CONTRIBUTIONS TO BINARY-OUTPUT COMPUTER EXPERIMENTS AND LARGE-SCALE COMPUTER EXPERIMENTS

A Dissertation
Presented to
The Academic Faculty

By

Chih-Li Sung

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Industrial & Systems Engineering

Georgia Institute of Technology

August 2018

CONTRIBUTIONS TO BINARY-OUTPUT COMPUTER EXPERIMENTS AND LARGE-SCALE COMPUTER EXPERIMENTS

Approved by:

Dr. C. F. Jeff Wu, Advisor
School of Industrial & Systems Engineering
Georgia Institute of Technology

Dr. Benjamin Haaland, Advisor
School of Medicine
University of Utah

Dr. Cheng Zhu
Coulter Department of Biomedical Engineering
Georgia Institute of Technology

Dr. Roshan Vengazhiyil
School of Industrial & Systems Engineering
Georgia Institute of Technology

Dr. Ying Hung
Department of Statistics and Biostatistics
Rutgers University

Date Approved: May 1, 2018

To my beloved parents,

for their unconditional love and support.

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation to all who have influenced, inspired, and am hugely supportive of my work in their various capacities. Without their support, this dissertation would have come to fruition.

First and foremost, I would like to express my deep and sincere gratitude to my advisor, Prof. C. F. Jeff Wu, for his continued guidance, patience, and support in my research endeavors and career development. He is not only an incredible scholar in academia, he is also a dedicated and devoted mentor to his students. His philosophy in academic research and mentorship is the most important lesson I learned in my Ph.D. journey. I am very fortunate and honored to be his student.

Secondly, I owe a debt of gratitude to my co-advisor, Prof. Ben Haaland, for his incredible dedication in advising me. Whenever I have new ideas or struggles in my research, he is always there for me. Without his immense encouragement and patience, I would never be able to accomplish this work.

I would like to thank Prof. Ying Hung for her support in my research and for her career advice. She gave me countless advice that propelled not just my research and career, but also, invaluable life lessons.

I would also like to express my deepest appreciation to my committee members, Prof. Roshan Joseph and Prof. Cheng Zhu, for their generous help, efforts, insightful discussions and suggestions on my doctoral study and dissertation.

I would also like to extend my gratitude to my former advisor, Prof. Shao-Wei Cheng, at National Tsing Hua University. He always believed in me. When I was uncertain about my future, he encouraged me to study abroad and affirmed me that I can and will do well in a research career.

I am very grateful for my lab mates, Simon Mak, Wenjia Wang, David Zhao, Li-Hsiang Lin, Zhehui Chen, Yuan Wang, Li Gu, Dianpeng Wang, Heng Su, as well as all my friends,

who spent time with me and helped me throughout my graduate studies. They made my time in Georgia Institute of Technology, wonderful and memorable, and left an indelible mark in my life.

Last but not least, I would like to express my heartfelt appreciation towards my family for their continuous love and support. I am especially thankful for my wife, Margot Hung, who moved from Taiwan to Atlanta, in support of my graduate studies. She is my greatest inspiration and motivation. This dissertation is dedicated to them.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Computer experiments have played an increasingly important role in science and technology and received enormous attention from industries and research institutes. One prominent example is the redesign of a new rocket engine by the U.S. Air Force[1].

This dissertation makes contributions in two important aspects of computer experiments: (i) binary-output computer experiments and (ii) large-scale computer experiments. For (i), the dissertation contains two chapters: a new emulation method in Chapter 1 and a novel calibration method in Chapter 2, respectively. For (ii), the dissertation contains two chapters, in which new computationally efficient search limiting techniques for local Gaussian process approximation are developed in Chapter 3, and a new model, which is called multi-resolution function ANOVA, is proposed in Chapter 4.

In Chapter 1, we study the emulation problem of computer experiments whose response is binary. Such non-Gaussian observations are common in some computer experiments. Motivated by the analysis of a class of cell adhesion experiments, we introduce a generalized Gaussian process model for binary responses, which shares some common features with standard Gaussian process models. In addition, the proposed model incorporates a flexible mean function that can capture different types of time series structures. Asymptotic properties of the estimators are derived, and an optimal predictor as well as its predictive distribution are constructed. Their performance is examined via two simulation studies. The methodology is applied to study computer simulations for cell adhesion experiments. The fitted model reveals important biological information in repeated cell bindings, which is not directly observable in lab experiments.

In Chapter 2, we develop a calibration method for binary-output computer experiments. Calibration refers to the estimation of unknown parameters which are present in computer experiments but not available in physical experiments. An accurate estimation of these parameters is important because it provides a scientific understanding of the underlying

system which is not available in physical experiments. Most of the work in the literature are limited to the analysis of continuous responses. Motivated by a study of cell adhesion experiments, we propose a new calibration method for binary responses. This method is shown to be semiparametric efficient and the estimated parameters are asymptotically consistent. Numerical examples are given to demonstrate the finite sample performance. The proposed method is applied to analyze a class of T cell adhesion experiments. The findings can shed new light on the settings of kinetic parameters in single molecular interactions which are important in the study of the immune system.

In Chapter 3, we develop two computationally efficient search limiting techniques for local Gaussian process approximation, which can be used in large-scale computer experiments. Gaussian process models are commonly used as emulators for computer experiments. However, developing a Gaussian process emulator can be computationally prohibitive when the number of experimental samples is even moderately large. Local Gaussian process approximation [2] was proposed as an accurate and computationally feasible emulation alternative. However, constructing local sub-designs specific to predictions at a particular location of interest remains a substantial computational bottleneck to the technique. In this chapter, two computationally efficient neighborhood search limiting techniques are proposed, a maximum distance method and a feature approximation method. Two examples demonstrate that the proposed methods indeed save substantial computation while retaining emulation accuracy.

In Chapter 4, we propose a novel model, multi-resolution functional ANOVA, for large-scale and many-input computer experiments that have become typical. More generally, this model can be used for large-scale and many-input non-linear regression problems. An overlapping group lasso approach is used for estimation, ensuring computational feasibility in a large-scale and many-input setting. New results on consistency and inference for the (potentially overlapping) group lasso in a high-dimensional setting are developed and applied to the proposed multi-resolution functional ANOVA model. Importantly, these results allow

us to quantify the uncertainty in our predictions. Numerical examples demonstrate that the proposed model enjoys marked computational advantages. Data capabilities, both in terms of sample size and dimension, meet or exceed best available emulation tools while meeting or exceeding emulation accuracy.

CHAPTER 1
A GENERALIZED GAUSSIAN PROCESS MODEL FOR COMPUTER EXPERIMENTS WITH BINARY TIME SERIES

1.1 Introduction

Cell adhesion plays an important role in many physiological and pathological processes. This research is motivated by the analysis of a class of cell adhesion experiments called micropipette adhesion frequency assays, which is a method for measuring the kinetic rates between molecules in their native membrane environment. In a micropipette adhesion frequency assay, a red blood coated in a specific ligand is brought into contact with cell containing the native receptor for a predetermined duration, then retracted. The output of interest is binary, indicating whether a controlled contact results in adhesion. If there is an adhesion between molecules at the end of contact, retraction will stretch the red cell. If no adhesion resulted, the red cell will not be stretched. The kinetics of the molecular interaction can be derived through many repeated trials. In theory, these contacts should be independent Bernoulli trials. However, there is a memory effect in the repeated tests and the quantification of such a memory effect is scientifically important [3, 4].

A cost-effective way to study the repeated adhesion frequency assays is through computer experiments, which study real systems using complex mathematical models and numerical tools such as finite element analysis [5]. They have been widely used as alternatives to physical experiments or observations, especially for the study of complex systems. For cell adhesion, performing physical experiments (i.e., lab work) is time-consuming and often involves complicated experimental manipulation. Therefore, instead of performing the experiments only based on the actual lab work, computer simulations based on mathematical models are conducted to provide an efficient way to examine the complex mechanisms

behind the adhesion.

The analysis of computer experiments has three objectives: (i) to build a model that captures the nonlinear relationship between inputs and outputs; (ii) to estimate the unknown parameters in the model and deduce properties of the estimators; (iii) to provide an optimal predictor for untried input settings, also called "emulator" or "surrogate model", and quantify its predictive uncertainty [6, 5]. This objective (iii) is crucial because computer simulations are generally expensive or time-consuming to perform and therefore the emulators based on computer simulations are used as surrogates to perform sensitivity analysis, process optimization, calibration, etc. In particular, it is critical for calibration problems in which the emulators and physical experiments are integrated so that some unknown calibration parameters can be estimated. In the literature, Gaussian process (GP) model, use of which achieves the three objectives, is widely used for the analysis of computer experiments. A GP model accommodates nonlinearity using GP and provides an optimal predictor with an interpolation property. The applications of GP can be found in many fields in science and engineering.

The conventional GP models are developed for continuous outputs with a Gaussian assumption, which does not hold in some scientific studies. For example, the focus of the cell adhesion frequency assays is to elicit the relationship between the setting of kinetic parameters/covariates and the adhesion score, which is binary. For binary outputs, the Gaussian assumption is not valid and GP models cannot be directly applied. Binary outputs are common in computer experiments, but the extensions of GP models to non-Gaussian cases have received scant attention in computer experiment literature. Although there are intensive studies of generalized GP models for non-Gaussian data in machine learning and spatial statistics literature, such as [7], [8], [9], [10] and [11], the asymptotic properties of estimators have not been systematically studied. Moreover, an analogy to the GP predictive distribution for binary data is important for uncertainty quantification in computer experiments, which has not yet been developed to the best of our knowledge.

Apart from the non-Gaussian responses, analysis of the repeated cell adhesion frequency assays poses another challenge, namely, how to incorporate a time series structure with complex interaction effects. It was discovered that cells appear to have the ability to remember the previous adhesion events and such a memory has an impact on the future adhesion behaviors [3, 4]. The quantification of the memory effect and how it interacts with the settings of the kinetic parameters in the binary time series are important but cannot be obtained by direct application of the conventional GP models. To consider the time series structure, a common practice is to construct a spatial-temporal model. However, a separable correlation function (e.g., [12, 13]) in which space and time are assumed to be independent is often implemented as a convenient way to address the computational issue. As a result, the estimation of interaction between space and time, which is of major interest here, is not allowed for. Even in the cases where nonseparable correlation functions (e.g., [12, 14]) are implemented, the interaction effect is still not easily interpretable. Therefore, a new model that can model binary time series and capture interaction effects is called for.

To achieve the objectives in the analysis of computer experiments and overcome the aforementioned limitations with binary time series outputs, we introduce a new class of models in this article. The idea is to generalize GP models to non-Gaussian responses and incorporate a flexible mean function that can estimate the time series structure and its interaction with the input variables. In particular, we focus on binary responses and introduce a new model which is analogous to the GP model with an optimal interpolating predictor. Rigorous studies of estimation, prediction, and inference are required for the proposed model and the derivations are complicated by the nature of binary responses and the dependency of time series. Since binary responses with serial correlations can be observed in computer experiments, the proposed method can be readily applicable to other fields beyond cell biology. For example, in manufacturing industry computer simulations are often conducted for the failure analysis where the outputs of interest are binary, i.e., failure or success [15]. Examples can also be found in other biological problems where binary

outputs are observed and evolve in time, such as neuron firing simulations, cell signaling pathways, gene transcription, and recurring diseases [16, 17]. The proposed method can also be broadly applied beyond computer experiments. In many scientific experiments, such as medical research and social studies, binary repeated measurements are commonly observed with serial correlations. In these situations, the proposed method can be implemented to provide a flexible nonlinear model that quantifies the correlation structure and explains the complex relationship between inputs and binary outputs. More examples can be found in functional data analysis, longitudinal data analysis, and machine learning.

The remainder of this article is organized as follows. The new class of models is discussed in Section 1.2. In Section 1.3 and 1.4, asymptotic properties of the estimators are derived and the predictive distributions are constructed. Finite sample performance is demonstrated by simulations in Section 1.5. In Section 1.6, the proposed method is illustrated with the analysis of computer experiments for cell adhesion frequency assays. Concluding remarks are given in Section 1.7. Mathematical proofs and algorithms are provided in Appendix A. An implementation for our method can be found in `binaryGP` [18] in `R` [19].

1.2 Model

1.2.1 Generalized Gaussian process models for binary response

We first introduce a model for binary responses in computer experiments which is analogous to the conventional GP models for continuous outputs. Suppose a computer experiment has a d-dimensional input setting $\mathbf{x} = (x_1, \ldots, x_d)'$ and for each setting the binary output is denoted by $y(\mathbf{x})$ and randomly generated from a Bernoulli distribution with probability $p(\mathbf{x})$. Using a logistic link function, the Gaussian process model for binary data can be written as

$$\text{logit}(p(\mathbf{x})) = \alpha_0 + \mathbf{x}'\boldsymbol{\alpha} + Z(\mathbf{x}), \tag{1.1}$$

where $p(\mathbf{x}) = \mathbb{E}[y(\mathbf{x})]$, α_0 and $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_d)'$ are the intercept and linear effects of the mean function of $p(\mathbf{x})$, and $Z(\cdot)$ is a zero mean Gaussian process with variance σ^2, correlation function $R_{\boldsymbol{\theta}}(\cdot, \cdot)$, and unknown correlation parameters $\boldsymbol{\theta}$.

Various choices of correlation functions have been discussed in the literature. For example, the *power exponential correlation function* is commonly used in the analysis of computer experiments [5]:

$$R_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left\{ -\sum_{l=1}^{d} \frac{(x_{il} - x_{jl})^p}{\theta_l} \right\}, \tag{1.2}$$

where $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_d)$, the power p controls the smoothness of the output surface, and the parameter θ_l controls the decay of correlation with respect to the distance between x_{il} and x_{jl}. Recent studies have shown that a careful selection of the correlation function, such as orthogonal Gaussian processes proposed by [20], can resolve the identifiability issue in the estimation of Gaussian process models [21, 22, 23]. This is particularly important in the application of calibration problems where the parameter estimation plays a significant role. Depending on the objectives of the studies, different correlation functions can be incorporated into the proposed model and the theoretical results developed herein remain valid.

Similar extensions of GP models to binary outputs have been applied in many different fields. For example when $\mathbf{x}$ represents a two-dimensional spatial domain, (1.1) becomes the spatial generalized linear mixed model proposed by [8]. In a Bayesian framework, Gaussian process priors are implemented for classification problems, such as in [7] and [24]. Despite successful applications of these models, theoretical studies on the estimation and prediction properties are not available. Therefore, one focus of this chapter is to provide theoretical supports for the estimation and prediction in (1.1).

1.2.2 Generalized Gaussian process models for binary time series

In this section, we introduce a new model for the analysis of computer experiments with binary time series, which is an extension of (1.1) that takes serial correlations between binary observations into account. Suppose for each setting of a computer experiment, a sequence of *binary time series* outputs $\{y_t(\mathbf{x})\}_{t=1}^T$ is randomly generated from Bernoulli distributions with probabilities $\{p_t(\mathbf{x})\}_{t=1}^T$. A generalized Gaussian process model for binary time series can be written as:

$$\text{logit}(p_t(\mathbf{x})) = \eta_t(\mathbf{x}) = \sum_{r=1}^{R} \varphi_r y_{t-r}(\mathbf{x}) + \alpha_0 + \mathbf{x}'\boldsymbol{\alpha} + \sum_{l=1}^{L} \boldsymbol{\gamma}_l \mathbf{x} y_{t-l}(\mathbf{x}) + Z_t(\mathbf{x}), \qquad (1.3)$$

where $p_t(\mathbf{x}) = \mathbb{E}[y_t(\mathbf{x})|H_t]$ is the conditional mean given the previous information $H_t = \{y_{t-1}(\mathbf{x}), y_{t-2}(\mathbf{x}), \ldots\}$. In model (1.3), $\{\varphi_r\}_{r=1}^R$ represents an autoregressive (AR) process with order R and $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_d)'$ represents the effects of $\mathbf{x}$. The d-dimensional vector $\boldsymbol{\gamma}_l$ represents the interaction between the input and the past outputs and provides the flexibility of modeling different time series structures with different inputs. Given that the interactions between $\mathbf{x}$ and time are captured by $\mathbf{x}y_{t-l}$, Z_t is assumed to vary independently over time to reduce modeling and computational complexity. Further extensions can be made by replacing $Z_t(\mathbf{x})$ with a spatio-temporal Gaussian process $Z(t, \mathbf{x})$, but the computational cost will be higher. Without the Gaussian process assumption in (1.3), the mean function is closely related to the Zeger-Qaqish model [25] and its extensions in [4] and [26], all of which take into account the autoregressive predictors in logistic regression.

Model (1.3) extends the applications of conventional GP to binary time series generated from computer experiments. The model is intuitively appealing; however, the issues of estimation, prediction, and inference are not straightforward due to the nature of binary response and the dependency structure.

1.3 Inference

Since model (1.1) can be written as a special case of model (1.3) when $R = 0, L = 0$ and $T = 1$, derivations herein are mainly based on model (1.3) with additional discussions given for (1.1) when necessary.

1.3.1 Estimation

Given n input settings $\mathbf{x}_1, \ldots, \mathbf{x}_n$ in a computer experiment, denote $y_{it} \equiv y_t(\mathbf{x}_i)$ as the binary output generated from input $\mathbf{x}_i$ at time t, where $\mathbf{x}_i \in \mathbb{R}^d$, $i = 1, ..., n$, and $t = 1, ..., T$. Let N be the total number of the outputs, i.e., $N = nT$. In addition, at each time t, denote $\mathbf{y}_t$ as an n-dimensional vector $\mathbf{y}_t = (y_{1t}, ..., y_{nt})'$ with conditional mean $\boldsymbol{p}_t = (p_{1t}, \ldots, p_{nt})'$, where $p_{it} = \mathbb{E}(y_{it}|H_{it})$ and $H_{it} = \{y_{i,t-1}, y_{i,t-2}, \ldots\}$. Based on the data, model (1.3) can be rewritten into matrix form as follows:

$$\text{logit}(\boldsymbol{p}) = \boldsymbol{X}'\boldsymbol{\beta} + \boldsymbol{Z}, \quad \boldsymbol{Z} \sim \mathcal{N}(\mathbf{0}_N, \Sigma(\boldsymbol{\omega})), \tag{1.4}$$

where $\boldsymbol{p} = (\boldsymbol{p}_1', \ldots, \boldsymbol{p}_T')', \boldsymbol{\beta} = (\varphi_1, \ldots, \varphi_R, \alpha_0, \boldsymbol{\alpha}', (\boldsymbol{\gamma}_1', \ldots, \boldsymbol{\gamma}_L')')', \boldsymbol{\omega} = (\sigma^2, \boldsymbol{\theta})'$, $\boldsymbol{Z} = (Z_1(\mathbf{x}_1), \ldots, Z_1(\mathbf{x}_n), \ldots, Z_T(\mathbf{x}_1), \ldots, Z_T(\mathbf{x}_n))'$, $\boldsymbol{X}$ is the model matrix $(X_1', \ldots, X_T')'$, X_t is an $n \times (1+R+d+dL)$ matrix with i-th row defined by $(X_t)_i = (1, y_{i,t-1}, \ldots, y_{i,t-R}, \mathbf{x}_i',$ $\mathbf{x}_i' y_{i,t-1}, \ldots, \mathbf{x}_i' y_{i,t-L})$, and $\Sigma(\boldsymbol{\omega})$ is an $N \times N$ covariance matrix defined by

$$\Sigma(\boldsymbol{\omega}) = \sigma^2 \boldsymbol{R}_{\boldsymbol{\theta}} \otimes I_T \tag{1.5}$$

with $(\boldsymbol{R}_{\boldsymbol{\theta}})_{ij} = R_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{x}_j)$. Model (1.1) can also be rewritten in the same way by setting $R = 0, L = 0$ and $T = 1$.

With the presence of time series and their interaction with the input settings in model (1.3), we can write down the partial likelihood (PL) function [27, 28] according to the formulation of [29]. Given the previous information $\{H_{it}\}_{i=1,\ldots,n;t=1,\ldots,N}$, the PL for $\boldsymbol{\beta}$ can

be written as

$$PL(\boldsymbol{\beta}|\boldsymbol{Z}) = \prod_{i=1}^{n}\prod_{t=1}^{T}(p_{it}(\boldsymbol{\beta}|\boldsymbol{Z}))^{y_{it}}(1-p_{it}(\boldsymbol{\beta}|\boldsymbol{Z}))^{1-y_{it}}, \tag{1.6}$$

where $p_{it}(\boldsymbol{\beta}|\boldsymbol{Z}) = \mathbb{E}_{\boldsymbol{\beta}|\boldsymbol{Z}}[y_{it}|H_{it}]$. Then, the integrated quasi-PL function for the estimation of $(\boldsymbol{\beta}, \boldsymbol{\omega})$ is given by

$$|\Sigma(\boldsymbol{\omega})|^{-1/2}\int \exp\{\log PL(\boldsymbol{\beta}|\boldsymbol{Z}) - \frac{1}{2}\boldsymbol{Z}'\Sigma(\boldsymbol{\omega})^{-1}\boldsymbol{Z}\}d\boldsymbol{Z}. \tag{1.7}$$

Note that, for model (1.1) where no time series effect is considered, (1.6) and (1.7) should be replaced by the likelihood function

$$L(\boldsymbol{\beta}|\boldsymbol{Z}) = \prod_{i=1}^{n}(p_{i1}(\boldsymbol{\beta}|\boldsymbol{Z}))^{y_{i1}}(1-p_{i1}(\boldsymbol{\beta}|\boldsymbol{Z}))^{1-y_{i1}}$$

and the integrated quasi-likelihood function

$$|\Sigma(\boldsymbol{\omega})|^{-1/2}\int \exp\{\log L(\boldsymbol{\beta}|\boldsymbol{Z}) - \frac{1}{2}\boldsymbol{Z}'\Sigma(\boldsymbol{\omega})^{-1}\boldsymbol{Z}\}d\boldsymbol{Z}, \tag{1.8}$$

respectively. Hereafter, we provide the framework for the integrated quasi-PL function (1.7), but the result can be applied to the integrated quasi-likelihood function (1.8) by assuming $R = 0, L = 0$ and $T = 1$.

Because of the difficulty in computing the integrated quasi-PL function, a *penalized quasi-PL* (PQPL) function is used as an approximation. Similar to the procedure in [30], the integrated quasi-partial log-likelihood can be approximated by Laplace's method [31].

Ignoring the multiplicative constant and plugging (1.5) in $\Sigma(\boldsymbol{\omega})$, the approximation yields

$$-\frac{1}{2}\log|I_n+\sigma^2\boldsymbol{W}(\boldsymbol{R_\theta}\otimes I_T)|+ \sum_{i=1}^{n}\sum_{t=1}^{T}\left(y_{it}\log\frac{p_{it}(\boldsymbol{\beta}|\tilde{\boldsymbol{Z}})}{1-p_{it}(\boldsymbol{\beta}|\tilde{\boldsymbol{Z}})}+\log(1-p_{it}(\boldsymbol{\beta}|\tilde{\boldsymbol{Z}}))\right)-\frac{1}{2\sigma^2}\tilde{\boldsymbol{Z}}'(\boldsymbol{R_\theta}\otimes I_T)^{-1}\tilde{\boldsymbol{Z}}, \tag{1.9}$$

where $\boldsymbol{W}$ is an $N\times N$ diagonal matrix with diagonal elements $W_{it}=p_{it}(\boldsymbol{\beta}|\tilde{\boldsymbol{Z}})(1-p_{it}(\boldsymbol{\beta}|\tilde{\boldsymbol{Z}}))$, $p_{it}(\boldsymbol{\beta}|\tilde{\boldsymbol{Z}})=\mathbb{E}_{\boldsymbol{\beta}|\tilde{\boldsymbol{Z}}}[y_{it}|H_{it}]$, and $\tilde{\boldsymbol{Z}}=\tilde{\boldsymbol{Z}}(\boldsymbol{\beta},\boldsymbol{\omega})$ is the solution of $\sum_{i=1}^{n}\sum_{t=1}^{T}\mathbf{e}_{it}(y_{it}-p_{it}(\boldsymbol{\beta}|\boldsymbol{Z}))=(\boldsymbol{R_\theta}\otimes I_T)^{-1}\boldsymbol{Z}/\sigma^2$, where $\mathbf{e}_{it}$ is a unit-vector where $((t-1)n+i)$-th element is one. The estimator $\hat{\boldsymbol{\beta}}$ which maximizes the PQPL function (1.9) is called *maximum quasi-PL estimator*. Thus, similar to the derivations in [30] for score equations of a penalized quasi-likelihood function, the score equations of the PQPL function for $\boldsymbol{\beta}$ and $\boldsymbol{\omega}$ are

$$\sum_{i=1}^{n}\sum_{t=1}^{T}X_{it}(y_{it}-p_{it}(\boldsymbol{\beta},\boldsymbol{\omega}))=0$$

and

$$\sum_{i=1}^{n}\sum_{t=1}^{T}\mathbf{e}_{it}(y_{it}-p_{it}(\boldsymbol{\beta},\boldsymbol{\omega}))=(\boldsymbol{R_\theta}\otimes I_T)^{-1}\boldsymbol{Z}/\sigma^2,$$

where $p_{it}(\boldsymbol{\beta},\boldsymbol{\omega})=\mathbb{E}_{\boldsymbol{\beta},\boldsymbol{\omega}}[y_{it}|H_{it}]$. The solution to the score equations can be efficiently obtained by an iterated weighted least squares (IWLS) approach as follows. In each step, one first solves for $\boldsymbol{\beta}$ in

$$(\boldsymbol{X}'\boldsymbol{V}(\boldsymbol{\omega})^{-1}\boldsymbol{X})\boldsymbol{\beta}=\boldsymbol{X}'\boldsymbol{V}(\boldsymbol{\omega})^{-1}\tilde{\boldsymbol{\eta}}, \tag{1.10}$$

where $\boldsymbol{V}(\boldsymbol{\omega})=\boldsymbol{W}^{-1}+\sigma^2(\boldsymbol{R_\theta}\otimes I_T)$, $\boldsymbol{W}$ is an $N\times N$ diagonal matrix with diagonal elements $W_{it}=p_{it}(\boldsymbol{\beta},\boldsymbol{\omega})(1-p_{it}(\boldsymbol{\beta},\boldsymbol{\omega}))$, and $\tilde{\eta}_{it}=\log\frac{p_{it}(\boldsymbol{\beta},\boldsymbol{\omega})}{1-p_{it}(\boldsymbol{\beta},\boldsymbol{\omega})}+\frac{y_{it}-p_{it}(\boldsymbol{\beta},\boldsymbol{\omega})}{p_{it}(\boldsymbol{\beta},\boldsymbol{\omega})(1-p_{it}(\boldsymbol{\beta},\boldsymbol{\omega}))}$, and then sets

$$\hat{\boldsymbol{Z}}=\sigma^2(\boldsymbol{R_\theta}\otimes I_T)\boldsymbol{V}(\boldsymbol{\omega})^{-1}(\tilde{\boldsymbol{\eta}}-\boldsymbol{X}'\hat{\boldsymbol{\beta}}) \tag{1.11}$$

and replaces $p_{it}(\boldsymbol{\beta}, \boldsymbol{\omega})$ with $p_{it}(\hat{\boldsymbol{\beta}}, \boldsymbol{\omega}) = \left(\frac{\exp\{\boldsymbol{X}'\hat{\boldsymbol{\beta}} + \hat{\boldsymbol{Z}}\}}{\mathbf{1}_N + \exp\{\boldsymbol{X}'\hat{\boldsymbol{\beta}} + \hat{\boldsymbol{Z}}\}} \right)_{it}$.

Estimation of the correlation parameters $\boldsymbol{\theta}$ and variance σ^2 is obtained by the restricted maximum likelihood (REML) approach [32] because it is known to have smaller bias comparing with the maximum likelihood approach [33]. See also [34] and [35] for details. According to [36, 34], the REML estimators of σ^2 and $\boldsymbol{\theta}$ can be solved by minimizing the following negative log-likelihood function with respect to $\boldsymbol{\omega}$,

$$L(\boldsymbol{\omega}) = \frac{N-m}{2}\log(2\pi) - \frac{1}{2}\log(|\boldsymbol{X}'\boldsymbol{X}|) + \frac{1}{2}\log(|\boldsymbol{V}(\boldsymbol{\omega})|) + \frac{1}{2}\log(|\boldsymbol{X}'\boldsymbol{V}(\boldsymbol{\omega})^{-1}\boldsymbol{X}|) + \frac{1}{2}\tilde{\boldsymbol{\eta}}'\Pi(\boldsymbol{\omega})\tilde{\boldsymbol{\eta}}, \tag{1.12}$$

where $m = 1+R+d+dL$ and $\Pi(\boldsymbol{\omega}) = \boldsymbol{V}(\boldsymbol{\omega})^{-1} - \boldsymbol{V}(\boldsymbol{\omega})^{-1}\boldsymbol{X}(\boldsymbol{X}'\boldsymbol{V}(\boldsymbol{\omega})^{-1}\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{V}(\boldsymbol{\omega})^{-1}$.

Therefore, the estimators $\hat{\boldsymbol{\beta}}$ and $\hat{\boldsymbol{\omega}}$ ($\equiv (\hat{\sigma}^2, \hat{\boldsymbol{\theta}})'$) can be obtained by iteratively solving (1.10), (1.11) and minimizing (1.12). The explicit algorithm is given in Appendix A.1. Note that $\boldsymbol{V}(\boldsymbol{\omega})$ is a block diagonal matrix, i.e., a square matrix having main diagonal blocks square matrices such that the off-diagonal blocks are zero matrices. Therefore the computational burden for the matrix inversion of $\boldsymbol{V}(\boldsymbol{\omega})$ can be alleviated by the fact that the inverse of a block diagonal matrix is a block diagonal matrix, composed of the inversion of each block.

1.3.2 Asymptotic Properties

Asymptotic results are presented here to show that the estimators $\hat{\boldsymbol{\beta}}, \hat{\sigma}^2$ and $\hat{\boldsymbol{\theta}}$ obtained in Section 1.3.1 are asymptotically normally distributed when $N(= nT)$ becomes sufficiently large. In the present context both n and T are sufficiently large. The assumptions are given in Appendix A.2, and the proofs are stated in Appendix A.3 and A.4. These results are developed along the lines described in [4] and [37, 38].

Theorem 1.3.1. *Under assumptions A.2.1 and A.2.2, the maximum quasi-PL estimator for*

the fixed effects $\boldsymbol{\beta}$ are consistent and asymptotically normal as $N \to \infty$,

$$\sqrt{N}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}) = \Lambda_N^{-1} \frac{1}{\sqrt{N}} S_N(\boldsymbol{\beta}, \boldsymbol{\omega}) + o_p(1)$$

and

$$\sqrt{N}\Lambda_N^{1/2}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}) \xrightarrow{d} \mathcal{N}(\mathbf{0}, I_m),$$

where m is the size of the vector $\boldsymbol{\beta}$ (i.e., $m = 1 + R + d + dL$), the sample information matrix

$$\Lambda_N = \frac{1}{N} \sum_{i=1}^{n} \sum_{t=1}^{T} X_{it} X_{it}' p_{it}(\boldsymbol{\beta}, \boldsymbol{\omega})(1 - p_{it}(\boldsymbol{\beta}, \boldsymbol{\omega})),$$

and $S_N(\boldsymbol{\beta}, \boldsymbol{\omega}) = \sum_{i=1}^{n} \sum_{t=1}^{T} X_{it}(y_{it} - p_{it}(\boldsymbol{\beta}, \boldsymbol{\omega}))$.

Remark 1.3.2. For model (1.1), the estimator $\hat{\boldsymbol{\beta}}$ can be obtained by minimizing the penalized quasi-likelihood (PQL) function, which can be written as (1.9) with $T = 1$. Under assumption A.2.1 and the application of central limit theorem, such estimator has the same asymptotic properties as in Theorem 1.3.1 with $N = n$.

For models (1.1) and (1.3), we have the following asymptotic properties for $\hat{\boldsymbol{\omega}}$.

Theorem 1.3.3. *Denote $[\Gamma_N(\boldsymbol{\omega})]_{i,j} = \partial^2 L(\boldsymbol{\omega}) / \partial\omega_i \partial\omega_j$ and $J_N(\boldsymbol{\omega}) = [\mathbb{E}_{\boldsymbol{\omega}} \Gamma_N(\boldsymbol{\omega})]^{1/2}$. Then, under assumptions A.2.3 and A.2.4, as $N \to \infty$,*

$$J_N(\hat{\boldsymbol{\omega}})(\hat{\boldsymbol{\omega}} - \boldsymbol{\omega}) \xrightarrow{d} \mathcal{N}(\mathbf{0}, I_{d+1}).$$

1.4 Construction of Predictive Distribution

For computer experiments, the construction of an optimal predictor and its corresponding predictive distribution is important for uncertainty quantification, sensitivity analysis, process optimization, and calibration [5].

First, some notation is introduced. For some untried setting $\mathbf{x}_{n+1}$, denote the predictive probability at time s by $p_s(\mathbf{x}_{n+1}) = \mathbb{E}[y_s(\mathbf{x}_{n+1})|H_s]$, where $H_s = \{y_{n+1,s-1}, y_{n+1,s-2}, \ldots\}$.

Assume that $D_{n+1,s}$ represents the "previous information" including $\{y_{n+1,s-1}, y_{n+1,s-2}, \ldots, p_{n+1,s-1}, p_{n+1,s-2}, \ldots\}$ at $\mathbf{x}_{n+1}$ and $\{y_{it}, p_{it}\}$, where $i = 1, \ldots, n$ and $t = 1, \ldots, T$. Also, let $Logitnormal(\mu, \sigma^2)$ represent a logit-normal distribution P, where $P = \exp\{X\}/(1 + \exp\{X\})$ and X has a univariate normal distribution with μ and variance σ^2. Denote the first two moments of the distribution by $\mathbb{E}[P] = \kappa(\mu, \sigma^2)$ and $\mathbb{V}[P] = \tau(\mu, \sigma^2)$. In general, there is no closed form expression for $\kappa(\mu, \sigma^2)$ and $\tau(\mu, \sigma^2)$, but it can be easily computed by numerical integration such as in the package `logitnorm` [39] in `R` [19]. More discussions on logit-normal distribution can be found in [40, 41, 42].

We first present a lemma which shows that, given $D_{n+1,s}$, the conditional distribution of $p_s(\mathbf{x}_{n+1})$ in model (1.3) is logit-normal. This result lays the foundation for the construction of predictive distribution. The proof is given in Appendix A.5.

Lemma 1.4.1. *For model* (1.3)*, the conditional distribution of* $p_s(\mathbf{x}_{n+1})$ *can be written as*

$$p_s(\mathbf{x}_{n+1})|D_{n+1,s} \sim Logitnormal(m(D_{n+1,s}), v(D_{n+1,s})),$$

where

$$m(D_{n+1,s}) = \sum_{r=1}^{R} \varphi_r y_{n+1,s-r} + \alpha_0 + \mathbf{x}'_{n+1}\boldsymbol{\alpha} + \sum_{l=1}^{L} \boldsymbol{\gamma}_l \mathbf{x}_{n+1} y_{n+1,s-l} + \boldsymbol{r}'_{\boldsymbol{\theta}} \boldsymbol{R}_{\boldsymbol{\theta}}^{-1} \left(\log \frac{\boldsymbol{p}_s}{\mathbf{1}_n - \boldsymbol{p}_s} - \boldsymbol{\mu}_s \right),$$

$v(D_{n+1,s}) = \sigma^2 \left(1 - \boldsymbol{r}'_{\boldsymbol{\theta}} \boldsymbol{R}_{\boldsymbol{\theta}}^{-1} \boldsymbol{r}_{\boldsymbol{\theta}}\right)$, $\boldsymbol{r}_{\boldsymbol{\theta}} = (R_{\boldsymbol{\theta}}(\mathbf{x}_{n+1}, \mathbf{x}_1), \ldots, R_{\boldsymbol{\theta}}(\mathbf{x}_{n+1}, \mathbf{x}_n))'$, $\boldsymbol{R}_{\boldsymbol{\theta}} = \{R_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{x}_j)\}$, $\boldsymbol{p}_s = (p_s(\mathbf{x}_1), \ldots, p_s(\mathbf{x}_n))'$*, and* $(\boldsymbol{\mu}_s)_i = \sum_{r=1}^{R} \varphi_r y_{i,s-r} + \alpha_0 + \mathbf{x}'_i\boldsymbol{\alpha} + \sum_{l=1}^{L} \boldsymbol{\gamma}_l \mathbf{x}_i y_{i,s-l}$.

Remark 1.4.2. For model (1.1), the result in Lemma 1.4.1 can be applied by having $R = 0, L = 0, s = 1$ and $T = 1$. Then, $D_{n+1,s}$ can be written as D_{n+1} containing only $\{p_{1,1}, \ldots, p_{n,1}\}$, and we have the conditional distribution

$$p(\mathbf{x}_{n+1})|D_{n+1} \sim Logitnormal(m(D_{n+1}), v(D_{n+1})),$$

where $m(D_{n+1}) = \alpha_0 + \mathbf{x}'_{n+1}\boldsymbol{\alpha} + \boldsymbol{r}'_{\boldsymbol{\theta}} \boldsymbol{R}_{\boldsymbol{\theta}}^{-1} (\log \frac{\boldsymbol{p}_1}{\mathbf{1} - \boldsymbol{p}_1} - \boldsymbol{\mu}^n)$, $v(D_{n+1}) = \sigma^2(1 - \boldsymbol{r}'_{\boldsymbol{\theta}} \boldsymbol{R}_{\boldsymbol{\theta}}^{-1} \boldsymbol{r}_{\boldsymbol{\theta}})$,

$\boldsymbol{\mu}^n = (\alpha_0 + \mathbf{x}_1'\boldsymbol{\alpha}, \ldots, \alpha_0 + \mathbf{x}_n'\boldsymbol{\alpha})'$, $\boldsymbol{r}_{\boldsymbol{\theta}} = (R_{\boldsymbol{\theta}}(\mathbf{x}_{n+1}, \mathbf{x}_1), \ldots, R_{\boldsymbol{\theta}}(\mathbf{x}_{n+1}, \mathbf{x}_n))'$, and $\boldsymbol{R}_{\boldsymbol{\theta}} = \{R_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{x}_j)\}$.

Based on Lemma 1.4.1, the prediction of $p_s(\mathbf{x}_{n+1})$ for some untried setting $\mathbf{x}_{n+1}$ and its variance can then be obtained in the next theorem. The proof is given in Appendix A.6. The definition of minimum mean squared prediction error of p given D is first stated as follows,

$$\hat{p} = \hat{p}(D) = \arg\min_{\eta} \mathbb{E}_F[(p - \eta)^2],$$

where $F(\cdot)$ is the joint distribution of (p, D) and $\mathbb{E}_F[\cdot]$ denotes expectation under distribution $F(\cdot)$.

Theorem 1.4.3. *Given* $D_{n+1,s} = \{\boldsymbol{y}_1', \ldots, \boldsymbol{y}_T', \boldsymbol{p}_1', \ldots, \boldsymbol{p}_T', y_{n+1,s-1}, \ldots, y_{n+1,1}, p_{n+1,s-1}, \ldots, p_{n+1,1}\}$,

(i) the minimum mean squared prediction error (MMSPE) predictor of $p_s(\mathbf{x}_{n+1})$*, denoted by* $\hat{p}_s(\mathbf{x}_{n+1})$*, is*

$$\begin{aligned} &\mathbb{E}\left[p_s(\mathbf{x}_{n+1})|D_{n+1,s}\right] = \kappa(m(D_{n+1,s}), v(D_{n+1,s})) \\ \textit{with variance} \quad &\mathbb{V}\left[p_s(\mathbf{x}_{n+1})|D_{n+1,s}\right] = \tau(m(D_{n+1,s}), v(D_{n+1,s})); \end{aligned}$$

(ii) the MMSPE predictor is an interpolator, i.e., if $\mathbf{x}_{n+1} = \mathbf{x}_i$ *for* $i = 1, \cdots, n$*, then* $\hat{p}_s(\mathbf{x}_{n+1}) = \mathbb{E}\left[p_s(\mathbf{x}_{n+1})|D_{n+1,s}\right] = p_s(\mathbf{x}_i)$ *and the predictive variance is 0;*

(iii) the q*-th quantile of the conditional distribution* $p(\mathbf{x}_{n+1})|D_{n+1,s}$ *is*

$$\frac{\exp\{m(D_{n+1,s}) + z_q\sqrt{v(D_{n+1,s})}\}}{1 + \exp\{m(D_{n+1,s}) + z_q\sqrt{v(D_{n+1,s})}\}},$$

where z_q *is the* q*-th quantile of the standard normal distribution.*

Theorem 1.4.3 shows that, given $D_{n+1,s}$, the new predictor for binary data can interpolate the underlying probabilities which generate the training data. According to Theorem

1.4.3(iii) and the fact that $v(D_{n+1,s})$ increases with the distance to the training data, this result shows an increasing predictive uncertainty for points away from the training data. This predictive property is desirable and consistent with the conventional GP predictor.

In practice, only the binary outputs are observable and the underlying probabilities are not available in the training data. Thus, the following results construct the MMSPE predictor of $p_s(\mathbf{x}_{n+1})$ given $\boldsymbol{Y} = (\boldsymbol{y}_1', \ldots, \boldsymbol{y}_T', y_1(\mathbf{x}_{n+1}), \ldots, y_{s-1}(\mathbf{x}_{n+1}))'$. These results can be used for prediction and quantification of the predictive uncertainty, such as constructing predictive confidence intervals for untried settings.

Corollary 1.4.4. *Given* $\boldsymbol{Y} = (\boldsymbol{y}_1', \ldots, \boldsymbol{y}_T', y_1(\mathbf{x}_{n+1}), \ldots, y_{s-1}(\mathbf{x}_{n+1}))'$,

(i) The MMSPE predictor of $p_s(\mathbf{x}_{n+1})$ *is*

$$\hat{p}_s(\mathbf{x}_{n+1}) = \mathbb{E}\left[p_s(\mathbf{x}_{n+1})|\boldsymbol{Y}\right] = \mathbb{E}_{\boldsymbol{p}|\boldsymbol{Y}}\left[\kappa(m(D_{n+1,s}), v(D_{n+1,s}))|\boldsymbol{Y}\right] \tag{1.13}$$

with variance

$$\mathbb{V}\left[p_s(\mathbf{x}_{n+1})|\boldsymbol{Y}\right] = \mathbb{E}_{\boldsymbol{p}|\boldsymbol{Y}}\left[\tau(m(D_{n+1,s}), v(D_{n+1,s}))|\boldsymbol{Y}\right] + \mathbb{V}_{\boldsymbol{p}|\boldsymbol{Y}}\left[\kappa(m(D_{n+1,s}), v(D_{n+1,s}))|\boldsymbol{Y}\right], \tag{1.14}$$

where $\boldsymbol{p} = (\boldsymbol{p}_1', \cdots, \boldsymbol{p}_T', p_1(\mathbf{x}_{n+1}), \ldots, p_{s-1}(\mathbf{x}_{n+1}))'$.

(ii) When $\mathbf{x}_{n+1} = \mathbf{x}_i$, *the MMSPE predictor becomes* $\hat{p}_s(\mathbf{x}_i) = \mathbb{E}_{\boldsymbol{p}|\boldsymbol{Y}}\left[p_s(\mathbf{x}_i)|\boldsymbol{Y}\right]$ *with variance* $\mathbb{V}_{\boldsymbol{p}|\boldsymbol{Y}}\left[p_s(\mathbf{x}_i)|\boldsymbol{Y}\right]$.

Remark 1.4.5. For model (1.1), the results of Theorem 1.4.3 and Corollary 1.4.4 can be applied by assuming $s = 1$ and $T = 1$.

Without the information of the underlying probabilities, the predictor does not interpolate all the training data as in Theorem 1.4.3 (ii). From Corollary 1.4.4, when $\mathbf{x}_{n+1} = \mathbf{x}_i$, the predictor is still unbiased but the corresponding variance is nonzero. Instead, the variance becomes $\mathbb{V}_{\boldsymbol{p}|\boldsymbol{Y}}\left[p_s(\mathbf{x}_i)|\boldsymbol{Y}\right]$, which is due to the uncertainty of the underlying probability.

The proof is similar to Theorem 1.4.3 (ii). To show the empirical performance of the predictive distribution in Corollary 1.4.4, a one-dimensional example is illustrated in Figure 1.1. Consider the true probability function, $p(x) = 0.4\exp(-1.2x)\cos(3.5\pi x) + 0.4$, which is represented by a black dotted line, and the training set that contains 12 evenly-spaced inputs and the corresponding binary outputs represented by red dots. The blue line is the MMSPE predictor constructed by equation (1.13) and the gray region is the corresponding 95% confidence band constructed by the 2.5%- and 97.5%-quantiles. It appears that the proposed predictor and the confidence band reasonably capture the underlying probability.

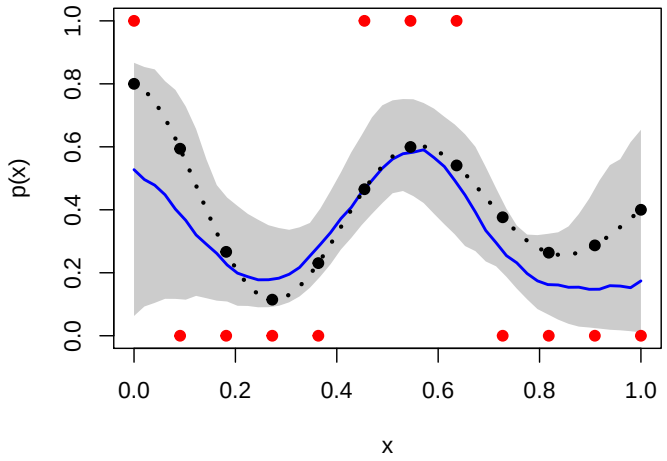


Figure 1.1: Illustration of predictive distribution. Black dotted line represents the true probability function, red dots represent the binary response data, black dots represent the true probabilities at the chosen locations, and the emulator is represented by the blue line, with the gray shaded region providing a pointwise 95% confidence band.

Although there is no closed form expression for the distribution of $\boldsymbol{p}|\boldsymbol{Y}$, the random samples from $\boldsymbol{p}|\boldsymbol{Y}$ can be easily generated by the Metropolis-Hastings (MH) algorithm. See Appendix A.7 for the explicit algorithm. Based on these samples, the expectation and variance in Corollary 1.4.4 can be approximated by using a Monte Carlo method. For example, let $\{\boldsymbol{p}^{(j)}\}_{j=1,\ldots,J}$ be the J random samples generated from distribution $\boldsymbol{p}|\boldsymbol{Y}$, then the MMSPE predictor of $p_s(\mathbf{x}_{n+1})$ in Corollary 1.4.4 can be approximated by

$$\mathbb{E}_{\boldsymbol{p}|\boldsymbol{Y}}\left[\kappa(m(D_{n+1,s}), v(D_{n+1,s}))|\boldsymbol{Y}\right] \approx \frac{1}{J}\sum_{j=1}^{J}\kappa(m(D_{n+1,s}^{(j)}), v(D_{n+1,s}^{(j)})),$$

where $D_{n+1,s}^{(j)} = \{\boldsymbol{p}^{(j)}, \boldsymbol{Y}\}$. Similar idea can be applied to compute $\mathbb{V}[p_s(\mathbf{x}_{n+1})|\boldsymbol{Y}]$. The quantiles of $p_s(\mathbf{x}_{n+1})|\boldsymbol{Y}$ can be obtained from random samples $\{p_s^{(1)}, \ldots, p_s^{(J)}\}$, where $p_s^{(j)}$ is generated from $p_s(\mathbf{x}_{n+1})|D_{n+1,s}^{(j)}$ following a logit-normal distribution by Lemma 1.4.1.

When the historical time series for an untried setting (i.e., $y_1(\mathbf{x}_{n+1}), \ldots, y_{s-1}(\mathbf{x}_{n+1})$ in Corollary 1.4.4) is not available, we can emulate a completely new time series (or batch of time series) with input $\mathbf{x}_{n+1}$. The idea is to generate draws from the conditional distribution $p_s(\mathbf{x}_{n+1})|\boldsymbol{Y}$ for future outputs, starting from $s = 0$, and take pointwise median of the random draws. This idea is similar to the dynamic emulators introduced by [43] for continuous outputs. The random samples from $p_s(\mathbf{x}_{n+1})|\boldsymbol{Y}$ can be generated by the fact $f(p_s(\mathbf{x}_{n+1}), \boldsymbol{p}|\boldsymbol{Y}) = f(\boldsymbol{p}|\boldsymbol{Y})f(p_s(\mathbf{x}_{n+1})|\boldsymbol{p}, \boldsymbol{Y})$, where $f(p_s(\mathbf{x}_{n+1})|\boldsymbol{p}, \boldsymbol{Y})$ is a logit-normal distribution provided in Lemma 1.4.1. As mentioned above, the random samples from $f(\boldsymbol{p}|\boldsymbol{Y})$ can be generated through the MH algorithm. Therefore, generating a draw from $p_s(\mathbf{x}_{n+1})|\boldsymbol{Y}$ consists of two steps: (i) generating the "previous" probability values $\boldsymbol{p}^*$ given output $\boldsymbol{Y}$ from the distribution $\boldsymbol{p}|\boldsymbol{Y}$ through the MH algorithm, and (ii) based on the sample $\boldsymbol{p}^*$, draw a sample $p_s^*(\mathbf{x}_{n+1})$ from $p_s(\mathbf{x}_{n+1})|\boldsymbol{p}^*, \boldsymbol{Y}$, which is a logit-normal distribution, and also draw a sample $y_s^*(\mathbf{x}_{n+1})$ from a Bernoulli distribution with parameter $p_s^*(\mathbf{x}_{n+1})$. An explicit algorithm is given in Appendix A.8.

1.5 Simulation Studies

In Section 1.5.1, we conduct simulations generated from Gaussian processes to demonstrate the estimation performance. In Section 1.5.2, the prediction performance is examined by comparing several existing methods using the data generated from a modified Friedman function [44].

1.5.1 Estimation Performance

Consider a 5-dimensional input space, $d = 5$, and the input $\mathbf{x}$ is randomly generated from a regular grid on $[0, 1]^5$. The binary output, $y_t(\mathbf{x})$ at time t, is simulated by a Bernoulli dis-

tribution with probability $p_t(\mathbf{x})$ calculated by (1.3) and $\alpha_0 = 0.5$, $\boldsymbol{\alpha} = (-3, 2, -2, 1, 0.5)'$, $\varphi_1 = 0.8$, $\sigma^2 = 1$, and the power exponential correlation function (1.2) is chosen with $\boldsymbol{\theta} = (0.5, 1.0, 1.5, 2.0, 2.5)'$ and $p = 2$. Four sample size combinations of n and T are considered in the simulations.

The potential confounding between the polynomials in the mean function and the zero-mean Gaussian process can lead to the lack of identifiability, which will cause the estimated mean model to lose interpretability. In order to tackle this problem, [20] proposed an orthogonal Gaussian process model whose stochastic part is orthogonal to the mean function. The key idea is to construct the correlation function that achieves the orthogonality. The orthogonal correlation function derived from the exponential correlation functions with power $p = 2$ is given in equation (8) of [20]. We implemented the orthogonal correlation function (abbreviated as OGP) as well as the power exponential correlation function (abbreviated as PE) in the simulation.

The estimation results for the linear function coefficients are summarized in Table 1.1 based on 100 replicates for each sample size combination. In general, the proposed approach can estimate the linear function coefficients $(\alpha_0, \boldsymbol{\alpha}, \varphi_1)$ reasonably well. Compared with PE, the estimation improvement using OGP is reported as IMP. It appears that the estimation accuracy can be further improved by the use of orthogonal correlation functions. Therefore, *orthogonal correlation functions are generally recommended* when estimation is of major interest, such as in variable selection and calibration problems.

The parameter estimation results for σ^2 and PE correlation parameters $\boldsymbol{\theta}$ are reported in Table 1.2. The estimation with OGP has similar results, so we omit them to save space. The proposed approach tends to overestimate the correlation parameters for small sample size. This is not surprising because the estimation of correlation parameters is more challenging and the same phenomenon is observed in conventional GP models (see [45]). This problem can be ameliorated by the increase of sample size as shown in Table 1.2. Given the same number of total sample size, $(n = 200, T = 50)$ and $(n = 500, T = 20)$, it appears that a

Table 1.1: Estimation of linear coefficients. The values are the average estimates over 100 replicates, while the values in parentheses are the standard deviation of the estimates. The parameter settings are $\alpha_0 = 0.5, \alpha_1 = -3, \alpha_2 = 2, \alpha_3 = -2, \alpha_4 = 1, \alpha_5 = 0.5$, and $\varphi_1 = 0.8$.

n	T	Corr	$\hat{\alpha}_0$	$\hat{\alpha}_1$	$\hat{\alpha}_2$	$\hat{\alpha}_3$	$\hat{\alpha}_4$	$\hat{\alpha}_5$	$\hat{\varphi}_1$
200	20	PE	0.46	-2.71	1.82	-1.82	0.91	0.46	0.72
			(0.11)	(0.15)	(0.13)	(0.12)	(0.09)	(0.10)	(0.11)
		OGP	0.48	-2.77	1.84	-1.85	0.91	0.46	0.71
			(0.12)	(0.12)	(0.10)	(0.09)	(0.08)	(0.08)	(0.10)
		IMP (%)	4	2	1	1.5	0	0	-1.25
200	50	PE	0.45	-2.68	1.80	-1.79	0.90	0.46	0.70
			(0.07)	(0.10)	(0.09)	(0.08)	(0.07)	(0.06)	(0.07)
		OGP	0.47	-2.75	1.83	-1.83	0.92	0.46	0.71
			(0.08)	(0.09)	(0.06)	(0.06)	(0.06)	(0.05)	(0.08)
		IMP (%)	4	2.3	1.5	2	2	0	1.25
500	20	PE	0.47	-2.76	1.82	-1.83	0.91	0.48	0.73
			(0.09)	(0.14)	(0.12)	(0.10)	(0.08)	(0.07)	(0.07)
		OGP	0.49	-2.81	1.89	-1.88	0.95	0.46	0.74
			(0.08)	(0.09)	(0.07)	(0.06)	(0.06)	(0.06)	(0.07)
		IMP (%)	4	1.7	3.5	2.5	4	4	1.25
500	50	PE	0.45	-2.75	1.83	-1.83	0.92	0.45	0.74
			(0.06)	(0.07)	(0.06)	(0.06)	(0.06)	(0.05)	(0.04)
		OGP	0.47	-2.80	1.87	-1.87	0.93	0.47	0.75
			(0.06)	(0.05)	(0.04)	(0.04)	(0.03)	(0.03)	(0.04)
		IMP (%)	4	1.7	2	2	1	2	1.25

Table 1.2: Estimation of correlation parameters and variance. The values are the average estimates over 100 replicates, while the values in parentheses are the standard deviation of the estimates. The parameter settings are $\theta_1 = 0.5, \theta_2 = 1.0, \theta_3 = 1.5, \theta_4 = 2, \theta_5 = 2.5$, and $\sigma^2 = 1$.

n	T	$\hat{\theta}_1$	$\hat{\theta}_2$	$\hat{\theta}_3$	$\hat{\theta}_4$	$\hat{\theta}_5$	$\hat{\sigma}^2$
200	20	0.86	1.80	2.35	3.30	4.10	0.82
		(0.81)	(1.13)	(1.41)	(1.76)	(1.85)	(0.07)
200	50	0.65	1.55	2.38	3.01	3.80	0.79
		(0.16)	(0.63)	(1.12)	(1.25)	(1.49)	(0.05)
500	20	0.61	1.17	1.93	2.66	3.24	0.87
		(0.16)	(0.25)	(0.54)	(0.96)	(1.17)	(0.05)
500	50	0.57	1.16	1.78	2.37	3.11	0.87
		(0.08)	(0.16)	(0.35)	(0.39)	(0.68)	(0.03)

larger n can improve the estimation accuracy more effectively.

Based on the construction of predictive distribution in Section 1.4, we can emulate a new time series with an untried input. Here we generate 100 random untried inputs to examine its prediction performance. The prediction performance is evaluated by the following two measures. Define the 100 random untried inputs ($n_{\text{test}} = 100$) by $\mathbf{x}_1^*, \ldots, \mathbf{x}_{100}^*$, the misclassification rate (MR) is calculated by

$$\frac{1}{n_{\text{test}} T} \sum_{i=1}^{n_{\text{test}}} \sum_{t=1}^{T} (y_t(\mathbf{x}_i^*) - \hat{y}_t(\mathbf{x}_i^*))^2,$$

where $\hat{y}_t(\mathbf{x}_i^*)$ is the predictive binary response by the proposed method. Since the underlying probabilities are known in the simulation settings, we can also evaluate the prediction performance by the root mean squared prediction error

$$\text{RMSPE} = \left(\frac{1}{n_{\text{test}} T} \sum_{i=1}^{n_{\text{test}}} \sum_{t=1}^{T} (p_t(\mathbf{x}_i^*) - \hat{p}_t(\mathbf{x}_i^*))^2 \right)^{1/2},$$

where $\hat{p}_t(\mathbf{x}_i^*)$ is the predictive probability. The MR and RMSPE results are given in Table 1.3. Overall, the proposed predictor has the misclassification rate less than 17.3% and the

Table 1.3: Comparison of RMSPEs and misclassification rates.

	$n = 200$ $T = 20$	$n = 200$ $T = 50$	$n = 500$ $T = 20$	$n = 500$ $T = 50$
MR (%)	17.22 (1.55)	17.27 (1.39)	17.08 (1.58)	16.83 (1.19)
RMSPE	0.1188 (0.0066)	0.1193 (0.0058)	0.1058 (0.0060)	0.1053 (0.0045)

root mean squared prediction error less than 0.12. Also, with the increase of sample size, the prediction error, in terms of both MR and RMSPE, decreases in general.

Furthermore, the predictive distributions can be used to quantify the prediction uncertainty. The predictive distributions with two random untried inputs are shown in Figure 1.2, where the green dotted lines represent the true probability, the red dashed lines represent the MMSPE predictors obtained in Corollary 1.4.4. From Figure 1.2, it appears that the MMSPE predictors provide accurate predictions in both cases. Moreover, the predictive distributions provide rich information for statistical inference. For example, we can construct 95% predictive confidence intervals for the two untried settings as indicated in blue in Figure 1.2.

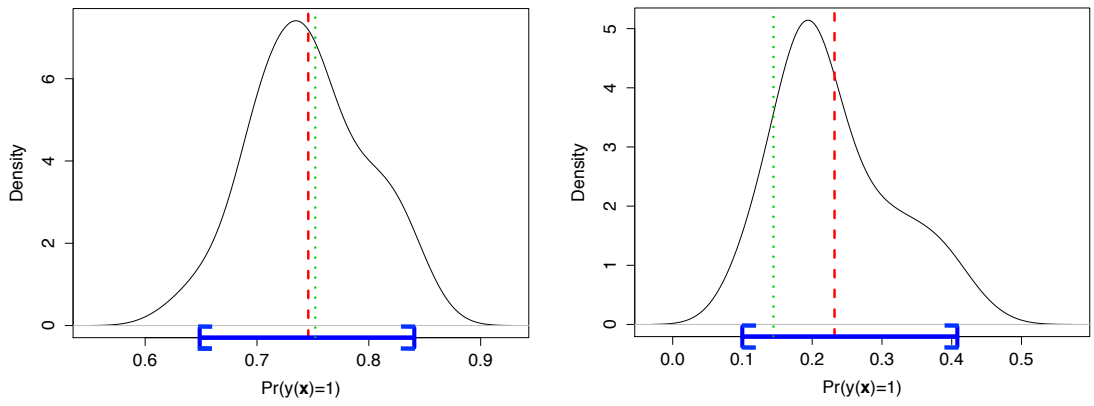


Figure 1.2: Predictive distributions. The green dotted lines are the true probability, the red dashed lines are the MMSPE predictors, and the 95% predictive confidence intervals are indicated in blue.

1.5.2 Prediction Performance

To examine the performance of the proposed model as an emulator, we compare its prediction accuracy with four existing methods: (1) the logistic regression model, (2) a combination of logistic regression model with time series mean function, (3) the Bayesian generalized Gaussian process model [7], which incorporates a Gaussian process prior but does not take into account the time series structure, and (4) the functional Gaussian process proposed by [46], which captures the serial correlation by functional data analysis techniques. These methods are respectively implemented by `R` [19] using packages `binaryGP` [18], `stat` [19], a modification of `stat`, `kernlab` [47] and `GPFDA` [48] adapted to classification.

The simulated data are generated by a modification of the Friedman function [44],

$$\text{logit}(p_t(\mathbf{x})) = y_{t-1}(\mathbf{x}) + \frac{1}{3}\left[10\sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5\right] - 5,$$

where $\mathbf{x} \in [0,1]^5$ and the Friedman function is given in the brackets with intercept -5 and scale $1/3$ to ensure $p_t(\mathbf{x})$ is uniformly located at $[0,1]$. The input $\mathbf{x}$ is randomly generated from $[0,1]^5$ and the corresponding binary output $y_t(\mathbf{x})$ at time t is generated by a Bernoulli distribution with probability $p_t(\mathbf{x})$. The size of the training data is set to be $n = 200, T = 20$.

The prediction performance is evaluated by RMSPE using 100 randomly generated untried settings ($n_{\text{test}} = 100, T = 20$). The results for the five methods based on 100 replicates are shown in the left panel of Figure 1.3. In general, the proposed method has lower RMSPE than the other four methods. By incorporating a Gaussian process to model the nonlinearity, the proposed method outperforms the straightforward combination of logistic regression model and time series structure. On the other hand, comparing with Bayesian generalized Gaussian process model (i.e., `kernlab` in Figure 1.3), the proposed method further improves the prediction accuracy by taking into account the time series structure. The computation time, for model fitting and prediction, is given in the right panel of Figure 1.3. The proposed method is faster than `GPFDA`. Comparing with `glm` and

`glm_ts`, the major computational difficulty lies in the estimation of correlation parameters, which is a common issue in conventional GPs because there is no analytical solution for the parameter estimation. The `kernlab` has better computational performance since it assumes that all the correlation parameters are the equal and estimated by analytic approximation, that is, $\theta_1 = \ldots, \theta_d$ in (1.2). However, the computation time of `kernlab` is expected to increase if this assumption is relaxed to a correlation function with different correlation parameters as in (1.2).

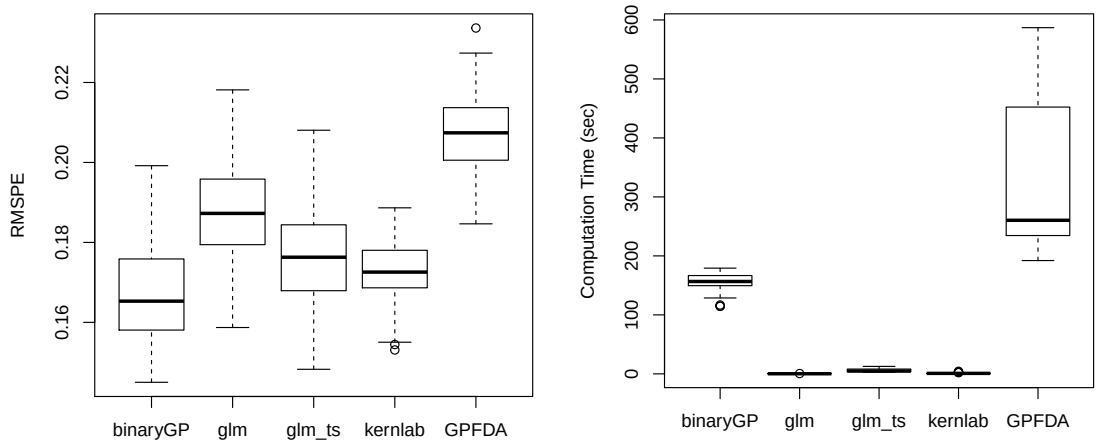


Figure 1.3: Comparison of prediction performance in terms of accuracy (left) and computation time (right). binaryGP: proposed method, glm: logistic regression, glm_ts: logistic regression with time-series mean function, kernlab: Bayesian generalized GP, and GPFDA: functional Gaussian process model.

1.6 Computer Experiments for Cell Adhesion Frequency Assay

In an earlier study based on *in vitro* experiments, an important memory effect was discovered in the repeated adhesion experiments of the micropipette adhesion frequency assay. However, only limited variables of interest can be studied in the lab because of the technical complexity of the biological setting and the complicated experimental manipulation. Therefore, computer simulation experiments are performed to examine the complex mechanisms behind repeated receptor-ligand binding to rigorously elucidate the biological mechanisms

behind the memory effect.

In these computer experiments, two surfaces are simulated to reflect the two opposing membranes in the adhesion frequency assays. The molecules on the surfaces are permitted to interact for the contact duration and then separated for a period of waiting time to simulate the retract-approach phase of the assays. The computer experiments are constructed based on a kinetic proofreading model for receptor modification and solved through a Gillespie algorithm [49], which is a stochastic simulation algorithm. The contact is scored as 1 or 0 depending on whether at least one bond or no bond is observed, respectively. The process is repeated until the given number of contacts is completed.

The biological system investigated here is the T Cell Receptor (TCR) binding to antigen peptide bound to Major Histocompatibility Complex (pMHC), which has previously been shown to exhibit memory in repeated contacts [3]. The TCR is the primary molecule involved in detecting foreign antigens which are presented on pMHC molecules expressed by infected cells. Memory in serial interactions of these foreign antigens may be a mechanism which underlies the major properties of T cell antigen recognition: sensitivity, specificity, and context discrimination. It has largely remained uninvestigated due to the small time scales at which the mechanism operates and the complexity of the experimental system. Although there are many possible cellular mechanisms which may induce this behavior, we investigate a specific mechanism, called free induction mechanism [50], in this study as to how this memory may be controlled: pMHC binding to a single TCR within a cluster upregulates the kinetics of all TCRs within that cluster.

The free induction mechanism has six control variables given in Table 1.4. The range of each control variable in Table 1.4 is given by physical principles or estimated through similar molecular interactions. The design of the control variables is a 60-run OA-based Latin hypercube designs [51]. For each run, it consists of 50 replicates and each replicate has 100 repeated contacts ($T = 100$).

The proposed estimation method is implemented with orthogonal correlation functions

Table 1.4: Control variables in cell adhesion frequency assay experiments.

Variable	Description	Range
$x_{K_{f,p}}$	on-rate enhancement of activated TCRs	(1,100)
$x_{K_{r,p}}$	off-rate enhancement of activated TCRs	(0.1,100)
$x_{T_{half}}$	half-life of cluster activation	(0.1,10)
x_{Tc}	cell-cell contact time	(0.1,10)
x_{Tw}	waiting time in between contacts	(0.1,10)
x_{Kc}	kinetic proofreading modification rate for activation of cluster	(0.1,10)

derived from the power exponential correlation function with $p = 2$, which can be found in equation (8) in [20]. We start with a large model in which the mean function includes all the main effects of the control variables and their interactions with the past time series output y_{t-1}. The model is written as:

$$\begin{aligned}\text{logit}(p_t(\mathbf{x})) = -0.07 + \hat{\varphi}_1 y_{t-1}(\mathbf{x}) + \hat{\alpha}_1 x_{K_{f,p}} + \hat{\alpha}_2 x_{K_{r,p}} + \hat{\alpha}_3 x_{T_{half}} + \hat{\alpha}_4 x_{Tc} + \hat{\alpha}_5 x_{Tw} + \hat{\alpha}_6 x_{Kc} + \\ (\hat{\gamma}_1 x_{K_{f,p}} + \hat{\gamma}_2 x_{K_{r,p}} + \hat{\gamma}_3 x_{T_{half}} + \hat{\gamma}_4 x_{Tc} + \hat{\gamma}_5 x_{Tw} + \hat{\gamma}_6 x_{Kc}) y_{t-1}(\mathbf{x}) + Z_t(\mathbf{x}),\end{aligned}$$

where all the control variables are standardized to $[0, 1]$, $\hat{\sigma} = 0.43$ and the estimated correlation parameters are $\hat{\boldsymbol{\theta}} = (\hat{\theta}_{K_{f,p}}, \hat{\theta}_{K_{r,p}}, \hat{\theta}_{T_{half}}, \hat{\theta}_{Tc}, \hat{\theta}_{Tw}, \hat{\theta}_{Kc}) = (3.28, 1.70, 7.77, 0.06, 4.78, 0.74)$. Estimation results for the mean function coefficients are given in Table 1.5 with p values calculated based on the asymptotic results in Theorem 1.3.1. We use these p values to perform variable selection and identify significant effects for the mean function. According to Table 1.5, $x_{T_{half}}$ has no significant effect in the mean function at the 0.01 level. By removing $x_{T_{half}}$, the model can be updated as

$$\begin{aligned}\text{logit}(p_t(\mathbf{x})) = &- 0.07 + 0.14 y_{t-1}(\mathbf{x}) + 0.13 x_{K_{f,p}} + 0.37 x_{K_{r,p}} + 0.47 x_{Tc} - 0.08 x_{Tw} + 0.15 x_{Kc} \\ &+ (0.16 x_{K_{f,p}} + 0.23 x_{K_{r,p}} - 0.09 x_{T_{half}} + 0.23 x_{Tc} - 0.17 x_{Tw} + 0.36 x_{Kc}) y_{t-1}(\mathbf{x}) + Z_t(\mathbf{x}),\end{aligned}$$

where $\hat{\sigma} = 0.44$, the estimated correlation parameters are $\hat{\boldsymbol{\theta}} = (\hat{\theta}_{K_{f,p}}, \hat{\theta}_{K_{r,p}}, \hat{\theta}_{T_{half}}, \hat{\theta}_{Tc}, \hat{\theta}_{Tw}, \hat{\theta}_{Kc}) = (3.27, 1.71, 7.77, 0.06, 4.81, 0.74)$. Based on the updated model, among

Table 1.5: Estimation results.

	Value	Standard deviation	Z score	p value
$\hat{\varphi}_1$	0.14	0.02	7.96	0.0000
$\hat{\alpha}_1$	0.13	0.02	5.96	0.0000
$\hat{\alpha}_2$	0.37	0.02	17.82	0.0000
$\hat{\alpha}_3$	0.00	0.02	0.08	0.9331
$\hat{\alpha}_4$	0.47	0.02	22.54	0.0000
$\hat{\alpha}_5$	-0.08	0.02	-3.79	0.0001
$\hat{\alpha}_6$	0.15	0.02	6.86	0.0000
$\hat{\gamma}_1$	0.16	0.03	5.2	0.0000
$\hat{\gamma}_2$	0.23	0.03	7.68	0.0000
$\hat{\gamma}_3$	-0.09	0.03	-2.92	0.0035
$\hat{\gamma}_4$	0.23	0.03	7.28	0.0000
$\hat{\gamma}_5$	-0.17	0.03	-5.47	0.0000
$\hat{\gamma}_6$	0.36	0.03	11.83	0.0000

the interaction effects, all the control variables except $x_{T_{half}}$ are significant for inducing memory in the free induction model.

The application of this statistical approach to the analysis of simulations and experimental data will be powerful in illuminating the unknown biological mechanism, and also informs the next round of experiments by advising future manipulations. Additionally, developments on the calibration of computer experiments based upon the proposed predictive distribution will help provide insight into the range of possible values of variables, such as the increases in kinetic rates, which are difficult to determine through existing methods due to the small time scale at which this mechanism operates and the limits of existing experimental techniques.

Besides estimation, the proposed method also provides predictors which can serve as efficient and accurate emulators for untried computer experiments. The construction of emulator is an important step for future research on calibration where computer experiment outputs under the same settings of the lab experiments are required but not necessarily available. To assess the predictive performance, we compare the proposed predictor with the four existing methods discussed in Section 1.5 based on a 10-fold cross-validation study. The prediction error measured by the misclassification rate is reported in Figure 1.4, which

shows that the proposed predictor has a smaller prediction error compared to the other alternatives.

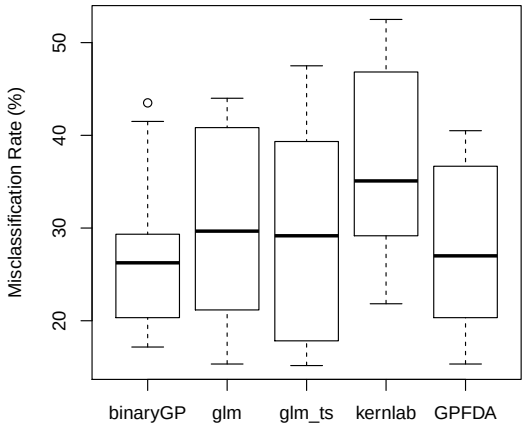


Figure 1.4: Comparison of misclassification rate. binaryGP: proposed method, glm: logistic regression, glm_ts: logistic regression with time-series mean function, kernlab: Bayesian generalized GP, and GPFDA: functional Gaussian process model.

1.7 Summary and Concluding Remarks

In spite of the prevalence of Gaussian process models in the analysis of computer experiments, their applications are limited to the Gaussian assumption on the responses. Motivated by the study of cell adhesion where the computer simulation responses are binary time series, a generalized Gaussian process model is proposed in this chapter. The estimation procedure is introduced and asymptotic properties are derived. An optimal predictor and its predictive distribution are constructed which can be used for uncertainty quantification and calibration of future computer simulations. An `R` package is available for implementing the proposed methodology. The methodology is applied to analyze stochastic computer simulations for a cell adhesion mechanism. The results reveal important biological information which is not available in lab experiments and provide valuable insights on how the next round of lab experiments should be conducted.

The current work can be extended in several directions. First, we will extend the

proposed method to other non-Gaussian data, such as the count data. It is conceivable that the current estimation procedure can be directly extended to other exponential family distributions, but different predictive distributions are expected for different types of non-Gaussian data. Second, the computational cost in the proposed procedure can be further reduced. In particular, the inversion of $\boldsymbol{R}_{\boldsymbol{\theta}}$ can be computationally prohibitive when sample size is large. This computational issue has been addressed for conventional GP models in the recent literature. Extensions of these methods (e.g., [2, 52]) to binary responses deserve further attention. Third, many mathematical models underlying the computer simulations contain unknown parameters, which need to be estimated using data from lab experiments. This problem is called calibration and much work has been done in the computer experiment literature. However, the existing methods (e.g., [53], [23] and [54]) are only applicable under the Gaussian assumption. Based upon the model and prediction procedure proposed herein, we will work on developing a calibration method for non-Gaussian data.

CHAPTER 2
CALIBRATION FOR COMPUTER EXPERIMENTS WITH BINARY RESPONSES

2.1 Introduction

To study a scientific problem by experimentation, there are generally two different approaches. One is to conduct physical experiments in a laboratory and the other is to perform computer simulations for the study of real systems using mathematical models and numerical tools, such as finite element analysis. Computer experiments have been widely adopted as alternatives to physical experiments, especially for studying complex systems where physical experiments are infeasible, inconvenient, risky, or too expensive. For example, [1] study high-fidelity simulations for turbulent flows in a swirl injector, which are used in a wide variety of engineering applications. In computer experiments, there are two sets of input variables. One is the set of general inputs that represents controllable quantities which are also present in physical experiments, while the other is the set of unknown parameters that represents certain inherent attributes of the underlying systems but cannot be directly controlled or measured in physical experiments. These unknown parameters are called *calibration parameters* in the literature [5]. The focus of this paper is *calibration* which refers to the estimation of the calibration parameters using data collected from both physical and computer experiments, so that the computer outputs can closely match the physical responses. [53] first developed a Bayesian method for calibration and has made a large impact in various fields where computer experiments are used [55, 56, 54, 23, 57].

An accurate estimation of calibration parameters is important because it can provide scientific insight that may not be directly obtainable in physical experiments. For example, calibration parameters in the implosion simulations are not measurable in physical

experiments, the understanding of which provides important information regarding the yield stress of steel and the resulting detonation energy [58]. In the study of high-energy laser radiative shock system, one of the calibration parameters is the electron flux limiter, which is useful in predicting the amount of heat transferred between cells of a space-time mesh in the simulation but cannot be controlled in physical experiments [54].

This paper is motivated by a calibration problem in a study of molecular interactions, where the output of interest is binary. We study the molecular interaction by an important type of single molecular experiments called micropipette adhesion frequency assays [59]. It is the only published method for studying the kinetic rates of cell adhesion, which plays an important role in many physiological and pathological processes. Typically, there are two ways to perform micropipette adhesion frequency experiments: conducting physical experiments in a laboratory, and studying the complex adhesion mechanism by computer experiments based on a kinetic proofreading model through a Gillespie algorithm [49]. For both physical and computer experiments, the output of interest is binary, which indicates whether a controlled contact results in adhesion or not [60, 3, 50].

Binary outputs are common in many applications. For example, in manufacturing applications computer simulations are often conducted for failure analysis where the outputs of interest are binary, i.e., failure or success [15]. In other biological problems, binary outputs are observed and evolve in time, such as neuron firing simulations, cell signaling pathways, gene transcription, and recurring diseases [16, 17]. Despite numerous scientific studies with binary outputs, there has been few studies for binary responses. Most of the calibration methods are developed for continuous outputs. Extensions of existing calibration methods to binary outputs are not straightforward for two reasons. First, calibration relies on statistical modeling for both computer experiments and physical experiments, but the required modeling techniques for binary outputs are different from those for continuous outputs. Second, the conventional approach to estimating calibration parameters is to match the computer outputs and physical responses, while our interest for binary responses is to

match the underlying probability functions for the computer and physical outputs.

In this paper, we develop a general framework for calibration problems with binary outputs. Calibration parameters are estimated by minimizing the discrepancy between the underlying probability functions in physical experiments and computer experiments. The remainder of the paper is organized as follows. In Section 2.2, the calibration procedure is described in details. Theoretical properties of the proposed method are stated in Section 2.3, where we show that the estimation procedure is consistent and semiparametric efficient. Numerical studies are conducted in Section 2.4 to demonstrate the finite sample performance of estimation. In Section 2.5, the proposed framework is implemented in micropipette adhesion experiments. Concluding remarks are given in Section 2.6. Detailed derivations are provided in Appendix B.

2.2 Calibration by L_2 projection

Suppose n binary outputs are observed from physical experiments and are denoted by $(y_1^p, \cdots, y_n^p)$, where the superscript p stands for "physical" and y_i^p is the ith observation taking value 0 or 1. Let Ω denote a d-dimensional experimental region for the control variables $\mathbf{x}$, which is a convex and compact subset of $\mathbb{R}^d$. For each output, the corresponding setting of the control variable is denoted by $\mathbf{x}_i$, where $i = 1, \ldots, n$. Suppose the probability of observing $y = 1$ is assumed to have the following model,

$$\eta(\mathbf{x}_i) := Pr(y_i^p = 1|\mathbf{x}_i) = g(\xi_0(\mathbf{x}_i)), \tag{2.1}$$

where g is a pre-specified link function. For the binary outputs, we assume g to be the commonly used logistic function, i.e., $g(x) = 1/(1 + \exp\{-x\})$. The function $\xi_0(\cdot)$ is unknown and often called the *true process* in the computer experiment literature [53, 23, 57].

To study the same scientific problem, a more cost-effective way is to conduct computer

simulations (or called computer experiments in this paper). Apart from the control variables $\mathbf{x}$, computer experiments involve calibration parameters, denoted by θ and $\theta \in \Theta$ which is a compact subset of $\mathbb{R}^q$. These parameters are of scientific interest but their "true" values are unknown. The binary output from computer experiment is denoted by $y^s(\mathbf{x}, \theta)$ with the superscript s standing for "simulation". The conditional expectation of $y^s(\mathbf{x}, \theta)$ can be written as

$$p(\mathbf{x}, \theta) := Pr(y^s = 1|\mathbf{x}, \theta),$$

where $(\mathbf{x}, \theta) \in \Omega \times \Theta$. Even though computer experiments require less experimental manipulation and have smaller risk compared to physical experiments, they can also be computationally intensive (e.g., the Λ-cold dark matter model in [61] and the high-fidelity simulation in [1]). Therefore, it is not practical to have simulation conducted over the entire experimental region $\Omega \times \Theta$. Instead, the computer experiments are conducted by employing a careful design of experiment, such as space-filling designs [5], on a subset of the experimental region.

The goal of calibration is to search for the setting of the calibration parameters such that the outputs from physical experiments fit as closely as possible to the corresponding outputs of computer experiments. This problem is rigorously formulated by [53] in a Bayesian framework. Despite many successful applications using the Bayesian approach (e.g., [62, 58, 63]), recent studies have raised concerns about the *nonidentifiability* issue of the calibration parameters in [53]. See [55, 64, 56, 54]. To tackle this problem, [23, 57] propose a frequentist framework based on the method of L_2 *projection*. The idea is to estimate the calibration parameters by minimizing the L_2 distance between the physical output and the computer output. It was shown in [23, 57] that the calibration method achieves estimation consistency with an optimal convergence rate.

Although there is a rich literature on calibration, the existing approaches focus mainly on continuous outputs. Inspired by the optimality of the frequentist approach proposed by [57], we develop a calibration framework for binary outputs using the idea of L_2 projection.

Ideally, θ can be obtained by minimizing the discrepancy measured by the L_2 distance between the underlying probabilities of success in the physical and computer experiments, respectively. This can be written as

$$\theta^* = \arg\min_{\theta\in\Theta} \|\eta(\cdot) - p(\cdot,\theta)\|_{L_2(\Omega)}, \tag{2.2}$$

where the L_2 norm is defined by $\|f\|_{L_2(\Omega)} = (\int_\Omega f^2)^{1/2}$. The direct calculation of (2.2), however, is not feasible because the true process $\xi(\cdot)$ in (2.1) is unknown and therefore $\eta(\cdot)$ is unknown. Furthermore, $p(\cdot,\cdot)$ is often unknown because the computer outputs are binary.

Instead of solving (2.2) directly, we propose to perform L_2 projections based on the estimates of $\eta(\cdot)$ and $p(\cdot,\theta)$. First, the true process $\xi(\cdot)$ is estimated by a kernel logistic regression, that is,

$$\hat{\xi}_n := \arg\max_{\xi\in\mathcal{N}_\Phi(\Omega)} \frac{1}{n}\sum_{i=1}^{n} (y_i^p \log g(\xi(\mathbf{x}_i)) + (1-y_i^p)\log(1-g(\xi(\mathbf{x}_i)))) + \lambda_n\|\xi\|^2_{\mathcal{N}_\Phi(\Omega)}, \tag{2.3}$$

where $\|\cdot\|_{\mathcal{N}_\Phi(\Omega)}$ is the norm of the reproducing kernel Hilbert space $\mathcal{N}_\Phi(\Omega)$ generated by a given positive definite reproducing kernel Φ, and $\lambda_n > 0$ is a tuning parameter, which can be chosen by some model selection criterion like cross-validation. The optimal function (2.3) has the form $\hat{\xi}_n(\mathbf{x}) = \hat{b} + \sum_{i=1}^n \hat{a}_i\Phi(\mathbf{x}_i,\mathbf{x})$, where $\hat{b}$ and $\{\hat{a}_i\}_{i=1}^n$ can be solved by the iteratively re-weighted least squares algorithm. Detailed discussions can be found in [65, 66, 67, 68]. Based on the estimated true process, we then have $\hat{\eta}_n = g(\hat{\xi}_n)$. Because the computer outputs are binary, $p(\cdot,\cdot)$ is not observable and needs to be estimated. Therefore, we assume that a surrogate model $\hat{p}_N(\cdot,\cdot)$ can be constructed as a good approximation to $p(\cdot,\cdot)$ based on N computer outputs, where N is assumed to be larger than n because computer experiments are usually cheaper than physical experiments. Recent studies on generalized Gaussian process models, such as [7], [10], and [69], can be used as the surrogate model for binary computer experiments. Given $\hat{\eta}_n$ and $\hat{p}_N(\cdot,\cdot)$, we are ready to estimate the

calibration parameters by minimizing the L_2 projection as follows,

$$\hat{\theta}_n = \arg\min_{\theta\in\Theta} \|\hat{\eta}_n(\cdot) - \hat{p}_N(\cdot,\theta)\|_{L_2(\Omega)}. \tag{2.4}$$

2.3 Theoretical Properties

This section consists of two parts. Theoretical properties for the physical experiment modeling, $\hat{\eta}_n$, are first discussed in Section 2.3.1 and then the results for the calibration parameters $\hat{\theta}_n$ are given in Section 2.3.2.

2.3.1 Asymptotic Results for Physical Experiment Modeling

We start with a result developed by [70] for general nonparametric regression (Lemma 11.4 and 11.5 in [70]). Denote ξ_0 as the true process, g as a logistic function, and

$$\eta_0(\mathbf{x}) = g(\xi_0(\mathbf{x})). \tag{2.5}$$

Suppose $\mathcal{F}$ is the class of all regression functions equipped with the Sobolev norm $\|\cdot\|_{H^m(\Omega)}$, which is defined by

$$\|\xi\|^2_{H^m(\Omega)} = \|\xi\|^2_{L_2(\Omega)} + \sum_{i=1}^{m} \left\| \frac{\partial^i \xi}{\partial x^i} \right\|^2_{L_2(\Omega)}.$$

Let

$$\hat{\xi}'_n := \arg\max_{\xi\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} (y_i^p \log g(\xi(\mathbf{x}_i)) + (1-y_i^p)\log(1-g(\xi(\mathbf{x}_i)))) + \lambda_n \|\xi\|^2_{H^m(\Omega)},$$

for some $\lambda_n > 0$. Then the convergence rate of $\hat{\xi}'_n$ is given in the following lemma.

Lemma 2.3.1. *Let $\xi_0 \in \mathcal{F}$. Assume that there exists some nonnegative k_0 and k_1 so that*

$$k_0^2 \le g(\xi_0(\mathbf{x})) \le 1 - k_0^2 \quad \textit{and} \quad |\partial g(z)/\partial z| \ge k_1 > 0 \quad \textit{for all} \quad |z - z_0| \le k_1,$$

where $z_0 = \xi_0(\mathbf{x})$ *and* $\mathbf{x} \in \Omega$. *For* $\lambda_n^{-1} = O(n^{2m/(2m+1)})$, *we have*

$$\|\hat{\xi}_n'\| = O_p(1), \|g(\hat{\xi}_n') - g(\xi_0)\|_{L_2(\Omega)} = O_p(\lambda_n^{1/2}),$$

and

$$\|\hat{\xi}_n' - \xi_0\|_{L_2(\Omega)} = O_p(\lambda_n^{1/2}).$$

In fact, the norms of some reproducing kernel Hilbert spaces (RKHS) are equivalent to Sobolev norms. For instance, the RKHS generated by the Matérn kernel function, given by

$$\Phi(\mathbf{x}, \mathbf{x}') = \frac{1}{\Gamma(\nu)2^{\nu-1}}(2\sqrt{\nu}\|\rho(\mathbf{x} - \mathbf{x}')\|)^{\nu} K_{\nu}(2\sqrt{\nu}\|\rho(\mathbf{x} - \mathbf{x}')\|), \tag{2.6}$$

where $\nu \geq 1$ and $\rho \in \mathbb{R}_+^d$ are parameters and K_ν is a Bessel function with parameter ν, is equal to the (fractional) Sobolev space $H^{\nu+d/2}(\Omega)$, and the corresponding norms $\|\cdot\|_{\mathcal{N}_\Phi(\Omega)}$ and $\|\cdot\|_{H^{\nu+d/2}(\Omega)}$ are equivalent [71, 57]. Therefore, as a consequence of Lemma 2.3.1, we have the following proposition for $\hat{\xi}_n$ obtained by (2.3).

Proposition 2.3.2. Suppose that $\xi_0 \in \mathcal{F} = \mathcal{N}_\Phi(\Omega)$, and $\mathcal{N}_\Phi(\Omega)$ can be embedded into $H^m(\Omega)$. Then, for $\lambda_n^{-1} = O(n^{2m/(2m+d)})$, the estimator $\hat{\xi}_n$ in (2.3) and $\hat{\eta}_n = g(\hat{\xi}_n)$ satisfy

$$\|\hat{\xi}_n\|_{\mathcal{N}_\Phi(\Omega)} = O_p(1), \quad \|\hat{\eta}_n - \eta_0\|_{L_2(\Omega)} = O_p(\lambda_n^{1/2}),$$

and

$$\|\hat{\xi}_n - \xi_0\|_{L_2(\Omega)} = O_p(\lambda_n^{1/2}).$$

Proposition 2.3.2 suggests that one may choose $\lambda_n \asymp n^{-2m/(2m+d)}$ to obtain the best convergence rate $\|\hat{\eta}_n - \eta_0\|_{L_2(\Omega)} = O_p(n^{-m/(2m+d)})$, where $a_n \asymp b_n$ denotes that the two positive sequences a_n and b_n have the same order of magnitude.

2.3.2 Asymptotic Results for $\hat{\theta}_n$

In this section, we show that $\hat{\theta}_n$ obtained by the L_2 calibration in (2.4) is consistent with the true calibration parameter θ^* in (2.2) and follows an asymptotic normal distribution. These results rely on some regularity conditions as well as assumptions on the fitted nonparametric model for the physical experiments and the emulator built by computer experiments.

The regularity assumptions and the detailed proofs are given in the Appendix B. Additional assumptions are highlighted below.

Assumption 2.3.1. Assumptions B1-B4 are related to the nonparametric models and Assumptions C1 and C2 are related to the emulators.

B1: $\xi \in \mathcal{N}_\Phi(\Omega)$ and $\mathcal{N}_\Phi(\Omega, \rho)$ is Donsker for all $\rho > 0$.

B2: $\|\hat{\eta} - \eta\|_{L_2(\Omega)} = o_p(1)$.

B3: $\|\hat{\xi}\|_{\mathcal{N}_\Phi(\Omega)} = O_p(1)$.

B4: $\lambda_n = o_p(n^{-1/2})$.

C1: $\|\hat{p}_N - p\|_{L_\infty(\Omega\times\Theta)} = o_p(N^{-1/2})$.

C2: $\|\frac{\partial \hat{p}_N}{\partial \theta_i} - \frac{\partial p}{\partial \theta_i}\|_{L_\infty(\Omega\times\Theta)} = o_p(N^{-1/2})$ for $i = 1, \ldots, q$.

The Donsker property is an important concept in the theoretical studies of empirical processes. The definition and detailed discussion are referred to [72] and [73]. [23] showed that if the conditions of Proposition 2.3.2 hold and $m > d/2$, then $\mathcal{N}_\Phi(\Omega, \rho)$ is a Donsker. The authors also mentioned that under Assumption A1 and $\mathbb{E}[\exp\{C|Y_i^p - \eta(\mathbf{x}_i)|\}] < +\infty$ for some $C > 0$, the conditions of Proposition 2.3.2 are satisfied. Therefore, by choosing a suitable sequence of λ_n, say $\lambda_n \asymp n^{-2m/(2m+d)}$, one can show that condition B4 holds and B2 and B3 are ensured by Proposition 2.3.2. Assumptions C1 and C2 assume that the approximation error caused by emulation in computer experiments is negligible compared

to the estimation error caused by the measurement error in physical experiments. Given the fact that the cost for computer experiments is usually cheaper than physical experiments, this assumption is reasonable because the sample size of computer experiments is usually larger than that of physical experiments (i.e., $N > n$). Under some regularity conditions, the emulators constructed by the existing methods, such as [7], [10], and [69], satisfy the assumptions and can be applied in this framework.

Theorem 2.3.3. *Under Assumption 2.3.1 and the regularity conditions in Appendix B.1, we have*

$$\hat{\theta}_n - \theta^* = 2V^{-1}\left(\frac{1}{n}\sum_{i=1}^{n}(Y_i^p - \eta(\mathbf{x}_i))\frac{\partial p}{\partial \theta}(\mathbf{x}_i, \theta^*)\right) + o_p(n^{-1/2}), \tag{2.7}$$

where $V = \mathbb{E}\left[\frac{\partial^2}{\partial\theta\partial\theta^T}(\eta(X) - p(X,\theta^*))^2\right]$.

According to the central limit theorem and the results in Theorem 2.3.3, $\hat{\theta}_n$ has an asymptotically normal distribution as follows.

Corollary 2.3.4. *Under the assumptions in Theorem 2.3.3, we have*

$$\sqrt{n}(\hat{\theta}_n - \theta^*) \xrightarrow{d} \mathcal{N}(0, 4V^{-1}WV^{-1}),$$

provided that W *is positive definite and can be written as*

$$W = \mathbb{E}\left[\eta(X)(1-\eta(X))\frac{\partial p}{\partial \theta}(X,\theta^*)\frac{\partial p}{\partial \theta^T}(X,\theta^*)\right]. \tag{2.8}$$

In calibration problems, the parameter of interest is a q-dimensional calibration parameter θ^*, while the parameter space of model (2.1) contains an infinite dimensional function space which covers ξ. Therefore, the calibration problem is regarded as a semiparametric problem, and if one method can reach the highest estimation efficiency for semiparametric problem, we call it *semiparametric efficient*. We refer to [74] and [73] for more details. The following theorem shows that, similar to its counterpart for continuous outputs [23], this method enjoys the semiparametric efficiency.

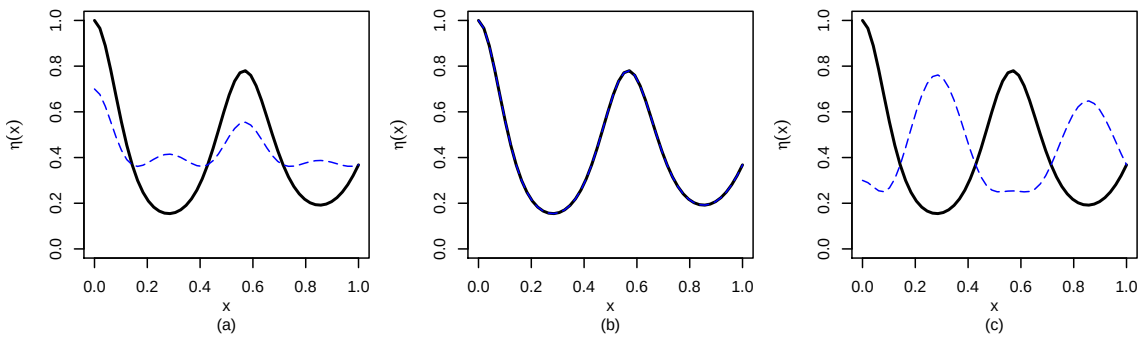


Figure 2.1: True functions in the physical experiment and computer experiment. Block line represents the true function of the physical experiment, and blue line represents the true function of the computer experiment with calibration parameter (a) $\theta = 0$; (b) $\theta = 0.3$; (c) $\theta = 1$.

Theorem 2.3.5. *Under the Assumptions in Appendix B.1, the L_2 calibration method* (2.4) *is semiparametric efficient.*

2.4 Numerical Study

Assume that the binary physical outputs are randomly generated from a Bernoulli distribution denoted by $Y^p \sim Ber(\eta(x))$, where

$$\eta(x) = \exp\{\exp(-0.5x)\cos(3.5\pi x) - 1\},$$

$x \in \Omega = (0, 1)$, and $\eta(x) \in (0, 1)$ for all $x \in \Omega$. The binary computer outputs Y^s are randomly generated from $Ber(p(x, \theta))$, where

$$p(x, \theta) = \eta(x) - |\theta - 0.3| \exp(-0.5x)\cos(3.5\pi x),$$

$x \in \Omega = (0, 1), \theta \in \Theta = (0, 1)$, and $p(x, \theta) \in (0, 1)$ for all $x \in \Omega$ and $\theta \in \Theta$. Figure 2.1 shows the functions $\eta(x)$ (black lines) and $p(x, \theta)$ with three different calibration parameters (blue dashed lines). In this example, the true calibration parameter $\theta^* = 0.3$ because it leads to no discrepancy between the physical and computer experiments (Figure 2.1(b)).

Consider the physical experiments with sample size n, where the inputs $\{x_i\}_{i=1}^n$ are

Table 2.1: Mean and standard deviation (SD) of the estimated calibration parameters in 100 replicates.

n	N	θ^*	Mean	SD
50	150	0.3	0.3085	0.1761
100	300	0.3	0.2986	0.1372

selected with equal space in $[0, 1]$. The sample size for computer experiments is N, and the inputs $\{(x_i, \theta_i)\}_{i=1}^N$ are uniformly selected from $[0, 1]^2$. The calibration parameter is estimated by (2.4), in which $\hat{\eta}_n(x)$ is obtained by the kernel logistic regression (2.3) and the Matérn kernel function (2.6) is chosen with $\nu = 1.5$. The parameters ρ and λ_n are chosen via cross-validation. $\hat{p}_N(x, \theta)$ is obtained by the generalized Gaussian process in [7], where the radial basis function kernel,

$$\Phi_\phi((x_i, \theta_i), (x_j, \theta_j)) = \exp\left\{-\frac{(x_i - x_j)^2 + (\theta_i - \theta_j)^2}{2\phi}\right\},$$

is chosen with the parameter ϕ chosen via cross-validation.

The calibration performance based on 100 replicates are summarized in Table 2.1. The proposed L_2 calibration method leads to a small calibration bias, which is less than 0.01, and the standard deviation decreases when the sample size increases, which agrees with the asymptotic results developed in Section 2.3.

2.5 Applications in single molecular studies

The adaptive immune system defends the organism against diseases by recognition of pathogens by the T cell. T cell receptor (TCR) is the primary molecule on T cell in detecting foreign antigens which are present on major histocompatibility complex (pMHC) molecule expressed by infected cells. Failure to recognize pathogens can result in immune deficiency. False recognition can lead to autoimmune diseases. Therefore, how TCR discriminates different peptides is a central question in the research on adaptive immunity.

To understand the molecular interactions on T cells, there are two approaches. One is to preform experiments in a lab and the other is to conduct computer simulations. Here we focus on the study of molecular interactions in micropipette adhesion frequency assays.

In a lab, the micropipette adhesion frequency assay is performed as follows. A red blood cell (RBC) pressurized by micropipette aspiration is used to present the ligands and to detect binding with the receptor on a T cell. The T cell is put into controlled contact with the RBC for a constant area and a preprogrammed duration and then retracted. The output of interest is binary, indicating whether a controlled contact results in adhesion or not. If there is an adhesion between molecules at the end of the contact, retraction will stretch the red blood cell and the RBC membrane will be elongated; otherwise the RBC will smoothly restore its spherical shape.

Although physical experiments allow accurate measurements of the adhesion frequency, they are time-consuming and often involve complicated experimental manipulation. Moreover, only limited variables of interest can be studied in the lab because of the technical complexity of the biological settings. Therefore, a cost-effective approach is to illuminate the unknown biological mechanism in cell adhesion through computer simulations. For the micropipette adhesion frequency assays, computer simulations can be conducted based on a kinetic proofreading model and simulated through a Gillespie algorithm [49]. Figure 2.2 illustrates the computer model for the micropipette adhesion frequency assays.

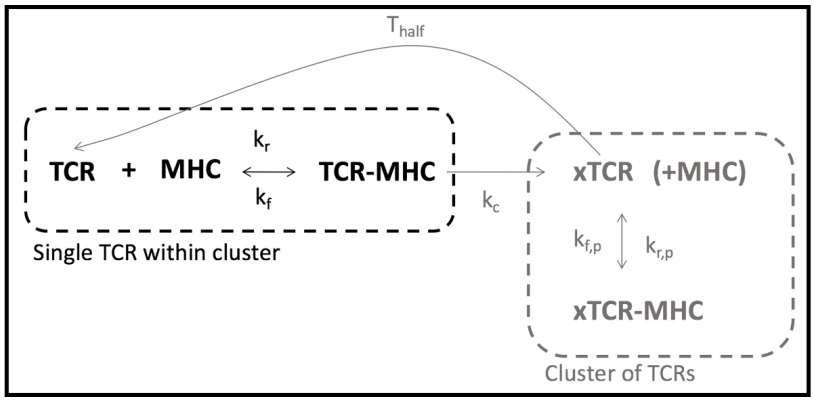


Figure 2.2: Illustration of the computer experiments

Table 2.2: Control variables in cell adhesion frequency assay experiments.

Variable	Description	Range
x_{Tc}	cell-cell contact time	(0.1,10)
x_{Tw}	waiting time in between contacts	(0.1,10)
$x_{K_{f,p}}$	on-rate enhancement of activated TCRs	(1,100)
$x_{K_{r,p}}$	off-rate enhancement of activated TCRs	(0.1,100)
$x_{T_{half}}$	half-life of cluster activation	(0.1,10)
x_{Kc}	kinetic proofreading modification rate for activation of cluster	(0.1,10)

There are two shared control variables, denoted by x_{Tc} and x_{Tw}, in both physical and computer experiments. Additionally, four calibration parameters, denoted by $x_{K_{f,p}}$, $x_{K_{r,p}}$, $x_{T_{half}}$, and x_{Kc}, only appear in the computer simulations. Their values are of biological interest but cannot be measured or controlled in the lab experiments. The detailed descriptions for these variables are given in Table 2.2. For the lab experiments, the values of x_{Tc} and x_{Tw} are randomly chosen from the sample space $(0.1, 10)^2$ with size $n = 3,081$, and the corresponding outputs are generated by conducting the lab experiments. The design for computer experiments is a 60-run OA-based Latin hypercube design [51], and for each run it consists of 100 replicates, i.e., $N = 6,000$.

Similar to the estimation procedure in Section 2.4, the L_2 projection procedure can be implemented and the estimated calibration parameters are

$$(x_{K_{f,p}}, x_{K_{r,p}}, x_{T_{half}}, x_{Kc}) = (4.48, 0.95, 2.23, 0.45),$$

and the corresponding L_2 distance is 0.05. Plugging in the estimated calibration parameters to the emulator, the adhesion probabilities obtained from computer experiments (red dashed lines) are compared with those from physical experiments (black lines) in Figure 2.3 as a function of the two control variables, contact time and waiting time. It appears that the emulator with the estimated calibration parameters can reasonably capture the trend observed in the physical experiments. The proposed calibration procedure provides insight into the values of the calibration parameters in the T cell adhesion experiments, which are

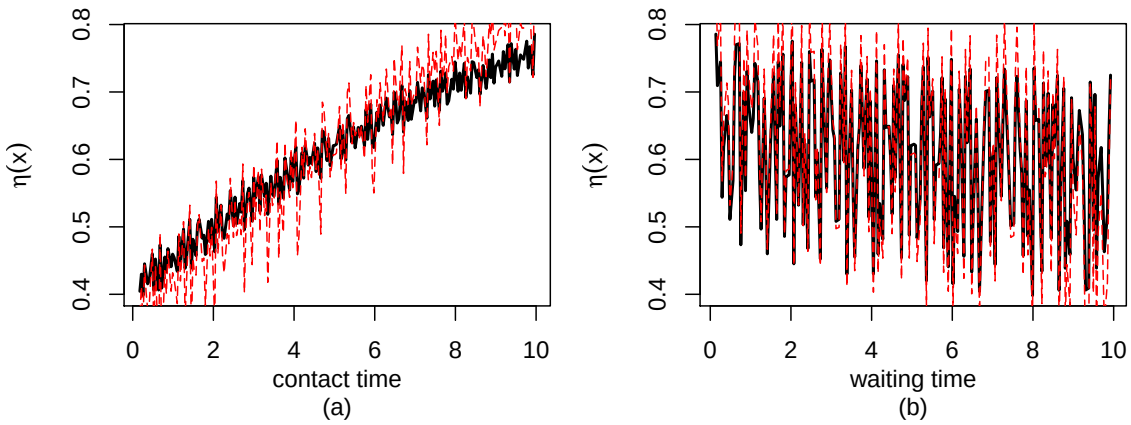


Figure 2.3: Fitted functions for physical and computer experiments. Black lines represent the fitted model $\hat{\eta}_n$ based on the physical experiments, and red lines represent the fitted model $\hat{p}_N$ based on the computer experiments.

difficult to determine by physical experiments due to the small time scale at which this mechanism operates and the limitation of existing experimental techniques.

2.6 Summary and Concluding Remarks

How to estimate the calibrate parameters in computer experiments is an important problem, but the existing calibration methods mainly focus on continuously outputs. Motivated by an analysis of single molecular experiments, we propose a new calibration framework for binary responses. The method of estimated calibration parameters is shown to be asymptotically consistent and semiparametric efficient. Our numerical studies confirm the estimation accuracy in finite-sample performance, and the application in single molecular studies illustrates that the proposed calibration method reveals important insight on the underlying adhesion mechanism which cannot be directly observed through existing methods.

Our work lays the foundation for calibration problems with binary responses. This work can be extended in several directions. First, it can be extended to other non-Gaussian data, such as count data. To do so, the logistic function g in (2.1) can be replaced by other link functions of the exponential family type, such as the log function for Poisson distribution. The true process ξ can then be estimated by maximizing the objective function in (2.3) where

the likelihood function of the Bernoulli distribution is replaced by other exponential family distributions. The theoretical results in Section 2.3, however, cannot be directly applied to other exponential family distributions. Moreover, aside from the proposed frequentist framework, a Bayesian framework for binary responses is worth exploring. In the recent literature, the nonidentifiability issue in typical Bayesian framework has been addressed for continuous responses, such as the orthogonal Gaussian process [75] and the projected kernel calibration [76]. Extensions of these methods to binary responses deserve further attention.

CHAPTER 3
EXPLOITING VARIANCE REDUCTION POTENTIAL IN LOCAL GAUSSIAN PROCESS SEARCH

3.1 Introduction

Due to continual advances in computational capabilities, researchers across fields increasingly rely on computer simulations in lieu of prohibitively costly or infeasible physical experiments. One example is [77], who use computer simulations to investigate the interaction of energetic particles with solids. Physical effects such as elastic energy loss when a particle penetrates a solid, particle transmission through solids, and radiation damage are explored. These processes can be approximated by simulating the trajectories of all moving particles in a solid based on mathematical models. An example in linguistics is the study of language evolution [78], which is made challenging by the unobserved nature of language origin. Modeling techniques such as genetic algorithms can be used to simulate the process of natural selection and make it possible to explore a virtual evolution. While computer simulations provide a feasible alternative to many physical experiments, simulating from mathematical models is often itself expensive, in terms of both time and computation, and many researchers seek inexpensive approximations to their computationally demanding computer models—so-called *emulators*.

Gaussian process (GP) models [6] play an important role as emulators for computationally expensive computer experiments. They provide an accurate approximation to the relationship between simulation output and untried inputs at a reduced computational cost, and provide appropriate (statistical) measures of predictive uncertainty. A major challenge in building a GP emulator for a large-scale computer experiment is that it necessitates decomposing a large ($N \times N$) correlation matrix. For dense matrices, this requires around $O(N^3)$

time, where N is the number of experimental runs. Inference for unknown parameters can demand hundreds of such decompositions to evaluate the likelihood, and its derivatives, under different parameter settings for even the simplest Newton-based maximization schemes. This means that for a computer experiment with as few as $N = 10^4$ input-output pairs, accurate GP emulators cannot be constructed without specialized computing resources.

There are several recent approaches to emulating large-scale computer experiments, most of which focus on approximation of the GP emulator due to the infeasibility of actual GP emulation. Examples include covariance tapering which replaces the dense correlation matrix with a sparse version [79], multi-step interpolation which successively models global, then more and more local behavior while controlling the number of non-zero entries in the correlation matrix at each stage [80], and multiresolution modeling with Wendland's compactly supported basis functions [81]. Alternatively, [82] developed an R package called `bigGP` that combines symmetric-multiprocessors and GPU facilities to handle N as large as $67,275$ without approximation. Nevertheless, computer model emulation is meant to avoid expensive computer simulation, not be a major consumer of it. Another approach, proposed by [83], is to sample input-output pairs according to a specific design structure, which leads to substantial savings in building a GP emulator. That method, however, can be limited in practice due to the restriction to sparse grid designs.

In this chapter, [2]'s local GP approach is considered. The approach is modern, scalable, and easy to implement with limited resources. The essential idea focuses on approximating the GP emulator at a particular location of interest via a relatively small subset of the original design, thus requiring computation on only a modest subset of the rows and columns of the large $(N \times N)$ covariance matrix. This process is then repeated across predictive locations of interest, ideally largely in parallel. The determination of this local subset for each location of interest is crucial since it greatly impacts the accuracy of the corresponding local GP emulator. [2] proposed a greedy search to sequentially augment the subset according to an appropriate criteria and that approach yields reasonably accurate GP emulators. More

details are presented in Section 3.2.

A bottleneck in this approach is that a complete iterative search for the augmenting point requires looping over $O(N)$ data points at each iteration. In Section 3.3, motivated by the intuition that there is little potential benefit in including a data point far from the prediction location, two new neighborhood search limiting techniques are proposed, the *maximum distance method* and the *feature approximation method*. Two examples in Section 3.4 show that the proposed methods substantially speed up the local GP approach while retaining its accuracy. A brief discussion follows in Section 3.5. Mathematical proofs are provided in Appendix C.

3.2 Preliminaries

3.2.1 Gaussian Process Model

A Gaussian process (GP) is a stochastic process whose finite dimensional distributions are defined via a mean function $\mu(x)$ and a covariance function $\Sigma(x, x')$, for d-dimensional inputs x and x'. In particular, for N input x-values, say X_N, which define the N-vector $\mu(X_N)$ and $N \times N$ matrix $\Sigma(X_N, X_N)$, and a corresponding N-vector of responses Y_N, the responses have distribution $Y_N \sim \mathcal{N}(\mu(X_N), \Sigma(X_N, X_N))$. The scale $\sigma^2 > 0$ is commonly separated from the process correlation function, $Y_N \sim \mathcal{N}(\mu(X_N), \sigma^2\Phi(X_N, X_N))$, where the $N \times N$ matrix $\Phi(X_N, X_N) = (\Phi(x_i, x_j))$ is defined in terms of a correlation function $\Phi(\cdot, \cdot)$, with $\Phi(x, x) = 1$. As an example, consider the often-used separable *Gaussian correlation function*

$$\Phi_\Theta(x, x') = \exp\left\{-\sum_{j=1}^{d}(x_j - x'_j)^2/\theta_j\right\}, \text{where } \Theta = (\theta_1, \ldots, \theta_d),\ \theta_j > 0, j = 1, \ldots, d. \tag{3.1}$$

As this correlation decays exponentially fast in the squared distance between x_j and x'_j at rate θ_j, the sample paths are infinitely differentiable and the resulting predictor is an interpolator.

The GP model is popular because inference for $\mu(\cdot), \sigma^2$, and Θ is easy and prediction is highly accurate. A popular inferential choice is maximum likelihood, with corresponding log likelihood (up to an additive constant)

$$\ell(\mu, \sigma^2, \Theta) = -\frac{1}{2}\left\{n\log(\sigma^2) + \log(\det(\Phi_\Theta(X_N, X_N))) + (Y_N - \mu(X_N))^T \Phi_\Theta(X_N, X_N)^{-1}(Y_N - \mu(X_N))/\sigma^2\right\}$$

and the MLEs of $\mu(\cdot), \sigma^2$, and Θ are

$$(\hat{\mu}(\cdot), \hat{\sigma}^2, \hat{\Theta}) = \arg\max_{\mu, \sigma^2, \Theta} \ell(\mu, \sigma^2, \Theta). \tag{3.2}$$

Here, $\mu(\cdot)$ and its estimate are described somewhat vaguely. Common choices are $\mu(\cdot) \equiv 0$, $\mu(\cdot) = \mu$, or $\mu(\cdot) = h(\cdot)^T\beta$, for a vector of relatively simple basis functions $h(\cdot)$. More details on inference can be found in [84] or [5]. Importantly, the predictive distribution of $Y(x)$ at a new setting x can be derived for fixed parameters by properties of the conditional multivariate normal distribution. In particular, it can be shown that $Y(x)|X_N, Y_N \sim \mathcal{N}(\mu_N(x), V_N(x))$, where

$$\mu_N(x) = \mu(x) + \Phi_\Theta(x, X_N)\Phi_\Theta(X_N, X_N)^{-1}(Y_N - \mu(X_N)), \tag{3.3}$$

$$V_N(x) = \sigma^2(\Phi_\Theta(x, x) - \Phi_\Theta(x, X_N)\Phi_\Theta(X_N, X_N)^{-1}\Phi_\Theta(X_N, x)). \tag{3.4}$$

In a practical context, the parameters $\mu(\cdot)$, σ^2, and Θ can be replaced by their estimates (3.2) and it can be argued that the corresponding predictive distribution is better approximated by a t-distribution than normal (see 4.1.3 in [5]). Either way, $\hat{\mu}_N(x)$ is commonly taken as the *emulator*, and $V_N(x)$ captures uncertainty.

3.2.2 Local Gaussian Process Approximation

A major difficulty in computing the emulator (3.3) and its predictive variance (3.4) is solving the linear system $\Phi_{\hat{\Theta}}(X_N, X_N)y = \Phi_{\hat{\Theta}}(X_N, x)$, since it requires $O(N^2)$ storage and around $O(N^3)$ computation for dense matrices. A promising approach is to search small sub-designs that approximate GP prediction and inference from the original design [2]. The idea of the method is to focus on prediction at a location, x, using a subset of the full data $X_n(x) \subseteq X_N$. Intuitively, the sub-design $X_n(x)$ may be expected to be comprised of X_N close to x. For typical choices of $\Phi_\Theta(x, x')$, correlation between elements x, x' in the input space decays quickly for x' far from x, and x''s that are far from x have vanishingly small influence on prediction. Ignoring them in order to work with much smaller, $n \times n$ matrices brings big computational savings, ideally with little impact on accuracy. Figure 3.1 displays a smaller sub-design ($n = 7$) near location $x = 0.5$ extracted from the original design ($N = 21$). Although the emulator (dotted line) performs very poorly from 0 to 0.3 and from 0.6 to 1.0, the sub-design provides accurate and robust prediction at $x = 0.5$.

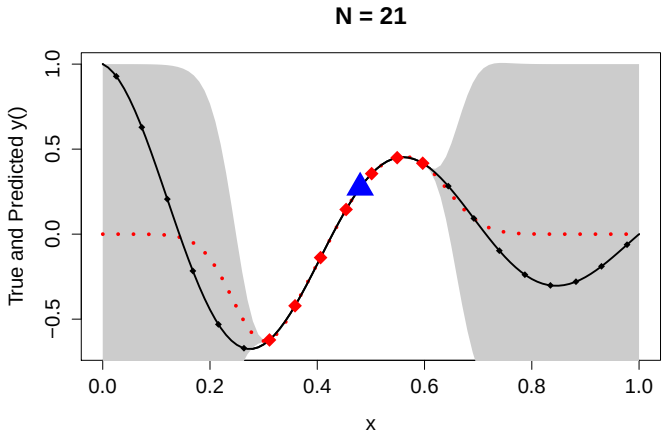


Figure 3.1: An example sub-design $X_7(x)$ for a one dimensional input. Dots represent the full design, X_{21}, the triangle represents the point of interest $x = 0.5$ and the diamonds represent the sub-design, $X_7(x)$. Based on the sub-design $X_7(x)$, the emulator is represented as the dotted line, with the shaded region providing a pointwise 95% confidence band.

For an accurate and robust emulator, a smaller predictive variance (3.4) for each x

is desirable. We seek a small sub-design $X_n(x) \subseteq X_N$ for each location of interest x, which minimizes the predictive variance (3.4) corresponding to the sub-design $X_n(x)$. This procedure is then repeated for each location of interest x. The identification of sub-designs and subsequent prediction at each such x can be parallelized immediately, providing a substantial leap in computational scalability. However, searching for the optimal sub-design, which involves choosing n from N input sites, is a combinatorially huge undertaking. A sensible idea is to build up $X_n(x)$ by *n nearest neighbors* (NNs) close to x, and the result is a valid probability model for $Y(x)|X_n(x), Y(X_n(x))$ [85]. [2] proposed a greedy, iterative search for the sub-design, starting from a small NN set X_{n_0} and sequentially choosing the x_{j+1} which provides the greatest reduction in predictive variance to augment $X_j(x)$, for $j = n_0, n_0 + 1, \ldots, n$. That is,

$$x_{j+1} = \underset{\substack{u \in X_N \setminus X_j(x), \\ X_{j+1} = X_j(x) \cup u}}{\arg\min} \; V_{j+1}(x) \tag{3.5}$$

and $X_{j+1}(x) = X_j(x) \cup x_{j+1}$. Both the greedy and NN schemes can be shown to have computational order $O(n^3)$ (for fixed N) when the scheme is efficiently deployed for each update $j \to j + 1$. Specifically, the matrix inverse $\Phi_\Theta(X_{j+1}, X_{j+1})^{-1}$ in $V_{j+1}(x)$ can be updated efficiently using partitioned inverse equations [86]. Before the greedy subsample selection proceeds, correlation parameters can be initialized to reasonable fixed values to be used throughout the sub-design search iterations. After a sub-design has been selected for a particular location, a local MLE can be constructed. Thus, only $O(n^3)$ cost is incurred for building the local subset and subsequent local parameter estimation. For details and implementation, see the `laGP` package for R [87]. An initial *overall* estimate of the correlation parameters can be obtained using the Latin hypercube design-based block bootstrap subsampling scheme proposed by [88], which has been shown to consistently estimate *overall* lengthscale θ_j-values in a computationally tractable way, even with large N.

The greedy scheme, searching for the next design point in $X_N \setminus X_j(x)$ to minimize the predictive variance (3.5), is still computationally expensive, especially when the design size N is large. For example, the new x_{j+1} based on (3.5) involves searching over $N - j$ candidates. In that case, the greedy search method still contains a serious computational bottleneck in spite of its improvements relative to solving the linear system in (3.3) for GP prediction and inference. [89] recognized this issue and accelerated the search by exporting computation to graphical processing units (GPUs). They showed that the GPU scheme with local GP approximation and massive parallelization can lead to an accurate GP emulator for a one-million-run full design, with the GPUs providing approximately an order of magnitude speed increase. [90] noticed that the progression of $x_{j+1}, j = 1, 2, \ldots$ *qualitatively* takes on a ribbon and ring pattern in the input space and suggested a computationally efficient heuristic based on one dimensional searches along rays emanating from the predictive location of interest x.

In Section 3.3, two computationally efficient and accuracy preserving neighborhood search methods are proposed. Both neighborhood searches reduce computation by decreasing the number of candidate design points examined. It is shown that only locations within a particular distance of either the prediction location x or the current sub-design, or locations in particular regions within a *feature space*, can have *substantial* influence on prediction. Using these techniques, it is possible to search a *much* smaller candidate set at each stage, leading to huge reductions in computation and increases in scalability.

3.3 Reduced Search in Local Gaussian Process

For prediction at location x, there is intuitively little benefit to considering input locations that are distant from x (relative to the correlation decay) as the response value at these locations is nearly independent of the response at x. In Section 3.3.1, a *maximum distance* bound and corresponding algorithm are provided, and in Section 3.3.2, a *feature approximation* bound and corresponding modification to the algorithm are provided. The algorithms furnish

a dramatically reduced set of potential design locations which need to be examined, in a computationally efficient and scalable manner. Notably, for the algorithms presented below, Θ is fixed. Updating of Θ could follow [2], where an overall estimate of Θ is generated initially, then the local design formed, then a *local* Θ estimated. The below algorithms, and subsquent complexity comparison, focus on updating the local design, while the updating of Θ is considered as an offline procedure.

3.3.1 Maximum Distance Method

Here x is the particular location of interest, in terms of emulation/prediction, and $X_j(x)$ is the greedy sub-design at stage j. To augment the sub-design $X_j(x)$, we downplay locations distant from x with little loss.

Assume that the underlying correlation function is radially decreasing after appropriate linear transformation of the inputs: there is a strictly decreasing function ϕ so that $\Phi_\Theta(x, x') = \phi(\|\Theta(x - x')\|_2)$ for some Θ. In practice, Θ can be estimated using the local MLE as discussed in Section 3.2.2, using as a starting value the *overall*, consistent estimate from the sub-design search iterations. Now, consider a candidate input location x_{j+1} at stage $j + 1$ of the greedy sub-design search for an input location to add to the design and take $d_{\min}(x_{j+1})$ to be the minimum (Mahalanobis-like) distance between the candidate point x_{j+1} and the current design and location of interest,

$$d_{\min}(x_{j+1}) = \min\{\|\Theta(x - x_{j+1})\|_2, \|\Theta(x_1 - x_{j+1})\|_2, \\ \|\Theta(x_2 - x_{j+1})\|_2, \ldots, \|\Theta(x_j - x_{j+1})\|_2\}. \tag{3.6}$$

We use the term Mahalanobis-*like* distance to emphasize that the rescaling and rotation of the inputs induced by Θ is not related to the variance-covariance matrix of the input locations. For example, consider the sub-design $X_j(x)$ with two-dimensional inputs shown in Figure 3.2 for $j = 8$.

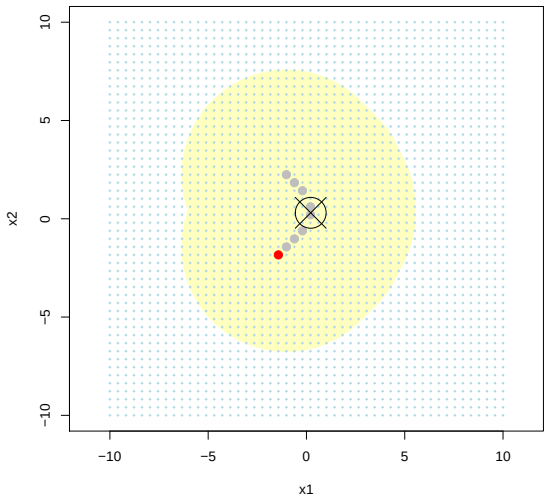


Figure 3.2: An example sub-design $X_8(x)$ with two-dimensional inputs. The circled $\times$ represents the location of interest. With $\Theta = \mathrm{diag}(1/\sqrt{3}, 1/\sqrt{3})$, the dots $\bullet$ represent current design points $X_8(x)$, the dot $\bullet$ represents the new input location x_9, and the shaded region represents the candidate points x_* with $d_{\min}(x_*) < 3.07$.

Based on the local design scheme introduced in Section 3.2 and (3.5), the sub-design $X_n(x)$ is built up through the choices of x_{j+1} to sequentially augment $X_j(x)$, at each stage aiming to minimize predictive variance. Proposition 3.3.1 provides an alternate formula for this variance, which is used to greatly reduce the number of candidates in the minimization problem.

Proposition 3.3.1. The predictive variance $V_j(x)$ in (3.4) can be represented via the recurrence

$$V_{j+1}(x) = V_j(x) - \sigma^2 R(x_{j+1}). \tag{3.7}$$

Here, $R(x_{j+1})$ is the (scaled) *reduction in variance*,

$$R(x_{j+1}) = \frac{(\Phi_\Theta(x, x_{j+1}) - \Phi_\Theta(x_{j+1}, X_j)\Phi_\Theta(X_j, X_j)^{-1}\Phi_\Theta(X_j, x))^2}{\Phi_\Theta(x_{j+1}, x_{j+1}) - \Phi_\Theta(x_{j+1}, X_j)\Phi_\Theta(X_j, X_j)^{-1}\Phi_\Theta(X_j, x_{j+1})}. \tag{3.8}$$

The recurrence relation (3.7) is useful for searching candidates to entertain. Further, minimizing variance after adding the new input location x_{j+1} is equivalent to maximizing

reduction in variance $R(x_{j+1})$.

Theorem 3.3.2. *Suppose $\Phi : \Omega \times \Omega \to \mathbb{R}$ is a symmetric positive-definite kernel on a compact set $\Omega \subseteq \mathbb{R}^d$ and there exists a strictly decreasing function $\phi : \mathbb{R}^+ \to \mathbb{R}$ such that $\Phi_\Theta(x, y) = \phi(\|\Theta(x - y)\|_2)$ for some Θ. Then, for $\delta > 0$, $R(x_{j+1}) \leq \delta$ if*

$$d_{\min}(x_{j+1}) \geq \phi^{-1}\left(\sqrt{\frac{\delta}{(1+\sqrt{j}\|\Phi_\Theta(X_j, X_j)^{-1}\Phi_\Theta(X_j, x)\|_2)^2 + j\delta/\lambda_{\min}}}\right), \tag{3.9}$$

where $\lambda_{\min}$ is the minimum eigenvalue of $\Phi_\Theta(X_j, X_j)$.

This result indicates that candidate locations that are sufficiently distant from the location of interest and the current sub-design do not have potential to reduce the variance more than δ. Importantly, if the full set of design locations X_N is stored in a data structure such as a *k-d tree* [91], then the set of candidate locations that do not satisfy (3.9) can be identified in $O(\log N)$ time, with constant depending on δ, dimension of the input space, and stage j; This provides a computationally efficient and readily scalable technique for reducing the set of potential candidate locations.

Our Algorithm 1 is a starting point for efficiently selecting sub-designs for prediction at location x. In the algorithm, a larger value of δ is desirable since this leads to fewer candidate design locations to search. One way to obtain a relatively large value of δ is to examine the variance reductions on the set of k nearest neighbors which are not yet in the sub-design, as is shown in Step 2. The number of nearest neighbors k is a tuning parameter. A larger value of k provides a larger variance reduction and therefore excludes more candidate design locations, albeit at an additional computational expense since the variance reduction must be checked at each of these locations. Intuitively, larger-scale and higher-dimensional problems might be expected to benefit from a larger k—larger-scale problems because the cost of computing the variance reductions across k might be relatively modest compared to the potential cost of computing the variance reductions across a large candidate set, higher-dimensional problems because more points are needed to explore the

surface of the hyper-sphere of points near the location of interest. A large value of δ can be obtained by applying the heuristic proposed in [90]. From the result of Theorem 3.3.2, $T(X_j)$ in Step 3, which indicates the region such that

$$d_{\min}(x_{j+1}) \leq \phi^{-1}\left(\sqrt{\frac{\delta}{(1+\sqrt{j}\|\Phi_\Theta(X_j,X_j)^{-1}\Phi_\Theta(X_j,x)\|_2)^2 + j\delta/\lambda_{\min}}}\right), \quad (3.10)$$

gives the subset of candidate locations that have potential to reduce the variance more than δ.

For each update $j \to j+1$, the algorithm *ideally* involves $O(j^2 + j\log N)$ computations in Step 3: $O(j\log N)$ for eliminating search locations, and $O(j^2)$ for computing the right-hand side of (3.10). In particular, the matrix inverse $\Phi_\Theta(X_j,X_j)^{-1}$ can be updated via the partitioned inverse equations [86] with $O(j^2)$ cost at each iteration. Analysis of the computational complexity of obtaining (an approximation to) the minimum eigenvalue of $\Phi_\Theta(X_j,X_j)$ is more challenging. It is convenient to work with the reciprocal of the maximum eigenvalue of $\Phi_\Theta(X_j,X_j)^{-1}$, for which relatively efficient algorithms, such as the power or Lanczos method, exist [92]. If the starting vector is not orthogonal to the target eigenvector, then convergence of the (less efficient, but easier to analyze) power method is geometric with rate depending on the ratio between the two largest eigenvalues of $\Phi_\Theta(X_j,X_j)^{-1}$ (see equation 9.1.5 in [92]). While this rate and the constants in front are not fixed across j, they can be bounded, with the exception of the influence of the starting vector, across all subsets of the full dataset. The starting vector might be expected to be increasingly collinear with the target eigenvector as j increases, thereby improving the rate bound. All together this implies an approximately constant number of iterations, each costing $O(j^2)$, is required to approximate $\lambda_{\min}$ for each j. Another perspective chooses a random starting vector, for which [93] provide respective average and probabilistic bounds of $O(j^2\log j)$ for the power method and $O(j^2\log^2 j)$ for the Lanczos method. If standard eigen-decomposition routines that return all the eigenvalues are used, then the j^2 term in

the computational complexity is j^3. The inverse function $\phi^{-1} : \mathbb{R} \to \mathbb{R}$ can be computed in roughly constant time by a root-finding algorithm or even computed exactly for many choices of Φ. For example, for the power correlation function, $\Phi_\Theta(x, y) = \exp\{-\|\Theta(x - y)\|_2^p\}$, the ϕ can be formed as $\phi(u) = \exp\{-u^p\}$, so $\phi^{-1}(v) = (-\log v)^{1/p}$. When a large n is required, computation of $\lambda_{\min}$ might be numerically unstable. A remedy in that case may be to stop the search when $\lambda_{\min}$ falls below a prespecified threshold, or perhaps to introduce a penalty inversely proportional to $\lambda_{\min}$.

Algorithm 1 Maximum distance search method in local Gaussian process.

1: Set $j = 1$ and x_1 as the point closest to the predictive location x. Throughout, let $X_j(x) \equiv X_j = \{x_1, x_2, \ldots, x_j\}$, dropping the explicit (x) argument.

2: Let $N_{jk}(x)$ denote the k nearest neighbors to x in $X_N \setminus X_j$, the candidate locations not currently in the sub-design. Set δ_{j+1} equal to the maximum variance reduction from $N_{jk}(x)$,

$$\delta_{j+1} = \max_{u \in N_{jk}(x)} R(u), \tag{3.11}$$

where $R(\cdot)$ is shown in (3.8).

3: Set $y = \phi^{-1}\left(\sqrt{\frac{\delta_{j+1}}{(1+\sqrt{j}\|\Phi_\Theta(X_j,X_j)^{-1}\Phi_\Theta(X_j,x)\|_2)^2 + j\delta_{j+1}/\lambda_{\min}}}\right)$, where $\Phi_\Theta(x, x') = \phi(\|\Theta(x - x')\|_2)$ and $\lambda_{\min}$ is the minimum eigenvalue of $\Phi_\Theta(X_j, X_j)$. For

$$T(X_j) = \{u \in X_N \setminus X_j : \|\Theta(u - v)\|_2 \le y \text{ for some } v \in \{x, X_j\}\}, \tag{3.12}$$

take

$$x_{j+1} = \arg \max_{u \in T(X_j)} R(u).$$

4: Set $j = j + 1$ and repeat 2 and 3 until either the reduction in variance $R(x_{j+1})$ falls below a prespecified threshold, or the local design budget is met.

3.3.2 Feature Approximation Method

In addition to the maximum distance method and associated algorithm, an approximation via eigen-decomposition can be applied to reduce the potential locations in a computationally efficient manner. Suppose that Φ is a symmetric positive-definite kernel on a compact set

$\Omega \subseteq \mathbb{R}^d$ and $P : L_2(\Omega) \to L_2(\Omega)$ is the integral operator

$$Pv(x) := \int_\Omega \Phi(x, y)v(y)dy, \quad v \in L_2(\Omega), \quad x \in \Omega. \tag{3.13}$$

Mercer's theorem guarantees the existence of a countable set of positive eigenvalues $\{\lambda_j\}_{j=1}^\infty$ and an orthonormal set $\{\varphi_j\}_{j=1}^\infty$ in $L_2(\Omega)$ consisting of the corresponding eigenfunctions of P, $P\varphi_j = \lambda_j\varphi_j$ [71]. The eigenfunctions φ's here are continuous on Ω and Φ has the absolutely and uniformly convergent representation

$$\Phi(x, y) = \sum_{j=1}^{\infty} \lambda_j\varphi_j(x)\varphi_j(y).$$

In particular, Φ can be approximated uniformly over inputs in terms of a finite set of eigenfunctions

$$\Phi(x, y) \approx \sum_{j=1}^{D} \lambda_j\varphi_j(x)\varphi_j(y) \tag{3.14}$$

for some moderately large integer D. For some kernel functions, closed form expressions exist. For example, the Gaussian correlation function (3.1) (on $\mathbb{R}^d$, with weighted integral operator) has eigenfunctions given by products of Gaussian correlations and Hermite polynomials [94]. More generally, [95] show high-quality approximations to these eigen-decompositions can be obtained via Nyström's method.

Theorem 3.3.3. *Assume $\Phi : \Omega \times \Omega \to \mathbb{R}$ is a symmetric positive-definite kernel on a compact set $\Omega \subseteq \mathbb{R}^d$ which can be approximated via D eigenfunctions* (3.14)*. Then, the reduction in variance* (3.8) *has approximate representation*

$$R(x_{j+1}) \approx \|C_{X_j}(x)\|_2^2 \cos^2(\vartheta), \tag{3.15}$$

where ϑ is the angle between $C_{X_j}(x)$ and $C_{X_j}(x_{j+1})$,

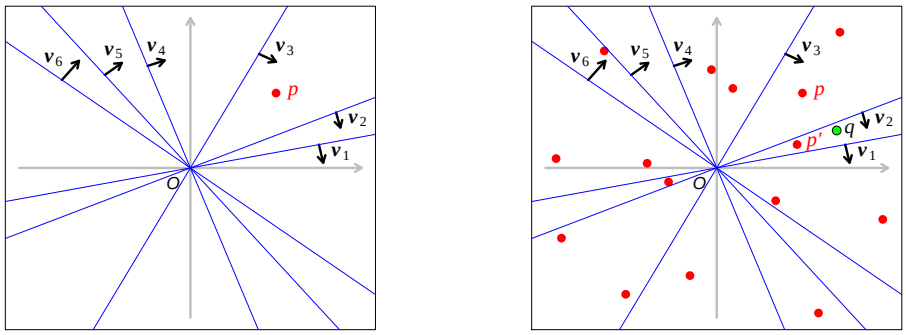
$$
\begin{aligned}
C_{X_j}(t) &= [I - U(X_j)[U^T(X_j)U(X_j)]^{-}U^T(X_j)]U(t), \\
U(t) &= \left(\sqrt{\lambda_1}\varphi_1(t), \ldots, \sqrt{\lambda_D}\varphi_D(t)\right)^T, \quad \text{and} \\
U(X_j) &= \left[U(x_1), \ldots, U(x_j)\right],
\end{aligned}
$$

for eigenfunctions $\varphi_i(t)$, $t \in \Omega$ and corresponding ordered eigenvalues $\lambda_1 \geq \ldots \geq \lambda_D$.

According to this approximation, the candidate set can be reduced by transforming the inputs into a feature space. A modified algorithm is suggested as follows. The variance reduction threshold in (3.11) now places a restriction on the *angle* between $C_{X_j}(x)$ and $C_{X_j}(x_{j+1})$, where we would like to exclude points *outside* the cones

$$\cos^2(\vartheta) \leq \frac{\delta_{j+1}}{\|C_{X_j}(x)\|_2^2}. \tag{3.16}$$

A feature approximation is shown in Algorithm 2. To reduce the computational burden in checking (3.16), the values of the first D eigenfunctions at the full dataset X_N, $U(X_N)$, could be computed in advance and stored based on a locality-sensitive hashing (LSH) scheme [96]. LSH is a method for answering approximate similarity-search queries in high-dimensional spaces. The basic idea is to use special locality-sensitive functions to *hash* points into "buckets" such that "nearby" points map to the same bucket with high probability. Many similarity measures have corresponding LSH functions that achieve this property. For instance, the hashing functions for cosine-similarity are the normal vectors of random hyperplanes through the origin, denoted for example as $v_1, \ldots, v_k$. Depending on its side of these random hyperplanes, a point p is placed in bucket $h_1(p), \ldots, h_k(p)$, where $h_i(p) = \text{sign}(v_i^T p)$. A simple example, following [97], is provided in Figure 3.3. Figure 3.3a illustrates the hashing process for a point p, where the point p is hashed into the bucket $(h_1(p), \ldots, h_6(p)) = (-1, -1, 1, 1, 1, 1)$ by the definition $h_i(p) = \text{sign}(v_i^T p), i = 1, \ldots, 6$



(a) Hashing. (b) Search.

Figure 3.3: Illustration of locality-sensitive hashing (LSH) scheme. Lines — are random hyperplanes through origin, and $v_1, \ldots, v_6$ (arrows $\rightarrow$) are the corresponding normal vectors. Dots ● present stored data points, and the dot ● with circle presents the query data point.

(when the point p is above the hyperplane, the inner product is negative, otherwise the inner product is positive). Similarly, other points are placed in their corresponding buckets. In the search process, shown in Figure 3.3b, the query point q is mapped to the bucket $(h_1(q), \ldots, h_6(q)) = (-1, 1, 1, 1, 1, 1)$, which matches the bucket of point p'. Thus, the hashing and search processes retrieve p' as the *most similar* neighbor of q. Also, since the one different label in the buckets of p and q implies that the angular difference is close to $\pi/6$ (six hyperplanes), p is retrieved when querying the points whose angular difference from q is less than $\pi/6$. Many more than six hyperplanes are needed to ensure that the returned angle similarity is approximately correct. In a standard LSH scheme, the hashing process is performed several times by different sets of random hyperplanes, and the search procedure iterates over these random sets of hyperplanes. More details and examples can be seen in [96],[97], and [98].

Apart from cosine-similarity, [99] showed for the pairwise similarity

$$\frac{y_k^T A_j y_h}{\|G_j y_k\|_2 \|G_j y_h\|_2},$$

where $y_k, y_h \in \mathbb{R}^d$, $G_j^T G_j = A_j$ and A_j is a $d \times d$ positive-definite matrix that is updated

for each iteration j, the hash function can be defined as:

$$h_{A_j}(y) = \begin{cases} 1 & r^T G_j y \geq 0 \\ 0 & \text{otherwise} \end{cases}, \tag{3.17}$$

where the vector r is chosen at random from a d-dimensional Gaussian distribution. Let $G_j = I - U(X_j)[U^T(X_j)U(X_j)]^- U^T(X_j)$ and $A_j = G_j$ (G_j is symmetric and idempotent), then $\cos(\vartheta)$ in (3.15) can be represented as

$$\cos(\vartheta) = \frac{U(x_{j+1})^T A_j U(x)}{\|G_j U(x_{j+1})\|_2 \|G_j U(x)\|_2}.$$

Thus, in the feature approximation method, an LSH scheme can be employed by storing $U(X_N)$ in advance and updating the hash function (3.17) at each iteration, where y is replaced by $U(y)$. At query time, similar points are hashed to the same bucket with the query $U(x)$ and the results are guaranteed to have a similarity within a small error after repeating the procedure several times. In particular, for each update $j \to j+1$, given that the LSH method guarantees retrieval of points within the radius $(1+\epsilon)M$ from the query point $U(x)$, where M is the distance of the true nearest neighbor from $U(x)$, the method requires $O(D^2 + jDN^{1/(1+\epsilon)})$ computational cost, $O(D^2)$ for updating matrix G_j (via the partitioned inverse equations [86]) and computing the hash function $h_{A_j}(y)$ (via the implicit update in [99]), and $O(jDN^{1/(1+\epsilon)})$ for identifying the hashed query [99], where D is the number of eigenfunctions in Theorem 3.3.3. In Section 3.4, two examples show the benefit from the LSH approach in the feature approximation method.

As an illustration of how cones in feature space relate to the design space, consider a full design X_N consisting of 2500 $\mathrm{Unif}(0,1)$ data points, plotted in dots in the *left* panel of Figure 3.4. The correlation function is $\Phi(x,x') = \exp\{-\|(x-x')/10\|_2^2\}$ and the predictive location of interest is $x = (0.5, 0.5)$, shown as a triangle in the *left* panel. The first 7 design points are chosen greedily and indicated with numbers. The *right* panel shows the first two

Algorithm 2 Feature approximation modification to Algorithm 1.

In Step 3 of Algorithm 1, replace $T(X_j)$ with $T^*(X_j)$, where

$$T^*(X_j) = \{u \in X_N \setminus X_j : \|\Theta(u-v)\|_2 \leq y \text{ and } \cos^2(\vartheta) \geq \delta_{j+1}/\|C_{X_j}(x)\|_2^2 \text{ for some } v \in \{x, X_j\}\},$$

and ϑ, $C_{X_j}(x)$ are defined in Theorem 3.3.3. Then,

$$x_{j+1} = \arg \max_{u \in T^*(X_j)} R(u).$$

components of the feature space (the first two eigenfunctions evaluated at the design points), labeled correspondingly. The vector $C_{X_7}(x)$ is denoted as the middle dotted line in the *right* panel, with $|\vartheta| \leq \pi/20$ shown as the outer dotted lines. Design points falling within these cones are shown in shaded region in both panels. The design points in the *left* panel which fall in the stripe have the most potential to reduce predictive variance.

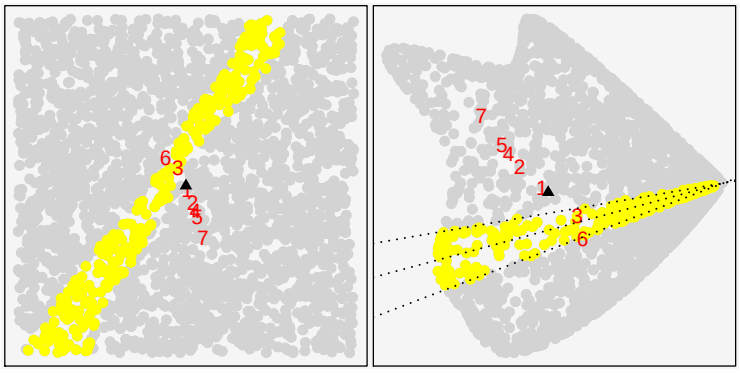


Figure 3.4: Dots ● and ● represent design points in the original space (*left*) and a $D = 2$ dimensional feature space approximation (*right*). Location of interest and current design are annotated with triangle and numbers, respectively. Vector $C_{X_7}(x)$ and cones $|\vartheta| \leq \pi/20$ shown with dotted lines. Design points falling within these cones are shown in shaded region in both panels.

The storage requirements and computational complexity of each update $j \to j+1$ at each predictive location of interest for the proposed algorithms are summarized in Table 3.1. Notably, if n local data points are needed, then the updating costs are incurred for $j = 1, \ldots, n$. Costs across predictive locations are ideally incurred largely in parallel. Here, the original greedy approach proposed in [2] is referred to as exhaustive search. Recall

Table 3.1: Complexity comparison between exhaustive search and two proposed methods for each update $j \to j+1$. The notation $|\cdot|$ denotes the cardinality of a set, and ϵ is a pre-specified value for the LSH method. *The complexity of pre-computation for feature approximation method is $O(D^3)$.

	Exhaustive Search	Maximum Distance Method		Feature Approximation Method with D Features*	
		w/o k-d tree	w/ k-d tree	w/o LSH	w/ LSH
Storage	N	N	N	ND	ND
Overhead		$O(j^2+jN)$	$O(j^2+j\log(N))$	$O(j^2+D^2N)$	$O(j^2+D^2+jDN^{1/(1+\epsilon)})$
Search	$O(j^2N)$	$O(j^2\lvert T(X_j)\rvert)$	$O(j^2\lvert T(X_j)\rvert)$	$O(j^2\lvert T^*(X_j)\rvert)$	$O(j^2\lvert T^*(X_j)\rvert)$

that $T(X_j)$ and $T^*(X_j)$ are the candidate sets from maximum distance method and feature approximation method, respectively. Let $|\cdot|$ denote the cardinality of a set. Since $|T(X_j)|$ and $|T^*(X_j)|$ are expected to be much smaller than N, the computational cost of the two proposed algorithms can be substantially reduced at each stage j relative to the original greedy search. Overhead cost, for computing benchmarks and eliminating search locations, is required for both methods. Also, with a k-d tree or LSH search method, the specially adapted data structure indeed improves overhead computational costs ($O(j^2+jN) \to O(j^2+j\log(N))$ and $O(j^2+D^2N) \to O(j^2+D^2+jDN^{1/(1+\epsilon)})$, respectively). Considering the two proposed methods, $|T^*(X_j)|$ might be expected to be much smaller than $|T(X_j)|$ if the correlation function is well approximated by the finite set of eigenfunctions and eigenvalues, and the dimension of input is not too large, since distance becomes a very powerful exclusion criteria in even moderately high-dimensional space. The maximum distance method has smaller storage and overhead requirements. Section 3.4 presents two examples implementing the two proposed methods and shows the comparison.

3.4 Examples

Two examples are discussed in this section: a two-dimensional example that demonstrates the algorithm and visually illustrates the reduction of candidates; a larger-scale, higher-dimensional example. Both examples show the proposed methods considerably outperform-

ing the original search method with respect to computation time. All numerical studies were conducted using R [19] on a laptop with 2.4 GHz CPU and 8GB of RAM. The k-d tree and LSH were implemented via R package `RANN` [100] and modifications to the source code of the Python package `scikit-learn` [101, 102], and accessed in R through the `rPython` package [103]. Notably, the Lanczos method offered little consistent advantage for the scale of local datasets ($j \leq 30$) entertained below, and relatively typical of practical situations. As such, reported timings correspond to computing $\lambda_{\max}\left(\Phi(X_j, X_j)^{-1}\right)$ via the full eigen-decomposition using `eigen`.

3.4.1 Two-dimensional problem of size $N = 50^2$

Consider a computer experiment with full set of design locations X_N consisting of a regular 50×50 grid on $[-10, 10]^2$ (2500 design points, light small dots in Figure 3.5), and take the predictive location of interest x to be $(0.216, 0.303)$ (circled $\times$ in Figure 3.5). Set $\sigma^2 = 1$, and consider the Gaussian correlation function

$$\Phi_\Theta(x, y) = \exp\left\{-\left(\frac{(x_1 - y_1)^2}{\theta_1} + \frac{(x_2 - y_2)^2}{\theta_2}\right)\right\},$$

with $\theta_1 = \theta_2 = 3$. This correlation function implies the ϕ in Algorithm 1 is $\phi(u) = \exp\{-u^2\}$ and $\Theta = \mathrm{diag}(1/\sqrt{\theta_1}, 1/\sqrt{\theta_2})$. Then we have $\phi^{-1}(v) = \sqrt{-\log v}$.

Figure 3.5 illustrates the sub-design selection procedure shown in Algorithm 1, in which $k = 8$ nearest neighbors (from the candidate set) are used to generate the threshold in Step 2. In Figure 3.5, the dots represent the current design $X_j(x)$ and the optimal augmenting point x_{j+1}, and the points that are excluded from the search for that location are those which fall outside the shaded region. The panels in the figure correspond to greedy search steps $j \in \{3, 16, 29\}$. Notably, the optimal additional design points illustrated in Figure 3.5 are not always the nearest neighbors to the location of interest. In this example, only 7.40% (185/2500) of candidates need to be searched in the beginning. Even after

choosing thirty data points, there is no need to search much more than half of the full data (56.92%=1423/2500).

Continuing the same example, Figure 3.5 shows substantial improvement from the feature approximation method. In the example, a $D = 500$-dimensional feature space approximation is pre-computed using Nyström's method [95]. The points annotated with $+$s are the points that are not excluded from the search. In fact, the number of candidates which need to be searched is usually reduced at least 10-fold and in many cases 50- or 100-fold, or more.

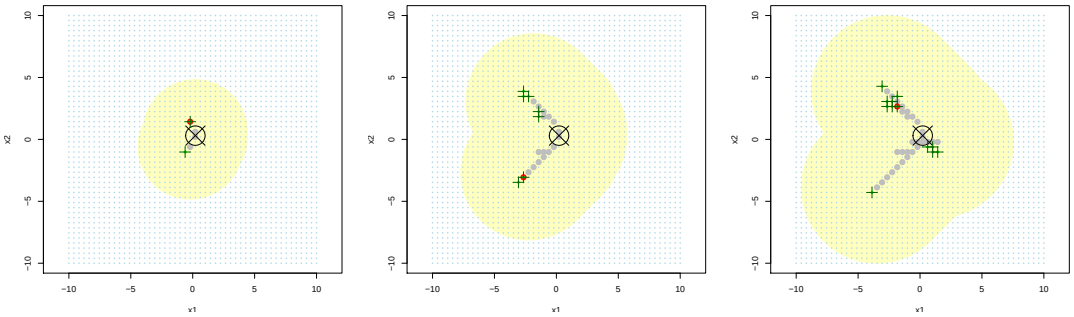


Figure 3.5: Left, middle, and right panels respectively illustrate selection at $j = 3, 16$, and 29. The circled $\times$ is the location of interest, $(0.216, 0.303)$. Dots $\bullet$ are the current design points; dots $\bullet$ are the optimal x_{j+1}; points which are excluded from the search based on maximum distance method are those which fall outside the shaded region. Points which are excluded from the search based on feature approximation method are those which are not annotated with a $+$.

While the maximum distance method and original greedy approach proposed in [2] produce the same sub-designs and, in turn, the same predictive variances, the feature approximation method is *approximate* and can produce different sub-designs and slightly different predictive variances (not *necessarily* inflated due to greedy nature of search). Table 3.2 shows relative differences in predictive variance resulting from feature approximation method with $D = 10, 200$, and 500 features as compared to take maximum distance method (or equivalently the original greedy approach). The number of search candidates is listed in

Table 3.2: The relative difference in variance of the emulator at location $(0.216, 0.303)$ between maximum distance search as a baseline and feature approximation search with number of features D: 10, 200 and 500. Baseline variance by maximum distance search is shown in the last column. The value in parentheses is the number of search candidates.

Relative Difference (# of searching candidates)	$D = 10$		$D = 200$		$D = 500$		Variance by Maximum Distance Method
Stage 10	0	(842)	0.178	(4)	0	(47)	1.95×10^{-6} (844)
Stage 15	0.006	(1057)	-0.76	(7)	0	(69)	9.35×10^{-7} (1040)
Stage 20	0.018	(1149)	-0.722	(30)	0	(12)	6.12×10^{-7} (1168)
Stage 25	-0.155	(1332)	-0.091	(4)	0	(116)	1.66×10^{-7} (1295)
Stage 30	0.009	(1459)	0.024	(20)	0	(2)	1.28×10^{-8} (1423)

parentheses. The relative difference in predictive variance is defined as

$$\frac{V_{j,FA}(x) - V_{j,MD}(x)}{V_{j,MD}(x)},$$

where $V_{j,FA}(x)$ and $V_{j,MD}(x)$ denote the predictive variance of the emulator at location x at stage j using the feature approximation method and maximum distance method, respectively. As might be expected, a larger number of features, $D = 500$, reduces the search candidates without any loss in variance reduction. For $D = 200$, although there are small differences in predictive variance, the discrepancies may be small enough to be of little practical consequence. At stage 15, 20, and 25, the predictive variance for $D = 200$ is even smaller than that of the maximum distance method, due to the greedy nature of the searches. Notably, if a small number of features, say $D = 10$, is chosen, feature approximation search offers little improvement over the maximum distance method in terms of search reduction, even though the predictive variances are similar to maximum distance method. In this case, $D = 200$ features might be a reasonable choice, balancing ease of computation and small predictive variance.

To further compare the performance of the proposed methods with the original greedy approach (exhaustive search), a Sobol's quasi-random sequence [104] of 100 predictive

locations was generated. Table 3.3 shows the average computation time and proportion of search candidates for the proposed methods and exhaustive search over the 100 predictive locations. The proportion of search candidates for the maximum distance method ran from 22.78% to 39.15%. The method also marginally speeds up computation time from 21 to 18 seconds with a k-d tree data structure. Although the feature approximation method needed 6 seconds for computing the features in advance, the proportion of search candidates for feature approximation method with $D = 200$ was reduced to 5.88% at stage 30. The computation time, on an ordinary laptop, was less than 15 seconds for 30 stages of iteration in the $N = 50^2$ experiment. Relative average predictive variance increases due to using the feature approximation method, both with and without LSH, are shown in Table 3.4. At stage 30 the average predictive variance increases due to using the feature approximation method are, with and without LSH, 4.6% and 1.4%, respectively, potentially small enough to be disregarded in a practical context. The LSH data structure also marginally reduces search time from 13 to 11 seconds. Recall that the feature approximation method with LSH approximates both the covariance function and the cosine similarity measure, so the candidate set is slightly different from the one without LSH. While in this moderately-sized problem the k-d tree and LSH data structures do not greatly improve the computational cost (at stage 30, k-d tree: $21 \to 18$, LSH: $13 \to 11$), in a larger-scale problem the improvements due to incorporating a k-d tree or LSH data structure can be relatively substantial.

3.4.2 6-dimensional problem of size $N = 5 \times 10^4$

Even more substantial reductions in the number of search candidates are seen for both methods in a larger-scale, higher-dimensional setting. In this example, we generated a 6-dimensional Sobol's quasi-random sequence of size $N = 5 \times 10^4$ in a $[-1, 1]^6$ for the design space and the predictive locations were chosen from a Sobol's quasi-random sequence of size 20. We took $\sigma^2 = 1$ and tuning parameter $k = 30$, and used the correlation function $\Phi_\Theta(x, y) = \exp\{-\sum_{i=1}^{6} \frac{(x_i - y_i)^2}{\theta_i}\}$ with $\theta_i = 1.5, i = 1, \ldots, 6$.

Table 3.3: Average time (seconds) comparison between exhaustive search and two proposed methods in two-dimensional setting with $N = 50^2$ over 100 Sobol predictive locations. The values in parentheses are the average percentage searched of full design. *Pre-computation time for feature approximation method is 6 seconds.

Seconds (Candidates %)	Exhaustive Search	Maximum Distance Method		*Feature Approximation Method with $D = 200$	
		w/o KD-tree	w/ KD-tree	w/o LSH	w/ LSH
Stage 10	11	3 (22.78%)	2 (22.78%)	3 (2.69%)	2 (1.98%)
Stage 15	19	5 (28.04%)	4 (28.04%)	5 (3.31%)	4 (2.90%)
Stage 20	30	9 (32.68%)	7 (32.68%)	8 (7.57%)	6 (5.56%)
Stage 25	44	14 (36.10%)	12 (36.10%)	10 (6.41%)	8 (5.66%)
Stage 30	61	21 (39.15%)	18 (39.15%)	13 (5.88%)	11 (6.65%)

Table 3.4: The relative difference in average predictive variance of the emulator between maximum distance search as a baseline and feature approximation search with number of features $D = 200$ over 100 Sobol predictive locations in 2-dimensional setting.

Relative Difference	Feature Approximation Method with $D = 200$		Average Variance by Maximum Distance Method
	w/o LSH	w/ LSH	
Stage 10	0.192	0.168	4.15×10^{-6}
Stage 15	0.348	0.140	1.08×10^{-6}
Stage 20	0.171	0.066	4.34×10^{-7}
Stage 25	0.011	-0.124	2.26×10^{-7}
Stage 30	0.046	0.014	1.62×10^{-7}

Table 3.5 shows the comparison between exhaustive search and the two proposed methods. The two methods outperform exhaustive search in terms of computation time. Further, the number of candidates searched for both methods are less than 10% $(= 5000/50000)$ across all 30 stages. While the exhaustive search took 3423 seconds ($\approx$ 1 hours) for 30 stage iterations, 240 seconds (4 minutes) were required for maximum distance method. Incorporating a k-d tree data structure, the computation time decreased to 193 seconds ($\approx$ 3.2 minutes). Compared to the 2-dimensional example in Section 3.4.1, incorporating a k-d tree data structure has moderately more computational benefit in this larger-scale setting.

The feature approximation method, as expected, has a smaller-sized candidate set than

Table 3.5: Time (seconds) comparison between exhaustive search and two proposed methods in 6-dimensional setting with $N = 5 \times 10^4$ over 20 Sobol predictive locations. The values in parentheses shows the percentage searched of full design. *Pre-computation time for feature approximation method was 26 seconds.

Seconds (Candidates %)	Exhaustive Search	Maximum Distance Method		*Feature Approximation Method with $D = 300$	
		w/o KD-tree	w/ KD-tree	w/o LSH	w/ LSH
Stage 10	488	24 (2.77%)	10 (2.77%)	74 (1.71%)	26 (1.7%)
Stage 15	953	50 (4.27%)	28 (4.27%)	126 (3.34%)	45 (3.68%)
Stage 20	1601	93 (5.84%)	62 (5.84%)	199 (4.77%)	76 (5.16%)
Stage 25	2423	154 (7.34%)	115 (7.34%)	296 (5.28%)	121 (5.21%)
Stage 30	3423	240 (8.62%)	193 (8.62%)	435 (6.70%)	189 (6.38%)

the maximum distance method. Using $D = 300$ features, less than 2% average predictive variance increases at stage 30 are observed due to approximation, as shown in Table 3.6. On the other hand, due to the moderately expensive computation in Algorithm 2 using $D = 300$ features, feature approximation search without LSH is more time-consuming than the maximum distance method. As shown in Table 3.1, the computation of more design points incurs higher computational costs in order of D^2 for feature approximation search without LSH (complexity $O(j^2 + D^2N)$). With an LSH approximate similarity-search method, computation time is reduced by 189 seconds ($\approx$ 3 minutes) across all 30 stages. While the feature approximation approach outperforms exhaustive search, it appears to be most useful when the maximum distance approach is very conservative, such as in the two-dimensional case in Section 3.4.1.

3.5 Conclusion and Discussion

The two methods considered here can be extended for selecting more than one point in each stage j in a straight-forward manner. For example, suppose two points are to be selected in each stage. Let $j' = 2j$, $X_{j'}$ be the current sub-design at stage j, and $x_{j'+1}$ and $x_{j'+2}$ be two points selected at the stage $j + 1$. Proposition 3.3.1 can be extended to $V_{j+1}(x) = V_j(x) - \sigma^2 R^*(x_{j'+1}, x_{j'+2})$ for a function R^*, Theorem 3.3.2 can narrow

Table 3.6: The relative difference in average predictive variance of the emulator between maximum distance search as a baseline and feature approximation search with number of features $D = 300$ over 20 Sobol predictive locations in 6-dimensional setting.

Relative Difference	Feature Approximation Method with $D = 300$		Average Variance by Maximum Distance Method
	w/o LSH	w/ LSH	
Stage 10	0.049	0.047	0.2328
Stage 15	0.030	0.032	0.2120
Stage 20	0.023	0.022	0.1997
Stage 25	0.017	0.017	0.1913
Stage 30	0.016	0.016	0.1850

the window of potential *pairs* of candidate locations, to say $T'(X_j)$, and Algorithm 1 can be updated accordingly. On the other hand, retaining good computational properties in a batch-sequential framework is not straight-forward. For example, searching for the optimal candidates, $(x_{j'+1}, x_{j'+2}) = \arg\max_{(u_1,u_2)\in T'(X_j)} R^*(u_1, u_2)$, can be very expensive, say $O(|T'(X_j)|^2)$, compared to searching for one point in each stage. Efficiently augmenting multiple points at each stage, for example by alternating maximizations on $x_{j'+1}$ and $x_{j'+2}$, might be worth exploring in future work.

The essential ideas of the proposed approaches have potential for application in search space reduction in global optimization. Consider the following example. [105] modified the *maximum entropy design* [106] for use as a sequential algorithm to efficiently construct a space-filling design in computer experiments. They showed that the algorithm can be simplified to selecting a new point that maximizes the so-called *sequential maximum entropy criterion*

$$x_{j+1} = \arg \min_{u\in\mathcal{D}\setminus X_j} \Phi_\Theta(u, X_j)\Phi_\Theta(X_j, X_j)^{-1}\Phi_\Theta(X_j, u),$$

where $\mathcal{D}$ is a *discrete* design space. For this global optimization problem, let

$$d_{\max}(x_{j+1}) = \max\{\|\Theta(x_1 - x_{j+1})\|_2, \|\Theta(x_2 - x_{j+1})\|_2, \ldots, \|\Theta(x_j - x_{j+1})\|_2\}$$

and $\delta > 0$. It can be shown that if $d_{\max}(x_{j+1}) \leq \phi^{-1}\left(\sqrt{\lambda_{\max}\delta}\right)$, where $\lambda_{\max}$ is the maximum eigenvalue of $\Phi_\Theta(X_j, X_j)$, then the objective function $\Phi_\Theta(x_{j+1}, X_j)\Phi_\Theta(X_j, X_j)^{-1}\Phi_\Theta(X_j, x_{j+1}) > \delta$. Thus, similar to Algorithm 1, a maximum distance approach could be used to eliminate search candidates. For other specific global optimization problems, detailed examination is needed.

An implicit disadvantage of these methods is the impact of the correlation parameters. Take the example in Figure 3.2, where $d_{\min}(x_9) < 3.07$. From the definition (3.6) of $d_{\min}(x_{j+1})$, if $\Theta = (1/\sqrt{\theta}, 1/\sqrt{\theta})$, then the larger θ is, the bigger the search area, the shaded region in Figure 3.2. When θ is large, the correlation is close to one and the data points tend to be highly correlated, implying that every data point in the full design carries important information for each predictive location. Thus, the algorithm requires more computation for "easier" problems—i.e., with a "flatter" surfaces. On the other hand "flatter" surfaces do not require large sub-designs to achieve small predictive variance.

An improvement worth exploring is how to determine of the number of features D in the feature approximation method. Cross-validation to minimize predictive variance of an emulator may present an attractive option. An examination of the choice between the maximum distance and feature approximation methods might be desirable. Although using them both in concert guarantees a smaller candidate set in the feature approximation method, pre-computation of the features constitutes a moderately expensive sunk cost in terms of computation and storage. In the example in Section 3.4.2, a $500 \times 50,000$ matrix needed to be computed and stored in advance. In either case, the two methods outperform exhaustive search, as shown in Table 3.5.

CHAPTER 4

MULTI-RESOLUTION FUNCTIONAL ANOVA FOR LARGE-SCALE, MANY-INPUT COMPUTER EXPERIMENTS

4.1 Introduction

Computer models are implementations of complex mathematical models using computer codes. They are used to study systems of interest for which physical experimentation is either infeasible or very limited. For example, [107] model crystalline micro-structure of alloys as a function of solidification velocity. Another example is the simulation of population-wide cardiovascular effects based on salt intake in the U.S. presented in [108].

Calibration, exploration, and optimization of a computer model requires the response given many potential inputs. Computer models are often too computationally demanding for free generation of input/response combinations. A well-established solution to this problem is the use of *emulators* [6, 5]. This solution involves evaluting the response at a series of well-distributed inputs. Then, an emulator of the computer model is built using the collected data. Calibration, exploration, or optimization can then be carried out on the emulator directly [109, 5, 110, 111, 112, 84].

A standard method for building emulators after computer experiments is Gaussian process [5], or almost equivalently [113] reproducing kernel Hilbert space regression [114]. Gaussian process modeling leverages known properties of the underlying response surface to produce both predictions and uncertainty quantification after an experiment. Gaussian process emulation is mathematically simple and enables statistical uncertainty quantification via confidence intervals.

Unfortunately, the use of Gaussian process emulators is limited for large-scale computer experiments. Let $X = \{x_1, \ldots, x_n\}$ denote the set of input locations for the experiment,

$f(x)$ the computer model response at input x, and $\Phi(x, x')$ the kernel function at inputs x and x'. Further, let $\Phi(X, X)$ denote the $n \times n$ matrix with entries $\Phi(x_i, x_j)$ and $f(X)$ the length n vector of responses $f(x_i)$. The simplest form of Gaussian process emulator is then found by solving for the n vector α with $\Phi(X, X)\alpha = f(X)$. There are at least three major challenges that prevent using the Gaussian process emulator as n gets large, ranked roughly in order of consequence for typical combinations of sample size, kernel, and experimental design. *(i)* More than $n^2/2$ values are needed to represent $\Phi(X, X)$, which can cause memory challenges, particularly on a personal computer. *(ii)* Numeric solutions to $\Phi(X, X)\alpha = f(X)$ can be highly unstable, so that more data can lead to less accurate results. *(iii)* The computational complexity for solving the linear system $\Phi(X, X)\alpha = f(X)$ can be burdensome for large n.

Overcoming these problems, which are also key bottlenecks for many related statistical methods, is an active area of research. While much progress has been made in this area, much work remains. There have been partial solutions proposed in the literature: using less smooth kernels can address *(ii)* [71], covariance tapering *(i,ii)* [79, 115], a nugget effect *(ii)* [116], multi-step emulators *(i,ii)* [80], specialized design *(i,iii)* [83], and parallelization and computational methods *(ii)* [82]. To address all three challenges simultaneously, one must exploit features present in the response surface. Local approaches to emulation address *(i)*,*(ii)*, and *(iii)* using the principal that only a fraction of the total responses from an experiment are needed to achieve accurate prediction at a particular input of interest [52, 90, 2, 89].

This article discusses a new multi-resolution functional ANOVA (MRFA) approach to emulation of large-scale (large n) and many-input (many-dimensional x) computer experiments. The MRFA operates by exploiting features which are commonly encountered in practical computer models. The remainder of this article is organized as follows. In Section 4.2, we provide background and preliminary results, then introduce the MRFA model. In Section 4.3, we formulate the model fitting as an overlapping group lasso problem

and discuss efficient model fitting, as well as tuning parameter selection. In Section 4.4, we present new results on consistency and large-sample hypothesis testing for the high-dimensional, potentially overlapping, group lasso problem. The test is then inverted to obtain pointwise confidence intervals on the regression function. Basis function selection is discussed in Section 4.5. In Section 4.6, we present a few illustrative examples showcasing the capabilities of the MRFA technique in a large-scale, many-input setting. Finally, in Section 4.7, we close with a brief discussion. Proofs are provided in the Appendix D.

4.2 Multi-Resolution Functional ANOVA

The motivation for the multi-resolution functional ANOVA emulator is as follows. First, note that a function with a low-dimensional input can easily be approximated given a large number of responses provided sufficient smoothness. One does not have to use anything as complex as even the simplest Gaussian process regression to achieve good emulation, and in many cases Gaussian process regression would fail for the reasons discussed in the introduction. For example, if one has $n = 100,000$, then a Gaussian process emulator has $100,000$ basis functions, which is far more than necessary for arbitrarily high-accuracy approximation of most low-dimensional functions. Consider the example shown in Figure 4.1. In the example, 1000 evenly spaced data points are collected. Using Wendland's kernel [117] with $k = 4$ and width 0.75 implies $\Phi(X, X)$ has condition number 4.6×10^{22}, so that the inverse is not useful in a floating point setting. On the other hand, the true function is reasonably well-approximated by the set of five basis functions shown in gray in the left panel and very well-approximated by the set of 15 basis functions shown in gray in the right panel. This type of multi-resolution emulation [81] has been successfully employed for function approximation, particularly in a low-dimensional input setting.

Approximating easily in low-dimensions does not directly improve approximations in higher-dimensions, where coming up with a good set of basis functions is an onerous task. Roughly, if an unknown function has a high-dimensional input and no simplifying structure,

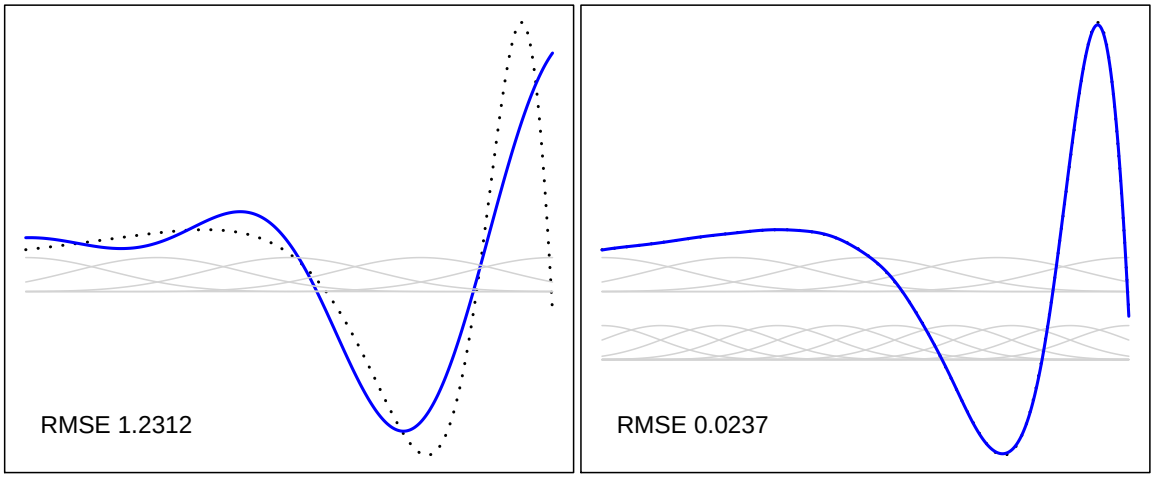


Figure 4.1: Multi-resolution example with 5 basis function (left panel) and 15 basis functions (right panel). Here, the true function is shown in dotted black, the emulator in solid blue, and the basis functions are Wendland's kernels with $k = 4$ and widths 0.75 and 0.50, shown in solid light gray.

then the exercise of trying to build an accurate emulator with finite data is essentially hopeless, so a means for detecting simplifying structure should be a corner-stone of any proposed technique.

Consider a relatively low-order functional ANOVA, where a function is represented as a sum of main effect functions, two-way interaction functions and so on. Functional ANOVA has played an important role in variable screening for many-input computer experiments. See for example Chap. 6.3 of [84] or Chap. 7.1 of [5]. Functional ANOVA has also been used for function approximation across a spectrum of other applications. For example, [118] used a functional ANOVA representation to approximate the variance of scrambled net quadrature and [119] approximated a general regression function using a functional ANOVA structure. By considering a function with a low-order functional ANOVA, the curse of dimensionality can be largely sidestepped. While this modeling approach can increase the flexibility of additive modeling, it retains much of the interpretability.

Our proposed multi-resolution, functional ANOVA approach respects two types of effect heredity [120], *(i)* in the order of functional ANOVA, so that higher-order interaction functions are only entertained if all their lower-dimensional components are present, and *(ii)*

in the resolution of approximation to these relatively low-dimensional component functions, so that not too many basis functions are used. The hope is that by targeting a simpler representation (low-order functional ANOVA model), which is amenable to low-dimensional approximation (via multi-resolution model), accurate emulators can be formed in a very large-scale and many-input setting.

For an integrable function $f : \Omega \to \mathbb{R},\ \Omega \subset \mathbb{R}^d$, a functional ANOVA can be defined recursively as follows. Let $f_\emptyset = \int_\Omega f(x)\mathrm{d}x$ and

$$f_u(x) = \int_{\Omega_{-u}} \left(f(x) - \sum_{v \subsetneq u} f_v(x) \right) \mathrm{d}x_{-u}. \tag{4.1}$$

Here, $u, v \subset \mathcal{D} = \{1, \ldots, d\}$ denote sets of indices and the notation $\int_{\Omega_{-u}} \cdots \mathrm{d}x_{-u}$ indicates integration over the variables not in u for a fixed value of x_u. Now, f can be represented via its ANOVA decomposition as

$$f(x) = \sum_{u \subseteq \mathcal{D}} f_u(x).$$

Note that in this decomposition, each component function $f_u(x)$ is a function of x that only depends on x_u. $f_\emptyset$ is often referred to as the *mean* function, $f_{\{i\}}(x)$, $i \in \mathcal{D}$ as the *main effect* functions, $f_{\{i,j\}}(x)$, $i, j \in \mathcal{D}, i \neq j$ as the *two-way interaction* functions, and so on. The terms in the functional ANOVA (4.1) are orthogonal in $L_2(\Omega)$, which ensures uniqueness of the representation. Generally, there is no closed form for the component functions f_u, so Monte Carlo techniques are commonly used to approximate them.

It turns out that if the full-dimensional function f lives in an reproducing kernel Hilbert space (RKHS) on $[0, 1]^d$ with a product kernel, then f can be represented as a sum of component functions f_u, which live in RKHS's whose kernels (and therefore norms) are determined by the full-dimensional kernel. This result is summarized in Theorem 4.2.1, whose proof is given in Appendix D.1. Define an RKHS $\mathcal{N}_\Phi(\Omega)$ for a symmetric positive-

definite kernel $\Phi : \Omega \times \Omega \to \mathbb{R}$ as the *closure* of the normed linear space,

$$\left\{ \sum_{x \in X} \beta_x \Phi(\cdot, x) \middle| \, \beta_x \in \mathbb{R}, \; x \in \Omega \right\},$$

with inner product $\sum_{x \in X} \sum_{y \in Y} \alpha_x \beta_y \Phi(x, y)$ for component functions $\sum_{x \in X} \alpha_x \Phi(\cdot, x)$ and $\sum_{y \in Y} \beta_y \Phi(\cdot, y)$.

Theorem 4.2.1. *Suppose $\Phi \in \Omega \times \Omega \to \mathbb{R}$ is a symmetric positive-definite kernel on $\Omega = [0, 1]^d$ and Φ is a product kernel, $\Phi(x, y) = \prod_{j=1}^{d} \phi_j(x_j, y_j)$. Then, any $f \in \mathcal{N}_\Phi([0, 1]^d)$ has representation $f = \sum_{u \subseteq \mathcal{D}} f_u$, where $f_u \in \mathcal{N}_{\Phi_u}([0, 1]^{|u|})$ and $\Phi_u = \prod_{j \in u} \phi_j$, where $|A|$ denotes the cardinality of a set A.*

The proposed emulator is a low-resolution representation of a low-order functional ANOVA, $\hat{f}_{\text{ANOVA}}$. Clearly, this process introduces approximation errors due to both the resolution and the order of the ANOVA. On the other hand, it is anticipated that for target functions encountered in practice, inaccuracy due to the low-order functional ANOVA and low-resolution approximation will be small. In other words, high-order interaction functions will be negligible and low-dimensional component functions will be well-approximated by a relatively small set of basis functions.

An MRFA emulator can be represented as

$$\hat{f}_{\text{MRFA}}(x) = \sum_{u \in \mathcal{E}} \sum_{r \leq R(u)} \hat{f}_{u,r}(x),$$

where $\mathcal{E}$ is a set of sets of indices which obeys strong effect heredity (if a set of indices is in $\mathcal{E}$, then every one of its subsets is also in $\mathcal{E}$) and $R(u) \in \mathbb{N}$ denotes the resolution level used to represent component function f_u. Note that this model has no restrictions on orthogonality of components functions. If each $\hat{f}_{u,r}$ is represented as a linear combination

of $n_u(r)$ basis functions $\varphi_u^{rk} : \mathbb{R}^{|u|} \to \mathbb{R}$, $k = 1, \ldots, n_u(r)$, then

$$\hat{f}_{\text{MRFA}}(x) = \sum_{u \in \mathcal{E}} \sum_{r \leq R(u)} \sum_{k=1}^{n_u(r)} \hat{\beta}_u^{rk} \varphi_u^{rk}(x_u).$$

For simplicity, level of resolution is taken in pre-specified increments indexed by positive integers.

From a statistical learning perspective, the order of functional ANOVA and resolution of representation can likely be gleaned from the collected data. This idea is adopted in the next section to enable the construction of MRFA emulators.

4.3 Estimation and Regularization

A straight-forward approach to finding a set of sets of indices $\mathcal{E}$ which obeys strong effect heredity, in both functional ANOVA and resolution, and allows construction of an accurate model is stepwise variable selection. Initial investigations along these lines indicate that stepwise variable selection is capable of producing a high-accuracy model, but introduces a very serious computational bottleneck to model fitting, particularly for large-scale and many-input problems. Alternatively, posing the problem as a penalized regression can provide huge computational savings.

[121] proposed the group lasso penalty which can be used to perform variable selection with grouped variables, for example a set of basis function evaluations. In the group lasso framework, the overall penalty term is the sum of unsquared L_2 norms of the coefficients of variables within groups. This type of penalty ensures that all the components of the groups have zero or non-zero coefficients simultaneously. [122] noticed that the group lasso penalty could be used to enforce a spectrum of effect hierarchies by employing an *overlapping group structure*. In particular, if a group of variables' *parents* (those variables which must be present if the group is present) are always included in the unsquared L_2 penalty component with the group of interest, then the group of variables can only have non-zero coefficients if

the parents have non-zero coefficients. One can consider the penalized loss function

$$Q = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \sum_{|u|=1}^{D_{\max}}\sum_{r=1}^{R_{\max}}\sum_{k=1}^{n_u(r)}\beta_u^{rk}\varphi_u^{rk}(x_{iu})\right)^2 + \lambda\sum_{|u|=1}^{D_{\max}}\sum_{r=1}^{R_{\max}}\sqrt{N_u(r)\sum_{v\subseteq u}\sum_{s\leq r}\sum_{k=1}^{n_v(s)}(\beta_v^{sk})^2}, \tag{4.2}$$

where $D_{\max}$ and $R_{\max}$ respectively denote maximal orders of functional ANOVA and resolution level, and $N_u(r) = \sum_{v\subseteq u}\sum_{s\leq r} n_v(s)$. Notably, $D_{\max} \ll d$ and $R_{\max} \ll n$ to ensure computational feasibility in a large-scale, many-input setting. Efficient, large-scale algorithms are available for coefficient estimation in the group lasso setting [123, 124]. In particular, the algorithm described in [123] is implemented in the `R` [19] package `grplasso` [125].

Although the algorithm in [123] is quite computationally efficient, storage requirements still have potential to cause computational infeasibility, particularly for a large-scale and many-input problem. We propose a modification of the algorithm where *candidate* basis function evaluations are added sequentially along the lasso path, as necessary to ensure effects heredity, rather than storing all the basis functions in advance. The modified algorithm is given in Appendix D.2. The algorithm starts from a candidate set consisting only of main effect functions with resolution level one and an initial penalty $\lambda_{\max}$ set as suggested in [123]. Then, the penalty parameter is gradually decreased and the model re-fit over steps. If the active set changes in a particular step, the candidate set is enlarged to include *child* basis function evaluations as required by effects heredity in functional ANOVA and resolution. A small value of the penalty parameter increment Δ is required to ensure that at most one new active group is included in each update. The algorithm stops when some convergence criterion is met, or alternatively memory limits are approached.

The accuracy of the emulator can depend strongly on the tuning parameter λ. When overfitting is not a major concern, for example when constructing an emulator or near

interpolator for a deterministic computer experiment, the smallest λ (corresponding to the most complex model) with no evidence of numeric instability could be taken, which in turn would give near interpolation of outputs at input locations in the data used for fitting. On the other hand, if overfitting is a concern, a few sensible choices for tuning parameter selection include cross-validation or classical information criteria such as Akaike information criterion (AIC) and Bayesian information criterion (BIC). Under some conditions, BIC is consistent for the true model when the set of candidate models contains the true model, while AIC will select a sequence of models which are asymptotically equivalent to the model whose average squared error is smallest among the candidate models. Generalized cross-validation (GCV) [126], leave-one-out cross-validation and AIC have similar asymptotic behavior. Delete-d cross-validation [127] is asymptotically equivalent to the generalized information criterion (GIC) with parameter $\lambda_n = n/(n-d)+1$. See [128], [129] and [127] for more details. The use of AIC and BIC for regularization parameter selection in penalized regression models has been discussed in recent literature (see [130] and [131]). [130] showed that BIC can consistently identify the true model for the smoothly clipped absolute deviation penalty [132], whereas the models selected by AIC and GCV tend to overfit. For the group lasso framework, our numerical results indicate BIC has slightly better performance than AIC. On the other hand, if parallel computing environments are available, cross-validation can be computationally efficient and can be used for selecting the tuning parameter λ.

In addition to prediction, uncertainty quantification is essential in practice. In Section 4.4, we develop some new theoretical results for estimation, and in turn prediction, inference. Further, an algorithm for constructing pointwise confidence intervals as a means to quantify one's uncertainty in the predicted values is provided in Appendix D.3.

4.4 Statistical Properties of the MRFA Emulator

In this section, we develop new consistency and inference results for the, possibly overlapping, group lasso problem in a large n, large p setting, and apply these results to the

MRFA emulator for the construction of confidence intervals. Notably, these results are very general and relate to the MRFA emulator only in the sense that the MRFA model forms an application case of particular interest. The results are developed along the lines described in [133], [134], and [135]. Suppose for a particular input location x, the true value is $y^*(x)$. Pointwise convergence is ensured via Theorem 4.4.1 which establishes consistency of the coefficient estimates under some assumptions. A pointwise confidence interval is constructed by inverting a one-dimensional hypothesis test of $H_0 : y^*(x) = \delta$, as provided in Theorem 4.4.3, after the model has been reparametrized so that $y^*(x)$ equals a particular coefficient in the model. The one-dimensional hypothesis test uses a decorrelated score function, that converges weakly to standard normal, following [135]. Details are provided below and in Appendix D.5.

For simplicity, the linear model is denoted by

$$y_i = \beta^T \varphi_i + \epsilon_i, \tag{4.3}$$

with $\mathbb{E}(\epsilon_i) = 0$ and $\mathbb{V}(\epsilon_i) = \sigma^2$ for $i = 1, ..., n$. In the context of the MRFA model, φ_i and $\beta \in \mathbb{R}^p$ respectively denote unique basis function evaluations (i.e. not duplicate basis function evaluations appearing in overlapping groups) at x_i and basis function coefficients, $\varphi_u^{rk}(x_i)$ and β_u^{rk}, $|u| \leq D_{\max}$, $r \leq R_{\max}$, and $k = 1, \ldots, n_u(r)$, and high-frequency "left-overs" ϵ_i are approximated by independent and identically distributed (i.i.d.) noise. Inference is considered in the $n \to \infty, p \to \infty, p \gg n$ setting for n i.i.d. pairs (φ_i, y_i). The following definitions are used. For two positive sequences a_n and b_n, we write $a_n \asymp b_n$ if for some $C, C' > 0$, $C \leqslant a_n/b_n \leqslant C'$. Similarly, we use $a_n \lesssim b_n$ to denote $a_n \leqslant Cb_n$ for some constant $C > 0$. The following l_2 consistency result can be established.

Theorem 4.4.1. *Suppose the estimated coefficients of the overlapping group lasso are $\hat{\beta}_{\lambda_n}$ (see (D.3)), and the true coefficients are β^*. Under assumptions on the m-sparse eigenvalues (see Definition D.5.2 and Assumption D.5.1) of $\mathbb{E}\varphi_i\varphi_i^T$, $\lambda_n \asymp \sqrt{\frac{\log p}{n}}$, and $\bar{d}^2 = o(\log n)$,*

with probability tending to 1 for $n \to \infty$,

$$\|\hat{\beta}_{\lambda_n} - \beta^*\|_2^2 \lesssim \frac{\bar{c}^2 s \bar{d} \log p}{n},$$

where $\bar{c}$, $\bar{d}$, and s denote the largest number of groups that a variable appears in, the size of the largest group, and the number of non-zero elements in unique representation β^*, respectively.

Remark 4.4.2. Potential dependency of $\bar{c}$, $\bar{d}$, and s on n is suppressed for notational simplicity.

In brief, the assumptions of Theorem 4.4.1 require a bounded largest eigenvalue of $\frac{1}{n}\varphi^T\varphi$, where $\varphi = (\varphi_1, ..., \varphi_n)^T$, and eigenvalues corresponding to its m-sparse eigenvectors bounded away from zero. For a detailed discussion, see [133]. To ensure the l_2 consistency in Theorem 4.4.1, we need that the numerator of the right hand side not grow too fast, $o(n)$. This in turn requires the size of groups, number of nonzero (true) coefficients, and number of groups that a variable appears in are relatively small compared with the sample size n. The dimension of MRFA representation can grow, but not so quickly that $\log p$ is large compared to n.

Now, we develop a theorem providing the large sample distribution of a decorrelated score statistic. Following [135], rewrite the linear model in terms of a parameter of interest $\theta \equiv \beta_j \in \mathbb{R}$ and a nuisance parameter $\gamma \equiv (\beta_1, \ldots, \beta_{j-1}, \beta_{j+1}, \ldots, \beta_p)^T \in \mathbb{R}^{p-1}$, $y_i = \theta Z_i + \gamma^T Q_i + \epsilon_i$, where $Z_i = \varphi_{ij}$ and $Q_i = (\varphi_{i1}, \ldots, \varphi_{i,j-1}, \varphi_{i,j+1}, \ldots, \varphi_{ip})^T$. Define a *decorrelated* score function

$$S(\theta, \gamma) = -\frac{1}{n\sigma^2} \sum_{i=1}^{n} (y_i - \theta Z_i - \gamma^T Q_i)(Z_i - w^T Q_i),$$

where $w = \mathbb{E}(Q_i Q_i^T)^{-1} \mathbb{E}(Q_i Z_i)$. The score function for the target parameter has been decorrelated with the nuisance parameter score function. Here, the full parameter vector

β, consisting of target and nuisance parameters θ and γ, can be estimated via the original overlapping group lasso problem, so that $\hat{\beta} = (\hat{\theta}, \hat{\gamma}^T)^T$. On the other hand, w can be estimated via

$$\hat{w} = \arg\min \|w\|_1, \text{ s.t. } \left\| \frac{1}{n} \sum_{i=1}^{n} Q_i (Z_i - w^T Q_i) \right\|_2 \leqslant \lambda' \tag{4.4}$$

and the error variance σ^2 can be estimated by a consistent estimator $\hat{\sigma}^2$. Note that λ' is another tuning parameter. The minimization is on the l_1 norm of w, since we want to ensure sparsity of $\hat{w}$. Let θ^* and γ^* denote the true values of θ and γ. The following (one-dimensional) inference result can be obtained. A proof is provided in Appendix D.6.

Theorem 4.4.3. *Under $H_0 : \theta^* = \theta_0, \lambda' \asymp \sqrt{\frac{\log p}{n}}$, and the assumptions of Theorem D.6.3,*

$$\sqrt{n} \hat{S}_{\hat{\sigma}^2}(\theta_0, \hat{\gamma}) \hat{I}_{\theta|\gamma}^{-1/2} \xrightarrow{\text{dist.}} \mathcal{N}(0, 1),$$

where $\hat{I}_{\theta|\gamma} = \frac{1}{n\hat{\sigma}^2} \sum_{i=1}^{n} Z_i (Z_i - \hat{w}^T Q_i)$, and $\hat{S}_{\hat{\sigma}^2}(\theta, \gamma) = -\frac{1}{n\hat{\sigma}^2} \sum_{i=1}^{n} (y_i - \theta Z_i - \gamma^T Q_i)(Z_i - \hat{w}^T Q_i)$.

The solution to optimization problem (4.4) can also be represented as

$$\hat{w} = \arg\min \left\| \frac{1}{n} \sum_{i=1}^{n} Q_i (Z_i - w^T Q_i) \right\|_2 + \lambda'' \|w\|_1, \tag{4.5}$$

where λ'' is a transformed tuning parameter. Notice that this is a lasso problem where the j-th response, $j = 1, \ldots, p-1$, is $(\frac{1}{n} \sum_{i=1}^{n} Q_i Z_i)_j$ and the covariate matrix is $\frac{1}{n} \sum_{i=1}^{n} Q_i Q_i^T$. The tuning parameter λ'' can be selected via cross-validation, aiming for a minimal sum of squared errors, or simply fixed. Theorem 4.4.3 requires all the assumptions of Theorem 4.4.1. In addition, it is required that the smallest eigenvalue of $\mathbb{E}(Q_i Q_i^T)$ is bounded away from zero, the number of nonzero elements in $w = \mathbb{E}(Q_i Q_i^T)^{-1} \mathbb{E}(Q_i Z_i)$ is small compared to n, and the tail probabilities of residuals and basis function evaluations are small in the sense that they are sub-Gaussian. For details, see Appendix D.6.

Let φ^* denote the basis function evaluations at a particular predictive location x^*, and y^* denote the predictive output, $y^* = \beta^T \varphi^*$. By extending φ^* to a basis of $\mathbb{R}^p$, $B = (\varphi^*, c_2, \ldots, c_p)$, the linear model (4.3) can be written as $y_i = \eta_1 \tilde{Z}_i + \eta_{(-1)}^T \tilde{Q}_i + \epsilon_i$, where $(\tilde{Z}_i, \tilde{Q}_i)^T = B^{-1}\varphi_i$ and $(\eta_1, \eta_{(-1)}^T)^T = B\beta$. Thus, the hypothesis test $H_0 : y^* = \eta_{10}$ is equivalent to $H_0 : \eta_1 = \eta_{10}$, and a $(1-\alpha) \times 100\%$ confidence interval on y^* can be constructed by inverting the hypothesis test, as stated in the following corollary. An algorithm for confidence interval construction is provided in Appendix D.3. In the algorithm, a simple construction for the matrix B is to take c_i as a unit vector with i-th element equaling one. Then, the inverse of B can be computed efficiently via partitioned matrix inverse results [86].

Corollary 4.4.4. *Under the assumptions of Theorem 4.4.3, a* $(1-\alpha) \times 100\%$ *confidence interval on* y^* *can be constructed as*

$$\left\{ y^* | \Phi^{-1}\left(\frac{\alpha}{2}\right) \leqslant \sqrt{n} \hat{S}_{\hat{\sigma}^2}(y^*, \hat{\eta}_{(-1)}) \hat{I}_{y^*|\eta_{(-1)}}^{-1/2} \leqslant \Phi^{-1}\left(1 - \frac{\alpha}{2}\right) \right\},$$

where $\hat{I}_{y^*|\eta_{(-1)}} = \frac{1}{n\hat{\sigma}^2} \sum_{i=1}^n \tilde{Z}_i (\tilde{Z}_i - \hat{w}^T \tilde{Q}_i)$, $\hat{S}_{\hat{\sigma}^2}(y^*, \eta_{(-1)}) = \frac{1}{n\hat{\sigma}^2} \sum_{i=1}^n (y_i - y^* \tilde{Z}_i - \eta_{(-1)}^T \tilde{Q}_i)(\tilde{Z}_i - \hat{w}^T \tilde{Q}_i)$, Φ *is the cumulative distribution function of the standard normal distribution, and* $\hat{\eta}_{(-1)}$ *is an estimator of* $\eta_{(-1)}$*, which can be obtained by plugging in the estimator of* β*.*

An illustration of these pointwise confidence intervals is shown in Figure 4.2. For the example, the true (deterministic) function is $f(x) = \exp(-1.4x)\cos(3.5\pi x)$, shown as a black dotted line, and we attempt to build an interpolator using 15 evenly spaced data points. The basis functions are Wendland's kernels with $k = 2$ and width 0.75. The tuning parameter λ'' is chosen via cross-validation at each untried input site of interest. The 95% confidence intervals are shown for (penalized regression) tuning parameters $\lambda = 0.25$ and 1.00 in the left and right panel, respectively. Given a set of testing samples of size 401, 97.8% and 99.0% of true values are contained by the respective confidence intervals for

$\lambda = 0.25$ and 1.00. Notice that for a larger sparsity parameter, corresponding to a simpler model, uncertainty is greater, reflecting more model bias. Recall that overfitting is less of a concern in emulation of deterministic computer experiments.

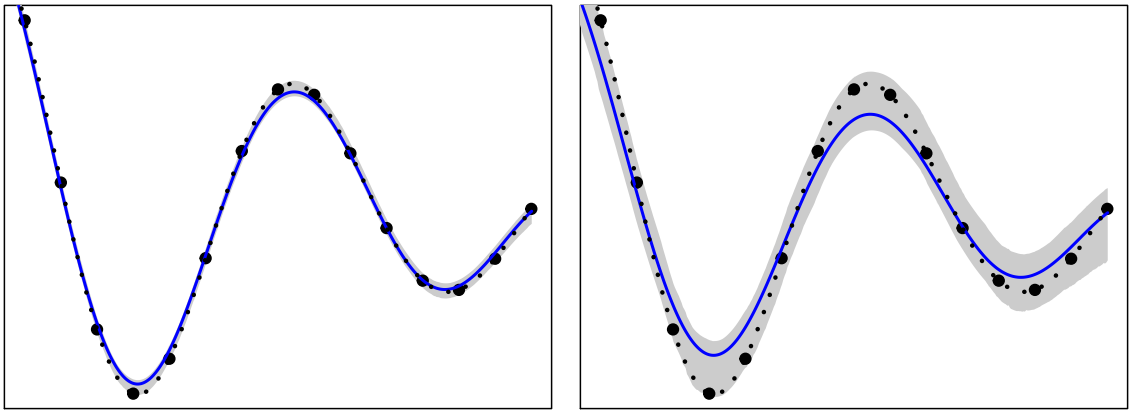


Figure 4.2: Illustration of confidence intervals for $\lambda = 0.25,\ 1.00$. Black dotted line represents the true deterministic function, black dots represent the collected data, and the MRFA model is represented as the blue line, with the gray shaded region providing a pointwise 95% confidence band.

Optimization problems (4.4) and equivalently (4.5) can be very computationally challenging when N is large. In particular, for $R_{\max} = 10$ and $D_{\max} = 10$ (as used in the examples later), p is nearly 10^7, making storage of the $Q_i : (p-1) \times 1$, $i = 1, \ldots, n$ infeasible without specialized computational resources. In Appendix D.4, we provide a large N modification to the confidence interval algorithm in Appendix D.3. In the modification, only those nuisance basis function evaluations which have been included for consideration up to the selected stage of the group lasso problem are considered in Q_i, reducing the size of Q_i by several orders of magnitude. Given the reduced Q_i, we propose to estimate w via a ridge regression, since sparsity of w relative to the sample size N is ensured by default. While the intervals are computationally feasible in a large scale, many-input setting, their coverage is somewhat liberal, and we apply a post-hoc correction, as proposed by [136]. The idea is to regard σ^2 as a tuning parameter and then apply a cross-validation method to the confidence interval constructed by Corollary 4.4.4 to find the σ^2 which most closely achieves the nominal coverage $(1-\alpha) \times 100\%$.

4.5 Basis function selection

Basis functions of a given input dimension should be selected so that they are capable of approximating a broad spectrum of practically encountered target functions, with flexibility increasing as the level of resolution increases.

For a particular dimensionality of component function $m = |u|$, a reasonable building block for a set of basis functions is a positive definite function. The function $\phi : \mathbb{R}^m \to \mathbb{R}$ is positive definite if $\sum_{i,j} \alpha_i \alpha_j \phi(x_i - x_j) \geq 0$ for any $\alpha_i \in \mathbb{R}$, $x_i \in \mathbb{R}^m$ and strictly positive for distinct x_i if at least one α_i is non-zero. These could be constructed by integrating the full-dimensional kernel over margins as indicated in Theorem 4.2.1. More simply, the kernels could be selected to ensure a desired smoothness of the target component functions. Common example kernels include the Matérn or squared exponential correlation functions, and Wendland's kernels [117], which are used in the examples presented here. The *center* and *scale* of these basis functions, or *kernels*, can be adjusted via c and h, respectively, in the representation $\phi((x - c)/h)$. For a particular resolution level, a straightforward choice is to take as basis functions a set of kernels with centers well-spread through the input space. The scale should be chosen large enough to ensure the desired smoothness of the target function, but not so large that numeric issues arise in parameter estimation. The number of centers, and in turn coefficients, concretely describes the complexity of the resolution level. Take as an example the 5 basis functions shown in light gray in the left panel of Figure 4.1. With centers $0, 0.25, \ldots, 1$ and width 0.75, these 5 basis functions are capable of approximating a broad range of relatively smooth and slowly varying target functions. For the next resolution level, the same basic kernel can be used again, but with a denser set of centers and correspondingly smaller scale. Take once again the example basis functions shown in Figure 4.1. The 10 second-level resolution basis functions with centers $0, 0.11, \ldots, 1$ and width 0.5 augment the first-level resolution basis functions to allow approximation of an even broader range of target functions. Note that for a fixed

dimensionality m and resolution level r the span of these basis functions forms a linear subspace of the RKHS associated with kernel $\phi((\cdot - \cdot)/h_r)$, where h_r denotes the bandwidth for the highest (or finest) resolution level r. Another reasonable choice for basis functions could be polynomials of increasing degree.

4.6 Examples

Several examples are examined in this section, a ten-dimensional, large-scale example which demonstrates the algorithm and statistical inference, a larger-scale and many-input example with a relatively complicated underlying function, and a stochastic function example. A few popular exemplar functions are examined additionally. These examples show that the multi-resolution functional ANOVA typically substantially outperforms traditional Gaussian process methods in terms of computational time, emulator accuracy, model interpretability, and scalability. In addition, we also compare with the local Gaussian process method, which is a scalable method proposed by [2]. All the numerical results were obtained using `R` [19] on a server with 2.3 GHz CPU and 256GB of RAM. The traditional Gaussian process, local Gaussian process and MRFA approaches were compared and respectively implemented in `R` packages `mlegp` [137], `laGP` [87] and `MRFA` [138]. The default settings of the packages `mlegp` and `MRFA` were selected. For the package `laGP`, initial values and maximum values for correlation parameters were given as suggested in [87]. For `laGP` and `MRFA`, 10 CPUs were requested via `foreach` [139] for parallel computing.

In the implementation of the MRFA model, Wendland's kernels with $k = 2$ are chosen, and at most 10-way interaction effects and 10 resolution levels ($R_{\max} = 10$ and $D_{\max} = 10$) are considered. For the tuning parameter setting λ, in Sections 4.6.1, 4.6.2 and 4.6.4 where the target functions are deterministic, the smallest λ, corresponding to the most complex model, without exceeding memory allocation is taken. In Section 4.6.3 where a stochastic target is considered, AIC, BIC and CV criteria were considered for choosing the tuning parameter and the comparison is explicitly discussed.

Table 4.1: Selected effects and resolution by model complexity.

λ	Selected inputs
1904.819	$\hat{f}_{\{3\},1}$
1885.866	$\hat{f}_{\{3\},1}, \hat{f}_{\{2\},1}$
551.225	$\hat{f}_{\{3\},1}, \hat{f}_{\{2\},1}, \hat{f}_{\{1\},1}$
87.544	$\hat{f}_{\{3\},1}, \hat{f}_{\{2\},1}, \hat{f}_{\{1\},1}, \hat{f}_{\{2,3\},1}$
$\vdots$	$\vdots$
0.003	$\hat{f}_{\{3\},1}, \hat{f}_{\{2\},1}, \hat{f}_{\{1\},1}, \hat{f}_{\{2,3\},1}, \hat{f}_{\{2\},2}, \hat{f}_{\{3\},2}, \hat{f}_{\{1\},2}, \hat{f}_{\{2,3\},2}, \hat{f}_{\{2\},3}, \hat{f}_{\{3\},3}, \hat{f}_{\{1\},3}, \hat{f}_{\{2,3\},3}$

4.6.1 10-dimensional data set

Consider a 10-dimensional, uniformly distributed input set of size N in a $[0,1]^{10}$ design space and $n = 10,000$ random predictive locations generated from the same design space. The deterministic target function

$$f(x_1, \ldots, x_{10}) = \sin(1.5x_1\pi) + 3\cos(3.5x_2\pi) + 5\exp(x_3) + 2\cos(x_2\pi)\sin(x_3\pi)$$

is considered. Note that changes in x_3 have a relatively large influence on the output. Further, x_1, x_2 and x_3 are active while $x_4, \ldots, x_{10}$ are inert. Table 4.1 presents the selected inputs by MRFA in the fitted model for $N = 1,000$. The main effect of x_3 with resolution level one is first entertained, and in the final fitted model ($\lambda = 0.003$) the influential inputs are correctly selected while the irrelevant inputs ($x_4, \ldots, x_{10}$) are also identified (in the sense that they do not appear in the fitted model). Noticeably, our algorithm nicely finds the basis functions which obey strong effect heredity in the final fitted model. For example, $\hat{f}_{\{3\},1}$, $\hat{f}_{\{2\},1}$, and $\hat{f}_{\{2,3\},1}$ are selected in the final fitted model.

Table 4.2 shows the performance of MRFA based on designs of increasing size N, in comparison to `mlegp` and `laGP`. The fitting time of `laGP` is not shown in the example (and the ones in the following sections) because the fitting process of the approach cannot be simply separated from prediction. Note that `mlegp` is only feasible at $N = 1,000$ in the numerical study, so we omit the result in the cases $N > 1000$. In contrast, it can

Table 4.2: Performance of 10-dimensional example with $n = 10,000$ random predictive locations.

	N	Fitting time (sec.)	Prediction time (sec.)	RMSE ($\times 10^{-5}$)	Variable detection
`mlegp`	1,000	1993	158	40.81	-
`laGP`	1,000	-	318	172998	-
	10,000	-	331	71027	-
	100,000	-	331	20437	-
	1,000,000	-	361	6893	-
`MRFA`	1,000	44	6	3.24	100%
	10,000	124	5	1.14	100%
	100,000	1325	5	0.72	100%
	1,000,000	61515	74	0.38	100%

be seen that MRFA is feasible and accurate for large problems. Furthermore, it is *much* faster to fit and predict from and, even in cases when traditional Gaussian process fitting is feasible, more accurate. In this example with several inert input variables, compared to local Gaussian process fitting, even though `laGP` is feasible for large problems, the accuracy of the emulators is not comparable with traditional Gaussian process fitting or MRFA. In particular, MRFA can improve the accuracy at least 10000-fold over the considered sample sizes and it is even faster than local Gaussian process fitting in the cases $N = 1,000$ and $N = 10,000$. In addition, in all examples, the true active variables (i.e., x_1, x_2, x_3 and the interaction effect) are correctly selected, while all inactive variables (i.e., $x_4, \ldots, x_{10}$) are excluded. This example demonstrates that the MRFA method is capable of not only providing an accurate emulator with much cheaper computational cost, but also identifying important variables which can be useful for model interpretation.

To demonstrate the statistical inference discussed in Section 4.4, confidence intervals on emulator predictions are compared. The evaluation includes coverage rate, average width of intervals, and average interval score [140]. Coverage rate is the proportion of the time that the interval contains the true value, while interval score combines the coverage rate and the

Table 4.3: Performance of prediction intervals in the 10-dimensional example at $N = 1,000$ with $n = 10,000$ random predictive locations.

	N	Coverage rate (%)	Average width ($\times 10^{-5}$)	Average interval score ($\times 10^{-5}$)
`mlegp`	1,000	75.09	6139.29	6313.56
`laGP`	1,000	82.18	313469	1324927
	10,000	92.85	126172	392917
	100,000	93.54	53313	85058
	1,000,000	93.20	24073	31478
`MRFA`	1,000	100.00	27.39	27.39
	10,000	98.56	3.12	3.39
	100,000	95.84	2.31	3.06
	1,000,000	97.69	1.64	1.94

width of intervals,

$$S_\alpha(l, u; x) = (u - l) + \frac{2}{\alpha}(l - x)\mathbb{1}\{x < l\} + \frac{2}{\alpha}(x - u)\mathbb{1}\{x > u\},$$

where l and u are the lower and upper confidence limits, and $(1-\alpha)\times 100\%$ is the confidence level. Note that a smaller score corresponds to a better interval.

Continue the above examples and consider 95% confidence intervals. Here, we consider the large N modification to the confidence interval algorithm, as given in Appendix D.4. The unmodified algorithm performs similarly for $N = 1,000$ and $N = 10,000$, but is not feasible for the larger sample sizes. The results of the evaluations are given in Table 4.3. It can be seen that the MRFA intervals have coverage rate close to the nominal coverage 95%, while `mlegp` yields very poor intervals that are both wide and only contain less than 80% of the true values. While `laGP` has a reasonable coverage rate, it yields very wide confidence intervals, which results in a poor interval score. In contrast, the confidence intervals of MRFA perform best in terms of the interval score, given their small width. Notably, the technique of [136] could also be applied to `mlegp` and `laGP` to bring their coverage near target, but their widths would still be much larger than MRFA.

4.6.2 Borehole function

In this subsection, we use a relatively complex target function for a variety of input dimensions to further examine the MRFA in a many-input context. The borehole function [141] represents a model of water flow through a borehole, and has input-output relation

$$f(\mathbf{x}) = \frac{2\pi T_u(H_u - H_l)}{\ln(r/r_w)(1 + \frac{2LT_u}{\ln(r/r_w)r_w^2 K_w} + \frac{T_u}{T_l})},$$

where $r_w \in [0.05, 0.15]$ is the radius of borehole (m), $r \in [100, 50000]$ is the radius of influence (m), $T_u \in [63070, 115600]$ is the transmissivity of upper aquifer (m^2/yr), $H_u \in [990, 1110]$ is the potentiometric head of upper aquifer (m), $T_l \in [63.1, 116]$ is the transmissivity of lower aquifer (m^2/yr), $H_l \in [700, 820]$ is the potentiometric head of lower aquifer (m), $L \in [1120, 1680]$ is the length of borehole (m), and $K_w \in [9855, 12045]$ is the hydraulic conductivity of borehole (m/yr). Here, all inputs are rescaled to the unit hypercube.

Similar to the setup in the previous subsection, N training locations along with $n = 10,000$ predictive locations are randomly generated from a uniform distribution on $[0, 1]^d$. Notice in the borehole experiment, there are eight active variables. We include $d - 8$ irrelevant variables for demonstration. Table 4.5 shows the performance of traditional Gaussian process, local Gaussian process, as well as MRFA based on designs of increasing size N and input dimension d. For a fixed d, the MRFA is feasible and accurate for large problems, while traditional Gaussian process fitting is only feasible for the experiment of size $1,000$. Note that the accuracy for $N = 1,000,000$ can be further improved if more memory allocation is in hand. Alternatives for the case where model fitting exceeds a user's limited budget are discussed in Section 4.7. In addition, in cases when traditional Gaussian process fitting is feasible, the fitting and prediction procedure of MRFA is *much* faster while retaining the accuracy (in some cases MRFA is much more accurate, see $d = 20$ and 60). Similar to the results in the previous subsection, local Gaussian process fitting is feasible

for large problems, but it is less accurate than both traditional Gaussian process and MRFA. With increasing d, the performance of MRFA varies only slightly, while traditional Gaussian process and local Gaussian process fitting perform *substantially* worse with larger d in terms of time cost and accuracy. This result is not surprising, since the irrelevant inputs are screened out (or equivalently, the influential inputs are identified) by our proposed algorithm, as demonstrated in Section 4.6.1. Notice that the $d = 20$ `mlegp` example has very poor accuracy. This example was explored quite extensively and for several random number seeds. In all cases, the likelihood function was highly ill-conditioned, resulting in very low accuracy. This numerical issue was also pointed out in [142].

4.6.3 Stochastic Function

In this subsection, a stochastic function is considered. In particular, this example demonstrates tuning parameter selection. We consider the following function, which was used in [143],

$$f(x_1, x_2, x_3, x_4, x_5, x_6) = \exp\left\{\sin([0.9 \times (x_1 + 0.48)]^{10})\right\} + x_2 x_3 + x_4 + \epsilon, \quad \epsilon \sim \mathcal{N}(0, 0.05^2),$$

where $x_i \in [0, 1], i = 1, \ldots, 6$. The function is nonlinear in x_1, x_2 and x_3, and linear in x_4. In x_1, it oscillates more quickly as it reaches the upper bound of the interval $[0, 1]$. x_5 and x_6 are irrelevant variables.

Here, we consider 5 replicates at each unique training location, $N = 5m$, as indicated in [144], along with $n = 10,000$ unique predictive locations randomly generated from a uniform distribution on $[0, 1]^d$. Since the choice of tuning parameter λ in (4.2) can be particularly crucial in stochastic function emulation, we consider AIC, BIC and 10-fold CV as selection criteria. For the implementation of 10-fold CV, 10 CPUs are requested for parallel computing. Table 4.5 shows the performance of traditional Gaussian process, local Gaussian process, and MRFA with these three selection criterion based on designs

Table 4.4: The borehole example with $n = 10,000$ random predictive locations. *Note that due to memory limits, in these cases $R_{\max} = 3$ and $D_{\max} = 3$ are considered instead.

d	Method	N	Fitting Time (sec.)	Prediction Time (sec.)	RMSE
10	`mlegp`	1,000	9405	99	0.5406
	`laGP`	1,000	-	324	2.2541
		10,000	-	327	1.0952
		100,000	-	326	0.5316
		1,000,000	-	343	0.2667
	`MRFA`	1,000	344	31	0.5659
		10,000	858	15	0.1777
		100,000	8753	72	0.1186
		1,000,000	160326	179	0.0901*
20	`mlegp`	1,000	12358	172	16.4539
	`laGP`	1,000	-	356	10.1838
		10,000	-	359	9.7302
		100,000	-	362	10.0245
		1,000,000	-	429	9.3887
	`MRFA`	1,000	278	24	0.5583
		10,000	786	14	0.1853
		100,000	8443	67	0.1220
		1,000,000	254457	214	0.0924*
60	`mlegp`	1,000	15999	186	3.5841
	`laGP`	1,000	-	599	20.6825
		10,000	-	600	34.3782
		100,000	-	638	45.3728
		1,000,000	-	924	51.2694
	`MRFA`	1,000	534	26	0.7034
		10,000	812	15	0.1770
		100,000	6482	50	0.1312
		1,000,000	150477	90	0.0980*

of increasing size N. It can be seen that, similar to the results in the previous subsections, traditional Gaussian process is only feasible at $N = 1,000$, while MRFA is feasible and accurate for large problems. Even when traditional Gaussian process is feasible, MRFA is much faster in terms of fitting and prediction, and more accurate with any tuning parameter selection method. Local Gaussian process fitting is feasible for large problems, but less accurate than MRFA and traditional Gaussian process. Among the three criterion, it can been seen that AIC, BIC and CV have no substantial difference except for the case $N = 100,000$.

Table 4.5: The 6-dimensional stochastic function example with $n = 10,000$ random predictive locations.

	N	Fitting Time (sec.)	Prediction Time (sec.)		Selection Time (sec.)	RMSE ($\times 10^{-1}$)
`mlegp`	1,000	2524	88			1.64
`laGP`	1,000	-	394			7.30
	10,000	-	439			6.07
	100,000	-	457			4.70
	1,000,000	-	433			3.85
`MRFA`	1,000	96	8	AIC	1	1.36
				BIC	1	1.36
				CV	92	1.32
	10,000	443	23	AIC	1	0.18
				BIC	1	0.19
				CV	423	0.26
	100,000	3752	13	AIC	1	80.53
				BIC	1	0.15
				CV	3801	0.14
	1,000,000	61504	103	AIC	1	0.01
				BIC	1	0.01
				CV	55849	0.05

Computationally, the tuning parameters can be chosen within 2 seconds using AIC or BIC, while the computational costs of CV can be considerable.

4.6.4 Other Functions

In this subsection, we present three more exemplar functions in comparison with `laGP` and `mlegp`, the 3-dimensional bending function [145], the 6-dimensional OTL circuit function [146], and the 10-dimensional wing weight function [147]. The details of these examples and their input ranges are given in Appendix D.8.

The comparison results are shown in Table 4.6. Similar to the results in the previous subsections, the results indicate the MRFA outperforms the traditional Gaussian process in terms of prediction accuracy, except for the wing function at $N = 1,000$ where the traditional Gaussian process fitting has better accuracy. The reason might be that the underlying wing weight function contains high-order interaction functions making it not

particularly well-suited to low-order representation. See (D.23) in the Appendix D.8. Nevertheless, even when the traditional Gaussian process fitting is feasible (at $N = 1,000$), the MRFA is much faster than traditional Gaussian process fitting. Local Gaussian process fitting is feasible for large problems and has better accuracy in the low-dimensional example (see Table 4.6(a)), but it is less accurate in the other two examples and in some cases slower than the MRFA.

4.7 Discussion

While large-scale and many-input nonlinear regression problems have become typical in the modern "big data" context, Gaussian process models are often infeasible due to memory and numeric issues. In this paper, we proposed a multi-resolution functional ANOVA (MRFA) model, which targets a low resolution representation of a low order functional ANOVA, with respect to strong effect heredity, to form an accurate emulator in a large-scale and many-input setting. Implementing a forward-stepwise variable selection technique via the group lasso algorithm, the representation can be efficiently identified without supercomputing resources. Moreover, we provide some theoretical results regarding consistency and inference for a potentially overlapping group lasso problem, which can be applied to the MRFA model. Our numerical study demonstrates that our proposed model not only successfully identifies influential inputs, but also provides accurate predictions for large-scale and many-input problems with a much faster computational time compared to traditional Gaussian process models.

The MRFA model has a similar flavor to multivariate adaptive regression splines (MARS) [44]. On the other hand, the flexibility in basis function choice along resolution levels, forward-stepwise variable selection via group lasso, and confidence interval development for the MRFA, are quite different. Moreover, empirical studies in [146] show the Gaussian process outperforming MARS in terms of prediction accuracy, while our numerical studies show MRFA outperforming Gaussian process.

Table 4.6: Performance of the bending, OTL circuit, and wing weight functions with $n = 10,000$ random predictive locations.

(a) Performance of the 3-dimensional bending function. *Note that due to memory limits, in the cases $R_{\max} = 3$ and $D_{\max} = 3$ are considered instead.

$d = 3$	N	Fitting time (sec.)	Prediction time (sec.)	RMSE ($\times 10^{-5}$)
`mlegp`	1,000	1807	140	5.64
`laGP`	1,000	-	310	0.66
	10,000	-	312	0.21
	100,000	-	311	0.08
	1,000,000	-	316	0.04
`MRFA`	1,000	49	8	2.16
	10,000	293	14	0.46
	100,000	3311	25	0.20
	1,000,000	113279	159	0.14*

(b) Performance of the 6-dimensional OTL circuit function. *Note that due to memory limits, in the cases $R_{\max} = 3$ and $D_{\max} = 3$ are considered instead.

$d = 6$	N	Fitting time (sec.)	Prediction time (sec.)	RMSE ($\times 10^{-4}$)
`mlegp`	1,000	3976	173	13.70
`laGP`	1,000	-	314	102.71
	10,000	-	301	27.01
	100,000	-	323	11.43
	1,000,000	-	328	4.80
`MRFA`	1,000	294	19	7.81
	10,000	798	17	2.05
	100,000	6688	82	1.42
	1,000,000	122075	133	1.18*

(c) Performance of the 10-dimensional wing weight function. *Note that due to memory limits, in the cases $R_{\max} = 1$ and $D_{\max} = 3$ are considered instead.

$d = 10$	N	Fitting time (sec.)	Prediction time (sec.)	RMSE ($\times 10^{-1}$)
`mlegp`	1,000	2922	228	1.56
`laGP`	1,000	-	327	19.74
	10,000	-	325	10.72
	100,000	-	329	5.04
	1,000,000	-	347	2.22
`MRFA`	1,000	1319	28	7.77
	10,000	1633	21	1.52
	100,000	12289	84	1.39
	1,000,000	168854	148	1.18*

The proposed MRFA indicates several avenues for future research. First, when the sample size is too large due to a user's limited budget (e.g., memory limitation), sub-sampling methods can be naturally applied to the MRFA approach. For example, [148] proposed *pasting Rvotes* and *pasting Ivotes* methods, which use random sampling and importance sampling, respectively. Moreover, *m-out-of-n bagging* (also known as *subagging*) [149, 150, 151] uses sub-samples for aggregation and might be expected to have similar accuracy to bagging, which uses bootstrap samples to improve the accuracy of prediction [152]. These sub-sampling methods provide the potential to extend the MRFA model to even larger data sets.

Next, if the basis functions are constructed by integrating the full-dimensional kernel over margins as indicated in Theorem 4.2.1, one may consider the native space norm with kernel Φ instead of the 2-norm in the penalized loss function (4.2). In fact, both norms were examined in our numeric studies and the results indicated that the penalized loss function with respective to the native space norm may increase computational costs without too much improvement in prediction accuracy. For example, for the 10-dimensional example in Section 4.6.1, with $N = 1,000$, the fitting with the native space norm costs 47 seconds while fitting with the 2-norm only costs 15 seconds, and both result in roughly the same RMSE.

Last but not least, it is conceivable that the MRFA approach can be generalized to a non-continuous, for example binary, response. One might proceed by replacing the residual sum of squares in (4.2) by the corresponding negative log-likelihood function, and extending the group lasso algorithm to other exponential families, as done in [123]. The inference results, however, cannot be directly applied to a non-continuous response.

Appendices

APPENDIX A

APPENDICES OF CHAPTER 1

A.1 Algorithm: Estimation of $(\boldsymbol{\beta}, \boldsymbol{\omega})$

1: Set initial values $\boldsymbol{\omega} = (\sigma^2, \boldsymbol{\theta}) = \mathbf{1}_{d+1}, \boldsymbol{\beta} = \mathbf{1}_m, p_{it} = 1$, and set $\tilde{\eta}_{it} = \log \frac{p_{it}}{1-p_{it}} + \frac{y_{it}-p_{it}}{p_{it}(1-p_{it})}$ for each i and t.

2: **repeat**

3: **repeat**

4: Set $\boldsymbol{W}$ as an $N \times N$ diagonal matrix with diagonal elements $W_{it} = p_{it}(1 - p_{it})$

5: Set $\boldsymbol{V} = \boldsymbol{W}^{-1} + \sigma^2(\boldsymbol{R_\theta} \otimes I_T)$

6: Update $\boldsymbol{\beta} = (\boldsymbol{X}'\boldsymbol{V}^{-1}\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{V}^{-1}\tilde{\boldsymbol{\eta}}$

7: Set $\boldsymbol{Z} = \sigma^2(\boldsymbol{R_\theta} \otimes I_T)\boldsymbol{V}^{-1}(\tilde{\boldsymbol{\eta}} - \boldsymbol{X}'\boldsymbol{\beta})$

8: Update $p_{it} = \left(\frac{\exp\{\boldsymbol{X}'\boldsymbol{\beta}+\boldsymbol{Z}\}}{\mathbf{1}_N+\exp\{\boldsymbol{X}'\boldsymbol{\beta}+\boldsymbol{Z}\}}\right)_{it}$ and $\tilde{\eta}_{it} = \log \frac{p_{it}}{1-p_{it}} + \frac{y_{it}-p_{it}}{p_{it}(1-p_{it})}$ for each i and t

9: **until** $\{\tilde{\eta}_{it}\}_{it}$ converges

10: Update $\boldsymbol{\omega} = \arg\min_{\boldsymbol{\omega}} L(\boldsymbol{\omega})$, where $L(\boldsymbol{\omega})$ is the negative log-likelihood function (1.12)

11: Update $(\sigma^2, \boldsymbol{\theta}) = \boldsymbol{\omega}$

12: **until** $\boldsymbol{\beta}$ and $\boldsymbol{\omega}$ converge

13: Return $\boldsymbol{\beta}$ and $\boldsymbol{\omega}$

A.2 Assumptions

1. The parameter $\boldsymbol{\beta}$ belongs to an open set $B \subseteq \mathbb{R}^m$ and the parameter $\boldsymbol{\omega}$ belongs to an open set $\Omega \subseteq \mathbb{R}^{d+1}$.

2. The model matrix X_{it} lies almost surely in a nonrandom compact subset of $\mathbb{R}^m$ such

that $Pr(\sum_{i=1}^{n}\sum_{t=1}^{T} X_{it}'X_{it} > 0) = 1$.

For any matrix A, define $\|A\| \equiv \sqrt{\text{tr}(A'A)}$; for the covariance matrix $\boldsymbol{V}(\boldsymbol{\omega})$, define $V_i(\boldsymbol{\omega}) \equiv \partial \boldsymbol{V}(\boldsymbol{\omega})/\partial\omega_i$ and $V_{ij}(\boldsymbol{\omega}) \equiv \partial \boldsymbol{V}(\boldsymbol{\omega})/\partial\omega_i\partial\omega_j$; for $\boldsymbol{\omega} \in \Omega$, denote $\xrightarrow{u}$ as uniform convergence of nonrandom functions over compact subsets of Ω.

3. $J_N(\boldsymbol{\omega})P_N(\boldsymbol{\omega})^{-1} \xrightarrow{d} W(\boldsymbol{\omega})$ for some nonsingular $W(\boldsymbol{\omega})$, which is continuous in $\boldsymbol{\omega}$, where $P_N(\boldsymbol{\omega}) = \text{diag}(\|\Pi(\boldsymbol{\omega})V_1(\boldsymbol{\omega})\|, \ldots, \|\Pi(\boldsymbol{\omega})V_{d+1}(\boldsymbol{\omega})\|)$ and $\Pi(\boldsymbol{\omega}) = \boldsymbol{V}(\boldsymbol{\omega})^{-1} - \boldsymbol{V}(\boldsymbol{\omega})^{-1}\boldsymbol{X}(\boldsymbol{X}'\boldsymbol{V}(\boldsymbol{\omega})^{-1}\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{V}(\boldsymbol{\omega})^{-1}$.

4. If there exists a sequence $\{r_N\}_{N\geq 1}$ with $\limsup_{N\to\infty} r_N/N \leq 1-\delta$, for some $\delta \in (0,1)$, such that for any compact subset $K \subseteq \Omega$, there exist constants $0 < C_1(K) < \infty$ and $C_2(K) > 0$ such that

$$\lim_{N\to\infty}\sup \max\{|\lambda_N|, |\lambda_N^i|, |\lambda_N^{ij}| : 1 \leq i,j \leq k\} < C_1(K) < \infty$$

and

$$\lim_{N\to\infty}\sup \min\{|\lambda_1|, |\lambda_{r_N}^i| : 1 \leq i \leq k\} > C_2(K) > 0,$$

uniformly in $\boldsymbol{\omega} \in K$, where $|\lambda_1| \leq \ldots \leq |\lambda_N|$ are the absolute eigenvalues of $\boldsymbol{V}(\boldsymbol{\omega})$, $|\lambda_1^i| \leq \ldots \leq |\lambda_N^i|$ are the absolute eigenvalues of $V_i(\boldsymbol{\omega})$, and $|\lambda_1^{ij}| \leq \ldots \leq |\lambda_N^{ij}|$ are the absolute eigenvalues of $V_{ij}(\boldsymbol{\omega})$.

Assumption 2 holds when the row vectors of $\boldsymbol{X}$ are linear independent. Thus, if only the linear effect is considered in the mean function, then orthogonal designs or orthogonal array-based designs, such as OA-based Latin hypercube designs [51], can be chosen for sampling schemes. The conditions for Assumption 4 can be referred to [38], in which the checkable conditions for rectangular lattice of data sites and irregularly located data sites are given. For instance, for rectangular lattice of data sites, with certain correlation functions, a sufficient condition is choosing data locations whose minimum distance is sufficiently

large. More details can be seen in [38]. Thus, space-filling designs, such as Latin hypercube designs [153] and maximin distance designs [154], can be chosen for sampling schemes.

A.3 Proof of Theorem 1.3.1

The model (1.4) can be seen as a binary time series model with random effects by multiplying an identity matrix on $\boldsymbol{Z}$, that is,

$$\text{logit}(\boldsymbol{p}) = \boldsymbol{X}\boldsymbol{\beta} + I_N\boldsymbol{Z}, \quad \boldsymbol{Z} \sim \mathcal{N}(\boldsymbol{0}_N, \Sigma(\boldsymbol{\omega})),$$

where I_N and $\boldsymbol{Z}$ are viewed as the model matrix and coefficients of random effects, respectively. Therefore, if the variance-covariance parameters are given, inference of $\boldsymbol{\beta}$ is a special case of the binary time series model with random effects in [4]. Therefore, following Theorem 1 in [4], the score function $S_N(\boldsymbol{\beta}, \boldsymbol{\omega})$ is asymptotically normally distributed.

A.4 Proof of Theorem 1.3.3

According to [30], one can view the inference on the variance-variance component as an iterative procedure for the linear mixed model

$$\tilde{\boldsymbol{\eta}} = \boldsymbol{X}\boldsymbol{\beta} + I_N\boldsymbol{Z} + \epsilon, \quad \epsilon \sim \mathcal{N}(\boldsymbol{0}_N, \boldsymbol{W}^{-1})$$

with the iterative weight $\boldsymbol{W}^{-1}$. Thus, it is a special case of the Gaussian general linear model in [37] with response vector $\tilde{\boldsymbol{\eta}}$ and variance-covariance component $\Sigma(\boldsymbol{\omega}) + \boldsymbol{W}^{-1}$ with parameters $\boldsymbol{\omega}$. Since the asymptotic distribution of REML estimators for the variance-covariance parameters has been shown in [37] for a Gaussian general linear model, the result directly follows as a special case of Corollary 3.3 in [37]. Note that Assumption 4 in the supplementary material A.2 implies the conditions for Corollary 3.3 in [37]. See the proof of Theorem 2.2 in [38].

A.5 Proof of Lemma 1.4.1

We start the proof by deriving the conditional distribution from a simple model (1.1) (without time-series), and then extend the result to prove Lemma 1.4.1. First, a definition and a lemma about multivariate log-normal distribution are in order.

Definition A.5.1. Suppose $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_n)'$ has a multivariate normal distribution with mean $\boldsymbol{\mu}_n$ and covariance variance $\boldsymbol{\Sigma}_{n\times n}$. Then $\boldsymbol{b} = \exp\{\boldsymbol{\xi}\}$ has a *multivariate log-normal distribution*. Denote it as $\boldsymbol{b} \sim \mathcal{LN}(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_{n\times n})$.

Lemma A.5.1. *Suppose $\boldsymbol{b}^n$ and b_{n+1} have a multivariate log-normal distribution*

$$\begin{pmatrix} \boldsymbol{b}^n \\ b_{n+1} \end{pmatrix} \sim \mathcal{LN}\left(\begin{pmatrix} \boldsymbol{\mu}^n \\ \mu_{n+1} \end{pmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{n\times n} & \boldsymbol{r} \\ \boldsymbol{r}' & \sigma^2_{n+1} \end{bmatrix} \right).$$

The conditional distribution of b_{n+1} given $\boldsymbol{b}^n$ is $b_{n+1}|\boldsymbol{b}^n \sim \mathcal{LN}(\mu^*, v^*)$, where $\mu^* = \mu_{n+1} + \boldsymbol{r}'\boldsymbol{\Sigma}^{-1}_{n\times n}(\log \boldsymbol{b}^n - \boldsymbol{\mu}^n)$ and $v^* = \sigma^2_{n+1} - \boldsymbol{r}'\boldsymbol{\Sigma}^{-1}_{n\times n}\boldsymbol{r}$.

Proof. Using transformation of a standard normal distribution, one can show that the joint probability density function of the multivariate log-normal distribution $\boldsymbol{b}^n$ is

$$g_{\boldsymbol{b}^n}(b_1, \ldots, b_n) = \frac{1}{(2\pi)^{n/2}|\boldsymbol{\Sigma}_{n\times n}|^{1/2}} \frac{1}{\prod_{i=1}^n b_i} \exp\{-\frac{1}{2}\left(\log \boldsymbol{b}^n - \boldsymbol{\mu}_n\right)' \boldsymbol{\Sigma}^{-1}_{n\times n}\left(\log \boldsymbol{b}^n - \boldsymbol{\mu}_n\right)\}.$$

Denote $\boldsymbol{b}^{n+1} = (b_1, \ldots, b_n, b_{n+1})$, $\boldsymbol{\mu}^{n+1} = (\mu_1, \ldots, \mu_n, \mu_{n+1})$ and

$$\boldsymbol{\Sigma}_{(n+1)\times(n+1)} = \begin{bmatrix} \boldsymbol{\Sigma}_{n\times n} & \boldsymbol{r} \\ \boldsymbol{r}' & \sigma_{n+1} \end{bmatrix}.$$

Then, the conditional probability density function of b_{n+1} given $\boldsymbol{b}^n$ can be derived as

$$\begin{aligned} g_{b_{n+1}|\boldsymbol{b}^n}(b_{n+1}|\boldsymbol{b}^n) &\propto g(b_1,\ldots,b_n,b_{n+1}) \\ &\propto \frac{1}{b_{n+1}}\exp\{-\frac{1}{2}\left(\log \boldsymbol{b}^{n+1}-\boldsymbol{\mu}_{n+1}\right)'\boldsymbol{\Sigma}^{-1}_{(n+1)\times(n+1)}\left(\log \boldsymbol{b}^{n+1}-\boldsymbol{\mu}_{n+1}\right)\}. \end{aligned}$$

Let $\boldsymbol{a}_1 = \log \boldsymbol{b}^n - \boldsymbol{\mu}^n$ and $\boldsymbol{a}_2 = \log b^{n+1} - \mu^{n+1}$. Applying the partitioned matrix inverse results (page 99 of [86]) gives

$$\begin{aligned} &\left(\log \boldsymbol{b}^{n+1}-\boldsymbol{\mu}^{n+1}\right)'\boldsymbol{\Sigma}^{-1}_{(n+1)\times(n+1)}\left(\log \boldsymbol{b}^{n+1}-\boldsymbol{\mu}^{n+1}\right) \\ =&\begin{bmatrix}\boldsymbol{a}_1' & \boldsymbol{a}_2'\end{bmatrix}\begin{bmatrix}\boldsymbol{\Sigma}_{n\times n} & \boldsymbol{r}\\ \boldsymbol{r}' & \sigma_{n+1}\end{bmatrix}^{-1}\begin{bmatrix}\boldsymbol{a}_1\\ \boldsymbol{a}_2\end{bmatrix} \\ =&(\boldsymbol{a}_2-\boldsymbol{r}'\boldsymbol{\Sigma}^{-1}_{n\times n}\boldsymbol{a}_1)'\sigma^{-1}_{22\cdot 1}(\boldsymbol{a}_2-\boldsymbol{r}'\boldsymbol{\Sigma}^{-1}_{n\times n}\boldsymbol{a}_1)+\boldsymbol{a}_1'\boldsymbol{\Sigma}^{-1}_{n\times n}\boldsymbol{a}_1 \\ =&(\boldsymbol{a}_2-\boldsymbol{r}'\boldsymbol{\Sigma}^{-1}_{n\times n}\boldsymbol{a}_1)^2/\sigma_{22\cdot 1}+\boldsymbol{a}_1'\boldsymbol{\Sigma}^{-1}_{n\times n}\boldsymbol{a}_1, \end{aligned}$$

where $\sigma_{22\cdot 1} = \sigma^2_{n+1} - \boldsymbol{r}'\boldsymbol{\Sigma}^{-1}_{n\times n}\boldsymbol{r}$ and is a real number.

Thus, the conditional probability density function of b_{n+1} given $\boldsymbol{b}^n$ can be simplified as

$$\begin{aligned} g_{b_{n+1}|\boldsymbol{b}^n}(b_{n+1}|\boldsymbol{b}^n) &\propto \frac{1}{b_{n+1}}\exp\{-\frac{1}{2\sigma_{22\cdot 1}}(\boldsymbol{a}_2-\boldsymbol{r}'\boldsymbol{\Sigma}^{-1}_{n\times n}\boldsymbol{a}_1)^2-\frac{1}{2}\boldsymbol{a}_1'\boldsymbol{\Sigma}^{-1}_{n\times n}\boldsymbol{a}_1\} \\ &\propto \frac{1}{b_{n+1}}\exp\{-\frac{1}{2\sigma_{22\cdot 1}}(\boldsymbol{a}_2-\boldsymbol{r}'\boldsymbol{\Sigma}^{-1}_{n\times n}\boldsymbol{a}_1)^2\} \\ &= \frac{1}{b_{n+1}}\exp\{-\frac{1}{2\sigma_{22\cdot 1}}\left(\log b_{n+1}-(\mu_{n+1}+\boldsymbol{r}'\boldsymbol{\Sigma}^{-1}_{n\times n}(\log \boldsymbol{b}^n-\boldsymbol{\mu}^n))\right)^2\}. \end{aligned}$$

Therefore, according to the probability density function of a log-normal distribution, we have $b_{n+1}|\boldsymbol{b}^n \sim \mathcal{LN}(\mu^*, v^*)$, where $\mu^* = \mu_{n+1} + \boldsymbol{r}'\boldsymbol{\Sigma}^{-1}_{n\times n}(\log \boldsymbol{b}^n - \boldsymbol{\mu}^n)$ and $v^* = \sigma_{22\cdot 1} = \sigma^2_{n+1} - \boldsymbol{r}'\boldsymbol{\Sigma}^{-1}_{n\times n}\boldsymbol{r}$. □

Lemma A.5.2. *Consider the model* (1.1) *(without time-series), given* $(p(\mathbf{x}_1),\ldots,p(\mathbf{x}_n))' = \boldsymbol{p}^n$*, the conditional distribution of* $p(\mathbf{x}_{n+1})$ *is a logit-normal distribution, that is,* $p(\mathbf{x}_{n+1})|\boldsymbol{p}^n \sim$

$Logitnormal(m(\boldsymbol{p}^n), v(\boldsymbol{p}^n))$ *with*

$$m(\boldsymbol{p}^n) = \mu(\mathbf{x}_{n+1}) + \boldsymbol{r}_{\boldsymbol{\theta}}'\boldsymbol{R}_{\boldsymbol{\theta}}^{-1}(\log\frac{\boldsymbol{p}^n}{\mathbf{1}-\boldsymbol{p}^n} - \boldsymbol{\mu}^n) \quad \textit{and} \quad v(\boldsymbol{p}^n) = \sigma^2(1-\boldsymbol{r}_{\boldsymbol{\theta}}'\boldsymbol{R}_{\boldsymbol{\theta}}^{-1}\boldsymbol{r}_{\boldsymbol{\theta}}),$$

where $\boldsymbol{\mu}^n = (\mu(\mathbf{x}_1), \ldots, \mu(\mathbf{x}_n))'$, $\mu(\mathbf{x}_i) = \alpha_0 + \mathbf{x}_i'\boldsymbol{\alpha}$, $\boldsymbol{r}_{\boldsymbol{\theta}} = (R_{\boldsymbol{\theta}}(\mathbf{x}_{n+1}, \mathbf{x}_1), \ldots, R_{\boldsymbol{\theta}}(\mathbf{x}_{n+1}, \mathbf{x}_n))'$, *and* $\boldsymbol{R}_{\boldsymbol{\theta}} = \{R_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{x}_j)\}$.

Proof. Let $\eta_i = \mu(\mathbf{x}_i) + Z(\mathbf{x}_i)$ and $b_i = \exp\{\eta_i\} = p(\mathbf{x}_i)/(1-p(\mathbf{x}_i))$ for $i = 1, \ldots, n+1$. Since $(\eta_1, \ldots, \eta_n, \eta_{n+1})' \sim \mathcal{N}(\boldsymbol{\mu}^{n+1}, \sigma^2\boldsymbol{R}_{\boldsymbol{\theta}}^*)$, where $\boldsymbol{\mu}^{n+1} = ((\boldsymbol{\mu}^n)', \mu(\mathbf{x}_{n+1}))'$ and

$$\boldsymbol{R}_{\boldsymbol{\theta}}^* = \begin{bmatrix} \boldsymbol{R}_{\boldsymbol{\theta}} & \boldsymbol{r}_{\boldsymbol{\theta}} \\ \boldsymbol{r}_{\boldsymbol{\theta}}' & 1 \end{bmatrix},$$

we have $(b_1, \ldots, b_n, b_{n+1})' \sim \mathcal{LN}(\boldsymbol{\mu}^{n+1}, \sigma^2\boldsymbol{R}_{\boldsymbol{\theta}}^*)$ by Definition A.5.1. Thus, using Jacobian of the transformation and Lemma A.5.1, we have

$$\begin{aligned}
&g_{p(\mathbf{x}_{n+1})|p(\mathbf{x}_1),\ldots,p(\mathbf{x}_n)}(p_{n+1}|p_1, \ldots, p_n) \\
=&g_{b_{n+1}|b_1,\ldots,b_n}(\frac{p_{n+1}}{1-p_{n+1}}|\frac{p_1}{1-p_1}, \ldots, \frac{p_n}{1-p_n})\frac{1}{(1-p_{n+1})^2} \\
\propto&\frac{1-p_{n+1}}{p_{n+1}}\exp\{-\frac{\left(\log\frac{p_{n+1}}{1-p_{n+1}} - (\mu(\mathbf{x}_{n+1}) + \boldsymbol{r}_{\boldsymbol{\theta}}'\boldsymbol{R}_{\boldsymbol{\theta}}^{-1}(\log\frac{\boldsymbol{p}^n}{\mathbf{1}-\boldsymbol{p}^n} - \boldsymbol{\mu}^n))\right)^2}{2\sigma^2(1-\boldsymbol{r}_{\boldsymbol{\theta}}'\boldsymbol{R}_{\boldsymbol{\theta}}^{-1}\boldsymbol{r}_{\boldsymbol{\theta}})}\}\frac{1}{(1-p_{n+1})^2} \\
\propto&\frac{1}{p_{n+1}(1-p_{n+1})}\exp\{-\frac{\left(\log\frac{p_{n+1}}{1-p_{n+1}} - (\mu(\mathbf{x}_{n+1}) + \boldsymbol{r}_{\boldsymbol{\theta}}'\boldsymbol{R}_{\boldsymbol{\theta}}^{-1}(\log\frac{\boldsymbol{p}^n}{\mathbf{1}-\boldsymbol{p}^n} - \boldsymbol{\mu}^n))\right)^2}{2\sigma^2(1-\boldsymbol{r}_{\boldsymbol{\theta}}'\boldsymbol{R}_{\boldsymbol{\theta}}^{-1}\boldsymbol{r}_{\boldsymbol{\theta}})}\}.
\end{aligned}$$

Therefore, according to the probability density function of a logit-normal distribution, we have $p(\mathbf{x}_{n+1})|\boldsymbol{p}^n \sim Logitnormal(m(\boldsymbol{p}^n), v(\boldsymbol{p}^n))$. □

Similarly, the result of Lemma A.5.2 can be extended to the general model (1.3). Given $\boldsymbol{Y} = (\mathbf{y}_1', \ldots, \mathbf{y}_T', y_{n+1,1}, \ldots, y_{n+1,s-1})'$, at a fixed time-step s, $p_s(\mathbf{x}_i)$ can be seen to have the model (1.1) with mean function $\mu(\mathbf{x}_i, \boldsymbol{Y}) = \sum_{r=1}^{R}\varphi_r y_{i,s-r} + \alpha_0 + \mathbf{x}_i'\boldsymbol{\alpha} + \sum_{l=1}^{L}\boldsymbol{\gamma}_l\mathbf{x}_i y_{i,s-l}$.

Thus, by Lemma A.5.2, denote $\boldsymbol{p}_s = (p_s(\mathbf{x}_1), \ldots, p_s(\mathbf{x}_n))'$, we have

$$p_s(\mathbf{x}_{n+1})|\boldsymbol{p}_s, \boldsymbol{Y} \sim Logitnormal(m(\boldsymbol{p}_s, \boldsymbol{Y}), v(\boldsymbol{p}_s, \boldsymbol{Y})),$$

where $m(\boldsymbol{p}_s, \boldsymbol{Y}) = \mu(\mathbf{x}_{n+1}, \boldsymbol{Y}) + \boldsymbol{r}_{\boldsymbol{\theta}}' \boldsymbol{R}_{\boldsymbol{\theta}}^{-1}(\log \frac{\boldsymbol{p}_s}{\mathbf{1}-\boldsymbol{p}_s} - \boldsymbol{\mu}^n)$, $\boldsymbol{\mu}^n = (\mu(\mathbf{x}_1, \boldsymbol{Y}), \ldots, \mu(\mathbf{x}_n, \boldsymbol{Y}))'$, and $v(\boldsymbol{p}_s, \boldsymbol{Y}) = \sigma^2(1 - \boldsymbol{r}_{\boldsymbol{\theta}}' \boldsymbol{R}_{\boldsymbol{\theta}}^{-1} \boldsymbol{r}_{\boldsymbol{\theta}})$. By the fact that $Z_t(\mathbf{x})$ is independent over time, which implies $p_s(\mathbf{x})$ is independent of $p_t(\mathbf{x})$ for any $t \neq s$, $p_s(\mathbf{x}_{n+1})|D_{n+1,s}$ and $p_s(\mathbf{x}_{n+1})|\boldsymbol{p}_s, \boldsymbol{Y}$ have the same distribution. So, $p_s(\mathbf{x}_{n+1})|D_{n+1,s} \sim Logitnormal(m(D_{n+1,s}), v(D_{n+1,s}))$, where $m(D_{n+1,s}) = m(\boldsymbol{p}_s, \boldsymbol{Y})$ and $v(D_{n+1,s}) = v(\boldsymbol{p}_s, \boldsymbol{Y})$.

A.6 Proof of Theorem 1.4.3

(i) First, one can show that if $(p_s(\mathbf{x}_{n+1}), D_{n+1,s})$ has a joint distribution for which the conditional mean of $p_s(\mathbf{x}_{n+1})$ given $D_{n+1,s}$ exists, then $\mathbb{E}[p(\mathbf{x}_{n+1})|D_{n+1,s}]$ is the minimum mean squared error predictor of $p(\mathbf{x}_{n+1})$. See Theorem 3.2.1 in [5]. Thus, by the result of Lemma 1.4.1, we have the conditional mean $\mathbb{E}[p(\mathbf{x}_{n+1})|D_{n+1,s}] = \kappa(m(D_{n+1,s}), v(D_{n+1,s}))$ with variance $\mathbb{V}[p(\mathbf{x}_{n+1})|D_{n+1,s}] = \tau(m(D_{n+1,s}), v(D_{n+1,s}))$.

(ii) If $\mathbf{x}_{n+1} = \mathbf{x}_i$ for $i = 1, \ldots, n$, then $m(D_{n+1,s}) = \log(p_s(\mathbf{x}_i)/(1 - p_s(\mathbf{x}_i)))$ and $v(D_{n+1,s}) = 0$, which implies that

$$\kappa(m(D_{n+1,s}), 0) = \exp\{m(D_{n+1,s})\}/(1 + \exp\{m(D_{n+1,s})\}) = p_s(\mathbf{x}_i)$$

and $\tau(m(D_{n+1,s}), 0) = 0$ by using transformation of a normal distribution. Thus, by Theorem 1.4.3 (i), we have $\mathbb{E}[p_s(\mathbf{x}_{n+1})|D_{n+1,s}] = p_s(\mathbf{x}_i)$ and $\mathbb{V}[p_s(\mathbf{x}_{n+1})|D_{n+1,s}] = 0$.

(iii) Let $X \sim \mathcal{N}(m(D_{n+1,s}), v(D_{n+1,s}))$, $P = \exp\{X\}/(1 + \exp\{X\})$, which has the distribution $Logitnormal(m(D_{n+1,s}), v(D_{n+1,s}))$, and $Q(q; D_{n+1,s})$ be the q-th quantile of P. Consider the function $f(x) = \log(x/(1-x))$. The derivative is $f'(x) = 1/(x(1-x))$.

Thus, for $0 < x < 1$ the derivative is positive and the $f(x)$ function is increasing in x. Then,

$$
\begin{aligned}
& Pr\left\{P > Q(q; D_{n+1,s})\right\} = q \\
\Leftrightarrow & Pr\left\{\frac{\exp\{X\}}{1+\exp\{X\}} > Q(q; D_{n+1,s})\right\} = q \\
\Leftrightarrow & Pr\left\{f(\frac{\exp\{X\}}{1+\exp\{X\}}) > f(Q(q; D_{n+1,s}))\right\} = q \\
\Leftrightarrow & Pr\left\{X > \log\frac{Q(q; D_{n+1,s})}{1-Q(q; D_{n+1,s})}\right\} = q
\end{aligned}
$$

$$
\begin{aligned}
\Leftrightarrow & Pr\left\{\frac{X - m(D_{n+1,s})}{\sqrt{v(D_{n+1,s})}} > \frac{1}{\sqrt{v(D_{n+1,s})}}\left(\log\frac{Q(q; D_{n+1,s})}{1-Q(q; D_{n+1,s})} - m(D_{n+1,s})\right)\right\} = q \\
\Leftrightarrow & \frac{1}{\sqrt{v(D_{n+1,s})}}\left(\log\frac{Q(q; D_{n+1,s})}{1-Q(q; D_{n+1,s})} - m(D_{n+1,s})\right) = z_q \\
\Leftrightarrow & Q(q; D_{n+1,s}) = \frac{\exp\{m(D_{n+1,s}) + z_q\sqrt{v(D_{n+1,s})}\}}{1+\exp\{m(D_{n+1,s}) + z_q\sqrt{v(D_{n+1,s})}\}}.
\end{aligned}
$$

A.7 Algorithm: Metropolis-Hastings Algorithm

for $j = 1$ to J **do**
  Set $N_s = nT + s - 1$.
  Start with a zero vector $\boldsymbol{p}$ of size N_s.
  for $k = 1$ to N_s **do**
    Generate a random value p_k^* from $Logitnormal(m(\boldsymbol{p}_{-k}, \mathbf{y}_{-k}), v(\boldsymbol{p}_{-k}, \mathbf{y}_{-k}))$.
    Generate an uniform random variable $U \sim Unif(0, 1)$.
    if $U < \min\{1, \frac{f(y_k|p_k^*)}{f(y_k|p_k)}\}$ **then**
      Set $\boldsymbol{p} = (p_1, \ldots, p_k^*, \ldots, p_{N_s})$.
  Set $\boldsymbol{p}^{(j)} = \boldsymbol{p}$
Return $\{\boldsymbol{p}^{(j)}\}_{j=1,\ldots,J}$.

In the algorithm, we first sample a value for the k-th component p_k from the conditional distribution of p_k given $p_j, y_j, j \neq k$, which is $Logitnormal(m(\boldsymbol{p}_{-k}, \mathbf{y}_{-k}), v(\boldsymbol{p}_{-k}, \mathbf{y}_{-k}))$, where

$$m(\boldsymbol{p}_{-k}, \mathbf{y}_{-k}) = \mu_t(\mathbf{x}_i) - \sum_{k \neq j} \frac{Q_{kj}}{Q_{kk}} \left(\log \frac{p_k}{1 - p_k} - \mu_t(\mathbf{x}_i) \right), \quad v(\boldsymbol{p}_{-k}, \mathbf{y}_{-k}) = \frac{\sigma^2}{Q_{kk}},$$

in which $\mu_t(\mathbf{x}_i) = \sum_{r=1}^{R} \varphi_r y_{i,t-r} + \mathbf{x}_i' \boldsymbol{\alpha} + \sum_{l=1}^{L} \boldsymbol{\gamma}_l \mathbf{x}_i y_{i,t-l}$ and Q_{kj} is the (k, j)-element of $\boldsymbol{R}_{\boldsymbol{\theta}}^{-1}$. Similar to [8], we use the single-component MH algorithm, that is, to update only a single component at each iteration. Moreover, the proposed distribution $f(p_k)$ is used for the single MH algorithm, so that the probability of accepting a new p_k^* is the minimum of 1 and $\frac{f(p_k^*|y_k) f(p_k)}{f(p_k|y_k) f(p_k^*)} \left(= \frac{f(y_k|p_k^*)}{f(y_k|p_k)} \right)$.

A.8 Algorithm: Dynamic Binary Emulator

for $j = 1$ to J **do**
  Set $N = nT$.
  Start with a zero vector $\boldsymbol{p}$ of size N.
  for $i = 1$ to N **do**
    Generate a random value p_k^* from $Logitnormal(m(\boldsymbol{p}_{-k}, \mathbf{y}_{-k}), v(\boldsymbol{p}_{-k}, \mathbf{y}_{-k}))$.
    Generate an uniform random variable $U \sim Unif(0, 1)$.
    if $U < \min\{1, \frac{f(y_k|p_k^*)}{f(y_k|p_k)}\}$ **then**
      Set $\boldsymbol{p} = (p_1, \ldots, p_k^*, \ldots, p_N)$.
  Set $\boldsymbol{p}_{n+1} = \boldsymbol{p}, \boldsymbol{Y}_{n+1} = \boldsymbol{Y}$, zero vectors $\boldsymbol{p}_{\text{new}}$ and $\mathbf{y}_{\text{new}}$ of size T.
  for $t = 1$ to T **do**
    Given $D_{n+1,t} = \{\boldsymbol{p}_{n+1}, \boldsymbol{Y}_{n+1}\}$, draw a sample $p_t(\mathbf{x}_{n+1})$ from $Logitnormal(m(D_{n+1,t}), v(D_{n+}$ and then draw a sample $y_t(\mathbf{x}_{n+1})$ from a Bernoulli distribution with parameter $p_t(\mathbf{x}_{n+1})$.
    Update $\boldsymbol{p}_{n+1} = (\boldsymbol{p}'_{n+1}, p_t(\mathbf{x}_{n+1}))'$, $\boldsymbol{Y}_{n+1} = (\boldsymbol{Y}'_{n+1}, y_t(\mathbf{x}_{n+1}))'$, $(\boldsymbol{p}_{\text{new}})_t = p_t(\mathbf{x}_{n+1})$, and $(\mathbf{y}_{\text{new}})_t = y_t(\mathbf{x}_{n+1})$.

13: Set $\boldsymbol{p}_{\text{new}}^{(j)} = \boldsymbol{p}_{\text{new}}$ and $\mathbf{y}_{\text{new}}^{(j)} = \mathbf{y}_{\text{new}}$.

14: Take pointwise median from $\{\boldsymbol{p}_{\text{new}}^{(j)}\}_{j=1,\ldots,J}$ and $\{\mathbf{y}_{\text{new}}^{(j)}\}_{j=1,\ldots,J}$.

APPENDIX B

APPENDICES OF CHAPTER 2

B.1 Assumptions

The regularity conditions on the models are given below. For any $\theta \in \Theta \subset \mathbb{R}^q$, write $\theta = (\theta_1, \ldots, \theta_q)$. Denote $e_i = y_i^p - \eta(\mathbf{x}_i)$.

A1: The sequences $\{\mathbf{x}_i\}$ and $\{e_i\}$ are independent; x_i's are i.i.d. from a uniform distribution over Ω; and $\{e_i\}$ is a sequence of i.i.d. random variables with zero mean and finite variance.

A2: θ^* is the unique solution to (2.2) and is an interior point of Θ.

A3: $V := \mathbb{E}\left[\frac{\partial^2}{\partial\theta\partial\theta^T}(\eta(X) - p(X,\theta^*))^2\right]$ is invertible.

A4: There exists a neighborhood $U \subset \Theta$ of θ^* such that

$$\sup_{\theta\in U}\left\|\frac{\partial p}{\partial\theta_i}(\cdot,\theta)\right\|_{\mathcal{N}_\Phi(\Omega)} < +\infty, \quad \frac{\partial^2 p}{\partial\theta_i\partial\theta_j}(\cdot,\cdot) \in C(\Omega\times U),$$

for all $\theta \in U$ and all $i, j = 1, \ldots, q$.

B.2 Proof of Theorem 2.3.3

Proof. The proof is developed along the lines described in Theorem 1 of [23]. We first prove the consistency, $\hat{\theta}_n \xrightarrow{p} \theta^*$. It suffices to prove that $\|\hat{\eta}_n(\cdot) - \hat{p}_N(\cdot,\theta)\|_{L_2(\Omega)}$ converges

to $\|\eta(\cdot) - p(\cdot,\theta)\|_{L_2(\Omega)}$ uniformly with respect to $\theta \in \Theta$ in probability, which is ensured by

$$\begin{aligned}
&\|\hat{\eta}_n(\cdot) - \hat{p}_N(\cdot,\theta)\|^2_{L_2(\Omega)} - \|\eta(\cdot) - p(\cdot,\theta)\|^2_{L_2(\Omega)} \qquad \text{(B.1)}\\
&= \int_\Omega \left(\hat{\eta}_n(z) - \eta(z) - \hat{p}_N(z,\theta) + p(z,\theta)\right)\left(\hat{\eta}_n(z) + \eta(z) - \hat{p}_N(z,\theta) - p(z,\theta)\right) dz\\
&\leq \left(\|\hat{\eta}_n - \eta\|_{L_2(\Omega)} + \|\hat{p}_N(\cdot,\theta) - p(\cdot,\theta)\|_{L_2(\Omega)}\right)\left(\|\hat{\eta}_n(\cdot)\|_{L_2(\Omega)} + \|\eta(\cdot)\|_{L_2(\Omega)} + \|\hat{p}_N(\cdot,\theta)\|_{L_2(\Omega)} + \|p(\cdot,\theta)\|_{L_2}\right.
\end{aligned}$$

where the inequality follow from the Schwarz inequality and the triangle inequality. Denote the volume of Ω by $Vol(\Omega)$. It can be shown that

$$\|f\|_{L_2(\Omega)} \leq Vol(\Omega)\|f\|_{L_\infty(\Omega)}$$

holds for all $f \in L_\infty(\Omega)$. Thus, we have

$$\begin{aligned}
\|\hat{p}_N(\cdot,\theta) - p(\cdot,\theta)\|_{L_2(\Omega)} &\leq Vol(\Omega)\|\hat{p}_N(\cdot,\theta) - p(\cdot,\theta)\|_{L_\infty(\Omega)}\\
&\leq Vol(\Omega)\|\hat{p}_N - p\|_{L_\infty(\Omega\times\Theta)}, \qquad \text{(B.2)}
\end{aligned}$$

and $\|f(\cdot)\|_{L_2(\Omega)} \leq Vol(\Omega)$ for $f(\cdot) = \hat{\eta}(\cdot), \eta(\cdot), \hat{p}_N(\cdot,\theta)$, and $p(\cdot,\theta)$ because $\|f(\cdot)\|_{L_\infty(\Omega)} \leq 1$. Then, combining (B.2) and assumptions B2 and C1, we have that (B.1) converges to 0 uniformly with respect to $\theta \in \Theta$, which proves the consistency of $\hat{\theta}_n$.

Since $\hat{\theta}_n$ minimizes (2.4), by invoking assumptions A1,A2 and A4, we have

$$\begin{aligned}
0 &= \frac{\partial}{\partial\theta}\|\hat{\eta}_n(\cdot) - \hat{p}_N(\cdot,\hat{\theta}_n)\|^2_{L_2(\Omega)}\\
&= 2\int_\Omega \left(\hat{\eta}_n(z) - \hat{p}_N(z,\hat{\theta}_n)\right)\frac{\partial\hat{p}_n}{\partial\theta}(z,\hat{\theta}_n)dz,
\end{aligned}$$

and by assumption B2, C1 and C2, it implies

$$\int_\Omega \left(\hat{\eta}_n(z) - p(z,\hat{\theta}_n)\right)\frac{\partial p}{\partial\theta}(z,\hat{\theta}_n)dz = o_p(n^{-1/2}). \qquad \text{(B.3)}$$

Let $l(\xi) = \frac{1}{n}\sum_{i=1}^{n}\left(y_i^p \log g(\xi(\mathbf{x}_i)) + (1-y_i^p)\log(1-g(\xi(\mathbf{x}_i)))\right) + \lambda_n \|\xi\|^2_{\mathcal{N}_\Phi(\Omega)}$. From (2.3), we know that $\hat{\xi}_n$ maximizes l over $\mathcal{N}_\Phi(\Omega)$. Since $\hat{\theta}_n \xrightarrow{p} \theta^*$ and by assumption A4, $\frac{\partial p}{\partial \theta}(\cdot, \hat{\theta}_n) \in \mathcal{N}_\Phi(\Omega)$ with sufficiently large n. Define $h(z) = \frac{f(z)}{g(z)(1-g(z))}$ and write $\hat{h}_n = h(\hat{\xi}_n)$. Since $h(z) = 1$ for any $z \in \mathbb{R}$ when g is a logit function, we have

$$
\begin{aligned}
0 &= \frac{\partial}{\partial t} l(\hat{\xi}_n(\cdot) + t\frac{\partial p}{\partial \theta_j}(\cdot, \hat{\theta}_n))|_{t=0} \\
&= -\frac{1}{n}\sum_{i=1}^{n}[g(\hat{\xi}_n(\mathbf{x}_i)) - g(\xi(\mathbf{x}_i))]\hat{h}_n(\mathbf{x}_i)\frac{\partial p}{\partial \theta_j}(\mathbf{x}_i, \hat{\theta}_n) + \frac{1}{n}\sum_{i=1}^{n}(Y_i - g(\xi(\mathbf{x}_i)))\hat{h}_n(\mathbf{x}_i)\frac{\partial p}{\partial \theta_j}(\mathbf{x}_i, \hat{\theta}_n) \\
&\quad + 2\lambda_n < \hat{\xi}_n, \frac{\partial p}{\partial \theta_j}(\mathbf{x}_i, \hat{\theta}_n) >_{\mathcal{N}_\Phi(\Omega)} \\
&= -\frac{1}{n}\sum_{i=1}^{n}[g(\hat{\xi}_n(\mathbf{x}_i)) - g(\xi(\mathbf{x}_i))]\frac{\partial p}{\partial \theta_j}(\mathbf{x}_i, \hat{\theta}_n) + \frac{1}{n}\sum_{i=1}^{n}(Y_i - \eta(\mathbf{x}_i))\frac{\partial p}{\partial \theta_j}(\mathbf{x}_i, \hat{\theta}_n) \\
&\quad + 2\lambda_n < \hat{\xi}_n, \frac{\partial p}{\partial \theta_j}(\mathbf{x}_i, \hat{\theta}_n) >_{\mathcal{N}_\Phi(\Omega)} \\
&:= C_n + D_n + E_n. \qquad \text{(B.4)}
\end{aligned}
$$

We first consider C_n. Let $A_i(f, \theta) = [g(f(\mathbf{x}_i)) - g(\xi(\mathbf{x}_i))]\frac{\partial p}{\partial \theta_j}(\mathbf{x}_i, \theta)$ for $(f, \theta) \in \mathcal{N}_\Phi(\Omega, \rho) \times \Theta$ for some $\rho > 0$. Define the empirical process

$$
E_{1n}(f, \theta) = \frac{1}{\sqrt{n}}\sum_{i=1}^{n}\left\{A_i(f, \theta) - \mathbb{E}[A_i(f, \theta)]\right\},
$$

where $\mathbb{E}[A_i(f, \theta)] = \int_\Omega [g(f(\mathbf{z})) - g(\xi(\mathbf{z}))]\frac{\partial p}{\partial \theta_j}(\mathbf{z}, \theta) d\mathbf{z}$. By assumption B1, $\mathcal{N}_\Phi(\Omega, \rho)$ is Donsker. Thus, by Theorem 2.10.6 in [72], $\mathcal{F}_1 = \{g(f) - g(\xi) : f \in \mathcal{N}_\Phi(\Omega, \rho)\}$ is also Donsker because g is a Lipschitz functions. By assumption A4, the class $\mathcal{F}_2 = \{\frac{\partial p}{\partial \theta_j}(\cdot, \hat{\theta}_n), \theta \in U\}$ is Donsker. Since both $\mathcal{F}_1$ and $\mathcal{F}_2$ are uniformly bounded, by Example 2.10.8 in [72] the product class $\mathcal{F}_1 \times \mathcal{F}_2$ is also Donsker. Thus, the asymptotic equicontinuity property holds, which implies that for any $\epsilon > 0$ there exists a $\delta > 0$ such that

$$
\limsup_{n\to\infty} Pr\left(\sup_{\zeta \in \mathcal{F}_1 \times \mathcal{F}_2, \|\zeta\| \le \delta}\left|\frac{1}{\sqrt{n}}\sum_{i=1}^{n}(\zeta(\mathbf{x}_i) - \mathbb{E}(\zeta(\mathbf{x}_i)))\right| > \epsilon\right) < \epsilon,
$$

where $\|\zeta\|^2 := \mathbb{E}[\zeta(\mathbf{x}_i)^2]$. See Theorem 2.4 of [155]. This implies that for any $\epsilon > 0$ there exists a $\delta > 0$ such that

$$\limsup_{n\to\infty} Pr\left(\sup_{f\in\mathcal{N}_\Phi(\Omega,\rho),\theta\in U,\|g(f)-g(\xi)\|_{L_2(\Omega)}\leq\delta} |E_{1n}(f,\theta)| > \epsilon\right) < \epsilon. \tag{B.5}$$

Suppose $\varepsilon > 0$ is a fixed value. Assumption B3 implies that there exists $\rho_0 > 0$ such that $Pr(\|\hat{\xi}\|_{\mathcal{N}_\Phi} > \rho_0) \leq \varepsilon/3$. In addition, choose δ_0 to be a possible value of δ which satisfies (B.5) with $\epsilon = \varepsilon/3$ and $\rho = \rho_0$. Assumption B2 implies that $Pr(\|g(\hat{\xi}_n) - g(\xi)\|_{L_2(\Omega)} > \delta_0) < \varepsilon/3$. Define

$$\hat{\xi}_n^\circ = \begin{cases} \hat{\xi}_n, & \text{if } \|\hat{\xi}_n\|_{\mathcal{N}_\Phi(\Omega)} \leq \rho_0 \text{ and } \|g(\hat{\xi}_n) - g(\xi)\|_{L_2(\Omega)} \leq \delta_0, \\ \xi, & \text{otherwise.} \end{cases}$$

Then, for sufficiently large n, we have

$$\begin{aligned} Pr(|E_{1n}(\hat{\xi}_n,\hat{\theta}_n)| > \varepsilon) &\leq Pr(|E_{1n}(\hat{\xi}_n^\circ,\hat{\theta}_n)| > \varepsilon) + Pr(\|\hat{\xi}_n\|_{\mathcal{N}_\Phi(\Omega)} > \rho_0) + Pr(\|g(\hat{\xi}_n) - g(\xi)\|_{L_2(\Omega)} > \delta_0) \\ &\leq Pr(|E_{1n}(\hat{\xi}_n^\circ,\hat{\theta}_n)| > \varepsilon/3) + \varepsilon/3 + \varepsilon/3 \\ &\leq Pr\left(\sup_{f\in\mathcal{N}_\Phi(\Omega,\rho),\theta\in U,\|g(f)-g(\xi)\|_{L_2(\Omega)}\leq\delta} |E_{1n}(f,\theta)| > \varepsilon/3\right) + \varepsilon/3 + \varepsilon/3 \\ &\leq \varepsilon. \end{aligned}$$

The first and third inequalities follow from the definition of $\hat{\xi}_n^\circ$, and the last inequality follows from (B.5). Thus, this implies that $E_{1n}(\hat{\xi}_n,\theta)$ tends to zero in probability, which gives

$$\begin{aligned} o_p(1) &= E_{1n}(\hat{\xi}_n,\hat{\theta}_n) \\ &= \frac{1}{\sqrt{n}}\sum_{i=1}^n \left\{[g(\hat{\xi}_n(\mathbf{x}_i)) - g(\xi(\mathbf{x}_i))]\frac{\partial p}{\partial\theta_j}(\mathbf{x}_i,\hat{\theta}_n)\right\} - \frac{1}{\sqrt{n}}\int_\Omega [g(\hat{\xi}_n(\mathbf{z})) - g(\xi(\mathbf{z}))]\frac{\partial p}{\partial\theta_j}(\mathbf{z},\hat{\theta}_n)d\mathbf{z} \\ &= -\sqrt{n}C_n - \sqrt{n}\int_\Omega [g(\hat{\xi}_n(\mathbf{z})) - g(\xi(\mathbf{z}))]\frac{\partial p}{\partial\theta_j}(\mathbf{z},\hat{\theta}_n)d\mathbf{z}, \end{aligned}$$

which implies

$$\begin{aligned} C_n &= -\int_\Omega [g(\hat{\xi}_n(\mathbf{z})) - g(\xi(\mathbf{z}))] \frac{\partial p}{\partial \theta_j}(\mathbf{z}, \hat{\theta}_n) d\mathbf{z} + o_p(n^{-1/2}) \\ &= -\int_\Omega [\hat{\eta}_n(\mathbf{z}) - \eta(\mathbf{z})] \frac{\partial p}{\partial \theta_j}(\mathbf{z}, \hat{\theta}_n) d\mathbf{z} + o_p(n^{-1/2}). \end{aligned} \tag{B.6}$$

Then, by substituting (B.3) to (B.6) and using assumption A2, Taylor expansion can be applied to (B.6) at θ^*, which leads to

$$\begin{aligned} C_n &= -\int_\Omega [p(\mathbf{z}, \hat{\theta}_n) - \eta(\mathbf{z})] \frac{\partial p}{\partial \theta_j}(\mathbf{z}, \hat{\theta}_n) dz + o_p(n^{-1/2}) \\ &= -\left(\frac{1}{2} \int_\Omega \frac{\partial^2}{\partial \theta_i \partial \theta_j} [p(\mathbf{z}, \tilde{\theta}_n) - \eta(\mathbf{z})]^2 d\mathbf{z} \right) (\hat{\theta}_n - \theta^*) + o_p(n^{-1/2}), \end{aligned}$$

where $\tilde{\theta}_n$ lies between $\hat{\theta}_n$ and θ^*. By the consistency of $\hat{\theta}_n$, we then have $\tilde{\theta}_n \xrightarrow{p} \theta^*$. This implies that

$$\int_\Omega \frac{\partial^2}{\partial \theta \partial \theta^T} [p(\mathbf{z}, \tilde{\theta}_n) - \eta(\mathbf{z})]^2 d\mathbf{z} \xrightarrow{p} \int_\Omega \frac{\partial^2}{\partial \theta \partial \theta^T} [p(\mathbf{z}, \theta^*) - \eta(\mathbf{z})]^2 d\mathbf{z} = V,$$

so that

$$C_n = -\frac{1}{2} V(\hat{\theta}_n - \theta^*) + o_p(n^{-1/2}). \tag{B.7}$$

Next, we consider D_n. Define the empirical process

$$\begin{aligned} E_{2n}(\theta) &= \frac{1}{\sqrt{n}} \sum_{i=1}^n \left\{ e_i \frac{\partial p}{\partial \theta_j}(\mathbf{x}_i, \theta) - e_i \frac{\partial p}{\partial \theta_j}(\mathbf{x}_i, \theta^*) - \mathbb{E}\left[e_i \frac{\partial p}{\partial \theta_j}(\mathbf{x}_i, \theta) - e_i \frac{\partial p}{\partial \theta_j}(\mathbf{x}_i, \theta^*) \right] \right\} \\ &= \frac{1}{\sqrt{n}} \sum_{i=1}^n \left\{ e_i \frac{\partial p}{\partial \theta_j}(\mathbf{x}_i, \theta) - e_i \frac{\partial p}{\partial \theta_j}(\mathbf{x}_i, \theta^*) \right\}, \end{aligned}$$

where $\theta \in U$. Assumption A1 implies that the set $\{\zeta_\theta \in C(\mathbb{R} \times \Omega) : \zeta_\theta(e, \mathbf{x}) = e \frac{\partial p}{\partial \theta_j}(\mathbf{x}, \theta) - e \frac{\partial p}{\partial \theta_j}(\mathbf{x}, \theta^*), \theta \in U\}$ is a Donsker class, which ensures that $E_{2n}(\cdot)$ converges weakly in $L_\infty(U)$ to a tight Gaussian process, denoted by $G(\cdot)$. Without loss of generally, we assume $G(\cdot)$ has continuous sample paths. Then, by the continuous mapping theorem [156] and

the consistency of $\hat{\theta}_n$, we have $E_{2n}(\theta) \xrightarrow{p} G(\theta^*)$. Because $E_{2n}(\theta^*) = 0$ for all n, we have $G(\theta^*) = 0$. Then, we have $E_{2n}(\theta) \xrightarrow{p} 0$, which gives

$$D_n = \frac{1}{n}\sum_{i=1}^{n}(Y_i^p - \eta(\mathbf{x}_i))\frac{\partial p}{\partial \theta_j}(\mathbf{x}_i, \theta^*) + o_p(n^{-1/2}). \tag{B.8}$$

Lastly, we consider E_n. Applying assumption A4, B3, B4, we have

$$E_n \leq 2\lambda_n \|\hat{\xi}_n\|_{\mathcal{N}_\Phi(\Omega)} \left\| \frac{\partial p}{\partial \theta_j}(\cdot, \hat{\theta}_n) \right\|_{\mathcal{N}_\Phi(\Omega)} = o_p(n^{-1/2}). \tag{B.9}$$

By combining (B.4), (B.7), (B.8) and (B.9), we have

$$\hat{\theta}_n - \theta^* = 2V^{-1}\left\{ \frac{1}{n}\sum_{i=1}^{n}(Y_i^p - \eta(\mathbf{x}_i))\frac{\partial p}{\partial \theta}(\mathbf{x}_i, \theta^*) \right\} + o_p(n^{-1/2}).$$

□

B.3 Proof of Theorem 2.3.5

Proof. If suffices to show that $\hat{\theta}_n$ given in (2.4) has the same asymptotic variance as the estimator obtained by using maximum likelihood (ML) method. Consider the following q-dimensional parametric model indexed by γ,

$$\xi_\gamma(\cdot) = \xi(\cdot) + \gamma^T \frac{\partial p}{\partial \theta}(\cdot, \theta^*), \tag{B.10}$$

with $\gamma \in \mathbb{R}^q$. By combining (2.1) and (B.10), it becomes a traditional logistic regression model with coefficient γ. Regarding the model (2.1), the true value of γ is 0. Hence, under the regularity conditions of Theorem 2.3.3, the ML estimator has the asymptotic expression

$$\hat{\gamma}_n = \frac{1}{n}W^{-1}\sum_{i=1}^{n}(Y_i - \eta(\mathbf{x}_i))\frac{\partial p}{\partial \theta_j}(\mathbf{x}_i, \theta^*) + o_p(n^{-1/2}), \tag{B.11}$$

where W is defined in (2.8). Then a natural estimator for θ^* in (2.2) is

$$\hat{\theta}_n^{\mathrm{ML}} = \arg\min_{\theta\in\Theta} \|\xi_{\hat{\gamma}_n}(\cdot) - p(\cdot,\theta^*)\|_{L_2(\Omega)}. \tag{B.12}$$

Since the ML estimators (B.11) and (B.12) have the same expression as (3.22) and (3.23) in [23], it follows that

$$\hat{\theta}_n^{\mathrm{ML}} - \theta^* = 2V^{-1}\left(\frac{1}{n}\sum_{i=1}^{n}(Y_i - \eta(\mathbf{x}_i))\frac{\partial p}{\partial\theta}(\mathbf{x}_i,\theta^*)\right) + o_p(n^{-1/2}). \tag{B.13}$$

Therefore, since the asymptotic expression of the ML estimator in (B.13) has the same form as the asymptotic expression of the L_2 calibration given by (2.7), the L_2 calibration (2.4) is semiparametric efficient. □

APPENDIX C

APPENDICES OF CHAPTER 3

C.1 Proof of Proposition 3.3.1

In the variance definition (4), the variance of $Y(x)$ at stage $j+1$ is

$$V_{j+1}(x) = \sigma^2\{\Phi_\Theta(x,x) - \Phi_\Theta(x, X_{j+1})\Phi_\Theta(X_{j+1}, X_{j+1})^{-1}\Phi(X_{j+1}, x)\}. \tag{C.1}$$

Since X_{j+1} is comprised of X_j and x_{j+1}, (C.1) can be rewritten as

$$V_{j+1}(x) = \sigma^2\Big\{\Phi_\Theta(x,x) - \begin{bmatrix}\Phi_\Theta(x, x_{j+1}) & \Phi_\Theta(x, X_j)\end{bmatrix}\begin{bmatrix}\Phi_\Theta(x_{j+1}, x_{j+1}) & \Phi_\Theta(x_{j+1}, X_j)\\ \Phi_\Theta(X_j, x_{j+1}) & \Phi_\Theta(X_j, X_j)\end{bmatrix}^{-1}\begin{bmatrix}\Phi_\Theta(x, x_{j+1})\\ \Phi_\Theta(X_j, x)\end{bmatrix}\Big\}. \tag{C.2}$$

For simplicity, the second term of (C.2) can be written as a partitioned matrix, that is,

$$\begin{bmatrix}a_1^T & a_2^T\end{bmatrix}\begin{bmatrix}B_{11} & B_{12}\\ B_{21} & B_{22}\end{bmatrix}^{-1}\begin{bmatrix}a_1\\ a_2\end{bmatrix}, \tag{C.3}$$

where

$$a_1 = \Phi_\Theta(x, x_{j+1}), a_2 = \Phi_\Theta(X_j, x),$$

$$B_{11} = \Phi_\Theta(x_{j+1}, x_{j+1}), B_{12} = \Phi_\Theta(x_{j+1}, X_j) = B_{12}^T \text{ and } B_{22} = \Phi_\Theta(X_j, X_j).$$

Applying partitioned matrix inverse results [86] and simplifying (C.3) gives

$$a_2^T B_{22}^{-1} a_2 + (a_1 - B_{12}B_{22}^{-1}a_2)^T B_{11\cdot 2}^{-1}(a_1 - B_{12}B_{22}^{-1}a_2), \tag{C.4}$$

where $B_{11\cdot 2} = B_{11} - B_{12}B_{22}^{-1}B_{21}$.

Then, taking (C.4) into (C.2) leads to

$$\begin{aligned}
V(X_{j+1}) &= \sigma^2\{\Phi_\Theta(x,x) - a_2^T B_{22}^{-1} a_2 - (a_1 - B_{12}B_{22}^{-1}a_2)^T B_{11\cdot 2}^{-1}(a_1 - B_{12}B_{22}^{-1}a_2)\} \\
&= \sigma^2\{\Phi_\Theta(x,x) - \Phi_\Theta(x,X_j)\Phi_\Theta(X_j,X_j)^{-1}\Phi_\Theta(X_j,x) \\
&\quad - (a_1 - B_{12}B_{22}^{-1}a_2)^T B_{11\cdot 2}^{-1}(a_1 - B_{12}B_{22}^{-1}a_2)\} \\
&= V(X_j) - \sigma^2\{(a_1 - B_{12}B_{22}^{-1}a_2)^T B_{11\cdot 2}^{-1}(a_1 - B_{12}B_{22}^{-1}a_2)\} \\
&= V(X_j) - \sigma^2 R(x_{j+1}),
\end{aligned}$$

where

$$\begin{aligned}
R(x_{j+1}) &= (a_1 - B_{12}B_{22}^{-1}a_2)^T B_{11\cdot 2}^{-1}(a_1 - B_{12}B_{22}^{-1}a_2) \\
&= (a_1 - B_{12}B_{22}^{-1}a_2)^2 / B_{11\cdot 2} \\
&= \frac{(\Phi_\Theta(x,x_{j+1}) - \Phi_\Theta(x_{j+1},X_j)\Phi_\Theta(X_j,X_j)^{-1}\Phi_\Theta(X_j,x))^2}{\Phi_\Theta(x_{j+1},x_{j+1}) - \Phi_\Theta(x_{j+1},X_j)\Phi_\Theta(X_j,X_j)^{-1}\Phi_\Theta(X_j,x_{j+1})},
\end{aligned}$$

and the second equality holds since $B_{11\cdot 2}$ is a scalar.

C.2 Proof of Theorem 3.3.2

Since $(a-b)^2 \le (a+b)^2$ for $a, b \ge 0$, equation (8) can be bounded as

$$\begin{aligned}
R(x_{j+1}) &= \frac{(\Phi_\Theta(x,x_{j+1}) - \Phi_\Theta(x_{j+1},X_j)\Phi_\Theta(X_j,X_j)^{-1}\Phi_\Theta(X_j,x))^2}{\Phi_\Theta(x_{j+1},x_{j+1}) - \Phi_\Theta(x_{j+1},X_j)\Phi_\Theta(X_j,X_j)^{-1}\Phi_\Theta(X_j,x_{j+1})} \\
&\le \frac{(\Phi_\Theta(x,x_{n+1}) + \Phi_\Theta(x_{j+1},X_j)\Phi_\Theta(X_j,X_j)^{-1}\Phi_\Theta(X_j,x))^2}{\Phi_\Theta(x_{j+1},x_{j+1}) - \Phi_\Theta(x_{j+1},X_j)\Phi_\Theta(X_j,X_j)^{-1}\Phi_\Theta(X_j,x_{j+1})}.
\end{aligned}$$

Also, since

$$a^T B^{-1} b \le \|a\|_2 \|B^{-1} b\|_2$$

and

$$a^T B^{-1} a \leq \|a\|_2^2 \lambda_{\max}(B^{-1}) = \|a\|_2^2 / \lambda_{\min}(B),$$

where $\lambda_{\max}(\cdot)$ and $\lambda_{\min}(\cdot)$ denote the maximum and minimum eigenvalues of a specific matrix, respectively, the inequality becomes

$$R(x_{j+1}) \leq \frac{(\Phi_\Theta(x, x_{j+1}) + \|\Phi_\Theta(x_{j+1}, X_j)\|_2 \|\Phi_\Theta(X_j, X_j)^{-1} \Phi_\Theta(X_j, x)\|_2)^2}{1 - \|\Phi_\Theta(X_j, x_{j+1})\|_2^2 / \lambda_{\min}},$$

where $\lambda_{\min}$ is the minimum eigenvalue of $\Phi_\Theta(X_j, X_j)$.

Furthermore, according to the definition $d_{\min}(x_{j+1})$ of the minimum (Mahalanobis-like) distance as (6) and the definition $\phi(\cdot)$ as in Theorem 1, we have

$$\Phi_\Theta(u, x_{j+1}) \leq \phi(d_{\min}(x_{j+1})), \text{ for any } u \in \{x, X_j\},$$

which also implies

$$\|\Phi_\Theta(x_{j+1}, X_j)\|_2 = \|\Phi_\Theta(X_j, x_{j+1})\|_2 \leq \sqrt{j}\phi(d_{\min}(x_{j+1})),$$

therefore the inequality can be bounded as

$$R(x_{j+1}) \leq \frac{(\phi(d_{\min}(x_{j+1})) + \sqrt{j}\phi(d_{\min}(x_{j+1}))\|\Phi_\Theta(X_j, X_j)^{-1} \Phi_\Theta(X_j, x)\|_2)^2}{1 - j\phi^2(d_{\min}(x_{j+1}))/\lambda_{\min}}. \quad \text{(C.1)}$$

Thus, for $\delta > 0$, if

$$\frac{(\phi(d_{\min}(x_{j+1})) + \sqrt{j}\phi(d_{\min}(x_{j+1}))\|\Phi_\Theta(X_j, X_j)^{-1} \Phi_\Theta(X_j, x)\|_2)^2}{1 - j\phi^2(d_{\min}(x_{j+1}))/\lambda_{\min}} \leq \delta$$

or equivalently

$$d_{\min}(x_{j+1}) \geq \phi^{-1}\left(\sqrt{\frac{\delta}{(1+\sqrt{j}\|\Phi_\Theta(X_j,X_j)^{-1}\Phi_\Theta(X_j,x)\|_2)^2 + j\delta/\lambda_{\min}}}\right),$$

then by (C.1), $R(x_{j+1}) \leq \delta$.

C.3 Proof of Theorem 3.3.3

Define $U(t) = (\sqrt{\lambda_1}\varphi_1(t), \sqrt{\lambda_2}\varphi_2(t), \ldots, \sqrt{\lambda_D}\varphi_D(t))^T \in \mathbb{R}^{D\times 1}$, where $\varphi_i(\cdot), i = 1, \ldots, D$ is an orthonormal basis of $L^2(\Omega)$ consisting of the eigenfunctions of T, defined in (13), and $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_D$ are corresponding eigenvalues. According to (14), the approximated eigen-decomposition can be rewritten as

$$\Phi(x,y) \approx U^T(x)U(y).$$

Also, define a matrix $U(K) = [U(k_1), U(k_2), \ldots, U(k_n)] \in \mathbb{R}^{D\times n}$ for $K = (k_1, k_2, \ldots, k_n)$. Then, the reduction in variance $R(x_{j+1})$ in (8) can be approximated to the following:

$$\begin{aligned}
R(x_{j+1}) &= \frac{(\Phi_\Theta(x,x_{j+1}) - \Phi_\Theta(x_{j+1},X_j)\Phi_\Theta(X_j,X_j)^{-1}\Phi_\Theta(X_j,x))^2}{\Phi_\Theta(x_{j+1},x_{j+1}) - \Phi_\Theta(x_{j+1},X_j)\Phi_\Theta(X_j,X_j)^{-1}\Phi_\Theta(X_j,x_{j+1})} \\
&\approx \frac{(U^T(x_{j+1})U(x) - U^T(x_{j+1})U(X_j)[U^T(X_j)U(X_j)]^-U^T(X_j)U(x))^2}{U^T(x_{j+1})U(x_{j+1}) - U^T(x_{j+1})U(X_j)[U^T(X_j)U(X_j)]^-U^T(X_j)U(x_{j+1})} \\
&= \frac{(U^T(x_{j+1})[I - U(X_j)[U^T(X_j)U(X_j)]^-U^T(X_j)]U(x))^2}{U^T(x_{j+1})[I - U(X_j)[U^T(X_j)U(X_j)]^-U^T(X_j)]U(x_{j+1})},
\end{aligned}$$

where $[U^T(X_j)U(X_j)]^-$ denotes a generalized inverse of $[U^T(X_j)U(X_j)]$.

Let $C_{X_j}(t) = [I - U(X_j)[U^T(X_j)U(X_j)]^-U^T(X_j)]U(t)$. Then,

$$C_{X_j}^T(x_{j+1})C_{X_j}(x) = U^T(x_{j+1})[I - U(X_j)[U^T(X_j)U(X_j)]^-U^T(X_j)]U(x).$$

Similarly,

$$C_{X_j}^T(x_{j+1})C_{X_j}(x_{j+1}) = U^T(x_{j+1})[I - U(X_j)[U^T(X_j)U(X_j)]^- U^T(X_j)]U(x_{j+1}).$$

Therefore,

$$R(x_{n+1}) \approx \frac{(C_{X_j}^T(x_{j+1})C_{X_j}(x))^2}{C_{X_j}^T(x_{j+1})C_{X_j}(x_{j+1})} = \|C_{X_j}(x)\|_2^2 \cos^2(\vartheta),$$

where ϑ is the angle between $C_{X_j}(x)$ and $C_{X_j}(x_{j+1})$.

APPENDIX D

APPENDICES OF CHAPTER 4

D.1 Proof of Theorem 4.2.1

First, a useful lemma is given.

Lemma D.1.1. *Denote* $\mathcal{F}_u = \{\int_{\Omega_{-u}} \left(f(x) - \sum_{v \subset u} f_v(x)\right) \mathrm{d}x_{-u} | f \in \mathcal{N}_\Phi, f_v \in \mathcal{F}_v\}$. *Suppose* $\Phi \in \Omega \times \Omega \to \mathbb{R}$ *is a symmetric positive-definite kernel on* $\Omega = [0,1]^d$ *and* Φ *is a product kernel. Then,*

$$f_u \in \mathcal{F}_u = \{f_v + g_u | g_u \in \mathcal{N}_{\Phi_u}, v \subset u, f_v \in \mathcal{F}_v\},$$

where $\Phi_u = \prod_{j \in u} \phi_j$.

Proof. Initially consider a finite element. The proof proceeds by induction. For $u = \emptyset$, we have that if $f \in \mathcal{N}_\Phi$, then

$$f_\emptyset = \int_\Omega f(x)\mathrm{d}x = \int_\Omega \sum_{y \in X} \beta_y \Phi(x,y)\mathrm{d}x = \sum_{y \in X} \beta_y \int_\Omega \Phi(x,y)\mathrm{d}x := \alpha \in \mathbb{R}.$$

This shows $f_\emptyset \in \mathcal{F}_\emptyset = \{f(\cdot) = \alpha | \alpha \in \mathbb{R}\}$.

Let $f_u \in \mathcal{F}_u$ for any $|u| \leq k$. Note that $\int_{\Omega_{-u}} \mathrm{d}x_{-u} = 1$ for any u, since $\Omega = [0,1]^d$.

Thus, for $|u'| = k+1$,

$$\begin{aligned} f_{u'}(x) &= \int_{\Omega_{-u'}} \left(f(x) - \sum_{v \subset u'} f_v(x) \right) \mathrm{d}x_{-u'} = \int_{\Omega_{-u'}} f(x) \mathrm{d}x_{-u'} - \sum_{v \subset u'} f_v(x) \\ &= \sum_{y \in X} \beta_y \int_{\Omega_{-u'}} \Phi(x, y) \mathrm{d}x_{-u'} - \sum_{v \subset u'} f_v(x) \\ &= \sum_{y \in X} \beta_y \int_{\Omega_{-u'}} \prod_{j=1}^{d} \phi(x_j, y_j) \mathrm{d}x_{-u'} - \sum_{v \subset u'} f_v(x) \\ &= \sum_{y \in X} \beta_y \prod_{j \in u'} \phi_j(x_j, y_j) \int_{\Omega_{-u'}} \prod_{j \notin u'} \phi_j(x_j, y_j) \mathrm{d}x_{-u'} - \sum_{v \subset u'} f_v(x) \\ &= \sum_{y \in X} \tilde{\beta}_y \prod_{j \in u'} \phi_i(x_i, y_i) - \sum_{v \subset u'} f_v(x), \end{aligned}$$

where $\tilde{\beta}_y = \beta_y \int_{\Omega_{-u'}} \prod_{j \notin u'} \phi_j(x_j, y_j) \mathrm{d}x_{-u'}$. Hence, since $\sum_{y \in X} \tilde{\beta}_y \phi_{u'}(\cdot, y_i) \in \mathcal{N}_{\Phi_{u'}}$ and $f_v \in \mathcal{F}_v$ for any $|v| \leq k$, we have $f_{u'} \in \mathcal{F}_{u'} = \{f = f_v + g_{u'} | g_{u'} \in \mathcal{N}_{\Phi_{u'}}, v \subset u', f_v \in \mathcal{F}_v\}$. Therefore, by induction, $f_u \in \mathcal{F}_u = \{f_v + g_u | g_u \in \mathcal{N}_{\Phi_u}, v \subset u, f_v \in \mathcal{F}_v\}$ is true for any $u \subseteq D$.

Since any element of an RKHS is bounded [157], we may use the dominated convergence theorem [158] to interchange the integral and the limit of the finite sums to extend to an arbitrary element. □

By Lemma D.1.1, we have $f(x) = \sum_{u \subseteq D} f_u(x)$, where $f_u(x) \in \mathcal{F}_u = \{f_v + g_u | g_u \in \mathcal{N}_{\Phi_u}, v \subset u, f_v \in \mathcal{F}_v\}$. Thus, by the fact that $g_u^{(1)} + g_u^{(2)} \in \mathcal{N}_{\Phi_u}$ for $g_u^{(1)}, g_u^{(2)} \in \mathcal{N}_{\Phi_u}$, $f(x)$ can be represented as $f(x) = \sum_{u \subseteq D} f_u(x)$, where $f_u \in \mathcal{N}_{\Phi_u}$.

D.2 Algorithm for Estimation

1. Let $\mathcal{A}$ denote the set of active groups and $\mathcal{C}$ the set of candidate groups. Start with $\mathcal{A} = \emptyset$ and $\mathcal{C} = \{(u, r) | u = \{1\}, \ldots, \{d\}, r = 1\}$. Set an initial penalty $\lambda_{\max}$ and a small increment Δ.

2. Set up an overlapping group lasso algorithm which minimizes the penalized likelihood

function

$$\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \sum_{(u,r)\in\mathcal{C}}\sum_{k=1}^{n_u(r)}\beta_u^{rk}\varphi_u^{rk}(x_{iu})\right)^2 + \lambda\sum_{(u,r)\in\mathcal{C}}\sqrt{N_u(r)\sum_{v\subseteq u}\sum_{s\leq r}\sum_{k=1}^{n_v(s)}(\beta_v^{sk})^2}.$$

Denote the input-output function as $\hat{\beta}_\lambda = \texttt{grplasso}(\lambda, \mathcal{C}, \hat{\beta}_{\lambda+\Delta})$. The inputs include a penalty value λ, the candidate set $\mathcal{C}$ and the estimated coefficient with penalty value $\lambda + \Delta$, and the output $\hat{\beta}_\lambda$ is the corresponding estimated coefficient by the algorithm. Start with $\lambda = \lambda_{\max}$ and $\hat{\beta}_{\lambda+\Delta} = 0$.

3. Do $\hat{\beta}_\lambda = \texttt{grplasso}(\lambda, \mathcal{C}, \hat{\beta}_{\lambda+\Delta})$ and obtain the set of active groups $\mathcal{A}' \subseteq \mathcal{C}$ based on $\hat{\beta}_\lambda$. Set $\lambda = \lambda - \Delta$. If $\mathcal{A}' \setminus \mathcal{A} \neq \emptyset$, then $\mathcal{A} \leftarrow \mathcal{A}'$ and $\mathcal{C} \leftarrow \mathcal{C} \cup \mathcal{C}'$, where $\mathcal{C}'$ contains the new candidate groups necessary to satisfy strong effects heredity given the updated $\mathcal{A}$.

4. Repeat step 3 until some convergence criterion is met.

D.3 Confidence Interval Algorithm

1. Let φ^* denote the basis function evaluations at a particular predictive location x^*. Extend φ^* to a basis of $\mathbb{R}^p$ and denote it as $B = (\varphi^*, c_2, \ldots, c_p)$. Compute $(\tilde{Z}_i, \tilde{Q}_i)^T = B^{-1}\varphi_i$ for $i = 1, \ldots, n$ and $(\hat{\eta}_1, \hat{\eta}_{(-1)}^T) = B^T\hat{\beta}_\lambda$, where $\hat{\beta}_\lambda$ is the estimated coefficient with penalty λ.

2. Compute the estimated decorrelated score function

$$\hat{S}(0, \hat{\eta}_{(-1)}) = -\frac{1}{n\hat{\sigma}^2}\sum_{i=1}^{n}(y_i - \hat{\eta}_{(-1)}^T\tilde{Q}_i)(\tilde{Z}_i - \hat{w}^T\tilde{Q}_i),$$

where

$$\hat{w} = \arg\min\left\|\frac{1}{n}\sum_{i=1}^{n}\tilde{Q}_i(\tilde{Z}_i - w^T\tilde{Q}_i)\right\|_2 + \lambda''\|w\|_1,$$

and $\hat{\sigma}^2$ is a consistent estimator of σ^2. For example, σ^2 can be estimated by $\hat{\sigma}^2 = \frac{1}{n-s}\sum_{i=1}^{n}(y_i - \hat{\beta}_\lambda^T \varphi_i)^2$, where s the the number of non-zero elements in $\hat{\beta}_\lambda$. Another estimator is the cross-validation based variance estimator. Define the K cross-validation folds as $\{D_1, \ldots, D_K\}$ and compute

$$\hat{\sigma}^2 = \min_{\lambda} \frac{1}{n} \sum_{k=1}^{K} \sum_{i \in D_k} (y_i - (\hat{\beta}_\lambda^{(-k)})^T \varphi_i)^2,$$

where $\hat{\beta}_\lambda^{(-k)}$ is the overlapping group lasso estimate at λ over the data after the k^{th} fold is omitted. This estimator has been used for the variance estimation in lasso regression problems. See [159].

3. Compute the interval

$$[c_{\alpha/2}/b, c_{1-\alpha/2}/b],$$

where $c_{\alpha/2} = -\hat{S}(0, \hat{\eta}_{(-1)}) + \sqrt{\frac{b}{n}}\Phi^{-1}(\alpha/2)$, $c_{1-\alpha/2} = -\hat{S}(0, \hat{\eta}_{(-1)}) + \sqrt{\frac{b}{n}}\Phi^{-1}(1 - \alpha/2)$, $b = \frac{1}{n\hat{\sigma}^2}\sum_{i=1}^{n} \tilde{Z}_i(\tilde{Z}_i - \hat{w}^T \tilde{Q}_i)$. By some algebraic manipulation, one can show that this interval is same as the one in Corollary 4.4.4.

D.4 Confidence Interval Algorithm Modification for Large N

1. In Algorithm D.3, replace $\tilde{Q}_i$ by $\tilde{Q}_{*i}$ and p by p_*, where the nuisance φ_{ij}, $j = 1, \ldots, p_*$ only contain basis functions in the candidate groups at the selected λ, say $\mathcal{C}_\lambda$.

2. Replace $\hat{w}$ by

$$\hat{w}_* = \left(\sum_{i=1}^{n} \tilde{Q}_{*i}\tilde{Q}_{*i}^T + \eta I_{p_*-1}\right)^{-1} \left(\sum_{i=1}^{n} \tilde{Q}_{*i}\tilde{Z}_i\right) \tag{D.1}$$

with a small positive η, where I_{p_*-1} is a $(p_* - 1) \times (p_* - 1)$ identity matrix.

3. Define K cross-validation folds as $\{D_1, \ldots, D_K\}$ and partition the original samples $\{x_i, y_i\}_{i=1}^n$ via the k folds.

4. Regard $\hat{\sigma}^2$ in Algorithm D.3 as an unknown parameter. Let $\hat{u}^{(-k)}(x^*, \hat{\sigma}^2)$ and $\hat{l}^{(-k)}(x^*, \hat{\sigma}^2)$ be the upper and lower limits at a predictive location x^* by Algorithm D.3 over the data after the k^{th} fold is omitted, respectively.

5. Replace $\hat{\sigma}^2$ by

$$\hat{\sigma}_*^2 = \arg\min_{\hat{\sigma}^2} \left| \left(\frac{1}{n} \sum_{k=1}^{K} \sum_{i \in D_k} \mathbb{1}\{y_i \in [\hat{l}^{(-k)}(x_i, \hat{\sigma}^2), \hat{u}^{(-k)}(x_i, \hat{\sigma}^2)]\} \right) - (1-\alpha) \right|,$$

where $\mathbb{1}\{A\}$ is an indicator function of the set A.

D.5 Proof of Theorem 4.4.1

D.5.1 Notation and Reformulation

First, we introduce some additional notation. For a matrix $M = [M_{jk}]$, let $\|M\|_{\max} = \max_{j,k} |M_{jk}|$, $\|M\|_1 = \sum_{j,k} |M_{jk}|$, and $\|M\|_{l_\infty} = \max_j \sum_k |M_{jk}|$. For $v = (v_1, ..., v_p)^T \in \mathbb{R}^p$, and $1 \leqslant q < \infty$, define $\|v\|_q = (\sum_{i=1}^p |v_i|^q)^{1/q}$. Define $\|v\|_0 = |\{i : v_i \neq 0\}|$. For $S \subseteq \{1, ..., p\}$, let $v_S = \{v_j : j \in S\}$ and $\bar{S}$ be the complement of S. Given $a, d \in \mathbb{R}$, we use $a \vee b$ and $a \wedge b$ to denote the maximum and minimum of a and b.

For convenience, we restate the loss function as follows. Consider groups $J_1, ..., J_{p_n}$, where $J_j \subseteq \{1, ..., p\}$, and $\bigcup_{j=1}^{p_n} J_j = \{1, ..., p\}$. Notice that we do not require $J_{j_1} \bigcap J_{j_2} = \emptyset$. Define $C_k = \{j : k \in J_j\}$ and $c_k = |C_k|$. Thus, C_k is the set of indices of the groups variable k belongs to and c_k is the number of groups that variable k belongs to. We can also treat c_k as replicates of index k. Define the vector of variable k coefficients over all groups in which it appears $\beta_{kC_k}^Z = (\beta_{kj_{k1}}, \ldots, \beta_{kj_{kc_k}})^T$, where j_{kl} denotes the index of variable k within the l^{th} group in which it appears, and the vector of all coefficients $\beta^Z = ((\beta_{1C_1}^Z)^T, \ldots, (\beta_{pC_p}^Z)^T)^T$. Let $\beta_{J_j} = (\beta_{kj})_{k \in J_j}^T$, where β_{kj} is the coefficient of the k^{th}

variable and k is in j^{th} group. Let $d_j = |J_j|$. Consider the following optimization problem

$$\hat{\beta}^{Z,\lambda_n} = \arg\min_{\beta^Z} \left\{ \frac{1}{2n} \sum_{i=1}^{n} (y_i - \sum_{k=1}^{p} \left(\sum_{m=1}^{c_k} \beta_{kj_{km}} \right) \varphi_{ki})^2 + \lambda_n \sum_{j=1}^{p_n} \sqrt{d_j} \|\beta_{J_j}\|_2 \right\}, \quad \text{(D.2)}$$

where λ_n is a positive number. We define the overlapping group lasso estimator as

$$\hat{\beta}^{\lambda_n} = \left(\sum_{k=1}^{c_1} \hat{\beta}^{\lambda_n}_{1j_{1k}}, ..., \sum_{k=1}^{c_p} \hat{\beta}^{\lambda_n}_{pj_{pk}} \right)^T, \quad \text{(D.3)}$$

in which we stress λ_n since it will influence the solution of (D.2). Notice that by this definition, the least squares term becomes $\frac{1}{2n} \sum_{i=1}^{n} (y_i - \hat{\beta}^T \varphi_i)^2$, which is the same as in original group lasso case. We use $\frac{1}{2n}$ instead of $\frac{1}{n}$ for brevity of the Karush-Kuhn-Tucker (KKT) conditions, which are as following.

Proposition D.5.1. Let φ be the matrix with rows φ_i^T, $i = 1, \ldots, n$. Let ψ_j denote the j^{th} column of φ, for $j = 1, \ldots, p$. Necessary and sufficient conditions for $\hat{\beta}^Z$ to be a solution to (D.2) are

$$\begin{aligned} -\frac{1}{n} \psi_j^T (y - \varphi\hat{\beta}) + \frac{\lambda_n \sqrt{d_k} \hat{\beta}_{jk}}{\|\hat{\beta}_{J_k}\|_2} = 0, \qquad & \forall j \in J_k \text{ with } \hat{\beta}_{J_k} \neq 0 \\ \| -\frac{1}{n} \psi_j^T (y - \varphi\hat{\beta}) \|_2 \leqslant \lambda_n \sqrt{d_k}, \qquad & \forall j \in J_k \text{ with } \hat{\beta}_{J_k} = 0. \end{aligned}$$

The following lemma [134] states that at most n groups can be nonzero.

Lemma D.5.2. *Suppose $\lambda_n > 0$, a solution $\hat{\beta}^{Z,\lambda_n}$ exists such that the number of nonzero groups $|S(\hat{\beta}^{Z,\lambda_n})| \leqslant n$, the number of data points, where $S(\beta) = \{J_j : \hat{\beta}_{J_j} \neq 0\}$.*

Proof. The proof of Lemma 1 in [134] is also valid here. □

By Lemma D.5.2, for brevity, sometimes we say $\hat{\beta}^{\lambda_n}$ with $|S(\hat{\beta}^{Z,\lambda_n})| \leqslant n$, which is derived by combining (D.2) and (D.3), is the solution of (D.2). We will also write $\|y - \varphi\beta\|_2^2$ instead of $\sum_{i=1}^{n} \left(y_i - \sum_{k=1}^{p} \left(\sum_{m=1}^{c_k} \beta_{kj_{km}} \right) \varphi_{ki} \right)^2$. Let $\bar{c} = \max_j \{c_1, ..., c_p\}$

and $\bar{d} = \max_j\{d_1, ..., d_{p_n}\}$, the maximum number of groups a variable appears in and maximum group size, respectively. Let s be the number of nonzero elements in β^* and p be the dimension of β^*. Notice that s and p (as well as $\bar{c}$ and $\bar{d}$) can depend n.

D.5.2 Proof of Theorem 4.4.1

Our proof follows a similar line to [133], but extends their results to the overlapping group lasso. A sketch of the proof is as follows. We first define the coefficients obtained from the de-noised model as a de-noised estimator. Then, by showing the difference between the de-noised estimator and true coefficients, and the difference between de-noised estimator and the estimator obtained via overlapping group lasso are both small, we obtain l_2 convergence. All the proofs of the lemmas in this section are in Appendix D.7.

Before we state and prove the main result, we introduce a definition which is useful in the proof.

Definition D.5.1. Denote $y(\xi) = \varphi\beta^* + \xi\epsilon$ as a de-noised model with level ξ $(0 \leqslant \xi \leqslant 1)$, we define

$$\hat{\beta}^{\lambda,\xi} = \arg\min_{\beta} \frac{1}{2n}\|y(\xi) - \varphi\beta\|_2^2 + \lambda_n \sum_{j=1}^{p_n} \sqrt{d_j}\|\beta_{J_j}\|_2 \tag{D.4}$$

to be the de-noised estimator at noise level ξ, where $\hat{\beta}^{\lambda,\xi}$ is defined similarly as in (D.3).

In order to characterize the eigenvalues of a matrix under sparsity, we introduce the following definition, which can be found in [133].

Definition D.5.2. The m-sparse minimum and maximum eigenvalue of a matrix $C = \frac{1}{n}\varphi^T\varphi$ are $\phi_{\min}(m) = \min_{\beta:\|\beta\|_0 \leqslant m} \frac{\beta^T C\beta}{\beta^T\beta}$ and $\phi_{\max}(m) = \max_{\beta:\|\beta\|_0 \leqslant m} \frac{\beta^T C\beta}{\beta^T\beta}$. Also, denote $\phi_{\max} = \phi_{\max}((s\bar{c} + n)\bar{d})$ where s, $\bar{c}$, and $\bar{d}_n$ are defined as in section D.5.1.

Now we introduce an assumption concerning $\phi_{\min}(\cdot)$ and $\phi_{\max}$. Detailed discussion has been shown in [133].

Assumption D.5.1. There exist constants $0 < \kappa_{\min} \leqslant \kappa_{\max} < \infty$ such that $\liminf_{n\to\infty} \phi_{\min}(s\bar{c}\bar{d}\max\{\log n, \bar{c}\}) \geqslant \kappa_{\min}$ and $\limsup_{n\to\infty} \phi_{\max} \leqslant \kappa_{\max}$.

For continuity, we repeat Theorem 4.1 here.

Theorem 4.1. Under Assumption D.5.1, if $\lambda_n \asymp \sigma\sqrt{\frac{\log p}{n}}$ and $\bar{d}^2 = o(\log n)$, for the (overlapping) group lasso estimator constructed in (D.2) and (D.3), with probability tending to 1 for $n \to \infty$,

$$\|\hat{\beta}^{\lambda_n} - \beta^*\|_2^2 \lesssim \frac{\bar{c}^2 s\bar{d}\log p}{n}.$$

Let $\beta^{\lambda_n} = \hat{\beta}^{\lambda_n,0}$. The l_2-consistency can be obtained by bounding the bias and variance terms, i.e.

$$\|\hat{\beta}^{\lambda_n} - \beta^*\|_2^2 \leqslant 2\|\hat{\beta}^{\lambda_n} - \beta^{\lambda_n}\|_2^2 + 2\|\beta^{\lambda_n} - \beta^*\|_2^2.$$

Let $T = \{t : \beta_i^* \neq 0, \beta_{it}^* \text{ is a component of } \beta^{Z*}\}$ represent the set of indices for all the groups with possibly nonzero coefficient vectors. Let $s_n = |T|$. Thus, $s_n \leqslant s\bar{c}$. The solution β^{λ_n} can, for each value of λ_n, be written as $\beta^{\lambda_n} = \beta^* + \gamma^{\lambda_n}$, where γ^{λ_n} is defined as the solution of the following optimization problem:

$$\begin{aligned} \arg\min_{\gamma} \quad & f(\gamma, \gamma^Z) \\ \text{s.t.} \quad & \sum_{k=1}^{c_i} \beta_{ik}^Z = \beta_i^*, \quad i = 1, ..., p; \\ & \sum_{k=1}^{c_i} \gamma_{ij_{ik}}^Z = \gamma_i, \quad i = 1, ..., p, \end{aligned} \tag{D.5}$$

where

$$f(\gamma, \gamma^Z) = n\gamma^T A\gamma + \lambda_n \sum_{t\in T^c} \sqrt{d_t}\|\gamma_t^Z\|_2 + \lambda_n \sum_{t\in T} \sqrt{d_t}(\|\gamma_t^Z + \beta_t^Z\|_2 - \|\beta_t^Z\|_2),$$

where $A = \frac{1}{n}\varphi^T\varphi$. This optimization problem is obtained by plugging $\beta^* + \gamma^{\lambda_n}$ into (D.4). Notice the $\arg\min$ problem is with respect to γ instead of (γ, γ^Z).

Next, we state a lemma which bounds the l_2-norm of γ^{λ_n}. Its proof is provided in Appendix D.7.1.

Lemma D.5.3. *Under Assumption D.5.1, with a positive constant C, the l_2-norm of γ^{λ_n} is bounded for sufficiently large values of n by* $\|\gamma^{\lambda_n}\|_2 \leqslant \frac{\lambda_n\sqrt{\bar{c}s_n\bar{d}}}{n}\Big/\left(\sqrt{\frac{\kappa_{\min}}{2}(1-\frac{4\bar{d}}{\log n})} - \sqrt{\frac{2\kappa_{\max}\bar{d}^2}{\log n}}\right)$.

Now, we bound the variance term. For every subset $M \subset \{1, ..., p\}$ with $|M| \leqslant n$, denote $\hat{\theta}^M \in \mathbb{R}^{|M|}$ the restricted least square estimator of the noise ϵ,

$$\hat{\theta}^M = (\varphi_M^T\varphi_M)^{-1}\varphi_M^T\epsilon. \tag{D.6}$$

Now we state lemmas, which bound the l_2-norm of this estimator, and are also useful for the following parts of this development. First we define sub-exponential variables, sub-exponential norms, sub-Gaussian variables, and sub-Gaussian norms.

Definition D.5.3. (sub-exponential variable and sub-exponential norm) A random variable X is called sub-exponential if there exists some positive constant K_1 such that $\mathbb{P}(|X| > t) \leqslant \exp(1 - t/K_1)$ for all $t \geqslant 0$. The sub-exponential norm of X is defined as $\|X\|_{\psi_1} = \sup_{q\geqslant 1} q^{-1}(\mathbb{E}|X|^q)^{1/q}$.

Definition D.5.4. (sub-Gaussian variable and sub-Gaussian norm) A random variable X is called sub-Gaussian if there exists some positive constant K_2 such that $\mathbb{P}(|X| > t) \leqslant \exp(1 - t^2/K_2)$ for all $t \geqslant 0$. The sub-Gaussian norm of X is defined as $\|X\|_{\psi_2} = \sup_{q\geqslant 1} q^{-1/2}(\mathbb{E}|X|^q)^{1/q}$.

Lemma D.5.4. *Let $\bar{m}_n$ be a sequence with $\bar{m}_n = o(n)$ and $\bar{m}_n \to \infty$ for $n \to \infty$*

$$\max_{M:|M|\leqslant\bar{m}_n} \|\theta^M\|_2^2 \leqslant C^2\frac{\bar{m}_n\log p}{n\phi^2_{\min}(\bar{d})}.$$

Proof. See Appendix D.7.2. □

Now define $A_{\lambda_n,\xi}$ to be

$$A_{\lambda_n,\xi} = \left\{ k : \lambda_n \frac{\sqrt{d_k}\hat{\beta}_{jk}}{\|\hat{\beta}_{J_k}\|_2} = \frac{1}{n}\psi_j^T(Y(\xi) - \varphi\hat{\beta}), \text{ with } j \in J_k \right\},$$

which represents the set of active groups for the de-noised problem.

Lemma D.5.5. *If, for a fixed value of λ_n, the number of active variables of the de-noised estimators $\hat{\beta}^{\lambda_n,\xi}$ is for every $0 \leqslant \xi \leqslant 1$ bounded by m', then*

$$\|\hat{\beta}^{\lambda_n,0} - \hat{\beta}_n^{\lambda}\|_2^2 \leqslant C \max_{M:|M|\leqslant m'} \|\theta^M\|_2^2.$$

Proof. See Appendix D.7.3. □

The next lemma provides an asymptotic upper bound on the number of selected variables.

Lemma D.5.6. *For $\lambda_n \geqslant \sqrt{\frac{\log p}{n}}$, the maximal number of selected variables, $\sup_{0\leqslant\xi\leqslant 1} \sum_{k\in A_{\lambda,\xi}} d_k$, is bounded, with probability tending to 1 for $n \to \infty$, by*

$$\sup_{0\leqslant\xi\leqslant 1} \sum_{k\in A_{\lambda,\xi}} d_k \leqslant C_1 s_n \bar{d}\bar{c}.$$

Proof. See Appendix D.7.4. □

Now combining Lemmas D.5.4, D.5.5, and D.5.6, we have

$$\|\hat{\beta}^{\lambda_n,0} - \hat{\beta}^{\lambda_n}\|_2^2 \leqslant C \frac{s\bar{d}\bar{c}^2 \log p}{n\phi_{\min}^2(s\bar{d}\bar{c}^2)}.$$

Combining this and Lemma D.5.3, gives

$$\begin{aligned}\|\hat{\beta}^{\lambda_n}-\beta\|_2^2 &\leqslant C\frac{s\bar{d}\bar{c}^2\log p}{n\phi_{\min}^2(s\bar{d}\bar{c}^2)}+\frac{\lambda_n^2\bar{c}^2 s\bar{d}}{n^2}\Bigg/\left(\sqrt{\frac{\kappa_{\min}}{2}(1-\frac{4\bar{d}}{\log n})}-\sqrt{\frac{2\kappa_{\max}\bar{d}^2}{\log n}}\right)^2\\ &\leqslant C\frac{s\bar{d}\bar{c}^2\log p}{n}+C\frac{\bar{c}^2 s\bar{d}\log p}{n}\Bigg/\left(\sqrt{\frac{\kappa_{\min}}{2}(1-\frac{4\bar{d}}{\log n})}-\sqrt{\frac{2\kappa_{\max}\bar{d}^2}{\log n}}\right)^2\\ &\lesssim \frac{\bar{c}^2 s\bar{d}\log p}{n},\end{aligned}$$

which completes the proof of Theorem 4.4.1.

D.6 Proof of Theorem 4.4.3

In this section we will prove Theorem 4.4.3. A sketch of proof is as follows, following the overall approach in [135]. First, we introduce a decorrelated score function, and prove the decorrelated function converges weakly to a normal distribution under l_2-consistency, which is stated in Theorem D.6.1. The result is then applied to the overlapping group lasso model with known variance of error. Then by showing the difference between the decorrelated score function with known variance and decorrelated score function with estimated variance is small, we finish the proof of Theorem 4.4.3.

D.6.1 Hypothesis Test based on Decorrelated Function and l_2-Consistency

In this section, we will introduce a decorrelated score function, and prove several results similar to [135] but with l_2-consistency instead of l_1. Suppose we are given n independently identically distributed $U_1, ..., U_n$, which come from the same probability distribution following from a high dimensional statistical model $\mathcal{P} = \{\mathbb{P}_\beta : \beta \in \Omega\}$, where β is a p dimensional unknown parameter and Ω is the parameter space. Let the true value of β be β^*, which is sparse in the sense that the number of non-zero elements of β is much smaller than n, order $\log n$. We consider the case in which we are interested in only one parameter. Suppose $\beta = (\theta, \gamma)$, where $\theta \in \mathbb{R}$ and $\gamma \in \mathbb{R}^{p-1}$. Let θ^* and γ^* be the true value of θ

and γ, respectively. For simplicity, we assume the null hypothesis is $H_0 : \theta^* = 0$, which can be generalized to the case $\theta^* = \theta_0$ in a straight forward manner. Suppose the negative log-likelihood function is

$$\ell(\theta, \gamma) = \frac{1}{n} \sum_{i=1}^{n} (-\log f(U_i; \theta, \gamma)),$$

where f is the p.d.f. corresponding to the model $\mathbb{P}_\beta$, which it will be assumed has at least two continuous derivatives with respect to β. The information matrix for β is defined as $I = \mathbb{E}_\beta(\nabla^2 \ell(\beta))$, and the partial information matrix is $I_{\theta|\gamma} = I_{\theta\theta} - I_{\theta\gamma} I_{\gamma\gamma}^{-1} I_{\gamma\theta}$, where $I_{\theta\theta}$, $I_{\theta\gamma}$, $I_{\gamma\gamma}$, and $I_{\gamma\theta}$ are the corresponding partitions of I. Let $I^* = \mathbb{E}_{\beta^*}(\nabla^2 \ell(\beta^*))$.

In this paper, we are considering testing parameters for high dimensional models and, as mentioned in [135], the traditional score function does not have a simple limiting distribution in the high dimensional setting. Thus, we use a decorrelated score function as mentioned in [135] defined as

$$S(\theta, \gamma) = \nabla_\theta \ell(\theta, \gamma) - w^T \nabla_\gamma \ell(\theta, \gamma),$$

where $w = I_{\gamma\gamma}^{-1} I_{\gamma\theta}$. Notice that $\mathbb{E}_\beta(S(\beta) \nabla_\gamma \ell(\beta)) = 0$. Suppose we are given the estimator $\hat{\beta} = (\hat{\theta}, \hat{\gamma})$ and tuning parameter λ'. We estimate $\hat{w}$ by solving

$$\hat{w} = \arg\min \|w\|_1, \text{ s.t. } \|\nabla^2_{\theta\gamma} \ell(\hat{\beta}) - w^T \nabla^2_{\gamma\gamma} \ell(\hat{\beta})\|_2 \leqslant \lambda'. \tag{D.7}$$

We use this method to estimate w because since w has dimension d which is much greater than n, we need some sparsity of w, which is useful in the rest part of this paper. Thus, we can obtain estimated decorrelated score function $\hat{S}(\theta, \hat{\gamma}) = \nabla_\theta \ell(\theta, \hat{\gamma}) - \hat{w}^T \nabla_\gamma \ell(\theta, \hat{\gamma})$.

Along the same lines as [135], we need the following assumptions. Assumption D.6.1 states that the estimators $\hat{\beta}$ and $\hat{w}$ converge to zero. However, we assume l_2-consistency here, which is weaker than the condition in [135].

Assumption D.6.1. Assume that

$$\lim_{n\to\infty} \mathbb{P}_{\beta^*}(\|\hat{\gamma} - \gamma^*\|_2 \lesssim \eta_1(n)) = 1 \text{ and } \lim_{n\to\infty} \mathbb{P}_{\beta^*}(\|\hat{w} - w^*\|_1 \lesssim \eta_2(n)) = 1,$$

where $w^* = I^{*-1}_{\gamma\gamma} I^*_{\gamma\theta}$, and $\eta_1(n)$ and $\eta_2(n)$ converges to 0, as $n \to \infty$.

Assumption D.6.2 states that the derivative of log-likelihood function is near zero at the true parameters.

Assumption D.6.2. Assume that

$$\lim_{n\to\infty} \mathbb{P}_{\beta^*}(\|\nabla_\gamma l(0, \gamma^*)\|_\infty \lesssim \eta_3(n)) = 1,$$

for some $\eta_3(n) \to 0$, as $n \to \infty$.

Assumption D.6.3 states that the Hessian matrix is relative smooth, so that we can use λ' to control $\eta_4(n)$.

Assumption D.6.3. Assume that for $\gamma_\nu = \nu\gamma^* + (1-\nu)\hat{\gamma}$ with $\nu \in [0, 1]$,

$$\lim_{n\to\infty} \mathbb{P}_{\beta^*}(\sup_{\nu\in[0,1]} \|\nabla^2_{\theta\gamma} l(0, \gamma_\nu) - \hat{w}^T \nabla^2_{\gamma\gamma} l(0, \gamma_\nu)\|_2 \lesssim \eta_4(n)) = 1,$$

for some $\eta_4(n) \to 0$, as $n \to \infty$.

Assumption D.6.4 is the central limit theorem for a linear combination of the score functions.

Assumption D.6.4. For $v^* = (1, -w^{*T})^T$, it holds that

$$\frac{\sqrt{n} v^{*T} \nabla l(0, \gamma^*)}{\sqrt{v^T I^* v}} \xrightarrow{\text{dist.}} N(0, 1),$$

where $I^* = \mathbb{E}_{\beta^*}(\nabla^2 l(0, \gamma^*))$. Furthermore, assume that $C' \leqslant I^*_{\theta|\gamma} < \infty$, where $I^*_{\theta|\gamma} = I^*_{\theta\theta} - w^{*T} I^*_{\gamma\theta}$, and $C' > 0$ is a constant.

Assumption D.6.5 states that we can estimate the information matrix relatively accurately.

Assumption D.6.5. Assume

$$\lim_{n\to\infty} \mathbb{P}_{\beta^*}(\|\nabla^2 l(\hat{\beta}) - I^*\|_{\max} \lesssim \eta_5(n)) = 1$$

for some $\eta_5(n) \to 0$, as $n \to \infty$.

Now under Assumptions D.6.1 to D.6.5, we can prove a version of Theorem 3.5 in [135] which applies to the (potentially) overlapping group lasso.

Theorem D.6.1. *Under Assumptions D.6.1 to D.6.5, with probability tending to one,*

$$n^{1/2}|\hat{S}(0,\hat{\gamma}) - S(0,\gamma^*)| \lesssim n^{1/2}(\eta_2(n)\eta_3(n) + \eta_1(n)\eta_4(n)). \tag{D.8}$$

If $n^{1/2}(\eta_2(n)\eta_3(n) + \eta_1(n)\eta_4(n)) = o(1)$*, we have*

$$n^{1/2}\hat{S}(0,\hat{\gamma})I_{\theta|\gamma}^{*-1/2} \xrightarrow{\text{dist.}} N(0,1). \tag{D.9}$$

Proof. See Theorem 3.5 in [135]. The only difference is under l_2-consistency,

$$|I_1| \leqslant \|\nabla^2_{\theta\gamma} l(0,\widetilde{\gamma}) - \hat{w}^T \nabla^2_{\gamma\gamma} l(0,\widetilde{\gamma})\|_2 \|\hat{\gamma} - \gamma^*\|_2 \lesssim \eta_1(n)\eta_4(n).$$

□

Corollary D.6.2. *Assume that Assumptions D.6.1 to D.6.5 hold. It also holds that* $\|w^*\|_1\eta_5(n) = o(1)$*,* $\eta_2(n)\|I^*_{\theta\gamma}\|_\infty = o(1)$*, and* $n^{1/2}(\eta_2(n)\eta_3(n) + \eta_1(n)\eta_4(n)) = o(1)$*. Under* $H_0 : \theta^* = 0$*, we have for any* $t \in \mathbb{R}$*,*

$$\lim_{n\to\infty} |\mathbb{P}_{\beta^*}(\hat{U}_n \leqslant t) - \Phi(t)| = 0, \tag{D.10}$$

where $\hat{U} = n^{1/2}\hat{S}(0,\hat{\gamma})\hat{I}_{\theta|\gamma}^{-1/2}$.

Proof. See the proof of Corollary 3.7 in [135]. □

D.6.2 Linear model and the corresponding decorrelated score function

Now we apply the consequences of the general results to the linear model as described in the previous section. In this section we first assume that the variance of noise is known. Consider the linear regression, $y_i = \theta^* Z_i + \gamma^{*T} Q_i + \epsilon_i$, where $Z_i \in \mathbb{R}$, $Q_i \in \mathbb{R}^{p-1}$, and the error ϵ_i satisfies $\mathbb{E}(\epsilon_i) = 0$, $\mathbb{E}(\epsilon_i^2) = \sigma^2$ for $i = 1, ..., n$. Let $\varphi_i = (Z_i, Q_i^T)^T$ denote the collection of all covariates for subject i. We first assume σ^2 is known.

Consider the overlapping group lasso estimator (D.3), the decorrelated score function is

$$S(\theta,\gamma) = -\frac{1}{n\sigma^2}\sum_{i=1}^{n}(y_i - \theta Z_i - \gamma^T Q_i)(Z_i - w^T Q_i),$$

where $w = \mathbb{E}_\beta(Q_i Q_i^T)^{-1}\mathbb{E}_\beta(Z_i Q_i)$. Since the distribution of the design matrix does not depend on β, we can replace $\mathbb{E}_\beta(\cdot)$ by $\mathbb{E}(\cdot)$ for notation simplicity. Under the null hypothesis, $H_0 : \theta^* = 0$, the decorrelated score function can be estimated by

$$\hat{S}(0,\hat{\gamma}) = -\frac{1}{n\sigma^2}\sum_{i=1}^{n}(y_i - \hat{\gamma}^T Q_i)(Z_i - \hat{w}^T Q_i),$$

where

$$\hat{w} = \arg\min \|w\|_1, \text{ s.t. } \left\| \frac{1}{n}\sum_{i=1}^{n} Q_i(Z_i - w^T Q_i) \right\|_2 \leqslant \lambda'.$$

The (partial) information matrices are

$$I^* = \sigma^{-2}\mathbb{E}(Q_i Q_i^T), \text{ and } I^*_{\theta|\gamma} = \sigma^{-2}(\mathbb{E}(Z_i^2) - \mathbb{E}(Z_i Q_i^T)\mathbb{E}(Q_i Q_i^T)^{-1}\mathbb{E}(Q_i Z_i)),$$

which can be estimated by

$$\hat{I} = \frac{1}{n\sigma^2}\sum_{i=1}^{n} Q_i Q_i^T, \text{ and } \hat{I}_{\theta|\gamma} = \sigma^{-2}\left\{\frac{1}{n}\sum_{i=1}^{n} Z_i^2 - \hat{w}^T\left(\frac{1}{n}\sum_{i=1}^{n} Q_i Z_i\right)\right\},$$

respectively. Thus, the score test statistic is $\hat{U}_n = n^{1/2}\hat{S}(0,\hat{\gamma})\hat{I}_{\theta|\gamma}^{-1/2}$.

The following theorem states the asymptotic distribution $\hat{U}_n$ under null hypothesis.

Theorem D.6.3. *Assume that*

1. $\lambda_{\min}(E(Q_i Q_i^T)) \geqslant 2\kappa_{\min}$ *for some constant* $\kappa > 0$, *and* $\limsup_{n\to\infty} \phi_{\max} \leqslant \kappa_{\max}$, *where* $\phi_{\max}$ *is defined in Definition D.5.2.*

2. *Let* $S = supp(\beta^*)$ *and* $S' = supp(w^*)$ *satisfy* $|S| = s$ *and* $|S'| = s'$. *Let* $\bar{c}$ *be the maximal number of replicates,* $\bar{d}$ *be the maximal number of group size. Assume* $n^{-1/2}(s \vee s^*)\log p = o(1)$, $\bar{d}^2 = o(\log n)$ *and* $\frac{\bar{c}^2\bar{d}}{\log p} = o(1)$.

3. ϵ_i, $w^{*T}Q_i$, *and* φ_{ij} *are all sub-Gaussian with* $\|\epsilon_i\|_{\Psi_2} \leqslant C$, $\|w^{*T}Q_i\|_{\Psi_2} \leqslant C$, *and* $\|\varphi_{ij}\|_{\Psi_2} \leqslant C$, *where* C *is a positive constant.*

4. $\lambda' \asymp \sqrt{\frac{\log p}{n}}$ *and* $\lambda \asymp \sigma\sqrt{\frac{\log p}{n}}$.

Then under $H_0 : \theta^* = 0$ *for each* $t \in \mathbb{R}$,

$$\lim_{n\to\infty} |\mathbb{P}_{\beta^*}(\hat{U}_n \leqslant t) - \Phi(t)| = 0.$$

Proof. Before the proof, we need the following lemmas in [135], which is used to ensure the assumptions of Theorem D.6.1 and Corollary D.6.2 hold. The proofs of Lemma D.6.4, D.6.6, and D.6.7 can be found in [135].

Lemma D.6.4. *Under the conditions of Theorem D.6.3, with probability at least* $1 - p^{-1}$, $\|\frac{1}{n}\sum_{i=1}^{n}(Z_i Q_i - \hat{w}^T Q_i Q_i^T)\|_\infty \leqslant C\sqrt{\frac{\log p}{n}}$, *for some* $C > 0$.

Lemma D.6.5. *Under the conditions of Theorem D.6.3, with probability at least* $1 - p^{-1}$*,*

$$\|\hat{\beta} - \beta^*\|_2^2 \leqslant C_1 \frac{\bar{c}^2 s\bar{d} \log p}{n}, \text{ and } (\hat{\beta} - \beta^*)^T H_\varphi (\hat{\beta} - \beta^*) \leqslant C_1 \kappa_{\max} \frac{\bar{c}^2 s\bar{d} \log p}{n},$$

where $H_\varphi = n^{-1} \sum_{i=1}^n \varphi_i \varphi_i^T$ *and the constant* $C_1 > 0$*.*

Proof. See Appendix D.7.5. □

Lemma D.6.6. *Under the conditions of Theorem D.6.3, with probability at least* $1 - p^{-1}$*,*

$$\|\hat{w} - w^*\|_1 \leqslant 8C\kappa^{-1} s' \sqrt{\frac{\log p}{n}},$$

where $C > 0$ *is a constant.*

Lemma D.6.7. *Under the conditions of Theorem D.6.3, it holds that* $T^* \xrightarrow{\text{dist.}} N(0, 1)$*, and*

$$\sup_{x \in \mathbb{R}} |\mathbb{P}_{\beta^*}(T^* \leqslant x) - \Phi(x)| \leqslant C n^{-1/2},$$

where $T^* = n^{1/2} S(0, \gamma^*) / I_{\theta|\gamma}^{*1/2}$ *and* C *is a positive constant not depending on* β^**.*

Now we can check that the assumptions of Theorem D.6.1 and Corollary D.6.2 hold, which finishes the proof of Theorem D.6.3. □

Next we introduce some lemmas which give properties of sub-exponential variables and norms, as well as sub-Gaussian variables and norms, which will be used in the proof of Theorem 4.4.3.

Lemma D.6.8. *(Bernstein Inequality) Let* $X_1, ..., X_n$ *be independent mean* 0 *sub-exponential random variables and let* $K = \max_i \|X_i\|_{\Psi_1}$*. Then for any* $t > 0$*,*

$$\mathbb{P}_{\beta^*}\left(\frac{1}{n}\left|\sum_{i=1}^n X_i\right| \geqslant t\right) \leqslant 2\exp\left[-C\min\left(\frac{t^2}{K^2}, \frac{t}{K}\right)n\right],$$

where $C > 0$ *is a constant.*

Lemma D.6.9. *Under the conditions of Theorem D.6.3 with probability at least* $1 - p^{-1}$, $\|\frac{1}{n}\sum_{i=1}^{n} \varphi_i \epsilon_i\|_\infty \leqslant C\sqrt{\frac{\log p}{n}}$, *for some* $C > 0$.

The proofs of Lemmas D.6.8 and D.6.9 can be found in [135]. Now, we can begin the proof of Theorem 4.4.3.

Proof. The proof is similar to [135] with a few changes. It is enough to show for any $\epsilon > 0$,

$$\lim_{n\to\infty} \sup_{\beta^* \in \Omega_0} \mathbb{P}_{\beta^*}(|\widetilde{U}_n - \hat{U}_n| \geqslant \epsilon) = 0. \tag{D.11}$$

Notice that $|\widetilde{U}_n - \hat{U}_n| = |\hat{U}_n||1 - \frac{\sigma^*}{\hat{\sigma}}|$. For a sequence of positive constants $t_n \to 0$ to be chosen later, we can show that $\lim_{n\to\infty} \sup_{\beta^* \in \Omega_0} \mathbb{P}_{\beta^*}(|\hat{U}_n| \geqslant t_n^{-1}) = 0$. It remains to show that

$$\lim_{n\to\infty} \sup_{\beta^* \in \Omega_0} \mathbb{P}_{\beta^*}\left(|1 - \frac{\sigma^*}{\hat{\sigma}}| \geqslant t_n\right) = 0. \tag{D.12}$$

Notice that

$$\hat{\sigma}^2 - \sigma^{*2} = \left(\frac{1}{n}\sum_{i=1}^{n} \epsilon_i^2 - \sigma^{*2}\right) + \hat{\Delta}^T H_\varphi \hat{\Delta} - 2\hat{\Delta}^T \frac{1}{n}\sum_{i=1}^{n} \epsilon_i \varphi_i, \tag{D.13}$$

where $\hat{\Delta} = \hat{\beta} - \beta^*$. Since $\|\epsilon_i^2\|_{\psi_1} \leqslant 2C^2$, by Lemma D.6.8, $|\frac{1}{n}\sum_{i=1}^{n} \epsilon_i^2 - \sigma^{*2}| \leqslant C\sqrt{\frac{\log n}{n}}$, for some constant C, with probability tending to one. By Lemma D.6.5, we have $\Delta^T H_\varphi \Delta \leqslant C_1 \kappa_{\max} \frac{\bar{c}^2 s\bar{d} \log p}{n}$, for some constant C_1, with probability tending to one. By Lemma D.5.6 and Lemma D.6.5, we have

$$\begin{aligned} \|\hat{\Delta}\|_1 &\leqslant C_1 s\bar{d}\bar{c}^2 \|\hat{\Delta}\|_2 \\ &\leqslant C_2 s\bar{d}\bar{c}^2 \sqrt{\frac{\bar{c}^2 s\bar{d} \log p}{n}}, \end{aligned}$$

for some constant $C_2 > 0$. By Lemma D.6.9, we have

$$\left\| \frac{1}{n} \sum_{i=1}^{n} \epsilon_i \varphi_i \right\|_\infty \leqslant C_3 \sqrt{\frac{\log p}{n}}.$$

Thus,

$$\begin{aligned} \left| \hat{\Delta}^T \frac{1}{n} \sum_{i=1}^{n} \epsilon_i \varphi_i \right| &\leqslant \|\hat{\Delta}\|_1 \left\| \frac{1}{n} \sum_{i=1}^{n} \epsilon_i \varphi_i \right\|_\infty \\ &\leqslant C_4 s\bar{d}\bar{c}^2 \sqrt{\bar{c}^2 s\bar{d}} \frac{\log p}{n}, \end{aligned}$$

for some constant $C_4 > 0$. Thus, by (D.13), we have

$$|\hat{\sigma}^2 - \sigma^{*2}| \leqslant C_0 \sqrt{\frac{\log n}{n}} \vee (\bar{c}^2 s\bar{d})^{3/2} \frac{\log p}{n},$$

for some constant C_0, with probability tending to one. Thus,

$$|1 - \frac{\sigma^*}{\hat{\sigma}}| = \hat{\sigma}^{-2} |1 + \frac{\sigma^*}{\hat{\sigma}}| |\hat{\sigma}^2 - \sigma^{*2}| \lesssim |\hat{\sigma}^2 - \sigma^{*2}| \lesssim \sqrt{\frac{\log n}{n}} \vee (\bar{c}^2 s\bar{d})^{3/2} \frac{\log p}{n},$$

with probability tending to one, because $\sigma^{*2} > C^2$ and $\hat{\sigma}^2 = \sigma^{*2} + o_{\mathbb{P}}(1)$. Thus, if we choose $t_n \gtrsim \sqrt{\frac{\log n}{n}} \vee (\bar{c}^2 s\bar{d})^{3/2} \frac{\log p}{n}$, then (D.12) holds and (D.11) holds. Then by Theorem D.6.3, the result holds. □

D.7 Proofs of Lemmas

D.7.1 Proof of Lemma D.5.3

Proof. For simplicity, we use λ instead of λ_n, γ instead of γ^λ, and γ^Z instead of $\gamma^{Z,\lambda}$ in Appendix D.7. In this proof we will use γ_t instead of γ_{J_t} for brevity. Let $\gamma^Z(T)$ be the vector with elements $\gamma^Z_{ij_{ik}}(T) = \gamma^Z_{ij_{ik}} I_{\{\beta_i^* \neq 0\}}$. Similarly, $\gamma^Z_{ij_{ik}}(T^c) = \gamma^Z_{ij_{ik}} I_{\{\beta_i^* = 0\}}$. Thus, $\gamma^Z = \gamma^Z(T) + \gamma^Z(T^c)$. Notice $\{\beta_i^* \neq 0\} = \{i \in J_t, \text{ for some } t \in T\}$. Since $f(0,0) = 0$,

and (D.5) is a minimizing problem, we have $f(\gamma, \gamma^Z) \leqslant 0$. Since $\gamma^T C\gamma \geqslant 0$ for any γ, and $\|\beta_t^Z\|_2 - \|\gamma_t^Z + \beta_t^Z\|_2 \leqslant \|\gamma_t^Z\|_2$ for any $t \in T$, combining $f(\gamma, \gamma^Z) \leqslant 0$, we have $\sum_{t\in T^c} \sqrt{d_t}\|\gamma_t^Z\|_2 \leqslant \sum_{t\in T} \sqrt{d_t}\|\gamma_t^Z\|_2$. Also, we have

$$\sum_{t\in T} \sqrt{d_t}\|\gamma_t^Z\|_2 \leqslant \sqrt{\sum_{t\in T} d_t}\|\gamma^Z(T)\|_2 \leqslant \sqrt{s_n\bar{d}}\|\gamma^Z\|_2. \tag{D.14}$$

The first inequality is true because of Cauchy's inequality, and the second inequality is true because $\bar{d} = \max\{d_1, ..., d_n\}$ and $s_n = |T|$.

For any $\beta^\lambda_{ij_{im_1}}$ and $\beta^\lambda_{ij_{im_2}}$, if they are both not zero, by KKT conditions, we have

$$-\frac{1}{n}\psi_i^T(y - \beta^T\varphi) + \frac{\lambda\sqrt{d_{j_{im_1}}}\beta^\lambda_{ij_{im_1}}}{\|\beta_{J_{j_{im_1}}}\|_2} = 0, \text{ and } -\frac{1}{n}\psi_i^T(y - \beta^T\varphi) + \frac{\lambda\sqrt{d_{j_{im_2}}}\beta^\lambda_{ij_{im_2}}}{\|\beta_{J_{j_{im_2}}}\|_2} = 0,$$

which indicates

$$\frac{\lambda\sqrt{d_{j_{im_1}}}\beta^\lambda_{ij_{im_1}}}{\|\beta_{J_{j_{im_1}}}\|_2} = \frac{\lambda\sqrt{d_{j_{im_2}}}\beta^\lambda_{ij_{im_2}}}{\|\beta_{J_{j_{im_2}}}\|_2}.$$

Since $\lambda > 0$, we have $\beta^\lambda_{ij_{im_1}}\beta^\lambda_{ij_{im_2}} \geqslant 0$. Notice if $\beta^\lambda_{ij_{im_1}}$ or $\beta^\lambda_{ij_{im_2}}$ is zero, $\beta^\lambda_{ij_{im_1}}\beta^\lambda_{ij_{im_2}} \geqslant 0$ still holds. Together with the constraints of optimization problem, we have $\gamma^\lambda_{ij_{im_1}}\gamma^\lambda_{ij_{im_2}} \geqslant 0$, which indicates $\|\gamma^Z\|_2 \leqslant \|\gamma\|_2$. Thus, together with (D.14), we have

$$\sum_{t=1}^{p_n} \sqrt{d_t}\|\gamma_t^Z\|_2 \leqslant 2\sqrt{s_n\bar{d}}\|\gamma^Z\|_2 \leqslant 2\sqrt{s_n\bar{d}}\|\gamma\|_2. \tag{D.15}$$

Since $f(\gamma, \gamma^Z) \leqslant 0$, and ignoring the non-negative term $\lambda\sum_{t\in T^c} \sqrt{d_t}\|\gamma_t^Z\|_2$, it follows that

$$n\gamma^T C\gamma \leqslant \lambda\sqrt{s_n\bar{d}}\|\gamma^Z\|_2 \leqslant \lambda\sqrt{s_n\bar{d}}\|\gamma\|_2. \tag{D.16}$$

Next, we bound the term $n\gamma^T C\gamma$ from below. Pplugging the result into (D.16) will yield the desired upper bound on the l_2-norm of γ. Let $\|\gamma^Z_{(1)}\|_2 \geqslant \|\gamma^Z_{(2)}\|_2 \geqslant \cdots \geqslant \|\gamma^Z_{(p_n)}\|_2$

be the ordered block entries of γ. Let $\{u_n\}$ be a sequence of positive integers, such that $1 \leqslant u_n \leqslant p_n$ and define the set of u_n-largest groups as $U = \{k : \|\gamma_k^Z\|_2 \geqslant \|\gamma_{(u_n)}^Z\|_2\}$. Define analogously as before $\gamma^Z(U)$, $\gamma^Z(U^c)$, $\gamma(U)$, and $\gamma(U^c)$. Thus, $\gamma^T C\gamma = (\gamma(U) + \gamma(U^c))^T C(\gamma(U) + \gamma(U^c) = \|a+b\|_2^2$, where $a = \varphi\gamma(U)/\sqrt{n}$ and $b = \varphi\gamma(U^c)/\sqrt{n}$. Thus,

$$\gamma^T C\gamma = a^T a + 2b^T a + b^T b \geqslant (\|a\|_2 - \|b\|_2)^2. \tag{D.17}$$

Assume $l = \sum_{t=1}^{p_n} \|\gamma_t^Z\|_2$. Then for every $t = 1, ..., p_n$, $\|\gamma_{(t)}^Z\|_2 \leqslant l/t$, since $\gamma_{(t)}^Z$ is the t^{th} largest group with respect to $\|\cdot\|_2$. Thus,

$$\|\gamma^Z(U^c)\|_2^2 = \sum_{t=u_n+1}^{p_n} \|\gamma_{(t)}^Z\|_2 \leqslant \bigg(\sum_{t=1}^{p_n} \|\gamma_t^Z\|_2^2\bigg)^2 \sum_{t=u_n+1}^{p_n} \frac{1}{t^2} \leqslant \bigg(\sum_{t=1}^{p_n} \sqrt{d_t}\|\gamma_t^Z\|_2\bigg)^2 \frac{1}{u_n}, \tag{D.18}$$

where the last inequality is because

$$\sum_{t=u_n+1}^{p_n} \frac{1}{t^2} \leqslant \int_{s=u_n}^{\infty} \frac{1}{s^2} ds = \frac{1}{u_n},$$

and $\sqrt{d_t} \geqslant 1$.

Together with (D.15), we have $\|\gamma^Z(U^c)\|_2^2 \leqslant 4s_n\bar{d}\|\gamma^Z\|_2^2 \frac{1}{u_n}$. Since $\gamma(U)$ has at most $\sum_{t\in U} d_t$ non-zero coefficients, and $\sum_{t\in U} d_t \leqslant u_n\bar{d}$,

$$\begin{aligned}
\|a\|_2^2 &\geqslant \phi_{\min}\bigg(\sum_{t\in U} d_t\bigg)\|\gamma(U)\|_2^2 \geqslant \phi_{\min}\bigg(\sum_{t\in U} d_t\bigg)\|\gamma^Z(U)\|_2^2 \\
&= \phi_{\min}\bigg(\sum_{t\in U} d_t\bigg)(\|\gamma^Z\|_2^2 - \|\gamma^Z(U^c)\|_2^2) \geqslant \phi_{\min}\bigg(\sum_{t\in U} d_t\bigg)(1 - \frac{4s_n\bar{d}}{u_n})\|\gamma^Z\|_2^2 \\
&\geqslant \phi_{\min}(u_n\bar{d})(1 - \frac{4s_n\bar{d}}{u_n})\|\gamma^Z\|_2^2.
\end{aligned} \tag{D.19}$$

The first inequality is true because of the definition of $\phi_{\min}(\cdot)$, and the equality is true because $\gamma^Z = \gamma^Z(U) + \gamma^Z(U^c)$. From Lemma D.5.2, $\gamma(U^c)$ has at most n non-zero groups,

which indicates

$$\|b\|_2^2 \leqslant \phi_{\max}(n\bar{d})\|\gamma(U^c)\|_2^2 \leqslant \phi_{\max}\|\gamma(U^c)\|_2^2 \leqslant \bar{d}\phi_{\max}\|\gamma^Z(U^c)\|_2^2 \leqslant \frac{4\phi_{\max}s_n\bar{d}^2}{u_n}\|\gamma^Z\|_2^2. \tag{D.20}$$

The first inequality is true because the definition of $\phi_{\max}(\cdot)$, the third inequality is true is because of Cauchy's inequality, and the last inequality is true because of (D.15) and (D.18). Thus, plugging (D.19) and (D.20) into (D.17), and combining with the facts $\sum_{t\in U} d_t \leqslant \bar{d}u_n$ and $\phi_{\max} \geqslant \phi_{\min}(u_n)$, under Assumption D.5.1, for sufficient large n, we have

$$\begin{aligned}\|a\|_2 - \|b\|_2 &\geqslant \left(\sqrt{\phi_{\min}(u_n\bar{d})(1-\frac{4s_n\bar{d}}{u_n})} - \sqrt{\frac{4\phi_{\max}s_n\bar{d}^2}{u_n}}\right)\|\gamma^Z\|_2 \\ &\geqslant \left(\sqrt{\phi_{\min}(u_n\bar{d})(1-\frac{4s_n\bar{d}}{u_n})} - \sqrt{\frac{2\kappa_{\max}s_n\bar{d}^2}{u_n}}\right)\|\gamma^Z\|_2\end{aligned}$$

Let $u_n = s_n \log n$, under Assumption D.5.1, for large n, we have

$$\|a\|_2 - \|b\|_2 \geqslant \left(\sqrt{\frac{\kappa_{\min}}{2}(1-\frac{4\bar{d}}{\log n})} - \sqrt{\frac{2\kappa_{\max}\bar{d}^2}{\log n}}\right)\|\gamma^Z\|_2.$$

Together with (D.16), we have

$$\frac{\lambda\sqrt{s_n\bar{d}}}{n}\|\gamma^Z\|_2 \geqslant \gamma^T C\gamma \geqslant \left(\sqrt{\frac{\kappa_{\min}}{2}(1-\frac{4\bar{d}}{\log n})} - \sqrt{\frac{2\kappa_{\max}\bar{d}^2}{\log n}}\right)^2\|\gamma^Z\|_2^2.$$

Since by Cauchy's inequality, we have $\|\gamma^Z\|_2^2 \geqslant \|\gamma\|_2^2/\bar{c}$. Thus,

$$\|\gamma\|_2^2 \leqslant \frac{\lambda^2\bar{c}s_n\bar{d}}{n^2}\Big/\left(\sqrt{\frac{\kappa_{\min}}{2}(1-\frac{4\bar{d}}{\log n})} - \sqrt{\frac{2\kappa_{\max}\bar{d}^2}{\log n}}\right)^2,$$

which completes the proof. □

D.7.2 Proof of Lemma D.5.4

Proof. From (D.6), for every M with $|M| \leqslant \bar{m}_n$,

$$\|\theta^M\|_2^2 \leqslant \frac{1}{n^2\phi_{\min}^2(\bar{m}_n)}\|\varphi_M^T\epsilon\|_2^2. \tag{D.21}$$

By Lemma D.6.9, with probability at least $1 - d^{-1}$, $\|\sum_{i=1}^n \varphi_i\epsilon_i\|_\infty \leqslant C\sqrt{n\log p}$. Thus,

$$\max_{M:|M|\leqslant\bar{m}_n} \|\varphi_M^T\epsilon\|_2^2 \leqslant \bar{m}_n\|\sum_{i=1}^n \varphi_i\epsilon_i\|_\infty^2 \leqslant \bar{m}_n C^2 n\log p$$

where the first inequality is true because $\|\varphi_M^T\epsilon\|_2^2 \leqslant |M|\|\varphi_M^T\epsilon\|_\infty^2$, and $|M| \leqslant \bar{m}_n$. Thus,

$$\max_{M:|M|\leqslant\bar{m}_n} \|\theta^M\|_2^2 \leqslant C^2\frac{\bar{m}_n\log p}{n\phi_{\min}^2(\bar{m}_n)},$$

which finishes the proof. □

D.7.3 Proof of Lemma D.5.5

Proof. Before the proof, we state a lemma.

Lemma D.7.1. *For $x \in \mathbb{R}^q$, suppose $\hat{x}_1 = \arg\min_x f_1(x)$ and $\hat{x}_2 = \arg\min_x f_2(x)$ where $f_1(x) = \frac{1}{2}x^TA^TAx + b^Tx$ with $A \in \mathbb{R}^{n\times q}$ which is full rank and $b \in \mathbb{R}^q$. Also, $f_2(x) = f_1(x) + c^Tx$ with $c \in \mathbb{R}^q$. Let A^Z, b^Z and c^Z be defined in the same way as before. Let $g_1(y^Z) = \frac{1}{2}\|A^Zy^Z\|_2^2 + (b^Z)^Ty^Z + h(y^Z)$ and $g_2(y^Z) = \frac{1}{2}\|A^Zy^Z\|_2^2 + (b^Z)^Ty^Z + (c^Z)^Ty^Z + h(y^Z)$, where $h(y)$ is a convex function with respect to y and everywhere sub-differentiable, and define $\hat{y}_1^Z = \arg\min_y^Z g_1(y^Z)$ and $\hat{y}_2^Z = \arg\min_y^Z g_1(y^Z)$. Then we have*

$$\|\hat{y}_2 - \hat{y}_1\|_2 \leqslant \gamma\|\hat{x}_2 - \hat{x}_1\|_2.$$

Proof. Our proof is similar to [134], with the only difference that $\|A^Z(\hat{y}_1^Z - \hat{y}_2^Z)\|_2^2 + (c^Z)^T(\hat{y}_1^Z - \hat{y}_2^Z) = \|A(\hat{y}_1 - \hat{y}_2)\|_2^2 + c^T(\hat{y}_1 - \hat{y}_2)$. □

Let $M(\xi) = A_{\lambda,\xi}$. Let $0 = \xi_1 < ... < \xi_{J+1} = 1$ be the points of discontinuity of $M(\xi)$. At these locations, variables either join the active set or are dropped from the active set. Fix some j with $1 \leqslant j \leqslant J$. Denote by M_j be the set of active groups $M(\xi)$ for any $\xi \in (\xi_j, \xi_{j+1})$. Assuming

$$\forall \xi \in (\xi_j, \xi_{j+1}) : \|\hat{\beta}^{\lambda,\xi} - \hat{\beta}^{\lambda,\xi_j}\|_2 \leqslant C(\xi - \xi_j)\|\hat{\theta}^{M_j}\|_2 \tag{D.22}$$

is true, where θ^{M_j} is the restricted OLS estimator of noise. Then

$$\begin{aligned}
\|\hat{\beta}^{\lambda,0} - \hat{\beta}^{\lambda}\|_2 &\leqslant \sum_{j=1}^{J} \|\hat{\beta}^{\lambda,\xi_j} - \hat{\beta}^{\lambda,\xi_{j+1}}\|_2 \\
&\leqslant C \max_{M:|M|\leqslant m} \|\theta^M\|_2 \sum_{j=1}^{J} (\xi_{j+1} - \xi_j) \\
&= C \max_{M:|M|\leqslant m} \|\theta^M\|_2.
\end{aligned}$$

By replacing $\hat{x}_1$, $\hat{x}_2$, $\hat{y}_1$ and $\hat{y}_2$ with $\xi\hat{\theta}^{M_j}$, $\xi_j\hat{\theta}^{M_j}$, $\hat{\beta}^{\lambda,\xi}$ and $\hat{\beta}^{\lambda,\xi_j}$ in Lemma D.7.1, respectively, we obtain (D.22). Hence, we complete the proof. □

D.7.4 Proof of Lemma D.5.6

Proof. Our proof is similar to [133]. The only thing need to be noticed is that for (38) in [133], we have

$$\begin{aligned}
(\|(X^Z_{A_{\lambda,\xi}})^T X(\beta - \hat{\beta}^{\lambda,\xi})\|_2 + \|(X^Z_{A_{\lambda,\xi}})^T \epsilon\|_2)^2 &\leqslant 2(\|(X^Z_{A_{\lambda,\xi}})^T X(\beta - \hat{\beta}^{\lambda,\xi})\|_2^2 + \|(X^Z_{A_{\lambda,\xi}})^T \epsilon\|_2^2) \\
&\leqslant 2\bar{c}(\|X^T_{A_{\lambda,\xi}} X(\beta - \hat{\beta}^{\lambda,\xi})\|_2^2 + \|X^T_{A_{\lambda,\xi}} \epsilon\|_2^2).
\end{aligned}$$

□

D.7.5 Proof of Lemma D.6.5

Proof. The second inequality is trivial if we prove the first inequality. To prove the first inequality, we only need to prove under high probability we have there exist a constant $\kappa_{\min} > 0$ such that

$$\lim \inf_{n\to\infty} \phi_{\min}(\bar{c}s\bar{m}_n \max\{\log n, \bar{c}\}) \geqslant \kappa_{\min}.$$

For any $x \in \mathbb{R}^d$ with $\|x\|_0 \leqslant \bar{c}s\bar{m}_n \max\{\log n, \bar{c}\}$, we have

$$\begin{aligned}
\frac{x^T H_\varphi x}{\|x\|_2^2} &= \frac{x^T(H_\varphi - E(\varphi\varphi^T))x}{\|x\|_2^2} + \frac{x^T E(\varphi\varphi^T)x}{\|x\|_2^2} \\
&\geqslant 2\kappa_{\min} - \frac{\|x\|_1^2 \|H_\varphi - E(\varphi\varphi^T)\|_{\max}}{\|x\|_2^2} \\
&\geqslant 2\kappa_{\min} - C\bar{c}s\bar{d}\max\{\log n, \bar{c}\}\sqrt{\frac{\log p}{n}},
\end{aligned}$$

with probability at least $1 - p^{-1}$. As n goes to infinity, $C\bar{c}s\bar{d}\max\{\log n, \bar{c}\}\sqrt{\frac{\log p}{n}}$ converges to 0. Thus, the result holds. □

D.8 Description of Functions in Section 4.6.4

- The amount of deflection of a bending function is given by

$$D_e = \frac{4}{10^9}\frac{L^3}{bh^3},$$

  where the 3 inputs are L, b, and h.

- The midpoint voltage of a transformerless OTL circuit function is given by

$$V_m = \frac{(V_{b1} + 0.74)B(R_{c2} + 9)}{B(R_{c2} + 9) + R_f} + \frac{11.35R_f}{B(R_{c2} + 9) + R_f} + \frac{0.74R_f\beta(R_{c2} + 9)}{(B(R_{c2} + 9) + R_f)R_{c1}},$$

  where $V_{b1} = 12R_{b2}/(R_{b1} + R_{b2})$, and the 6 inputs are $R_{b1}, R_{b2}, R_f, R_{c1}, R_{c2}$, and B.

Table D.1: Input ranges of the OTL circuit function, the piston simulation function, and the wing weight function.

Bending		OTL circuit		Wing weight	
L	$\in [10, 20]$	R_{b1}	$\in [50, 150]$	S_w	$\in [150, 200]$
b	$\in [1, 2]$	R_{b2}	$\in [25, 70]$	W_{fw}	$\in [220, 300]$
h	$\in [0.1, 0.2]$	R_f	$\in [0.5, 3]$	A	$\in [6, 10]$
		R_{c1}	$\in [1.2, 2.5]$	Λ	$\in [-10, 10]$
		R_{c2}	$\in [0.25, 1.2]$	q	$\in [16, 45]$
		β	$\in [50, 300]$	R	$\in [0.5, 1]$
				t_c	$\in [0.08, 0.18]$
				N_z	$\in [2.5, 6]$
				W_{dg}	$\in [1700, 2500]$
				W_p	$\in [0.025, 0.08]$

- The wing weight function models a light aircraft wing, where the wing's weight is given by

$$W = 0.036 S_w^{0.758} W_{fw}^{0.0035} \left(\frac{A}{\cos^2(\Lambda)} \right)^{0.6} q^{0.006} R^{0.04} \left(\frac{100 t_c}{\cos(\Lambda)} \right)^{-0.3} (N_z W_{dg})^{0.49} + S_w W_p, \tag{D.23}$$

  where the 10 inputs are $S_w, W_{fw}, A, \Lambda, q, R, t_c, N_z, W_{dg}$, and W_p.

The input ranges are given in Table D.1.

REFERENCES

[1] S. Mak, C.-L. Sung, X. Wang, S.-T. Yeh, Y.-H. Chang, V. R. Joseph, V. Yang, and C. J. Wu, "An efficient surrogate model for emulation and physics extraction of large eddy simulations", *Journal of the American Statistical Association*, 2018, to appear.

[2] R. B. Gramacy and D. W. Apley, "Local Gaussian process approximation for large computer experiments", *Journal of Computational and Graphical Statistics*, vol. 24, no. 2, pp. 561–578, 2015.

[3] V. I. Zarnitsyna, J. Huang, F. Zhang, Y.-H. Chien, D. Leckband, and C. Zhu, "Memory in receptor-ligand-mediated cell adhesion", *Proceedings of the National Academy of Science, U.S.A.*, vol. 104, no. 46, pp. 18 037–18 042, 2007.

[4] Y. Hung, V. Zarnitsyna, Y. Zhang, C. Zhu, and C. F. J. Wu, "Binary time series modeling with application to adhesion frequency experiments", *Journal of the American Statistical Association*, vol. 103, no. 483, 2008.

[5] T. J. Santner, B. J. Williams, and W. I. Notz, *The design and analysis of computer experiments*. Springer New York, 2003.

[6] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, "Design and analysis of computer experiments", *Statistical Science*, vol. 4, no. 4, pp. 409–423, 1989.

[7] C. K. Williams and D. Barber, "Bayesian classification with Gaussian processes", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 12, pp. 1342–1351, 1998.

[8] H. Zhang, "On estimation and prediction for spatial generalized linear mixed models", *Biometrics*, vol. 58, no. 1, pp. 129–136, 2002.

[9] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*. the MIT Press, 2006.

[10] H. Nickisch and C. E. Rasmussen, "Approximations for binary Gaussian process classification", *Journal of Machine Learning Research*, vol. 9, pp. 2035–2078, 2008.

[11] B. Wang and J. Q. Shi, "Generalized Gaussian process regression model for non-Gaussian functional data", *Journal of the American Statistical Association*, vol. 109, no. 507, pp. 1123–1133, 2014.

[12] A. E. Gelfand, A. M. Schmidt, S. Banerjee, and C. Sirmans, "Nonstationary multivariate process modeling through spatially varying coregionalization", *Test*, vol. 13, no. 2, pp. 263–312, 2004.

[13] S. Conti and A. O'Hagan, "Bayesian emulation of complex multi-output and dynamic computer models", *Journal of Statistical Planning and Inference*, vol. 140, no. 3, pp. 640–651, 2010.

[14] T. E. Fricker, J. E. Oakley, and N. M. Urban, "Multivariate Gaussian process emulators with nonseparable covariance structures", *Technometrics*, vol. 55, no. 1, pp. 47–56, 2013.

[15] B. Yan, J. Qin, J. Dai, Q. Fan, and J. B. Bernstein, "Reliability simulation and circuit-failure analysis in analog and mixed-signal applications", *IEEE Transactions on Device and Materials Reliability*, vol. 9, no. 3, pp. 339–347, 2009.

[16] W. Gerstner, A. K. Kreiter, H. Markram, and A. V. Herz, "Neural codes: Firing rates and beyond", *Proceedings of the National Academy of Sciences*, vol. 94, no. 24, pp. 12 740–12 741, 1997.

[17] R. Mayrhofer, M. Affenzeller, H. Prähofer, G. Höfer, and A. Fried, "DEVS simulation of spiking neural networks", in *Proceedings of Cybernetics and Systems (EMCSR)*, vol. 2, 2002, 573–578.

[18] C.-L. Sung, *binaryGP: Fit and predict a Gaussian process model with (time-series) binary response*, R package version 0.2, 2017.

[19] R Core Team, *R: A language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria, 2015.

[20] M. Plumlee and V. R. Joseph, "Orthogonal Gaussian process models", *Statistica Sinica*, vol. 28, no. 2, pp. 601–619, 2018.

[21] J. S. Hodges and B. J. Reich, "Adding spatially-correlated errors can mess up the fixed effect you love", *The American Statistician*, vol. 64, no. 4, pp. 325–334, 2010.

[22] C. J. Paciorek, "The importance of scale for spatial-confounding bias and precision of spatial regression estimators", *Statistical Science*, vol. 25, no. 1, p. 107, 2010.

[23] R. Tuo and C. F. J. Wu, "Efficient calibration for imperfect computer models", *The Annals of Statistics*, vol. 43, no. 6, pp. 2331–2352, 2015.

[24] R. B. Gramacy and N. G. Polson, "Particle learning of Gaussian process models for sequential design and optimization", *Journal of Computational and Graphical Statistics*, vol. 20, no. 1, pp. 102–118, 2011.

[25] S. L. Zeger and B. Qaqish, "Markov regression models for time series: A quasi-likelihood approach", *Biometrics*, vol. 44, no. 4, pp. 1019–1031, 1988.

[26] M. A. Benjamin, R. A. Rigby, and D. M. Stasinopoulos, "Generalized autoregressive moving average models", *Journal of the American Statistical Association*, vol. 98, no. 461, pp. 214–223, 2003.

[27] D. R. Cox, "Regression models and life-tables", *Journal of the Royal Statistical Society, Series B*, vol. 34, no. 2, pp. 187–220, 1972.

[28] ——, "Partial likelihood", *Biometrika*, vol. 62, no. 2, pp. 269–276, 1975.

[29] E. Slud and B. Kedem, "Partial likelihood analysis of logistic regression and autoregression", *Statistica Sinica*, vol. 4, no. 1, pp. 89–106, 1994.

[30] N. E. Breslow and D. G. Clayton, "Approximate inference in generalized linear mixed models", *Journal of the American statistical Association*, vol. 88, no. 421, pp. 9–25, 1993.

[31] O. E. Barndorff-Nielsen and D. R. Cox, *Asymptotic techniques for use in statistics*. London: Chapman & Hall, 1997.

[32] H. D. Patterson and R. Thompson, "Recovery of inter-block information when block sizes are unequal", *Biometrika*, vol. 58, no. 3, pp. 545–554, 1971.

[33] ——, "Maximum likelihood estimation of components of variance", in *Proceedings of the 8th International Biometric Conference*, Biometric Society, Washington, DC, 1974, pp. 197–207.

[34] D. A. Harville, "Maximum likelihood approaches to variance component estimation and to related problems", *Journal of the American Statistical Association*, vol. 72, no. 358, pp. 320–338, 1977.

[35] S. R. Searle, G. Casella, and C. E. McCulloch, *Variance components*. New York: Wiley, 2009.

[36] D. A. Harville, "Bayesian inference for variance components using only error contrasts", *Biometrika*, vol. 61, no. 2, pp. 383–385, 1974.

[37] N. Cressie and S. N. Lahiri, "The asymptotic distribution of reml estimators", *Journal of Multivariate Analysis*, vol. 45, no. 2, pp. 217–233, 1993.

[38] ——, "Asymptotics for reml estimation of spatial covariance parameters", *Journal of Statistical Planning and Inference*, vol. 50, no. 3, pp. 327–341, 1996.

[39] T. Wutzler, *Logitnorm: Functions for the logitnormal distribution.* R package version 0.8.29, 2012.

[40] R Mead, "A generalised logit-normal distribution", *Biometrics*, vol. 21, no. 3, pp. 721–732, 1965.

[41] J Atchison and S. M. Shen, "Logistic-normal distributions: Some properties and uses", *Biometrika*, vol. 67, no. 2, pp. 261–272, 1980.

[42] P. Frederic and F. Lad, "Two moments of the logitnormal distribution", *Communications in Statistics Simulation and Computation*, vol. 37, no. 7, pp. 1263–1269, 2008.

[43] F. Liu and M. West, "A dynamic modelling strategy for bayesian computer model emulation", *Bayesian Analysis*, vol. 4, no. 2, pp. 393–411, 2009.

[44] J. H. Friedman, "Multivariate adaptive regression splines", *The Annals of Statistics*, vol. 19, no. 1, pp. 1–67, 1991.

[45] R. Li and A. Sudjianto, "Analysis of computer experiments using penalized likelihood in Gaussian kriging models", *Technometrics*, vol. 47, no. 2, pp. 111–120, 2005.

[46] J. Q. Shi and T. Choi, *Gaussian process regression analysis for functional data*. CRC Press, 2011.

[47] A. Karatzoglou, A. Smola, K. Hornik, and A. Zeileis, "Kernlab – an S4 package for kernel methods in R", *Journal of Statistical Software*, vol. 11, no. 9, pp. 1–20, 2004.

[48] J. Q. Shi and Y. Cheng, *GPFDA: Apply Gaussian process in functional data analysis*, R package version 2.2, 2014.

[49] D. T. Gillespie, "A general method for numerically simulating the stochastic time evolution of coupled chemical reactions", *Journal of Computational Physics*, vol. 22, no. 4, pp. 403–434, 1976.

[50] J. Huang, V. I. Zarnitsyna, B. Liu, L. J. Edwards, N. Jiang, B. D. Evavold, and C. Zhu, "The kinetics of two-dimensional TCR and pMHC interactions determine t-cell responsiveness", *Nature*, vol. 464, no. 7290, pp. 932–936, 2010.

[51] B. Tang, "Orthogonal array-based latin hypercubes", *Journal of the American Statistical Association*, vol. 88, no. 424, pp. 1392–1397, 1993.

[52] C.-L. Sung, R. B. Gramacy, and B. Haaland, "Exploiting variance reduction potential in local Gaussian process search", *Statistica Sinica*, vol. 28, no. 2, pp. 577–600, 2018.

[53] M. C. Kennedy and A. O'Hagan, "Bayesian calibration of computer models (with discussion)", *Journal of the Royal Statistical Society: Series B*, vol. 63, no. 3, pp. 425–464, 2001.

[54] R. B. Gramacy, D. Bingham, J. P. Holloway, M. J. Grosskopf, C. C. Kuranz, E. Rutter, M. Trantham, and R. P. Drake, "Calibrating a large computer experiment simulating radiative shock hydrodynamics", *The Annals of Applied Statistics*, vol. 9, no. 3, pp. 1141–1168, 2015.

[55] M. J. Bayarri, J. O. Berger, R. Paulo, J. Sacks, J. A. Cafeo, J. Cavendish, C.-H. Lin, and J. Tu, "A framework for validation of computer models", *Technometrics*, vol. 49, no. 2, pp. 138–154, 2007.

[56] M. Farah, P. Birrell, S. Conti, and D. D. Angelis, "Bayesian emulation and calibration of a dynamic epidemic model for A/H1N1 influenza", *Journal of the American Statistical Association*, vol. 109, no. 508, pp. 1398–1411, 2014.

[57] R. Tuo and C. F. J. Wu, "A theoretical framework for calibration in computer models: Parametrization, estimation and convergence properties", *SIAM/ASA Journal on Uncertainty Quantification*, vol. 4, no. 1, pp. 767–795, 2016.

[58] D. Higdon, J. Gattiker, B. Williams, and M. Rightley, "Computer model calibration using high-dimensional output", *Journal of the American Statistical Association*, vol. 103, no. 482, pp. 570–583, 2008.

[59] S. E. Chesla, P. Selvaraj, and C. Zhu, "Measuring two-dimensional receptor-ligand binding kinetics by micropipette", *Biophysical Journal*, vol. 75, pp. 1553–1572, 1998.

[60] B. T. Marshall, M. Long, J. W. Piper, T. Yago, R. P. McEver, and C. Zhu, "Direct observation of catch bonds involving cell-adhesion molecules", *Nature*, vol. 423, no. 6936, pp. 190–193, 2003.

[61] D. Higdon, J. Gattiker, E. Lawrence, C. Jackson, M. Tobis, M. Pratola, S. Habib, K. Heitmann, and S. Price, "Computer model calibration using the ensemble Kalman filter", *Technometrics*, vol. 55, no. 4, pp. 488–500, 2013.

[62] D. Higdon, M. Kennedy, J. C. Cavendish, J. A. Cafeo, and R. D. Ryne, "Combining field data and computer simulations for calibration and prediction", *SIAM Journal on Scientific Computing*, vol. 26, no. 2, pp. 448–466, 2004.

[63] T. Larssen, R. B. Huseby, B. J. Cosby, G. Høst, T. Høgåsen, and M. Aldrin, "Forecasting acidification effects using a bayesian calibration and uncertainty propagation approach", *Environmental Science & Technology*, vol. 40, no. 24, pp. 7841–7847, 2006.

[64] G. Han, T. J. Santner, and J. J. Rawlinson, "Simultaneous determination of tuning and calibration parameters for computer experiments", *Technometrics*, vol. 51, no. 4, pp. 464–474, 2009.

[65] P. J. Green and B. S. Yandell, "Semi-parametric generalized linear models", in *Proceedings 2nd International GLIM Conference, Lancaster, Lecture Notes in Statistics No. 32*, New York: Springer, 1985, pp. 44–55.

[66] T. Hastie and R. Tibshirani, *Generalized additive models*. New York: Chapman and Hall, 1990.

[67] G. Wahba, C. Gu, Y. Wang, and R Campbell, "Soft classification, a.k.a. risk estimation, via penalized log likelihood and smoothing spline analysis of variance", in *The Mathematics of Generalization, ed. D. H. Wolpert, Santa Fe Institute Studies in the Sciences of Complexity, Reading, MA: Addison-Wesley*, 1995, 329–360.

[68] J. Zhu and T. Hastie, "Kernel logistic regression and the import vector machine", *Journal of Computational and Graphical Statistics*, vol. 14, no. 1, pp. 185–205, 2005.

[69] C.-L. Sung, Y. Hung, W. Rittase, C. Zhu, and C. F. J. Wu, "A generalized Gaussian process model for computer experiments with binary time series", *Journal of the American Statistical Association*, 2018, under revision, arXiv preprint arXiv:1705.02511.

[70] S. van de Geer, *Empirical processes in m-estimation*. Cambridge University Press, 2000.

[71] H. Wendland, *Scattered data approximation*. Cambridge University Press, 2004, vol. 17.

[72] A. W. van der Vaart and J. A. Wellner, *Weak convergence and empirical processes: With applications to statistics*. Springer, New York, 1996.

[73] M. R. Kosorok, *Introduction to empirical processes and semiparametric inference*. Springer, New York, 2008.

[74] P. J. Bickel, C. A. J. Klaassen, Y Ritov, and J. A. Wellner, *Efficient and adaptive estimation for semiparametric models*. Johns Hopkins Univ. Press, Baltimore, MD., 1993.

[75] M. Plumlee, "Bayesian calibration of inexact computer models", *Journal of the American Statistical Association*, vol. 112, no. 519, pp. 1274–1285, 2017.

[76] R. Tuo, "Adjustments to computer models via projected kernel calibration", *SIAM/ASA Journal on Uncertainty Quantification*, vol. to appear, 2018.

[77] W. Eckstein, *Computer simulation of ion-solid interactions*. Springer Science & Business Media, 2013, vol. 10.

[78] A. Cangelosi and D. Parisi, *Simulating the evolution of language*. Springer Science & Business Media, 2012.

[79] R. Furrer, M. G. Genton, and D. Nychka, "Covariance tapering for interpolation of large spatial datasets", *Journal of Computational and Graphical Statistics*, vol. 15, no. 3, pp. 502–523, 2006.

[80] B. Haaland, P. Z. Qian, *et al.*, "Accurate emulators for large-scale computer experiments", *The Annals of Statistics*, vol. 39, no. 6, pp. 2974–3002, 2011.

[81] D. Nychka, S. Bandyopadhyay, D. Hammerling, F. Lindgren, and S. Sain, "A multiresolution Gaussian process model for the analysis of large spatial data sets", *Journal of Computational and Graphical Statistics*, vol. 24, no. 2, pp. 579–599, 2015.

[82] C. J. Paciorek, B. Lipshitz, W. Zhuo, C. G. Kaufman, R. C. Thomas, *et al.*, "Parallelizing Gaussian process calculations in R", *Journal of Statistical Software*, vol. 63, no. 10, pp. 1–23, 2015.

[83] M. Plumlee, "Fast prediction of deterministic functions using sparse grid experimental designs", *Journal of the American Statistical Association*, vol. 109, no. 508, pp. 1581–1591, 2014.

[84] K.-T. Fang, R. Li, and A. Sudjianto, *Design and modeling for computer experiments*. CRC Press, 2005.

[85] A. Datta, S. Banerjee, A. O. Finley, and A. E. Gelfand, "Hierarchical nearest-neighbor Gaussian process models for large geostatistical datasets", *Journal of the American Statistical Association*, vol. 111, no. 514, pp. 800–812, 2016.

[86] D. A. Harville, *Matrix algebra from a statistician's perspective*. Springer, 1997, vol. 157.

[87] R. B. Gramacy, "laGP: Large-scale spatial modeling via local approximate Gaussian processes in R", *Journal of Statistical Software*, vol. 72, no. 1, pp. 1–46, 2016.

[88] Y. Liu and Y. Hung, "Latin hypercube design-based block bootstrap for computer experiment modeling", Rutgers, Tech. Rep., 2015.

[89] R. B. Gramacy, J. Niemi, and R. M. Weiss, "Massively parallel approximate Gaussian process regression", *SIAM/ASA Journal on Uncertainty Quantification*, vol. 2, no. 1, pp. 564–584, 2014.

[90] R. B. Gramacy and B. Haaland, "Speeding up neighborhood search in local Gaussian process prediction", *Technometrics*, vol. 58, no. 3, pp. 294–303, 2016.

[91] J. L. Bentley, "Multidimensional binary search trees used for associative searching", *Communications of the ACM*, vol. 18, no. 9, pp. 509–517, 1975.

[92] G. H. Golub and C. F. Van Loan, *Matrix computations*. JHU Press, 1996.

[93] J. Kuczynski and H. Wozniakowski, "Estimating the largest eigenvalue by the power and lanczos algorithms with a random start", *SIAM Journal on Matrix Analysis and Applications*, vol. 13, no. 4, pp. 1094–1122, 1992.

[94] H. Zhu, C. K. Williams, R. Rohwer, and M. Morciniec, "Gaussian regression and optimal finite dimensional linear models", *Aston University*, 1997.

[95] C. Williams and M. Seeger, "Using the nyström method to speed up kernel machines", in *Proceedings of the 14th Annual Conference on Neural Information Processing Systems*, 2001, pp. 682–688.

[96] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality", in *Proceedings of the Thirtieth annual ACM symposium on Theory of computing*, ACM, 1998, pp. 604–613.

[97] B. Van Durme and A. Lall, "Online generation of locality sensitive hash signatures", in *Proceedings of the ACL 2010 Conference Short Papers*, Association for Computational Linguistics, 2010, pp. 231–235.

[98] J. Leskovec, A. Rajaraman, and J. D. Ullman, *Mining of massive datasets*. Cambridge University Press, 2014.

[99] P. Jain, B. Kulis, and K. Grauman, "Fast image search for learned metrics", in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, IEEE, 2008, pp. 1–8.

[100] S. Arya, D. Mount, S. E. Kemp, and G. Jefferis, *RANN: Fast nearest neighbour search (wraps arya and mount's ann library)*, R package version 2.5, 2015.

[101] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python", *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[102] M. Bawa, T. Condie, and P. Ganesan, "Lsh forest: Self-tuning indexes for similarity search", in *Proceedings of the 14th International Conference on World Wide Web*, ACM, 2005, pp. 651–660.

[103] C. J. G. Bellosta, *rPython: Package allowing r to call python*, R package version 0.0-6, 2015.

[104] P. Bratley and B. L. Fox, "Algorithm 659: Implementing sobol's quasirandom sequence generator", *ACM Transactions on Mathematical Software (TOMS)*, vol. 14, no. 1, pp. 88–100, 1988.

[105] C. Q. Lam and W. I. Notz, "Sequential adaptive designs in computer experiments for response surface model fit", PhD thesis, The Ohio State University, 2008.

[106] M. C. Shewry and H. P. Wynn, "Maximum entropy sampling", *Journal of Applied Statistics*, vol. 14, no. 2, pp. 165–170, 1987.

[107] J. Hötzer, M. Jainta, P. Steinmetz, B. Nestler, A. Dennstedt, A. Genau, M. Bauer, H. Köstler, and U. Rüde, "Large scale phase-field simulations of directional ternary eutectic solidification", *Acta Materialia*, vol. 93, pp. 194–204, 2015.

[108] K. Bibbins-Domingo, G. M. Chertow, P. G. Coxson, A. Moran, J. M. Lightwood, M. J. Pletcher, and L. Goldman, "Projected effect of dietary salt reductions on future cardiovascular disease", *New England Journal of Medicine*, vol. 362, no. 7, pp. 590–599, 2010.

[109] M. Pratola and D. Higdon, "Bayesian additive regression tree calibration of complex high-dimensional computer models", *Technometrics*, vol. 58, no. 2, pp. 166–179, 2016.

[110] J. Goh, D. Bingham, J. P. Holloway, M. J. Grosskopf, C. C. Kuranz, and E. Rutter, "Prediction and computer model calibration using outputs from multifidelity simulators", *Technometrics*, vol. 55, no. 4, pp. 501–512, 2013.

[111] K. Wang, C. Zhang, J. Su, B. Wang, and Y. Hung, "Optimisation of composite manufacturing processes with computer experiments and kriging methods", *International Journal of Computer Integrated Manufacturing*, vol. 26, no. 3, pp. 216–226, 2013.

[112] S. Asmussen and P. W. Glynn, *Stochastic simulation: Algorithms and analysis*. Springer Science & Business Media, 2007, vol. 57.

[113] M. Lukić and J. Beder, "Stochastic processes with sample paths in reproducing kernel hilbert spaces", *Transactions of the American Mathematical Society*, vol. 353, no. 10, pp. 3945–3969, 2001.

[114] G. Wahba, *Spline models for observational data*. Siam, 1990, vol. 59.

[115] C. G. Kaufman, D. Bingham, S. Habib, K. Heitmann, and J. A. Frieman, "Efficient emulators of computer experiments using compactly supported correlation functions, with an application to cosmology", *The Annals of Applied Statistics*, vol. 5, no. 4, pp. 2470–2492, 2011.

[116] P. Ranjan, R. Haynes, and R. Karsten, "A computationally stable approach to Gaussian process interpolation of deterministic computer simulation data", *Technometrics*, vol. 53, no. 4, pp. 366–378, 2011.

[117] H. Wendland, "Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree", *Advances in Computational Mathematics*, vol. 4, no. 1, pp. 389–396, 1995.

[118] A. B. Owen, "Monte carlo variance of scrambled net quadrature", *SIAM Journal on Numerical Analysis*, vol. 34, no. 5, pp. 1884–1910, 1997.

[119] C. J. Stone, M. H. Hansen, C. Kooperberg, Y. K. Truong, *et al.*, "Polynomial splines and their tensor products in extended linear modeling: 1994 wald memorial lecture", *The Annals of Statistics*, vol. 25, no. 4, pp. 1371–1470, 1997.

[120] C. F. J. Wu and M. S. Hamada, *Experiments: Planning, analysis, and optimization*, 2nd. John Wiley & Sons, 2009.

[121] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables", *Journal of the Royal Statistical Society: Series B*, vol. 68, no. 1, pp. 49–67, 2006.

[122] L. Jacob, G. Obozinski, and J.-P. Vert, "Group lasso with overlap and graph lasso", in *Proceedings of the 26th Annual International Conference on Machine Learning*, ACM, 2009, pp. 433–440.

[123] L. Meier, S. Van De Geer, and P. Bühlmann, "The group lasso for logistic regression", *Journal of the Royal Statistical Society: Series B*, vol. 70, no. 1, pp. 53–71, 2008.

[124] V. Roth and B. Fischer, "The group-lasso for generalized linear models: Uniqueness of solutions and efficient algorithms", in *Proceedings of the 25th international conference on Machine learning*, ACM, 2008, pp. 848–855.

[125] L. Meier, *Grplasso: Fitting user specified models with group lasso penalty*, R package version 0.4-5, 2015.

[126] P. Craven and G. Wahba, "Smoothing noisy data with spline functions", *Numerische Mathematik*, vol. 31, no. 4, pp. 377–403, 1978.

[127] J. Shao, "An asymptotic theory for linear model selection", *Statistica Sinica*, vol. 7, no. 2, pp. 221–242, 1997.

[128] R. Shibata, "Approximate efficiency of a selection procedure for the number of regression variables", *Biometrika*, vol. 71, no. 1, pp. 43–49, 1984.

[129] K.-C. Li, "Asymptotic optimality for c_p, c_l, cross-validation and generalized cross-validation: Discrete index set", *The Annals of Statistics*, vol. 15, no. 3, pp. 958–975, 1987.

[130] H. Wang, R. Li, and C.-L. Tsai, "Tuning parameter selectors for the smoothly clipped absolute deviation method", *Biometrika*, vol. 94, no. 3, pp. 553–568, 2007.

[131] Y. Zhang, R. Li, and C.-L. Tsai, "Regularization parameter selections via generalized information criterion", *Journal of the American Statistical Association*, vol. 105, no. 489, pp. 312–323, 2010.

[132] J. Fan and R. Li, "Variable selection via nonconcave penalized likelihood and its oracle properties", *Journal of the American statistical Association*, vol. 96, no. 456, pp. 1348–1360, 2001.

[133] N. Meinshausen and B. Yu, "Lasso-type recovery of sparse representations for high-dimensional data", *The Annals of Statistics*, vol. 37, no. 1, pp. 246–270, 2009.

[134] H. Liu and J. Zhang, "Estimation consistency of the group lasso and its applications.", in *AISTATS*, vol. 5, 2009, pp. 376–383.

[135] Y. Ning and H. Liu, "A general theory of hypothesis tests and confidence regions for sparse high dimensional models", *The Annals of Statistics*, vol. 45, no. 1, pp. 158–195, 2017.

[136] D. W. Apley, "An empirical adjustment of the uncertainty quantification in Gaussian process modeling", in *Statistical Perspectives of Uncertainty Quantification 2017*, https://pwp.gatech.edu/spuq-2017/program/, 2017.

[137] G. M. Dancik, *Mlegp: Maximum likelihood estimates of Gaussian processes*, R package version 3.1.4, 2013.

[138] C.-L. Sung, *MRFA: Fitting and predicting large-scale nonlinear regression problems using multi-resolution functional anova (MRFA) approach*, R package version 0.1, 2017.

[139] Revolution Analytics and S. Weston, *Foreach: Provides foreach looping construct for r*, R package version 1.4.3, 2015.

[140] T. Gneiting and A. E. Raftery, "Strictly proper scoring rules, prediction, and estimation", *Journal of the American Statistical Association*, vol. 102, no. 477, pp. 359–378, 2007.

[141] R. Kenett and S. Zacks, *Modern industrial statistics: Design and control of quality and reliability*. Pacific Grove, CA: Duxbury Press, 1998.

[142] B. MacDonald, P. Ranjan, and H. Chipman, "GPfit: An R package for fitting a Gaussian process model to deterministic simulator outputs", *Journal of Statistical Software*, vol. 64, no. 12, pp. 1–23, 2015.

[143] R. B. Gramacy and H. K. Lee, "Adaptive design and analysis of supercomputer experiments", *Technometrics*, vol. 51, no. 2, pp. 130–145, 2009.

[144] W. Wang and B. Haaland, "Controlling sources of inaccuracy in stochastic kriging", *Arxiv preprint arxiv:1706.00886*, 2017.

[145] M. Plumlee and D. W. Apley, "Lifted brownian kriging models", *Technometrics*, vol. 59, no. 2, pp. 165–177, 2017.

[146] E. N. Ben-Ari and D. M. Steinberg, "Modeling data from computer experiments: An empirical comparison of kriging with mars and projection pursuit regression", *Quality Engineering*, vol. 19, no. 4, pp. 327–338, 2007.

[147] A. Forrester, A. Keane, *et al.*, *Engineering design via surrogate modelling: A practical guide*. John Wiley & Sons, 2008.

[148] L. Breiman, "Pasting small votes for classification in large databases and on-line", *Machine Learning*, vol. 36, no. 1-2, pp. 85–103, 1999.

[149] P. Büchlmann and B. Yu, "Analyzing bagging", *Annals of Statistics*, vol. 30, no. 4, pp. 927–961, 2002.

[150] A. Buja and W. Stuetzle, "Observations on bagging", *Statistica Sinica*, vol. 16, no. 2, pp. 323–351, 2006.

[151] J. H. Friedman and P. Hall, "On bagging and nonlinear estimation", *Journal of Statistical Planning and Inference*, vol. 137, no. 3, pp. 669–683, 2007.

[152] L. Breiman, "Bagging predictors", *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[153] M. D. McKay, R. J. Beckman, and W. J. Conover, "Comparison of three methods for selecting values of input variables in the analysis of output from a computer code", *Technometrics*, vol. 21, no. 2, pp. 239–245, 1979.

[154] M. E. Johnson, L. M. Moore, and D. Ylvisaker, "Minimax and maximin distance designs", *Journal of Statistical Planning and Inference*, vol. 26, no. 2, pp. 131–148, 1990.

[155] E. Mammen and S. van de Geer, "Penalized quasi-likelihood estimation in partial linear models", *The Annals of Statistics*, vol. 25, no. 3, pp. 1014–1035, 1997.

[156] A. W. van der Vaart, *Asymptotic statistics (cambridge series in statistical and probabilistic mathematics)*. Cambridge University Press, 1998.

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

[158] R. G. Bartle, *The elements of integration and lebesgue measure*. John Wiley & Sons, 2014.

[159] J. Fan, S. Guo, and N. Hao, "Variance estimation using refitted cross-validation in ultrahigh dimensional regression", *Journal of the Royal Statistical Society: Series B*, vol. 74, no. 1, pp. 37–65, 2012.