Gaussian Process Regression for Non-Gaussian Data

Jinzhao Liu

Thesis submitted for the degree of

Doctor of Philosophy

School of Mathematics, Statistics & Physics

Newcastle University

Newcastle upon Tyne

United Kingdom

September 2021

Acknowledgements

I would like to convey my deep gratitude to many people who have helped and supported me during the four-years research at the Newcastle University.

Firstly, I would like to express my utmost appreciation to my supervisors, Professor Jian Qing Shi and Dr Tom Nye, for their suggestions, encouragements and patience. They have not only aided me in my study, but also enlightened me on being a researcher. Their enthusiasm and deep thinking have inspired me a lot. This thesis is not possible without their talents and knowledge.

My acknowledge goes out to all staff in the Department of Mathematics, Statistics and Physics for their helps. I immensely benefited from discussions with Dr Stuart Hall who explained a plenty of mathematical concepts and formulas related or unrelated to this thesis, especially on Riemannian manifolds.

A big thanks to my friends and colleagues at the Newcastle University and Durham University who provided warmly care and enjoyable time.

A special gratitude to my gorgeous cat DP Liu who has brought many laughs during my research years and released my pressures.

My appreciation goes out to Zhe Ji for her care, for her love and for being a wonderful partner to be with.

I would like to express acknowledge to my parents for their endless love and unconditional support which encouraged me to pursue a PhD.

Abstract

Gaussian process regression (GPR) is a kernel-based non-linear non-parametric model which is widely used in many research fields. Functional data analysis handles data which are in the form of functions, shapes or more general objects, such as a stochastic process. A challenge in GPR is how to deal with non-Gaussian distributed data which has been increasingly recorded in scientific and industrial fields. The thesis contains two main contributions: (i) extensions of GPR for non-Gaussian data, and (ii) the development of a Gaussian process functional regression (GPFR) model for manifold-valued data. First, we review the GPR model and the GPFR model with their properties and inference methods, followed by a summary of background mathematics for Riemannian manifolds, some special probability distributions and some useful algorithms. After the literature review, we introduce a truncated Gaussian process regression (TGPR) model which is an extension of Gaussian process regression for data with a truncated normal distribution. We also study a normal distribution with multi-truncation. In addition, Gaussian process regression is extended to model Gamma-distributed data and this model is denoted as GPRG. Afterwards, we generalise the GPRG model to high-dimensional outputs. Simulation studies are used to assess the performance of the TGPR and GPRG models have good prediction accuracy. We then move on to consider more complex manifold-valued data. We introduce a novel regression model for such data within a probabilistic framework, called wrapped Gaussian process functional regression. We describe an algorithm in which mean structure and covariance structure are estimated simultaneously and then updated iteratively. Various simulation experiments are presented, and these show our model outperforms some other models. The proposed method is applied to some flight trajectory data and provides accurate predictions for missing parts of the trajectories.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Research motivation

Over the past decades, statistical regression models have attracted more attention in scientific, engineering and social research. In these fields, Gaussian distributed data are widely assumed (Salkind, 2010). It follows that Gaussian process regression (GPR) is naturally a reasonable and essential technique for such data, which learns the non-linear relationship between one-dimensional or multi-dimensional predictor variables and Gaussian-distributed response variables non-parametrically with Bayesian interpretation. However, in the real world, many data are not normally distributed. One of challenging and important contributions of this thesis is to consider some regression models for non-Gaussian distributed data in which we use GP as a latent process and derive efficient algorithms for inference, such as estimation and prediction. Specifically, we first extend the Gaussian process regression model for truncated-Gaussian-distributed data and Gamma-distributed data via different methods. In addition, with the development of technology, more complex data can be recorded, such as manifold-valued functional data with replicated measurements which are also called batch data (group data). For example, we suppose there are two batches $s_1 = \{X_{1,i}(t), i = 1, ..., n_1\}$ and $s_2 = \{X_{2,j}(t), j = 1, ..., n_2\}$ where $X_{1,i}(t)$s refer to a set of similar stochastic processes and $X_{2,i}(t)$s refer to another set of similar stochastic processes. Due to lack of a vector space structure, it is not reasonable to apply traditional regression methods, such as GPR and functional linear regression (FLR), on a Riemannian manifold directly. Therefore, we introduce a novel regression framework

within a probabilistic framework for functional batch data (as known as group data) on Riemannian manifolds.

1.2 Gaussian process regression, functional data analysis and extensions

Since Gaussian process regression has many attractive properties, such as probabilistic prediction and a Bayesian framework, it is considered as an effective and essential model for regression or classification problems (Williams and Rasmussen, 2006). In addition, for non-linear problems, GPR is an alternative to unsupervised learning (MacKay, 1997). There is also a connection between GPR and neural networks: a neural network with one layer and an infinite number of neurons is equivalent to a Gaussian process (Neal, 2012). Moreover, by applying a Karhunen-Loéve orthogonal expansion (Wahba, 1990), a Gaussian process can be decomposed to an infinite weighted sum of eigen-functions whose coefficients follow independent normal distributions. Therefore, GPR is also an extension of a standard linear regression model.

As an extensively used regression model, GPR can be also considered as a Bayesian non-parametric model over stochastic process with a Gaussian process prior involving mean function and covariance function. The posterior is still a Gaussian process which provides a distribution of predictions. Flexible choices of covariance function, such as *squared exponential* (SE) covariance function and *Matérn class* (MC) covariance function, enable scholars to accommodate a wide range of non-linear regression problems. Furthermore, researchers can incorporate their prior knowledge into the regression model which makes GPR more efficient.

There are many applications of Gaussian process regression in the real world. For example, in spatial statistics and geostatistics, researchers have used the same model for some years under the name of *kriging* (Diggle et al., 2003). In addition, many regression and classification problems in machine learning have been successfully treated by GPR (Williams and Rasmussen, 2006). Moreover, some covariance functions are not restricted to only univariate predictor variables, and so, Gaussian process regression can naturally be applied to high-dimensional problems.

In the last two decades, functional data analysis (FDA) has become an attractive research field in statistics. On the one hand, functional data can be considered as realisations $\{X_1(t), X_2(t), ..., X_n(t)\}$ of a stochastic process $X(t)$ in a functional space, where n is the sample size. In addition, the stochastic process $X(t)$ is usually considered in $L^2[a, b]$, which means $E[\int_a^b X^2(t)dt] < \infty$. On the other hand, functional data can also be viewed as realisations $\{X_1(t), X_2(t), ..., X_n(t)\}$ of a stochastic process $X(t)$ on a set T, such as the time set $[T_1, T_2]$. The former is often applied for theoretical analysis and modelling, while the latter is more consistent with the actual observed data in the real world. In addition, functional data exist in many practical problems. For example, $X_i(t), i = 1, ..., n$, can represent the temperature at time t in year i, the pollution index at time t in place i, the spectral absorption rate of substance i where t refers to wavelength, the load of servers i at time t and so on. More complex functional data include brain data and neuroimaging data, which is one of the latest researches in neuroscience.

One essential tool in FDA is functional regression model which aims to capture the relationship between response variables and predictor variables. Both response variables and predictor variables could be functional or scalar. We focus on functional response variables with both functional and scalar covariates in this thesis. In a random function $X(t)$, $X(t_1)$ and $X(t_2)$ will not generally be independent for $t_1 \neq t_2$, and so we consider mean structure and covariance structure simultaneously.

There is a challenge for implementation and theory in FDA since the random variable is in the form a function which is intrinsically infinite-dimensional. On the other hand, functional data brings information in abundance. Measurement errors of accuracy can be viewed as a random noise around a smooth function which often contaminates observed data. When replicated measurements for a functional response for a subject can be made, measurement error can be accommodated (Wang et al., 2016).

There are many extensions of statistical methods to functional data, such as regression, classification and clustering with a wide class of applications. For example, Abraham et al. (2003), Serban and Wasserman (2005) extend k-means algorithm in FDA via B-spline basis functions and Fourier basis functions respectively. Shi and Wang (2008) propose Gaussian process functional regression (GPFR) for curve prediction and clustering. Müller et al. (2005) introduce a functional classification model based on functional binomial regres-

sion. McLean et al. (2014) provide a functional generalized additive model for regression problems in which the smooth independent functions are bivariate.

The traditional applications of Gaussian process regression model are restricted by some assumptions, such as the response variable following a normal distribution. Moreover, the curves in FDA typically take values in a Euclidean space. However, many observed data in the real world do not satisfy such assumptions.

To address this issue, Wang and Shi (2014) introduce a generalized Gaussian process functional regression for data which follow an exponential family distribution. In addition, Dai et al. (2018) propose functional principal component analysis on Riemannian manifolds, especially on spheres, which is carried out by the Riemannian logarithm map. Gaussian process regression is extended to Riemannian manifolds by an exponential wrapping function (Mallasto and Feragen, 2018). In this thesis, we further develop the model in Wang and Shi (2014) by using an ABC algorithm for prediction in Section 3.3. We also extend the model for multi-variate response variables. In Section 4.2.1, we combine the ideas in Mallasto and Feragen (2018) and Wang and Shi (2014) to give a concurrent regression model for functional data on Riemannian manifolds.

1.3 Statistical analysis on Riemannian manifolds

Regression is an ubiquitous and powerful tool in data analysis which reveals the relationship between predictor variables and response variables. Most well known regression models require an underlying linear space structure, such as Gaussian process regression (Shi and Choi, 2011), Bayesian lasso regression (Hans, 2009) and support vector regression (Smola and Schölkopf, 2004). However, statistical models for manifold-valued data are gaining popularity in various fields of scientific analysis, such as computational social science, medical imaging analysis and computer vision (Bronstein et al., 2017). As a result, it is desirable to develop methods which capture the complicated relation between predictors and response variables by regression models in non-Euclidean spaces, such as smooth Riemannian manifolds.

A popular topic in statistical analysis on Riemannian manifold is *manifold learning* or *subspace learning*. If the dimension of covariates is very large, ranging from thousands

to millions, it is important to reduce the dimension to avoid the *curse of dimensionality*. For example, Guhaniyogi and Dunson (2016) introduce compressed Gaussian processes which combine the low-rank Gaussian process and random feature compression. Calandra et al. (2016) develop manifold Gaussian process regression. In their model, a neural network is used to learn the low-dimensional latent feature space, and then the relationship between the low-dimensional latent feature space and response variables is modelled by a Gaussian process regression model. Landmarking with Gaussian processes is another essential method to landmark a manifold which transforms the objective function by active learning in Gaussian process. In addition, Moon and Pavlovic (2008) propose a dimension reduction technique by reformulating kernel setting which is manifold kernel dimensionality reduction in Gaussian process. In addition, Chen and Müller (2012) provide a manifold learning procedure based on nonlinear dimension reduction method for functional data.

In this thesis, we mainly focus on another compelling research field which is regression models for manifold-valued data. The lack of vector structure makes even straight forward issues such as computing a sample mean, much more challenging. For instance, if we apply an Euclidean statistical method to non-Euclidean data, the result might not even lie in the target space. Hence, sophisticated techniques for data analysis on Riemannian manifolds are needed. Geodesic regression (Fletcher, 2013) is a generalization of linear regression in which the univariate predictor variables are considered in $\mathbb{R}$ and the response variables take values on a Riemannian manifold. The author also derives a gradient descent algorithm to fit the model via derivatives of exponential map and Jacobi fields, since the least squares method has no analytical solution under this setting. Kim et al. (2014) extend geodesic regression to the multivariate case in which the predictor variables are in $\mathbb{R}^n$ and the response variables are still manifold-valued. In addition, the authors propose a variational gradient descent method based on parallel transport which is more convenient for high-dimensional input. Moreover, Cornea et al. (2017) introduce an intrinsic geodesic regression for typical manifold-valued response variables lying in a Riemannian symmetric space, with multiple covariates within Euclidean space. The authors also develop several methods to estimate the coefficients in this model. Multiple linear regression is generalised by Petersen and Müller (2019) for complex random objects in metric spaces and the authors develop local linear or polynomial regression as well. To the analysis of varying shapes,

Nava-Yazdani et al. (2020) provide an approach to decompose the tangent vectors into vertical and horizontal vectors, which can be applied to geodesic regression in Kendall's shape space efficiently. In addition, Lin and Yao (2020) study functional regression for scalar response variables and non-Euclidean predictors from an unknown manifold.

However, in the real world, non-Euclidean data do not always satisfy the assumption of a linear relationship. Pelletier (2006) proposes a non-parametric regression approach for response variables in Euclidean space and covariates in a closed Riemannian manifold by extending kernel regression estimation. For trajectory analysis on Riemannian manifolds, Su et al. (2014) provide a framework which registers automatically and promotes the analysis for time-warped trajectory via transported squared-root vector field and and $\mathbb{L}^2$ norm between these fields. Furthermore, Banerjee et al. (2016) introduce a kernel-based nonlinear regression model for both manifold-valued independent variables and manifold-valued dependent variables. In addition, Hinkle et al. (2012) extend polynomial regression on Riemannian manifolds. Cornea et al. (2017) develop a regression model for manifold-valued response variables in a Riemannian symmetric space and covariates in Euclidean space with applications in medical imaging analysis. Nevertheless, these models are restricted due to the lack of a probabilistic framework.

To break this limitation, Pigoli et al. (2016) introduce an additive linear model for manifold-valued data which is based on the exponential map. In particular, they transform the manifold-valued data into tangent spaces and then estimate the parameters of the additive linear model by the generalised least squares methods. Moreover, kriging prediction is also generalized into manifolds therein. Lin et al. (2018) discuss an extrinsic Gaussian process for regression and classification on manifolds in which the covariates have a manifold-valued structure and the response variables belong to Euclidean space. Additionally, Mallasto and Feragen (2018) propose a wrapped Gaussian process regression which is another non-parametric model with a probabilistic framework by linearizing the Riemannian manifold to a tangent space via the inverse exponential map. Moreover, the computation of wrapped Gaussian process regression is relatively cheap since the only difference between Gaussian process regression and wrapped Gaussian process regression is to calculate exponential map and logarithm map for each manifold-valued data point.

Other statistical analyses for manifold-valued data have been developed, especially

variants of principal component analysis (PCA). For example, Fletcher et al. (2004) introduce principal geodesic analysis which is a generalisation of principal component analysis on Riemannian manifolds by requiring the geodesic principal components to pass through the intrinsic mean, and calculate them approximately in Euclidean space. In addition, Huckemann et al. (2010) propose a method to obtain sample geodesic principal component for general quotients with computation on Riemannian manifolds directly rather than in Euclidean space. Another extension of principal component analysis for nested spheres is provided by Jung et al. (2012) via a decomposition method. In addition, Lila et al. (2016) propose a PCA technique for functional data on 2-dimensional Riemannian manifolds and the authors adopt a regularization method by a smoothing penalty coherent with geodesic distances. Moreover, Dai et al. (2018) formulate Riemannian functional principal component analysis by mapping data to the tangent space at the intrinsic Fréchet mean function, and then performing multivariate functional principal component analysis within that tangent space. Nonetheless, some manifolds cannot be embedded in Euclidean space in a natural way, such as $p \times p$ SPD matrices. In order to analyse functional data on more general Riemannian manifolds, Lin et al. (2019) introduce the theory to map manifold-valued data to a tensor Hilbert space instead of an ambient Euclidean space. They also focus on the foundational theory for functional data on Riemannian manifolds which paves a way to develop functional linear regression for manifold-valued data and intrinsic Riemannian functional principal component analysis. On the other hand, from a Bayesian perspective, Bhattacharya and Dunson (2010) extend non-parametric Bayesian methods and theory, including the Gibbs sampler for posterior computation, on manifolds and compact metric spaces. For some other topics, Le (2001) calculates a Fréchet mean of probability measures by using Jacobi fields and the contraction mapping theorem on locally symmetric Riemannian manifolds. Additionally, a gradient descent algorithm, named Competitive Learning Riemannian Quantization, to approximate a probability measure over Riemannian manifold is introduced (Le Brigant and Puechmorel, 2019). Moreover, Chakraborty et al. (2019) present a sampling algorithm for the normal distribution and computation of Fréchet mean recursively on Stiefel manifolds which are Riemannian homogeneous but not symmetric.

To the best of our knowledge, there is little literature about regression models for

the functional batch data on Riemannian manifolds within a probabilistic framework. Therefore, we provide a novel scheme for nonlinear non-parametric regression with uncertainty for batch data on smooth Riemannian manifolds. In this thesis, we construct this regression model by estimating mean structure and covariance structure simultaneously. Specifically, a functional regression model is used for mean structure with functional coefficients and scalar covariates, while a Gaussian process is used for covariance structure with functional predictor variables. We also assume the existence and uniqueness of a Fréchet population mean function for random curves on Riemannian manifold, as assumption B2 in Dai et al. (2018). In order to guarantee the exponential map and the inverse exponential map exist almost everywhere on a Riemannian manifold, we assume the injectivity radius of the Riemannian manifold is large enough [1]. For batch data, which is also called group data, we introduce a variation of WGPFR where all curves in a batch share a common mean structure. In addition, the wrapped Gaussian process functional regression with common mean structure plays an essential role in data analysis especially in long-term forecasting of the manifold-valued stochastic process. When the test data are distant from the training data, the output mainly depends on the mean structure. Therefore, the estimation of common mean structure determines the extrapolating accuracy.

1.4 Contributions and outlines of thesis

There are two main contributions in this thesis. One is to extend Gaussian process regression model for data which follow a truncated normal distribution or Gamma distribution. The second is a wrapped Gaussian process functional regression model for batch data on Riemannian manifolds. The novel methodologies, inference schemes and simulation studies are discussed in Chapter 3, Chapter 4 and Chapter 5.

The thesis is organised as follows.

In Chapter 2, we review the definition and inference of the Gaussian process regression model, convolved Gaussian process and Gaussian process functional regression model. This chapter also contains mathematical concepts, computational tools and notation for smooth Riemannian manifolds. In addition, a brief summary of some special probability

[1] The definition of injectivity radius is discussed in Section 2.4.1

distributions, such as the truncated normal distribution and wrapped Gaussian distribution, are presented. We discuss the differential evolution algorithm (DE) and approximate Bayesian computation (ABC) which are applied in hyper-parameter estimation and prediction.

In Chapter 3, we study Gaussian process regression for non-Gaussian data which follow a truncated normal distribution. Basically, we consider GPR as a latent process and the prediction can be calculated by Gibbs sampling. Analogously, in Gaussian process regression for Gamma-distributed data, GPR also plays a role as a latent process. The analytical forms of estimation and prediction are derived based on Gaussian approximation of the full conditional density $p(\boldsymbol{f}|\boldsymbol{y}, k, \boldsymbol{\theta})$ in Equation (3.18). However, since the Gamma distribution is not symmetric, the Gaussian approximation brings an error leading to inaccurate predictions. In prediction part, we use an approximate Bayesian computation method instead, since ABC is likelihood-free which avoids the influence from the symmetric property of the Gaussian approximation of the full conditional distribution whose real distribution is non-symmetric. In addition, the GPR for Gamma-distributed data is extended to high-dimensional data by a convolved Gaussian process.

In Chapter 4, a wrapped Gaussian process functional regression for manifold-valued functional batch data is introduced. We use a functional linear regression to model the tangent vector functions from the intrinsic Fréchet mean function to the functional data points, which is referred to as the mean structure. Then, we use a Gaussian process to model the tangent vector functions from the mean structure to the functional data, which is called the covariance structure. We propose an efficient iterative algorithm to update the mean structure and covariance structure. The mean structure can be updated and the covariance structure can be estimated iteratively until some convergence conditions have been satisfied. In Chapter 5, we present some numerical experiments, on 2-sphere and Kendall's shape space, for this model compared to other models, which shows that our model outperforms others. An application for flight trajectory data is also presented. Additionally, the sensitivity of wrapped Gaussian process functional regression model with various kernels is studied, which leads to the predictions of our model are accurate regardless of the choice of kernel.

In Chapter 6, we conclude this thesis and discuss future work.

Chapter 2

Preliminaries

2.1 Gaussian process regression

A Gaussian process is a special stochastic process in which any finite collection follows a multi-variate Gaussian distribution. Mean vector and covariance matrix specify a multi-variate Gaussian distribution, thus, mean function and covariance function (as known as *kernel*) determine a Gaussian process. In practice, people usually suppose the mean function of a GP is zero. However, we can choose a non-zero mean function based on specific context. For example, Shi et al. (2007), Wang and Shi (2014) use functional linear regression model as the mean function. The main challenge in Gaussian process regression or Gaussian process classification is to estimate the *hyper-parameters* in kernel, which is often denoted as $\boldsymbol{\theta}$. In this section, we firstly review Gaussian process regression model under *weight-space* and *function-space*. Afterwards, we discuss some methods to estimate the hyper-parameters in kernel. In addition, some other attractive and useful research fields on Gaussian process regression model are also discussed briefly, such as the selection of covariance functions and Gaussian process latent variable models.

2.1.1 Weight-space view

The linear regression model is explored and used widely in which the response variable can be represented as a linear combination of predictor variables (Montgomery et al., 2021). We start from considering the linear regression model under Bayesian framework (Bolstad and Curran, 2016). Suppose the training data set is denoted as $\mathcal{D} = \{\boldsymbol{x}_i, y_i | \boldsymbol{x}_i \in$

$\mathbb{R}^p, y_i \in \mathbb{R}, i = 1, ..., n\}$, where $\boldsymbol{x}_i$ refers to independent variables, y_i refers to a dependent variable and n refers to the number of data points. Our target is to derive the conditional distribution of response variable given covariates. In addition, a linear regression model with Gaussian white noise is

$$y_i = f(\boldsymbol{x}_i) + \epsilon, \;\; f(\boldsymbol{x}_i) = \boldsymbol{x}_i^T \boldsymbol{w} \tag{2.1}$$

where $\boldsymbol{w}$ is a vector of coefficients. The error between function $f(\boldsymbol{x}_i)$ and response variable y_i is defined by independent and identically distributed normal distribution with zero mean and covariance σ^2, which is denoted as

$$\epsilon \sim \mathcal{N}(0, \sigma^2)$$

where $\mathcal{N}(\mu, \sigma^2)$ refers to a normal distribution with mean μ and variance σ^2.

Therefore, the likelihood function can be obtained directly

$$\begin{aligned} p(\boldsymbol{y}|X, \boldsymbol{w}) &= \prod_{i=1}^{n} p(y_i|\boldsymbol{x}_i, \boldsymbol{w}) \\ &= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} \exp\big(-\frac{(y_i - \boldsymbol{x}_i^T \boldsymbol{w})^2}{2\sigma^2}\big) \\ &= \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\big(-\frac{|\boldsymbol{y} - X^T \boldsymbol{w}|^2}{2\sigma^2}\big) \\ &\sim \mathcal{N}(X^T \boldsymbol{w}, \sigma^2 \boldsymbol{I}) \end{aligned} \tag{2.2}$$

where $|\cdot|$ refers to Euclidean norm $X = (\boldsymbol{x}_1^T, ..., \boldsymbol{x}_n^T)^T$ and $\boldsymbol{y} = (y_1, ..., y_n)^T$.

The calculation of the posterior distribution over coefficient vectors in Bayesian linear regression model is based on the Bayes' rule, which is

$$\text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{marginal likelihood}}. \tag{2.3}$$

Since the marginal likelihood in Equation (2.3) is independent of weights, we can

formulate and simplify the Bayes' rule as

$$\begin{aligned} p(\boldsymbol{w}|X,\boldsymbol{y}) &= \frac{p(\boldsymbol{y}|\boldsymbol{w},X)p(\boldsymbol{w})}{p(\boldsymbol{y}|X)} \\ &= \frac{p(\boldsymbol{y}|\boldsymbol{w},X)p(\boldsymbol{w})}{\int p(\boldsymbol{y}|\boldsymbol{w},X)p(\boldsymbol{w})d\boldsymbol{w}} \\ &\propto p(\boldsymbol{y}|\boldsymbol{w},X)p(\boldsymbol{w}). \end{aligned} \tag{2.4}$$

Suppose the prior distribution of coefficients vector $\boldsymbol{w}$ is a multi-variate Gaussian distribution with zero mean and covariance matrix Σ, i.e. $\boldsymbol{w} \sim \mathcal{N}(\mathbf{0}, \Sigma)$. Therefore, the posterior distribution of coefficients vector $\boldsymbol{w}$ is

$$\begin{aligned} p(\boldsymbol{w}|X,\boldsymbol{y}) &\propto \exp\big(-\frac{1}{2\sigma^2}(\boldsymbol{y}-X^T\boldsymbol{w})^T(\boldsymbol{y}-X^T\boldsymbol{w})\big)\exp\big(-\frac{1}{2}\boldsymbol{w}^T\Sigma^{-1}\boldsymbol{w}\big) \\ &\propto \exp\big(-\frac{1}{2}(\boldsymbol{w}-\boldsymbol{\mu})^T(\frac{1}{\sigma^2}XX^T+\Sigma^{-1})(\boldsymbol{w}-\boldsymbol{\mu})\big) \end{aligned} \tag{2.5}$$

where $\boldsymbol{\mu} = \sigma^{-2}(\sigma^{-2}XX^T + \Sigma^{-1})^{-1}X\boldsymbol{y}$. By observing Equation (2.5), we can recognize that the posterior distribution follows a multi-variate Gaussian distribution. Specifically, we have

$$p(\boldsymbol{w}|X,\boldsymbol{y}) = \mathcal{N}\big(\sigma^{-2}(\sigma^{-2}XX^T+\Sigma^{-1})^{-1}X\boldsymbol{y}, (\sigma^{-2}XX^T+\Sigma^{-1})^{-1}\big). \tag{2.6}$$

For prediction, we integrate out all possible parameters by their posterior distribution which is the crucial difference between Bayesian community and frequentist community. The predictive distribution of a new input $\boldsymbol{x}^*$ in the Bayesian context with respect to the Gaussian posterior is given by

$$\begin{aligned} p(f(\boldsymbol{x}^*)|X,\boldsymbol{y},\boldsymbol{x}^*) &= \int p(f(\boldsymbol{x}^*)|\boldsymbol{x}^*,\boldsymbol{w})p(\boldsymbol{w}|X,\boldsymbol{y})d\boldsymbol{w} \\ &= \mathcal{N}\big(\frac{1}{\sigma^2}\boldsymbol{x}^*(\sigma^{-2}XX^T+\Sigma^{-1})^{-1}X\boldsymbol{y}, \boldsymbol{x}^*(\sigma^{-2}XX^T+\Sigma^{-1})^{-1}\boldsymbol{x}^*\big) \end{aligned} \tag{2.7}$$

As mentioned previously, the linear regression model learns the linear relationship between predictor variables and response variables which is a restriction, since the real relationship might be non-linear. To break this limitation, the predictor variables can be projected into a high dimensional feature space by basis functions, such as B-splines (Brigger et al., 2000) or power series (Niven, 1969), and in that space, the relationship

might be linear. Besides, we model the mapped data rather than the original data.

We now consider the Bayesian linear regression model in the context of high-dimensional feature space. Suppose the projected independent variables are referred to $\phi(\boldsymbol{x})$ and the corresponding design matrix becomes $\Phi(X)$. The model with noise is

$$y = f(\boldsymbol{x}) + \epsilon = \phi(\boldsymbol{x})^T \boldsymbol{w} + \epsilon. \tag{2.8}$$

Moreover, the predictive distribution is

$$\begin{aligned} p(f(\boldsymbol{x}^*)|X, \boldsymbol{y}, \boldsymbol{x}^*) \sim \mathcal{N}\big(&\frac{1}{\sigma^2}\phi(\boldsymbol{x}^*)(\sigma^{-2}\Phi(X)\Phi(X)^T + \Sigma^{-1})^{-1}\Phi(X)\boldsymbol{y}, \\ &\phi(\boldsymbol{x}^*)(\sigma^{-2}\Phi(X)\Phi(X)^T + \Sigma^{-1})^{-1}\phi(\boldsymbol{x}^*)\big) \end{aligned} \tag{2.9}$$

The Equation (2.9) can be rewritten as

$$\begin{aligned} p(f(\boldsymbol{x}^*)|X, \boldsymbol{y}, \boldsymbol{x}^*) \sim \mathcal{N}\big(&\phi(\boldsymbol{x}^*)^T\Sigma\Phi(X)(K + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{y}, \\ &\phi(\boldsymbol{x}^*)^T\Sigma\phi(\boldsymbol{x}^*) - \phi(\boldsymbol{x}^*)^T\Sigma\Phi(X)(K + \sigma^2\boldsymbol{I})^{-1}\Phi(X)^T\Sigma\phi(\boldsymbol{x}^*)\big) \end{aligned} \tag{2.10}$$

where $K = \Phi(X)^T\Sigma\Phi(X)$. Williams and Rasmussen (2006) show Equation (2.9) and Equation (2.10) are equivalent. The feature space enters in the form of $\phi(\boldsymbol{x}^*)\Sigma\Phi(X)$, $\phi(\boldsymbol{x}^*)^T\Sigma\phi(\boldsymbol{x}^*)$ and $\Phi(X)^T\Sigma\phi(\boldsymbol{x}^*)$ whose entries are invariably of $\phi(\boldsymbol{x})^T\Sigma\phi(\boldsymbol{x}')$. In addition, $\boldsymbol{x}$ and $\boldsymbol{x}'$ could either in the training data or the test data.

The kernel is defined as $k(\boldsymbol{x}, \boldsymbol{x}') = \phi(\boldsymbol{x})^T\Sigma\phi(\boldsymbol{x}')$ which is an inner product with respect to Σ. Since Σ is a positive definite matrix, we can define $(\Sigma^{1/2})^2 = \Sigma$ by applying singular value decomposition. Suppose $\psi(\boldsymbol{x}) = \Sigma^{1/2}\phi(\boldsymbol{x})$, the kernel can be represented by a dot product that is

$$k(\boldsymbol{x}, \boldsymbol{x}') = \psi(\boldsymbol{x}) \cdot \psi(\boldsymbol{x}'). \tag{2.11}$$

We briefly reviewed an explanation of Gaussian process regression by Bayesian linear regression, which is helpful for us to understand GPR in weight-space. Additionally, another explanation (in functional space) of GPR will be reviewed in the next part and the inferences of our models are derived based on it since we study the functional data.

2.1.2 Function-space view

In this section, we discuss GPR under the view of function space which is more related to our research in Chapter 3 and Chapter 4. We can derive the inference in function space directly that is equivalent to the analysis in the context of weight-space view. A Gaussian process $f(\cdot)$ is completely determined by mean function and covariance function

$$\mu(\boldsymbol{x}) = E[f(\boldsymbol{x})]$$
$$k(\boldsymbol{x}, \boldsymbol{x}') = E[(f(\boldsymbol{x}) - \mu(\boldsymbol{x}))(f(\boldsymbol{x}') - \mu(\boldsymbol{x}'))] = \mathrm{Cov}(f(\boldsymbol{x}), f(\boldsymbol{x}'))$$

Then, we can denote

$$f(\boldsymbol{x}) \sim \mathcal{GP}(\mu(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')). \tag{2.12}$$

Connecting to Section 2.1.1 in which the Bayesian linear regression model is assumed as $f(\boldsymbol{x}) = \phi(\boldsymbol{x})^T \boldsymbol{w}$ and $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \Sigma)$, we have

$$E[f(\boldsymbol{x})] = E[\phi(\boldsymbol{x})^T \boldsymbol{w}] = \phi(x)^T E[\boldsymbol{w}] = \boldsymbol{0}$$
$$E[f(\boldsymbol{x}) f(\boldsymbol{x}')] = \phi(\boldsymbol{x})^T E[\boldsymbol{w}\boldsymbol{w}^T]\phi(\boldsymbol{x}') = \phi(\boldsymbol{x})\Sigma\phi(\boldsymbol{x}') = k(\boldsymbol{x}, \boldsymbol{x}').$$

Suppose the training data is still $\mathcal{D} = \{\boldsymbol{x}_i, y_i | i = 1, , , ., n\}$, we have a multi-variate Gaussian distribution that is

$$\big(f(\boldsymbol{x}_1), ..., f(\boldsymbol{x}_n)\big) \sim \mathcal{N}(\boldsymbol{\mu}, K) \tag{2.13}$$

where $\boldsymbol{\mu} = (\mu(\boldsymbol{x}_1), ..., \mu(\boldsymbol{x_n}))^T$ and K is a $n \times n$ covariance matrix and the entry of the (i, j)-th element is $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$. To obtain the predictive distribution of a new input $\boldsymbol{x}^*$, it is convenient to the take advantage of properties of joint Gaussian distribution (Ahrendt, 2005)

$$\begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{f}(\boldsymbol{x}^*) \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu}(\boldsymbol{x}^*) \end{pmatrix}, \begin{pmatrix} K + \sigma^2 \boldsymbol{I} & K(\boldsymbol{x}^*, X)^T \\ K(\boldsymbol{x}^*, X) & k(\boldsymbol{x}^*, \boldsymbol{x}^*) \end{pmatrix} \right) \tag{2.14}$$

where $X = (\boldsymbol{x}_1, ..., \boldsymbol{x}_n)$, $\boldsymbol{y} = (y_1, ..., y_n)^T$ and $K(\boldsymbol{x}^*, X) = (k(\boldsymbol{x}^*, \boldsymbol{x}_1), ..., k(\boldsymbol{x}^*, \boldsymbol{x}_n))$ is a

$1 \times n$ covariance vector. The predictive distribution has an analytical form which is

$$\begin{aligned} p(\boldsymbol{f}(\boldsymbol{x}^*)|\mathcal{D}) \sim \mathcal{N}(K(\boldsymbol{x}^*, X)^T(K + \sigma^2\boldsymbol{I})^{-1}(\boldsymbol{y} - \boldsymbol{\mu}) + \boldsymbol{\mu}(\boldsymbol{x}^*), \\ k(\boldsymbol{x}^*, \boldsymbol{x}^*) - K(\boldsymbol{x}^*, X)^T(K + \sigma^2\boldsymbol{I})^{-1}K(\boldsymbol{x}^*, X)) \end{aligned} \tag{2.15}$$

When we compare Equation (2.15) to Equation (2.10), we can see that they are equivalent and the only difference is that the representation of lower dimensional feature space. Specifically, a kernel function is used in the former while the product of covariance vector and covariance matrix is used in the latter.

2.1.3 Estimation of hyper-parameters

There are many prediction problems in this thesis, such as in Section 2.2. In addition, the prediction of Gaussian process regression model depends on covariance function when the mean function is fixed or pre-trained and the value of covariance function depends on hyper-parameters. In this section, we discuss how to estimate hyper-parameters by empirical Bayesian approach.

The values of hyper-parameters $\boldsymbol{\theta}$ control shape and variability of regression curve. However, in practice, it is difficult to explain the physical meaning between hyper-parameters and data clearly. Misspecified estimation may lead to poor predictions especially for small sample size. Therefore, we use empirical Bayesian approach to estimate the hyper-parameters $\boldsymbol{\theta}$ by observed data rather than calculate the values directly.

The marginal distribution for response variables $\boldsymbol{y}$ given prior hyper-parameters $\boldsymbol{\theta}$ is

$$p(\boldsymbol{y}|\boldsymbol{\theta}) = \int p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{\theta})d\boldsymbol{f} \tag{2.16}$$

where $\boldsymbol{y}|\boldsymbol{f} = \prod_{i=1}^{n}\mathcal{N}(f(\boldsymbol{x}_i), \sigma^2)$ and $\boldsymbol{f}|\boldsymbol{\theta} = \mathcal{N}(\boldsymbol{0}, K)$. In addition, K is a $n \times n$ covariance matrix computed by covariance function with hyper-parameters $\boldsymbol{\theta}$. According to the definition of regression model with additive noise, the marginal distribution of $\boldsymbol{y}$ is $\mathcal{N}(\boldsymbol{0}, K + \sigma^2\boldsymbol{I}) = \mathcal{N}(\boldsymbol{0}, \Psi)$. Thus, the marginal log-likelihood function of $\boldsymbol{\theta}$ is

$$l(\boldsymbol{\theta}|X, \boldsymbol{y}) = -\frac{1}{2}\log(|\Psi(\boldsymbol{\theta})|) - \frac{1}{2}\boldsymbol{y}^T\Psi(\boldsymbol{\theta})^{-1}\boldsymbol{y} - \frac{n}{2}\log(2\pi) \tag{2.17}$$

and its derivative with respect to each hyper-parameter is given as

$$\frac{\partial l}{\partial \theta_j} = \frac{1}{2}\mathrm{tr}\big((\Psi^{-1}\boldsymbol{y}(\Psi^{-1}\boldsymbol{y})^T - \Psi^{-1})\frac{\partial \Psi}{\partial \theta_j}\big) \tag{2.18}$$

where $\mathrm{tr}(\cdot)$ refers to trace of a matrix. The second derivative is given by

$$\frac{\partial^2 l}{\partial \theta_i \partial \theta_j} = \frac{1}{2}\big((\Psi^{-1}\boldsymbol{y}(\Psi^{-1}\boldsymbol{y})^T - \Psi^{-1})(\frac{\partial^2 \Psi}{\partial \theta_i \partial \theta_j} - \frac{\partial \Psi}{\partial \theta_i}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_j}) - (\Psi^{-1}\boldsymbol{y}(\Psi^{-1}\boldsymbol{y})^T\frac{\partial \Psi}{\partial \theta_i}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_j}\big) \tag{2.19}$$

The hyper-parameters $\boldsymbol{\theta}$ can be estimated by maximising the marginal log-likelihood function (2.17) based on some iterative optimisation methods, such as the conjugate gradient descent algorithm (Press et al., 2007) by using the first order gradient in Equation (2.18) and the second order gradient (2.19). In Section 3.1.3, we use this method and these gradients to estimate the hyper-parameters of a GPR. Another approach for estimation is based on Markov Chain Monte Carlo (MCMC) when we know little knowledge about the initial values. The details of the MCMC algorithm and efficient implementation can be found in Chapter 3 of Shi and Choi (2011).

In Section 3.1, we used the minimization approach above, but in Section 3.3, a variant of genetic algorithm is used since the marginal likelihood function is different from (2.17).

2.1.4 Different covariance functions

In GPR, the hyper-parameters in kernel is similar to smoothing parameters in a spline model (Galway, 1992) and there are a variety of choices for covariance function. Based on the linearity, we can classify them as *linear covariance function* such as

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sum_{i=1}^{p} a_i x_i x_i'$$

and *nonlinear covariance function* such as *squared exponential covariance function*

$$k(\boldsymbol{x}, \boldsymbol{x}') = v_0 \exp\,(-\frac{1}{2}\sum_{i=1}^{p} w_i (x_i - x_i')^2)$$

where the hyper-parameters of linear covariance function and non-linear covariance function are $(a_1, ..., a_p)$ and $(v_0, w_1, ..., w_p)$ respectively. The linear covariance function is often used as a component of a covariance function which is a sum of a few different covariance function, such as the covariance function in Equation (2.20). In addition, the length scale of each covariate can be measured by the inverse of w_i. If the estimated value of w_i, i.e., $\hat{w}_i$, is extremely small, it suggests that x_i has little contribution to the covariance function. Therefore, this covariate x_i might be removed from the data set.

On the other hand, based on the stationarity (the kernel only depends on the difference between covariates, i.e. $k(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}-\boldsymbol{x}')$), we can classify them as *stationary covariance function* such as *powered exponential covariance function*

$$k(\boldsymbol{x}, \boldsymbol{x}') = v_0 \exp\left(-w\|\boldsymbol{x} - \boldsymbol{x}'\|^{\gamma}\right)$$

and *non-stationary covariance function* such as linear covariance function.

Moreover, *rational quadratic covariance function*

$$k(\boldsymbol{x}, \boldsymbol{x}') = (1 + s_{\alpha} w\|\boldsymbol{x} - \boldsymbol{x}'\|^2)^{-\alpha}$$

where $\alpha, w \leq 0$ and *Matérn covariance function*

$$k(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{\Gamma(v)2^{v-1}}(w\|\boldsymbol{x} - \boldsymbol{x}'\|)^{v} K_v(w\|\boldsymbol{x} - \boldsymbol{x}'\|)$$

where $w \leq 0$ and K_v is a modified Bessel function of order v, are also often used. More examples and properties of different covariance functions can be found in Abrahamsen (1997) and MacKay and Gibbs (1997).

A popular choice for GPR is

$$k(t_i, t_j) = v_0 \exp(-\frac{1}{2w_0}(t_i - t_j)^2) + a_0 + a_1 t_i t_j + \sigma^2 \delta_{ij} \tag{2.20}$$

where δ_{ij} is the Kronecker delta. Since GPR is actually a non-parametric model, it is not very sensitive to different covariance functions and the covariance function (2.20) can be used in most problems (Shi et al., 2007).

One of differences among the covariance functions above is the distance function between covariates. For example, people use weighted sum of squared differences in squared exponential convariance function and powered norm in powered exponential covariance function. In addition, the hyper-parameters control the smoothness of a Gaussian process. Therefore, people can select a suitable covariance function based on their knowledge about the distance function between covariates and the smoothness of a Gaussian process. In other words, the properties of different covariance functions can be used as a guideline of selection, especially for low-dimensional covariates $\boldsymbol{x}$. For properties of different kernels, readers can see Shi and Choi (2011) and Adler and Taylor (2009). If the dimension of covarites $\boldsymbol{x}$ is large (more than 2), we can use Bayes factor which is a method for model selection under the view of Bayesian statistics. Based on the ratio of two marginal distributions, the covariance function with higher Bayes factor can be considered as a choice. However, in practice, the marginal distributions are often not analytically tractable except for some simple regression models, such as linear regression model. To address this issue, researchers develop some Monte Carlo methods. For instance, Meng and Wong (1996) introduce a method to calculate the marginalizing constants which is named Bridge sampling. Additionally, Chib (1995) propose Chib's approximation to compute the marginal distribution approximately.

As a non-parametric regression model, GPR should be not sensitive to different covariance functions and in Chapter 5, we compared the performances of different kernels, such as squared exponential and rational quadratic, in wrapped Gaussian process function regression model.

2.1.5 Some other topics

People are often unable to observe some potential variables or processes in the real world. Therefore, models for latent variables are created to catch up the relationship between observed manifest variables and unobserved latent variables, such as the latent class analysis model and item response theory (Lee, 2007). In the extension of GPR, Lawrence and

Hyvärinen (2005) introduce Gaussian process latent variable model which is defined by

$$y(t) = f(z(t)) + \epsilon(t)$$
$$f(z(t)) \sim GPR(0, k(\boldsymbol{\theta})|z(t))$$

where y refers to manifest variable, z refers to latent variable and ϵ refers to Gaussian white noise. The latent variable $z(t)$ is assumed to follow Gaussian prior. In addition, some statistical inferences such as estimation and prediction can be derived by active set and sparse greedy approximation[1].

The prediction of Gaussian process regression is often accurate when the response variable is normally distributed. However, in practice, many data follow non-Gaussian distribution, such as binary distribution or Poisson distribution. Wang and Shi (2014) introduce a generalized Gaussian process regression model for non-Gaussian functional data. In this paper, the authors model the input of link function for response variable via a concurrent regression model in which the mean model is structured by a functional linear regression and the covariance model is structured by GPR with Gaussian process prior, i.e.

$$E(y(t)|\tau(t)) = h(\mu(t) + \tau(t))$$
$$\tau(t) = \tau(\boldsymbol{x}(t)) \sim GPR(0, k(\cdot, \cdot; \boldsymbol{\theta}))$$

where $h(\cdot)^{-1}$ is a given link function, $\mu(t)$ denotes functional linear regression model, $\tau(t)$ denotes latent variable, $\boldsymbol{x}(t)$ denotes functional covariates. This model is efficient for some data from exponential family (particularly for some symmetrically distributed data).

There is a possibility that different batch has different features, such as smoothness. Moreover, if the subjects and measuring equipment are different, heterogeneity might exist among batches. Hierarchical mixture Gaussian process regression model (Shi et al., 2005) is proposed to solve this problem in which the model is considered as weighted sum of a few Gaussian processes with identical covariance function but different values of hyper-

[1]See Section 3.3.2 for detail about active set and sparse greedy approximation.

parameters

$$y = f(\boldsymbol{x}) + \epsilon$$
$$f(\boldsymbol{x}) \sim \sum_{k=1}^{K} \pi_k GP(\boldsymbol{\theta}_k)$$

where these weights $(\pi_1, ..., \pi_K)$ are assumed following Dirichlet distribution

$$p(\pi_1, ..., \pi_K) \sim D(\delta, .., \delta)$$

and the hyper-parameters can be trained by a hybrid Markov Chain Monte Chain algorithm (see more details in Appendix in Shi et al. (2005)).

We further develop the Gaussian process regression for non-Gaussian data, especially for data following truncated Gaussian distribution and Gamma distribution in Chapter 3 and functional batch data on Riemannian manifolds in Chapter 4.

2.2 Convolved Gaussian process

In practice, the response variables are usually multi-dimensional. However, the response variable is normally assumed to be one-dimensional in Gaussian process regression. In Section 3.3, we introduce a regression model for gamma-distributed data by using Gaussian process as a latent process and also extend this model to high-dimensional response variable via convolved Gaussian process which is described as follows.

In this section, we consider a 2-dimensional dependent Gaussian processes which is denoted as $(y_1(\boldsymbol{x}), y_2(\boldsymbol{x}))$. The difficulty is to define the cross-variance that ensures the covariance matrix for the multi-variate response variables is still positive definite. Williams and Rasmussen (2006) introduce *multi-kriging* by applying Gaussian process for each dimension independently. However, this method cannot capture the covaring of the binary outputs. Higdon (2002) proves a Gaussian process $\xi(\boldsymbol{x})$ can be constructed using convo-

lution of a Gaussian white noise $\tau(\boldsymbol{x})$ and a smooth covariance function $h(\boldsymbol{x})$ as follows:

$$\begin{aligned}\xi(\boldsymbol{x}) &= h(\boldsymbol{x}) \star \tau(\boldsymbol{x}) \\ &= \int h(\boldsymbol{x}-\boldsymbol{\alpha})\tau(\boldsymbol{\alpha})\, d\boldsymbol{\alpha} \\ &= \int h(\boldsymbol{\alpha})\tau(\boldsymbol{x}-\boldsymbol{\alpha})\, d\boldsymbol{\alpha}\end{aligned} \tag{2.21}$$

where $\star$ denotes convolution.

Boyle and Frean (2005a) introduce an implementation for multi-variate Gaussian processes based on convolution, which is a sum of three Gaussian processes V_i, U_i and W_i where V_i is the convolution of a kernel h_i and a noise source X_i unique to that output, U_i is the convolution of a kernel k_i and a separate noise source X_0 that affects all outputs, W_i is a stationary Gaussian white noise process, see Figure 2.1.

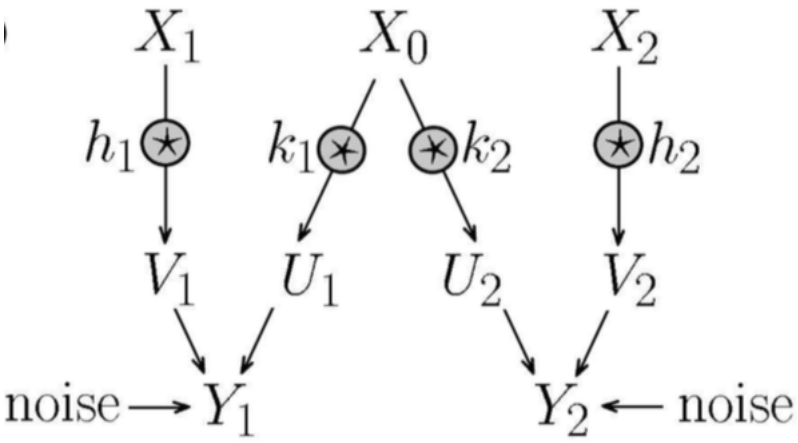


Figure 2.1: Two dependent Gaussian process by convolution.

Therefore, the dependent Gaussian processes can be defined as

$$\begin{aligned}Y_i(\boldsymbol{x}) &= U_i(\boldsymbol{x}) + V_i(\boldsymbol{x}) + W_i(\boldsymbol{x}), i = 1, 2 \\ U_i(\boldsymbol{x}) &= k_i(\boldsymbol{x}) \star X_0(\boldsymbol{x}), i = 1, 2 \\ V_i(\boldsymbol{x}) &= h_i(\boldsymbol{x}) \star X_i(\boldsymbol{x}), i = 1, 2\end{aligned} \tag{2.22}$$

where

$$
\begin{aligned}
k_1(\boldsymbol{x}) &= v_1 \exp\big(-\frac{1}{2}\boldsymbol{x}^T A_1 \boldsymbol{x}\big),\\
k_2(\boldsymbol{x}) &= v_2 \exp\big(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T A_2 (\boldsymbol{x}-\boldsymbol{\mu})\big),\\
h_1(\boldsymbol{x}) &= w_1 \exp\big(-\frac{1}{2}\boldsymbol{x}^T B_1 \boldsymbol{x}\big),\\
h_2(\boldsymbol{x}) &= w_2 \exp\big(-\frac{1}{2}\boldsymbol{x}^T B_2 \boldsymbol{x}\big),
\end{aligned}
\tag{2.23}
$$

are parameterized Gaussian kernels and $\{v_1, v_2, w_1, w_2, A_1, A_2, B_1, B_2\}$ are hyper-parameters.

The covariance matrix $\boldsymbol{C}$ is defined by

$$
\boldsymbol{C} = \begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix}
\tag{2.24}
$$

where

$$
\begin{aligned}
C_{11}^Y(\boldsymbol{d}) &= C_{11}^U(\boldsymbol{d}) + C_{11}^V(\boldsymbol{d}) + \delta_{\boldsymbol{x}\boldsymbol{x}^*}\sigma_1^2\\
C_{22}^Y(\boldsymbol{d}) &= C_{22}^U(\boldsymbol{d}) + C_{22}^V(\boldsymbol{d}) + \delta_{\boldsymbol{x}\boldsymbol{x}^*}\sigma_2^2\\
C_{12}^Y(\boldsymbol{d}) &= C_{12}^U(\boldsymbol{d})\\
C_{21}^Y(\boldsymbol{d}) &= C_{21}^U(\boldsymbol{d})
\end{aligned}
\tag{2.25}
$$

and $\delta_{\boldsymbol{x}\boldsymbol{x}^*}$ is the Kronecker delta which is defined by

$$
\delta_{\boldsymbol{x}\boldsymbol{x}^*} = \begin{cases} 1, & \text{if } \boldsymbol{x} = \boldsymbol{x}^* \\ 0, & \text{otherwise} \end{cases}
$$

Specifically, we define

$$
\begin{aligned}
C_{ii}^{U}(\boldsymbol{d}) &= \frac{\pi^{\frac{Q}{2}} v_i^2}{\sqrt{|A_i|}} \exp\big(-\frac{1}{4}\boldsymbol{d}^T A_i \boldsymbol{d}\big) \\
C_{12}^{U}(\boldsymbol{d}) &= \frac{(2\pi)^{\frac{Q}{2}} v_1 v_2}{\sqrt{|A_1 + A_2|}} \exp\big(-\frac{1}{2}(\boldsymbol{d}-\boldsymbol{\mu})^T \Sigma (\boldsymbol{d}-\boldsymbol{\mu})\big) \\
C_{21}^{U}(\boldsymbol{d}) &= \frac{(2\pi)^{\frac{p}{2}} v_1 v_2}{\sqrt{|A_1 + A_2|}} \exp\big(-\frac{1}{2}(\boldsymbol{d}+\boldsymbol{\mu})^T \Sigma (\boldsymbol{d}+\boldsymbol{\mu})\big) \\
C_{ii}^{V}(\boldsymbol{d}) &= \frac{\pi^{\frac{Q}{2}} w_i^2}{\sqrt{|B_i|}} \exp\big(-\frac{1}{4}\boldsymbol{d}^T B_i \boldsymbol{d}\big)
\end{aligned}
\tag{2.26}
$$

where Q is the dimension of covariate, $\boldsymbol{d} = \boldsymbol{x} - \boldsymbol{x}^*$ and $\Sigma = A_1(A_1 + A_2)^{-1}A_2 = A_2(A_1 + A_2)^{-1}A_1$ (Boyle and Frean, 2005b). Σ is symmetric since A_1 and A_2 are symmetric matrices. We get that $(A_1 + A_2)(A_1 + A_2)^{-1}(A_1 - A_2)$ is symmetric, since $(A_1 + A_2)(A_1 + A_2)^{-1}(A_1 - A_2) = A_1 - A_2$. Thus

$$
\begin{aligned}
&(A_1 + A_2)(A_1 + A_2)^{-1}(A_1 - A_2) \\
=&A_1(A_1 + A_2)^{-1}A_1 + A_2(A_1 + A_2)^{-1}A_1 - A_1(A_1 + A_2)^{-1}A_2 - A_2(A_1 + A_2)^{-1}A_2,
\end{aligned}
$$

we obtain that $A_2(A_1+A_2)^{-1}A_1 - A_1(A_1+A_2)^{-1}A_2$ is symmetric, since $A_1(A_1+A_2)^{-1}A_1 - A_2(A_1 + A_2)^{-1}A_2$ is symmetric. Furthermore,

$$
\begin{aligned}
&A_2(A_1 + A_2)^{-1}A_1 - A_1(A_1 + A_2)^{-1}A_2 \\
=&(A_2(A_1 + A_2)^{-1}A_1 - A_1(A_1 + A_2)^{-1}A_2)^{\top} \\
=&(A_2(A_1 + A_2)^{-1}A_1)^{\top} - (A_1(A_1 + A_2)^{-1}A_2)^{\top} \\
=&A_1(A_1 + A_2)^{-1}A_2 - A_2(A_1 + A_2)^{-1}A_1 \\
\Rightarrow&A_1(A_1 + A_2)^{-1}A_2 = A_2(A_1 + A_2)^{-1}A_1
\end{aligned}
$$

In addition, it is not necessary that $n_1 = n_2$. Specifically, C_{11} in Equation (2.24) is a $n_1 \times n_1$ matrix, C_{12} is a $n_1 \times n_2$ matrix, C_{21} is a $n_2 \times n_1$ matrix and C_{22} is a $n_2 \times n_2$ matrix. Therefore, the matrix C in Equation (2.24) can still be a cross-covariance matrix for two dependent Gaussian processes even they have different data points.

We denote the set of hyper-parameters $\boldsymbol{\Theta} = \{v_1, v_2, w_1, w_2, A_1, A_2, B_1, B_2, \mu, \sigma_1, \sigma_2\}$, which can be estimated by maximizing a likelihood function. The log-likelihood function

is given by

$$\mathcal{L} = -\frac{1}{2}\log|\boldsymbol{C}| - \frac{1}{2}\boldsymbol{y}^T\boldsymbol{C}^{-1}\boldsymbol{y} - \frac{N_1+N_2}{2}\log 2\pi \tag{2.27}$$

where N_1 is the amount of the first response variable y_1, N_2 is the amount of the second response variable y_2 and $\boldsymbol{y}^T = [y_{1,1}, ..., y_{1,N_1}, y_{2,1}, ..., y_{2,N_2}]$. The hyper-parameters $\boldsymbol{\Theta}$ can be found by either maximising a posterior or maximising the log-likelihood. Besides, Markov Chain Monte Carlo can be used to simulate the predictive distribution for y (Neal, 1993). As a result, the predictive distribution is still a normal distribution

$$\begin{aligned} E[y^*] &= \boldsymbol{k}^T\boldsymbol{C}^{-1}\boldsymbol{y} \\ Var(y^*) &= \kappa - \boldsymbol{k}^T\boldsymbol{C}^{-1}\boldsymbol{k} \end{aligned} \tag{2.28}$$

where $\kappa = C^Y_{ii}(0) = v_i^2 + w_i^2 + \sigma_i^2$ and $\boldsymbol{k} = [C^Y_{i1}(\boldsymbol{x}^* - \boldsymbol{x}_{1,1})\ ...\ C^Y_{i1}(\boldsymbol{x}^* - \boldsymbol{x}_{1,N_1})\ C^Y_{i2}(\boldsymbol{x}^* - \boldsymbol{x}_{2,1})\ ...\ C^Y_{i2}(\boldsymbol{x}^* - \boldsymbol{x}_{2,N_2})]$.

2.3 Gaussian process functional regression

The mean function in Gaussian process regression is supposed to be a constant or a known function. In section, we discuss the a model in which the mean function is trainable for functional data. Specifically, the response variables are considered as functional and predictor variables are considered as functional or scalar. Suppose the training data are $\{(y_m(t), \boldsymbol{x}_m(t), u_m)|\boldsymbol{x}_m(t) \in \mathbb{R}^Q, \boldsymbol{u}_m \in \mathbb{R}^p, y_m(t) \in \mathbb{R}, m = 1, ..., M\}$, where t usually refers to time. Shi et al. (2007) introduce Gaussian process functional regression for batch data as follows

$$\begin{aligned} y_m(t) &= \mu_m(t, \boldsymbol{u}_m) + \tau_m(\boldsymbol{x}_m(t)) + \epsilon_m(t) \\ \tau_m(\boldsymbol{x}_m(t)) &\sim GPR_m(0, k_m(\boldsymbol{\theta}_m)|\boldsymbol{x}_m) \end{aligned}$$

where $\mu_m(t)$ denotes the common mean structure of various curves, $\tau_m(\boldsymbol{x}(t))$ denotes the dependent error within the identical curve which is called covariance structure and $\epsilon_m(t)$ denotes independent random error. Gaussian process regression with zero mean function

is applied to model the covariance structure and it is determined by covariance function $k_m(\cdot)$ and hyper-parameter $\boldsymbol{\theta}_m$.

Scalar covariates $\boldsymbol{u}_m$ and time t determine the mean structure, meanwhile, functional covariates $\boldsymbol{x}_m(t)$ determines the covariance structure. The GPFR model is very flexible, because it incorporates mean structure and covariance structure of functional relationships. For instance, we can model the complex relationship of functional response variable and functional covariates by Gaussian process regression non-parametrically with little known physical connection between them. On the other hand, if we know the relationship parametrically, we can consider these functional covariates in mean structure. Another advantage of Gaussian process functional regression model is that it learns the mean structure and covariance structure simultaneously from all subjects or batch data; moreover, predictions are based on the estimated mean structure and observed data in the particular individual, which promotes the accuracy of prediction significantly. The choice of mean structure could be parametric or non-parametric. For example, Shi and Choi (2011) consider the functional linear regression as the mean structure in GPFR, whose predictions are more accurate than that of conventional Gaussian process regression model.

In Chapter 4, we extend the Gaussian process functional regression model to the non-Euclidean space, such as Riemannian manifolds. Specifically, a functional linear regression model in tangent spaces is considered as the mean structure and a Gaussian process regression model is considered as covariance structure.

2.4 Riemannian manifolds

Statistical and machine learning models are explored deeply in Euclidean space. However, people are recording lots of data in non-Euclidean spaces, especially on Riemannian manifolds. In this section, we review some concepts of Riemannian manifolds and present a few examples of Riemannian manifolds which are used in Chapter 4 and Chapter 5. There are more details in Chapter 4 in Do Carmo (1992) and Chavel (2006).

2.4.1 Concepts and notations

A d-dimensional *differentiable manifold* can be seen as a topological space that is similar to vector space structure locally and differentiable globally. Given a d-dimensional differentiable manifold $\mathcal{M}$ and a tangent space $T_p\mathcal{M}$, for $p \in \mathcal{M}$, a *Riemannian metric* $g_p : T_p\mathcal{M} \times T_p\mathcal{M} \to \mathbb{R}$ on $\mathcal{M}$ is a family of inner products which is smoothly varying through all $p \in \mathcal{M}$. Equipped with Riemannian metric, $(\mathcal{M}, g)$ is defined as a Riemannian manifold. *Geodesic* $\gamma(\cdot)$ is the shortest path connecting two local points $p \in \mathcal{M}$ and $q \in \mathcal{M}$, corresponding to straight line in vector space, such that $\gamma(0) = p$, $\frac{d}{dt}\gamma|_{t=0} = v$ and $\gamma(1) = q$ where $v \in T_p\mathcal{M}$ refers to a *tangent vector* from p to q (Jayasumana et al., 2013). The length of geodesic is named *geodesic distance*. Further, the *geodesic metric* $d_{\mathcal{M}}$ is a basic and essential metric on $\mathcal{M}$ and it can be parameterized by tangent vector $v \in T_p\mathcal{M}$ with the *Riemannian exponential map* or *exponential map* $\text{Exp} : \mathcal{M} \times T_p\mathcal{M} \to \mathcal{M}$. The exponential map is locally diffeomorphic onto a neighbourhood of p, and let $V(p)$ be the largest such neighbourhood. The minimum distance from p to the boundary of a maximal $V(p)$ is called the *injectivity radius* of an exponential map. Then within $V(p)$ the exponential map has an inverse, i.e. the *Riemannian logarithm map* or *inverse exponential map* $\text{Log} : p \times q \to T_p\mathcal{M}$. Under the view of operation, the inverse exponential map is a generalization of subtraction from vector space to a Riemannian manifold. Meanwhile, the classic addition operator can be extended onto a Riemannian manifold, leading to exponential map. In statistics, we often calculate the expectation of random variables and on a Riemannian manifold, we can define the expectation of a manifold-valued random variable as *Fréchet mean* (Fréchet, 1948) which is written by

$$E[X] = \{p | p \in \arg\min_{q \in \mathcal{M}} (E[d_{\mathcal{M}}(p, X)^2])\} \tag{2.29}$$

where X refers to a manifold-valued random variable and $E[X]$ is a set. When there is a unique mean $\bar{p}$, we denote $E[X] = \bar{p}$ (Mallasto and Feragen, 2018).

If $(\mathcal{M}, d_{\mathcal{M}})$ is a *complete metric space*, then these Riemannian manifolds are regarded as *geodesically complete*. In other words, there almost surely exists a unique shortest curve connecting any two points globally on a complete Riemannian manifoldexcept for antipodes (Section 2.3 in Dai et al. (2018)). It can be also considered as $\text{Exp}(\cdot, \cdot)$ is

defined on the entire *tangent bundle* $T\mathcal{M}$ and $\text{Log}(\cdot,\cdot)$ is defined on the entire complete Riemannian manifold. It is one of the basic and widely used assumptions of data analysis on Riemannian manifold which ensures almost sure existence and uniqueness of Fréchet means and geodesics between any two points. Therefore, we suppose that this assumption holds subsequently. Moreover, Karcher (1977) and Kendall (1990) introduce the conditions of existence and uniqueness of Fréchet means

In addition, suppose $c : I \to \mathcal{M}$ is a differentiable curve on $\mathcal{M}$ and V_0 is a tangent vector on $T_{c(t_)}\mathcal{M}$, where $t_0 \in I$. There exists a unique parallel vector field V along c, such that $V(t_0) = V_0$. Thus, we define $V(t)$ is a parallel transport $V(t_0)$ along c.

2.4.2 Operators between Euclidean space and Riemannian manifolds

For the consistency with Euclidean space, we present some useful operations which have very similar meaning between Riemannian manifolds and linear vector space (Kim et al., 2014).

Suppose $p_{\mathcal{M}}$ and $q_{\mathcal{M}}$ refer to two different points on $\mathcal{M}$ and the tangent vector from the former to the latter is denoted by $pq_{\mathcal{M}}$. Suppose there is a geodesic function $\gamma(t),\ t \in [0,1]$ on $\mathcal{M}$ given $\gamma(0) = p$ and $\gamma(1) = q$, a tangent vector can be considered as $pq_{\mathcal{M}} = \frac{d}{dt}\gamma|_{t=0}$. In other words, a tangent vector can be considered as in initial velocity of a moving particle on $\mathcal{M}$ whose initial position is p and final position is q. In addition, p_E and q_E refers to two Euclidean-valued data points and the vector from the former to the latter is given by pq_E. Some operations are shown in Table 2.1.

	Euclidean Space	Riemannian Manifold
Addition	$q_E = p_E + pq_E$	$q_{\mathcal{M}} = \text{Exp}(p_{\mathcal{M}}, pq_{\mathcal{M}})$
Subtraction	$pq_E = p_E - q_E$	$pq_{\mathcal{M}} = \text{Log}(p_{\mathcal{M}}, q_{\mathcal{M}})$
Distance	$\|p_E - q_E\|$	$\|\text{Log}(p_{\mathcal{M}}, q_{\mathcal{M}})\|$

Table 2.1: Some basic operators between Euclidean space and Riemannian manifold.

2.4.3 An example of Riemannian manifold: 2-Sphere

A simple example of a Riemannian manifold is the unit sphere S^2 in $\mathbb{R}^3$. Given any two points p and q on a S^2, there are many curves which pass through these two points.

As mentioned above, the shortest curve connecting p and q is exactly a geodesic $\gamma : [0,1] \to S^2$ and it is also known as the *the great circle* under the context of sphere. The initial conditions of geodesic are $\gamma(0) = p$, $\gamma(1) = q$ and $\gamma'(0) = v$. Besides, the inverse exponential map of p and q is denoted as $\mathrm{Log}(p, q)$, which is a tangent vector in the tangent space of p, say $T_p\mathcal{M}$. The norm of $\mathrm{Log}(p, q)$ equals to the geodesic distance from p to q and the direction of $\mathrm{Log}(p, q)$ is the initial velocity of $\gamma(\cdot)$. That is $d_{\mathcal{M}}(p, q) = \|\mathrm{Log}(p, q)\|$ and $\mathrm{Log}(p, q) = \gamma'(0)$. Therefore, the inner product of inverse exponential map provides a metric of S^2 which ensures that the analysis of S^2-valued data becomes tractable. These concepts can be extend onto hyper-spheres S^n, where $n > 2$. Since S^2 is a typical Riemannian manifold which is geodesically complete, the exponential map and inverse exponential map exist everywhere except antipodal pairs.

For a general spherical Riemannian manifold, supposing $p, q \in S^n$ and $n \in \mathbb{Z}^+$, it is necessary to formulate some mathematical tools for further computation in Chapter 5. For further details about these formula below, please see Zhang and Fletcher (2013).

The formula of geodesic distance is

$$d^1_{\mathcal{M}}(p, q) = \cos^{-1}(p^T q). \tag{2.30}$$

Moreover, since S^2 can be embedded into $\mathbb{R}^3$, an alternative is to measure the distance between p and q by the norm of their difference. That is

$$d^2_{\mathcal{M}}(p, q) = \|p - q\|. \tag{2.31}$$

The distance (2.30) describes the intrinsic metric between p and q while the distance function (2.31) describes the chordal metric. For simplicity, we use the latter to measure the distance between predictions and real data.

The formula of inverse exponential map is

$$\mathrm{Log}(p, q) = \frac{u}{\|u\|} d_{\mathcal{M}}(p, q), \text{ where } u = q - (p^T q)p, \ q \in S^n \backslash \{-p\}. \tag{2.32}$$

The formula of exponential map is

$$\text{Exp}(p, v) = \cos(\|v\|)p + \sin(\|v\|)\frac{v}{\|v\|},\ v = \text{Log}(p, q) \in T_p\mathcal{M}. \tag{2.33}$$

With a given data set $\mathcal{D} = \{x_i \in \mathbb{R}, y_i \in \mathcal{M} | i = 1, ..., N\}$, a geodesic regression model (Fletcher, 2013) is defined by

$$y_i = \text{Exp}(\text{Exp}(\mu, x_i\tau), \epsilon),\ \text{i=1,..,N},$$

where x_i is an independent variable, y_i is a dependent variable, N is sample size, μ is a manifold-valued point and τ is a tangent vector. The loss function with respect to μ can be defined by

$$E(\mu) = \frac{1}{2}\sum_{i=1}^{N} d(\text{Exp}(\mu, x_i v), y_i)^2.$$

For further computation, such as the gradient of a loss function with respect to the manifold-valued data point μ is given by

$$\begin{aligned} \nabla_\mu E &= -\sum_{i=1}^{N} d_\mu \text{Exp}(\mu, \tau)^\dagger \text{Log}(y_i, \text{Exp}(\mu, \tau)) \\ &= -\sum_{i=1}^{N} \left(\cos\left(\|v\|\right) w_i^\perp + w_i^\top \right) \end{aligned} \tag{2.34}$$

where $\mu \in \mathcal{M}$, $\tau \in T_\mu\mathcal{M}$, $y_i \in \mathcal{M}$, for $i = 1, ..., N$, $w_i = \text{Log}(y_i, \text{Exp}(\hat{\mu}, \tau))$, $w_i^\perp = \frac{\langle w_i, \tau \rangle}{\|\tau\|^2}\tau$, $w_i^\top = \text{Log}(y_i, \text{Exp}(\mu, \tau)) - \frac{\langle w_i, \tau \rangle}{\|\tau\|^2}\tau$, E refers to a loss function between predictions and real data, $\dagger$ refers to an adjoint of a linear operator, i.e. $< d_\mu \text{Exp}(\mu, \tau)a, b > = < a, d_\mu \text{Exp}(\mu, \tau)^\dagger b >$ and d_μ refers to a differential operator on μ with respect to a tangent vector on $T_\mu\mathcal{M}$.

In Chapter 4, we use Equation (2.34) for inference.

2.4.4 An example of Riemannian manifold: Kendall's shape space

As a well-developed Riemannian manifold, Kendall's shape space provides a very useful space for many theories and applications in statistics and machine learning. Thus, we

suppose there are no assumptions about the geometry of the objects after removing translation, scale and rotation. Moreover, Kendall's shape space is also a symmetric Riemannian manifold and then we can try to derive the gradient descent algorithm in estimation part.

In 2-dimensional shape analysis, *landmarking* is an approach to sketch a 2-dimensional shape by k points which contours a shape and such a $2k$-dimensional vector can be represented in a k-dimensional complex space $\mathbb{C}^k$. Researchers neglect translation of shapes by moving the centroid to be origin point. Thereafter, we obtain a linear complex subspace $V = \{z = (z_1, ..., z_k) \in \mathbb{C}^k | \sum_{i=1}^k z_i = 0\}$. V can be considered as a subspace passing along the origin of $\mathbb{C}^k$ which means V is identical to $\mathbb{C}^{k-1}$. Two shapes in $\mathbb{C}^{k-1}$ are equivalent if they are scaling or rotation of each other. The equivalent classes form a *complex projective space* denoted as $C\mathbb{P}^{k-2}$.

We also start from a fundamental 2-dimensional shape which is triangle. Obviously, 3 points are necessary to represent 3 vertexes of a triangle. Thus, there are 6 variables of the 3 points which structure a random variable in $\mathbb{C}^3$. Suppose p and q are two points in $\mathbb{C}^3$. After removing translation, scale and rotation, p and q would be in $C\mathbb{P}^1 = S^2$. Complex projective space can also be generalized to a high-dimensional case.

Analogously to Section 2.4.3, from Zhang and Fletcher (2013), we parameterize some formulas, such as exponential map and inverse exponential map for shape space. Suppose p and q are two data points in a general $C\mathbb{P}^{k-2}$, the geodesic distance can be computed by

$$d_{\mathcal{M}}(p, q) = \|p - q\| \tag{2.35}$$

The closed form of inverse exponential map is given by

$$\text{Log}(p, q) = \frac{\theta \cdot (q - \pi_p q)}{\|q - \pi_p q\|},\ \theta = \arccos \frac{|\langle p, q \rangle|}{\|p\| \|q\|},\ \pi_p q = \frac{p \cdot \langle p, q \rangle}{\|p\|^2} \tag{2.36}$$

The exponential map is parameterized as follows

$$\text{Exp}(p, v) = \cos\theta \cdot p + \frac{\|p\| \sin\theta}{\theta},\ \theta = \|\text{Log}(p, q)\| \tag{2.37}$$

For further details about basic formulae on S^2 and Kendall's shape space, see Zhang

and Fletcher (2013).

2.4.5 Sample Fréchet mean

For a set of manifold-valued data $\boldsymbol{y} = (y_1, y_2, ..., y_n)$, their sample Fréchet mean p is defined as the minimizer of the sum of squared distance function

$$p = \arg\min_{p \in \mathcal{M}} \sum_{i=1}^{n} d_{\mathcal{M}}(y_i, p)^2 \tag{2.38}$$

where $d_{\mathcal{M}}(\cdot, \cdot)$ denotes geodesic distance. In order to estimate the minimizer of (2.38), it is natural to use gradient descent algorithm introduced by Pennec (1999). Moreover, based on Do Carmo (1992), the gradient is given by

$$\nabla d_{\mathcal{M}}(p, y_i)^2 = -2\mathrm{Log}(y_i, p) \tag{2.39}$$

An implementation is shown in Algorithm 1.

Algorithm 1: Gradient Descend Algorithm for Sample Fréchet Mean.

1. Initialise a manifold-valued point p and set a step size ϵ and a small positive number τ;
2. $\nu = \frac{\epsilon}{n} \sum_{i=1}^{n} \mathrm{Log}(y_i, p)$;
3. While $\|\nu\| > \tau$, do $\nu = \frac{\epsilon}{n} \sum_{i=1}^{n} \mathrm{Log}(y_i, p)$ and $p = \mathrm{Log}(p, \nu)$; else, end.

At the beginning of this optimization algorithm, the initialised p is often not a precise estimation of Fréchet mean and its gradient is large. After it moves along the gradient, p is closer to the "real" Fréchet mean. We repeat these steps above until the gradient is small enough which leads to an approximate sample Fréchet mean of $\boldsymbol{y}$. See Chapter 2 in Pennec et al. (2019) for more details.

2.5 Some special distributions

Probability distributions are fundamental concepts and tools used in statistics, such as normal distribution. We list some special probability distributions below with some helpful formulas, which are used in this thesis.

2.5.1 Truncated normal distribution

The truncated normal distribution is a normal distributed random variable which has upper boundary or lower boundary or both. There are extensively applications of truncated normal distribution in statistics and econometrics (Greene, 2003; Botev, 2017).

To define a truncated normal distribution, we need identify the mean, the variance and boundaries which could be infinite. As previously, suppose the mean is μ, the variance is σ^2, the lower boundary and upper boundary are denoted as l and u respectively. Hence, a random variable X which follows a truncated normal distribution can be written as

$$X \sim \mathcal{N}(\mu, \sigma, l, u) \tag{2.40}$$

and its *probability density function* is given by

$$p(X; \mu, \sigma, l, u) = \frac{1}{\sigma} \frac{\phi(\frac{X-\mu}{\sigma})}{\Phi(\frac{u-\mu}{\sigma}) - \Phi(\frac{l-\mu}{\sigma})} \tag{2.41}$$

where $\phi(\cdot)$ refers to the probability density function of the standard normal distribution and $\Phi(\cdot)$ refers to *cumulative distribution function*. The expectation and variance of a two-sided truncated normal distribution are

$$E[X|l < X < u] = \mu + \sigma \frac{\phi(\alpha) - \phi(\beta)}{\Phi(\alpha) - \Phi(\beta)} \tag{2.42}$$

$$Var(X|l < X < u) = \sigma^2\big(1 + \frac{\alpha\phi(\alpha) - \beta\phi(\beta)}{\Phi(\beta) - \Phi(\alpha)} - (\frac{\phi(\alpha) - \phi(\beta)}{\Phi(\beta) - \Phi(\alpha)})^2\big) \tag{2.43}$$

where $\alpha = \frac{l-\mu}{\sigma}$ and $\beta = \frac{u-\mu}{\sigma}$. If $l = -\infty$ or $u = \infty$, it is easy to derive the expectation and variance for one-sided truncated normal distribution. When $l = -\infty$, we can substitute $\phi(\alpha)$ by 0 and $\Phi(\alpha)$ by 0; when $u = \infty$, we can substitute $\phi(\beta)$ by 0 and $\Phi(\beta)$ by 1. For further details about how these formulas are derived, see Johnson et al. (1995).

2.5.2 Wrapped Gaussian distribution

Normal distribution has wide applications in practice. However, it limits the random variable in Euclidean space which is not suitable for manifold-valued random variables. Mardia and Jupp (2009) extend normal distribution to Riemannian manifolds via a wrap-

ping function, which is called *wrapped Gaussian distribution*. In this part, we discuss the wrapping function used in Mallasto and Feragen (2018) in which the authors linearize a Riemannian manifold by a inverse Exponential map. In addition, some inferences, such as conditional distribution and predictive distribution, are also shown.

Suppose $\mathcal{M}$ denotes a d-dimensional Riemannian manifold and X is a random variable on $\mathcal{M}$. For $\mu \in \mathcal{M}$ and a symmetric positive definite matrix $K^{d\times d}$, we can define a wrapped Gaussian distribution as follows

$$\begin{aligned} X &= \mathrm{Exp}(\mu, \boldsymbol{v}) \\ \boldsymbol{v} &\sim \mathcal{N}(\mathbf{0}, K) \end{aligned} \tag{2.44}$$

where μ is called *basepoint* and $\boldsymbol{v}$ is a vector following a multi-variate Gaussian distribution with zero mean and covariance matrix K. To generate a data from this distribution, we sample a tangent vector $\boldsymbol{v}$ from $\mathcal{N}(\mathbf{0}, K)$ and a generated data point is $\mathrm{Exp}(\mu, v)$. The wrapped Gaussian distribution is formally denoted as

$$X \sim \mathcal{N}_{\mathcal{M}}(\mu, K) \tag{2.45}$$

Based on the Proposition 2.11 in Oller and Corcuera (1995) and discussion in Mallasto and Feragen (2018), $\mu \in \mathbb{E}[X]$ when the injectivity radius is infinite, where $\mathbb{E}[\cdot]$ denotes the Fréchet mean. In this thesis, we consider the wrapped Gaussian process functional regression model on some frequently-used and special Riemannian manifolds. Therefore, the mean of wrapped Gaussian distribution $\mathcal{N}_{\mathcal{M}}(\mu, K)$ is the Fréchet mean of manifold-valued random variable X.

The *jointly wrapped Gaussian distribution* of two manifold-valued random variables X_1 and X_2 is defined as

$$\begin{pmatrix} X_1 \\ X_2 \end{pmatrix} \sim \mathcal{N}_{\mathcal{M}_1\times\mathcal{M}_2}\left(\begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \begin{pmatrix} K_1 & K_{12} \\ K_{21} & K_2 \end{pmatrix}\right) \tag{2.46}$$

where $X_1 \sim \mathcal{N}_{\mathcal{M}_1}(\mu_1, K_1)$, $X_2 \sim \mathcal{N}_{\mathcal{M}_2}(\mu_2, K_2)$ and $K_{12} = K_{21}^T$. Mallasto and Feragen (2018) derived the conditional distribution of X_1 given X_2. Specifically, suppose $p_1 \in \mathcal{M}_1$, $p_2 \in \mathcal{M}_2$ are manifold-valed points, $B = \mathrm{Log}(\mu_1, p_1)$, $A = \log(\mu_2, p_2)$ are tangent vectors,

and K is covariance of (X_1, X_2). Thus, we have

$$\begin{aligned}
&\mathbb{P}\{X_1 = p_1 | X_2 = p_2\} \\
&= \frac{\mathbb{P}\{u \in B, v \in A\}}{\mathbb{P}\{v \in A\}} \\
&= \sum_{v \in A, u \in B} \frac{\mathcal{N}(\mathbf{0}, K_2)}{\mathbb{P}\{A\}} \frac{\mathcal{N}(\mathbf{0}, K)}{\mathcal{N}(\mathbf{0}, K_2)} \\
&= \sum_{v \in A, u \in B} \lambda_v \mathcal{N}(\mu_v, K_v) \\
&= \mathbb{P}\{Z = p_1\}
\end{aligned}$$

where u and v are tangent vectors, $Z \sim \mathrm{Exp}(\mu_1, \sum_{v \in A, u \in B} \lambda_v \mathcal{N}(\mu_v, K_v))$, $\mathcal{N}(\mu_v, K_v)$ is a predictive distribution (2.15) and μ_1 is the expectation of X_1.

The conditional distribution is given by

$$X_1 | (X_2 = p_2) \sim \mathrm{Exp}(\mu_1, \sum_{v \in A} \lambda_v \mathcal{N}(\mu_v, K_v)) \tag{2.47}$$

where

$$\begin{aligned}
\mathcal{N}(\mu_v, K_v) &= \frac{\mathcal{N}(\mathbf{0}, K)}{\mathcal{N}(\mathbf{0}, K_2)} \\
\mu_v &= K_{12} K_2^{-1} v \\
K_v &= K_1 - K_{12} K_2^{-1} K_{12}^T \\
\lambda_v &= \frac{\mathcal{N}(\mathbf{0}, K_2)}{\mathbb{P}\{A\}} \\
A &= \{v \in T_{\mu_2}\mathcal{M} | \mathrm{Exp}(\mu_2, v) = p_2\} \\
\mathbb{P}\{A\} &= \sum_{v \in A} \mathcal{N}(\mathbf{0}, K_2)
\end{aligned}$$

Within the assumption that the injectivity radius is infinite and since a Gaussian mixture distribution can be approximated a Gaussian distribution, we have the following approximation

$$X_1 | (X_2 = p_2) \sim \mathrm{Exp}(\mu_1, \mathcal{N}(\mu_{\mathrm{Log}(\mu_2, p_2)}, K_{\mathrm{Log}(\mu_2, p_2)})) \tag{2.48}$$

where

$$\mu_{\text{Log}(\mu_2,p_2)} = K_{12}K_2^{-1}\text{Log}(\mu_2, p_2)$$
$$K_{\text{Log}(\mu_2,p_2)} = K_1 - K_{12}K_2^{-1}K_{12}^T.$$

The conditional distribution of functional wrapped Gaussian process in Chapter 4 is similar. We suppose that $y_m(t)$ is a functional wrapped Gaussian process, the mean structure is $\mu_m(t)$, the covariance structure is $\tau_m(t)$ and $t \in T$. Then the conditional distribution of $y_m(t_1)$ given $y_m(t_2)$ at t_2 is

$$y_m(t_1)|(y_m(t_2) = p_2) \sim \text{Exp}(\mu_m(t_1), \tau_m(t_1)),\ t_1, t_2 \in T, \tag{2.49}$$

where

$$\tau_m(t_1) \sim \mathcal{N}(\mu_{\text{Log}(\mu_m(t_2),p_2)}, K_{\text{Log}(\mu_m(t_2),p_2)})$$
$$\mu_{\text{Log}(\mu_m(t_2),p_2)} = K_{12}K_2^{-1}\text{Log}(\mu_m(t_2), p_2),$$
$$K_{\text{Log}(\mu_m(t_2),p_2)} = K_1 - K_{12}K_2^{-1}K_{12}^T,$$
$$K_1, K_{12} \text{ and } K_2 \text{ are covariance matrices depended by a kernel } k(\cdot,\cdot).$$

We should notice that $\tau_m(t_1)$ is about t_1, since K_1, K_{12} are about t_1 and t_2 is given.

2.6 Differential evolution

A challenging problem is statistics and machine learning is parameter estimation, since for many models, it is difficult to obtain an analytical form of gradient. Differential evolution (DE) is an evolution search strategy and it is able to optimize a multi-variate real-valued function without using the gradient which is required by gradient descent algorithm or Newton method. Moreover, the loss function or fitness function in DE is not necessarily smooth, convex or even continuous. This algorithm often provides an outstanding solution regardless of whether the objective function is smooth or even continuous (Storn and Price, 1997). Since it is computational expensively to calculate the gradient of log-likelihood function (3.24) and (3.34), in this thesis, we use DE to estimate hyper-parameters by minimizing these two function.

In this iterative optimization technique, we firstly generate a lot of candidate solutions and let them move in search-space by incorporating with another candidate based on some formulae, which can be seen as evolution. If the evolution moves towards a correct direction, such as a smaller value of loss function, the new candidate is accepted; otherwise, it would be rejected (Price et al., 2006). This process is replicated for a given iterative number and it might avoid the results from converging the local minimum. However, it is not guaranteed the optimal solution would be discovered eventually.

We define $f : \mathbb{R}^n \to \mathbb{R}$ is the fitness value function (could be loss function or negative likelihood function) which measures the goodness of candidate solutions. The goal of differential evolution is to discover a solution which minimizes $f(\boldsymbol{x})$. The formal DE algorithm can be described as follows:

Algorithm 2: Differential Evolution Algorithm to Minimize $f(\cdot)$.

1. Initialise the population randomly in a search-space, suppose the population size is P and set $i = 1$;
2. For candidate $\boldsymbol{x}_i$, randomly select three different candidate solutions $\boldsymbol{a}$, $\boldsymbol{b}$ and $\boldsymbol{c}$ from the population ;
3. Randomly select a number m from $\{1, ..., n\}$ where n is the dimension of $\boldsymbol{x}_i$;
4. For each $j \in \{1, ..., n\}$, generate $r_j \sim U(0, 1)$. If $r_j < u$, then a new candidate is formed by $\boldsymbol{x}_{new} = \boldsymbol{a} + \nu(\boldsymbol{b} - \boldsymbol{c})$; otherwise, $\boldsymbol{x}_{new} = \boldsymbol{x}_i$, where $u \in [0, 1]$ is the probability of crossover and $\nu \in [0, 2]$ is the differential weight ;
5. If $f(\boldsymbol{x}_{new}) < f(\boldsymbol{x}_i)$, replace $\boldsymbol{x}_i$ by the better individual $\boldsymbol{x}_{new}$; otherwise, do nothing. $i = i + 1$;
6. If $i > P$, go next; otherwise, go to Step 2 ;
7. If meet termination conditions, go next; otherwise, go back to Step 2 ;
8. Return the candidate solution $\boldsymbol{x}^*$ with the best fitness value $f(\boldsymbol{x}^*)$.

2.7 Approximate Bayesian computation

In Section 3.3, we can make a prediction with an approximated full conditional distribution. However, the prediction is affected by the symmetric property of the approximation, which results to a inaccurate prediction. In order to avoid such influence, we apply approximate Bayesian computation for prediction.

In Bayesian statistics, a class of computational tools form approximate Bayesian computation (ABC) which is an effective likelihood free algorithm to estimate the posterior distribution of parameters. For statistical inference, the likelihood function plays an essential role since it expresses information from available data with a given parametric model. Analytic formula of likelihood function can be specifically derived for simple models. However, in the real world, most statistical models have complex structure leading to likelihood function analytical intractably. As a likelihood free method, ABC is applied in wide statistical realm, especially in parameter selection and model selection.

One important feature of approximate Bayesian computation is the reliance of data generating mechanism based on which we can generate pseudo-observations. In other words, ABC is a simulator-based method. Suppose we know the process of data-generating and the observed data $\boldsymbol{y} = (y_1, ..., y_n)$. The pseudo-observations can be simulated by $\boldsymbol{y}_{sim} \sim P(\boldsymbol{y}|\theta)$ since the generating mechanism is known, where θ refers to target parameter. If the distance between simulation $\boldsymbol{y}_{sim}$ and observed data $\boldsymbol{y}$ is within a tolerance, we accept this θ as a sample of its posterior distribution; otherwise, we may consider another θ. This rejection-ABC algorithm provides simulated data without closed form of probability density function, which is proposed by Tavaré et al. (1997). There are also some extensions of ABC, such as MCMC-ABC (Marjoram et al., 2003) and sequential Monte Carlo (Sisson et al., 2007).

Inspired by approximate Bayesian computation, in this thesis, we use the ABC algorithm for prediction instead of estimation. Specifically, we firstly sample from a Gaussian approximation to form full conditional distribution and then sample from a predictive distribution to make a prediction. See Section 3.3.4 for more details.

2.8 Gibbs sampling

One challenge in Bayesian statistics is the estimation of posterior distribution which involves calculation of integrals and for most models, the calculation is analytically intractable. Markov chain Monte Carlo technique provides possible sampling algorithms to obtain inferences from these posterior distribution without calculating the integrals (Andrieu et al., 2003).

The target expectation can be approximated by ergodic averages is the underlying logic of MCMC sampling. Therefore, it is viable for us to compute any statistics of a posterior distribution by lots of simulated samples. Now the problem is that how can we calculate the simulated samples. Gibbs sampling is a suitable and efficient algorithm to address this issue for random variables $\boldsymbol{x} = (x_1, ..., x_n)$ (Casella and George, 1992). Specifically, for the first random variable, Gibbs sampling draws sample from its conditional distribution when the other variables are fixed to their current values. After updating value of the first variable by the simulated sample, we sweep to the second random variable. This iterative process continuous until some convergence conditions have been satisfied.

The random variables are often initialized with random values which results in the early iteration might not be the accurate representation of real posterior distribution. These iterations are common to discarded and denoted as burn-in period. In addition, the MCMC theory ensures that stationary distribution is the desired posterior distribution (Yildirim, 2012). The Algorithm 3 describes the iterative processes of Gibbs sampling.

Algorithm 3: Algorithm for Gibbs Sampling.

1. Initialise the variables $\boldsymbol{x}^{(0)} = (x_1^{(0)}, ..., x_n^{(0)})$ randomly or by some prior knowledge and set $i = 0$, where n refers to the number of variables ;
2. Draw

$$x_1^{(i+1)} = p(X_1 = x_1 | X_2 = x_2^{(i)}, X_3 = x_3^{(i)}, ..., X_{n-1} = x_{n-1}^{(i)}, X_n = x_n^{(i)})$$
$$x_2^{(i+1)} = p(X_2 = x_2 | X_1 = x_1^{(i+1)}, X_3 = x_3^{(i)}, ..., X_{n-1} = x_{n-1}^{(i)}, X_n = x_n^{(i)})$$
$$\vdots$$
$$x_{n-1}^{(i+1)} = p(X_{n-1} = x_{n-1} | X_1 = x_1^{(i+1)}, X_2 = x_2^{(i+1)}, ..., X_{n-2} = x_{n-2}^{(i+1)}, X_n = x_n^{(i)})$$
$$x_n^{(i+1)} = p(X_n = x_n | X_1 = x_1^{(i+1)}, X_2 = x_2^{(i+1)}, ..., X_{n-2} = x_{n-2}^{(i+1)}, X_{n-1} = x_{n-1}^{(i+1)});$$

3. If convergence conditions have been satisfied, end; else $i = i + 1$ and back to Step 2.

In this thesis, we applied Gibbs sampling to draw samples following multi-variate truncated Gaussian from a corresponding multi-variate Gaussian distribution which is used for prediction.

Chapter 3

Gaussian Process Regression Model for Two Non-Gaussian Distributed Data

One assumption of a Gaussian process is that any finite collection of random variables follows a multi-variate normal distribution. However, in practice, it is unreasonable to suppose random variables in any scenario satisfy such assumption. For example, in financial mathematics, people use log-normal distribution for stock price (Knight et al., 2001) which is always positive. Moreover, in actuarial science, people often use gamma distribution to model the amount of insurance claims. Therefore, these applications of models for non-Gaussian distributed data encourage us to develop statistical models for such data set. Figure 3.1 shows that the age distribution of cancer incidence often follows the Gamma distribution (Belikov, 2017). Moreover, the data is downloaded Centers for Disease Control and Prevention Wide-ranging OnLine Data for Epidemiologic Research (CDC WONDER) online database (http://wonder.cdc.gov/cancer-v2012.HTML).

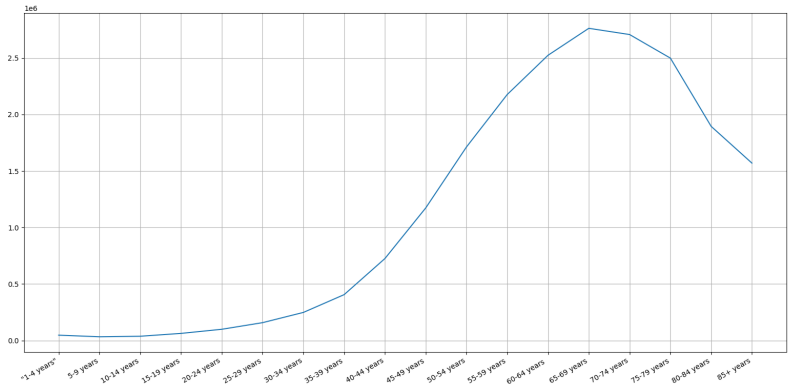


Figure 3.1: A Gamma-distributed real data set.

In this chapter, we study another two types of data which follow truncated normal distribution and Gamma distribution. Specifically, we extend Gaussian process regression to a *truncated Gaussian process regression* by assuming any collection of random variables follows a multi-variate truncated normal distribution. Moreover, we consider GPR as a latent process for Gamma-distributed data. In Section 3.1, we firstly highlight the definition of TGP, followed by the estimation and prediction. We also present the simulation study, in which, although the predictions from Gaussian process regression model are more accurate compared to truncated Gaussian process regression model, the predictions from the latter are constrained in the target interval while the predictions from the former are escaped from this interval. If people are not very sensitive to the accuracy of prediction but more sensitive to the predictive interval, TGP might be a choice for such application. In Section 2.2, we show how GPR is used for Gamma-distributed data and the inferences for prediction with an analytical solution and an approximate solution.

In other words, we model non-Gaussian data which breaks an assumption of Gaussian process regression and still in the framework of GPR.

3.1 Truncated Gaussian process regression

As discussed in Section 2.5.1, when a normally distributed random variable is constrained by an upper boundary or a lower boundary or both, it is supposed to be a truncated normal

distribution. There are also extensively applications of truncated normal distribution in statistics and econometrics (Greene, 2003; Botev, 2017). In this section, we try to break an assumption in Gaussian process regression that a response variable follows a normal distribution by using GPR for normal-distributed data.

3.1.1 Truncated Gaussian process

Inspired by defining a truncated normal distribution based on a normal distribution, we can define a truncated Gaussian process based on a Gaussian process. If we have a normal distribution, a lower boundary and a upper boundary, it is not difficult to derive some statistical formulae of such truncated normal distribution, such as probability density function, cumulative distribution function. Thus, we try to define a truncated Gaussian process based on a Gaussian process, lower boundaries and upper boundaries.

Analogous to the definition of Gaussian process, a truncated Gaussian process is a stochastic process in which any finite collection of random variables follows multi-variate truncated normal distribution and the boundaries are given smooth functions. We define a TGP as follows

$$h(\cdot) = TGP(f(\cdot), \mu(\cdot), k(\cdot, \cdot; \boldsymbol{\theta}); l(\cdot), u(\cdot)) \tag{3.1}$$

where $f(\cdot)$ denotes a Gaussian process with mean function $\mu(\cdot)$ and covariance function $k(\cdot, \cdot; \boldsymbol{\theta})$, $\boldsymbol{\theta}$ is hyper-parameters in covariance function, $l(\cdot)$ and $u(\cdot)$ are given smooth lower boundary function and upper boundary function respectively.

3.1.2 Truncated Gaussian process regression model

In order to derive the probability density function of a conventional truncated normal distribution, we firstly need to know the mean and variance of the related normal distribution and then constrain the random variable within a given interval. Analogously to the statistical inference of truncated normal distribution, in this part, we present the estimation for hyper-parameters in covariance function and prediction. Specifically, we estimate the hyper-parameters in covariance function by applying Gaussian process regression model to data directly and then constrain the prediction in a given interval by a Gibbs sampling

algorithm.

Suppose the training data are given as $\mathcal{D} = \{(\boldsymbol{x}_i, y_i) | \boldsymbol{x}_i \in \mathbb{R}^p, y_i \in \mathbb{R}, i = 1, ..., n\}$. A truncated Gaussian process regression model can be defined by

$$y_i(\boldsymbol{x}_i) \sim h(\boldsymbol{x}_i), \ i = 1, ..., n. \tag{3.2}$$

Instead of deriving the inference based on Bayesian scheme directly, we apply Gaussian process regression model for training data $\mathcal{D}$ to estimate the hyper-parameters. For more details about Gaussian process regression model, see Section 2.1.3. The expectation and variance for a new input $\boldsymbol{x}^*$ before constrain are given by

$$\begin{aligned} E[y(\boldsymbol{x}^*)] &= \boldsymbol{\phi}(\boldsymbol{x}^*)^T \Psi^{-1} \boldsymbol{y} \\ Var(y(\boldsymbol{x}^*)) &= \phi(\boldsymbol{x}^*) - \boldsymbol{\psi}(\boldsymbol{x}^*)^T \Psi^{-1} \boldsymbol{\psi}(\boldsymbol{x}^*) \end{aligned} \tag{3.3}$$

So far, the predictive distribution still follows a normal distribution and in practice, it is often multi-variate normal distribution. To make a prediction of truncated Gaussian process, the posterior Gaussian process should be constrained. Thus, the challenge converts to how to sample random variables following a multi-variate truncated normal distribution from a corresponding multi-variate normal distribution.

Kotecha and Djuric (1999) firstly introduce a Markov Chain Monte Carlo algorithm to generate random variables from a multi-variate truncated normal distribution. Furthermore, Wilhelm (2015) proposes a Gibbs sampler based on Geweke (1991) and Geweke (2005). Specifically, the author uses precision matrix rather than the covariance matrix which is more efficient in calculation, since the time-consuming computation of matrix inversions is not needed. We present a Gibbs sampler algorithm based on Wilhelm (2015) to yield a truncated Gaussian process from a Gaussian process.

The Gibbs sampler algorithm uses facts that the conditional distribution of truncated normal distribution is truncated normal again (Horrace, 2005) and it is based on full conditional distribution of multi-variate normal distribution. The full conditional distribution of a n-dimensional multi-variate normal distribution is denoted as

$p(z_i|\boldsymbol{z_{-i}}) = p(z_i|z_1, ..., z_{i-1}, z_{i+1}, ..., z_n)$ which is a normal distributed with mean

$$\mu_{z_i|\boldsymbol{z}_{-i}} = \mu_i + \Sigma_{i,-i}\Sigma^{-1}_{-i,-i}(\boldsymbol{z}_{-i} - \boldsymbol{\mu}_{-i}) = \mu_i - H^{-1}_{i,i}H_{i,-i}(\boldsymbol{z}_{-i} - \boldsymbol{\mu}_{-i})$$

and variance

$$\Sigma_{z_i|\boldsymbol{z}_{-i}} = \Sigma_{i,i} - \Sigma_{i,-i}\Sigma^{-1}_{-i,-i}\Sigma_{-i,i} = H^{-1}_{i,i}.$$

We can constrain the posterior Gaussian process to a truncated Gaussian process by applying a Gibbs sampler. Suppose independent variables in training data are $X = (\boldsymbol{x}_1, ..., \boldsymbol{x}_n)$ and make a prediction at a new input $\boldsymbol{x}^*$. The predictive expectation of Gaussian process regression is

$$\mu^* = \Sigma_{\boldsymbol{x}^*,X}\Sigma^{-1}_{X,X}\boldsymbol{y} = \boldsymbol{\psi}(\boldsymbol{x}^*)\Psi^{-1}\boldsymbol{y} \tag{3.4}$$

since we assume the mean function of Gaussian process prior is zero and the predictive variance of Gaussian process regression is

$$\sigma^* = \Sigma_{\boldsymbol{x}^*,\boldsymbol{x}^*} - \Sigma_{\boldsymbol{x}^*,X}\Sigma^{-1}_{X,X}\Sigma_{X,\boldsymbol{x}^*} = \phi(\boldsymbol{x}^*) - \boldsymbol{\psi}(\boldsymbol{x}^*)^T\Psi^{-1}\boldsymbol{\psi}(\boldsymbol{x}^*) \tag{3.5}$$

where $\boldsymbol{y} = (y_1, ..., y_n)$.

Since data points on a Gaussian process (including training data points and test data point) are related, it is not suitable to constrain the prediction at test data point only regardless of the constrains at training data points. In other words, we must constrain all predictions at training data points and test data point simultaneously. In addition, this problem can be considered as how to sample from a multi-variate truncated Gaussian distribution from a multi-variate Gaussian distribution.

To sample from multi-variate truncated normal distribution, we construct a Markov chain which draws from $p(f(\boldsymbol{x}^*)|X, \boldsymbol{y})$ subject to $l(\boldsymbol{x}^*) \leq f(\boldsymbol{x}^*) \leq u(\boldsymbol{x}^*)$. In addition, there is a transformation between sampling from a uni-variate truncated normal distribution and a uni-variate normal distribution (Greene, 2003; Hurn and Becker, 2004). Suppose that $f^{(j)}(\cdot)$ denotes samples drawn at the j-th MCMC iteration. The Algorithm 4 describes

the steps of the Gibbs sampler.

Algorithm 4: Algorithm for prediction from truncated Gaussian process.

1. Make a prediction at every $\boldsymbol{x}_i$ including $\boldsymbol{x}^*$, we obtain $\hat{\boldsymbol{f}} = (\hat{f}(\boldsymbol{x}_1), .,,, \hat{f}(\boldsymbol{x}_n), \hat{f}(\boldsymbol{x}^*))$;
2. Since the conditional variance σ^{*2} is independent from $\boldsymbol{y}$, we can pre-calculate it before running the Markov chain ;
3. Choose $\hat{\boldsymbol{f}}$ as a start value $\boldsymbol{f}^{(0)}$ of the chain;
4. Set $j = 1$, we sample from the conditional density $f^{(j)}(\boldsymbol{x}_i)|f^{(j)}(\boldsymbol{x}_1), ..., f^{(j)}(\boldsymbol{x}_{i-1}), f^{(j-1)}(\boldsymbol{x}_{i+1}), ..., f^{(j-1)}(\boldsymbol{x}^*)$ from $\boldsymbol{x}_i = \boldsymbol{x}_1$ to $\boldsymbol{x}_i = \boldsymbol{x}^*$;
5. $j = j + 1$ and replicate Step 4 until a very large j ;
6. Draw a uniform random variate $U \sim Uni(0, 1)$;
7. We draw from uni-variate conditional normal distributions with mean μ^* and variance σ^{*2}. For each realization y we can find a x such as $P(Z \leq z) = P(X \leq x)$:

$$\frac{\Phi(\frac{x-\mu}{\sigma}) - \Phi(\frac{l-\mu}{\sigma})}{\Phi(\frac{u-\mu}{\sigma}) - \Phi(\frac{l-\mu}{\sigma})} = \Phi(\frac{z-\mu}{\sigma}) = U$$

8. Draw $h(\boldsymbol{x}^*)$ from conditional uni-variate truncated normal distribution $TN(\mu^*, \sigma^{*2}, l_i, u_i)$ by:

$$h(\boldsymbol{x}^*) = \mu^* + \sigma^* \times \Phi^{-1}\left[U\left(\Phi(\frac{u_i - \mu^*}{\sigma^*}) - \Phi(\frac{l_i - \mu^*}{\sigma^*})\right) + \Phi(\frac{l_i - \mu^*}{\sigma^*})\right]$$

Equipped with Algorithm 4, a truncated Gaussian process can be generated from a corresponding Gaussian process. By setting a large iteration number, we can obtain a number of samples which are drawn from a truncated normal distribution. Moreover, the mean of these samples are our prediction at $\boldsymbol{x}^*$ based on TGP.

3.1.3 Simulation study

We firstly show comparison of two sets of samples drawn from a bi-variate Gaussian distribution and corresponding bi-variate truncated Gaussian distribution based on Algorithm

4. Specifically, the random variable X following a bi-variate Gaussian distribution which is

$$X \sim \mathcal{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0.3 & 0.1 \\ 0.1 & 0.3 \end{pmatrix} \right). \tag{3.6}$$

10000 samples are drawn from the distribution above and we then run the Gibbs sampling to drawn samples with lower boundary $(-1, -0.9)$ and upper boundary $(1, 0.9)$ which is displayed below

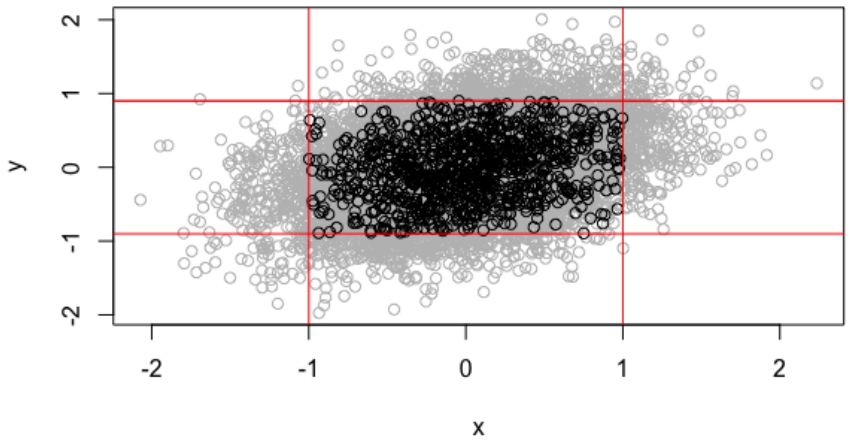


Figure 3.2: An example of Algorithm 4, where the gery points are drawn from distribution (3.6), black points are drawn from distribution (3.6) with boundaries and red lines are boundaries.

From the plot above, it is capable to constrain a multi-variate Gaussian distribution by a Gibbs sampler.

To show our model is able to constrain the predictions in a given interval, we test its performance on a toy data set where the observed data points are missed totally in a given interval. It is not necessary to compare some statistics such as root-mean-square-error, between our TGP model and GPR model, because the most important target of TGP is to constrain the predictions in a correct interval and then capture information based on training data as much as possible.

Suppose the independent variable is 1-dimensional which is $x \in \mathbb{R}$. Thus, a zero-mean

Gaussian process $\tau(x)$ can be specified with a given covariance function

$$k(x_i, x_j) = v \exp\big(-\frac{1}{2w}(x_i - x_j)^2\big) + a x_i x_j + \delta_{ij}\sigma^2$$

where the hyper-parameters are $\boldsymbol{\theta} = (v, w, a, \sigma)$. The toy data set is generated by $y(x) = \sin(3x)^3 + \tau(x)$ for $x \in (0.01, 0.35) \cup (0.7, 1)$ and $\boldsymbol{\theta} = (0.09, 0.49, 0.04, 0.09)$.

We use the method mentioned in Section 2.1.3 to estimate the hyper-parameters from training data and make predictions for the missing data of $x \in (0.35, 0.7)$ based on Algorithm 4. The results are visualised in Figure 3.3 in which the predictions of GPR are out of boundaries although its predictions near training data are more accurate than predictions of TGP. However, the numerical results present that the TGP model has ability to make a prediction within constrains. As discussed above, it is not reasonable to train an unconstrained model, make a prediction and then constrain the prediction, since all data points on a stochastic process are correlated and we must consider all constrains with the data together rather than just use a model and fit the data. Based on the formula in Algorithm 4, the consequences of Gibbs sampling have a strong relationship with the values of upper boundary and lower boundary, which affects the predictions of both training data and test data. In the future, a possible research could be the selection of boundaries based on training data.

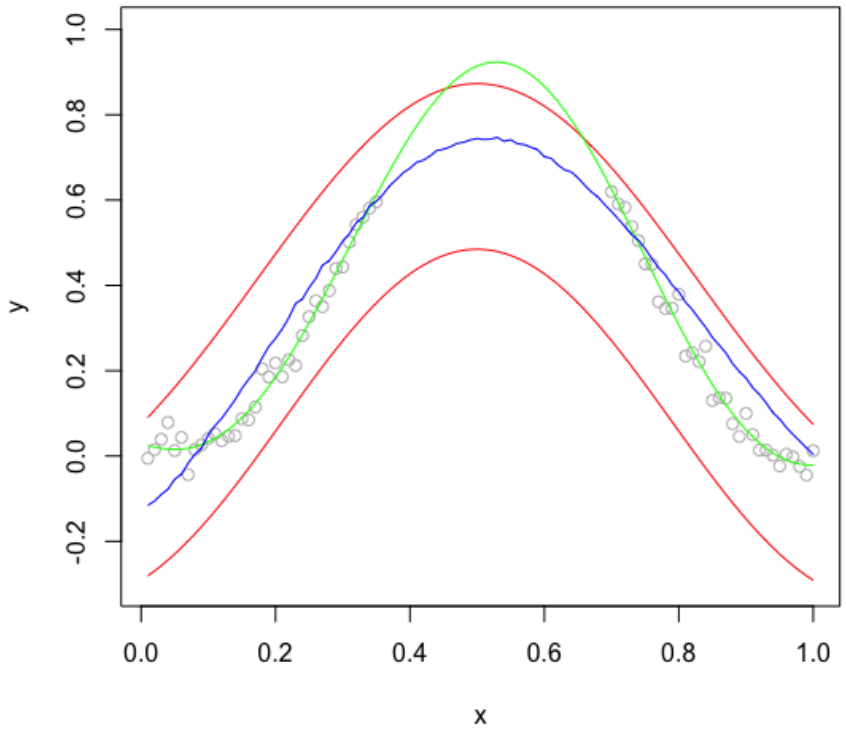


Figure 3.3: Numerical results for comparison where the grey points are training data, green curve is the prediction by Gaussian process regression, blue curve is the prediction by truncated Gaussian process regression, red curves are the lower boundary and upper boundary, respectively.

3.2 An extension of truncated normal distribution

The truncated normal distribution is well studied. However, in practice, a random variable might locate in two intervals of a normal distribution. As shown in Figure 3.4, a random variable locates in interval $[-2, -1]$ and $[1, 2]$ of a standard normal distribution. Since there is little literature on such distribution, in this part, we derive some foundational formulas.

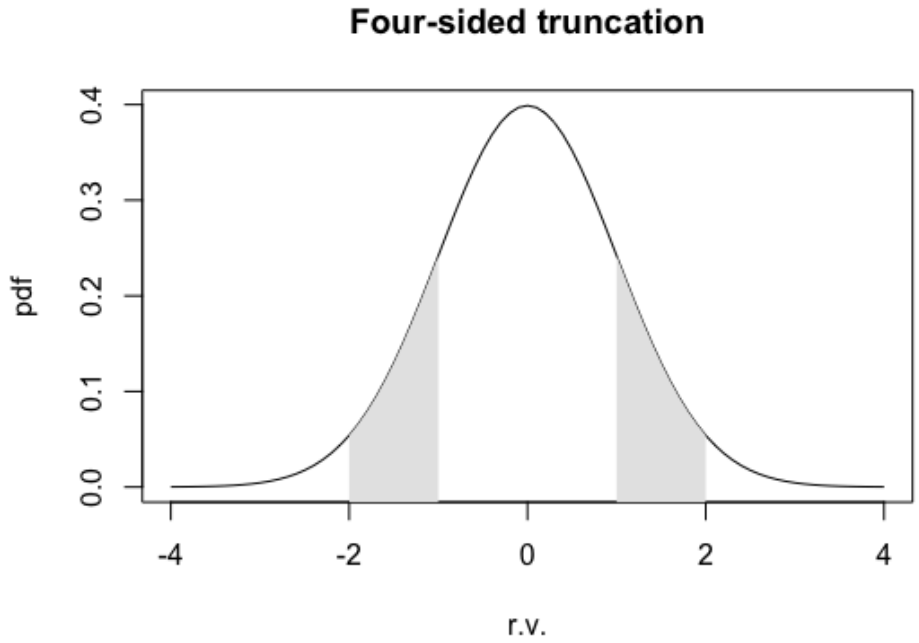


Figure 3.4: A random variable locates in [-2,-1] and [1,2] in a normal distribution.

To develop a truncated normal distribution satisfying the situation we discussed above, a truncated normal distribution can be extended to multi-boundaries. For example, a random variable $x \in [l_1, u_1] \cup [l_2, u_2]$, where $[l_1, u_1]$ and $[l_2, u_2]$ are two intervals in a normal distribution. We also derive some useful formulas on statistics, such as the probability density function, cumulative distribution function, expectation and variance. The formal definition of a multi-truncated normal distribution is given as follows

Definition 3.2.1. *A multi-constrained Gaussian distribution is the probability distribution derived from a normal distributed random variable which is bounded by not only one interval.*

Analogous to the truncated normal distribution, we can write the distribution as

$$X \sim \mathcal{MCN}^n(\mu, \sigma; l_i, u_i),\ i = 1, \ldots, n. \tag{3.7}$$

where n represents the number of constrained intervals where $-\infty \leq l_1 < u_1 < l_2 < \cdots < u_{n-1} < l_n < u_n \leq \infty$. In addition, we suppose the boundaries are symmetric, for example,

when $n = 2$, the distribution is shown in Figure 3.4 where the boundaries are $[-2, -1]$ and $[1, 2]$ which is symmetric from the mean 0.

3.2.1 Probability density function

Naturally, the probability density function is the priority what we need. Inspired by the idea of deriving the p.d.f. of a truncated normal distribution, we can try to infer the formula of normal distribution with multi-boundaries:

$$\begin{aligned} f(x|\mu, \sigma, \boldsymbol{l}, \boldsymbol{u}) &= \frac{1}{n\sigma} \frac{\phi(\xi)}{\sum_{i=1}^{n} (\Phi(\beta_i) - \Phi(\alpha_i))} \\ &= \frac{\phi(\xi)}{n\sigma Z}, \ x \in \bigcup_{i=1}^{n} (l_i, u_i) \end{aligned} \tag{3.8}$$

where $\xi = \frac{x-\mu}{\sigma}$, $\alpha_i = \frac{l_i-\mu}{\sigma}$, $\beta_i = \frac{u_i-\mu}{\sigma}$ and $Z = \sum_{i=1}^{n} \big(\Phi(\beta_i) - \Phi(\alpha_i)\big)$.

3.2.2 Cumulative distribution function

Cumulative distribution function also plays an important role in statistic. The c.d.f. of multi-constrained normal distribution is given by

$$\begin{aligned} F(x|\mu, \sigma, \boldsymbol{l}, \boldsymbol{u}) &= \int_{l_1}^{u_1} f(x|\mu, \sigma, l_1, u_1)dx + \cdots + \int_{l_n}^{u_n} f(x|\mu, \sigma, l_n, u_n)dx, \ x \in \bigcup_{i=1}^{n} (l_i, u_i) \\ &= \sum_{i=1}^{n} \int_{\alpha_i}^{\beta_i} \frac{\phi(\xi)}{n\sigma Z} d\xi \\ &= \frac{\Phi(\xi) - C_m}{nZ} \end{aligned} \tag{3.9}$$

where m represents the random variable x is in the m-th constrained interval (l_m, u_m) and $C_m = \Phi(\alpha_m) - \sum_{j=1}^{m-1} \big(\Phi(\beta_j) - \Phi(\alpha_j)\big)$.

3.2.3 Expectation

The expectation of a random variable is essential in prediction and it can be calculated by:

$$
\begin{aligned}
&E[x|\mu,\sigma,\boldsymbol{l},\boldsymbol{u}],\ x\in(l_i,u_i)\\
=&\int_{l_i}^{u_i} x\frac{\phi(\xi)}{n\sigma Z}dx\\
=&\frac{1}{n\sigma Z}\int_{l_i}^{u_i} x\frac{1}{\sqrt{2\pi}}\exp{(\frac{x-\mu}{\sigma})^2}dx\\
=&\frac{1}{n\sigma Z}\int_{l_i}^{u_i} \sigma\frac{x-\mu+\mu}{\sigma}\frac{1}{\sqrt{2\pi}}\exp{(\frac{x-\mu}{\sigma})^2}dx\\
=&\frac{1}{nZ}\int_{l_i}^{u_i} (\frac{x-\mu}{\sigma}+\frac{\mu}{\sigma})\frac{1}{\sqrt{2\pi}}\exp{(\frac{x-\mu}{\sigma})^2}dx\\
=&\frac{1}{nZ}\int_{l_i}^{u_i} (\frac{x-\mu}{\sigma})\frac{1}{\sqrt{2\pi}}\exp{(\frac{x-\mu}{\sigma})^2}dx+\frac{\mu}{n}\\
=&\frac{\mu}{n}+\frac{\sigma(\phi(\alpha_i)-\phi(\beta_i))}{nZ}
\end{aligned}
\tag{3.10}
$$

3.2.4 Variance

In practice, the variance is also a useful statistics especially in prediction with uncertainty. It is derived by

$$
\begin{aligned}
&E[x^2|\mu,\sigma,\boldsymbol{l},\boldsymbol{u}],\ x\in(l_i,u_i)\\
=&\int_{l_i}^{u_i} x^2\frac{\phi(\xi)}{n\sigma Z}dx\\
=&\frac{1}{n\sigma Z}\int_{l_i}^{u_i} x^2\frac{1}{\sqrt{2\pi}}\exp{(\frac{x-\mu}{\sigma})^2}dx\\
=&\frac{1}{n\sigma Z}\int_{l_i}^{u_i} \sigma^2(\frac{(x-\mu)^2}{\sigma^2}+\frac{2x\mu-\mu^2}{\sigma^2})\frac{1}{\sqrt{2\pi}}\exp{(\frac{x-\mu}{\sigma})^2}dx\\
=&\frac{1}{n\sigma Z}\int_{l_i}^{u_i} \sigma^2\frac{(x-\mu)^2}{\sigma^2}\frac{1}{\sqrt{2\pi}}\exp{(\frac{x-\mu}{\sigma})^2}dx+\frac{2\mu}{nZ}E[x]-\frac{\mu^2}{nZ}\\
=&\sigma^2(\alpha_i\phi(\alpha_i)-\beta_i\phi(\beta_i)+\frac{1}{nZ})+\frac{2\mu}{nZ}E[x]-\frac{\mu^2}{nZ}
\end{aligned}
\tag{3.11}
$$

and thus

$$
\begin{aligned}
& Var(x|\mu,\sigma,\boldsymbol{l},\boldsymbol{u}),\ x\in(l_i,u_i) \\
=& E[x^2]-E[x]^2 \\
=& \sigma^2(\alpha_i\phi(\alpha_i)-\beta_i\phi(\beta_i)+\frac{1}{nZ})+\frac{2\mu}{nZ}E[x]-\frac{\mu^2}{nZ}-(\frac{\mu}{n}+\frac{\sigma(\phi(\alpha_i)-\phi(\beta_i))}{nZ})^2 \\
=& \sigma^2(1+\frac{\alpha_i\phi(\alpha_i)-\beta_i\phi(\beta_i)}{nZ})-\sigma^2(\frac{nZ}{\phi(\alpha_i)-\phi(\beta_i)})^2
\end{aligned}
\tag{3.12}
$$

For more details on the calculation of the expectation and variance, see Johnson et al. (1995).

Researchers can try to develop the truncated Gaussian process with multi-boundaries if interested. However, people should notice that a random variable might jump between different constrained intervals. For example, suppose a random variable follows a truncated Gaussian distribution with two boundaries (l_1,u_1) and (l_2,u_2), the expectation of predictive distribution at t_1 might locate in interval (l_1,u_1) while that of t_2 might locate in interval (l_2,u_2) and the expectation backs to interval (l_1,u_1) for the next time point. In other words, the expectations of t_1 and t_2 might be far away even though t_1 and t_2 are very close.

3.3 Gaussian process regression for Gamma-distributed data

Gamma distribution is a continuous probability function including two parameters, which is a general case of exponential distribution, Erlang distribution and chi-squared distribution. As a member in exponential family, it plays an important role in many applications. For example, the size of insurance claims can be modelled by gamma distribution (Boland, 2007) and rainfalls can also be modelled by gamma distribution (Aksoy, 2000). Additionally, in bacterial gene expression, gamma distribution is used to represent the copy number of constitutively expressed protein (Friedman et al., 2006). However, a basic assumption in Gaussian process regression is that a response variable follows a normal distribution. In this part, our motivation is to extend Gaussian process regression for Gamma-distributed data.

In practice, people might use log-transformation which maps the data to an approx-

imate Gaussian distribution and then apply GPR to fit the mapped data. Nevertheless, the mapped data are still non-Gaussian and thus it is not reasonable to use GPR directly. For comparison, we also shown the numerical result for this method in Section 3.3.4.

3.3.1 The model

We suppose the training data are $\mathcal{D} = \{(\boldsymbol{x}_i, y_i) | \boldsymbol{x}_i \in \mathbb{R}^p, y_i \in \mathbb{R}, i = 1, ..., n\}$ in which $\boldsymbol{x}_i$ refers to a p-dimensional vector of predictor variable and y_i refers to a Gamma-distributed response variable. Since a Gamma distribution depends on two parameters k and θ. Parameters k and θ are appeared more common in econometrics while parameters k and $\beta = \frac{1}{\theta}$ are appeared more common in Bayesian statistics. Therefore, in both fields, people seems pay more attention to θ in a Gamma distribution. That is the reason we model θ by a GP and treat k as a constant. Therefore, we suppose k is a constant which can be estimated by cross-validation and θ, a random variable, is estimated by Gaussian process regression. In other words, θ can be considered as a latent process and thus, we use Gaussian process to model this latent process.

Since the parameter θ is positive, we use an exponential form to represent it. We define the Gaussian process regression model for Gamma-distribution data as

$$\begin{aligned} y_i &\sim Gamma(k, \theta_i) \\ &= Gamma(k, e^{f(\boldsymbol{x}_i)}), \ i = 1, ..., n \end{aligned} \tag{3.13}$$

where $f(\cdot)$ refers to a Gaussian process. The mean function of $f(\cdot)$ is considered as zero and the covariance function is selected in advance, such as squared exponential covariance function or linear covariance function. Inspired from Wang and Shi (2014), since the likelihood function is analytically intractable, we use Laplace method or Gaussian approximation to approximate it and the predictive distribution has an analytical form which is computational efficiently. However, the prediction is not accurate due to the Gaussian approximation ignores the non-symmetric property of Gamma distribution. Therefore, in order to improve the predictive performance of our model, we use approximate Bayesian computation for prediction rather than the analytical form. The numerical results show that with the application of ABC, the accuracy of prediction is improved.

3.3.2 Inference based on Laplace method

One important goal in objective function optimization is to estimate the parameters in a model. As a typical non-parametric model, we need to estimate the hyper-parameters in covariance function of a Gaussian process regression model.

We firstly discuss an approach on model learning and predicting based on empirical Bayesian method and Laplace approximation (Williams and Rasmussen, 2006). With the assumption that response variables are independent pairwise, the likelihood function can be derived by the products of probability density functions of all response variables:

$$\begin{aligned} p(\boldsymbol{y}|\boldsymbol{f}) &= \frac{1}{\Gamma(k)e^{f_1 k}} y_1^{k-1} e^{-\frac{y_1}{e^{f_1}}} \times \cdots \times \frac{1}{\Gamma(k)e^{f_n k}} y_n^{k-1} e^{-\frac{y_n}{e^{f_n}}} \\ &= \prod_{i=1}^{n} \frac{1}{\Gamma(k)e^{f_i k}} y_i^{k-1} e^{-\frac{y_i}{e^{f_i}}} \end{aligned} \tag{3.14}$$

where $\Gamma(\cdot)$ is the *Gamma function*, f_i is the short form of $f(\boldsymbol{x}_i)$, $\boldsymbol{y} = (y_1, ..., y_n)$ and $\boldsymbol{f} = (f_1, ..., f_n)$.

To estimate the hyper-parameters $\boldsymbol{\theta}$ in covariance function of a Gaussian process $f(\cdot)$ by empirical Bayesian learning, we need the marginal distribution of $\boldsymbol{y}$ which can be computed by integrating the conditional distribution $p(\boldsymbol{y}|\boldsymbol{f})$ and the probability density function $p(\boldsymbol{f}|X, \boldsymbol{\theta})$ over $\boldsymbol{f}$. Thus, the marginal distribution can be written by

$$p(\boldsymbol{y}|X, \boldsymbol{\theta}) = \int p(\boldsymbol{y}|\boldsymbol{f}) p(\boldsymbol{f}|X, \boldsymbol{\theta})\, d\boldsymbol{f} \tag{3.15}$$

where $X = (\boldsymbol{x}_1, ..., \boldsymbol{x}_n)$.

The marginal log-likelihood is given as

$$\begin{aligned} l(\boldsymbol{\theta}) &= \log(p(\boldsymbol{y}|X, \boldsymbol{\theta})) \\ &= \log(\int p(\boldsymbol{y}|\boldsymbol{f}) p(\boldsymbol{f}|X, \boldsymbol{\theta})\, d\boldsymbol{f}) \\ &= \log(\int e^{\gamma(\boldsymbol{f})}\, d\boldsymbol{f}) \end{aligned} \tag{3.16}$$

where $\gamma(\boldsymbol{f}) = \log(p(\boldsymbol{y}|\boldsymbol{f})) + \log(p(\boldsymbol{f}|X, \boldsymbol{\theta}))$.

It is very difficult to derive an analytical form of the integral in Equation (3.16), which is also a widely-existed problem in calculating a posterior distribution. In this section,

instead of computing an analytical form of this marginal log-likelihood function, we use Laplace method to approximate the integral, which is

$$l(\boldsymbol{\theta}) \approx \gamma(\hat{\boldsymbol{f}}) + \frac{n}{2}\log(2\pi) - \frac{1}{2}\log|B + \Psi^{-1}| \tag{3.17}$$

where $\hat{\boldsymbol{f}}$ maximises the $\gamma(\boldsymbol{f})$. The hyper-parameters $\boldsymbol{\theta}$ can be estimated by empirical Bayesian learning which maximises the approximate marginal log-likelihood function (3.17). However, it is noticeable that $\gamma(\boldsymbol{f})$ depends on $\boldsymbol{\theta}$ and $l(\boldsymbol{\theta})$ depends on $\gamma(\hat{\boldsymbol{f}})$. Therefore, we can use an iterative algorithm to estimate $\boldsymbol{\theta}$.

Algorithm 5: Iterative algorithm for estimating hyper-parameters $\boldsymbol{\theta}$.

1. Select the initial value of hyper-parameters $\boldsymbol{\theta}^{(j)}$ and set $j = 0$;
2. Calculate $\boldsymbol{f}^{(j)}$ by maximizing $\gamma(\boldsymbol{f})$ given $\boldsymbol{\theta}^{(j)}$;
3. Calculate $\boldsymbol{\theta}^{(j+1)}$ by maximizing $l(\boldsymbol{\theta})$ in (3.17) given $\boldsymbol{f}^{(j)}$;
4. $j = j + 1$, replicate Step 2 and Step 3 until both $\boldsymbol{f}$ and $\boldsymbol{\theta}$ converge.

The first derivative and second derivative of $\gamma(\boldsymbol{f})$ with respect to $\boldsymbol{f}$ and $l(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ are essential to implement Step 2 and Step 3 efficiently in Algorithm 5. The derivatives are given as

$$\frac{\partial\gamma(\boldsymbol{f})}{\partial\boldsymbol{f}} = (\frac{y_1}{e^{f_1}}, \ldots, \frac{y_n}{e^{f_n}}) - \boldsymbol{k} - \Psi^{-1}\boldsymbol{f}$$

$$\frac{\partial^2\gamma(\boldsymbol{f})}{\partial\boldsymbol{f}\partial\boldsymbol{f}^T} = -B - \Psi^{-1}$$

where $\boldsymbol{k} = (k, k, \ldots, k)$ is a n-dimensional vector, $B = \mathrm{diag}(\frac{y_1}{e^{f_1}}, \ldots, \frac{y_n}{e^{f_n}})$ is a $n \times n$ diagonal matrix and Ψ^{-1} is the inverse covariance matrix.

For simplicity, we suppose there are only two hyper-parameters $\boldsymbol{\theta} = (\theta_1, \theta_2)$ in covariance function and the first derivatives and second derivatives of $l(\boldsymbol{\theta})$ with respect to θ_1 and θ_2 are given as

$$\frac{\partial l(\boldsymbol{\theta})}{\partial\theta_1} = -\frac{1}{2}\mathrm{tr}(\Psi^{-1}\frac{\partial\Psi}{\partial\theta_1}) + \frac{1}{2}\boldsymbol{y}\Psi^{-1}\frac{\partial\Psi}{\partial\theta_1}\Psi^{-1}\boldsymbol{y} - \frac{1}{2}\mathrm{tr}\big((B + \Psi^{-1})^{-1}\Psi^{-1}\frac{\partial\Psi}{\partial\theta_1}\Psi^{-1}\big)$$

$$\frac{\partial l(\boldsymbol{\theta})}{\partial \theta_2} = -\frac{1}{2}\mathrm{tr}(\Psi^{-1}\frac{\partial \Psi}{\partial \theta_2}) + \frac{1}{2}\boldsymbol{y}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_2}\Psi^{-1}\boldsymbol{y} - \frac{1}{2}\mathrm{tr}\big((B+\Psi^{-1})^{-1}\frac{\partial \Psi}{\partial \theta_2}\Psi^{-1}\big)$$

$$\begin{aligned}\frac{\partial^2 l(\boldsymbol{\theta})}{\partial \theta_1^2} = \frac{1}{2}\mathrm{tr}(\Psi^{-1}\frac{\partial \Psi}{\partial \theta_1}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_1}) - \frac{1}{2}\mathrm{tr}(\Psi^{-1}\frac{\partial^2 \Psi}{\partial \theta_1^2}) - \frac{1}{2}\boldsymbol{y}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_1}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_1}\Psi^{-1}\boldsymbol{y} \\ -\frac{1}{2}\boldsymbol{y}\Psi^{-1}\frac{\partial^2 \Psi}{\partial \theta_1^2}\Psi^{-1}\boldsymbol{y} + \frac{1}{2}\boldsymbol{y}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_1}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_1}\Psi^{-1}\boldsymbol{y} \\ +\frac{1}{2}\mathrm{tr}\big((B+\Psi^{-1})^{-1}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_1}\Psi^{-1}(B+\Psi^{-1})^{-1}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_1}\Psi^{-1} \\ -(B+\Psi^{-1})^{-1}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_1}\Psi^{-1} + (B+\Psi^{-1})^{-1}\Psi^{-1}\frac{\partial^2 \Psi}{\partial \theta_1^2} \\ -(B+\Psi^{-1})^{-1}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_1}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_1}\Psi^{-1}\big)\end{aligned}$$

$$\begin{aligned}\frac{\partial^2 l(\boldsymbol{\theta})}{\partial \theta_1 \partial \theta_2} = \frac{1}{2}\mathrm{tr}(\Psi^{-1}\frac{\partial \Psi}{\partial \theta_2}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_1}) - \frac{1}{2}\mathrm{tr}(\Psi^{-1}\frac{\partial^2 \Psi}{\partial \theta_1 \partial \theta_2}) - \frac{1}{2}\boldsymbol{y}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_2}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_1}\Psi^{-1}\boldsymbol{y} \\ -\frac{1}{2}\boldsymbol{y}\Psi^{-1}\frac{\partial^2 \Psi}{\partial \theta_1 \partial \theta_2}\Psi^{-1}\boldsymbol{y} + \frac{1}{2}\boldsymbol{y}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_1}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_2}\Psi^{-1}\boldsymbol{y} \\ +\frac{1}{2}\mathrm{tr}\big((B+\Psi^{-1})^{-1}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_2}\Psi^{-1}(B+\Psi^{-1})^{-1}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_1}\Psi^{-1} \\ -(B+\Psi^{-1})^{-1}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_2}\Psi^{-1} + (B+\Psi^{-1})^{-1}\Psi^{-1}\frac{\partial^2 \Psi}{\partial \theta_1 \partial \theta_2} \\ -(B+\Psi^{-1})^{-1}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_1}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_2}\Psi^{-1}\big)\end{aligned}$$

$$\frac{\partial^2 l(\boldsymbol{\theta})}{\partial \theta_2^2} = \frac{1}{2}\mathrm{tr}(\Psi^{-1}\frac{\partial \Psi}{\partial \theta_2}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_2}) - \frac{1}{2}\mathrm{tr}(\Psi^{-1}\frac{\partial^2 \Psi}{\partial \theta_2^2}) - \frac{1}{2}\boldsymbol{y}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_2}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_2}\Psi^{-1}\boldsymbol{y}$$
$$-\frac{1}{2}\boldsymbol{y}\Psi^{-1}\frac{\partial^2 \Psi}{\partial \theta_2^2}\Psi^{-1}\boldsymbol{y} + \frac{1}{2}\boldsymbol{y}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_2}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_2}\Psi^{-1}\boldsymbol{y}$$
$$+\frac{1}{2}\mathrm{tr}\big((B+\Psi^{-1})^{-1}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_2}\Psi^{-1}(B+\Psi^{-1})^{-1}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_2}\Psi^{-1}$$
$$-(B+\Psi^{-1})^{-1}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_2}\Psi^{-1} + (B+\Psi^{-1})^{-1}\Psi^{-1}\frac{\partial^2 \Psi}{\partial \theta_2^2}$$
$$-(B+\Psi^{-1})^{-1}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_2}\Psi^{-1}\frac{\partial \Psi}{\partial \theta_2}\Psi^{-1}\big)$$

However, Rue et al. (2009) demonstrate that asymptotic error rate of Equation (3.17) is $O(1)$ since the dimension of $\boldsymbol{f}$ increases with the sample size n. Moreover, the derivatives above contain many inverse matrices which are computational expensively. In order to avoid these troubles, we use another approximate method to estimate hyper-parameters $\boldsymbol{\theta}$ which is based on Gaussian approximation.

3.3.3 Inferences based on Gaussian approximation

Instead of calculating the marginal log-likelihood $l(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log\{p(y_i|k,\boldsymbol{\theta})\}$ via Laplace method, we consider a more efficient method (Wang and Shi, 2014), which provides a simpler form of likelihood function and thus requires less calculation in derivatives of likelihood function. The assumption of data in their paper is different from us, which leads to different likelihood function and predictive distribution.

Specifically, we represent $p(\boldsymbol{y}|k,\boldsymbol{\theta})$ as follows

$$\tilde{p}(\boldsymbol{y}|k,\boldsymbol{\theta}) \triangleq \left.\frac{p(\boldsymbol{f},\boldsymbol{y}|k,\boldsymbol{\theta})}{\tilde{p}_G(\boldsymbol{f}|\boldsymbol{y},k,\boldsymbol{\theta})}\right|_{\boldsymbol{f}=\tilde{\boldsymbol{f}}} \tag{3.18}$$

where $\tilde{p}_G(\boldsymbol{f}|\boldsymbol{y},k,\boldsymbol{\theta})$ is the Gaussian approximation to the full conditional density $p(\boldsymbol{f}|\boldsymbol{y},k,\boldsymbol{\theta})$ and $\tilde{\boldsymbol{f}}$ is the mode of the full conditional density of $\boldsymbol{f}$ for a given $\boldsymbol{\theta}$.

The numerator of Equation (3.18), which is the joint distribution of response variables

$\boldsymbol{y}$ and latent variables $\boldsymbol{f}$, can be decomposed by

$$\begin{aligned} p(\boldsymbol{f}, \boldsymbol{y}|k, \boldsymbol{\theta}) &= p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{\theta}) p(\boldsymbol{f}|\boldsymbol{\theta}) \\ &= \exp\Big(\log\big(p(\boldsymbol{f}|\boldsymbol{\theta})\big) + \sum_{i=1}^{n} \log\big(p(y_i|f_i, k)\big) \Big) \end{aligned} \tag{3.19}$$

where $p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{\theta})$ is the product of gamma distributions and $p(\boldsymbol{f}|\boldsymbol{\theta})$ is a multi-variate Gaussian distribution. In order to maintain $p(\boldsymbol{f}, \boldsymbol{y}|k, \boldsymbol{\theta})$ in the framework of Gaussian distribution, we approximate $\log\big(p(y_i|f_i, k)\big)$ by Taylor expression to the second order at $f_i^{(0)}$, that is

$$g_i(f_i) \approx g_i(f_i^{(0)}) + \frac{g_i'(f_i^{(0)})}{1!}(f_i - f_i^{(0)}) + \frac{g_i''(f_i^{(0)})}{2!}(f_i - f_i^{(0)})^2 \tag{3.20}$$

where $g_i(f_i) = \log(p(y_i|f_i, k))$. By defining

$$a_i = \frac{y_i}{e^{f_i^{(0)}}} + f_i^{(0)} \frac{y_i}{e^{f_i^{(0)}}} - k$$
$$d_i = \frac{y_i}{e^{f_i^{(0)}}}$$

$g_i(f_i)$ can be approximated as follows

$$g_i(f_i) \approx g_i(f_i^{(0)}) + a_i f_i - \frac{1}{2} d_i f_i^2 \tag{3.21}$$

Therefore, we obtain an approximation of the joint distribution $p(\boldsymbol{f}, \boldsymbol{y}|k, \boldsymbol{\theta})$, i.e.

$$p(\boldsymbol{f}, \boldsymbol{y}|\boldsymbol{\theta}, k) \propto \exp\{-\frac{1}{2}\boldsymbol{f}^T \Psi^{-1} \boldsymbol{f} - \frac{1}{2}\boldsymbol{f}^T D \boldsymbol{f} + \boldsymbol{a}^T \boldsymbol{f}\} \tag{3.22}$$

where $\boldsymbol{a} = (a_1, ..., a_n)$, $D = \text{diag}(d_1, ..., d_n)$ and Ψ is the covariance matrix of multivariate normal distribution $\boldsymbol{f}$ with hyper-parameters $\boldsymbol{\theta}$. Then, based on the Fisher scoring algorithm (Fahrmeir and Lang, 2001), the Gaussian approximation can be computed:

Algorithm 6: Fisher scoring algorithm for estimating Gaussian approximation.

1. Select the initial value of every $f_i^{(j)}$ and set $j = 0$;
2. Calculate $\boldsymbol{f}^{(j)}$ from $(\Psi^{-1} + D)\boldsymbol{f}^{(j)} = \boldsymbol{a}$;
3. Update $\boldsymbol{a}$ and D by $\boldsymbol{f}^{(j)}$;
4. $j = j + 1$, replicate Step 2 and Step 3 until this process converges at $\tilde{\boldsymbol{f}}$;
5. We get the Gaussian approximation of the full conditional distribution which is

$$\tilde{p}_G(\boldsymbol{f}|\boldsymbol{y}, k, \boldsymbol{\theta}) \sim \mathcal{N}(\tilde{\boldsymbol{f}}, (\Psi^{-1} + D)^{-1}) \tag{3.23}$$

Equipped with distributions in Equation (3.22) and (3.23), the hyper-parameters $\boldsymbol{\theta}$ can be estimated by maximising the approximate marginal log-likelihood (3.18) which can be formulated as

$$\begin{aligned} l(\boldsymbol{\theta}) &= \log(p(\boldsymbol{y}|X, k, \boldsymbol{\theta})) \\ &\approx \log(\tilde{p}(\boldsymbol{y}|k, \boldsymbol{\theta})) \\ &= \log(\frac{p(\tilde{\boldsymbol{f}}, \boldsymbol{y}|k, \boldsymbol{\theta})}{\tilde{p}_G(\tilde{f}|\boldsymbol{y}, k, \boldsymbol{\theta})}) \\ &= \log(p(\tilde{\boldsymbol{f}}, \boldsymbol{y}|k, \boldsymbol{\theta})) - \log(\tilde{p}_G(\tilde{\boldsymbol{f}}|\boldsymbol{y}, k, \boldsymbol{\theta})) \\ &\propto -\log(|(\Psi^{-1} + D)^{-1}|) \end{aligned} \tag{3.24}$$

In practice, we use differential evolution algorithm to estimate $\boldsymbol{\theta}$. Another important inference in statistics is to calculate the predictive expectation and predictive variance. Suppose the independent variable for test data is given by $\boldsymbol{x}^* \in \mathbb{R}^p$, thus, the expectation is computed by

$$E[y^*|\mathcal{D}] = \int E[y^*|f^*, \mathcal{D}]p(f^*|\mathcal{D})df^* \tag{3.25}$$

where $f^* = f(\boldsymbol{x}^*)$ and $y^* = \frac{1}{\Gamma(k)e^{f(\boldsymbol{x}^*)k}} y_i^{k-1} e^{-\frac{y_i}{f(\boldsymbol{x}_i)}}$. Since $(\boldsymbol{f}, f^*)$ is jointly Gaussian distributed, f^* still follows a normal distribution

$$f^*|\mathcal{D} \sim N(\boldsymbol{b}^T\tilde{\boldsymbol{f}}, \boldsymbol{b}^T\Omega\boldsymbol{b} + \sigma^{*2}) \tag{3.26}$$

where $\boldsymbol{b}^T = \boldsymbol{\psi}(\boldsymbol{x}^*)^T\Psi^{-1}$, $\sigma^* = \phi(\boldsymbol{x}^*) - \boldsymbol{\psi}(\boldsymbol{x}^*)^T\Psi^{-1}\boldsymbol{\psi}(\boldsymbol{x}^*)$ and $\Omega = (\Psi^{-1} + D)^{-1}$. As a consequence, the integral in Equation (3.25) has an analytical form, that is

$$E[y^*|\mathcal{D}] = k \times \exp\big(\frac{-(\boldsymbol{b}^T\tilde{\boldsymbol{f}})^2 + (\boldsymbol{b}^T\Omega\boldsymbol{b} + \sigma^{*2} + \boldsymbol{b}^T\tilde{\boldsymbol{f}})^2}{2(\boldsymbol{b}^T\Omega\boldsymbol{b} + \sigma^{*2})}\big) \tag{3.27}$$

The variance of $y^*|\mathcal{D}$ can be yielded by

$$Var(y^*|\mathcal{D}) = E[Var(y^*|f^*, \mathcal{D})] + Var(E[y^*|f^*, \mathcal{D}]) \tag{3.28}$$

where

$$E[Var(y^*|f^*, \mathcal{D})] = \int Var[y^*|f^*, \mathcal{D}]p(f^*|\mathcal{D})df^*$$
$$Var(E[y^*|f^*, \mathcal{D}]) = \int E[y^*|f^*, \mathcal{D}]^2 p(f^*|\mathcal{D})df^* - E[y^*|\mathcal{D}]^2$$

Since $y^*|f^*, \mathcal{D}$ is gamma distributed, we have

$$E[y^*|f^*, \mathcal{D}] = ke^{f^*}$$
$$Var(y^*|f^*, \mathcal{D}) = ke^{2f^*}$$

Therefore,

$$E[Var(y^*|f^*, \mathcal{D})] = k \times \exp\big(\frac{(2(\boldsymbol{a}^T\Omega\boldsymbol{a} + \sigma^{*2}) + \boldsymbol{a}^T\tilde{\boldsymbol{f}})^2 - (\boldsymbol{a}^T\tilde{\boldsymbol{f}})^2}{2(\boldsymbol{a}^T\Omega\boldsymbol{a} + \sigma^{*2})}\big)$$

$$\begin{aligned} Var(E[y^*|f^*, \mathcal{D}]) = k^2 \times \exp\big(&\frac{-(\boldsymbol{b}^T\tilde{\boldsymbol{f}})^2 + (\boldsymbol{b}^T\Omega\boldsymbol{b} + \sigma^{*2} + \boldsymbol{b}^T\tilde{\boldsymbol{f}})^2}{\boldsymbol{b}^T\Omega\boldsymbol{b} + \sigma^{*2}}\big) \\ - k^2 \times \exp\big(&\frac{(2(\boldsymbol{b}^T\Omega\boldsymbol{b} + \sigma^{*2}) + \boldsymbol{b}^T\tilde{\boldsymbol{f}})^2 - (\boldsymbol{b}^T\tilde{\boldsymbol{f}})^2}{\boldsymbol{b}^T\Omega\boldsymbol{b} + \sigma^{*2}}\big) \end{aligned}$$

And fortunately, we obtain an analytical form of predictive variance

$$\begin{aligned} Var(y^*|\mathcal{D}) = k \times \exp\big(&\frac{(2(\boldsymbol{b}^T\Omega\boldsymbol{b}+\sigma^{*2})+\boldsymbol{b}^T\tilde{\boldsymbol{f}})^2-(\boldsymbol{b}^T\tilde{\boldsymbol{f}})^2}{2(\boldsymbol{b}^T\Omega\boldsymbol{b}+\sigma^{*2})}\big) \\ + k^2 \times \exp\big(&\frac{-(\boldsymbol{b}^T\tilde{\boldsymbol{f}})^2+(\boldsymbol{b}^T\Omega\boldsymbol{b}+\sigma^{*2}+\boldsymbol{b}^T\tilde{\boldsymbol{f}})^2}{\boldsymbol{b}^T\Omega\boldsymbol{b}+\sigma^{*2}}\big) \\ - k^2 \times \exp\big(&\frac{(2(\boldsymbol{b}^T\Omega\boldsymbol{b}+\sigma^{*2})+\boldsymbol{b}^T\tilde{\boldsymbol{f}})^2-(\boldsymbol{b}^T\tilde{\boldsymbol{f}})^2}{\boldsymbol{b}^T\Omega\boldsymbol{b}+\sigma^{*2}}\big) \end{aligned} \tag{3.29}$$

The Gaussian approximation $\tilde{p}_G(\boldsymbol{f}|\boldsymbol{y},k,\boldsymbol{\theta})$ follows a normal distribution which is symmetric. However, according to empirical results, $p(\boldsymbol{f}|\boldsymbol{y},k,\boldsymbol{\theta})$ shows a unsymmetric distribution which leads to inaccurate predictions if we simply replace $p(\boldsymbol{f}|\boldsymbol{y},k,\boldsymbol{\theta})$ by $\tilde{p}_G(\boldsymbol{f}|\boldsymbol{y},k,\boldsymbol{\theta})$. To address this issue, we use approximate Bayesian computation to $p(\boldsymbol{f}|\boldsymbol{y},k,\boldsymbol{\theta})$ rather than using the Gaussian approximation directly.

3.3.4 Approximate Bayesian computation for prediction

Approximate Bayesian computation consists of many computational methods to estimate the posterior distribution in Bayesian statistic, which can be traced back to Rubin (1984). The most important advantage of ABC is that this algorithm bypasses the likelihood function. Likelihood function is of central importance since it describes the probability of data given a statistical model. People can derive an analytical form of likelihood function for simple models. However, for complex models, the likelihood function is often computational expensively or even elusive, such as $l(\boldsymbol{\theta})$ in Equation (3.16). Therefore, we can use ABC to estimate the posterior distribution approximately without deriving the likelihood function.

In ABC rejection algorithm, we first sample from the prior distribution of hyper-parameters $\boldsymbol{\theta}$ which is denoted by $\hat{\boldsymbol{\theta}}$. Besides, the simulated training data set $\hat{\mathcal{D}}$ can be specified by the given statistical model and $\hat{\boldsymbol{\theta}}$. If the distance, such as Euclidean norm, between the simulated training data $\hat{\mathcal{D}}$ and the real training data $\mathcal{D}$ is larger than a strictly positive tolerance, the sampled hyper-parameters $\hat{\boldsymbol{\theta}}$ is discarded; otherwise, we accept $\hat{\boldsymbol{\theta}}$. After a number of replications, without deriving the exact likelihood function, ABC rejection algorithm provides an approximate posterior distribution of targeted parameters.

In this section, we aim to apply approximate Bayesian computation for prediction

rather than estimation and as mentioned previously, the predictive expectation is given by

$$\begin{aligned} E[y^*|\mathcal{D}] &= \int E[y^*|f^*, \mathcal{D}]p(f^*|\mathcal{D})df^* \\ &= \int ke^{f^*}p(f^*|\mathcal{D})df^* \\ &\approx \sum_{h=1}^{N} ke^{f^{*h}} \end{aligned}$$

and the predictive variance is

$$\begin{aligned} Var(y^*|\mathcal{D}) &= E[Var(y^*|f^*, \mathcal{D})] + Var(E[y^*|f^*, \mathcal{D}]) \\ &= \int Var(y^*|f^*, \mathcal{D})p(f^*|\mathcal{D})df^* + \int E[y^*|f^*, \mathcal{D}]^2 p(f^*|\mathcal{D})df^* - E[y^*|\mathcal{D}]^2 \\ &\approx \sum_{h=1}^{N} ke^{2f^{*h}} + \sum_{h=1}^{N} k^2 e^{2f^{*h}} - \sum_{h=1}^{N} k^2 e^{2f^{*h}} \\ &= \sum_{h=1}^{N} ke^{2f^{*h}} \end{aligned}$$

which depend on the sample of f^*. In addition,

$$\begin{aligned} p(f^*|\mathcal{D}) &= \int p(f^*|\boldsymbol{f}, \mathcal{D})p(\boldsymbol{f}|\mathcal{D})d\boldsymbol{f} \\ &\approx \sum_{h=1}^{N} p(f^*|\boldsymbol{f}^h) \end{aligned}$$

where $p(f^*|\boldsymbol{f})$ is normal distributed and N refers to a large integer. Therefore, our aim converges to find $p(\boldsymbol{f}|\mathcal{D})$ by approximate Bayesian computation.

Instead of sampling from prior distribution of $\boldsymbol{\theta}$, we draw $\boldsymbol{f}$ from its approximated full conditional distribution (3.23). Then, the simulated training data $\hat{\boldsymbol{y}}$ can be calculated by the expectation of Gamma distribution. The goodness of $\boldsymbol{f}$ is determined by the distance between $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$. If the distance is smaller than a positive tolerance, we suppose this sample $\boldsymbol{f}$ might be drawn from the real posterior distribution and store the value of $\boldsymbol{f}$ and repeat this process for a number of times, which leads to an approximate posterior distribution of $p(\boldsymbol{f}|\mathcal{D})$. We can draw lots of samples from $p(\boldsymbol{f}|\mathcal{D})$ and calculate $p(f^*|\mathcal{D})$ which is used for prediction $E[y^*|\mathcal{D}]$. Algorithm 7 summaries these steps.

Algorithm 7: Approximate Bayesian Computation for Prediction.

1. Draw $\boldsymbol{f}^{(j)} = (f_1^{(j)}, ..., f_n^{(j)})$ from the Gaussian approximation $\tilde{p}_G(\boldsymbol{f}|\boldsymbol{y}, k, \boldsymbol{\theta})$ in Equation (3.23) and set $j = 0$;
2. Draw $\hat{y}_i$ form $Gamma(k, \exp(f_i^{(j)}))$;
3. Calculate the distance between the simulated training data $\hat{\boldsymbol{y}}$ and the observed data $\boldsymbol{y}$. If $d(\boldsymbol{y}, \hat{\boldsymbol{y}}) < \epsilon$, where ϵ is a strictly positive tolerance, we accept and store $\boldsymbol{f}^{(j)}$; else, we reject $\boldsymbol{f}^{(j)}$. $j = j + 1$ and back to Step 1 until j reaches a large number. The distribution of stored values is written by $p_{ABC}(\boldsymbol{f}|\mathcal{D})$;
4. Draw $\boldsymbol{f}^{(r)}$ from $p_{ABC}(\boldsymbol{f}|\mathcal{D})$, then, $p(f^*|\mathcal{D}) \approx \sum_{r=1}^{R} p(f^*|\boldsymbol{f}^{(r)})$ with appropriately large R ;
5. Repeat Step 4 N times, we get $p(f^{*(1)}|\mathcal{D}), ..., p(f^{*(N)}|\mathcal{D})$'s. Then, $E[y^*|\mathcal{D}] \approx \sum_{h=1}^{N} k e^{f^{*(h)}}$ and $Var(y^*|\mathcal{D}) \approx \sum_{h=1}^{N} k e^{2f^{*(h)}}$ where $f^{*(h)}$ is drawn from $p(f^{*(h)})$.

We discussed the prediction of Gaussian process regression for Gamma-distributed data by approximate Bayesian computation which neglects the exact form of likelihood function. By taking advantage of Gaussian approximation of full conditional density $p(\boldsymbol{f}|\boldsymbol{y}, k, \boldsymbol{\theta})$, the algorithm can be applied efficiently. Specifically, the accuracy of prediction is improved via Algorithm 7 compared to the solution in Equation (3.27).

3.3.5 Convolved Gaussian processes regression for multi-variate Gamma-distributed data

The Gaussian process regression for Gamma-distributed data can solve a class of regression problems with probabilistic predictions. However, the response variable is restricted to one dimension. To model multi-dimensional Gamma-distributed data by GPR, we can model each dimension independently. Nevertheless, this idea cannot capture the correlation between dependent variables in different dimensions. Therefore, we use convolved Gaussian processes, see Section 2.2, to model the latent processes. In the case of 2-dimensional response variables, suppose the training data are denoted as $\mathcal{D} = \{(\boldsymbol{x}_{1,i}, y_{1,i}), (\boldsymbol{x}_{2,j}, y_{2,j}) | \boldsymbol{x}_{1,i} \in \mathbb{R}^p, \boldsymbol{x}_{2,j} \in \mathbb{R}^q, y_{1,i} \in \mathbb{R}, y_{2,j} \in \mathbb{R}, i = 1, ..., n_1, j = 1, ..., n_2\}$ and k_1, k_2 are given for the first dimensional output and the second dimensional output respectively. Moreover, it is not necessary to assume $n_1 = n_2$, since we model two cor-

related stochastic process rather than a bi-variate random variable. However, for more application, n_1 is not necessarily equal to n_2 as discussed in Section 2.2. The model is defined as

$$\begin{aligned} y_1(\boldsymbol{x}_i) &\sim Gamma(k_1, \exp(f_1(\boldsymbol{x}_i))), \\ y_2(\boldsymbol{x}_i) &\sim Gamma(k_2, \exp(f_2(\boldsymbol{x}_i))) \end{aligned} \tag{3.30}$$

where k_1, k_2 are constants and $(f_1(\boldsymbol{x}_i), f_2(\boldsymbol{x}_i))$ follows a convolved Gaussian process. The likelihood function is written by

$$\begin{aligned} p(\boldsymbol{y}|\boldsymbol{f}) &= \frac{1}{\Gamma(k_1)e^{f_{1,1}k_1}} y_{1,1}^{k_1-1} e^{-\frac{y_{1,1}}{e^{f_{1,1}}}} \times \cdots \times \frac{1}{\Gamma(k_1)e^{f_{1,n_1}k_1}} y_{1,n_1}^{k_1-1} e^{-\frac{y_{1,n_1}}{e^{f_{1,n_1}}}} \times \\ &\quad \frac{1}{\Gamma(k_2)e^{f_{2,1}k_2}} y_{2,1}^{k_2-1} e^{-\frac{y_{2,1}}{e^{f_{2,1}}}} \times \cdots \times \frac{1}{\Gamma(k_2)e^{f_{2,n_2}k_2}} y_{2,n_2}^{k_2-1} e^{-\frac{y_{2,n_2}}{e^{f_{2,n_2}}}} \\ &= \prod_{i=1}^{n_1} \frac{1}{\Gamma(k_1)e^{f_{1,i}k_1}} y_{1,i}^{k_1-1} e^{-\frac{y_{1,i}}{e^{f_{1,i}}}} \times \prod_{i=1}^{n_2} \frac{1}{\Gamma(k_2)e^{f_{2,i}k_2}} y_{2,i}^{k_2-1} e^{-\frac{y_{2,i}}{e^{f_{2,i}}}} \end{aligned} \tag{3.31}$$

where $\Gamma(\cdot)$ is the Gamma function, $\boldsymbol{y} = (y_{1,1}, ..., y_{1,n_1}, y_{2,1}, ..., y_{2,n_2})$ and $\boldsymbol{f} = (f_{1,1}, ..., f_{1,n_1}, f_{2,1}, ..., f_{2,n_2})$.

As discussed in Section 3.3.2, it is computational expensively to derive the first order and second order of derivatives of likelihood function and the asymptotic error rate is $O(1)$. We do not use Laplace method and jump to the approximate the full conditional distribution by a Gaussian distribution directly, the corresponding likelihood function is

$$\tilde{p}(\boldsymbol{y}|k_1, k_2, \boldsymbol{\theta}) \triangleq \left. \frac{p(\boldsymbol{f}, \boldsymbol{y}|k_1, k_2, \boldsymbol{\theta})}{\tilde{p}_G(\boldsymbol{f}|\boldsymbol{y}, k_1, k_2, \boldsymbol{\theta})} \right|_{\boldsymbol{f}=\tilde{\boldsymbol{f}}} \tag{3.32}$$

where $\tilde{p}_G(\boldsymbol{f}|\boldsymbol{y}, k, \boldsymbol{\theta})$ is the Gaussian approximation to the full conditional density $p(\boldsymbol{f}|\boldsymbol{y}, k, \boldsymbol{\theta})$ and $\tilde{\boldsymbol{f}}$ is the mode of the full conditional density of $\boldsymbol{f}$ for a given $\boldsymbol{\theta}$. Besides, the joint distribution becomes

$$\begin{aligned} p(\boldsymbol{f}, \boldsymbol{y}|k_1, k_2, \boldsymbol{\theta}) &= p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{\theta}) p(\boldsymbol{f}|\boldsymbol{\theta}) \\ &= \exp\Big(\log\big(p(\boldsymbol{f}|\boldsymbol{\theta})\big) + \sum_{i=1}^{n_1} \log\big(p(y_{1,i}|f_{1,i}, k_1) + \sum_{i=1}^{n_2} \log\big(p(y_{2,i}|f_{2,i}, k_2)\big)\Big) \end{aligned} \tag{3.33}$$

The Taylor expression to the second order of $\log\big(p(y_i|f_i, k_1)\big)$, denoted by $g_{1,i}$, and

$\log\big(p(y_i|f_i,k_2)\big)$, denoted by $g_{2,i}$, are

$$g_{1,i}(f_{1,i}) \approx g_{1,i}(f_{1,i}^{(0)}) + \frac{g'_{1,i}(f_{1,i}^{(0)})}{1!}(f_{1,i} - f_{1,i}^{(0)}) + \frac{g''_{1,i}(f_{1,i}^{(0)})}{2!}(f_{1,i} - f_{1,i}^{(0)})^2,\ i = 1, ..., n_1$$
$$g_{2,i}(f_{2,i}) \approx g_{2,i}(f_{2,i}^{(0)}) + \frac{g'_{2,i}(f_{2,i}^{(0)})}{1!}(f_{2,i} - f_{2,i}^{(0)}) + \frac{g''_{2,i}(f_{2,i}^{(0)})}{2!}(f_{2,i} - f_{2,i}^{(0)})^2,\ i = 1, ..., n_2$$

By defining

$$a_{1,i} = \frac{y_{1,i}}{e^{f_{1,i}^{(0)}}} + f_{1,i}^{(0)}\frac{y_{1,i}}{e^{f_{1,i}^{(0)}}} - k_1,\ i = 1, ..., n_1$$
$$a_{2,i} = \frac{y_{2,i}}{e^{f_{2,i}^{(0)}}} + f_{2,i}^{(0)}\frac{y_{2,i}}{e^{f_{2,i}^{(0)}}} - k_2,\ i = 1, ..., n_2$$
$$d_{1,i} = \frac{y_{1,i}}{e^{f_{1,i}^{(0)}}},\ i = 1, ..., n_1$$
$$d_{2,i} = \frac{y_{2,i}}{e^{f_{2,i}^{(0)}}},\ i = 1, ..., n_2$$

$g_{1,i}(f_{1,i})$ and $g_{2,i}(f_{2,i})$ can be approximated as follows

$$g_{1,i}(f_{1,i}) \approx g_{1,i}(f_{1,i}^{(0)}) + a_{1,i}f_{1,i} - \frac{1}{2}d_{1,i}f_{1,i}^2,\ i = 1, ..., n_1$$
$$g_{2,i}(f_{2,i}) \approx g_{2,i}(f_{2,i}^{(0)}) + a_{2,i}f_{2,i} - \frac{1}{2}d_{2,i}f_{2,i}^2,\ i = 1, ..., n_2$$

Therefore, we obtain an approximation of the joint distribution $p(\boldsymbol{f}, \boldsymbol{y}|k_1, k_2, \boldsymbol{\theta})$ which is

$$p(\boldsymbol{f}, \boldsymbol{y}|\boldsymbol{\theta}, k_1, k_2) \propto \exp\{-\frac{1}{2}\boldsymbol{f}^T\Psi^{-1}\boldsymbol{f} - \frac{1}{2}\boldsymbol{f}^T D\boldsymbol{f} + \boldsymbol{a}^T\boldsymbol{f}\}$$

and

$$D = \begin{pmatrix} D_1 & 0 \\ 0 & D_2 \end{pmatrix}$$

where D_1 is a $n_1 \times n_1$ diagonal matrix of $d_{1,1}, ...d_{1,n_1}$ and D_2 is a $n_2 \times n_2$ diagonal matrix of $d_{2,1}, ..., d_{2,n_2}$ and $\boldsymbol{a} = (a_{1,1}, ..., a_{1,n_1}, a_{2,1}, ..., a_{2,n_2})$. After using the Fisher scoring algorithm , we get the Gaussian approximation of full conditional density $p(\boldsymbol{f}|\boldsymbol{y}, k_1, k_2,)$. Then, based on the approximate method, the negative marginal log-likelihood of 2-dimensional

Gaussian process regression model for Gamma-distributed data becomes

$$l(\boldsymbol{\theta}) = \sum_{i=1}^{n_1} \frac{y_{1,i}}{\exp(\tilde{f}_{1,i})} + \sum_{i=1}^{n_2} \frac{y_{2,i}}{\exp(\tilde{f}_{2,i})} + \frac{1}{2}\log|\Psi| + \frac{1}{2}\tilde{\boldsymbol{f}}^T\Psi^{-1}\tilde{\boldsymbol{f}} - \frac{1}{2}\log|(\Psi^{-1}+D)^{-1}|$$
$$+n_1\log\Gamma(k_1) + n_2\log\Gamma(k_2) - (k_1-1)\log\prod_{i=1}^{n_1} y_{1,i} - (k_2-1)\log\prod_{i=1}^{n_2} y_{2,i} \tag{3.34}$$

where $\tilde{\boldsymbol{f}} = [\tilde{\boldsymbol{f}}_1, \tilde{\boldsymbol{f}}_2]$. The estimated hyper-parameters can be obtained by optimising this negative marginal log-likelihood.

We extend the approximate Bayesian computation for high-dimensional case and suppose $\boldsymbol{x}^*$ to be a new input. For simplicity, a 2-dimensional case has been considered as before. This algorithm is also efficiently for prediction, since the numerical results show that the prediction is more accurate than that of the analytical solution.

Algorithm 8: Approximate Bayesian computation for multi-variate prediction.

1. Draw $\boldsymbol{f}^{(j)} = (f_{1,1}^{(j)}, ..., f_{1,n_1}^{(j)}, f_{2,1}^{(j)}, ..., f_{2,n_2}^{(j)})$ from the Gaussian approximation $\tilde{p}_G(\boldsymbol{f}|\boldsymbol{y}, k_1, k_2, \boldsymbol{\theta})$ in Equation (3.23), $j = 0$;
2. Draw $\hat{y_{1,i}}$ form $Gamma(k_1, \exp(f_{1,i}^{(j)}))$ for $i = 1, ..., n_1$ and $\hat{y_{2,i}}$ form $Gamma(k_2, \exp(f_{2,i}^{(j)}))$ for $i = 1, ..., n_2$;
3. Calculate the distance between the simulated training data $\hat{\boldsymbol{y}}$ and the observed data $\boldsymbol{y}$. If $d(\boldsymbol{y}, \hat{\boldsymbol{y}}) < \epsilon$, where ϵ is a strictly positive tolerance, we accept and store $\boldsymbol{f}^{(j)}$; else, we reject $\boldsymbol{f}^{(j)}$. $j = j + 1$ and back to Step 1 until j reaches a large number. The distribution of stored values is written by $p_{ABC}(\boldsymbol{f}|\mathcal{D})$;
4. Draw $\boldsymbol{f}^{(r)}$ from $p_{ABC}(\boldsymbol{f}|\mathcal{D})$, then, $p(f^*|\mathcal{D}) \approx \sum_{r=1}^{R} p(f^*|\boldsymbol{f}^{(r)})$ with appropriately large R ;
5. Repeat Step 4 N times, we get $p(f^{*(1)}|\mathcal{D}), ..., p(f^{*(N)}|\mathcal{D})$'s. Then, $E[y^*|\mathcal{D}] \approx \sum_{h=1}^{N} k_d e^{f^{*(h)}}$ and $Var(y^*|\mathcal{D}) \approx \sum_{h=1}^{N} k_d e^{2f^{*(h)}}$ where $f^{*(h)}$ is drawn from $p(f^{*(h)})$.

As discussed above, the parameter k in Gamma distribution is given. In practice, if there is no prior knowledge about k, we can calculate it by cross-validation.

3.3.6 Simulation study for one-dimensional outputs

In the simulation study of Gaussian process regression model for Gamma-distributed data, we generate some data sets following gamma distribution and the parameter θ following Gaussian process with sine mean function. Because of computational reasons, we choose squared exponential covariance function which only includes two hyper-parameters.

In this section, we test our model on abundant toy data sets generated from

$$y(x) = \frac{1}{N}\sum_{i=1}^{N} v_i(x)$$

where x is covariate, $v_i(x) \sim \text{Gamma}(k, \exp(\sin(x) + f))$, N is the number of data points sampled from Gamma distribution and f follows a zero mean Gaussian process with squared exponential covariance function

$$k(x_i, x_j) = v \exp\big(-\frac{1}{2w}(x_i - x_j)^2\big) + \delta_{ij}\sigma^2.$$

Gaussian process regression is an effective model to capture the non-linear relationship between high-dimensional independent variables and one-dimensional dependent variables. However, for simplicity, in this simulation study, the independent variable is assumed to be 1-dimensional. Without loss of generality, we choose an appropriate N that is 1000. Suppose 30 data points are equally spaced in interval $(0, 5)$ in which 25 data points are randomly selected as the training data and the remaining 5 data points are used as test data. The first step is to implement the method in Section 3.3.3 which approximates the likelihood function and estimates the hyper-parameters via empirical Bayesian learning. To compare the performances of two prediction methods, one is the analytical form in Equation (3.27) and the other is based on approximate Bayesian computation (algorithm 7), we repeat this numerical experiment for 100 times. In other words, our model is trained on thousands of training data points and tested on hundreds of test data points, which leads to the numerical results are non-trivial. The root-mean-square-error and correlation coefficient between true data and predictions are computed. In addition, we also consider different parameter k ($k = 0.5$, $k = 1$ and $k = 2$) and different data sizes (30 data points, 50 data points and 70 data points) when we generate the data. To sum up, the data is

generated by

$$y_i = \frac{1}{1000}\sum_{l=1}^{1000} v_l(x_i),\ i = 1, ..., n$$
$$v_l(x_i) \sim Gamma(k, \exp(\sin(x_i)) + f) \tag{3.35}$$

where x_i is equally spaced in $(0, 5)$, $k \in [0.5, 1, 2]$, $n \in [30, 50, 70]$, f is sampled from a multivariate normal distribution with zero mean and the i, j-th entry of its covariance matrix is $k(x_i, x_j)$.

At the beginning of simulation study, we consider 20 different candidate ks which are equally spaced in interval $(0.5, 3)$. Then, cross-validation is applied to determine the suitable k. In details, for each candidate k, we calculate corresponding root-mean-square-error and correlation coefficient r between test data points and predictions from analytical form. Among these 20 values of k, the one with minimal $rmse$ is selected as the optimal estimate, denoted as $\hat{k}$. In order to make predictions from ABC algorithm based on $\hat{k}$. We calculate corresponding a, D, $\tilde{\boldsymbol{f}}$ and $\boldsymbol{\theta}$. Moreover, ABC is able to improve the prediction of our model, which is confirmed by the numerical results in Table 3.1. Due to the expensive computation of differential evolution algorithm, we set the initial population size is 50 and the number of evolution is 150.

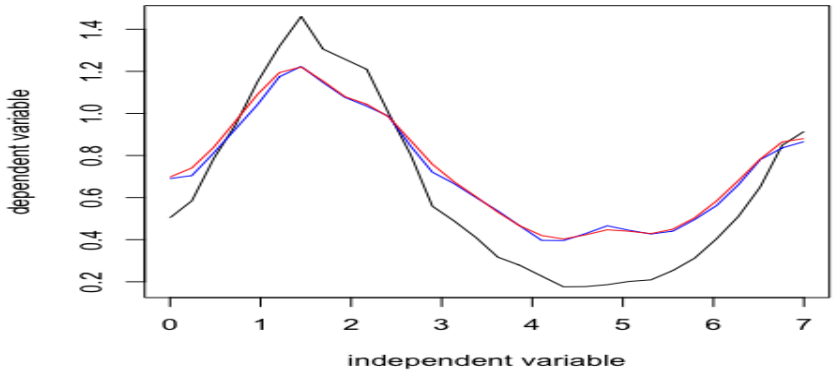
	$k = 0.5$	$k = 1$	$k = 2$
rmse			
25 train data	**0.1921***	**0.2003***	**0.3690***
	0.2087†	0.2919†	0.4092†
45 train data	**0.0881***	**0.1349***	**0.1823***
	0.1359†	0.1520†	0.1938†
65 train data	**0.0797***	**0.1123***	0.1794*
	0.0994†	0.1128†	**0.1408**†
r			
25 train data	0.9377*	0.9874*	0.9834*
	0.9761†	0.9864†	0.9835†
45 train data	0.9904*	0.9942*	0.9958*
	0.9889†	0.9948†	0.9962†
65 train data	0.9879*	0.9954*	0.9957*
	0.9829†	0.9960†	0.9970†

Table 3.1: Root-mean-squared-error and correlation coefficient of Gaussian process regression for Gamma-distributed data with different ks and data sizes. Each numerical experience has 5 test data points. The bold writing means minimal rmse.

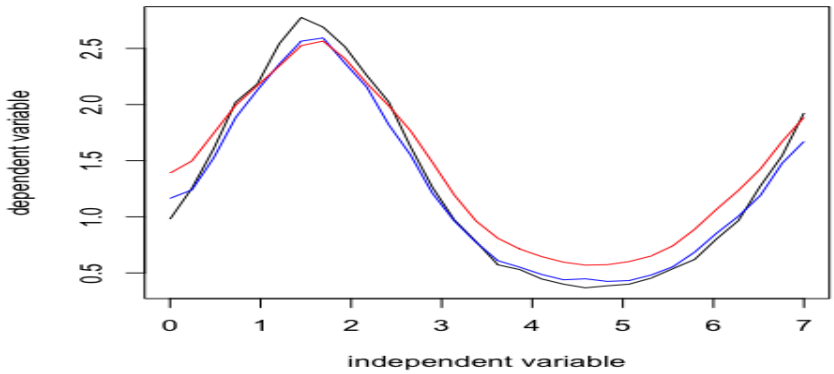
The numerical results show that our model is effective for interpolation and its performance improves with the increasing number of training data. It is clear that the predictions of ABC algorithm (numbers labelled by $*$) are more accurate than that of analytical form (numbers labelled by $\dagger$) for most training data sets. In addition, for fixed k, such as $k = 0.5$, $k = 1$ or $k = 2$, the $rmse$ decreases with the increase of training data size. For example, the $rmse$ of approximate Bayesian computation for the 65 training data with $k = 0.5$ is 0.0797 which is less than 0.1921 for 25 training data and 0.0881 for the 45 training data. However, for a fixed sample size, the accuracy of prediction performance become worse with growing k. For instance, if we fix that $k = 0.5$, the $rmse$ of ABC is 0.1921 for the 25 training data, 0.0881 for the 45 training data and 0.0797 for the 65 training data. It is not surprising that the decline of predictive error with rise of the training data size, since our model is able to learn more information from more data. We can see

that sometimes, the predictions with lower rmse may has low correction coefficient, such as $k = 0.5$ and training data points are 25. The reason is that we formed an approximated posterior distribution for ABC, and this distribution is generated based on the distance between simulated *data* and real data. In other words, the prediction from ABC only cares about the accuracy of prediction rather than the correction, while the prediction from analytical solution is derived from its assumption which also focus on the relation between real data and prediction.

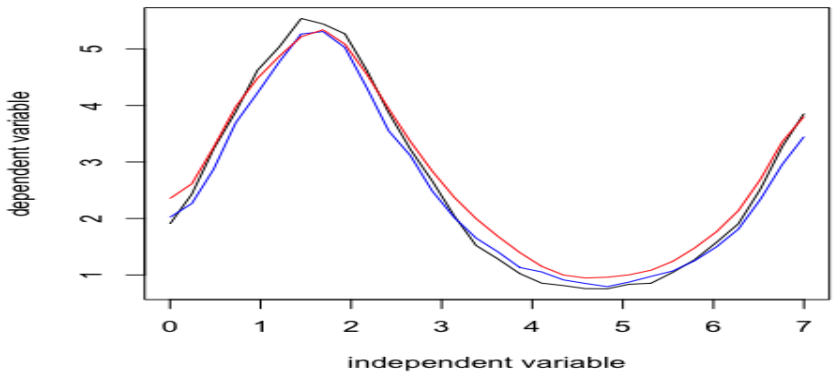
Nonetheless, the interesting phenomenon that the reduction of predictive ability with enlargement of k needs further exploration. The data visualisation for approximate Bayesian computation and analytical solution with different ks and the training data sizes are presented in Figure 3.5, 3.6 and 3.7.



(a) Data visualisation for 1-dimensional Gamma-distributed data with $k = 0.5$.

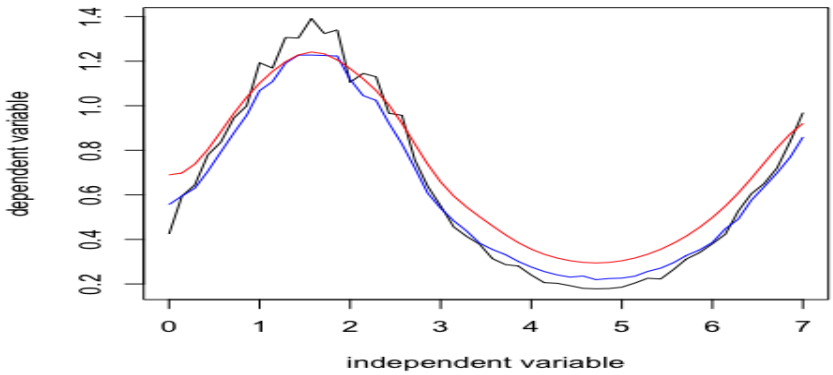


(b) Data visualisation for 1-dimensional Gamma-distributed data with $k = 1$.

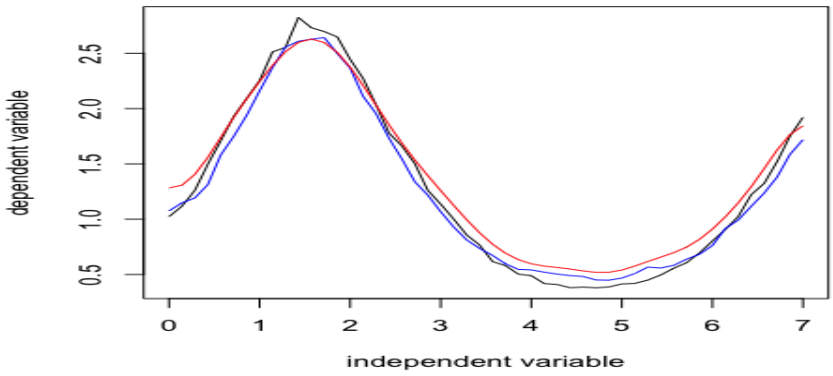


(c) Data visualisation for 1-dimensional Gamma-distributed data with $k = 2$.

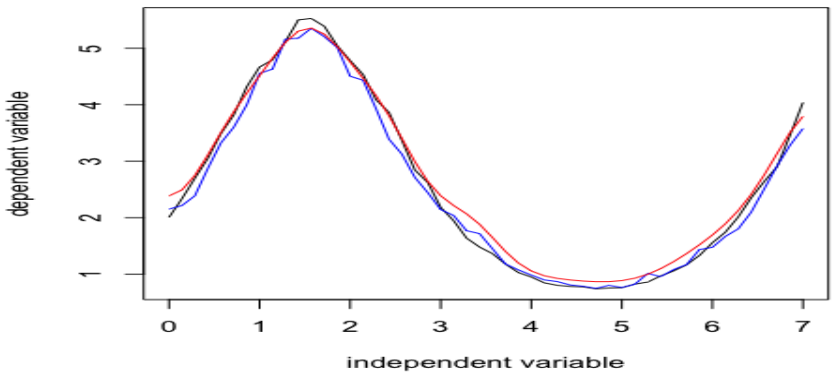
Figure 3.5: Data visualization for 1-dimensional Gamma-distributed data with the 25 training data where blue curves refer to predictions from approximate Bayesian computation, red curves refer to predictions from analytical solution and black curves refer to real data.



(a) Data visualisation for 1-dimensional Gamma-distributed data with $k = 0.5$.

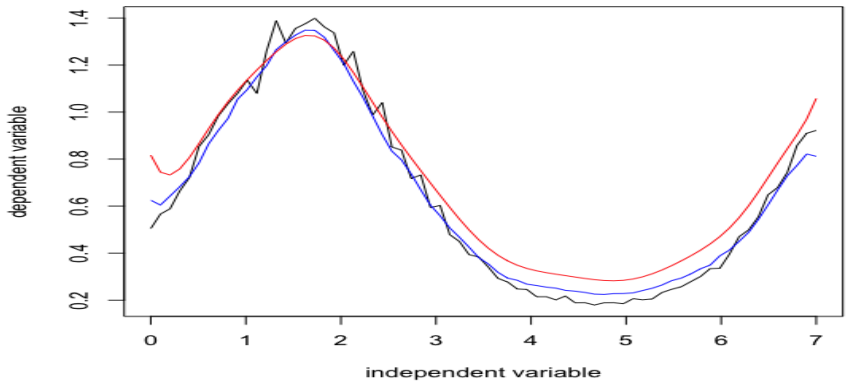


(b) Data visualisation for 1-dimensional Gamma-distributed data with $k = 1$.

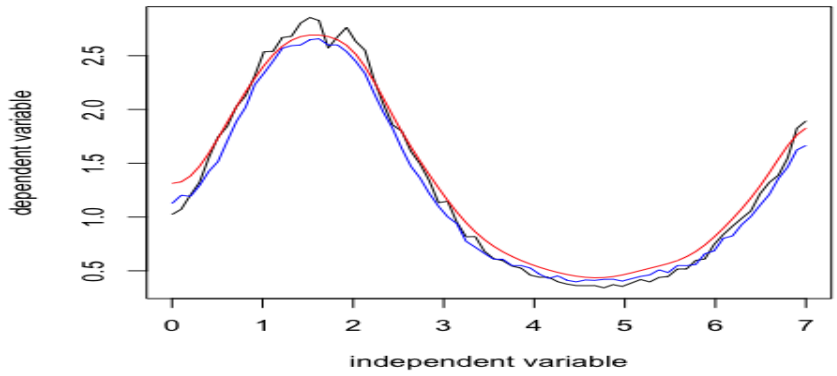


(c) Data visualisation for 1-dimensional Gamma-distributed data with $k = 2$.

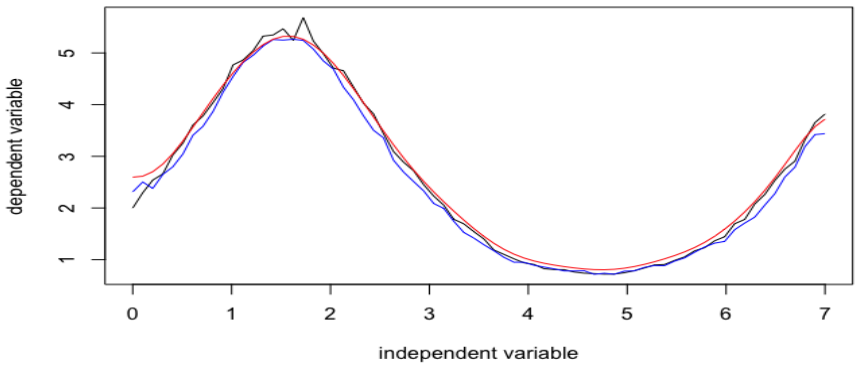
Figure 3.6: Data visualization for 1-dimensional Gamma-distributed data with the 45 training data where blue curves refer to predictions from approximate Bayesian computation, red curves refer to predictions from analytical solution and black curves refer to real data.



(a) Data visualisation for 1-dimensional Gamma-distributed data with $k = 0.5$.



(b) Data visualisation for 1-dimensional Gamma-distributed data with $k = 1$.



(c) Data visualisation for 1-dimensional Gamma-distributed data with $k = 2$.

Figure 3.7: Data visualization for 1-dimensional Gamma-distributed data with the 65 training data where blue curves refer to predictions from approximate Bayesian computation, red curves refer to predictions from analytical solution and black curves refer to real data.

We also show the numerical results for the method which maps the data via log-transformation and uses GPR directly In Table 3.2. Compared to Table 3.1, the predictive errors from GPRG are less for many cases, especially when the training data size is large. For example, when training data size is 25, the predictions of GPR are more accurate than that of GPRG, but when training data size is 45 or 65, the predictions of GPRG are more accurate. We should notice that, even GPR could provide accurate predictions in many cases but the distribution of data do not satisfy the assumption of GPR. Thus, it might be not reasonable to use GPR directly.

	$k = 0.5$	$k = 1$	$k = 2$
rmse			
25 train data	0.1547	0.1696	0.2304
45 train data	0.1381	0.1670	0.1996
65 train data	0.1106	0.1563	0.1781

Table 3.2: Root-mean-squared-error and correlation coefficient of Gaussian process regression for Gamma-distributed data with different ks and data sizes. Each numerical experience has 5 test data points. The bold writing means minimal rmse.

3.3.7 Simulation study for two-dimensional outputs

Moreover, in practice, the outputs are often high-dimensional. Thus, we consider two-dimensional dependent variables in this simulation study. Suppose there are $2N$ data points by taking N from dimension 1 and N from dimension 2 respectively. The independent variable is uniformly spaced in interval $(0, 5)$. All samples are generated by the formula as follows

$$y_i(x) = \frac{1}{N}\sum_{j=1}^{N} v_{ij}(x), \text{ for } i = 1, 2$$

where $v_{1j}(x) \sim \text{Gamma}(k_1, \exp(\sin(x) + f_1))$, $v_{2j}(x) \sim \text{Gamma}(k_2, \exp(\sin(x) + \cos(x) + f_2))$, f_1 and f_2 follow a convolved Gaussian process with covariance function defined in Equation (2.24). We do not test the performance of high-dimensional model repeatedly since it is very computational expensively to extimate the hyper-parameters by a differential evolution method in CGP. Analogously to the simulation study of 1-dimensional case, we

train this model on many date sets with various scenarios. We also use cross-validation to select k_1 and k_2 which are equally spaced in interval $[0, 5]$ with 10 steps. And the number of training data points are 25 and 45 for each scenario. Due to the expensive computation of differential evolution algorithm, we set the initial population size is 50 and the number of evolution is 150. The numerical results are shown in Table 3.3 where we assume $k_1 = k_2$ and in Table 3.4 where we assume $k_1 \neq k_2$.

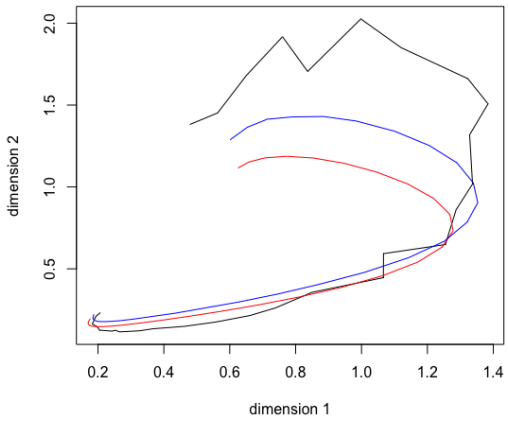
	$k_1 = k_2 = 0.5$	$k_1 = k_2 = 1.5$
rmse		
25 train data	$\mathbf{0.0704}^*$	$\mathbf{0.2556}^*$
	$0.1477^\dagger$	$0.6301^\dagger$
45 train data	$\mathbf{0.0629}^*$	$\mathbf{0.1589}^*$
	$0.1116^\dagger$	$0.4149^\dagger$
r		
25 train data	0.9918^*	0.9975^*
	$0.9815^\dagger$	$0.9983^\dagger$
45 train data	0.9782^*	0.9989^*
	$0.9599^\dagger$	$0.9987^\dagger$

Table 3.3: Root-mean-squared-error and correlation coefficient of Gaussian process regression for Gamma-distributed data with same k_1, k_2 and data size. Each numerical experience has 5 test data.

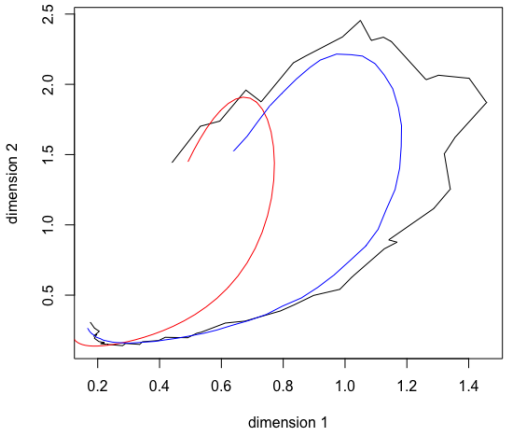
	$k_1 = 1, k_2 = 2$	$k_1 = 2, k_2 = 1$
rmse		
25 train data	**0.0989***	**0.1019***
	0.1106†	0.4050†
45 train data	**0.0642***	**0.0868***
	0.3674†	0.2231†
r		
25 train data	0.9937*	0.9983*
	0.9781†	0.9967†
45 train data	0.9997*	0.9990*
	0.9996†	0.9992†

Table 3.4: Root-mean-squared-error and correlation coefficient of Gaussian process regression for Gamma-distributed data with different k_1, k_2 and data size. Each numerical experience has 5 test data.

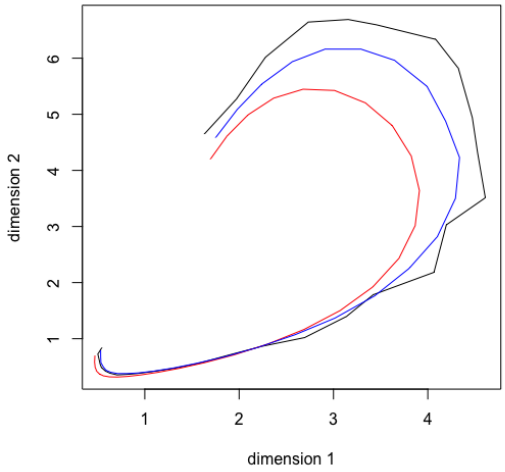
From tables above, we can see that our model leads to more precise predictions with more training data points in both scenarios ($k_1 = k_2$ and $k_1 \neq k_2$). For example, on the one hand, when the training data is increased by 20 data points, the *rmse* of approximate Bayesian computation algorithm is decreased from 0.0704 to 0.0629 with $k_1 = k_2 = 0.5$ and from 0.2556 to 0.1589 with $k_1 = k_2 = 1.5$. On the other hand, the error of prediction has been declined form 0.0989 to 0.0642 with $k_1 = 1, k_2 = 2$ and from 0.1019 to 0.0868 with $k_1 = 2, k_2 = 1$. The phenomenon has occurred here as well that the predictions of our model become inaccurate when the parameters k is increasing. Numerical results for these two scenarios are visualised in Figure 3.8 and 3.9.



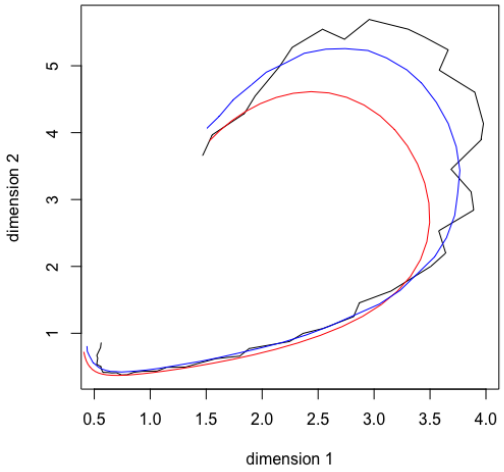
(a) Data visualisation for 2-dimensional gamma-distributed data with $k_1 = 0.5$, $k_2 = 0.5$ and 25 training data.



(b) Data visualisation for 2-dimensional gamma-distributed data with $k_1 = 0.5$, $k_2 = 0.5$ and 45 training data.

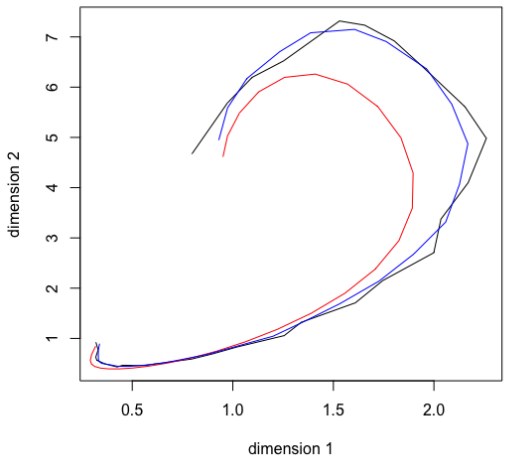


(c) Data visualisation for 2-dimensional gamma-distributed data with $k_1 = 1.5$, $k_2 = 1.5$ and 25 training data.

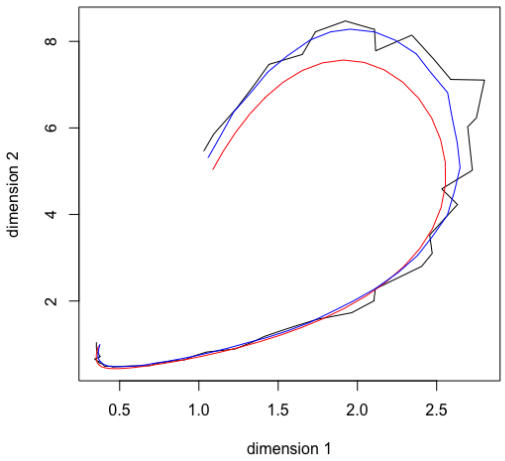


(d) Data visualisation for 2-dimensional gamma-distributed data with $k_1 = 1.5$, $k_2 = 1.5$ and 45 training data.

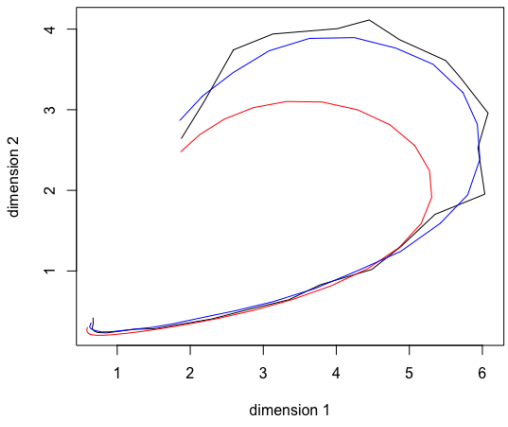
Figure 3.8: Data visualisation for 2-dimensional gamma-distributed data for identical parameters k_1 and k_2 in which blue curves refer to predictions from approximate Bayesian computation, red curves refer to predictions from analytical solution and black curves refer to real data.



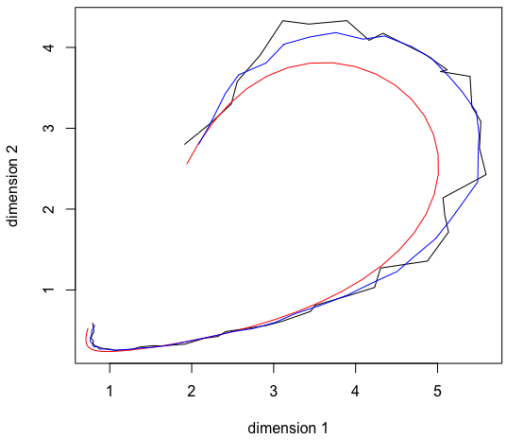
(a) Data visualisation for 2-dimensional gamma-distributed data with $k_1 = 1$, $k_2 = 2$ and 25 training data.



(b) Data visualisation for 2-dimensional gamma-distributed data with $k_1 = 1$, $k_2 = 2$ and 45 training data.



(c) Data visualisation for 2-dimensional gamma-distributed data with $k_1 = 2$, $k_2 = 1$ and 25 training data.



(d) Data visualisation for 2-dimensional gamma-distributed data with $k_1 = 2$, $k_2 = 1$ and 45 training data.

Figure 3.9: Data visualisation for 2-dimensional gamma-distributed data for different parameters k_1 and k_2 in which blue curves refer to predictions from approximate Bayesian computation, red curves refer to predictions from analytical solution and black curves refer to real data.

Chapter 4

Wrapped Gaussian Process Functional Regression for Batch Data on Riemannian Manifolds

With development of technology, people have observed and recorded a variety of data with complex structure, such as manifold-valued data, which provides ground for novel statistical analysis. It is an attractive topic for scholars to consider models and inferences for data on Riemannian manifolds. For example, Kim et al. (2014) propose a multivariate linear regression on Riemannian manifolds for diffusion weighted imaging data and Banerjee et al. (2015) introduce a kernel regression on Riemannian manifolds for medical data analysis. Flight trajectory data can also be considered on a Riemannian manifold, i.e. S^2. In this chapter, we try to model such data via GPR. Although Gaussian process regression is a powerful non-parametric tool in statistics and machine learning, it is not suitable for manifold-valued data directly. Therefore, we consider a variation of GPR in the context of manifold-valued data. In Section 4.2.1, we introduce the main contribution of this thesis: a wrapped Gaussian process functional regression (WGPFR) model on Riemannian manifolds. Moreover, motivated by a special data set where the manifold-valued curves in a batch share a common mean structure, we introduce a variation of WGPFR which has common mean structure in Section 4.2.2. Then, in Section 4.3, an approximate model is proposed which is computational efficiently and provides accurate

predictions for both WGPFR with individual mean structure and common mean structure. Additionally, in Chapter 5, we compare the numerical experiments which involves different scenarios of prediction of different models, such as the model in Mallasto and Feragen (2018).

4.1 A motivating data set

We begin this model by describing a data set which consists of repeated measurement for different subjects. The earth is roughly a sphere and can be modelled as a copy of S^2. Certain data sets, for example hurricane trajectories, can be therefore considered as a manifold-valued random curve (Su et al., 2014). An actual application of our model concerns data collected from flight trajectories (the data are downloaded from https://data.variflight.com), which are shown in Figure 4.1, in which the red curves represent flights from Shanghai to London on British Airlines and the black curves represent flight trajectories of Eastern China Airlines between the same destinations. Therefore, these sets of trajectories can naturally be split into two batches and the model with common mean structure can be used to the flight trajectory data.

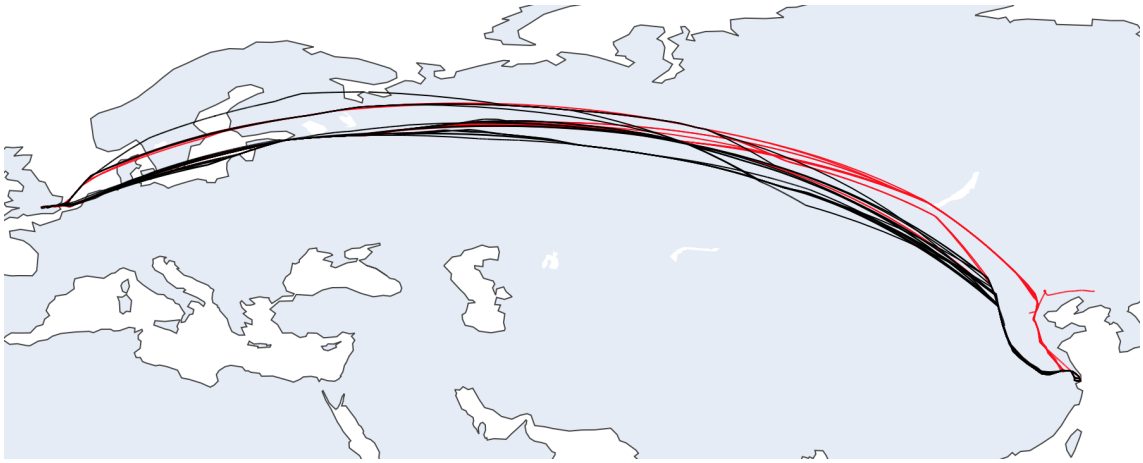


Figure 4.1: Some of original flight route data. The black trajectories are from British Airways and the red curves are from Eastern China Airlines.

The original data includes time, height, speed, longitude and latitude of each flight. We select the position of each airplane as the response variable, which can be transformed onto S^2 using longitude and latitude; in addition, the company and time are regarded as

non-functional and functional covariates. For future research, the height and speed can be added as functional covariates. The response variables $y(t)$ are trajectories of airplanes which are recorded as longitude and latitude and so is a continuous functional variable.

4.2 Wrapped Gaussian process functional regression

4.2.1 A model with individual mean structure

Before we introducing the model for batch data, such as the flight trajectories, we firstly consider a simple model that there is only one curve in a batch. Suppose the data comprise M continuously observed curves on $\mathcal{M}$, so that the m-th curve is denoted as $y_m(t), t \in \mathcal{T},\ m = 1, \ldots, M$, which is a set of $\mathcal{M}$-valued random functions. Associated with the m-th curve, we observe real-valued functional covariates $\boldsymbol{x}_m(t) \in \mathbb{R}^Q$ and scalar covariates $\boldsymbol{u}_m \in \mathbb{R}^p$ which are defined in a conventional way in a functional space of real-valued functions and in an Euclidean space, respectively. In order to identify the non-linear probabilistic relationship between response variables and the corresponding covariates by regression model on Riemannian manifold, we propose the model for batch data on Riemannian manifolds as follows

$$y_m(t) = \mathrm{Exp}(\mu_m(t), \tau_m(t)),\ y_m(t) \in \mathcal{M}, \text{for } m = 1, ..., M \tag{4.1}$$

where $\mu_m(t) \in \mathcal{M}$ refers to the mean structure and $\tau_m(t) \in T_{\mu_m(t)}\mathcal{M}$ refers to the covariance structure. In addition, for a stochastic process, a mean structure can be considered as its trend and a covariance stricture can be considered as a correlation between different points on it.

Due to the curse of dimensionality, it is difficult to infer non-linear relationship non-parametrically between a functional response variable and multi-dimensional functional covariates even though both of them are observed in a space for real-valued functions (Shi et al., 2007). Therefore, we assume that the mean structure $\mu_m(\cdot)$ depends on the scalar covariates $\boldsymbol{u}_m$ only, and the nonlinear concurrent relationship between $y_m(t) \in \mathcal{M}$ and $\boldsymbol{x}_m(t)$ is modelled via a vector-valued function $\tau_m(t)$ in tangent space $T_{\mu_m(t)}\mathcal{M}$.

The data set is denoted as $\mathcal{D} = \{y_{mi}, \boldsymbol{u}_m, \boldsymbol{x}_{mi}, t_{mi} |\ y_{mi} \in \mathcal{M}, \boldsymbol{x}_{mi} \in \mathbb{R}^Q, t_{mi} \in \mathbb{R}, i =$

$1, ..., N_m,\ \boldsymbol{u}_m \in \mathbb{R}^p, m = 1, \ldots, M\}$, in which $y_{mi} = y_m(t_{mi})$ and $\boldsymbol{x}_{mi} = \boldsymbol{x}_m(t_{mi})$ are the observations of functional response variables and functional predictor variables for the m-th curve at time point t_{mi} respectively, $\boldsymbol{u}_m$ is the batch-specific scalar covariates.

As a common assumption, the existence and uniqueness of Fréchet mean $\mu_0(t)$ for $y_m(t), m = 1, ..., M$, are pre-required for an intrinsic analysis (Dai et al., 2018; Bhattacharya et al., 2003; Petersen and Müller, 2019). In addition, the injectivity radius is supposed to be large enough so that the exponential map and inverse exponential map exist almost everywhere on $\mathcal{M}$.

Equipped with any manifold-valued curve $\mu_*(t)$, the mean structure for the m-th curve, $\mu_m(t)$, is defined via exponential map at $\mu_*(t)$ of a tangent vector $\boldsymbol{u}_m^T\boldsymbol{\beta}(t)$

$$\mu_m(t) = \mathrm{Exp}(\mu_*(t), \boldsymbol{u}_m^T\boldsymbol{\beta}(t)) \tag{4.2}$$

where $\mu_*(t)$ is a pre-selected curve on Riemannian manifold playing a similar role of the line $y = 0$ in a 2-dimensional Euclidean coordinate. In estimation part, we can just use the intrinsic Fréchet population mean $\mu_0(t)$, which is defined in the next section, as a choice of $\mu_*(t)$ and it is more intuitive since $\mu_0(t)$ is the "mean curve" of all $y_m(t)$. In practice, we use sample Fréchet mean $\hat{\mu}_0(t)$ instead of population Fréchet mean $\mu_0(t)$, and the distance between $\hat{\mu}_0(t)$ and $\mu_0(t)$ tends to 0 when sample size tends to infinity under some assumptions[1]. Moreover, the definition of other parameters are given as follows, $\boldsymbol{u}_m \in \mathbb{R}^p$ is a p-dimensional vector of batch-specific covariates, $\boldsymbol{\beta}(t) = (\boldsymbol{\beta}_1(t), ..., \boldsymbol{\beta}_p(t))$ where $\boldsymbol{\beta}_j(t) \in T_{\mu_0(t)}\mathcal{M},\ j = 1, \ldots, p$, is a d-dimensional vector of functional coefficients. For example, if we have one-dimensional indicator covariate, i.e. $u_m = 1$ stands for batch A and $u_m = 2$ for batch B, then the second part of the right hand side in Equation (4.2) is simplified to ${\boldsymbol{u}_m}^T\boldsymbol{\beta}(t) = \boldsymbol{\beta}_1(t) + \boldsymbol{\beta}_2(t)u_m$. Therefore, when t is fixed, each $\boldsymbol{\beta}_j(t)$ is a tangent vector in $T_{\mu_0(t)}\mathcal{M}$. Since we assume $\mathcal{M}$ is a closed Riemannian submanifold of a Euclidean space, which means the Riemannian manifold can be embedded into a Euclidean space, it is reasonable to suppose that $\boldsymbol{V}(t) = (\boldsymbol{v}_1(t), \cdots, \boldsymbol{v}_d(t))$ is a basis of the tangent space $T_{\mu_0(t)}\mathcal{M}$ where each $\boldsymbol{v}_l(t),\ l = 1, ..., d$, denotes a d-dimensional vector playing a similar role to a basis vector in coordinate systems. For example, when $d = 2$,

[1] See Assumption A1,B1,B2,B3,B4 and proposition 2 in Dai et al. (2018)

$\boldsymbol{v}_1(t)$ and $\boldsymbol{v}_2(t)$ can be considered as basis vectors in a 2-dimensional vector space which are $(1,0)$ and $(0,1)$ respectively. However, without loss of generality, it is not necessary to assume $(\boldsymbol{v}_1(t),...,\boldsymbol{v}_d(t))$ is orthonormal pairwise. We can calculate the Karhunen-Loéve expansion of any component $\nu(t)$ in $\boldsymbol{v}_l(t)$, that is

$$\begin{aligned}\nu_q(t) &= \sum_{k=1}^{\infty} w_{qk}\phi_k(t) \\ &\approx \sum_{k=1}^{K} w_{qk}\phi_k(t)\end{aligned} \tag{4.3}$$

where q refers to the q-th dimension of $\boldsymbol{v}_l(t)$, $\phi_k(t)$ refers to the k-th element in a basis function and, such as B-spline w_{qk} refers to a weight of $\phi_k(t)$.

Hence, the tangent vector-valued function $\boldsymbol{\beta}_j(t)$ can be approximated by a set of convential Euclidean basis functions such as wavelet or B-splines,

$$\begin{aligned}\boldsymbol{\beta}_j(t) &= \sum_{l=1}^{d} a_{jl}\boldsymbol{v}_l(t) \\ &= \sum_{l=1}^{d} a_{jl}(\sum_{k=1}^{K} b_{1kl}\phi_k(t),...,\sum_{k=1}^{K} b_{dkl}\phi_k(t)) \\ &= \sum_{l=1}^{d} a_{jl}\sum_{k=1}^{K}\boldsymbol{b}_{kl}\phi_k(t) \\ &= \sum_{k=1}^{K}\boldsymbol{c}_{jk}\phi_k(t).\end{aligned} \tag{4.4}$$

where $\boldsymbol{c}_{jk} = (c_{j1k},...,c_{jdk})$ and in the rest of the conv, we use $\boldsymbol{c}_{jk}$ to represent a vector of weights. But, we should notice that, in the inference part, each weight is estimated independently.

Since $\boldsymbol{\beta}_j(t)$ is modelled along the tangent spaces of $\mu_0(t)$ which is a Euclidean subspace,

we can parameterize the mean structure as follows

$$\begin{aligned}\mu_m(t) &= \text{Exp}(\mu_0(t), \boldsymbol{u}_m^T \boldsymbol{\beta}(t)) \\ &= \text{Exp}(\mu_0(t), \sum_{j=1}^{p} u_{mj} \boldsymbol{\beta}_j(t)) \\ &= \text{Exp}(\mu_0(t), \sum_{j=1}^{p} u_{mj} \sum_{k=1}^{K} \boldsymbol{c}_{jk} \phi_k(t))\end{aligned} \tag{4.5}$$

In addition, the covariance structure $\tau_m(\cdot)$ is defined as a Gaussian process which models the tangent vectors from a mean structure to corresponding manifold-valued response variables and it is also the covariance between manifold-valued data.

$$\tau_m(\cdot) = Cov(y_m(\cdot), y_m(\cdot)) \sim GP(0, k(\cdot, \cdot; \boldsymbol{\theta}_{ml})) \tag{4.6}$$

where $k(\cdot, \cdot)$ refers to a kernel, such as squared exponential kernel and $\boldsymbol{\theta}$ refers to the hyper-parameters of the kernel. In other words, the covariance structure can be roughly considered a zero mean Gaussian process.

The correlation of different elements in $\tau_m(t) = (\tau_{m,1}(t), ..., \tau_{m,d}(t))$ could be computed via a cross-covariance function model, such as the convolved Gaussian process. Nevertheless, the size of a covariance matrix in a GP is $n \times n$ while the size of a cross-covariance matrix in CGP is $n^d \times n^d$, which is computationally expensive. In the following of this chapter, we still consider different elements independently.

Furthermore, the covariance structure models the tangent vectors from mean structure to corresponding manifold-valued curve which is calculated via inverse exponential map

$$\tau_m(t) = \text{Log}(\mu_m(t), y_m(t)), \text{ for } m = 1, ..., M. \tag{4.7}$$

Covariance structure can be represented by basis eigen-functions in Dai et al. (2018) or wrapped Gaussian process in Mallasto and Feragen (2018). We adopt the latter since it is capable to include the high-dimensional covariates $\boldsymbol{x}_m(t)$ and provides a probabilistic framework for prediction.

In this chapter,a stochastic process on a Riemannian manifold is considered as the "sum" of two parts: a mean structure determines the trend and a covariance structure

determines the randomness. For example, the flight trajectories (which are data on a S^2) from Shanghai to London are similar. But, there are some differences between them which are caused by weather, air traffic control and so on. In this case, the main trend of these flight routes are is modelled by a mean structure and the differences between them are model by a covariance structure.

In order to explain the mechanism of the model with individual mean structure schematically, we use Figure 4.2 below. Specifically, a sample Fréchet mean μ_* is shown as a blue curve. After we add the covariate information $\boldsymbol{u}_m^T\boldsymbol{\beta}(t)$ to μ_*, we get a mean structure μ_m which is the green curve. A manifold-valued stochastic process can be generated if we add random noise $\tau_m(t)$ to the mean structure μ_m, which is the black curve. Moreover, the black points are observed data.

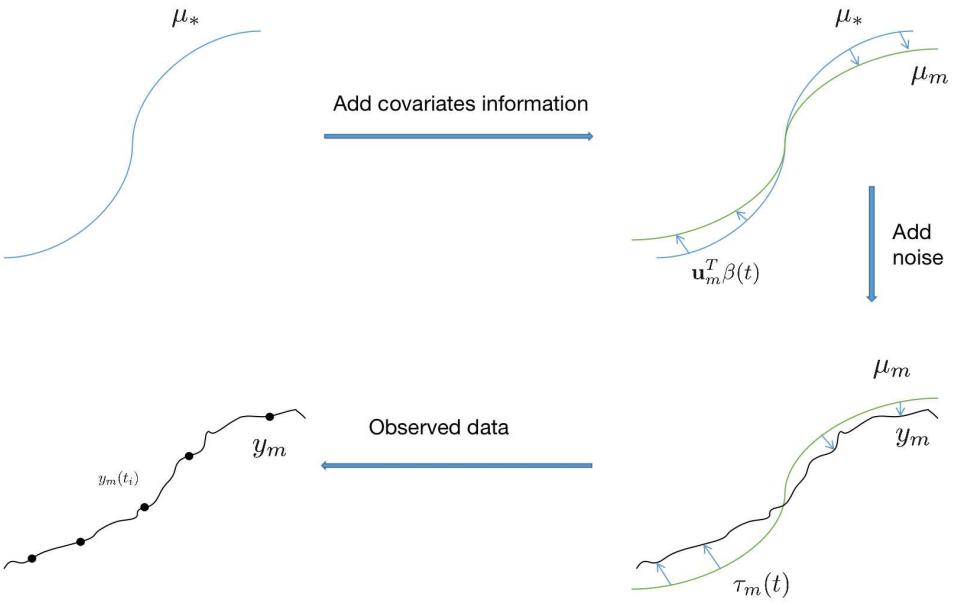


Figure 4.2: Schematic explanation of WGPFR with individual mean structure.

Estimation of mean structure

In this section, our aim is to estimate the functional tangent vector $\boldsymbol{\beta}_j(t), j = 1, ..., p$, which can be computed by a sum of weighted Euclidean basis functions in $T_{\mu_0(t)}\mathcal{M}$.

Therefore, it is necessary to define the distance function between manifold-valued points. Since the inverse exponential map exists almost everywhere under the assump-

tion that $\mathcal{M}$ is geodesically complete and then the norm of inverse exponential map is reasonable to be considered as a distance function between points on such Riemannian manifolds, that is

$$d_{\mathcal{M}}(\cdot,\cdot) = \|\mathrm{Log}(\cdot,\cdot)\|$$

see Table 1 in Kim et al. (2014) for more details about distance and other basic operations on Riemannian manifolds.

For the m-th curve, The data are observed at time point t_{mi} is denoted as $\mathcal{D} = \{y_{mi}, \boldsymbol{u}_m, \boldsymbol{x}_{mi}, t_{mi} |\ y_{mi} \in \mathcal{M}, \boldsymbol{x}_{mi} \in \mathbb{R}^Q, t_{mi} \in \mathbb{R}, i = 1, ..., N_m,\ \boldsymbol{u}_m \in \mathbb{R}^p, m = 1, \ldots, M\}$, in which $y_{mi} = y_m(t_{mi})$ and $\boldsymbol{x}_{mi} = \boldsymbol{x}_m(t_{mi})$ are the observations of functional response variables and functional predictor variables for the m-th curve at time point t_{mi} respectively, $\boldsymbol{u}_m$ is the batch-specific scalar covariates. For any time point t, we firstly define the intrinsic population mean function $\mu_0(t)$ for all $y_m(t), m = 1, ..., M$, under the assumption of existence and uniqueness:

$$\mu_0(t) = \arg\min_{p \in \mathcal{M}} \sum_{m=1}^{M} E[d_{\mathcal{M}}(y_m(t), p)^2]. \tag{4.8}$$

When t is fixed, $\mu_0(t)$ refers to a point on a Riemannian manifold; when t is variable, $\mu_0(t)$ refers to a curve on a Riemannian manifold. Additionally, since each $y_m(t) \in \mathcal{M}$ is continuous, $\mu_0(t)$ is also continuous.[2]

The population quantities of intrinsic Fréchet mean function are estimated by the observations of all $y_m(t)$. Therefore, in practice, the value of intrinsic Fréchet means are approximated by the sample Fréchet means:

$$\hat{\mu}_0(t) \approx \arg\min_{p \in \mathcal{M}} \frac{1}{M} \sum_{m=1}^{M} d_{\mathcal{M}}(y_m(t), p)^2 \tag{4.9}$$

as discussed in Section 2.4.5, $\hat{\mu}_0(t)$ can be estimated by Algorithm 1.

With ideal tangent vector-valued function $\boldsymbol{\beta}(t)$, the mean structure should be as accurate as possible and the accuracy is measured by the distance between $\mu_m(t)$ and $y_m(t)$. In other words, given the observed data set $\mathcal{D}$, the estimation of unknown functional tangent

[2] see equation (1) in Dai et al. (2018)

vectors $\boldsymbol{\beta}(t)$ is converted to estimation of coefficients c_{jlk} by minimising the loss function below

$$\frac{1}{2}\sum_{m=1}^{M}\sum_{i=1}^{N_m} d_{\mathcal{M}}(\mu_m(t_{mi}), y_m(t_{mi}))^2 \tag{4.10}$$

where $\mu_m(t)$ is defined in Equation (4.5).

Therefore, the loss function can be expanded as

$$\begin{aligned} E(C) &= \frac{1}{2}\sum_{m=1}^{M}\sum_{i=1}^{N_m} \|\mathrm{Log}(\mu_m(t_{mi}), y_m(t_{mi}))\|^2 \\ &= \frac{1}{2}\sum_{m=1}^{M}\sum_{i=1}^{N_m} \|\mathrm{Log}(\mathrm{Exp}(\hat{\mu}_0(t_{mi}), \boldsymbol{u}_m^T\boldsymbol{\beta}(t_{mi})), y_m(t_{mi}))\|^2 \\ &= \frac{1}{2}\sum_{m=1}^{M}\sum_{i=1}^{N_m} \|\mathrm{Log}(\mathrm{Exp}(\hat{\mu}_0(t_{mi}), \sum_{j=1}^{p} u_{mj}\sum_{k=1}^{K}\boldsymbol{c}_{jk}\phi_k(t_{mi})), y_m(t_{mi}))\|^2 \end{aligned} \tag{4.11}$$

where $C = (\boldsymbol{c}_{11}, ..., \boldsymbol{c}_{1K}, ..., \boldsymbol{c}_{p1}, ..., \boldsymbol{c}_{pK})$.

As most optimization problems, we firstly attempt to use a gradient descent algorithm to estimate the parameters in a loss function. The coefficients c_{jlk} can be estimated by descending the gradient of (4.11) with respect to $c_{jlk}, j = 1, ..., p, l = 1, ..., d, k = 1, ..., K$, which is given by

$$\begin{aligned} \nabla_{c_{jlk}} E &= \nabla_{\boldsymbol{u}_m\boldsymbol{\beta}(t_{mi})} E \frac{\partial \boldsymbol{u}_m^T\boldsymbol{\beta}(t_{mi})}{\partial c_{jlk}} \\ &= -\sum_{m=1}^{M}\sum_{i=1}^{N_m} d_{\boldsymbol{u}_m^T\boldsymbol{\beta}(t_{mi})}\mathrm{Exp}(\hat{\mu}_0(t_{mi}), \boldsymbol{u}_m^T\boldsymbol{\beta}(t_{mi}))^\dagger \\ &\quad \mathrm{Log}(\mathrm{Exp}(\hat{\mu}_0(t_{mi}), \boldsymbol{u}_m^T\boldsymbol{\beta}(t_{mi})), y_m(t_{mi}))\frac{\partial \boldsymbol{u}_m^T\boldsymbol{\beta}(t_{mi})}{\partial c_{jlk}} \end{aligned} \tag{4.12}$$

where $d_{\boldsymbol{u}_m^T\boldsymbol{\beta}(t_{mi})}\mathrm{Exp}(\hat{\mu}_0(t_{mi}), \boldsymbol{u}_m^T\boldsymbol{\beta}(t_{mi}))^\dagger$ is an adjoint operator since the differential operator occurs on the tangent space of $\mu_m(t_{mi})$ but the tangent vector is on the tangent space of $\mathrm{Exp}(\mu_m(t_{mi}), \tau_m(t_{mi}))$ (see Equation (7) in Fletcher (2013)) and it is a self-adjoint operator which plays a similar role to the parallel transport (Kim et al., 2014). Specifically, the gradient $\nabla_{\boldsymbol{u}_m\boldsymbol{\beta}(t_{mi})} E$ lies in $T_{\hat{\mu}_0(t_{mi})}\mathcal{M}$. However, $\mathrm{Log}(\mathrm{Exp}(\hat{\mu}_0(t_{mi}), \sum_{j=1}^{p} u_{mj}\sum_{k=1}^{K}\boldsymbol{c}_{jk}\phi_k(t_{mi})), y_m(t_{mi}))$ is a tangent vector on the tangent space of $\mathrm{Exp}(\hat{\mu}_0(t_{mi}), \sum_{j=1}^{p} u_{mj}\sum_{k=1}^{K}\boldsymbol{c}_{jk}\phi_k(t_{mi}))$. Therefore, $d_{\boldsymbol{u}_m^T\boldsymbol{\beta}(t_{mi})}\mathrm{Exp}(\hat{\mu}_0(t_{mi}), \boldsymbol{u}_m^T\boldsymbol{\beta}(t_{mi}))^\dagger$ is actually a parallel transport which brings

each error (tangent vector) $\text{Log}(\text{Exp}(\hat{\mu}_0(t_{mi}), \sum_{j=1}^{p} u_{mj} \sum_{k=1}^{K} \boldsymbol{c}_{jk}\phi_k(t_{mi})), y_m(t_{mi}))$ from $T_{\hat{y}_m(t_{mi})}\mathcal{M}$ to $T_{\hat{\mu}_0(t_{mi})}\mathcal{M}$ where $\hat{y}_m(t_{mi}) = \text{Log}(\text{Exp}(\hat{\mu}_0(t_{mi}), \sum_{j=1}^{p} u_{mj} \sum_{k=1}^{K} \boldsymbol{c}_{jk}\phi_k(t_{mi})), y_m(t_{mi}))$.

Therefore, we can obtain the estimated coefficient $\hat{c}_{jlk}$ by setting some convergence conditions. However, $d_{\boldsymbol{u}_m^T\boldsymbol{\beta}(t_{mi})}\text{Exp}(\hat{\mu}_0(t_{mi}), \boldsymbol{u}_m^T\boldsymbol{\beta}(t_{mi}))^{\dagger}\text{Log}(\text{Exp}(\hat{\mu}_0(t_{mi}), \boldsymbol{u}_m^T\boldsymbol{\beta}(t_{mi})), y_m(t_{mi}))$ depends on the specific Riemannian manifold.

In order to avoid the derivatives of the exponential map of composite function, we consider an alternative of the gradient descent algorithm by expanding the mean structure, which is more convenient for functional manifold-valued data. We propose a two-stage approach that in the first procedures, the estimation of mean structure, the covariance structure is not considered, and in the second procedures, we estimate the covariance structure based on the estimated mean structure.

The goal of minimizing Equation (4.11) is to find weights c_{jlk} of a series of basis functions which could model a tangent vector $\text{Log}(\hat{\mu}_0(t_{mi}), y_m(t_{mi}))$. In other words, at each time point t_{mi}, we aim to calculate a minimizer c_{jlk} defined by

$$\underset{c_{jlk}\in\mathbb{R}}{\arg\min} \|\text{Log}(\text{Exp}(\hat{\mu}_0(t_{mi}), \sum_{j=1}^{p} u_{mj} \sum_{k=1}^{K} \boldsymbol{c}_{jk}\phi_k(t_{mi})), y_m(t_{mi}))\|^2 \tag{4.13}$$

which is equivalent to

$$\begin{aligned} &\text{Exp}(\hat{\mu}_0(t_{mi}), \sum_{j=1}^{p} u_{mj} \sum_{k=1}^{K} \boldsymbol{c}_{jk}\phi_k(t_{mi})) = y_m(t_{mi}) \\ \Leftrightarrow\ &\sum_{j=1}^{p} u_{mj} \sum_{k=1}^{K} \boldsymbol{c}_{jk}\phi_k(t_{mi}) = \text{Log}(\hat{\mu}_0(t_{mi}), y_m(t_{mi})) \end{aligned} \tag{4.14}$$

Specifically, it is

$$u_{m1}\phi_1(t)\boldsymbol{c}_{11} + \cdots + u_{mp}\phi_K(t)\boldsymbol{c}_{pK} = \text{Log}(\hat{\mu}_0(t), y_m(t)) \tag{4.15}$$

Since the batch-specific covariates $u_{mj}, j = 1, ..., p$, the basis functions $\phi_k(t), k = 1, ..., K$, the response variables $y_m(t), m = 1, ..., M$ are given and the sample Fréchet mean $\hat{\mu}_0(t)$ can be computed. Equation (4.15) shows that the optimization of mean structure is a standard multiple linear regression model with no intercept. Thus c_{jlk} can be estimated

by a least square approach.

The estimated mean structure for the m-th curve is given by

$$\hat{\mu}_m(t) = \text{Exp}(\hat{\mu}_0(t), \sum_{j=1}^{p} u_{mj} \sum_{k=1}^{K} \hat{\boldsymbol{c}}_{jk}\phi_k(t)) \tag{4.16}$$

where $\hat{\boldsymbol{c}}_{jk} = (\hat{c}_{j1k}, ..., \hat{c}_{jdk})$.

Estimation of covariance structure

From the section above, we obtain a mean structure $\mu_m(t)$ which is a continuous function on $\mathcal{M}$. In this part, we estimate the covariance structure by supposing the data is a manifold-valued function following a wrapped Gaussian process with functional mean structure $\mu_m(t)$, which is defined as

$$y_m(t) \sim \mathcal{GP}_{\mathcal{M}}(\mu_m(t), \tau_m(t)) \tag{4.17}$$

where $\mu_m(t) \in \mathcal{M}$ refers to the mean structure of Equation (4.2) and $\tau_m(t) \in T_{\mu_m(t)}\mathcal{M}$ is a tangent vector-valued function modelling the tangent vector from mean structure to random curves. The tangent vectors are usually multi-dimensional, therefore, the covariance matrix is supposed to be a block matrix and each block refers to a covariance matrix for one dimension. For example, assume there is wrapped Gaussian process on a 2-dimensional Riemannian manifold and covariance matrices for the first dimension and for the second dimensional are A and B respectively. Thus the covariance matrix for the wrapped Gaussian process is

$$\begin{pmatrix} A & 0 \\ 0 & B \end{pmatrix}$$

However, the correlation between different dimensions are ignored. One approach to address this issue to apply the convolved Gaussian process (Boyle and Frean, 2005a). For computational reasons, we use d independent Gaussian processes instead to model each dimension of tangent vector respectively.

.From the discussion in Section 2.1.2, the hyper-parameters $\boldsymbol{\theta}$ in kernel function for the

l-th dimension of the m-th curve can be estimated by maximising the marginal likelihood function

$$(2\pi)^{-\frac{N_m}{2}} \det(\Sigma)^{-\frac{1}{2}} e^{-\frac{1}{2}\mathrm{Log}_l(\hat{\boldsymbol{\mu}}_m, \boldsymbol{y}_m)^T \Sigma^{-1} \mathrm{Log}_l(\hat{\boldsymbol{\mu}}_m, \boldsymbol{y}_m)} \tag{4.18}$$

where $\mathrm{Log}_l(\hat{\boldsymbol{\mu}}_m, \boldsymbol{y}_m)$ refers to the l-th dimension of each element in $\mathrm{Log}(\hat{\boldsymbol{\mu}}_m, \boldsymbol{y}_m)$, $\hat{\boldsymbol{\mu}}_m = (\hat{\mu}_m(t_{m1}), ..., \hat{\mu}_m(t_{mN_m}))$ and $\boldsymbol{y}_m = (y_{m1}, ..., y_{mN_m})$. In addition, Σ is covariance matrix with a given kernel $k(\cdot, \cdot)$. Moreover, the kernel could be squared exponential kernel, linear kernel or Matérn kernel. A suitable kernel is based on the specific application.

With the estimated hyper-parameters $\hat{\boldsymbol{\theta}}$ in kernel, the estimation of covariance structure is given as

$$\hat{\tau}(t) \sim GP(k(t, \boldsymbol{x})^T k(\boldsymbol{x}, \boldsymbol{x})^{-1} \mathrm{Log}(\hat{\boldsymbol{\mu}}, \boldsymbol{y}), k(t, t) - k(t, \boldsymbol{x}) k(\boldsymbol{x}, \boldsymbol{x})^{-1} k(t, \boldsymbol{x}) | \hat{\boldsymbol{\theta}}) \tag{4.19}$$

where $\boldsymbol{x}$ is the vector of observed functional covariates.

Update mean structure and covariance structure

After obtaining the estimated mean structure and covariance structure, we are able to make predictions with given new inputs. In order to improve the performance of our model, we introduce an algorithm which can update the estimated mean structure and update the estimated covariance structure iteratively.

The loss function of the m-th curve at time point t_{mi} with the estimated mean structure and estimated covariance structure is given as

$$E = \sum_{m=1}^{M} \sum_{i=1}^{N_m} d_{\mathcal{M}}(\mathrm{Exp}(\mathrm{Exp}(\hat{\mu}_0(t_{mi}), \sum_{j=1}^{p} u_{mj} \sum_{k=1}^{K} \hat{\boldsymbol{c}}_{jk} \phi_k(t_{mi})), \hat{\tau}_m(t_{mi})), y_m(t_{mi}))^2 \tag{4.20}$$

For simplicity, we still use a vector $\hat{\boldsymbol{c}}_{jk} = (\hat{c}_{j1k}, ..., \hat{c}_{jdk})$ to represent a set of weights of a basis function. However, we should notice that the weights are calculated independently.

Therefore, we can derive the gradient of (4.20) with respect to $\hat{c}_{jlk}$ which is

$$\nabla_{\hat{c}_{jlk}} E = -\sum_{m=1}^{M}\sum_{i=1}^{N_m} d_{\mu_m(t_{mi})}\mathrm{Exp}(\mu_m(t_{mi}), \hat{\tau}_m(t_{mi}))^{\dagger}\mathrm{Log}(y_m(t_{mi}), \mathrm{Exp}(\mu_m(t_{mi}), \hat{\tau}_m(t_{mi})))) \\ \frac{\partial \mathrm{Exp}(\hat{\mu}_0(t_{mi}), \sum_{j=1}^{p} u_{mj} \sum_{k=1}^{K} \boldsymbol{c}_{jk}\phi_k(t_{mi}))}{\partial \sum_{j=1}^{p} u_{mj} \sum_{k=1}^{K} \boldsymbol{c}_{jk}\phi_k(t_{mi})} u_{mj}\phi_k(t_{mi}). \tag{4.21}$$

The coefficients c_{jlk} can be updated by the gradient descent algorithm (4.21) directly. However, the implementation is computationally expensive (as discussed in Section 4.2.1). Thus, we can alternatively use another two-stage method to update the coefficients c_{jlk}. Specifically, in the first step, we update the mean structure with the given estimated covariance structure; secondly, we estimate the coefficients c_{jlk} by converting the optimization to a multiple linear regression.

The mean structure can be updated by a gradient descent algorithm where the gradient is

$$\nabla_{\mu_m(t_{mi})} E = -d_{\mu_m(t_{mi})}\mathrm{Exp}(\mu_m(t_{mi}), \hat{\tau}_m(t_{mi}))^{\dagger}\mathrm{Log}(y_m(t_{mi}), \mathrm{Exp}(\mu_m(t_{mi}), \hat{\tau}_m(t_{mi})))). \tag{4.22}$$

In practice, a variational method for gradient descent algorithm (Kim et al., 2014) can be used as a substitution for gradient in Equation (4.22), which perserves equivalence. Thereafter, we update the mean structure from $\hat{\mu}_m(t)$ to $\hat{\mu}_m^{(1)}(t)$ ($^{(1)}$ means the 1-st iteration). As discussed in Section 4.2.1, the estimation of mean structure converts to a multiple linear regression, the coefficients c_{jlk} can be estimated by least squares approach where the model is

$$u_{m,1}\phi_1(t_{mi})\boldsymbol{c}_{11} + \cdots + u_{m,p}\phi_K(t_{mi})\boldsymbol{c}_{pK} = \mathrm{Log}(\hat{\mu}_0(t_{mi}), \hat{\mu}_m^{(1)}(t_{mi}))$$

With the updated mean structure $\hat{\mu}_m^{(1)}(t)$, we can re-calculate the hyper-parameters in covariance structure. The only difference from Section 4.2.1 is substituting the estimated mean structure by the updated mean structure.

4.2.2 A model with common mean structure

The response variables should be close if the predictor variables are identical. Therefore, it is more reasonable to model a common mean structure for each subject (airline route) instead of a mean structure for each curve (the model in Section 4.2.1). The common mean structure means the curves in a batch share a mean structure. Thus, the model with common mean structure is similar to the model for batch data. For simplicity, we suppose there are two subjects for the flight trajectory example, corresponding to the two airlines. The data is denoted as

$$\mathcal{D} = \{y_{s,m_s}(t_{s,m_s,i}), t_{s,m_s,i}, \boldsymbol{u}_s\}$$

where $s = 1, 2$ refers to the subject, $m_s = 1, ..., M_s$ refers to the curve of subject s, $t_{s,m_s,i} \in \mathbb{R}^+$ refers to a time point, $\boldsymbol{u}_s \in \mathbb{R}^p$ refers to scalar covariates, $y_{s,m_s}(t_{s,m_s,i}) \in \mathcal{M} = S^2$ refers to a data point on Riemannian manifolds at time point $t_{s,m_s,i}$ of the m_s-th curve and $i = 1, ..., N_{m_s}$.

Formally, the model can be defined as

$$y_{s,m_s}(t) = \mathrm{Exp}(\mu_s(t), \tau_{s,m_s}(t)) \tag{4.23}$$

where $\mu_s(t)$ refers to the common mean structure of batch s and $\tau_{s,m_s}(t)$ refers to the covariance structure of the curve m_s in batch s. We should notice that the Riemannian manifold $\mathcal{M}$ is not necessarily a 2-sphere. The manifold $\mathcal{M}$ could be Kendall's shape space or any other Riemannian manifold whose injectivity radius is large enough.

For the sake of completeness, we will also express the procedure of estimating mean structure, estimating covariance structure and updating part.

We also explain the mechanism of the model with individual mean structure schematically. As shown in Figure 4.3, a sample Fréchet mean μ_* is shown as a blue curve. After we add the covariate information $\boldsymbol{u}_m^T\boldsymbol{\beta}(t)$ to μ_*, we get a mean structure μ_m which is the green curve. Thress manifold-valued stochastic processes in batch s can be generated if we add three different random noises $\tau_{s,1}(t)$, $\tau_{s,2}(t)$ and $\tau_{s,3}(t)$ to the mean structure μ_s respectively, which are black curves. Moreover, the black points are observed data.

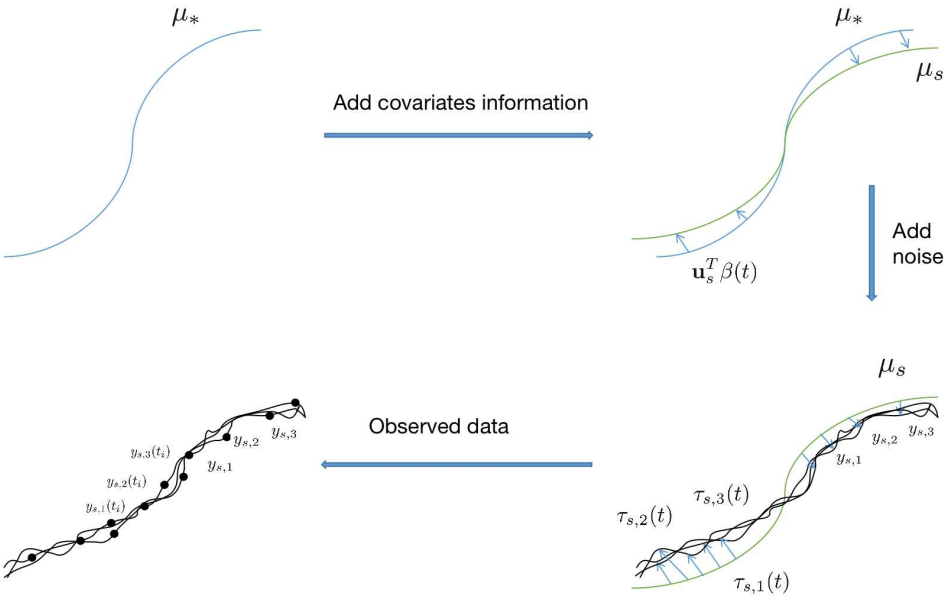


Figure 4.3: Schematic explanation of WGPFR with common mean structure.

Estimation of common mean structure

In this section, we firstly estimate the common mean structure of each subject, so that many manifold-valued curves of an identical subject share a single manifold-valued mean function. In the instance of flight routes, we regard $\mu_1(t)$ and $\mu_2(t)$ as the proposed common mean trajectories of Eastern China Airlines and British Airways from Shanghai to London respectively, which can be estimated from recorded data points by sample Fréchet means directly. The common mean structure $\mu_s(t)$ is different from our original definition (4.1). For model (4.1), each curve has a mean structure and for model (4.23). Meanwhile, the covariance structure is still a Gaussian process at this moment, which models the dependent error in each individual trajectory.

Analogously to the mean structure in Equation (4.5), we define the common mean structure of batch s as

$$\mu_s(t) = \mathrm{Exp}(\hat{\mu}_0(t), \sum_{j=1}^{p} u_{sj} \sum_{k=1}^{K} \boldsymbol{c}_{jk}\phi_k(t)) \tag{4.24}$$

where $\hat{\mu}_0(t)$ is sample Fréchet mean of all data.

The coefficients of basis functions $\phi_k(t)$ (which is defined in (4.5)) can be estimated by

minimizing a loss function

$$\frac{1}{2}\sum_{s=1}^{2}\sum_{m_s=1}^{M_s}\int d_{\mathcal{M}}(\mu_s(t), y_{s,m_s}(t))^2 dt$$

.

In practice, the loss function is approximated by

$$\begin{aligned}
E &= \frac{1}{2}\sum_{s=1}^{2}\sum_{m_s=1}^{M}\int d_{\mathcal{M}}(\mu_s(t), y_{s,m_s}(t))^2 dt \\
&\approx \frac{1}{2}\sum_{s=1}^{2}\sum_{m_s=1}^{M_s}\sum_{i=1}^{N_{m_s}} d_{\mathcal{M}}(\mu_s(t_{s,m_s,i}), y_{s,m_s}(t_{s,m_s,i}))^2 \\
&= \frac{1}{2}\sum_{s=1}^{2}\sum_{m_s=1}^{M}\sum_{i=1}^{N_{m_s}} \|\mathrm{Log}(\mathrm{Exp}(\hat{\mu}_0(t_{s,m_s,i}), \sum_{j=1}^{p} u_{sj}\sum_{k=1}^{K}\boldsymbol{c}_{jk}\phi_k(t_{s,m_s,i}), y_{s,m_s}(t_{s,m_s,i})))\|^2.
\end{aligned} \tag{4.25}$$

Compare to Equation (4.11), the loss function (4.25) is defined as the sum of distances between the common mean structure $\mu_s(t)$ and curve $y_m(t)$ instead of that between each mean structure $\mu_m(t)$ and curve $y_m(t)$. Specifically, if there is only one curve in a batch, we can use the model (4.1) and loss function (4.11); if there are many curves in a batch and these curves share a mean structure, we can use the loss function (4.25).

Since a loss function is often optimized by gradient descent algorithm. We can estimate the coefficient c_{jlk} by moving along the gradient of E in Equation (4.25) with respect to c_{jlk}, which is

$$\begin{aligned}
\nabla_{c_{jlk}} E &= \nabla_{\mu_s(t_{s,m_s,i})} E \frac{\partial \mathrm{Exp}(\hat{\mu}_0(t_{s,m_s,i}), \sum_{j=1}^{p} u_{sj}\sum_{k=1}^{K}\boldsymbol{c}_{jk}\phi_k(t_{s,m_s,i}))}{\partial \boldsymbol{c}_{jk}} \\
&= -\sum_{s=1}^{2}\sum_{m_s=1}^{M_s}\sum_{i=1}^{N_{m_s}} d_{\mu_s(t_{s,m_s,i})}\mathrm{Exp}(\hat{\mu}_0(t_{s,m_s,i}), \sum_{j=1}^{p} u_{sj}\sum_{k=1}^{K}\boldsymbol{c}_{jk}\phi_k(t_{s,m_s,i}))^{\dagger} \\
&\quad \mathrm{Log}(\mathrm{Exp}(\hat{\mu}_0(t_{s,m_s,i}), \sum_{j=1}^{p}\sum_{k=1}^{K} u_s\boldsymbol{c}_{jk}\phi_k(t_{s,m_s,i})), y_{s,m_s}(t_{s,m_s,i})) \\
&\quad \frac{\partial \mathrm{Exp}(\hat{\mu}_0(t_{s,m_s,i}), \sum_{j=1}^{p} u_{sj}\sum_{k=1}^{K}\boldsymbol{c}_{jk}\phi_k(t_{s,m_s,i}))}{\partial \sum_{j=1}^{p} u_{sj}\sum_{k=1}^{K}\boldsymbol{c}_{jk}\phi_k(t_{s,m_s,i})} u_s\phi_k(t_{s,m_s,i})
\end{aligned} \tag{4.26}$$

As we discussed above, to avoid the complicated calculation of gradient, the estimation

of coefficients in mean structure can be implemented alternatively by a least squared method. Suppose the estimated coefficient and the estimated mean structure are denoted by $\hat{c}_{jlk}$ and $\hat{\mu}_s(t)$ respectively.

The equation for estimating the common mean structure in wrapped Gaussian process functional regression model is

$$\hat{\mu}_s(t) = \text{Exp}(\hat{\mu}_0(t), \sum_{j=1}^{p} u_{sj} \sum_{k=1}^{K} \hat{\boldsymbol{c}}_{jk}\phi_k(t)) \tag{4.27}$$

which can be used to make a prediction of a common mean structure.

Estimation of covariance structure

We use the estimated coefficients $\hat{c}_{jlk}$ from the section above to estimate the hyper-parameters $\boldsymbol{\theta}$ in the kernel of the covariance structure for the m-th trajectory of subject s, which is modelled by

$$\tau_{m_s}(t_{s,m_s,i}) = \text{Log}(\hat{\mu}_s(t_{s,m_s,i}), y_{s,m_s}(t_{s,m_s,i})) \tag{4.28}$$

We can suppose that $y_{s,m_s}(t)$ follows a wrapped Gaussian process where the mean function is defined as $\hat{\mu}_s(t)$. Thus, for the m_s-the curve, the hyper-parameters $\boldsymbol{\theta}_{m_s l},\ l = 1, ..., d$ in kernel function $k : \mathbb{R}^Q \times \mathbb{R}^Q \to \mathbb{R}$ can be approximated by maximizing the marginal likelihood function that is

$$\boldsymbol{\theta}_{m_s l} \approx \arg\max_{\boldsymbol{\theta}_{m_s l}} \frac{1}{|\Sigma|^{-\frac{1}{2}}} e^{-\frac{1}{2}\text{Log}_l(\hat{\boldsymbol{\mu}}_s,\ \boldsymbol{y}_{s,m_s})^T \Sigma^{-1} \text{Log}_l(\hat{\boldsymbol{\mu}}_s,\ \boldsymbol{y}_{s,m_s})} \tag{4.29}$$

where $\text{Log}_l(\hat{\boldsymbol{\mu}}_s,\ \boldsymbol{y}_{s,m_s})$ refers to the l-th dimension of each element in $\text{Log}(\hat{\boldsymbol{\mu}}_s,\ \boldsymbol{y}_{s,m_s})$, $\hat{\boldsymbol{\mu}}_s = (\hat{\mu}_s(t_{s,m_s,1}), ..., \hat{\mu}_s(t_{s,m_s,N_{m_s}}))$ and $\boldsymbol{y}_{s,m_s} = (y_{s,m_s}(t_{s,m_s,1}), ..., y_{s,m_s}(t_{s,m_s,N_{m_s}}))$.

The initially estimated coefficients $\hat{C} = (\hat{\boldsymbol{c}}_{11}, ..., \hat{\boldsymbol{c}}_{pK})$ in common mean structure and initially estimated hyper-parameters $\hat{\boldsymbol{\theta}}$ of kernel function in covariance structure are be labelled as $\hat{C}^{(0)}$ and $\hat{\boldsymbol{\theta}}^{(0)}$ respectively.

The predictive distribution calculated from the common mean structure and covariance

structure is given by

$$\hat{y}_m^{(0)}(t) = \mathrm{Exp}\Big(\hat{\mu}_s^{(0)}(t), \hat{\tau}_{m_s}(t)|\boldsymbol{\theta}_1^{(0)}, ..., \boldsymbol{\theta}_d^{(0)}\Big) \tag{4.30}$$

where $\hat{\tau}_{m_s}(t) \sim GP(k(\boldsymbol{t},t)^T k(\boldsymbol{t},\boldsymbol{t})^{-1}\mathrm{Log}(\hat{\boldsymbol{\mu}}_s, \boldsymbol{y}), k(t,t) - k(\boldsymbol{t},t)k(\boldsymbol{t},\boldsymbol{t})^{-1}k(\boldsymbol{t},t)|\hat{\boldsymbol{\theta}})$ and $\boldsymbol{t}$ refers to observed functional covariates.

Updating mean structure and covariance structure

Furthermore, we try to update the coefficients $\hat{C}$ based on $\hat{\boldsymbol{\theta}}_l^{(0)}, l = 1, ..., d$, from Step 2. In other words, we aim to find new c_{jlk} which minimizes a new loss function

$$E^{(0)} = \sum_{s=1}^{2}\sum_{m_s=1}^{M_s}\int d_{\mathcal{M}}(y_{s,m_s}(t), \hat{y}_{s,m_s}^{(0)}(t))^2 dt. \tag{4.31}$$

Since the new loss function is also a composite function which involves an exponential map in an exponential map, we can calculate the gradient of (4.31) with respect to $\hat{c}_{jlk}^{(0)}$. In particular, we have

$$\begin{aligned}
\nabla_{\hat{c}_{jlk}^{(0)}} E^{(0)} &= \nabla_{\hat{\mu}_s^{(0)}(t)} E^{(0)} \frac{\partial \mathrm{Exp}(\mathrm{Exp}(\hat{\mu}_0(t_{s,m_s,i}), \sum_{j=1}^{p} u_{sj} \sum_{k=1}^{K} \hat{\boldsymbol{c}}_{jk}^{(0)} \phi_k(t_{s,m_s,i})), \hat{\tau}_{m_s}(t)|\hat{\boldsymbol{\theta}}_{m_s}^{(0)})}{\partial \hat{c}_{jlk}} \\
&= -d_{\hat{\mu}_s^{(0)}(t)} \mathrm{Exp}(\hat{\mu}_s^{(0)}(t), \hat{\tau}_{m_s}(t)|\hat{\boldsymbol{\theta}}_{m_s}^{(0)})^{\dagger} \mathrm{Log}(y_{s,m_s}(t), \mathrm{Exp}(\hat{\mu}_s^{(0)}(t), \hat{\tau}_{m_s}(t)|\hat{\boldsymbol{\theta}}_{m_s}^{(0)})) \\
&\quad \frac{\partial \mathrm{Exp}(\mathrm{Exp}(\hat{\mu}_0(t_{s,m_s,i}), \sum_{j=1}^{p} u_{sj} \sum_{k=1}^{K} \hat{\boldsymbol{c}}_{jk}^{(0)} \phi_k(t_{s,m_s,i})), \hat{\tau}_{m_s}(t)|\hat{\boldsymbol{\theta}}_{m_s}^{(0)})}{\partial \hat{c}_{jlk}}
\end{aligned} \tag{4.32}$$

The gradient $\nabla_{\hat{c}_{jlk}^{(0)}} E^{(0)}$ is actually on $T_{\hat{\mu}_0(t)}\mathcal{M}$, however the error between real data $(y_{s,m_s}(t))$ and predictions $(\mathrm{Exp}(\hat{\mu}_s^{(0)}(t), \hat{\tau}_{m_s}(t)|\hat{\boldsymbol{\theta}}_{m_s}^{(0)}))$ are on $T_{\hat{\mu}_s(t)}\mathcal{M}$. By this argument, $d_{\hat{\mu}_s^{(0)}(t)}\mathrm{Exp}(\hat{\mu}_s^{(0)}(t), \hat{\tau}_{m_s}(t)|\hat{\boldsymbol{\theta}}_{m_s}^{(0)})^{\dagger}$ brings the error from $T_{\hat{\mu}_0(t)}\mathcal{M}$ to $T_{\hat{\mu}_s(t)}\mathcal{M}$. After using chain rule of $\sum_{j=1}^{p} u_{sj} \sum_{k=1}^{K} \hat{\boldsymbol{c}}_{jk}^{(0)} \phi_k(t_{s,m_s,i})$ with respect to $c_{jlk}^{(0)}$ and $\mathrm{Exp}(\hat{\mu}_0(t_{s,m_s,i}), \sum_{j=1}^{p} u_{sj} \sum_{k=1}^{K} \hat{\boldsymbol{c}}_{jk}^{(0)} \phi_k(t_{s,m_s,i}))$ with respect to $\sum_{j=1}^{p} u_{sj} \sum_{k=1}^{K} \hat{\boldsymbol{c}}_{jk}^{(0)} \phi_k(t_{s,m_s,i})$, we can obtain the equation above.

Analogous to Section 4.2.1, in order to avoid the derivatives of the composite function on Riemannian manifolds, $c_{jlk}^{(1)}$ can be estimated by a two-stage method that we mentioned

in Section 4.2.1. For some simple Riemannian manifolds, we can calculate the gradient directly; for some complex Riemannian manifolds, we can use a variational gradient descent algorithm (Kim et al., 2014). With the updated mean structure $\mu_s^{(1)}(t)$, we can also re-estimate $\boldsymbol{\theta}_{m_s l}^{(1)}$, $l = 1, ..., d$, by maximizing the likelihood function (4.29) given $c_{jlk}^{(1)}$, that is

$$\boldsymbol{\theta}_{m_s l}^{(1)} \approx \arg\max_{\boldsymbol{\theta}_{m_s l}^{(1)}} \frac{1}{|\Sigma|^{-\frac{1}{2}}} e^{-\frac{1}{2}\text{Log}_l(\hat{\boldsymbol{\mu}}_s^{(1)},\ \boldsymbol{y}_{s,m_s})^T \Sigma^{-1} \text{Log}_l(\hat{\boldsymbol{\mu}}_s^{(1)},\ \boldsymbol{y}_{s,m_s})} \tag{4.33}$$

where $\hat{\boldsymbol{\mu}}_s^{(1)} = (\hat{\mu}_s^{(1)}(t_{s,m_s,1}), ..., \hat{\mu}_s^{(1)}(t_{s,m_s,N_m}))$. By repeating this procedure until some convergence conditions have been satisfied (in practice, we use a early stopping mechanism that is if rmse is not decreasing, we just stop training and select the parameters with minimal rmse), a functional regression with a common mean structure and covariance structure on Riemannian manifolds is achieved.

The algorithm 9 summarizes the steps about how to estimate the coefficients c_{jlk} in common mean structures and the hyper-parameters in covariance structure.

Algorithm 9: Algorithm to estimate and update coefficients c_{jlk} and hyper-parameters $\boldsymbol{\theta}_{m_s l}$.

Input: $y_{s,m_s}(t_{s,m_s,i}), t_{s,m_s,i}, \boldsymbol{u}_s, m_s = 1, ..., M_s, i = 1, ..., N_{m_s}, s = 1, 2, iterate = 1$.

Output: $\hat{c}_{jlk}$ and $\hat{\boldsymbol{\theta}}_{m_s l}$.

1. Compute sample Fréchet mean function $\hat{\mu}_0(t)$ for all curves;
2. Estimate the coefficient $c_{jlk}^{(0)}$;
3. Estimate the hyper-parameter $\boldsymbol{\theta}_{m_s l}^{(0)}$ by maximizing the likelihood function of (4.29) ;
4. Update $c_{jlk}^{(iterate)}$;
5. Update $\boldsymbol{\theta}_{m_s l}^{(iterate)}$ with the updated $c_{jlk}^{(iterate)}$ from Step 4, $iterate += 1$;
6. Repeat Step 4 and 5 until some convergence conditions have been satisfied.

4.3 A more practical approximate model

In the sections above, we introduce a regression model for manifold-valued response variables and Euclidean predictor variables within a probabilistic framework, which is named

wrapped Gaussian process functional regression model. A mean structure (or the common mean structure) is considered on a Riemannian manifold and a covariance structure is considered in a vector field along the corresponding mean structure.

However, in practice, the manifold-valued data are often located on some special Riemannian manifolds, such as 2-sphere, hyper-sphere or Kendall's shape space, which can be embedded into a higher-dimensional Euclidean space. In addition, the exponential map and inverse exponential map exist almost everywhere on these Riemannian manifolds. Therefore, we can consider the mean structure and covariance structure within an identical tangent space of a manifold-valued point simultaneously, which is defined in Equation (4.34). Actually, numerical experiments have shown that this model is an accurate approximation of the previous model in Equation (4.1). Moreover, the degree of numerical rounding error may be less in this approximate model, since we must use exponential map and inverse exponential map on two different spaces (tangent space along $\mu_0(t)$ and tangent space along $\mu_m(t)$ or $\mu_s(t)$) in the original model but use exponential map and inverse exponential map in one space (tangent space along $\mu_0(t)$). As a consequence, the predictive error is better than that of the original model in some scenarios of simulation study and this is confirmed in Chapter 5.

In the following, we will define the model, describe inference procedures and show that the loss function for estimating mean structure can be converted to a standard multiple linear regression. The calculation is efficient, since in the optimization problem, it is not necessary to use the gradient descent algorithm.

4.3.1 An approximate model with individual mean structure

As previously, we suppose the m-th curve on a Riemannian manifold is denoted by $y_m(t),\ t \in \mathcal{T},\ m = 1, \ldots, M$, which forms a set of manifold-valued random functions. Associated with the m-th curve, we have observed real-valued Q-dimensional functional covariates $\boldsymbol{x}_m(t)$ and batch-specific scalar covariates $\boldsymbol{u}_m \in \mathbb{R}^p$ which are defined in a conventional way in a functional space of real-valued functions and in Euclidean space, respectively. In order to identify the nonlinear probabilistic relationship between response variables $y_m(t) \in \mathcal{M}$ and the corresponding covariates $\boldsymbol{x}_m(t)$ and $\boldsymbol{u}_m$ in which the mean structure and covariance structure are in the same space, we introduce an approximate

framework of wrapped Gaussian process functional regression model for batch data on Riemannian manifolds as follows

$$y_m(t) = \text{Exp}(\mu_0(t), \eta_m(t) + \tau_m(t)),\ y_m(t) \in \mathcal{M},\ \text{for } m = 1, \ldots, M. \tag{4.34}$$

In practice, we estimate $\mu_0(t)$ using the Fréchet sample mean of the data, under the assumption that the Fréchet sample mean $\hat{\mu}_0(t)$ is either $\mu_0(t)$ or close to $\mu_0(t)$ when the sample size is large, the sample mean is close to the population mean. Therefore, $\eta_m(t) \in T_{\mu_0(t)}\mathcal{M}$ refers to the functional tangent vectors from the intrinsic Fréchet population mean function $\mu_0(t)$ to mean structure of each curve $\mu_m(t)$ which is defined in Equation (4.36). In addition, $\tau_m(t) \in T_{\mu_0(t)}\mathcal{M}$ refers to the residual from $\text{Log}(\mu_0(t), \mu_m(t))$ to $\text{Log}(\mu_0(t), y_m(t))$, i.e.

$$\begin{aligned} \eta_m(t) &= \text{Log}(\mu_0(t), \mu_m(t)) \\ \tau_m(t) &= \text{Log}(\mu_0(t), y_m(t)) - \eta_m(t) \end{aligned} \tag{4.35}$$

The main difference between model (4.34) and model (4.1) is the space of covariance structure. Specifically, the covariance structure of model (4.34) is in $T_{\mu_0(t)}\mathcal{M}$ while that of model (4.1) is in $T_{\mu_m(t)}\mathcal{M}$.

The l-th element of the unknown tangent vector-valued function $\tau_m(t)$ is modelled by a zero mean Gaussian process with kernel $k(\cdot, \cdot; \boldsymbol{\theta}_{ml})$. When the dimension of $\boldsymbol{x}(t)$ is large, we could use a parametric covariance function (Shi and Choi, 2011). We should notice that the model is still in the framework of wrapped Gaussian process functional regression and we cannot model the covariance structure by a standard GPR directly. Analogously to the mean structure in Section 4.2.1, we define the mean structure of approximate model as

$$\mu_m(t) = \text{Exp}(\mu_0(t), \eta_m(t)) \tag{4.36}$$

where tangent vector filed $\eta_m(t) = \boldsymbol{u}_m^T \boldsymbol{\beta}(t) \in T_{\mu_0(t)}\mathcal{M}$, $\boldsymbol{u}_m \in \mathbb{R}^p$ is a batch-specific p-dimensional vector of scalar covariates, $\boldsymbol{\beta}(t) = (\boldsymbol{\beta}_1(t), ..., \boldsymbol{\beta}_p(t))$ is a collection of p vector fields along $\mu_0(t)$ and must be estimated from the data, $\boldsymbol{\beta}_j(t) \in T_{\mu_0(t)}\mathcal{M}$ is a tangent

vector field. For each t, suppose $(\boldsymbol{v}_1(t), ..., \boldsymbol{v}_d(t))$ is a basis of $T_{\mu_0(t)}\mathcal{M}$ (not necessarily orthonormal). Beasides, for any component $\nu(t)$ in $\boldsymbol{v}_l(t)$, we can calculate its Karhunen-Loéve expansion as

$$\begin{aligned} \nu(t) &= \sum_{k=1}^{\infty} w_k \phi_k(t) \\ &\approx \sum_{k=1}^{K} w_k \phi_k(t). \end{aligned} \tag{4.37}$$

Furthermore, any tangent-vector valued function can be approximated by a set of conventional Euclidean basis functions, e.g. B-spline:

$$\begin{aligned} \boldsymbol{\beta}_j(t) &= \sum_{l=1}^{d} a_{jl} \boldsymbol{v}_l(t) \\ &= \sum_{l=1}^{d} a_{jl} (\sum_{k=1}^{K} b_{1kl}\phi_k(t), ..., \sum_{k=1}^{K} b_{dkl}\phi_k(t)) \\ &= \sum_{l=1}^{d} a_{jl} \sum_{k=1}^{K} \boldsymbol{b}_{kl}\phi_k(t) \\ &= \sum_{k=1}^{K} \boldsymbol{c}_{jk}\phi_k(t). \end{aligned} \tag{4.38}$$

We can therefore parametrize the mean structure as follows

$$\begin{aligned} \mu_m(t) &= \mathrm{Exp}(\mu_0(t), \eta_m(t)) \\ &= \mathrm{Exp}(\mu_0(t), \sum_{j=1}^{p} u_{mj}\boldsymbol{\beta}_j(t)) \\ &= \mathrm{Exp}(\mu_0(t), \sum_{j=1}^{p} u_{mj} \sum_{k=1}^{K} \boldsymbol{c}_{jk}\phi_k(t)) \end{aligned} \tag{4.39}$$

Thus, the estimation of unknown functional tangent vectors $\boldsymbol{\beta}(t)$ is converted to the estimation of coefficients c_{jlk} by minimising the loss function

$$\sum_{m=1}^{M} \int \frac{1}{2} d_{\mathcal{M}}(\mu_m(t), y_m(t))^2 dt. \tag{4.40}$$

Moreover, the covariance structure models the residual, which is also functional tangent

vector in $T_{\mu_0(t)}\mathcal{M}$, can be defined as

$$\tau_m(\boldsymbol{x}_m(t)) = \text{Log}(\mu_0(t), y_m(t)) - \eta_m(t), \text{ for } m = 1, ..., M. \tag{4.41}$$

Inference

In this part, we firstly outline how to estimate $\eta_m(t)$ in mean structure without considering the covariance structure. As mentioned previously, each element in a functional tangent vector $\boldsymbol{\beta}_j(t)$ can be represented by a weighted sum of basis functions in vector space. In addition, the norm of inverse exponential map can be chosen as the distance function between manifold-valued points, since the Riemannian manifolds are assumed to be geodesically complete.

The intrinsic Fréchet population mean function $\mu_0(t)$ is also estimated via the sample Fréchet means $\hat{\mu}_0(t)$. We can parametrize the loss function (4.40) as

$$\begin{aligned}
E &= \sum_{m=1}^{M} \int \frac{1}{2} d_{\mathcal{M}}(\mu_m(t), y_m(t))^2 dt \\
&\approx \frac{1}{2} \sum_{m=1}^{M} \sum_{i=1}^{N_m} \|\text{Log}(\mu_m(t_{mi}), y_m(t_{mi}))\|^2 \\
&= \frac{1}{2} \sum_{m=1}^{M} \sum_{i=1}^{N_m} \|\text{Log}(\text{Exp}(\hat{\mu}_0(t_{mi}), \boldsymbol{u}_m^T \boldsymbol{\beta}(t_{mi})), y_m(t_{mi}))\|^2 \\
&= \frac{1}{2} \sum_{m=1}^{M} \sum_{i=1}^{N_m} \|\text{Log}(\text{Exp}(\hat{\mu}_0(t_{mi}), \sum_{j=1}^{p} u_{mj} \sum_{k=1}^{K} \boldsymbol{c}_{jk} \phi_k(t_{mi})), y_m(t_{mi}))\|^2 \\
&= \frac{1}{2} \sum_{m=1}^{M} \sum_{i=1}^{N_m} \| \sum_{j=1}^{p} u_{mj} \sum_{k=1}^{K} \boldsymbol{c}_{jk} \phi_k(t_{mi}) - \text{Log}(\hat{\mu}_0(t_{mi}), y_m(t_{mi}))\|^2
\end{aligned} \tag{4.42}$$

Observing the loss function (4.42), the optimization problem of estimating mean structure converts to a standard multiple linear regression model, since $\boldsymbol{u}_m^T \boldsymbol{\beta}(t)$ models the functional tangent vector $\text{Log}(\mu_0(t), y_m(t)) \in T_{\mu_0(t)}\mathcal{M} \subset \mathbb{R}^d$.

$$\boldsymbol{u}_m^T \boldsymbol{\beta}(t_{mi}) = \text{Log}(\hat{\mu}_0(t_{mi}), y_m(t_{mi}))$$

$$u_{m,1}\phi_1(t_{mi})\boldsymbol{c}_{11} + \cdots + u_{m,p}\phi_K(t_{mi})\boldsymbol{c}_{pK} = \text{Log}(\hat{\mu}_0(t_{mi}), y_m(t_{mi}))$$

Thus $\boldsymbol{c}_{jk}$ can be computed by a least square approach. The estimated mean structure

for the m-th curve is written as

$$\begin{aligned}\hat{\mu}_m(t) &= \text{Exp}(\hat{\mu}_0(t), \hat{\eta}_m(t)) \\ &= \text{Exp}(\hat{\mu}_0(t), \sum_{j=1}^{p} u_{mj} \sum_{k=1}^{K} \hat{\boldsymbol{c}}_{jk} \phi_k(t))\end{aligned} \tag{4.43}$$

Equipped with the estimated mean structure, we can calculate the covariance structure by supposing the manifold-valued random function $y_m(t)$ follows a wrapped Gaussian process with mean structure $\hat{\mu}_m(t)$. The random effect $\tau_m(\boldsymbol{x}_m(t)) \in T_{\hat{\mu}_0(t)}\mathcal{M}$ is a zero mean Gaussian process prior modelling the residuals between $\text{Log}(\hat{\mu}_0(t), y_m(t))$ and $\text{Log}(\hat{\mu}_0(t), \hat{\mu}_m(t))$. We use independent kernels for each dimension of the covariance structure. A more general method to consider the correlation between coordinates will be discussed later.

Using the estimated tangent vector $\hat{\eta}_m(t) \in T_{\mu_0(t)}\mathcal{M}$, we define $z_m(t) = \text{Log}(\hat{\mu}_0(t), y_m(t)) - \hat{\eta}_m(t)$. In other words, $z_m(t)$ refers to a tangent vector modelling the difference between $\text{Log}(\hat{\mu}_0(t), \hat{\mu}_m(t))$ and $\text{Log}(\hat{\mu}_0(t), y_m(t))$ in $T_{\hat{\mu}_0(t)}\mathcal{M}$. We assume that $z_{ml}(t)$ are uncorrelated pairwise and we have already observed data at $t_{mi}, i = 1, ..., N_m$. Then, we can calculate the values of $z_{ml}(t)$ at $t = t_{m1}, ..., t_{mN_m}$, we have

$$\boldsymbol{z}_{ml} = (z_{ml}(t_{m1}), ..., z_{ml}(t_{mN_m}))^T \sim \mathcal{N}(\boldsymbol{0}, K_{ml}) \tag{4.44}$$

where K_l is a $N_m \times N_m$ covariance matrix depending on hyper-parameters $\boldsymbol{\theta}_{ml}$. The hyper-parameters for the l-th dimension, $\boldsymbol{\theta}_l$, can be estimated by maximising sum of marginal likelihood functions for each batch, i.e.

$$(2\pi)^{-\frac{N_m}{2}} \det(K_{ml})^{-\frac{1}{2}} e^{-\frac{1}{2}\boldsymbol{z}_{ml}^T K_l^{-1} \boldsymbol{z}_{ml}} \tag{4.45}$$

where K_{ml} and $\boldsymbol{z}_{ml}$ are defined above.

Then for any new input t_m^*, we know that $\boldsymbol{z}_m(t_m^*) = (z_{ml}(t_m^*), ..., z_{md}(t_m^*)) \in T_{\hat{\mu}_0(t_m^*)}\mathcal{M}$

and the conditional distribution of z^*_{ml} is

$$\begin{aligned} z^*_{ml}|\mathcal{D}_{ml} &\sim \mathcal{N}(\mu^*_{ml}, \Sigma^*_l) \\ \mu^*_{ml} &= \boldsymbol{k}^{*T}_{ml} K^{-1}_{ml} \boldsymbol{z}_{ml} \\ \Sigma^*_{ml} &= k^{**}_{ml} - \boldsymbol{k}^{*T}_{ml} K^{-1}_{ml} \boldsymbol{k}^*_{ml} \end{aligned} \tag{4.46}$$

where $\mathcal{D}_{ml} = \{z_{ml}(t_{m1}), ..., z_{ml}(t_{mN_m})\}$, $\boldsymbol{k}^*_{ml} = (k_{ml}(t_{m1}, t^*_m), ..., k_{ml}(t_{mN_m}, t^*_m))$ and $k^{**}_{ml} = k_{ml}(t^*_m, t^*_m)$.

We can also update the estimated mean structure by the estimated covariance structure in Equation (4.44). Specifically, equipped with the estimated covariance structure, we can update the coefficients c_{jlk} of mean structure by minimizing the new loss function below

$$\begin{aligned} E^{(0)} &= \sum_{m=1}^{M} \int \frac{1}{2} d_{\mathcal{M}}(y_m(t), \hat{y}^{(0)}_m(t))^2 dt \\ &\approx \sum_{m=1}^{M} \sum_{i=1}^{N_m} \frac{1}{2} d_{\mathcal{M}}(y_m(t_{mi}), \hat{y}^{(0)}_m(t_{mi}))^2 \\ &= \frac{1}{2} \sum_{m=1}^{M} \sum_{i=1}^{N_m} \|\mathrm{Log}(y_m(t_{mi}), \hat{y}^{(0)}_m(t_{mi}))\|^2 \\ &= \frac{1}{2} \sum_{m=1}^{M} \sum_{i=1}^{N_m} \|\mathrm{Log}(y_m(t_{mi}), \mathrm{Exp}(\hat{\mu}_0(t_{mi}), \sum_{j=1}^{p} u_{mj} \sum_{k=1}^{K} \boldsymbol{c}_{jk} \phi_k(t_{mi}) + \hat{z}_m))\|^2 \\ &= \frac{1}{2} \sum_{m=1}^{M} \sum_{i=1}^{N_m} \| \sum_{j=1}^{p} \sum_{k=1}^{K} u_{mj} \boldsymbol{c}_{jk} \phi_k(t_{mi}) + \hat{z}_m(t_{mi}) - \mathrm{Log}(\hat{\mu}_0(t_{mi}), y_m(t_{mi}))\|^2 \end{aligned} \tag{4.47}$$

where $\hat{z}_m = (\hat{\boldsymbol{z}}_{m1}(t_{mi}), ..., \hat{\boldsymbol{z}}_{md}(t_{mi}))$.

Thus, with observed data and estimated hyper-parameters $\boldsymbol{\Theta}^{(0)}$, the updated mean structure should model $\mathrm{Log}(\hat{\mu}_0(t_{mi}), y_m(t_{mi})) - \hat{z}_m$ as accuracy as possible for each time point t_{mi}, that is

$$\begin{gathered} \boldsymbol{u}^T_m \boldsymbol{\beta}(t) = \mathrm{Log}(\hat{\mu}_0(t), y_m(t_{mi})) - \hat{z}_m \\ u_{m,1}\phi_1(t)\boldsymbol{c}_{11} + \cdots + u_{m,p}\phi_K(t)\boldsymbol{c}_{pK} = \mathrm{Log}(\hat{\mu}_0(t_{mi}), y_m(t)) - \hat{z}_m \end{gathered}$$

Therefore, the updating procedure of the mean structure converts to a standard multiple linear regression. By using the least square approach, we can obtain the updated functional

coefficients for mean structure, which we denote $C^{(1)}$. Additionally, with the updated functional coefficients $C^{(1)}$ and mean structure $\hat{\mu}_m^{(1)}(t)$, we can re-calculate the hyper-parameters in covariance structure.

The mean structure and covariance structure can be updated iteratively until satisfying some convergence conditions. For example, we could limit the number of iteration or stop training when the root-mean-squared-error is not decreased.

4.3.2 An approximate model with common mean structure

In this part, the data set is also considered as the repeated measurements for different subjects, such as the flight trajectories of Eastern China Airlines and British Airlines.

The data is denoted as

$$\mathcal{D} = \{y_{s,m_s}(t_{s,m_s,i}), t_{s,mi}, \boldsymbol{u}_s\}$$

where $s = 1, 2$ refers to the subject, $m_s = 1, ..., M_s$ refers to the curve of subject s, $t_{s,m_s,i} \in \mathbb{R}^+$ refers to a time point, $\boldsymbol{u}_s \in \mathbb{R}^p$ refers to scalar covariates, $y_{s,m_s}(t_{s,m_s,i}) \in \mathcal{M}$ refers to a manifold-valued point at time $t_{s,m_s,i}$ and $i = 1, ..., N_{m_s}$. Hence, we can define an approximate model where the common mean structure and covariance structure are in the identical tangent space, i.e.

$$y_{s,m_s}(t) = \mathrm{Exp}(\mu_s(t), \eta_{s,m_s}(t) + \tau_{s,m_s}(t)) \tag{4.48}$$

For the sake of completeness, we will explain the estimation of common mean structure, covariance structure and iterative procedure for updating.

Inference

We firstly estimate the common mean structure for the manifold-valued curves in a batch. For instance, we can still regard $\mu_1(t)$ and $\mu_2(t)$ as the mean trajectories of Eastern China Airlines and British Airlines from Shanghai to London respectively.

The common mean structure can be estimated by minimizing the loss function

$$\frac{1}{2}\sum_{s=1}^{2}\sum_{m_s=1}^{M_s}\int d_{\mathcal{M}}(y_{s,m_s}(t),\mathrm{Exp}(\hat{\mu}_0(t),\boldsymbol{u}_s^T\boldsymbol{\beta}(t))^2dt.$$

For some geodesically complete Riemannian manifolds, such as the sphere and Kendall's shape space, the distance function can be taken to be Euclidean norm of inverse exponential map. Therefore, loss function can be approximated by

$$\begin{aligned}
E &= \sum_{s=1}^{2}\sum_{m_s=1}^{M_s}\int \frac{1}{2}d_{\mathcal{M}}(y_{s,m_s}(t),\mathrm{Exp}(\hat{\mu}_0(t),\boldsymbol{u}_s^T\boldsymbol{\beta}(t)))^2dt \\
&\approx \frac{1}{2}\sum_{s=1}^{2}\sum_{m_s=1}^{M_s}\sum_{i=1}^{N_{m_s}} d_{\mathcal{M}}(y_{s,m_s}(t_{s,m_s,i}),\mathrm{Exp}(\hat{\mu}_0(t_{s,m_s,i}),\boldsymbol{u}_s^T\boldsymbol{\beta}(t_{s,m_s,i})))^2 \\
&= \frac{1}{2}\sum_{s=1}^{2}\sum_{m_s=1}^{M_s}\sum_{i=1}^{N_{m_s}} \|\mathrm{Log}(y_{s,m_s}(t_{s,m_s,i}),\mathrm{Exp}(\hat{\mu}_0(t_{s,m_s,i}),\sum_{j=1}^{p}u_{sj}\sum_{k=1}^{K}\boldsymbol{c}_{jk}\phi_k(t_{s,m_s,i})))\|^2 \\
&= \frac{1}{2}\sum_{s=1}^{2}\sum_{m_s=1}^{M_s}\sum_{i=1}^{N_{m_s}} \|\sum_{j=1}^{p}u_{sj}\sum_{k=1}^{K}\boldsymbol{c}_{jk}\phi_k(t_{s,m_s,i}) - \mathrm{Log}(\hat{\mu}_0(t_{s,m_s,i}) - y_{s,m_s}(t_{s,m_s,i}))\|^2
\end{aligned} \tag{4.49}$$

Analogous to the discussion in Section 4.2.2, the minimization problem converts to a multiple linear regression. Suppose that the estimated tangent vector function in mean structure is written by $\hat{\eta}_s(t)$, the equation for estimating common mean structure of subject s is given by

$$\begin{aligned}
\hat{\mu}_s(t) &= \mathrm{Exp}(\hat{\mu}_0(t),\sum_{j=1}^{p}u_{sj}\sum_{k=1}^{K}\hat{\boldsymbol{c}}_{jk}\phi_k(t)) \\
&= \mathrm{Exp}(\hat{\mu}_0(t),\hat{\eta}_s(t))
\end{aligned} \tag{4.50}$$

which provides a common mean structure for repeated measurements on Riemannian manifolds.

We use the estimated functional tangent vectors $\hat{\eta}_s(t)$ to estimate the hyper-parameters $\boldsymbol{\theta}$ in kernel of the covariance structure for the m-th trajectory in subject s. The covariance

structure is also alternatively considered in $T_{\hat{\mu}_0(t)}\mathcal{M}$, which is modelled by

$$z_{s,m_s}(t) = \mathrm{Log}(\hat{\mu}_0(t), y_{s,m_s}(t)) - \hat{\eta}_s(t) \tag{4.51}$$

Therefore, we obtain

$$\boldsymbol{z}_{s,m_s,l} = (z_{s,m_s,l}(t_{s,m_s,1}), ..., z_{s,m_s,l}(t_{s,m_s,N_{m_s}})) \sim \mathcal{N}(\mathbf{0}, K_l) \tag{4.52}$$

Then for any new input t^*_{s,m_s}, $\boldsymbol{z}_{s,m_s}(t^*_{s,m_s}) = (z_{s,m_s,1}(t^*_{s,m_s}), ..., z_{s,m_s,d}(t^*_{s,m_s})) \in T_{\hat{\mu}_0(t^*_{s,m_s})}\mathcal{M}$ and the conditional distribution of $z^*_{s,m_s,l} = z_{s,m_s,l}(t^*_{s,m_s})$ is

$$\begin{aligned} z^*_{s,m_s,l}|\mathcal{D}_{s,m_s} &\sim \mathcal{N}(\mu^*_{s,m_s,l}, \Sigma^*_{ml}) \\ \mu^*_{s,m_s,l} &= \boldsymbol{k}^{*T}_{ml} K^{-1}_{ml} \boldsymbol{z}_{s,m_s,l} \\ \Sigma^*_{ml} &= k^{**}_{ml} - \boldsymbol{k}^{*T}_{ml} K^{-1}_{ml} \boldsymbol{k}^*_{ml} \end{aligned} \tag{4.53}$$

where $\mathcal{D}_{s,m_s} = \{z_{s,m_s,l}(t_{s,m_s,1}), ..., z_{s,m_s,l}(t_{s,m_s,N_{s,m_s}})\}$, $\boldsymbol{k}^*_{ml} = (k_{ml}(t_{s,m_s,1}, t^*_{s,m_s}), ..., k_{ml}(t_{s,m_s,N_{m_s}}, t^*_{s,m_s}))$ and $k^{**}_{ml} = k_{ml}(t^*_{s,m_s}, t^*_{s,m_s})$. Thus the hyper-parameters $\boldsymbol{\theta}_{ml}$, $l = 1, ..., d$, in kernel $k(\cdot, \cdot; \boldsymbol{\theta}_{ml})$ can be obtained by maximizing the marginal likelihood function that is a multivariate normal distribution

$$\boldsymbol{\theta}_l \approx \arg\max_{\boldsymbol{\theta}_{ml}} \frac{1}{|K_{ml}|^{-\frac{1}{2}}} e^{-\frac{1}{2}\boldsymbol{z}^T_{s,m_s,l} K^{-1}_{ml} \boldsymbol{z}_{s,m_s,l}} \tag{4.54}$$

where $\boldsymbol{z}_{s,m_s,l}$ and K_{ml} are defined above.

The initially estimated coefficients $\hat{c}^{(0)}_{jlk}$ in common mean structure and initially estimated hyper-parameters $\boldsymbol{\Theta} = (\hat{\boldsymbol{\theta}}_1, ..., \hat{\boldsymbol{\theta}}_d)$ in covariance structure could be labelled as $C^{(0)}$ and $\boldsymbol{\Theta}^{(0)}$ respectively, since they have not been updated.

The predictive distribution calculated from the mean structure and covariance structure is given by

$$\hat{y}^{(0)}_{s,m_s}(t) = \mathrm{Exp}\Big(\hat{\mu}_0(t), \hat{\eta}_s(t) + \hat{z}_{s,m_s}(t)|\boldsymbol{\Theta}^{(0)}\Big). \tag{4.55}$$

Furthermore, we can update the functional coefficients $c^{(0)}_{jlk}$ based on $\boldsymbol{\Theta}^{(0)}$. In other words,

our goal is to find new c_{jlk} which minimize a new loss function

$$
\begin{aligned}
E^{(0)} &= \sum_{s=1}^{2}\sum_{m_s=1}^{M_s}\int \frac{1}{2} d_{\mathcal{M}}(y_{s,m_s}(t), \hat{y}^{(0)}_{s,m_s}(t))^2 dt \\
&\approx \sum_{s=1}^{2}\sum_{m_s=1}^{M_s}\sum_{i=1}^{N_{m_s}} \frac{1}{2} d_{\mathcal{M}}(y_{s,m_s}(t_{s,m_s,i}), \hat{y}^{(0)}_{s,m_s}(t_{s,m_s,i}))^2 \\
&= \frac{1}{2}\sum_{s=1}^{2}\sum_{m_s=1}^{M_s}\sum_{i=1}^{N_{m_s}} \|\mathrm{Log}(y_{s,m_s}(t_{s,m_s,i}), \hat{y}^{(0)}_{s,m_s}(t_{s,m_s,i}))\|^2 \\
&= \frac{1}{2}\sum_{s=1}^{2}\sum_{m_s=1}^{M_s}\sum_{i=1}^{N_{m_s}} \|\mathrm{Log}(y_{s,m_s}(t_{s,m_s,i}), \mathrm{Exp}(\hat{\mu}_0(t_{s,m_s,i}), \sum_{j=1}^{p} u_{sj} \sum_{k=1}^{K} \boldsymbol{c}_{jk}\phi_k(t_{s,m_s,i}) + \hat{z}_{s,m_s}))\|^2 \\
&= \frac{1}{2}\sum_{s=1}^{2}\sum_{m_s=1}^{M_s}\sum_{i=1}^{N_{s,m_s}} \| \sum_{j=1}^{p}\sum_{k=1}^{K} u_s \boldsymbol{c}_{jk}\phi_k(t_{s,m_s,i}) + \hat{z}_{s,m_s} - \mathrm{Log}(\hat{\mu}_0(t_{s,m_s,i}), y_{s,m_s}(t_{s,m_s,i}))\|^2
\end{aligned}
\tag{4.56}
$$

The problem of updating mean structure can be converted to a multiple linear regression. With the updated mean structure $\mu_s^{(1)}(t)$, we can also obtain $\boldsymbol{\theta}_{ml}^{(1)}$, $l = 1, ..., d$, by maximizing the likelihood function given $c_{jlk}^{(1)}$, that is

$$
\boldsymbol{\theta}_{ml}^{(1)} \approx \arg\max_{\boldsymbol{\theta}_{ml}^{(1)}} \frac{1}{|K_{ml}|^{-\frac{1}{2}}} \exp\big(-\frac{1}{2}\boldsymbol{z}_{s,m_s,l}^T K_{ml}^{-1} \boldsymbol{z}_{s,m_s,l}\big). \tag{4.57}
$$

By repeating the iterative procedure until some convergence conditions have been satisfied, a concurrent functional regression on Riemannian manifold with uncertainty has been achieved.

We can see that the steps of inference are similar to the inference in Section 4.2.1. However, the difference is the estimation of mean structures. The algorithm 10 summarizes the steps above.

Algorithm 10: Algorithm to estimate and update coefficients c_{jlk} and hyper-parameters $\boldsymbol{\theta}_{m_s l}$.

Input: $y_{s,m_s}(t_{s,m_s,i}), t_{s,m_s,i}, \boldsymbol{x}_{s,m_s}(t_{s,m_s,i}), \boldsymbol{u}_s, m_s = 1, ..., M_s, i = 1, ..., N_{m_s}, s = 1, 2, iterate = 1.$

Output: $\hat{c}_{jlk}$ and $\hat{\boldsymbol{\theta}}_{m_s l}$.

1. Compute sample Fréchet mean function $\hat{\mu}_0(t)$ for all curves;
2. Estimate the coefficient $c_{jlk}^{(0)}$;
3. Estimate the hyper-parameter $\boldsymbol{\theta}_{m_s l}^{(0)}$ by maximizing the likelihood function of (4.54) ;
4. Update $c_{jlk}^{(iterate)}$;
5. Update $\boldsymbol{\theta}_{m_s l}^{(iterate)}$ with the updated $c_{jlk}^{(iterate)}$, $iterate += 1$;
6. Repeat Step 4 and 5 until some convergence conditions have been satisfied.

The numerical comparison between model (4.1) and the approximate model (4.34) is presented in the next section. From the experiments, we can see that in some special Riemannian manifolds, such as sphere and Kendall's shape space, it provides an accurate approximation.

Chapter 5

Simulation Study and Data Application for Manifold-Valued Models

We demonstrate the wrapped Gaussian process functional regression model on two simulated manifold-valued data sets and one real data set. First, in Section 5.2, our model is applied on the S^2 which is a typical smooth Riemannian manifold with large injectivity radius (see Section 2.4.1 for more details) and easy to visualize. Next, in Section 5.3, we test its performance of prediction on Kendall's shape space which is often used in computer vision application. Finally, in Section 5.4, we study the relationship between some real predictor variables (such as time and flight company) and the corresponding response variables (flight routes). In addition, several other models are also used for comparison. For example, we implement a functional linear regression model for manifold-valued data (the mean structure of WGPFR) and wrapped Gaussian process regression with Fréchet mean model, for various scenarios of predictions. We also compute the performance of the approximate wrapped Gaussian process functional regression model in which we consider the mean structure and covariance structure for each data point in the identical tangent space (see Section 4.3). More details about numerical results and comparison are described in the following sections.

5.1 Wrapped Gaussian process Fréchet mean regression

The wrapped Gaussian process regression with Fréchet mean point can be seen as an manifold-valued extension of a Gaussian process regression with constant mean and the wrapped Gaussian process with geodesic regression can be considered as an extension of GPR with linear regression mean function, respectively. We summarize this model in this section.

Suppose there is a manifold-valued prior Gaussian process with given training data $(\boldsymbol{x}_i, y_i)$, where $\boldsymbol{x}_i \in \mathbb{R}$, $y_i \in \mathcal{M}$ and $i = 1, ..., n$, for test data $(\boldsymbol{x}^*, y^*)$. The model is defined as

$$y(\boldsymbol{x}) = \text{Exp}(p, \tau(\cdot, \cdot; \boldsymbol{\theta})) \tag{5.1}$$

and the approximated predictive distribution of y^* is given by

$$\begin{aligned} y^*|\boldsymbol{y} &\sim \text{Exp}(m, v) \\ v &\sim \mathcal{N}(k^* K^{-1} \text{Log}(m, \boldsymbol{y}), k^{**} - k^* K^{-1} k^{*T}) \\ k^* &= (k(\boldsymbol{x}_1, \boldsymbol{x}^*), ..., k(\boldsymbol{x}_n, \boldsymbol{x}^*)) \\ k^{**} &= k(\boldsymbol{x}^*, \boldsymbol{x}^*) \end{aligned} \tag{5.2}$$

where $\boldsymbol{y} = (y_1, ..., y_n)$, $k(\cdot, \cdot)$ denotes a covariance function, m denotes the Fréchet mean of $\boldsymbol{y}$, $\text{Log}(m, \boldsymbol{y}) = (\text{Log}(m, y_1), ..., \text{Log}(m, y_n))$ and K denotes a $n \times n$ covariance matrix in which the ij-th entry is $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

Although the predictive distribution of $y^*|\boldsymbol{y}$ cannot be guaranteed to follow a wrapped Gaussian distribution, its expectation is a maximum a posterior and thus it is reasonable to use the expectation of $y^*|\boldsymbol{y}$ as a prediction.

Compared to our wrapped Gaussian process functional regression model, the "mean structure" of wrapped Gaussian process Fréchet mean regression is the Fréchet mean p of all manifold-valued data which is only one point (under the assumption of existence and uniqueness) on Riemannian manifolds. If there are more than one manifold-valued curve, the wrapped Gaussian process regression is unable to capture the curve-specific information while the mean structure $\mu_m(t)$ in WGPFR learns from curve-specific covariates. It

is clearly that p is a fixed point but $\mu_m(t)$ is varying with t. As a consequence, in some cases, such as the flight trajectory data, the predictive error of wrapped Gaussian process functional regression model should less than that of wrapped Gaussian process Fréchet mean regression model, since the mean structure in WGPFR can capture the trend.

5.2 S^2

In this section, we consider two types of numerical experiments. In the first, there are eight non-linear curves on a S^2 and we delete some data points of the 8-th curve in different ways to form different scenarios of prediction. On the one hand, the data points are deleted uniformly at random or in a contiguous block to test the interpolating performance. On the other hand, the last 10 data points of the 8-th curve are deleted to test the long-term extrapolating performance and a small tail (5 data points) of the 8-th curve is deleted to test the short-term extrapolation performance. In this case, each curve has a distinct mean structure. The second kind of data is a little different. There are two groups of batch data on a S^2 in which each group has four similar curves. The difference from case one is that we suppose each group shares a common mean structure. Some data points are deleted on the 4-th curve in group two by same methods in the first kind of numerical experiment. This type of manifold-valued data can be used to test the performance of the model with common mean structure introduced in Section 4.2.2.

5.2.1 Data simulation for model with individual mean structure

The simulated data are shown in Figure 5.1 and in the remainder of this subsection we explain how the data were generated.

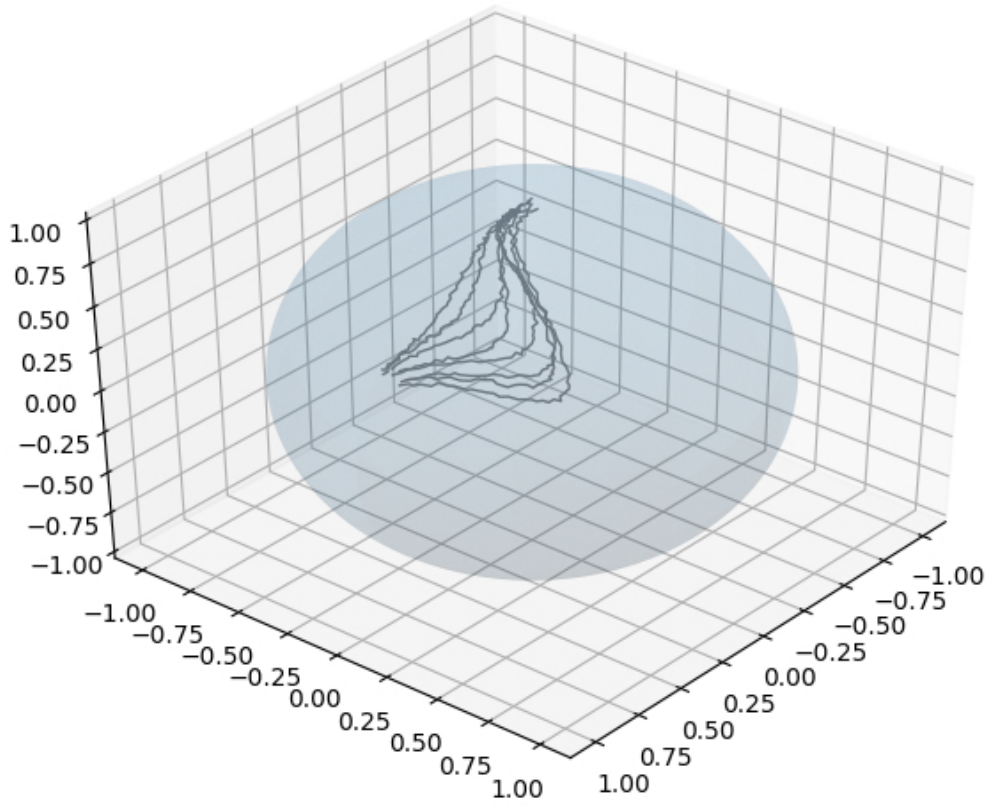


Figure 5.1: Visualization for a data set simulated from model with individual mean structure.

Specifically, the WGPFR model consists of two parts: the mean structure and the covariance structure, see model (4.1) and (4.23). We firstly generate the mean structure based on covariates defined by $u_m = (1, m-1)$ for each curve. All eight curves start from an identical point $p_0 = (0, 0, 1)$ and the functional covariate $\boldsymbol{x}$ is supposed as time t which is equally spaced in interval $(0, 1)$. The mean structure for the m-th curve is given by $\mu_m(t) = \text{Exp}(p_0, (t, 0.1 \times m \times \sin(2.5 \times t)^3, 0))$. Then, we determine the kernel of covariance structure which is discussed at Equation (2.20)

$$k(t_i, t_j) = v_0 \exp(-\frac{1}{2w_0}(t_i - t_j)^2) + a_0 + a_1 t_i t_j + \sigma^2 \delta_{ij}$$

Since S^2 can be embedded into $\mathbb{R}^3$, for each dimension of tangent vector of covariance structure, the hyper-parameters $(v_0, 2_0, a_0, a_1, \sigma)$ are given as $\boldsymbol{\theta}_1 = (0.15, 1, 0.2, 0.01, 0.002)$, $\boldsymbol{\theta}_2 = (0.08, 2, 0.17, 0.005, 0.0015)$ and $\boldsymbol{\theta}_3 = (0.11, 1.3, 0.07, 0.015, 0.0025)$. Given these

three Gaussian processes with zero mean and kernel, we can generate a tangent vector for the covariance structure. However, in practice, the tangent vector may not in the target tangent space because of numerical rounding. For example, if the ideal tangent vector is $(1, 1, 0)$, but our generated tangent vector might be $(0.99, 1, 0.01)$, which will result in the exponential map and inverse exponential map incalculable (defined in Equation (2.33) and Equation (2.32) respectively). Therefore, the tangent vector must be projected into the correct tangent space, which is denoted as $tv_3(t)|\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\theta}_3$. The subscript 3 in tv_3 refers to 3-dimensional. The final part is a random error ϵ which is sampled from three independent and identical distributions $\mathcal{N}(0, 0.02)$, we also must map the random error into the correct tangent spaces. As a consequence, the formula to generate manifold-valued data for numerical experiments is written as

$$y_m(t) = \text{Exp}(\text{Exp}(\mu_m(t), 0.1 \times tv_3(t)|\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\theta}_3), \epsilon), \ m = 1, ..., M. \tag{5.3}$$

The functional covariate t is equally spaced in interval $(0, 1)$ with 60 points. We then generate a data set for eight non-linear manifold-valued curves, that is $M = 8$.

In order to estimate the coefficients in mean structure, it is necessary to identify the basis functions. Specifically, B-splines are used whose degree is 5 and 40 knots are equally spaced in interval $(0, 1.1)$.

5.2.2 Simulation study for model with individual mean structure

In this section, we firstly delete some data points from the 8-th curve in different ways and then calculate the predictions for these deleted data points. The performance of WGPFR model can be assessed by comparing the root mean square error between predictions and real data points. We select all the data points of the first 7 curves and 45 random data points of the 8-th curve as our training data set, so the remaining 15 data points in curve 8 are used for prediction. This scenario of prediction is denoted as Type 1 prediction. There is another scenario of interpolation where all data points in a given contiguous block are totally missing, which is denoted as Type 2 prediction. In practice, the entire data points in the first 7 curves and the remaining of the 8-th curve are used as training data. For example, if the 60 data points in the 8-th curve are written as $(y_{8,1}, y_{8,2}, ..., y_{8,60})$, we

can retain the data $(y_{8,1}, ..., y_{8,25})$ and $(y_{8,45}, ..., y_{8,60})$ as part of training data and use $(y_{8,26}, ..., y_{8,44})$ as test data to form the Type 2 prediction. As to Type 3 prediction, the long-term forecasting, data points $(y_{8,1}, ..., y_{8,50})$ in the 8-th curve and all 7 other curves are supposed as training data which is a typical extrapolation and it is more difficult than interpolation. Analogously to the long-term forecasting, we also test the performance of WGPFR for short-term forecasting, the Type 4 prediction. In Type 4 prediction, we choose $(y_{8,1}, ..., y_{8,55})$ in the 8-th curve as part of training data and the rest as test data. The only difference of training data between these four scenarios of prediction is the training data and test data in the 8-th curve, the differences are shown in Table 5.1.

	training data points in the 8-th curve	test data points
Type 1	45 random data points in the 8-th curve	the rest 15 data points in the 8-th curve
Type 2	$(y_{8,1}, ..., y_{8,25})$ $\cup$ $(y_{8,45}, ..., y_{8,60})$	$(y_{8,26}, ..., y_{8,44})$
Type 3	$(y_{8,1}, ..., y_{8,50})$	$(y_{8,51}, ..., y_{8,60})$
Type 4	$(y_{8,1}, ..., y_{8,55})$	$(y_{8,56}, ..., y_{8,60})$

Table 5.1: The training data and the test data of the 8-th curve in different scenarios.

For comparison, in each scenario, the same training data set is used for several other models, such as functional linear regression on Riemannian manifold(FLRM) and wrapped Gaussian process Fréchet mean regression (WGFmR). Specifically, FLRM is the mean structure (4.2) without covariance structure; WGPFmR consists of mean structure and covariance structure in which the mean structure is the Fréchet mean point for all training data. In addition, the WGPFmR model does not have the updating part.

We replicate each simulation study 100 times because of the randomness in optimization. Thus, we test the performance of our model on thousands of test data points. We use WGPFR* and WGPFR† to denote the original model (4.1) and approximate model (4.34) respectively.

Table 5.2 shows the reference numbers linking to the different models we are going to compare

	Equation Number
WGPFR*	(4.1)
WGPFR†	(4.34)
FLRM	(4.5)
WGFmR	(5.1)

Table 5.2: Different models used in numerical experiments.

The numerical results reported in Table 5.3 are the average of root-mean-square-error in every single replication. Using the embedding $S^2 \subseteq \mathbb{R}^3$, it is reasonable to use the Euclidean norm between points as a distance function (chordal metric), which provides a method to calculate the rmse. The correlation coefficient r between predictions and test data is also calculated. Specifically, rmse is calculated by

$$\text{rmse} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}$$

and r is calculated by

$$\text{rmse} = \frac{\sum_{i=1}^{N}(\hat{y}_i - \bar{\hat{y}})\sum_{i=1}^{N}(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{N}(\hat{y}_i - \bar{\hat{y}})^2 \sum_{i=1}^{N}(y_i - \bar{y})^2}}$$

where $\hat{y}_i$ is the prediction of y_i, $\bar{y} = \frac{1}{N}\sum_{i=1}^{N} y_i$ and $\bar{\hat{y}} = \frac{1}{N}\sum_{i=1}^{N} \hat{y}_i$.

	Type 1	Type 2	Type 3	Type 4
rmse				
WGPFR*	0.0180	0.0171	0.0138	0.0126
WGPFR†	**0.0163**	**0.0134**	**0.0119**	**0.0115**
FLRM	0.0357	0.0476	0.0474	0.0351
WGFmR	0.1249	0.1517	0.2878	0.2211
r				
WGPFR*	0.9995	0.9779	0.9922	0.9870
WGPFR†	0.9996	0.9792	0.9924	0.9858
FLRM	0.9994	0.9939	0.9884	0.9860
WGPFmR	0.9740	0.6643	0.2149	0.1741

Table 5.3: Root-mean-square-error and correlation coefficient for several models with four types of predictions and equally spaced data where bold number refers to the minimal rmse.

In Type 1 and Type 2 prediction, WGPFR† provides the best result when the prediction of WGPFR* is precise as well. In these two scenarios, the approximate model outperforms the original model, which is because, in practice, there are some calculation errors when we map the predictive tangent vectors to the target tangent spaces in the usage of exponential map and inverse exponential map of mean structure in the original model. However, there is no such error in the approximate model, since we consider the mean structure and covariance structure in the same tangent spaces. FLRM provides a reasonably good result for prediction while the performance of WGPFmR is relatively inaccurate (0.0357 compared to 0.1249), because FLRM only learns the mean structure of training data which might be more useful to forecasting. This conjecture is confirmed empirically in Type 3 prediction and type 4 prediction.

In Type 3 prediction, the original WGPFR model and approximate WGPFR give accurate predictions and the FLRM still performs well. However, WGPFmR is failed to predict, since when test data are distant from the training data, the output of GRP is usually inaccurate and then the mean structure mainly determines the accurate of prediction. This is also the reason that WGPFR* and WGPFR† predict test data more

precisely in Type 4 prediction. In Type 4 prediction which is also known as short-term forecasting, these two WGPFR models outperform FLRM and WGPFmR. The prediction of FLRM is still better than that of WGPFmR, since in forecasting, the results rely mainly on the mean structure. In addition, the accuracy of both FLRM and WGPFmR become better compared to Type 3 prediction. This is because, in short-term forecasting, the test data are not so far away from the training data.

We present the 95% predictive interval for Type 3 prediction in Figure 5.2.

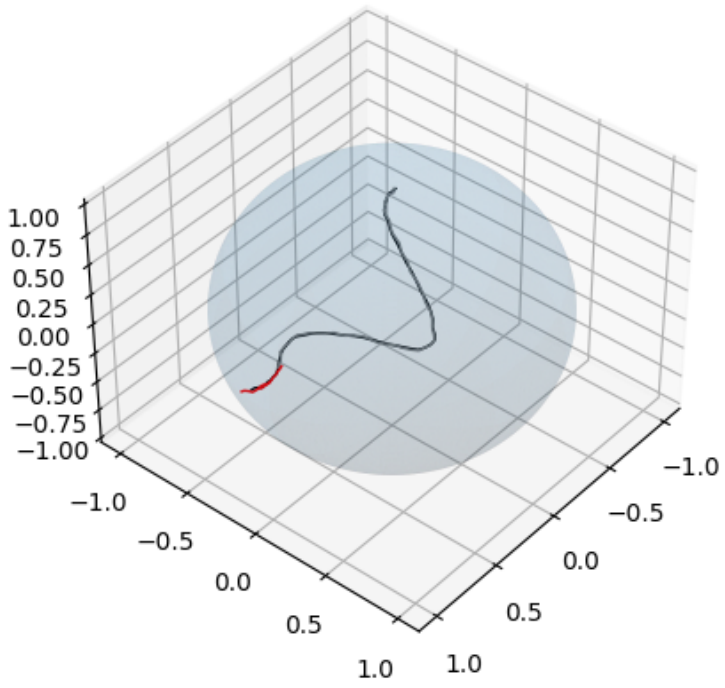


Figure 5.2: The predictive interval of Type 3 prediction in a numerical experiment. The red curves are 95% predictive interval, the black curve is prediction and the gray curve is real data.

5.2.3 Sensitivity of kernel for model with individual mean structure

To compare the sensitivity of our model with various choice of kernels, we use the Type 1 data set where the test data are randomly located in the 8-th simulated curve. Specifically, we generate data using the kernel (2.20), but use different kernels in both estimation and

prediction parts. The kernels are: squared exponential kernel (SE); *piecewise polynomial* kernel (PP) with $q = 2$; Matérn class kernel(MC) with $\nu = 2$ and rational quadratic kernel (RQ). For more details of these kernels, see Section 2.1.4.

The values of rmse and correlation coefficient are calculated between test data and predictions, and average rmse of all replications for the original model (4.1) and approximate model (4.34) are reported in Table 5.4 and Table 5.5 respectively.

From the numerical results, we can see that our model is not sensitive to kernel. Specifically, squared exponential kernel provides the best predictions among the four different kernels, while the performances of the other three kernels are still accurate. However, the root-mean-square-errors of Matérn class kernel in both original model and approximate model are not good. This consequence might suggest us that, in practice, the choice of kernel would not affect the accuracy of prediction or analysis, except Matérn class kernel.

	SE	PP	MC	RQ
rmse	**0.0162**	0.0171	0.0300	0.0168
r	0.9995	0.9996	0.9919	0.9996

Table 5.4: Root-mean-square-error and correlation coefficient for different choices of kernels with individual mean structure and equally spaced data.

	SE	PP	MC	RQ
rmse	**0.0168**	0.0256	0.0256	0.0174
r	0.9996	0.9997	0.9978	0.9996

Table 5.5: Root-mean-square-error and correlation coefficient for different choice of kernel of approximate model with individual mean structure and equally spaced data.

From the tables above, we can see that SE kernel outperforms than others. The reason might be that in the data-generating kernel (2.20) has some common parameters with SE which have greater values while the other parameters have smaller values, which means kernel (2.20) leads to similar covaraince with SE kernel.

5.2.4 Data generation for model with common mean structure on S^2

The simulated data are shown in Figure 5.3 and in the remainder of this subsection we explain how the data were generated.

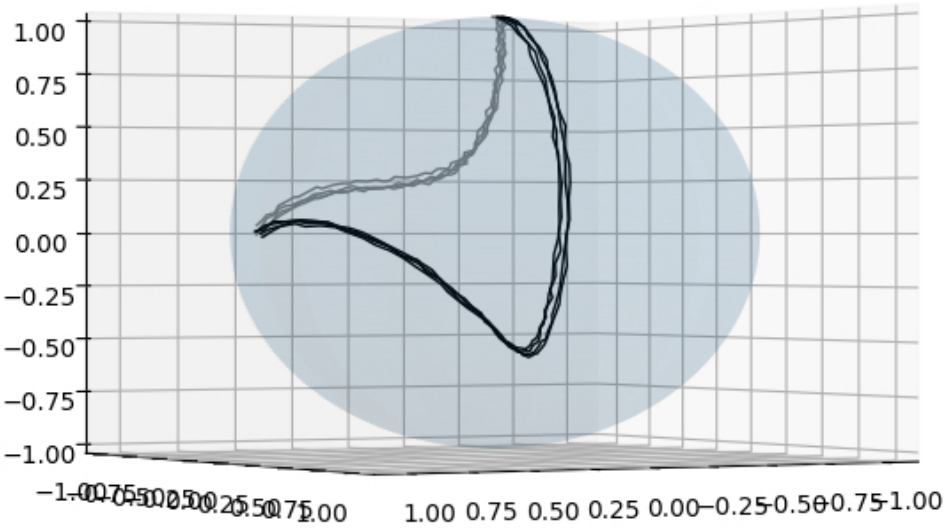


Figure 5.3: Visualization of a data set simulated from model with common mean structure.

As mentioned above, the WGPFR model consists of three parts, the mean structure, the dependent error (covariance structure) and the independent error. We firstly consider how to generate two mean structures based on covariates. The covariates for these two mean structures are taken to be $u_1 = (1, 0)$ and $u_2 = (1, 1)$, respectively. All curves in the two batches start from an identical point which is $p_0 = (0, 0, 1)$. The kernel of the covariance structure is selected as

$$k(t_i, t_j) = v_0 \exp(-\frac{1}{2w_0}(t_i - t_j)^2) + a_0 + a_1 t_i t_j + \sigma^2 \delta_{ij}.$$

For the first element of tangent vector of covariance structure, the hyper-parameters

are $\boldsymbol{\theta}_1 = (0.15, 1, 0.2, 0.01, 0.002)$. Analogously, the hyper-parameters of the second and third element are $\boldsymbol{\theta}_2 = (0.08, 2, 0.17, 0.005, 0.0015)$ and $\boldsymbol{\theta}_3 = (0.11, 1.3, 0.07, 0.015, 0.0025)$ respectively, which are the same as values of the hyper-parameters in the model with individual mean structure. Once such three Gaussian processes with zero mean are determined, we can generate the covariance structure. We also project the tangent vector into the correct tangent space, and the projection is denoted as $tv_3(t)|\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\theta}_3$. The final part is a random error whose distribution is still $\mathcal{N}(0, 0.02)$, we map the random error into the correct tangent space as well. As a consequence, the formula for generating the data is

$$y_{s,m_s}(t) = \text{Exp}(\text{Exp}(\text{Exp}(p_0, (t, (1 + m_s) * \sin(2t), 0)), 0.2 \times tv_3(t)), \epsilon) \tag{5.4}$$

where $s = 1, 2$, $m = 1, ..., M_s$ and $N = 1, ..., N_{m_s}$.

In addition, the functional covariate $\boldsymbol{x}$ is also supposed to be time t which is equally spaced in interval $(0, 1)$ with 60 points. For a given time point t, we can generate a manifold-valued data point using formula (5.4). The same B-spline basis functions in the simulation study 5.2.1 are used here.

5.2.5 Simulation study for model with common mean structure

We randomly select some data points on the 4-th curve in batch two as test data and we use the same way in Section 5.2.2 to form four scenarios. We also replicate the simulation study 100 times which means we have thousands of test data points for each type of prediction. The average rmse of all replications are reported in Table 5.6.

	Type 1	Type 2	Type 3	Type 4
rmse				
WGPFR*	**0.0278**	**0.0147**	**0.0205**	**0.0157**
WGPFR†	0.0325	0.0259	0.0273	0.0252
FLRM	0.0562	0.0636	0.0574	0.0445
WGFmR	0.1488	0.2688	0.5446	0.4072
r				
WGPFR*	0.9990	0.9806	0.9840	0.9812
WGPFR†	0.9871	0.6695	0.6725	0.8777
FLRM	0.9988	0.9846	0.9997	0.9907
WGFmR	0.9727	0.5807	0.2947	0.3422

Table 5.6: Root-mean-square-error and correlation coefficient for several models with four types of data sets with common mean structure and equally spaced data.

The qualitative consequences are similar to result in Table 5.3: WGPFR* and WGPFR† give very precise predictions for both Type 1 prediction and Type 2 prediction, while the performance of FLRM and WGPFmR are relative worse, since FLRM only models the common mean structure and the mean structure of WGPFmR is only the Fréchet mean point which cannot capture the mean trend over time. However, the performance of FLRM in Type 2 prediction is more accurate than WGPFmR (0.0636 compared to 0.2688). The reason is that the 4-th curve in group two is close to the common mean structure of group and FLRM can learn this common mean structure from curves 1,2 and 3 in group two. However, WGPFmR is unable to learn such mean trend from curves 1,2 and 3 in group two.

For the forecasting scenarios (Type 3 prediction and Type 4 prediction), WGPFR* and WGPFR† still provide accurate outputs. Moreover, we can see that the performance of FLRM is quite similar to that of WGPFR* (0.1333 vs 0.1251 and 0.0801 vs 0.5067). It seems that the predictions of WGPFmR fail since prediction of GPR is not good when the test data are distant from the training data and, Fréchet mean point is a poor model of trend of mean structure. Since the test data point of short-term forecasting is closer to the training data than that in long-term forecasting, the predictive error of WGPFmR in Type

4 prediction is relatively better than the result in Type 3 prediction (0.5067 compared to 0.6437).

We also present the prediction for long-term forecasting in Figure 5.4. The figures of other type of prediction are shown in supplementary materials.

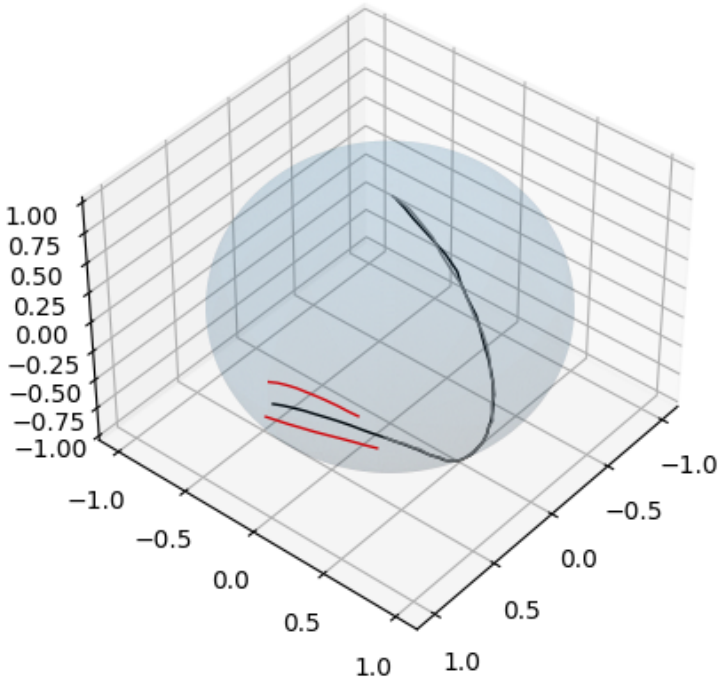


Figure 5.4: The predictive interval of Type 3 prediction. The red curves are 95% predictive interval, the black curve is prediction and the gray curve is real data.

When the number of measurements increases, the predictions should tend to the real data. In other words, if we observe more curves of both batches, the error of predictions would decrease. Thus, we extend the number of curves in each group to 5, 10 and 20 to verify this property. Additionally, we use the same time points, covariates, kernel and hyper-parameters in the covariance function to generate these curves. We compute the root-mean-square-error for each extension and replicate this procedure 20 times. Table 5.7 displays the numerical results.

	Type 1	Type 2	Type 3	Type 4
5 curves	0.0237	0.0235	0.0213	0.0166
10 curves	0.0180	0.0172	0.0183	0.0148
20 curves	**0.0159**	**0.0112**	**0.0173**	**0.0143**

Table 5.7: Root-means-square-error between predictions and real data with different number of curves for original model when the data is equally spaced.

	Type 1	Type 2	Type 3	Type 4
5 curves	0.0238	0.0141	0.0199	0.0165
10 curves	0.0186	0.0080	0.0188	0.0150
20 curves	**0.0153**	**0.0071**	**0.0191**	**0.0107**

Table 5.8: Root-means-square-error between predictions and real data with different number of curves for approximate model when the data is equally spaced.

5.2.6 Sensitivity to kernel for model with common mean structure

We use the same training data, test data and kernel discussed above to investigate the sensitivity of WGPFR model to different kernels. The average root-mean-squared-error and correlation coefficient statistics are reported in Table 5.9 and Table 5.10 for the original model and approximate model respectively.

	SE	PP	MC	RQ
rmse	**0.0281**	0.0286	0.0382	0.0285
r	0.9993	0.9993	0.9983	0.9993

Table 5.9: Root-mean-square-error and correlation coefficient for different choice of kernel for original model with common mean structure and equally spaced data.

	SE	PP	MC	RQ
rmse	0.0358	0.0431	**0.0325**	0.0348
r	0.9874	0.7216	0.9937	0.9843

Table 5.10: Root-mean-square-error and correlation coefficient for different choice of kernel for approximate model with common mean structure and equally spaced data.

From this table, we can see that the performance of WGPFR model is excellent regardless of which kernel we select, which suggests our model is not sensitive to changes in the kernel of wrapped Gaussian process. Similar to Table 5.4 and Table 5.5, Matérm class kernel is found to be relatively inaccurate.

5.2.7 Unequally spaced training data

In the above numerical experiments, we assume the data are equally spaced in a given time interval and we have seen that our model outperforms some other models, such as functional linear regression model on Riemannian manifolds and wrapped Gaussian process Fréchet mean regression model under thus assumption. However, in the real world, the assumption for equally spaced data set may not hold. In order to test the performance of our model under more general scenarios, we set the response variables and functional covariates to be denser in the first half and sparser in the second half. Specifically, we still generate 8 curves on a S^2. Each curve contains 50 data points in the first half and 10 data points in the second half of the corresponding time interval. Thus, the unequally spaced data set is generated via setting 50 data points, in time interval $(0, 0.5)$ and 10 data points in time interval $(0.5, 1)$. The formulas for data generation are the same as (5.3) and (5.4). Without loss of consistency, we use the same kernel and hyper-parameters to generate the data. In order to form the various simulation scenarios, some data points are deleted in the same ways as discussed above, which results in interpolating and extrapolating problems. The simulation studies for the four types of prediction are replicated 100 times respectively. In other words, we also compare the performance about interpolation and extrapolation of various models on the generated dense-sparse data set.

The average root-mean-square-error and correlation coefficient of the model with individual mean structure and unequally spaced data for all four Type predictions are given in Table 5.11. The numerical results of model with common mean structure and unequally spaced data for all four Type predictions are given in Table 5.12.

	Type 1	Type 2	Type 3	Type 4
rmse				
WGPFR*	0.0165	0.0125	0.0193	0.0168
WGPFR†	**0.0157**	**0.0104**	**0.0184**	**0.0146**
FLRM	0.0231	0.0323	0.0301	0.0251
WGFmR	0.1147	0.0509	0.3648	0.3927
r				
WGPFR*	0.9993	0.9937	0.9920	0.9645
WGPFR†	0.9990	0.9918	0.9874	0.9515
FLRM	0.9990	0.9945	0.9987	0.9954
WGFmR	0.9576	.9838	0.1977	0.0929

Table 5.11: Root-mean-square-error and correlation coefficient for several models with four types of data sets with individual mean structure and unequally spaced data.

	Type 1	Type 2	Type 3	Type 4
rmse				
WGPFR*	**0.0266**	0.0542	0.0297	0.0243
WGPFR†	0.0324	**0.0224**	**0.0430**	**0.0352**
FLRM	0.0603	0.0637	0.0441	0.0371
WGFmR	0.1314	0.1123	0.6170	0.6965
r				
WGPFR*	0.9994	0.7295	0.9812	0.9435
WGPFR†	0.9878	0.6894	0.9990	0.9985
FLRM	0.9986	0.9837	0.9812	0.9927
WGFmR	0.9741	0.6952	0.2662	0.2140

Table 5.12: Root-mean-square-error and correlation coefficient for several models with four types of data sets with common mean structure and unequally spaced data.

Since the calculation error of exponential map and inverse exponential map inherent in WGPFR† is less than that of WGPFR*, the approximate wrapped Gaussian process functional regression outperforms other models, while the results of original wrapped Gaussian

process functional regression are still good. In addition, the wrapped Gaussian process Fréchet mean regression model is better in interpolation compared to functional linear regression model in model with individual mean structure. However, in model with common mean structure where the data in the same batch share a common mean structure, the performance of FLRM is relatively better than WGPFmR, since the FLRM can capture the mean trend of different batch. It is not surprising that the performance of the predictions for short-term forecasting are more accurate than that of long-term forecasting.

To sum up, our model is efficient in both equally spaced data set and unequally spaced data set. Moreover, in many scenarios, the prediction of our model is better than some other models. Because S^2 is a special Riemannian manifold which can be embedded into $\mathbb{R}^3$, the approximate wrapped Gaussian process functional regression model is a little better than original wrapped Gaussian process function regression model. This also suggests us that for some special and simple Riemannian manifolds, we may use WGPFR† instead, which is more efficient and effective.

In addition, when the number of repeated measurements increases, the predictions should converge to the real data, which means if we generate more curves for each batch, the predictive error would decrease. Therefore, we also generate 5, 10 and 20 curves of each batch for the original model and approximate model respectively. The numerical results are shown in Table 5.13 and Table 5.14.

	Type 1	Type 2	Type 3	Type 4
5 curves	0.0218	0.0122	0.0254	0.0235
10 curves	0.0149	0.0100	0.0253	0.0210
20 curves	**0.0131**	**0.0097**	**0.0214**	**0.0162**

Table 5.13: Root-means-square-error between predictions and real data with different number of curves when the data is unequally spaced.

	Type 1	Type 2	Type 3	Type 4
5 curves	0.0218	0.0127	0.0251	0.0270
10 curves	0.0149	0.0106	0.0226	0.0266
20 curves	**0.0131**	**0.0099**	**0.0222**	**0.0236**

Table 5.14: Root-means-square-error between predictions and real data with different number of curves of approximate model when the data is unequally spaced.

In the next sections, we present some complex data set (the shape data) and real data set (the flight trajectory data).

5.3 Kendall's shape space

S^2 is a special Riemannian manifold which has many nice properties, such as the existence of the exponential map and inverse exponential map everywhere except the antipole. In this section, we are going to consider a more complex manifold-valued data set and test the performance of our models on that. Shape analysis is one area in which Riemannian manifolds are used in medical image analysis and computer vision. Kendall (1984) and Bookstein et al. (1986) proposed groundbreaking work on modern shape analysis which focus on the geometry of objects after removing scale, rotation and translation. Specifically, the shape of an subject can be represented as a point in a Riemannian manifold. Therefore, we consider our model in Kendall's space shape.

Specifically, we use the wrapped Gaussian process functional regression model to fit and predict the evolution of shapes which is typically batch data on Riemannian manifolds. Based on Kendall's shape space (Kendall, 1984), a 2-dimensional shape can be represented as a point in a *complex projective space*. In other words, an evolution of a shape is a curve in Kendall's shape space and each shape of this evolution is a data point on that curve. For computational reasons, we only generate data for the model with common mean structure (defined in (4.23)) with equally spaced data. We generate 3 curves of each evolution of shape and each curve consists of 10 points (10 shapes), so that each shape is evolved 10 times. In other words, there are 3 curves in every batch and 10 data points in every curve. Each shape in the analysis corresponds to a configuration with 40 landmarks.

Analogously to the various scenarios of numerical experiments of WGPFR on S^2, we also test the performance for interpolation by Type 1 prediction and Type 2 prediction, then test the performance for extrapolation by Type 3 prediction and Type 4 prediction. The details of these different types of prediction are discussed below.

5.3.1 Data generation of shape data for the model with common mean structure

The WGPFR model consists of three parts, the mean structure, the dependent error (covariance structure) and the independent error. We firstly consider generating mean structure based on covariates. The covariates for two batches are $u_1 = (1, 0)$ and $u_2 = (1, 1)$, respectively, as previously. All six curves start from an identical shape p that is a regular 40-gon in $\mathbb{R}^1$ and the functional covariate $\boldsymbol{x}$ is supposed to be time t which is sampled from interval $(0, 1)$. The equation of generating mean structure is

$$\mu_s(t) = \mathrm{Exp}(p, (tv_{s,1}, ..., tv_{s,80})), \ s = 0, 1 \tag{5.5}$$

where $tv_{0,l} = \sin(t_{0,i})^3 \times \sin(l)$, $tv_{1,l} = 2 \times \sin(t_{1,i})^3 \times \sin(l) \times \cos(l)$, $l = 1, ..., 80$ and $t_{m_s,i}$ refers to the i-th time point of group s.

For computational reasons, we use the squared exponential kernel in wrapped Gaussian process which only has 3 hyper-parameters. In addition, we sample tangent vectors from 80 independent Gaussian processes since each configuration has 80 dimensions and then project them into correct tangent spaces. The projected tangent vector is denoted as $tv_3(t)|\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_{80}$. Therefore, the formula of generating data is

$$y_{m_s}(t) = \mathrm{Exp}(\mathrm{Exp}(\mu_s, tv_3(t)|\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_{80}), \epsilon) \tag{5.6}$$

where ϵ is independent random error in which each element follows a standard normal distribution. A visualization of the generated data is shown in Figure 5.5.

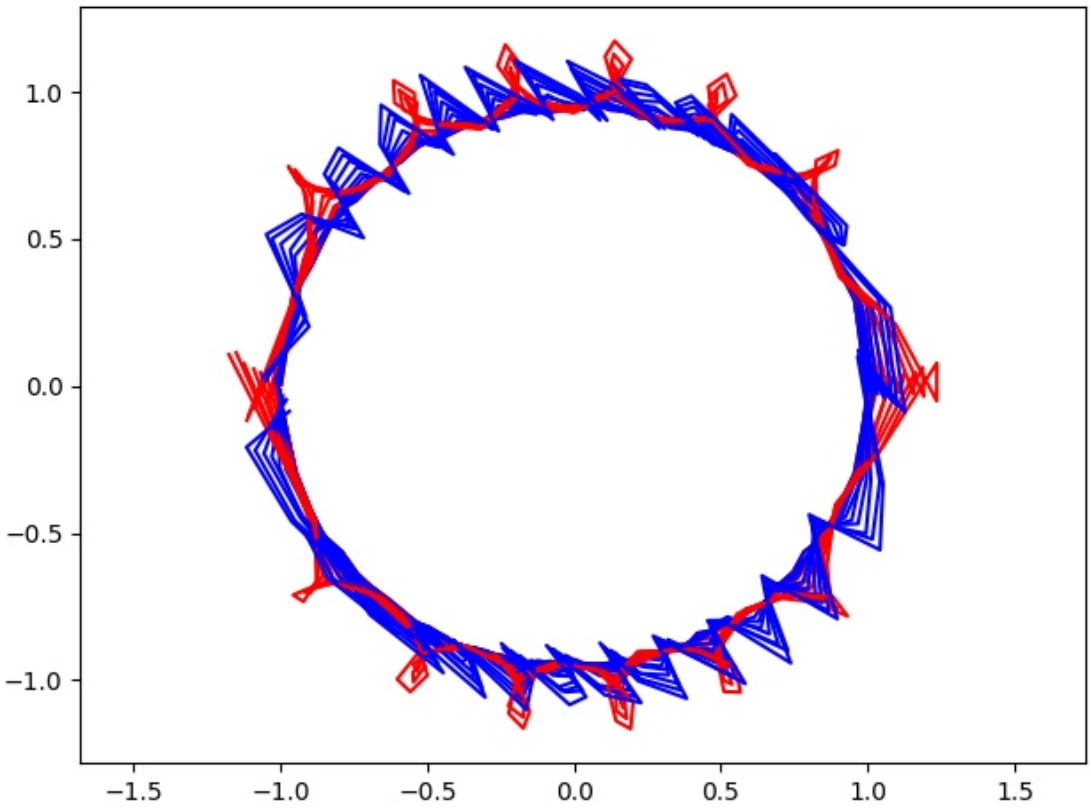


Figure 5.5: Evolution of red shapes is the first batch and evolution of blue shape is the second batch.

5.3.2 Simulation study for the model with common mean structure

We estimate the mean structure and covariance structure as discussed in Section 4.2.1 and Section 4.2.1. The training data consist of all data in batch one together with the first and the second curve in batch two whereas the difference between these types of prediction is the handling of the third curve in batch two. Specifically, for Type 1 prediction, 8 points in Kendall's shape space are randomly selected as part of training data and the remaining 2 points are used as the test data; for Type 2 prediction, 3 data points in a contiguous block are supposed to be the test data, thus the remaining are part of training data; with regard to Type 3 prediction, we add the first half into the training data and the remaining for the test data; as for the Type 4 prediction, the last 3 points are considered as the test data and the remaining are added into the training data. We can see that Type 3 prediction focus the a long-term of stochastic process (the evolution of a shape) while

Type 4 prediction focus on a short-term. Thus we call them as long-term forecasting and short-term forecasting, respectively.

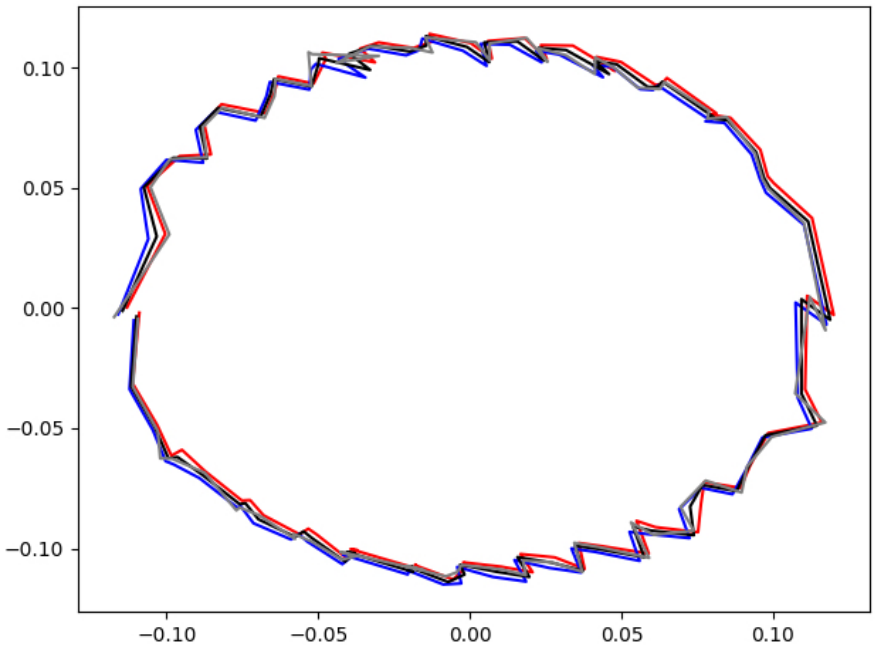
Each numerical experiment is repeated for 100 times. In addition, to compare the performances among different regression models, we also test wrapped Gaussian process functional regression model, approximate wrapped Gaussian process functional regression model, functional linear regression model on Riemannian manifold (the mean structure) and wrapped Gaussian process Fréchet mean regression model.

The root-mean-squared-error and correlation coefficient are given in Table 5.15. WGPFR model and its approximation achieve the best prediction results in not only interpolation but also extrapolation. However, we can see that Type 4 prediction of FLR and WGPFR† are worse than that of Type 3 prediction, which are the opposite of the examples in Section 5.2.2 and Section 5.2.5.

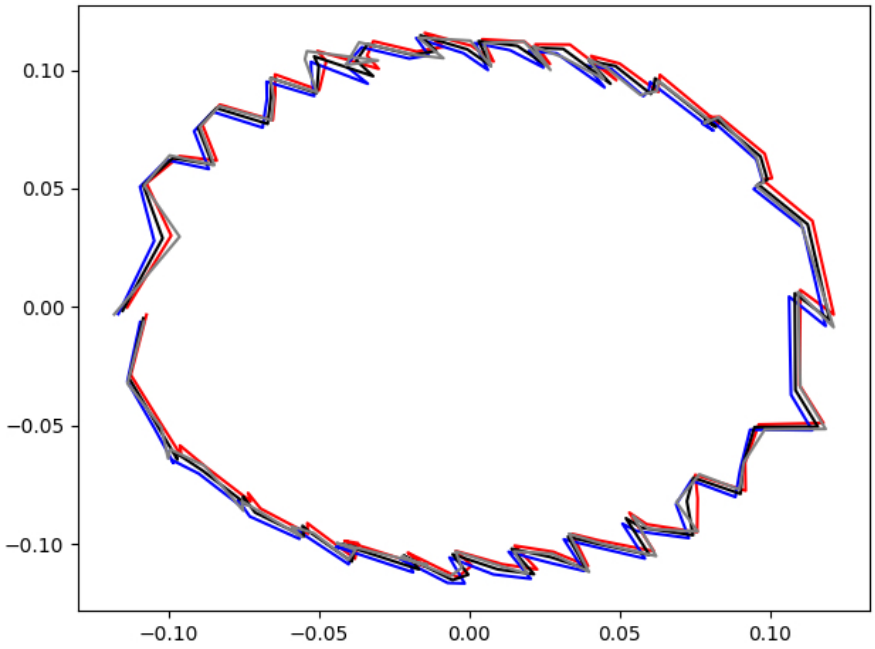
	Type 1	Type 2	Type 3	Type 4
rmse				
WGPFR*	0.0178	0.0175	0.0232	**0.0217**
WGPFR†	**0.0153**	**0.0128**	**0.0080**	0.0267
FLR	0.0435	0.0430	0.0453	0.0465
WGFmR	1.5016	1.5080	1.3732	1.2814
r				
WGPFR*	0.7775	0.8041	0.8468	0.7056
WGPFR†	0.8038	0.8881	0.1329	0.6658
FLR	0.8250	0.6931	0.8311	0.7000
WGFmR	0.0106	-0.0012	0.0163	-0.1456

Table 5.15: Root-mean-squared-error and correlation coefficient of several models with four types of shape data sets.

Some predicted shapes of long-term forecasting are presented in Figure 5.6a and Figure 5.6b.



(a) 95% prediction interval for a predicted shape where the red shape is upper bound, the blue shape is lower bound, gray shape is real data and the black shape is our prediction.



(b) 95% prediction interval for a predicted shape where the red shape is upper bound, the blue shape is lower bound, gray shape is real data and the black shape is our prediction.

Figure 5.6: Data Visualization for Prediction of Shape Data

When the number of curves in each batch increases, the predictions should converge

to the real data. We verify this property by calculating the root-mean-square-error with different numbers of curves in each batch: 5, 10 and 20 curves. The numerical results are displayed in Table 5.16.

	Type 1	Type 2	Type 3	Type 4
5 curves	0.0138	0.0127	0.0159	0.0157
10 curves	0.0123	0.0099	0.0146	0.0125
20 curves	**0.0107**	**0.0082**	**0.0140**	**0.0113**

Table 5.16: Root-means-square-error between predictions and real data with different numbers of curves.

5.4 Flight trajectory data

In the sections above, we presented the performance of WGPFR and its approximation on various toy data sets on two Riemannian manifolds, S^2 and Kendall's shape space, for different scenarios, such as short-term forecasting and long-term forecasting. Both the original models and approximate models outperform the other models in prediction, although there is no significant difference from predictive errors between these two models.

In this section, we will test our model on a real data set.

Before we train our model, it is necessary to pre-process the raw data. In this step, we firstly set the takeoff time of each flight to be 0 and the landing time to be 1, excluding taxi time. There are hundreds of flight trajectories of British Airlines and Eastern China Airlines, but generally, it is common for curves to have missing data. As a result, 25 trajectories of each company in which the number of observed data points in every single flight is greater than 600 were randomly selected. In order to obtain smooth manifold-valued curves from the data, some kernel smoothing functions with small bandwidth were applied to the longitude and latitude of the training data. For computational reasons, we choose 100 data points of each smoothed trajectory as training data.

To model the mean structure for flight trajectory data, we use company as the batch-specific covariate. In practice, for Easten China Airlines, the covariate is defined as 0; and for British Airlines, the covariate is defined as 1. Estimation of the mean structure and covariance structure were described in Section 4.2.1 and 4.2.1, respectively. The parame-

ters of basis functions in mean structure and hyper-parameters in covariance structure can be updated iteratively, which as proposed in Section 4.2.1. Afterwards, the predictions of WGPFRs model has been compared to FLRM and WGFmR for the same flight trajectory data.

The overall prediction of WGPFR model should be better than the other models, since FLRM model only learns from mean structure and ignores the dependent error. Meanwhile, the mean function of WGFmR is a point on S^2 and the prediction should have significant error when data are far away from the Fréchet mean. This is verified by numerical results in Table 5.18.

We select another flight trajectory of British Airlines which satisfies the number of observations. After the same pre-processing steps, 60 data points are added into the training data and the remaining 40 data points are used as the test data. In order to test the performance of interpolation and extrapolation for the real data set, we form the Type 1 prediction by randomly choosing these 60 data points and form the forecasting by selecting the first 60 points in the trajectory as training data. In addition, we test the capability for short-term forecasting and long-term forecasting by supposing different test data. Specifically, the 60-th to 70-th points and the 60-th to final data points of the flight trajectory can be selected respectively to form two scenarios of prediction which are denoted as short-term and long-term, respectively. The training data and test data on this curve are shown in Table 5.17 of different scenarios for prediction.

The performance of WGPFR and approximate WGPFR are compared to FLR and WGFmR on the flight trajectory data. In Table 5.18, the root-mean-squared-error of 20 repeated numerical experiments is displayed. We can see that WGPFR model outperforms FLR and WGFmR for interpolation. For the long-term prediction, the rmse is much smaller than that of the short-term prediction. As mentioned previously, the reason is that GPR fails to predict well when the test data are distant from the training data. The prediction of WGFmR is less accurate since the mean structure (Fréchet mean) is only a manifold-valued point and the test data set are not close to that point. However, FLR provides the best predictions in both short-term and long-term forecasting. The reason is that flight trajectories are smoothed and then mean structure (FLR) might be enough to such manifold-valued stochastic process. Furthermore, if we still consider a covariance

structure, it may enlarge the predictive error.

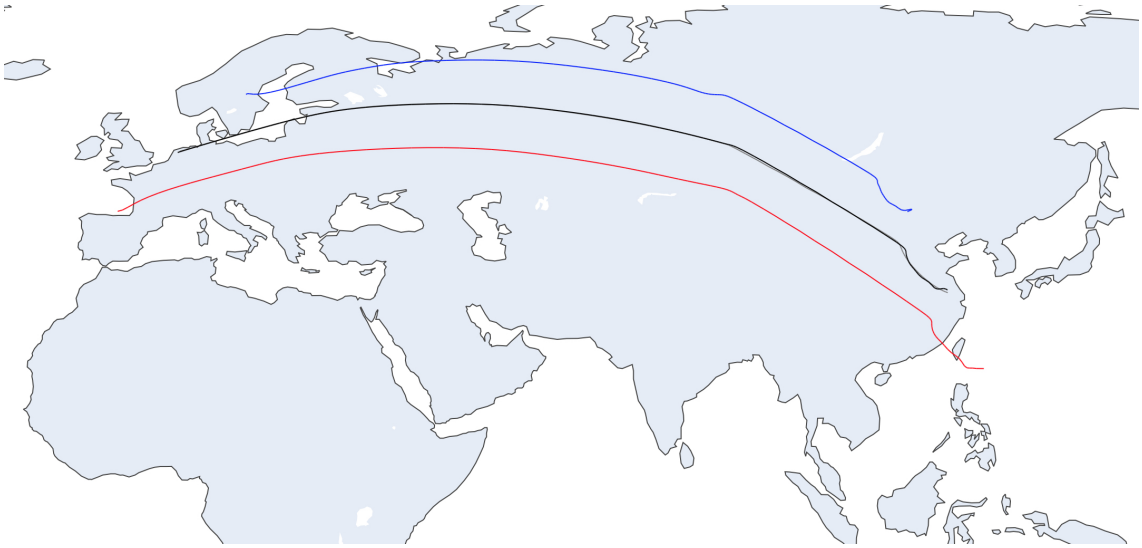
	training data	test data
Type 1	60 random data points	remaining 40 data points
Short-term	$1 \sim 60$ data points	$60 \sim 70$ data points
Long-term	$1 \sim 60$ data points	$60 \sim 100$ data points

Table 5.17: The training data and the test data for different scenarios of prediction.

	Type 1	Short-term	Long-term
WGPFR*	0.0050	0.0126	0.0492
WGPFR†	**0.0049**	0.0133	0.0521
FLR	0.0066	**0.0092**	**0.0083**
WGFmR	0.0187	0.0162	0.3569

Table 5.18: Root-mean-squared-error of several models for flight trajectory data.

We also visualize the results in Figure 5.7a for Type 1 prediction and Figure 5.7b for Type 3 prediction. In these figures, we can see that the start point and end point of flight routes are not in Shanghai and London, respectively. The reason is that in there is no information about taking off time and landing time in the raw data set, thus we delete 100 data points at the beginning and 100 time points in the end.



(a) 95% prediction interval for a flight in which the red curve is upper bound, the blue curve is lower bound, gray curve is real data and the black curve is our prediction.

(b) 95% prediction interval for a flight in which the red curve is upper bound, the blue curve is lower bound, gray curve is real data and the black curve is our prediction.

Figure 5.7: Data visualization for prediction of flight trajectory data.

Chapter 6

Conclusion and Future Work

The main contribution of this thesis is to develop several regression models based on Gaussian process for non-Gaussian distributed data, in particular for data with a truncated Gaussian distribution, a Gamma distribution and manifold-valued data.

In Chapter 3, we defined a truncated Gaussian process and considered a truncated Gaussian process regression model which used a Gibbs sampler for sampling multivariate truncated Gaussian data from a multidimensional Gaussian distribution. As a consequence, the predictions are located on the prescribed domain, which means we successfully constrain the range of prediction. However, the expectation of the predictive distribution has a strong relationship with the choice of upper boundary and lower boundary. We suggest researchers select the boundaries carefully and some prior knowledge might be helpful. Future work could involve the selection of suitable boundaries based on the combination of training data and prior knowledge. Moreover, we could suppose the lower boundary and upper boundary are unknown functions which can be estimated or approximated by data. We also introduce an extension of the truncated Gaussian distribution which includes multi-boundaries, with its probability density function, cumulative distribution function, expectation and variance.

Gamma-distributed data are another example of data which frequently crop up in real world application but which are non-Gaussian. We propose a regression model for Gamma-distributed data by applying Gaussian process as a latent process. The hyper-parameters in the covariance function can be estimated via a Laplace method. Nonetheless, due to the

expensive computation and complicated gradients, it is more reasonable and suitable to use a Gaussian approximation method to represent the likelihood function, which provides a way to calculate the hyper-parameters through a differential evolution algorithm. The probability density functions of Gamma distribution and Gaussian distribution provides an analytical form for prediction. This closed form is very cheap for computation but a little inaccurate, since the Gaussian approximation neglects the skewness of the true distribution. In order to improve the performance of our model, approximate Bayesian computation is used which draws samples from the posterior Gaussian approximation. This trick accelerates convergence since the initial values are not generalised randomly. For most numerical experiments in both univariate dependent variables and multivariate dependent variables, approximate Bayesian computation outperforms the analytical solution. Nevertheless, as mentioned in Section 3.3.6, as k in Gamma distribution increases, the performance of out model becomes worse. This phenomenon needs further exploration.

For data with more complex structure, especially the manifold-valued response variables and Euclidean-valued predictor variables, we introduce a novel probabilistic regression model for functional batch data on smooth Riemannian manifolds in which a mean structure and a covariance structure are estimated simultaneously. Specifically, the mean structure learns from the batch-specific covariates via a functional linear regression and the covariance structure learns from other functional or scalar covariates based on a wrapped Gaussian process. This framework also provides a method to model multi-dimensional functional covariates. In order to improve the predictions of our model, we propose an iterative algorithm to update mean structure and covariance structure based on a variational gradient descent algorithm. In practice, people often analyse data on some special Riemannian manifolds, such as the sphere and Kendall's shape space. Therefore, we can consider tangent vectors in the tangent spaces along the intrinsic Fréchet population mean function $\mu_0(t)$, which is more efficient in computation while the numerical results are significantly close compared to the original model. A variety of empirical results show an improvement of performance of our models (the original model and the approximate model) compared to another existing model (the wrapped Gaussian process regression model with Fréchet mean). Our model could be tested on more real data sets, such as data from medical image analysis. Furthermore, in the future, we could consider this model under more general

situations, especially for non-complete geodesically Riemannian manifolds or non-smooth spaces. Another direction of future work is that we could consider tangent vectors which are dependent via a convolved Gaussian process.

Bibliography

Abraham, C., Cornillon, P.-A., Matzner-Løber, E., and Molinari, N. (2003). Unsupervised curve clustering using b-splines. *Scandinavian journal of statistics*, 30(3):581–595.

Abrahamsen, P. (1997). A review of gaussian random fields and correlation functions. *Norsk Regnesentral/Norwegian Computing Center*, page 64.

Adler, R. J. and Taylor, J. E. (2009). *Random fields and geometry*. Springer Science & Business Media.

Ahrendt, P. (2005). The multivariate gaussian probability distribution. *Technical University of Denmark, Tech. Rep*, page 203.

Aksoy, H. (2000). Use of gamma distribution in hydrological analysis. *Turkish Journal of Engineering and Environmental Sciences*, 24(6):419–428.

Andrieu, C., De Freitas, N., Doucet, A., and Jordan, M. I. (2003). An introduction to mcmc for machine learning. *Machine learning*, 50(1):5–43.

Banerjee, M., Chakraborty, R., Ofori, E., Okun, M. S., Viallancourt, D. E., and Vemuri, B. C. (2016). A nonlinear regression technique for manifold valued data with applications to medical image analysis. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Banerjee, M., Chakraborty, R., Ofori, E., Vaillancourt, D., and Vemuri, B. C. (2015). Nonlinear regression on riemannian manifolds and its applications to neuro-image analysis. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 719–727. Springer.

Belikov, A. V. (2017). The number of key carcinogenic events can be predicted from cancer incidence. *Scientific reports*, 7(1):1–8.

Bhattacharya, A. and Dunson, D. B. (2010). Nonparametric bayesian density estimation on manifolds with applications to planar shapes. *Biometrika*, 97(4):851–865.

Bhattacharya, R., Patrangenaru, V., et al. (2003). Large sample theory of intrinsic and extrinsic sample means on manifolds. *Annals of statistics*, 31(1):1–29.

Boland, P. J. (2007). *Statistical and probabilistic methods in actuarial science*. CRC Press.

Bolstad, W. M. and Curran, J. M. (2016). *Introduction to Bayesian statistics*. John Wiley & Sons.

Bookstein, F. L. et al. (1986). Size and shape spaces for landmark data in two dimensions. *Statistical science*, 1(2):181–222.

Botev, Z. I. (2017). The normal law under linear restrictions: simulation and estimation via minimax tilting. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 79(1):125–148.

Boyle, P. and Frean, M. (2005a). Dependent gaussian processes. In *Advances in neural information processing systems*, pages 217–224.

Boyle, P. and Frean, M. (2005b). Multiple output gaussian process regression.

Brigger, P., Hoeg, J., and Unser, M. (2000). B-spline snakes: a flexible tool for parametric contour detection. *IEEE Transactions on image processing*, 9(9):1484–1496.

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., and Vandergheynst, P. (2017). Geometric deep learning: Going beyond euclidean data. *IEEE Signal Processing Magazine*, 34:18–42.

Calandra, R., Peters, J., Rasmussen, C. E., and Deisenroth, M. P. (2016). Manifold gaussian processes for regression. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 3338–3345. IEEE.

Casella, G. and George, E. I. (1992). Explaining the gibbs sampler. *The American Statistician*, 46(3):167–174.

Chakraborty, R., Vemuri, B. C., et al. (2019). Statistics on the stiefel manifold: theory and applications. *The Annals of Statistics*, 47(1):415–438.

Chavel, I. (2006). *Riemannian geometry: a modern introduction*, volume 98. Cambridge university press.

Chen, D. and Müller, H.-G. (2012). Nonlinear manifold representations for functional data. *The Annals of Statistics*, 40(1):1–29.

Chib, S. (1995). Marginal likelihood from the gibbs output. *Journal of the american statistical association*, 90(432):1313–1321.

Cornea, E., Zhu, H., Kim, P., and Ibrahim, J. G. (2017). Regression models on riemannian symmetric spaces. *Journal of the Royal Statistical Society. Series B, Statistical methodology*, 79(2):463.

Dai, X., Müller, H.-G., et al. (2018). Principal component analysis for functional data on riemannian manifolds and spheres. *The Annals of Statistics*, 46(6B):3334–3361.

Diggle, P. J., Ribeiro, P. J., and Christensen, O. F. (2003). An introduction to model-based geostatistics. In *Spatial statistics and computational methods*, pages 43–86. Springer.

Do Carmo, M. (1992). Mathematics: Theory & applications. *Riemannian Geometry*, pages xiv+–300.

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

Fletcher, P. T. (2013). Geodesic regression and the theory of least squares on riemannian manifolds. *International journal of computer vision*, 105(2):171–185.

Fletcher, P. T., Lu, C., Pizer, S. M., and Joshi, S. (2004). Principal geodesic analysis for the study of nonlinear statistics of shape. *IEEE transactions on medical imaging*, 23(8):995–1005.

Fréchet, M. (1948). Les éléments aléatoires de nature quelconque dans un espace distancié. In *Annales de l'institut Henri Poincaré*, volume 10, pages 215–310.

Friedman, N., Cai, L., and Xie, X. S. (2006). Linking stochastic dynamics to population distribution: an analytical framework of gene expression. *Physical review letters*, 97(16):168302.

Galway, L. (1992). Spline models for observational data.

Geweke, J. (1991). Efficient simulation from the multivariate normal and student-t distributions subject to linear constraints and the evaluation of constraint probabilities. In *Computing science and statistics: Proceedings of the 23rd symposium on the interface*, volume 571, page 578. Fairfax, Virginia: Interface Foundation of North America, Inc.

Geweke, J. (2005). *Contemporary Bayesian econometrics and statistics*, volume 537. John Wiley & Sons.

Greene, W. H. (2003). *Econometric analysis*. Pearson Education India.

Guhaniyogi, R. and Dunson, D. B. (2016). Compressed gaussian process for manifold regression. *The Journal of Machine Learning Research*, 17(1):2472–2497.

Hans, C. (2009). Bayesian lasso regression. *Biometrika*, 96(4):835–845.

Higdon, D. (2002). Space and space-time modeling using process convolutions. In *Quantitative methods for current environmental issues*, pages 37–56. Springer.

Hinkle, J., Muralidharan, P., Fletcher, P. T., and Joshi, S. (2012). Polynomial regression on riemannian manifolds. In *European Conference on Computer Vision*, pages 1–14. Springer.

Horrace, W. C. (2005). Some results on the multivariate truncated normal distribution. *Journal of multivariate analysis*, 94(1):209–221.

Huckemann, S., Hotz, T., and Munk, A. (2010). Intrinsic shape analysis: Geodesic pca for riemannian manifolds modulo isometric lie group actions. *Statistica Sinica*, pages 1–58.

Hurn, S. and Becker, R. (2004). *Contemporary Issues in Economics and Econometrics: Theory and Application*. Edward Elgar Publishing.

Jayasumana, S., Hartley, R., Salzmann, M., Li, H., and Harandi, M. (2013). Kernel methods on the riemannian manifold of symmetric positive definite matrices. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 73–80.

Johnson, N. L., Kotz, S., and Balakrishnan, N. (1995). *Continuous univariate distributions, volume 2*, volume 289. John wiley & sons.

Jung, S., Dryden, I. L., and Marron, J. (2012). Analysis of principal nested spheres. *Biometrika*, 99(3):551–568.

Karcher, H. (1977). Riemannian center of mass and mollifier smoothing. *Communications on pure and applied mathematics*, 30(5):509–541.

Kendall, D. G. (1984). Shape manifolds, procrustean metrics, and complex projective spaces. *Bulletin of the London mathematical society*, 16(2):81–121.

Kendall, W. S. (1990). Probability, convexity, and harmonic maps with small image i: uniqueness and fine existence. *Proceedings of the London Mathematical Society*, 3(2):371–406.

Kim, H. J., Adluru, N., Collins, M. D., Chung, M. K., Bendlin, B. B., Johnson, S. C., Davidson, R. J., and Singh, V. (2014). Multivariate general linear models (mglm) on riemannian manifolds with applications to statistical analysis of diffusion weighted images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2705–2712.

Knight, J. L., Satchell, S., et al. (2001). *Return distributions in finance*. Elsevier.

Kotecha, J. H. and Djuric, P. M. (1999). Gibbs sampling approach for generation of truncated multivariate gaussian random variables. In *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No. 99CH36258)*, volume 3, pages 1757–1760. IEEE.

Lawrence, N. and Hyvärinen, A. (2005). Probabilistic non-linear principal component analysis with gaussian process latent variable models. *Journal of machine learning research*, 6(11).

Le, H. (2001). Locating fréchet means with application to shape spaces. *Advances in Applied Probability*, 33(2):324–338.

Le Brigant, A. and Puechmorel, S. (2019). Quantization and clustering on riemannian manifolds with an application to air traffic analysis. *Journal of Multivariate Analysis*, 173:685–703.

Lee, S.-Y. (2007). *Structural equation modeling: A Bayesian approach*, volume 711. John Wiley & Sons.

Lila, E., Aston, J. A., Sangalli, L. M., et al. (2016). Smooth principal component analysis over two-dimensional manifolds with an application to neuroimaging. *The Annals of Applied Statistics*, 10(4):1854–1879.

Lin, L., Mu, N., Cheung, P., Dunson, D., et al. (2018). Extrinsic gaussian processes for regression and classification on manifolds. *Bayesian Analysis*.

Lin, Z. and Yao, F. (2020). Functional regression on the manifold with contamination. *Biometrika*.

Lin, Z., Yao, F., et al. (2019). Intrinsic riemannian functional data analysis. *The Annals of Statistics*, 47(6):3533–3577.

MacKay, D. and Gibbs, M. (1997). Efficient implementation of gaussian processes. *Neural Computation*.

MacKay, D. J. (1997). Gaussian processes-a replacement for supervised neural networks?

Mallasto, A. and Feragen, A. (2018). Wrapped gaussian process regression on riemannian manifolds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5580–5588.

Mardia, K. V. and Jupp, P. E. (2009). *Directional statistics*, volume 494. John Wiley & Sons.

Marjoram, P., Molitor, J., Plagnol, V., and Tavaré, S. (2003). Markov chain monte carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 100(26):15324–15328.

McLean, M. W., Hooker, G., Staicu, A.-M., Scheipl, F., and Ruppert, D. (2014). Functional generalized additive models. *Journal of Computational and Graphical Statistics*, 23(1):249–269.

Meng, X.-L. and Wong, W. H. (1996). Simulating ratios of normalizing constants via a simple identity: a theoretical exploration. *Statistica Sinica*, pages 831–860.

Montgomery, D. C., Peck, E. A., and Vining, G. G. (2021). *Introduction to linear regression analysis*. John Wiley & Sons.

Moon, K. and Pavlovic, V. (2008). Regression using gaussian process manifold kernel dimensionality reduction. In *2008 IEEE Workshop on Machine Learning for Signal Processing*, pages 14–19. IEEE.

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

Nava-Yazdani, E., Hege, H.-C., Sullivan, T. J., and von Tycowicz, C. (2020). Geodesic analysis in kendall's shape space with epidemiological applications. *Journal of Mathematical Imaging and Vision*, pages 1–11.

Neal, R. M. (1993). *Probabilistic inference using Markov chain Monte Carlo methods*. Department of Computer Science, University of Toronto Toronto, ON, Canada.

Neal, R. M. (2012). *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media.

Niven, I. (1969). Formal power series. *The American Mathematical Monthly*, 76(8):871–889.

Oller, J. and Corcuera, J. M. (1995). Intrinsic analysis of statistical estimation. *The Annals of Statistics*, 23(5):1562–1581.

Pelletier, B. (2006). Non-parametric regression estimation on closed riemannian manifolds. *Journal of Nonparametric Statistics*, 18(1):57–67.

Pennec, X. (1999). Probabilities and statistics on riemannian manifolds: Basic tools for geometric measurements. In *NSIP*, volume 3, pages 194–198. Citeseer.

Pennec, X., Sommer, S., and Fletcher, T. (2019). *Riemannian geometric statistics in medical image analysis*. Academic Press.

Petersen, A. and Müller, H.-G. (2019). Fréchet regression for random objects with euclidean predictors. *The Annals of Statistics*, 47(2):691–719.

Pigoli, D., Menafoglio, A., and Secchi, P. (2016). Kriging prediction for manifold-valued random fields. *Journal of Multivariate Analysis*, 145:117–131.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (2007). *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press.

Price, K., Storn, R. M., and Lampinen, J. A. (2006). *Differential evolution: a practical approach to global optimization*. Springer Science & Business Media.

Rubin, D. B. (1984). Bayesianly justifiable and relevant frequency calculations for the applies statistician. *The Annals of Statistics*, pages 1151–1172.

Rue, H., Martino, S., and Chopin, N. (2009). Approximate bayesian inference for latent gaussian models by using integrated nested laplace approximations. *Journal of the royal statistical society: Series b (statistical methodology)*, 71(2):319–392.

Salkind, N. (2010). *Encyclopedia of Research Design*. Number Vols. 1-0. SAGE Publications.

Serban, N. and Wasserman, L. (2005). Cats: clustering after transformation and smoothing. *Journal of the American Statistical Association*, 100(471):990–999.

Shi, J., Wang, B., Murray-Smith, R., and Titterington, D. (2007). Gaussian process functional regression modeling for batch data. *Biometrics*, 63:714–723.

Shi, J. Q. and Choi, T. (2011). *Gaussian process regression analysis for functional data*. CRC Press.

Shi, J. Q., Murray-Smith, R., and Titterington, D. M. (2005). Hierarchical gaussian process mixtures for regression. *Statistics and computing*, 15(1):31–41.

Shi, J. Q. and Wang, B. (2008). Curve prediction and clustering with mixtures of gaussian process functional regression models. *Statistics and Computing*, 18(3):267–283.

Sisson, S. A., Fan, Y., and Tanaka, M. M. (2007). Sequential monte carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 104(6):1760–1765.

Smola, A. J. and Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and computing*, 14(3):199–222.

Storn, R. and Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341–359.

Su, J., Kurtek, S., Klassen, E., Srivastava, A., et al. (2014). Statistical analysis of trajectories on riemannian manifolds: bird migration, hurricane tracking and video surveillance. *The Annals of Applied Statistics*, 8(1):530–552.

Tavaré, S., Balding, D. J., Griffiths, R. C., and Donnelly, P. (1997). Inferring coalescence times from dna sequence data. *Genetics*, 145(2):505–518.

Wahba, G. (1990). *Spline models for observational data*. SIAM.

Wang, B. and Shi, J. Q. (2014). Generalized gaussian process regression model for non-gaussian functional data. *Journal of the American Statistical Association*, 109(507):1123–1133.

Wang, J.-L., Chiou, J.-M., and Müller, H.-G. (2016). Functional data analysis. *Annual Review of Statistics and Its Application*, 3:257–295.

Wilhelm, S. (2015). Gibbs sampler for the truncated multivariate normal distribution.

Williams, C. K. and Rasmussen, C. E. (2006). *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA.

Yildirim, I. (2012). Bayesian inference: Gibbs sampling. *Technical Note, University of Rochester*.

Zhang, M. and Fletcher, T. (2013). Probabilistic principal geodesic analysis. *Advances in neural information processing systems*, 26.