, its weight matrix is constructed by using the estimated hidden membership probabilities $\{\pi_{c,i}\}_{c,i}$. The weight matrix, denoted

by $\mathbb{W}^{(h)}$, is a $(C-1)$-by-$(C-1)$ block matrix, where each block matrix itself is a diagonal matrix. Each block on the diagonal of $\mathbb{W}^{(h)}$ contains the current estimate of the variance $\pi_{c,i}(1-\pi_{c,i})$ for the i-th observation in the c-th ($c \in \{1,\ldots,C-1\}$) hidden group. The off-diagonal block located at row c and column c' of $\mathbb{W}^{(h)}$ contains the negative cross product $-\pi_{c,i}\pi_{c',i}$. The construction is demonstrated in (4.32).

$$\left.\begin{aligned}\mathbf{W}_{cc}^{(h)} &= \begin{bmatrix}\pi_{c,1}(1-\pi_{c,1}) & & & \\ & \pi_{c,2}(1-\pi_{c,2}) & & \\ & & \ddots & \\ & & & \pi_{c,n}(1-\pi_{c,n})\end{bmatrix} \\ \mathbf{W}_{cc'}^{(h)} &= \begin{bmatrix}-\pi_{c,1}\pi_{c',1} & & & \\ & -\pi_{c,2}\pi_{c',2} & & \\ & & \ddots & \\ & & & -\pi_{c,n}\pi_{c',n}\end{bmatrix}\end{aligned}\right\} \Rightarrow \mathbb{W}^{(h)} = \begin{bmatrix}\mathbf{W}_{11}^{(h)} & \mathbf{W}_{12}^{(h)} & \cdots & \mathbf{W}_{1(C-1)}^{(h)} \\ \mathbf{W}_{21}^{(h)} & \mathbf{W}_{22}^{(h)} & \cdots & \mathbf{W}_{2(C-1)}^{(h)} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{W}_{(C-1)1}^{(h)} & \mathbf{W}_{(C-1)2}^{(h)} & \cdots & \mathbf{W}_{(C-1)(C-1)}^{(h)}\end{bmatrix} \tag{4.32}$$

The construction of the weight matrices for the observed layer is similar to that for the hidden layer. Conditional on being in a particular hidden group, there is one pseudo logistic regression subproblem. So in each M-step, there are C different IRLSs (one for each subproblem) to be performed. Because each subproblem maintains its own set of parameters, their respective weight matrices are also different. For the c-th IRLS problem, its weight matrix, denoted as $\mathbb{W}_c^{(o)}$ (where the subscript c indicates the index of the hidden group it conditions on), is a $(K-1)$-by-$(K-1)$ block matrix. Each block $\mathbf{W}_{kk|c}^{(o)}$ on the diagonal of $\mathbb{W}_c^{(o)}$ is a diagonal matrix with $\{p_{k,i|c}(1-p_{k,i|c})\}_{i=1}^n$ as the diagonal elements. An off-diagonal block $\mathbf{W}_{kk'|c}^{(o)}$ is also a diagonal matrix, but

with $\{-p_{k,i|c}p_{k',i|c})\}_{i=1}^{n}$ as its diagonal values. The construction details are illustrated in (4.33).

$$\left.\begin{aligned}\mathbf{W}^{(o)}_{kk|c} &= \begin{bmatrix} p_{k,1|c}(1-p_{k,1|c}) & & & \\ & p_{k,2|c}(1-p_{k,2|c}) & & \\ & & \ddots & \\ & & & p_{k,n|c}(1-p_{k,n|c}) \end{bmatrix} \\ \mathbf{W}^{(o)}_{kk'|c} &= \begin{bmatrix} -p_{k,1|c}p_{k',1|c} & & & \\ & -p_{k,2|c}p_{k',2|c} & & \\ & & \ddots & \\ & & & -p_{k,n}p_{k',n|c} \end{bmatrix}\end{aligned}\right\}$$

$$\Rightarrow \mathbb{W}^{(o)}_{c} = \begin{bmatrix} \mathbf{W}^{(o)}_{11|c} & \mathbf{W}^{(o)}_{12|c} & \cdots & \mathbf{W}^{(o)}_{1(K-1)|c} \\ \mathbf{W}^{(o)}_{21|c} & \mathbf{W}^{(o)}_{22|c} & \cdots & \mathbf{W}^{(o)}_{2(K-1)|c} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{W}^{(o)}_{(K-1)1|c} & \mathbf{W}^{(o)}_{(K-1)2|c} & \cdots & \mathbf{W}^{(o)}_{(K-1)(K-1)|c} \end{bmatrix} \quad (4.33)$$

We have defined the response vector $\mathbb{y}$, the block matrices $\mathbb{X}^{(h)}$ and $\mathbb{X}^{(o)}_{c}$'s for the hidden and observed layer predictor variables, and the weight matrices $\mathbb{W}^{(h)}$ and $\mathbb{W}^{(o)}_{c}$ for the ILRS-type computation in the component subproblems. The last piece that is needed is the vector consisting of the conditional membership probabilities $q_{c,i}$'s. This is the major difference between a usual multi-class logistic regression and a logistic-regression-like subproblem in HMLRM.

The values of $\{q_{c,i}\}_{c,i}$ are involved in both the hidden layer and the observed layer of HMLRM. In the hidden layer, they act as the response for the pseudo logistic regression. Each $q_{c,i}$ indicates the probability that the i-th observation belongs to the c-th hidden group based on the information at the current stage of EM iterations. So the difference between $q_{c,i}$ and the response of a real logistic regression is that $q_{c,i}$ is a continuous real number in $(0, 1)$ while a usual logistic response is either 0 or 1. In the observed layer, the value of $q_{c,i}$ quantifies the contribution of the i-th observation to the c-th subproblem. This can be seen from the formula of (4.23). For a given observation with index i, the values of $(\mathbf{y})_k$ and $\mathbf{x}^{(o)}_i$ are the same regardless which hidden group it belongs to. If we further assume $\{p_{k,i|c}\}_{c=1}^{C}$ are also the same across

all the C hidden groups[16], then the higher the value of $q_{c,i}$ the larger the i-th piece of gradient is. This would results in faster update for the c-th subproblem than others.

In order to be able to use $\{q_{c,i}\}_{c,i}$ for updating hidden-layer parameters, we first arrange $q_{c,i}$'s in a n-by-C matrix, which is denoted as $\mathcal{Q}$. In the matrix, each row is corresponding to one observation and each column is corresponding to one hidden group. Since the conditional probabilities on each row must add up to one, we can remove the first column without losing any information. The remaining columns are stacked vertically to produce a long vector $\mathbb{q}$ of length $n(C-1)$. This construction is illustrated in (4.34). The column vector $\mathbb{q}$ will be used as the "response" for the logistic regression subproblem in the hidden layer.

$$\mathcal{Q} = \begin{bmatrix} q_{1,1} & \cdots & q_{C-1,1} & \cancel{q_{C,1}} \\ q_{1,2} & \cdots & q_{C-1,2} & \cancel{q_{C,2}} \\ \vdots & & \vdots & \vdots \\ q_{1,i} & \cdots & q_{C-1,i} & \cancel{q_{C,i}} \\ \vdots & & \vdots & \vdots \\ q_{1,n} & \cdots & q_{C-1,n} & \cancel{q_{C,n}} \end{bmatrix}_{n\times C} \quad \Rightarrow \quad \mathbb{q} = \begin{bmatrix} q_{1,1} \\ q_{1,2} \\ \vdots \\ \underline{q_{1,n}} \\ \vdots \\ \vdots \\ \underline{\phantom{q_{1,n}}} \\ q_{C-1,1} \\ q_{C-1,2} \\ \vdots \\ q_{C-1,n} \end{bmatrix} \tag{4.34}$$

We also create a diagonal matrix for each column in $\mathcal{Q}$. Each diagonal matrix is denoted as $\mathcal{Q}_c$ for $c \in \{1, \ldots, C\}$, which is copied $K-1$ times to be stacked diagonally to create the diagonal block matrix $\mathbb{Q}_c$. The construction is illustrated in (4.35). This block matrix $\mathbb{Q}_c$ together with the matrix $\mathbb{W}_c^{(o)}$ will be used to construct the weight matrix for a subproblem in the observed layer.

$$\mathcal{Q}_c = \begin{bmatrix} q_{c,1} & & & \\ & q_{c,2} & & \\ & & \ddots & \\ & & & q_{c,n} \end{bmatrix} \xrightarrow{\text{copy } K-1 \text{ times}} \mathbb{Q}_c = \begin{bmatrix} \mathcal{Q}_c & & & \\ & \mathcal{Q}_c & & \\ & & \ddots & \\ & & & \mathcal{Q}_c \end{bmatrix}_{n(K-1)\times n(K-1)} \tag{4.35}$$

[16]This assumption is only used to simplify our analysis. It would not hold in reality, otherwise all of those C subproblems can be reduced to one problem.

After all necessary matrices being well defined, we can rewrite the update rules (4.25) and (4.26) for the Newton's method in the form of weight least square regressions. They are given by

$$\mathbb{\alpha}^{(t+1)} = \mathbb{\alpha}^{(t)} - \left(\mathbb{X}^{(h)\top}\mathbb{W}^{(h)}\mathbb{X}^{(h)}\right)^{-1}\mathbb{X}^{(o)\top}\left(\mathbb{q} - \mathbb{r}\right), \tag{4.36}$$

$$\mathbb{\beta}_c^{(t+1)} = \mathbb{\beta}_c^{(t)} - \left(\mathbb{X}^{(o)\top}\mathbb{Q}_c\mathbb{W}_c^{(o)}\mathbb{X}^{(h)}\right)^{-1}\mathbb{X}^{(h)\top}\mathbb{Q}_c\left(\mathbb{y} - \mathbb{p}_c\right). \tag{4.37}$$

By using the above formulation, we can write out the matrix version of our EM algorithm, which is given below.[17]

E-step Use the parameter estimates $\hat{\mathbb{\alpha}}^{(t)}$ and $\hat{\mathbb{\beta}}_c^{(t)}$ from the previous step to construct the vectors $\mathbb{q}^{(t)}$, $\mathbb{r}^{(t)}$ and $\mathbb{p}_c^{(t)}$ and the matrices $\mathbb{W}^{(h)(t)}$, $\mathbb{Q}_c^{(t)}$ and $\mathbb{W}_c^{(o)(t)}$, where $c = 1, \ldots, C$.

M-step Update the estimates of the parameters for each logistic regression subproblems. Each subproblem is solved by using Newton's method. In the hidden layer, using the following update rules.

$$\hat{\mathbb{\alpha}}^{(\tau+1|t)} = \hat{\mathbb{\alpha}}^{(\tau|t)} - \left(\mathbb{X}^{(h)\top}\mathbb{W}^{(h)(\tau+1|t)}\mathbb{X}^{(h)}\right)^{-1}\mathbb{X}^{\top}\left(\mathbb{q}^{(t)} - \mathbb{r}^{(t)}\right).$$

In the observed layer, using the following update rules for solving each subproblem by using Newton's method.

$$\hat{\mathbb{\beta}}_c^{(\tau+1|t)} = \hat{\mathbb{\beta}}_c^{(\tau|t)} - \left(\mathbb{X}^{(o)\top}\mathbb{Q}_c^{(t)}\mathbb{W}_c^{(o)(\tau|t)}\mathbb{X}^{(h)}\right)^{-1}\mathbb{X}^{\top}\left(\mathbb{y} - \mathbb{p}_c^{(\tau|t)}\right), \quad \text{for } c = 1, \ldots, C.$$

Algorithm

Algorithm 4 shows the procedure for estimating the parameters for a HMLRM using EM algorithm. The algorithm is a usual single-threaded implementation.

[17] We use t and τ as superscripts to indicate two levels of iterations. The letter t is used as the index for each EM iteration, while the letter τ is used as the index for each Newton's update for solving a maximization problem in the M-step of the t-th EM iteration. So τ-iterations are nested in each t-iteration. The quantities with the superscript t are obtained directly as the result of each M-step. The quantities with the superscript τ are kept as the intermediate updates in the Newton's iterative procedure.

Algorithm 4 Non-parallel EM for HMLRM

function TRAINHMLRM(*data*, $\mathbf{A}^{(0)}$, $\mathbf{B}_c^{(0)}$'s, ϵ_1, ϵ_2, *maxIter*)
  $param^{(0)} \leftarrow \{\mathbf{A}^{(0)}, \mathbf{B}_c^{(0)}\text{'s}\}$
  $nollk^{(0)} \leftarrow$ calculate negative log-likelihood for observed data using $param^{(0)}$
  $relAbsDiffNollk \leftarrow \infty$
  $relDiffParams \leftarrow \infty$
  $t = 0$
  while $relAbsDiffNollk >= \epsilon$ and $relDiffParams >= \epsilon$ **do**
    $\pi^{(t)} \leftarrow$ compute sigmoid or softmax for a chunk of $\mathbf{X}^{(h)}$ using $\mathbf{A}^{(t)}$
    $p^{(t)} \leftarrow$ compute sigmoid or softamx for a chunk of $\mathbf{X}^{(o)}$ using $\mathbf{B}_c^{(t)}$'s
    $q^{(t)} \leftarrow P(Z|Y)$'s calculated from $\pi^{(t)}$ and $p^{(t)}$
    Define target function $\mathcal{Q}^{(h)}(A, q = q^{(t)})$ (E-step for hidden layer)
    Define gradient $g^{(h)}(A, q = q^{(t)})$ and Hessian $H^{(h)}(A, q = q^{(t)})$
    $\mathbf{A}^{(t+1)} \leftarrow \arg\min_{\mathbf{A}}(-\mathcal{Q}^{(h)})$ (M-step for the hidden layer)
    for each hidden group c **do**
      Define function $\mathcal{Q}_c^{(o)}(\mathbf{B}_c, q = q^{(t)})$ (E-step for the observed layer)
      Define gradient $g_c^{(o)}(A, q = q^{(t)})$ and Hessian $H_c^{(o)}(A, q = q^{(t)})$
      $\mathbf{B}^{(t+1)} \leftarrow \arg\min_{\mathbf{B}_c}(-\mathcal{Q}_c^{(o)})$ (M-step for the observed layer)
    end for
    $param^{(t+1)} \leftarrow \{\mathbf{A}_c^{(t+1)}, \mathbf{B}_c^{(t+1)}\text{'s}\}$
    $relDiffParams \leftarrow \frac{\left\|param^{(t+1)} - param^{(t)}\right\|}{\left\|param^{(t)}\right\|}$
    $nollk^{(t+1)} \leftarrow$ calculate negative log-likelihood for observed data using $param^{(t+1)}$
    $relAbsDiffNollk \leftarrow \frac{\left|nollk^{(t+1)} - nollk^{(t)}\right|}{nollk^{(t)}}$
    $t = t + 1$
    if $t \geq maxIter$ **then**
      break
    end if
  end while
  return ($param^{(t)}$, $nollk^{(t)}$)
end function

4.1.5 HMLRM as a None-linear Model

We can show that using HMLRM results in a non-linear decision boundary. This property of HMLRM greatly improves the classification performance of the model. The nonlinearity of the decision boundary is the result of using the hidden layer which explicitly models subpopulations.

When the response Y is binary and the hidden layer has two groups, the classification rule for HMLRM is given by

$$P(Y=1|X=x) \underset{1}{\overset{0}{\lessgtr}} P(Y=0|X=x). \tag{4.38}$$

The expression of $P(Y=1|X=x)$ are given by

$$\begin{aligned} P(Y=1|X,\boldsymbol{\theta}) &= P(Y=1,Z=1|X=x,\boldsymbol{\theta}) + P(Y=1,Z=2|X=x,\boldsymbol{\theta}) \\ &= P(Y=1|Z=1,X=x,\boldsymbol{\theta}) \cdot P(Z=1|\boldsymbol{\theta}) + \\ &\qquad P(Y=1|Z=2,X=x,\boldsymbol{\theta}) \cdot P(Z=2|\boldsymbol{\theta}) \\ &= \frac{e^{x^T\beta_1}}{1+e^{x^T\beta_1}} \cdot \pi + \frac{e^{x^T\beta_2}}{1+e^{x^T\beta_2}} \cdot (1-\pi). \end{aligned}$$

The expression of $P(Y=0|X=x)$ is given by

$$P(Y=0|X=x,\boldsymbol{\theta}) = \frac{1}{1+e^{x^T\beta_1}} \cdot \pi + \frac{1}{1+e^{x^T\beta_2}} \cdot (1-\pi).$$

So the classification rule (4.38) for HMLRM can be further written as[18]

$$\begin{aligned} & \pi \left(\frac{e^{x^T\beta_1}-1}{1+e^{x^T\beta_1}} \right) \underset{1}{\overset{0}{\lessgtr}} (1-\pi) \left(\frac{1-e^{x^T\beta_2}}{1+e^{x^T\beta_2}} \right) \\ \Rightarrow \quad & \pi \cdot \tanh\left(\frac{x^T\beta_1}{2} \right) \underset{1}{\overset{0}{\lessgtr}} (\pi-1) \cdot \tanh\left(\frac{x^T\beta_2}{2} \right), \end{aligned}$$

which gives the the following classification boundary:

$$\left\{ x \in R^d | f(x) = \pi \cdot \tanh\left(\frac{x^T\beta_1}{2} \right) + (1-\pi) \cdot \tanh\left(\frac{x^T\beta_2}{2} \right) = 0 \right\}. \tag{4.39}$$

[18]Here, we use the hyperbolic tangent function $\tanh(x) = \frac{e^{2x}-1}{e^{2x}+1}$ to simplify the expression.

Let $\text{sigmoid}(x) = \frac{1}{1+e^{-x}}$ be a sigmoid function. Since the hyperbolic tangent function *tanh* is a rescaled logistic sigmoid function, i.e. $\tanh(x) = 2\text{sigmoid}(2x) - 1$, the classification boundary can be further written as

$$\begin{aligned} & f(x) = 2\pi\text{sigmoid}(x^T\beta_1) + 2(1-\pi)\text{sigmoid}(x^T\beta_2) - 1 = 0 \\ \Rightarrow \quad & \pi \cdot \text{sigmoid}(x^T\beta_1) + (1-\pi) \cdot \text{sigmoid}(x^T\beta_2) = \frac{1}{2}. \end{aligned} \tag{4.40}$$

So the decision is constructed as a weighted sigmoid functions where the weights are the probabilities that an observation belongs to each hidden group.

Example 4.1.1 ***(HMLRM with two hidden groups)*** *In this example, we use $C = 2$, $K = 2$, $d^{(h)} = 0$, $d^{(o)} = 2$. This means that the HMLRM has two groups in the hidden layer and two categories for the response variable. We are not modelling the the membership probabilities π_1 and π_2 in the hidden layer and they are constants for all observations. The dimension for the observed layer is fixed to two (that is there are two predictor variables for modeling the Bernoulli probabilities for the response variable).*

In the current setting of HMLRM, the decision boundary, adapted from (4.39), is given by

$$f(x_1, x_2) = \pi \cdot \tanh\left(\frac{\beta_{11}x_1 + \beta_{12}x_2}{2}\right) + (1-\pi) \cdot \tanh\left(\frac{\beta_{21}x_1 + \beta_{22}x_2}{2}\right) = 0,$$

where β_{ij} denotes the regression coefficient for the j-th predictor variable if the observation belongs to the i-th hidden group. We are not using intercepts in the model.

The 3D surface plots of the function $f(x_1, x_2)$ are provided for different values of π. We set the values of the linear coefficients to be relative small so that the probabilities change slowly. This is helpful to investigate the surfaces visually. The plots are shown in Figure 4.1.

When $\pi = 0$, only the first component in $f(x_1, x_2)$ takes effect. So the surface is nothing but a two-dimensional sigmoid function where $\beta_{11} = 3$ and $\beta_{12} = 1$ are the coefficients in the linear exponent. Similarly, when $\pi = 1$, the $f(x_1, x_2)$ reduces to the second sigmoid function with $\beta_{21} = 1$ and $\beta_{22} = 3$ being the linear coefficients.

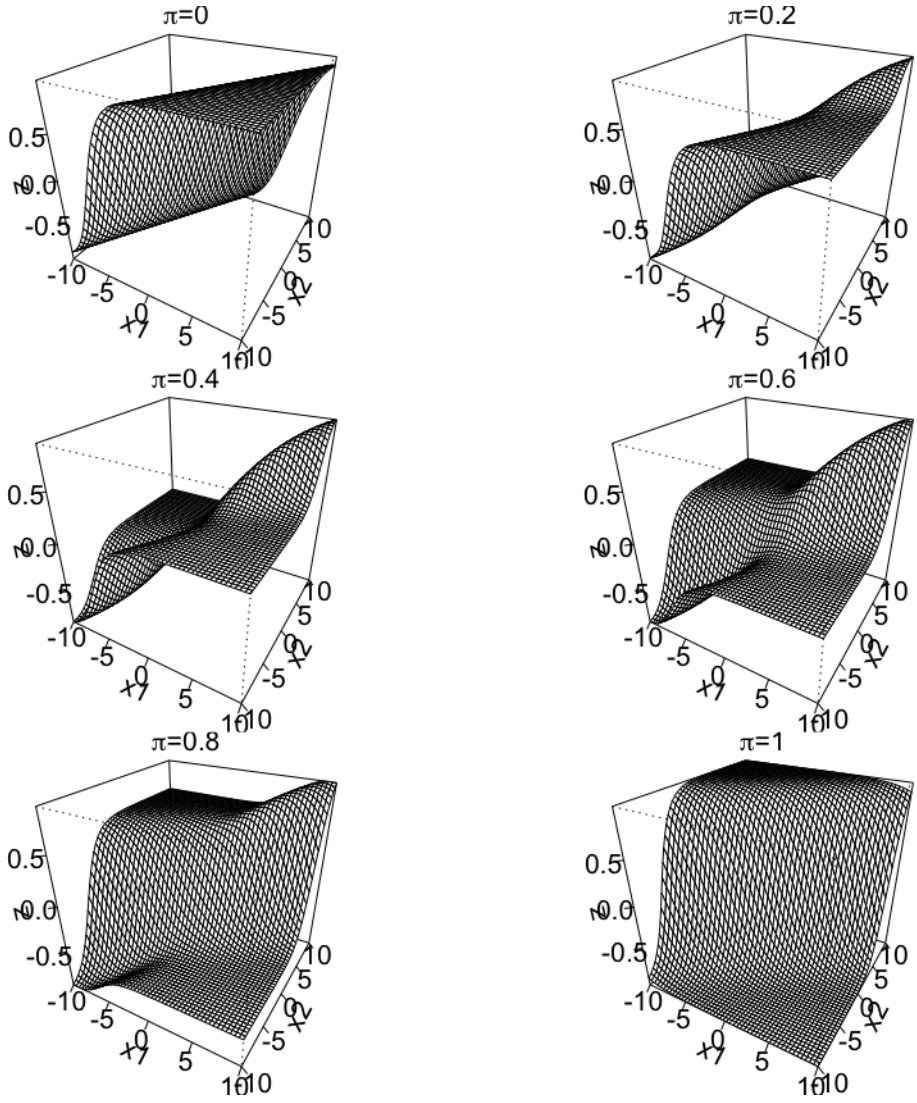

Fig. 4.1. 3D surface plots for the weight tanh function. The coefficients for the first component are $\beta_{11} = 0.8$ and $\beta_{12} = -0.5$. The coefficients for the second components are $\beta_{21} = -0.5$ and $\beta_{22} = 0.8$. Each plot is corresponding to one value of $\pi \in \{0, 0.2, 0.4, 0.6, 0.8, 1\}$.

However, things are becoming more interesting when the value of π *is somewhere between 0 and 1. The surface of the function becomes twisted and departs further away from either extremes as* π *approaches* 0.5.

We further project the 3D surfaces onto a 2D plane to visualize the decision boundaries. In order to show the non-linearity of the decision boundaries, we choose to focus on the range of π between 0.4 and 0.6. In Figure 4.1, the surfaces are mostly twisted within this range. The projected contour plots with decision boundaries are shown in Figure 4.2. Indeed, we can see from Figure 4.2 that with a proper value of π HMLRM with two hidden groups has the ability to model a nonlinear decision boundary.

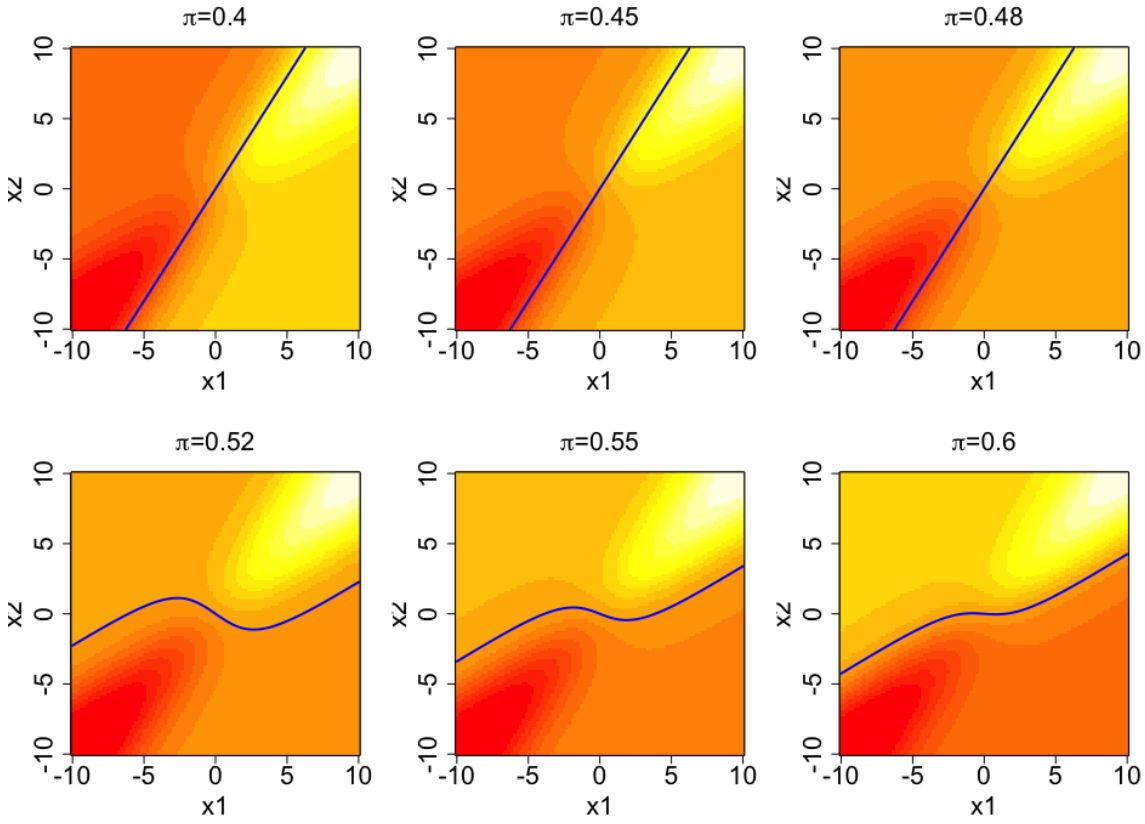

Fig. 4.2. 2D contour plots for the weight tanh function. The coefficients for the first component are $\beta_{11} = 0.8$ and $\beta_{12} = -0.5$. The coefficients for the second components are $\beta_{21} = -0.5$ and $\beta_{22} = 0.8$. Each plot is corresponding to one value of $\pi \in \{0.4, 0.45, 0.48, 0.52, 0.55, 0.6\}$. The decision boundaries are indicated by the blue curves.

In Example 4.1.1, we use only two hidden groups and it is enough to demonstrate the nonlinearity of HMLRM. In practice, using just two hidden groups may not be

enough for a complex dataset, adding more hidden groups would help to reveal more information about the structure. In the extreme case, the surface is given by

$$f(\boldsymbol{x}|\boldsymbol{\beta}) = \int \text{logit}(\boldsymbol{x}^T\boldsymbol{\beta}) f(\boldsymbol{\beta}) d\boldsymbol{\beta},$$

where $f(\boldsymbol{\beta})$ is a density function of the parameter vector $\boldsymbol{\beta}$. This can be used when we believe that there infinitely many groups.

4.2 HMLRM for Big Data

When the sample size is large, the previously presented algorithm will be inefficient. As the size of the matrices grows, the cost of computation involving those matrices will become the bottleneck of the algorithm. In addition, the size of a extremely large data set can easily reaches scale of Terabyte or even Petabyte. With the current technology, a data set of this scale does not fit the storage on a single computer and distributed file systems are needed to save large amount of data.

In order to apply HMLRM on big data, a parallel version of our algorithm is needed. There has been effort to develop new technologies in the past decade to facilitate data analysis on big data. Currently, there are two major solutions for big data computation, Hadoop and Spark.

Hadoop consists of an implementation of distributed file system (HDFS, readers can refer to [66] for the description) which decentralizes data storage and the MapReduce computing model (see [67]) which is designed for performing computation on data in HDFS. For a large data set, it is difficult or impossible to fit in a single computer's storage (of which an hard disk is the most common device). Using HDFS allows us to break a large data file into smaller blocks which are then stored distributively across a cluster of computer nodes in the format of key-value pairs. Hadoop's MapReduce computing model describes a computation job (e.g. estimating a statistical model) in terms of one or multiple map and reduce steps which takes advantage of the data format in HDFS. For example, counting the word frequencies

in a document can be carried out by mapping each word to a tuple of (word, 1) and then adding up the value 1's for the same word.

By using HDFS and MapReduce, many data analysis tasks can be scaled up in terms of the size of data sets they can handle. There exist several statistical data analysis solutions based on Hadoop, such as RHadoop[19] and Tessera[20].

Though Hadoop has been quite popular for big data analysis, one of its major drawbacks is that it requires heavy disk I/O. The intermediate results need to be written to and read from disk between two tasks. In exploratory data analysis (EDR), a data set usually needs to be processed repeatedly where each time a different algorithm is applied to investigate one or more aspects of the data. Moreover, many statistical or machine learning methods rely on iterative algorithms where intermediate results need to be fed into subsequent iterations. The fact that using Hadoop requires heavy disk I/O in between each map and reduce steps makes data analysis, especially EDR, inefficient.

Apache Spark [69] is an alternative approach for big data analysis. It is an implementation of an in-memory computing model and can access HDFS to use it as a storage backend. Spark's in-memory computing model is suitable for low-latency applications and iterative computations. At the core of Spark is the data abstraction called Resilient Distributed Dataset (RDD) [70] which enables applications to persist data in memory (instead of disk). Along with wide variety of supported operators, such as mappers, reducers, joins, group-bys, and filters, computation on big data can be performed more efficiently at memory speed[21].

[19]RHadoop, developed by Revolution Analytics, is a collection of five R packages that allow users to manage and analyze data with Hadoop. Details about RHadoop can be found at `https://github.com/RevolutionAnalytics/RHadoop/wiki`.

[20]The Tessera is a collection of three R packages that provide a computational environment for Divide and Recombine approach of data analysis. Details about Tessera can be found at `http://tessera.io/` and in the related paper [68].

[21]According to `http://spark.apache.org`, Spark can run programs up to 100x faster than Hadoop MapReduce in memory, or 10x faster on disk.

Though Spark includes several tool boxes, such as MLlib[22] for machine learning and GraphX[23] for graph computation, many statistical models are still missing. In this section, we will show an implementation of HMLRM in Apache Spark, which is an attempt to apply statistical modelling on a big data platform. In the simulation study, we will test its speed performance.

4.2.1 Data Partition

In order to parallelize the estimation algorithm for HMLRM, we first need to consider data partition. After we can break a large data set into smaller chunks so that each chunk is of reasonable size for being computed on an individual core, we can carry out computation for each chunk. Then, the results from all chunks can be combined to achieve the global result as if the whole data set were analyzed at once.

Our approach seems similar to Divide-and-Recombine (D&R) described in [68]. However D&R does not relay on communication between computing processes on different data chunks. Thus the result of D&R is not guaranteed to be globally optimal. This is true when the data analysis procedure requires iterations and the calculation of the intermediate quantities relies on the whole data set, for example estimating a logistic regression requires iterative optimization. By using Spark, data chunks are allowed to be stored in memory which provides faster access during computation and inter-node communication through network. So we can achieve global solutions more efficiently than Hadoop based approaches in many situations.

We assume the data for estimating a HMLRM is arranged in a matrix with each row corresponding to one sample. The matrix is divided into smaller chunks with each chunk as a subset of the data containing multiple rows. This is illustrated in the following picture.

[22] `http://spark.apache.org/mllib/`.

[23] `http://spark.apache.org/graphx/`.

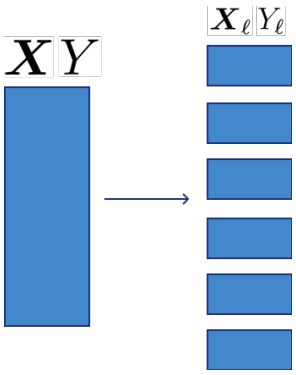

Fig. 4.3. Illustration of data partition for parallelizing HMLRM estimation.

We use ℓ as the index for the data chunks and the total number of the chunks is denoted as L. In the extreme case, each row can be a chunk by itself.

4.2.2 Parallel Computation

For each chunk of data, we will calculate its own pieces of $\mathcal{Q}_\ell^{(h)}$, $\mathcal{Q}_{c,\ell}^{(o)}$, $\mathbb{g}_\ell^{(h)}$, $\mathbb{g}_{c,\ell}^{(o)}$'s, $\mathbb{H}_\ell^{(h)}$, and $\mathbb{H}_{c,\ell}^{o}$. They can be calculated by using the same formulae in (4.13), (4.18), (4.21), (4.22), (4.23), and (4.24), except that the chunked data is used this time.

Calculating those quantities for each chunk instead the whole data set reduces the computational cost. Since the calculation for one chunk does not involve other chunks, the calculation for all chunks can be carried out in parallel.

Under the assumption that the samples being independent, the results from all chunks can be combined to produce their global counterparts as follows.

$$\mathcal{Q}^{(h)} = \sum_{\ell=1}^{L} \mathcal{Q}_\ell^{(h)}, \quad \mathbb{g}^{(h)} = \sum_{\ell=1}^{L} \mathbb{g}_\ell^{(h)}, \quad \mathbb{H}^{(h)} = \sum_{\ell=1}^{L} \mathbb{H}_\ell^{(h)}.$$
$$\mathcal{Q}_c^{(o)} = \sum_{\ell=1}^{L} \mathcal{Q}_{c,\ell}^{(o)}, \quad \mathbb{g}_c^{(o)} = \sum_{\ell=1}^{L} \mathbb{g}_{c,\ell}^{(o)}, \quad \mathbb{H}_c^{o} = \sum_{\ell=1}^{L} \mathbb{H}_{c,\ell}^{o}.$$

It is the combining step that requires communication between different chunks.

4.2.3 Implementation in Spark

In order to implement HMLRM in Spark, we rely on RDD to store the chunked data and its transformation and action methods for computation. As we mentioned before, RDD is Spark's data abstraction which is implemented to represent a large data set distributedly in memory. The representation of data in RDD facilitate various operations which belong to two types: transformations and actions[24].

A transformation operation passes each dataset element through a function and produces a new RDD of the transformed data. For example, a `softmax` function can be applied to each sample to calculate the hidden membership probabilities, which is carried out by Spark's `map` function as `dataRdd.map(softmax)`. Other Spark's transformation operations include `filter`, `flatMap`, `mapPartitions` and so on.

An action operation aggregate all the dataset elements in an RDD through some function and returns a single value. For example, taking the sum of data chunk's likelihood to calculate the likelihood for the whole dataset is an action, which can be carried out in Spark as `chunkLikelihoodRdd.sum()`. Other Spark's action operations include `reduce`, `collect`, count and so on.

In our implementation, we use RDD to represent chunked data in the memory of a cluster. Given the current estimate of the parameters, transformations with properly created functions are applied to compute the RDDs of hidden membership probabilities, hidden layer conditional probabilities, observing layer probabilities, complete likelihoods, gradients and hessians. Then, reduce actions are applied on the resulting RDDs to sum up the RDD elements to obtain the global value of complete likelihood, gradients and hessians. Those global quantities will be fed into an optimizer to perform optimization to find the next update of the parameters. This is illustrated in the following picture.

[24] `http://spark.apache.org/docs/latest/programming-guide.html`

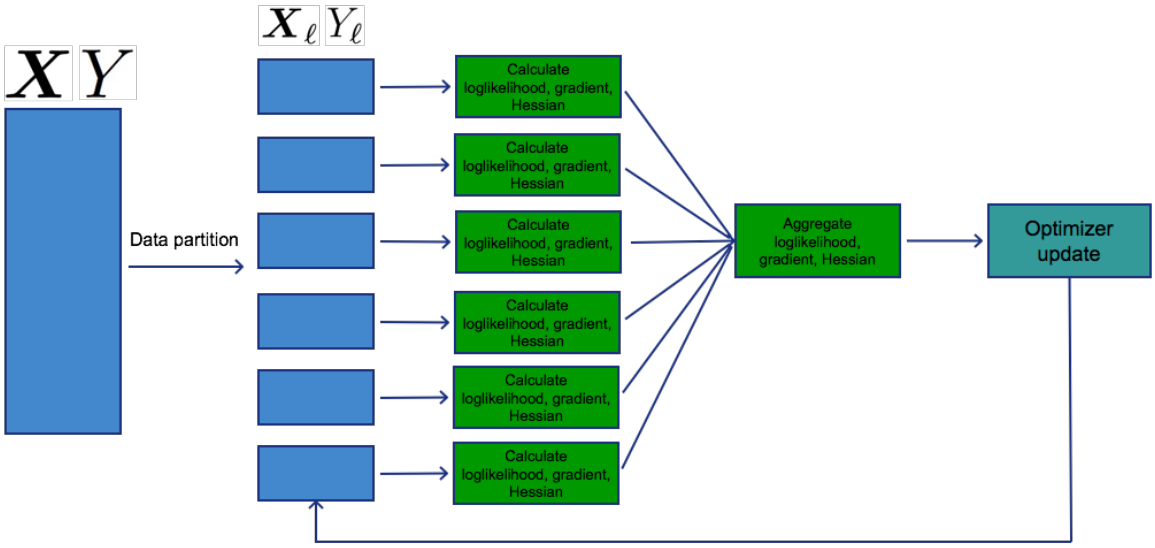

Fig. 4.4. Illustration of model estimation of HMLRM with data partition and parallel computation.

The modified algorithm for HMLRM estimation in Spark is shown in Algorithm 5.

4.3 Simulation Study

In this section, we demonstrate some empirical properties of HMLRM using simulation study. In particular, we how the fitting performance and generalization performance of the model. In addition, we will also show the performance of the Spark version of HMLRM and discuss some practical concerns regarding its implementation in Python Spark and application in data analysis.

4.3.1 Model Fitting and Prediction

In the first simulation study, we evaluate the performance of parameter estimation, model fitting and prediction on unseen data using HMLRM. We use Algorithm 4 and implement it in Python. The optimization in M-steps is realized by calling Scipy's minimizer function with L-BFGS-B method. L-BFGS-B is one of the quasi-

Algorithm 5 Parallel EM for HMLRM in Spark

function TRAINHMLRM(*distData*, $\mathbf{A}^{(0)}$, $\mathbf{B}_c^{(0)}$'s, ϵ_1, ϵ_2, *maxIter*)
  $param^{(0)} \leftarrow \{\mathbf{A}^{(0)}, \mathbf{B}_c^{(0)}\text{'s}\}$
  $nollk^{(0)} \leftarrow$ calculate negative log-likelihood for observed data using $param^{(0)}$
  $relAbsDiffNollk \leftarrow \infty$
  $relDiffParams \leftarrow \infty$
  $t = 0$
  while $relAbsDiffNollk >= \epsilon_1$ and $relDiffParams >= \epsilon_2$ **do**
    $\pi^{(t)} \leftarrow$ *distData.map*(compute sigmoid or softmax for a chunk of $\mathbf{X}_\ell^{(h)}$ using $\mathbf{A}^{(t)}$)
    $p^{(t)} \leftarrow$ *distData.map*(compute sigmoid or softamx for a chunk of $\mathbf{X}_\ell^{(o)}$ using $\mathbf{B}_c^{(t)}$'s)
    $q^{(t)} \leftarrow$ RDD of $P(Z|Y)$ calculated from $\pi^{(t)}$ and $p^{(t)}$
    Define $\mathcal{Q}^{(h)}(\mathbf{A}, q = q^{(t)})$ (E-step for hidden layer)
    $\mathbf{A}^{(t+1)} \leftarrow \arg\min_{\mathbf{A}}(-\mathcal{Q}^{(h)})$ (M-step for the hidden layer)
    for each hidden group c **do**
      Define function $\mathcal{Q}_c^{(o)}(\mathbf{B}_c, q = q^{(t)})$ (E-step for the observed layer)
      $\mathbf{B}^{(t+1)} \leftarrow \arg\min_{\mathbf{B}_c}(-\mathcal{Q}_c^{(o)})$ (M-step for the observed layer)
    end for
    $param^{(t+1)} \leftarrow \{\mathbf{A}_c^{(t+1)}, \mathbf{B}_c^{(t+1)}\text{'s}\}$
    $relDiffParams \leftarrow \frac{\left\|param^{(t+1)} - param^{(t)}\right\|}{\left\|param^{(t)}\right\|}$
    $nollk^{(t+1)} \leftarrow$ calculate negative log-likelihood for observed data using $param^{(t+1)}$
    $relAbsDiffNollk \leftarrow \frac{\left|nollk^{(t+1)} - nollk^{(t)}\right|}{nollk^{(t)}}$
    $t = t + 1$
    if $t \geq maxIter$ **then**
      break
    end if
  end while
  return ($param^{(t)}$, $nollk^{(t)}$)
end function

Newton optimization methods that approximates the original BroydenFletcherGoldfarbShanno (BFGS) algorithm with reduced amount of computer memory. Instead of storing the whole dense matrix that approximate the inverse of Hessian, the limited version only stores the vectors that represent the approximation implicitly.

The true model we use here has three predictor variables in the hidden layer, four predictor variables in the observed layer, and three hidden groups. Including the intercepts, there are 23 parameters to estimate, 8 from the hidden layer and 15 from the observed layer. In each round of simulation, we generate 50,000 independent random samples where each predictor variable is Unif$(-1, 1)$. The parameters are estimated by applying the non-distributed algorithm on the dataset with the initial estimate of the parameters randomly generated from Unif$(-1, 1)$. This is repeated 100 times.

The estimates of the parameters are reported using boxplots which are shown in Figure 4.5. The mean and the standard deviation are also calculated and reported in Table 4.1. It can be seen that the estimated parameters with random initials perform quite well, which are close to the true parameters with small standard deviation. We can see that some plots show a few outliers, for example plot 2 and plot 7. During the simulation study, we found that there are cases that the parameters that are not estimated well. But the corresponding log-likelihood is smaller than that of the true parameters. This is an indication that a local minimum instead of the global minimum is found during optimization.

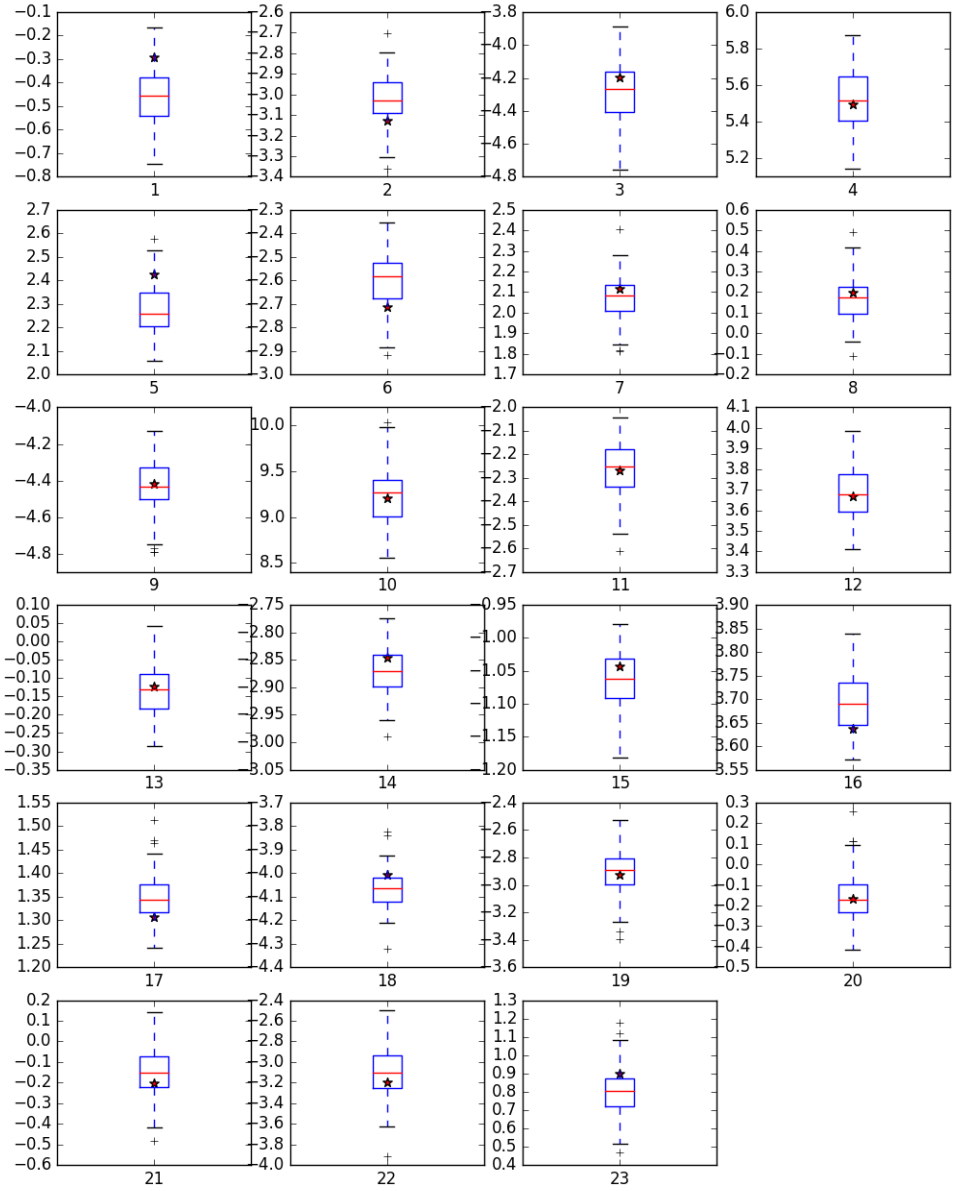

Fig. 4.5. Boxplots of estimated parameters for HMLRM using the non-distributed version. The star in each plot represents the true parameter value.

Table 4.1.
Numerical summary of the estimated parameters for HMLRM using the non-distributed version. The mean and standard deviations of the estimated parameters are reported in the table along with the true values of the parameters.

Parameter Id	True Parameters	Mean Estimates	Std. Dev.
1	-0.2909	-0.4612	0.1280
2	-3.1259	-3.0247	0.1202
3	-4.1954	-4.2725	0.1649
4	5.4946	5.5222	0.1540
5	2.4255	2.2714	0.1065
6	-2.7146	-2.6027	0.1159
7	2.1139	2.0720	0.1024
8	0.1981	0.1752	0.1033
9	-4.4183	-4.4307	0.1421
10	9.2092	9.2494	0.2957
11	-2.2667	-2.2665	0.1140
12	3.6660	3.6829	0.1348
13	-0.1229	-0.1356	0.0685
14	-2.8451	-2.8718	0.0425
15	-1.0431	-1.0628	0.0429
16	3.6378	3.6922	0.0607
17	1.3063	1.3460	0.0495
18	-4.0053	-4.0676	0.0778
19	-2.9242	-2.9090	0.1542
20	-0.1689	-0.1594	0.1180
21	-0.2010	-0.1528	0.1117
22	-3.1942	-3.0979	0.2383
23	0.8968	0.8011	0.1424

Besides parameter estimation, we are also interested in how well the estimated model fit the observed dataset and generalize to unseen data. We keep the same model setting and sample size as above. In each round of simulation, we generate a random dataset as the training set and estimate the parameters. Then we predict the response for the dataset and calculate the percentage that the predicted response agrees with the observed response. In order to evaluate the generalization performance, we generate an independent data set of size 10,000 as the test set. After the parameters are estimated, they are used on the test set to predict the response and calculate the percentage that the predicted response agrees with the observed response. We run this simulation 100 times. The results are reported in Figure 4.6. The average percentages for fitting training set and predicting test set are 0.7783 and 0.7779 respectively. They are quite close. The standard deviations are 0.0394 and 0.0401 respectively. The fact that the two percentages are so close is an indication that over-fitting might not be a big concern for HMLRM. this is different from a machine learning model where regularization is usually needed to avoid over-fitting.

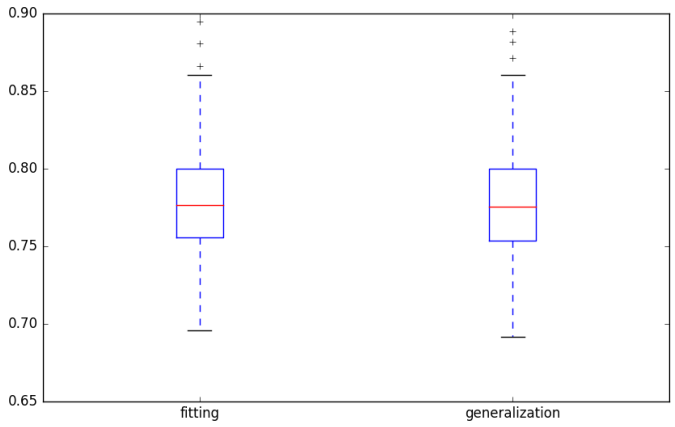

Fig. 4.6. Boxplots showing fitting and generalization performance. Left: percentage of the predicted responses that agree with the observed responses using the training set. Right: percentage of the predicted responses that agree with the observed responses using the test data.

4.3.2 Implementation in PySpark

In order to implement the parallel version of HMLRM estimation in Spark, we choose to use PySpark. This is Spark's Python API which allows Spark's core functionality to be used by Python programmers. Python is a very popular high level programming language among data scientists. Compared to R, Python has limited number of packages for statistical analysis. But it is much more powerful in data processing, integration with operating systems, and module development.

PySpark bridges Spark's JVMs and Python interpreter processes together so that Spark's JVM manages jobs and data RDDs and Python processes are responsible for processing data. On the driver side, we create a `SparkContext` object in Python which is mapped to the `SparkContext` in JVM by using Python's Py4J package. The JVM SparkContext spawns Spark's executors on worker nodes and each executor is a JVM which manages its local portion of data RDD. When computation is needed, each executor will launch a Python interpreter and send serialized data to it. The results will be serialized and sent back from Python to JVM after the computation finishes. Because of the bridging needed between Spark JMVs and Python processes, the efficiency of PySpark is suboptimal compared to its Scala counterpart.

While we can follow Algorithm 5 to implement the parallel model estimation, there are a few concerns we should bear in mind in practice.

First, when we divide a data set according to Section 4.2.1, we need to decide the number of chunks or equivalently the size of each chunk. In an extreme case, each sample can be a chunk by itself. However, this is not efficient especially in a high level programming language like Python. Since the computation on each partition is executed as a single task in Spark, samples in a dataset will be processed one by one and hence we will not be able to use vectorized matrix computation in Python.

Table 4.2 shows the elapsed time for estimating a HMLRM for the same dataset but with different number of partitions. The model we use has 5 predictors in the hidden layer, 10 predictors in the observed and 3 subpopulations. The sample size is

fixed to 50,000. The dataset is divided into 10, 20, 30, 40, 50, and 60 partitions. For each number of partitions, we recorded the time used for estimating the parameters. It can be seen in Table 4.2 that the elapsed time increase when we divide the dataset into more partitions.

Table 4.2.
Elapsed time of estimating HMLRM in Spark with different number of partitions. The sample size is fixed to 50,000.

Number of Partitions	Elapsed Time (sec)
10	1192.96941
20	1893.25854
30	2723.59362
40	3600.55789
50	4728.81499
60	4928.81704

Second, though spark applications can be deployed on computer clusters, the overhead of network communication is not negligible. This cost can become a bottle when Spark actions involving data shuffle are called. If a job can be fit on a multi-core workstation, using Spark's local mode will be able to reduce the cost of network communication.

In order to have an impression of the overhead of network communication, we compare the speed of our parallel algorithm in two different Spark deploy modes, local mode and yarn-client mode. When local mode is used, Spark will run the job in a single executor JVM and the parallelization is realized by the fact it could use multiple cores and each core run a thread to execute a task. In yarn-client mode, Spark delegate resource management to Hadoop's Yarn resource manager. In order to run the job, it will launch multiple executor JVMs across the cluster and each

is responsible for executing tasks. Since the executors in yarn-client mode reside in different nodes, their communication is vis the cluster's network and data and codes need to be serialized and unsecularized between different nodes. We simulate 50,000 observations from a simple HMRLM with 3 predictor variables in the hidden layer, 4 predictor variables in the observed layer, and 3 subpopulations. The data set is divided into different number of partitions. For each partition number, we estimate the parameters using the parallel algorithm in both local and yarn-client mode and record the time it uses for the estimation. We use 5 cores in the local mode and 5 executors with 1 core per executor in the yarn-client mode. The results are reported in Table 4.3.

Table 4.3.
Time used to estimate HMLRM in Spark using two different modes, local and yarn-client. In the algorithm, my naive gradient descent is used for optimization in M-step. The tolerance for EM convergence is set to 1e-3.

Number of Partitions	Local[5]	Yarn-client (5 executors, 1 core)
10	418.20794	343.69576
20	594.34379	340.89911
30	728.73377	676.55508
40	906.67940	845.82275
50	558.75216	1339.92693
60	1410.18719	3670.39308
70	1630.23311	3235.92743
80	2182.04236	5210.86841
90	2438.11216	3062.45193
100	2692.75372	4270.10367

We can see from Table 4.3 that when the number of partitions increases, the elapsed time for model estimation also increases. As we discussed above, one reason

is that smaller than necessary chunk size cause the job to fail to use the full potential of each core. Another reason is that, dividing dataset into more chunks will require more time to send back to the driver the computation results from the chunks. In addition, we can also see that when the number of partition becomes large, the performance of using yarn-client mode gets worse. This is because network communication between different executor JVMs across the cluster contributes a significant portion to the elapsed time.

Third, though the likelihood, gradient and Hessian are computed for each data chunk, they will be aggregated in Spark's driver process. In theory, those aggregated quantities can be passed to any optimization routine that takes them as inputs. For example, we first started the simulation study by simply calling Scipy's minimize routine in each M-step. However, when the sample size increases, Spark job starts to crash. One cause of this issue is that the intermediate steps in a usual optimization routine may result in memory footprints that are not suitable for parallelized applications. The optimization algorithms need to be overhauled to take into account data and computation parallelization. In the latest version of Spark (1.6.2 at the time of writing this thesis), there are only limited optimization algorithms provided by the MLlib toolbox, including gradient descent, stochastic gradient descent, and L-BFGS. Since they are only available in Scala and Java, we couldn't use them. In our simulation, we implemented a naive gradient descent routine to update the parameters in M-steps. In the future, more sophisticated implementation of the optimization algorithms is needed.

Finally, the performance of our algorithm depends on Spark's configuration, such as memory allocated to each executor, number of cores for each executor, and choice of resource manager. Allocating less than enough memory to executors would raise out of memory exceptions and crashes a Spark job. Since an executor JVM can utilize multiple cores to run tasks simultaneously, allocating more cores to an executor would improve Spark's performance. A resource manager affects the performance of Spark in

the way of how it allocates cluster resources, such as CPUs and memory, to executors. Tuning Spark is specific to a given application and is beyond the scope of this thesis.

4.4 Summary

In this chapter, we propose to use Hierarchical Mixed Logistic Regression Model (HMLRM) for exploring big data. Its hidden layer explicitly models the possible subpopulation structure, which is one way to handle heterogeneity in data and results in a non-linear classification boundary. The model construction and its estimation using EM algorithm are fully discussed. The separability of the parameter space is helpful to reduce the dimensionality during estimation. In order to apply HMLRM on big data with large sample size, we design a parallel algorithm in which a dataset is divided into smaller chunks and the computation of likelihoods and gradients are carried out in a map-reduce fashion during each EM iteration. The parallel algorithm is implemented in Apache Spark. We choose Spark because its in-memory computing model and data abstraction RDD provides better performance in terms of speed and disk I/O. The simulation study with the non-parallel algorithm shows that using EM is able to estimate the model parameters reasonably well and the model seems to be quite robust in terms of its prediction performance. The simulation with the parallel algorithm proves that our idea that we can build a big model for exploring big data is viable.

However, there are a few concerns that need to be addressed before Spark can be used for exploratory data analysis. First, since EDA requires interactive data analysis and the results need to be delivered in a timely manner, the low efficiency, mainly caused by network communication, could easily break the smoothness of this workflow and results in an unpleasant user experience. Second, the eco-system for Spark is not available yet. This is especially true for data exploration. Besides developing statistical models suitable for big data, many low level numerical routines

such as matrix operations and optimization algorithms need to be reimplemented for Spark.

5. CONCLUSION AND FUTURE WORK

In this thesis, we focus on developing effective methods for data exploration in regression. We proposed to use Optimal Kernel Group Transformation (OKGT) method to explore the relationship between Y and $\boldsymbol{X}$. By introducing the concept of group structure to additive model, we are allowed to consider interactions between predictor variables. Using kernel methods for non-parametric estimation in OKGT circumvents the curse of dimensionality so that the method scales well in number of predictor variables. The effectiveness of OKGT also relies on using a proper group structure. So we further developed Additive Group Structure Identification (AGSI) method and algorithms for finding the true additive group structure from data. By introducing a novel penalty to control the complexity of group structures, we could prove the selection consistency of AGSI. We believe that OKGT and AGSI together could be combined as a general framework for data exploration in regression.

We also developed the Hierarchical Mixed Logistic Regression Model (HMLRM) as an attempt to explore big data. The model explicitly models the subpopulation structure by introducing a hidden layer of regression. This is one way to attack heterogeneity which is common in big data. Further, we parallelized the EM algorithm for parameter estimation and implemented it in Spark. We evaluate the performance of HMLRM in Spark in terms of speed and discussed some practical concerns. We feel that more efficient statistical methods and computation tools needs to be developed for exploring big data.

The research in this thesis suggests several directions for future work. First, the current algorithms for AGSI is not efficient when the number of predictor variables is large. Exhaustively searching over all possible group structures is not feasible and the backward stepwise algorithm also does not scale well and is based on heuristics. A more efficient algorithm which is theoretically justified is needed. Second, in a high

dimensional setting, sparsity is usually expected among the predictor variables. So we need a procedure to recover sparsity to be combined in AGSI. Third, the proof of selection consistency for AGSI relies on the assumption that predictor variables are independent. We wonder if the same result holds if there is dependency. Fourth, there are lot to be done to bring statistics to the big data domain. Our mixed logistic model needs to be further polished, for example using more efficient optimization algorithms suitable for Spark, to be suitable for big data exploration. We also need to compare mixed logistic with other machine learning methods such as neural networks and machines in terms of their prediction performance. It is also interesting to know if there is any connection between mixed logistic and neural networks.

REFERENCES

REFERENCES

[1] J. W. Tukey, "Exploratory data analysis," 1977.

[2] A. Buja, T. Hastie, and R. Tibshirani, "Linear smoothers and additive models," *The Annals of Statistics*, pp. 453–510, 1989.

[3] L. Breiman and J. H. Friedman, "Estimating optimal transformations for multiple regression and correlation," *Journal of the American statistical Association*, vol. 80, no. 391, pp. 580–598, 1985.

[4] T. Hastie and R. Tibshirani, "Generalized additive models," *Statistical science*, pp. 297–310, 1986.

[5] T. Hofmann, B. Schölkopf, and A. J. Smola, "Kernel methods in machine learning," *The Annals of Statistics*, pp. 1171–1220, 2008.

[6] J. Shawe-Taylor and N. Cristianini, *Kernel methods for pattern analysis*. Cambridge university press, 2004.

[7] A. J. Smola and B. Schölkopf, *Learning with kernels*. Citeseer, 1998.

[8] G. Wahba, "An introduction to reproducing kernel hilbert spaces and why are they so useful," 2003.

[9] N. Aronszajn, "Theory of reproducing kernels," *Transactions of the American mathematical society*, vol. 68, no. 3, pp. 337–404, 1950.

[10] G. S. Kimeldorf and G. Wahba, "A correspondence between bayesian estimation on stochastic processes and smoothing by splines," *The Annals of Mathematical Statistics*, vol. 41, no. 2, pp. 495–502, 1970.

[11] B. Schölkopf, R. Herbrich, and A. J. Smola, "A generalized representer theorem," in *Computational learning theory*. Springer, 2001, pp. 416–426.

[12] B. Schölkopf and A. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, ser. Adaptive computation and machine learning. MIT Press, 2002. [Online]. Available: https://books.google.com/books?id=y8ORL3DWt4sC

[13] B. Schölkopf, A. Smola, and K.-R. Müller, "Kernel principal component analysis," in *International Conference on Artificial Neural Networks*. Springer, 1997, pp. 583–588.

[14] S. Akaho, "A kernel method for canonical correlation analysis," in *In Proceedings of the International Meeting of the Psychometric Society (IMPS2001*, 2001.

[15] K. Fukumizu, F. R. Bach, and M. I. Jordan, "Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces," *Journal of Machine Learning Research*, vol. 5, no. Jan, pp. 73–99, 2004.

[16] ——, "Kernel dimension reduction in regression," *The Annals of Statistics*, pp. 1871–1905, 2009.

[17] A. Gretton, K. Fukumizu, C. H. Teo, L. Song, B. Schölkopf, and A. J. Smola, "A kernel statistical test of independence," in *Advances in neural information processing systems*, 2007, pp. 585–592.

[18] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, "Text classification using string kernels," *Journal of Machine Learning Research*, vol. 2, no. Feb, pp. 419–444, 2002.

[19] C. S. Leslie, E. Eskin, and W. S. Noble, "The spectrum kernel: A string kernel for svm protein classification." in *Pacific symposium on biocomputing*, vol. 7, no. 7, 2002, pp. 566–575.

[20] L. Bo, X. Ren, and D. Fox, "Kernel descriptors for visual recognition," in *Advances in Neural Information Processing Systems 23*, J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, Eds. Curran Associates, Inc., 2010, pp. 244–252. [Online]. Available: http://papers.nips.cc/paper/4147-kernel-descriptors-for-visual-recognition.pdf

[21] L. Bo, K. Lai, X. Ren, and D. Fox, "Object recognition with hierarchical kernel descriptors," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011, pp. 1729–1736.

[22] A. Daemen and B. De Moor, "Development of a kernel function for clinical data," in *2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, 2009, pp. 5913–5917.

[23] M. Rosenblatt, "Conditional probability density and regression estimators," *Multivariate Analysis II*, vol. 25, p. 31, 1969.

[24] J. Fan, Q. Yao, and H. Tong, "Estimation of conditional densities and sensitivity measures in nonlinear dynamical systems," *Biometrika*, vol. 83, no. 1, pp. 189–206, 1996.

[25] M. Sugiyama, I. Takeuchi, T. Suzuki, T. Kanamori, H. Hachiya, and D. Okanohara, "Conditional density estimation via least-squares density ratio estimation," in *International Conference on Artificial Intelligence and Statistics*, 2010, pp. 781–788.

[26] B. W. Silverman, *Density estimation for statistics and data analysis*. CRC press, 1986, vol. 26.

[27] R. Sakia, "The box-cox transformation technique: a review," *The Statistician*, pp. 169–178, 1992.

[28] P. Burman, "Rates of convergence for the estimates of the optimal transformations of variables," *Annals of Statistics*, vol. 19, no. 2, pp. 702–723, 1991.

[29] C. R. Baker, "Joint measures and cross-covariance operators," *Transactions of the American Mathematical Society*, vol. 186, pp. 273–289, 1973.

[30] K. Fukumizu, F. R. Bach, and A. Gretton, "Statistical consistency of kernel canonical correlation analysis," *The Journal of Machine Learning Research*, vol. 8, pp. 361–383, 2007.

[31] A. Berlinet and T. C. Agnan, *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Kluwer Academic Publishers, 2004.

[32] Y. Sato, "Theoretical Considerations for Multivariate Functional Data Analysis," in *Proceedings 59th ISI World Statistics Congress*, no. August, Hong Kong, 2013, pp. 25–30. [Online]. Available: http://www.statistics.gov.hk/wsc/CPS021-P3-S.pdf

[33] F. Bach and M. Jordan, "Kernel independent component analysis," *The Journal of Machine Learning Research*, vol. 3, pp. 1–48, 2003.

[34] A. J. Smola and B. Schölkopf, "Sparse greedy matrix approximation for machine learning," 2000.

[35] F. R. Bach, "Sharp analysis of low-rank kernel matrix approximations." in *COLT*, vol. 30, 2013, pp. 185–209.

[36] P. Drineas and M. W. Mahoney, "On the nyström method for approximating a gram matrix for improved kernel-based learning," *Journal of Machine Learning Research*, vol. 6, no. Dec, pp. 2153–2175, 2005.

[37] J. J. Thompson, M. R. Blair, L. Chen, and A. J. Henrey, "Video game telemetry as a critical tool in the study of complex skill learning," *PloS one*, vol. 8, no. 9, p. e75129, 2013.

[38] R. McLendon, A. Friedman, D. Bigner, E. G. Van Meir, D. J. Brat, G. M. Mastrogianakis, J. J. Olson, T. Mikkelsen, N. Lehman, K. Aldape *et al.*, "Comprehensive genomic characterization defines human glioblastoma genes and core pathways," *Nature*, vol. 455, no. 7216, pp. 1061–1068, 2008.

[39] A. Fawzi, J.-B. Fiot, B. Chen, M. Sinn, and P. Frossard, "Structured dimensionality reduction for additive model regression," Institute of Electrical and Electronics Engineers, Tech. Rep., 2015.

[40] G. S. Watson, "Smooth regression analysis," *Sankhyā: The Indian Journal of Statistics, Series A*, pp. 359–372, 1964.

[41] E. A. Nadaraya, "On estimating regression," *Theory of Probability & Its Applications*, vol. 9, no. 1, pp. 141–142, 1964.

[42] G. Wahba, *Spline models for observational data*. Siam, 1990, vol. 59.

[43] C. Pan, Q. Huang, and M. Zhu, "Optimal kernel group transformation for exploratory regression analysis and graphics," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015, pp. 905–914.

[44] I. Steinwart and A. Christmann, *Support vector machines*. Springer Science & Business Media, 2008.

[45] C. Carmeli, E. De Vito, A. Toigo, and V. Umanitá, "Vector valued reproducing kernel hilbert spaces and universality," *Analysis and Applications*, vol. 8, no. 01, pp. 19–61, 2010.

[46] D.-X. Zhou, "The covering number in learning theory," *Journal of Complexity*, vol. 18, no. 3, pp. 739 – 767, 2002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0885064X02906357

[47] V. Vapnik, *The nature of statistical learning theory*. Springer Science & Business Media, 2013.

[48] P. L. Bartlett and S. Mendelson, "Rademacher and gaussian complexities: Risk bounds and structural results," *The Journal of Machine Learning Research*, vol. 3, pp. 463–482, 2003.

[49] T. Poggio and C. Shelton, "On the mathematical foundations of learning," *American Mathematical Society*, vol. 39, no. 1, pp. 1–49, 2002.

[50] D.-X. Zhou, "The covering number in learning theory," *Journal of Complexity*, vol. 18, no. 3, pp. 739–767, 2002.

[51] T. Kühn, "Covering numbers of Gaussian reproducing kernel Hilbert spaces," *Journal of Complexity*, vol. 27, no. 5, pp. 489–499, 2011. [Online]. Available: http://dx.doi.org/10.1016/j.jco.2011.01.005

[52] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, ser. Springer Series in Statistics. New York, NY, USA: Springer New York Inc., 2001.

[53] N. Alon, S. Ben-David, N. Cesa-Bianchi, and D. Haussler, "Scale-sensitive dimensions, uniform convergence, and learnability," *Journal of the ACM (JACM)*, vol. 44, no. 4, pp. 615–631, 1997.

[54] F. Cucker and S. Smale, "On the mathematical foundations of learning," *American Mathematical Society*, vol. 39, no. 1, pp. 1–49, 2001.

[55] F. Kuo, I. Sloan, G. Wasilkowski, and H. Woźniakowski, "On decompositions of multivariate functions," *Mathematics of computation*, vol. 79, no. 270, pp. 953–966, 2010.

[56] T. Kühn, "Covering numbers of gaussian reproducing kernel hilbert spaces," *Journal of Complexity*, vol. 27, no. 5, pp. 489–499, 2011.

[57] J. Fan, F. Han, and H. Liu, "Challenges of big data analysis," *National science review*, vol. 1, no. 2, pp. 293–314, 2014.

[58] N. O. Hodas and K. Lerman, "The simple rules of social contagion," *arXiv preprint arXiv:1308.5015*, 2013.

[59] K. E. Train, *Discrete choice methods with simulation*. Cambridge university press, 2009.

[60] P. Wang and M. L. Puterman, "Mixed logistic regression models," *Journal of Agricultural, Biological, and Environmental Statistics*, pp. 175–200, 1998.

[61] P. McCullagh and J. A. Nelder, *Generalized linear models*. CRC press, 1989, vol. 37.

[62] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977.

[63] C. M. Bishop, "Pattern recognition and machine learning (information science and statistics)," 2006.

[64] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[65] A. Björck, *Numerical methods for least squares problems*. Siam, 1996.

[66] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The hadoop distributed file system," in *2010 IEEE 26th symposium on mass storage systems and technologies (MSST)*. IEEE, 2010, pp. 1–10.

[67] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[68] S. Guha, R. Hafen, J. Rounds, J. Xia, J. Li, B. Xi, and W. S. Cleveland, "Large complex data: divide and recombine (d&r) with rhipe," *Stat*, vol. 1, no. 1, pp. 53–67, 2012.

[69] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets," in *Proceedings of the 2Nd USENIX Conference on Hot Topics in Cloud Computing*, ser. HotCloud'10. Berkeley, CA, USA: USENIX Association, 2010, pp. 10–10. [Online]. Available: http://dl.acm.org/citation.cfm?id=1863103.1863113

[70] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. J. Franklin, S. Shenker, and I. Stoica, "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing," in *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*. USENIX Association, 2012, pp. 2–2.

VITA

VITA

Chao Pan was born and raised in Shanghai, China. He majored in Finance in college, and received his bachelor degree in 2008 from Shanghai University of Finance and Economics. He then joined Department of Statistics at Purdue University where he received his master's degree in 2010 and expects to receive his Ph.D. in 2016.