WARP-U MCMC, STOCHASTIC BRIDGE SAMPLING AND SUPERVISED FUNCTIONAL PRINCIPLE COMPONENT ANALYSIS

A Dissertation

by

FEI DING

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Jianhua Huang
Co-Chair of Committee,	David Jones
Committee Members,	Debdeep Pati
	Darren DePoy
Head of Department,	Brani Vidakovic

May 2021

Major Subject: Statistics

Copyright 2021 Fei Ding

ABSTRACT

This dissertation focuses on Bayesian sampling, Bayesian evidence estimation and supervised functional principle component analysis (PCA) and the corresponding theoretical, computational and application challenges. In Bayesian statistics, strategies for estimating normalizing constants often require samples from the underlying target distribution, but obtaining these samples can be challenging, especially when the target is multi-modal. Some Markov chain Monte Carlo (MCMC) and adaptive importance sampling (AIS) methods are specifically designed to address this challenge, such as parallel tempering. However, tuning these algorithms can be time-consuming, and it is typically unclear which estimation strategy should be applied once the samples are obtained. We propose a new adaptive MCMC method to sample from multi-modal target densities and simultaneously perform much of the computation needed for our complementary normalizing constant estimator. Our approach adapts the bridge sampling estimation techniques and proposes a new version of the Warp-U bridge sampling estimator. An important aspect of our overall method is that it requires minimal tuning and is simpler to apply than many competing techniques. The ergodicity of our sampling algorithm is established. In functional data analysis, incorporating covariates into functional PCA can substantially improve the representation efficiency of the principal components and predictive performance. However, many existing functional PCA methods do not make use of covariates, and those that do often have high computational cost or make overly simplistic assumptions that are violated in practice. We propose a new framework, called Covariate Dependent Functional Principal Component Analysis (CD-FPCA), in which both the mean and covariance structure depend on covariates. We propose a corresponding estimation algorithm, which makes use of spline basis representations and roughness penalties, and is substantially more computationally efficient than competing approaches of adequate estimation and prediction accuracy. A key aspect of this work is our novel approach for modeling the covariance function and ensuring that it is symmetric positive semi-definite. We demonstrate the advantages of our three methods through simulation studies and astronomical data analysis.

DEDICATION

To my family, to whom I will remain forever indebted.

ACKNOWLEDGMENTS

First of all, I would like to express my deep gratitude to my advisor Dr. David Jones who helped me a lot with his knowledge and wisdom. My time in graduate school was enriched by his support. Without his continuous guidance, encouragement, this would not have been possible, and no word of thanks is enough for his wonderful mentorship.

I would also extend my gratitude to Dr. Shiyuan He who gives me sound advice and teach me useful skills. I would also like to thank Dr. Jianhua Huang whose insightful comments have been extremely helpful. Thanks also to Dr. Xiao-Li Meng, whose innovative ideas inspired me a lot. I am also grateful to Dr. Debdeep Pati and Dr. Darren DePoy for giving me helpful advice and meaningful comments.

Finally, I would like to take this formal opportunity to thank my family for their continuous encouragement and unflinching support.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a dissertation committee consisting of Drs. David Jones, Jianhua Huang, Debdeep Pati of the Department of Statistics and Darren DePoy of the Department of Physics and Astronomy.

Funding Sources

In this dissertation, the research work of the student is supported by NSF grants IIS-1900990, and CCF-1956219.

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

Bayesian sampling and normalizing constants estimation, e.g., Bayesian evidence estimation, are two fundamental problems in Bayesian statistics. Bayesian sampling methods are used for Bayesian parameter estimation, while Bayesian evidence is often used for model selection and hypothesis testing. These methods have gained popularity in a wide range of scientific fields and here we particularly highlight their use in astronomy. Another method that is widely used for the analysis of astronomical data is functional data analysis (FDA). Among a variety of functional data approaches, the most fundamental is functional principal component analysis (FPCA). This dissertation proposes a new Markov chain Monte Carlo (MCMC) method for Bayesian sampling, a new version of the Warp-U bridge estimator (Wang et al., 2020) for Bayesian evidence estimation and a new supervised functional PCA framework. These methods are then used to analyze several astronomical datasets.

Normalizing constants play a central role in modern statistical applications, e.g., Bayesian evidences are widely used in many scientific fields. For instance, Nelson et al. (2018) and Pullen and Morris (2014) discussed the computation of Bayesian evidences in the context of exoplanet detection and systems biology, respectively. Given the many scientific uses of normalizing constants, computationally and statistically efficient methods for estimating them are of high practical value, and many powerful algorithms have been introduced. On the other hand, some ubiquitous scenarios remain challenging, such as estimating normalizing constants for multi-modal target densities. Indeed, Nelson et al. (2018) applied numerous strategies for estimating the Bayesian evidence for the presence of an exoplanet orbiting a star and obtained somewhat divergent estimates, even after substantial calibration efforts. We make two main contributions in these fields of Bayesian sampling and Bayesian evidence estimation. Our first contribution is to introduce a new Markov chain Monte Carlo (MCMC) sampling algorithm, called *Warp-U MCMC*, which alternately applies the Warp-U transformation (Wang et al., 2020) and then its *inverse*. Our second contribution is to substantially improve the computational efficiency of the Warp-U bridge estimation strategy (Wang

et al., 2020).

Functional data analysis (FDA) is increasingly important in many scientific fields including astronomy, biology, and neuroscience. It is a powerful tool that can be used to jointly model collections of curves, time series, spatial structures, or other functional observations, and can address difficulties such as sparse and irregularly spaced measurements. The key to FDA is exploiting the widespread presence of underlying smoothness in real data to efficiently model similarities and differences between functional observations, e.g., the similarities and differences between time series capturing the changing brightness of stars of a given type. We propose an FPCA method that incorporates covariates information in a computationally efficient manner which overcomes the computational challenge, while simultaneously allowing both the mean function and the covariance function to depend on the covariates in a non-linear way, as well as permitting unbalanced sampling patterns.

2. WARP-U MCMC AND STOCHASTIC BRIDGE SAMPLING

2.1 Introduction

We consider the following general context: for an unnormalized probability density q with support Θ, we seek to estimate the normalizing constant $c = \int_\Theta q(\boldsymbol{\theta})\,\mathrm{d}\boldsymbol{\theta}$. In statistical approaches to this problem, there are typically two key tasks: (i) obtaining samples $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \cdots, \boldsymbol{\theta}_n$ from $\pi = \frac{1}{c}q$ or some related density (or densities), and (ii) constructing an estimate of c based on the samples. These tasks may be performed sequentially or in combination, depending on the specific strategy. If π is multi-modal, then substantial inefficiencies can result from not addressing this in both the sampling and estimation tasks, (i) and (ii), respectively. On the other hand, addressing multi-modality does not guarantee an efficient estimator of c.

We propose a powerful new algorithm that efficiently samples from a multi-modal target density and then applies a complementary estimation strategy which estimates the target normalizing constant. The estimation step is complementary in the sense that much of the necessary computation is completed during the sampling stage. Both our sampling and estimation steps make use of the stochastic Warp-U transformation proposed by Wang et al. (2020), which transforms multi-modal densities into approximately uni-modal ones. The key conceptual step of constructing a Warp-U transformation is to *induce* a mixture representation of the target $\pi(\boldsymbol{\theta}) = \sum_{k=1}^{K} w_k \pi_k(\boldsymbol{\theta})$, where intuitively π_k can be viewed as a 'component' of the target density, for $k = 1, \ldots, K$. We review further details of the Warp-U transformation in Section 2.2.1.

We make two main contributions that enable us to construct our overall algorithm. Our first contribution is to introduce a new Markov chain Monte Carlo (MCMC) sampling algorithm, called *Warp-U MCMC*, which alternately applies the Warp-U transformation and then its *inverse*. The algorithm thus updates a current draw from the target density by stochastically mapping it first to a uni-modal density draw then back to a (different) target density draw. The key to this approach is that the Warp-U transformation maps each 'component' π_k of the target to almost the same in-

termediate uni-modal density, meaning that the inverse Warp-U transformation maps to a random 'component' of the target density, and therefore easily explores many modes. Our simulation studies, demonstrate that this property makes our algorithm substantially more efficient than competing approaches, such as parallel tempering (Geyer, 1991).

The Warp-U transformation proposed by Wang et al. (2020) depends on a Gaussian mixture approximation to the target density π, but initially there are no target draws with which to estimate the mixture parameters. To address this issue we create an adaptive version of our sampling algorithm which starts by sampling from a uniform density, or any other crude approximation to the target density, and then proceeds in stages, updating the approximating density at each stage. Importantly, as with the Warp-U bridge sampling estimation strategy proposed by Wang et al. (2020), it is not necessary for the approximating density to be a precise representation of the target density; any reasonable approximation is sufficient. This means that the number of stages needed in our algorithm is typically small, e.g., less than ten. We prove that our adaptive MCMC algorithm is ergodic, following the approach of Roberts and Rosenthal (2007).

Our second contribution is to substantially improve the computational efficiency of the Warp-U bridge estimation strategy (Wang et al., 2020), which is the key to our normalizing constant estimation step. Suppose that draws from the target density have been obtained, e.g., using our Warp-U MCMC sampler. Warp-U bridge estimation applies the Warp-U transformation to the target density draws to obtain draws from an approximately uni-modal density $\tilde{\pi}$, and then applies standard bridge sampling estimation using these new draws and $\tilde{q} = c\tilde{\pi}$ (which is easily constructed from q). We prove that a substantial reduction in asymptotic variance can be obtained by instead applying standard bridge sampling to estimate the normalizing constant of each mixture 'component' $q_k = c\pi_k$, for $k = 1, \ldots, K$, and then combining the results to estimate c. Since this estimator is also computationally more efficient, it is superior to that proposed by Wang et al. (2020) (both approaches are asymptotically unbiased). Indeed, in our simulation studies, our normalizing constant estimator has substantially lower root mean squared error (RMSE), given fixed computational resources.

Many algorithms have been developed to sample from multi-modal densities, and there are also a number which simultaneously perform sampling and estimation of normalizing constants. A leading example of the latter technique is the Generalized Wang-Landau (GWL) algorithm proposed by Liang (2005), which is an energy based adaptive importance sampling method. The multi-stage approach used in our adaptive method is loosely based on the GWL algorithm, and earlier adaptive importance sampling strategies such as Liang (2002), Berg and Neuhaus (1991), and Wang and Landau (2001). There have been several extensions to the GWL algorithm, including Liang et al. (2007) and Bornn et al. (2013), but also some concerns about its convergence properties, e.g., Jacob et al. (2014) showed that only some variations reach the so-called flat histogram convergence criterion in finite time, whereas other variations do not. Furthermore, Wang et al. (2020) illustrated that the GWL normalizing constant estimator is sometimes inefficient, and the alternative strategy of applying Warp-U bridge estimation to the GWL draws (after weighted resampling) can substantially reduce RMSE (for fixed computational resources). Indeed, although it is conceptually appealing to combine sampling and estimation in a single step, existing techniques for performing these tasks separately are in some ways more developed. Perhaps the best known general strategy for sampling from multi-modal densities is parallel tempering, see Geyer (1991). On the other hand, parallel tempering does not perform any estimation, and therefore must be combined with a separate estimation strategy, e.g., bridge sampling. In our simulation studies, we find that our proposed strategy is computationally more efficient than parallel tempering for two reasons: firstly, more of the computation for the estimation step is performed by our Warp-U MCMC sampler than by parallel tempering, and secondly, our approach has only one chain and always accepts inter-mode proposals. Of course, we do not suggest that our Warp-U sampler is more efficient in all cases, but the evidence presented here does support the view that it may be useful in many practical cases.

Even when parallel tempering or another sampler is preferred to our Warp-U MCMC sampler, our *stochastic Warp-U bridge estimation* approach is still useful for the normalizing constant estimation step. Bridge sampling estimation was first proposed by Bennett (1976) and was later further

developed by Meng and Wong (1996). A number of other popular normalizing constant estimation methods are special cases of bridge sampling estimation, e.g., Mira and Nicholls (2004) showed that the multi-block Metropolis-Hastings estimators proposed by Chib and Jeliazkov (2001) are bridge sampling estimators. Transformations can be used to improve bridge sampling and other normalizing constant estimation strategies, e.g., Meng and Schilling (2002) proposed transformations for uni-modal target densities, and Wang et al. (2020) extended these ideas to multi-modal densities. Our approach is related to these methods, but differs in that we first split the target density into 'components' and then compute separate bridge sampling estimators for each, before finally combining the estimators.

A further advantage of our overall approach compared to many other methods is that it is generally simpler to implement, because the key tool is mixture distributions which are intuitive and straightforward to fit. A limitation is that the number of components in the mixture distribution has to be chosen (or inferred). However, this is conceptually and practically simpler than adjusting the tuning parameters in many other algorithms, e.g., the number and values of different temperature levels in parallel tempering. Indeed, we find that our approach is generally quite robust to the number of components, especially if the number is larger than the number of modes. Thus, with adequate computational resources, one can always avoid tuning by setting the number of components to be high initially. A second limitation of our adaptive Warp-U MCMC sampler is that it is likely inefficient in the case of very isolated modes, partly because it may take time to identify the modes, and partly because its advantages over a Metropolis-Hastings (MH) algorithm with a mixture density proposal rely on overlap between modes, and is diminished in the case of isolated modes. On the other hand, sampling from isolated modes with unknown locations is universally challenging.

The chapter related to Warp-U MCMC and stochastic bridge estimation is organized as follows. Section 2.2 briefly reviews bridge sampling estimation and Warp-U transformations before introducing our adaptive Warp-U MCMC sampler and the complementary stochastic bridge estimation method. Section 2.3 provides theoretical justification for our approach in the form of theroems

establishing the ergodicity of our sampler and the greater efficiency of stochastic bridge estimation compared with Warp-U bridge estimation. Section 2.4 provides several numerical studies demonstrating the utility of our method. Section 2.5 applies our approach to estimating the Bayesian evidence for exoplanets based on radial velocity (RV) datasets. Discussion and open problems are found in Section 2.6.

2.2 Warp-U Sampling and Estimation Methods

2.2.1 Bridge Sampling Estimation and Warp-U Transformations

Let q_1 and q_2 denote unnormalized densities with unknown normalizing constants c_1 and c_2, respectively, and for simplicity assume that both have support $\Theta = \mathbb{R}^d$. Furthermore, suppose that we are interested in estimating the ratio $r = c_1/c_2$, e.g., a Bayes factor. For this scenario, Bennett (1976) and Meng and Wong (1996) introduced bridge sampling estimation which is a strategy for estimating r *given* draws from the normalized densities $p_i = q_i/c_i$, for $i = 1, 2$. Their method is based on the key identity

$$r = \frac{c_1}{c_2} = \frac{E_{p_2}[q_1(\boldsymbol{\theta})\alpha(\boldsymbol{\theta})]}{E_{p_1}[q_2(\boldsymbol{\theta})\alpha(\boldsymbol{\theta})]}, \tag{2.1}$$

where α is the 'bridge' function (discussed below), and E_{p_i} denotes an expectation with respect to p_i, for $i = 1, 2$. The bridge sampling estimator of r is the sample counterpart of (2.1), i.e.,

$$\hat{r} = \frac{n_2^{-1} \sum_{j=1}^{n_2} q_1(\boldsymbol{\theta}_{2,j})\alpha(\boldsymbol{\theta}_{2,j})}{n_1^{-1} \sum_{j=1}^{n_1} q_2(\boldsymbol{\theta}_{1,j})\alpha(\boldsymbol{\theta}_{1,j})}, \tag{2.2}$$

where $\boldsymbol{\theta}_{i,1}, \ldots, \boldsymbol{\theta}_{i,n_i} \overset{iid}{\sim} p_i$, and hence n_i denotes the number of samples from q_i, for $i = 1, 2$. If the samples are obtained via MCMC then they will not usually be independent, but fortunately independence is not crucial in practice and is only assumed for the purposes of asymptotic theory. Under independence, Meng and Wong (1996) showed that the optimal α is $\alpha_{\text{opt}}(\boldsymbol{\theta}) \propto \frac{1}{s_1 q_1(\boldsymbol{\theta}) + r s_2 q_2(\boldsymbol{\theta})}$, where $s_i = \frac{n_i}{n_1+n_2}, i = 1, 2$, in the sense that this choice yields the smallest asymptotic variance for the estimator $\hat{r}$. The unknown r appears in α_{opt}, but Meng and Wong (1996) addressed this difficulty by introducing an iterative scheme which alternates between applying (2.2) (with $\alpha = \alpha_{\text{opt}}$)

and plugging the resulting $\hat{r}$ into α_{opt} until convergence.

In many cases, and in the current dissertation, there is only one unnormalized density $q = c\pi$, and hence only one normalizing constant c that needs to be estimated. In this scenario, we set $q_1 = q$ in (2.2), and choose $q_2 = p_2$ to be a normalized density (i.e., $c_2 = 1$), such as a Gaussian distribution. The choice of p_2 is important because the asymptotic variance of $\hat{r}$ decreases as the separation between $p_1 = \pi$ and p_2 decreases, where the separation divergence is defined as

$$H_A(p_1, p_2) = 1 - \int_{\Theta_1 \cap \Theta_2} [\eta_1 p_1^{-1}(\boldsymbol{\theta}) + \eta_2 p_2^{-1}(\boldsymbol{\theta})]^{-1} \, \mathrm{d}\boldsymbol{\theta}, \tag{2.3}$$

where $\eta_i = \frac{s_i^{-1}}{s_1^{-1}+s_2^{-1}}$, $s_i = \frac{n_i}{n_1+n_2}$, Θ_i is the support of p_i, for $i = 1, 2$, and H_A is known as the sample-size adjusted harmonic divergence. For efficiency, it is also important that p_2 can be evaluated and sampled from with minimal computational cost.

Given the above considerations, if $\pi = q/c$ is multi-modal, it is natural to choose p_2 to be a Gaussian mixture distribution approximating π, i.e.,

$$\phi_{\text{mix}}(\boldsymbol{\theta}) = \sum_{k=1}^{K} \phi^{(k)}(\boldsymbol{\theta}) = \sum_{k=1}^{K} w_k |\mathcal{S}_k^{-1}| \phi(\mathcal{S}_k^{-1}(\boldsymbol{\theta} - \boldsymbol{\mu}_k)), \tag{2.4}$$

where w_k, $\boldsymbol{\mu}_k$, and $\mathcal{S}_k$, denote a weight, mean vector, and scale matrix, respectively, for $k = 1, \ldots, K$, and ϕ is the density function of $\mathcal{N}_d(\mathbf{0}, \mathbf{I})$, i.e., the standard d-dimensional multivariate Gaussian distribution. In (2.4) and throughout, we use $\phi^{(k)}$ to denote the k-th component of ϕ_{mix} including its weight w_k. Standard bridge sampling estimation would proceed by applying (2.2) with densities q (unnormalized) and ϕ_{mix}. However, Wang et al. (2020) proposed an improved approach called Warp-U bridge estimation based on the idea of warp bridge sampling estimation introduced by Meng and Schilling (2002), as we now explain.

By the properties of f-divergences (Ali and Silvey, 1966), for any transformation $\mathcal{F}$ we have $H_A(p_1, p_2) \geq H_A(\mathcal{F}(p_1), \mathcal{F}(p_2))$. Since the asymptotic variance of $\hat{r}$ decreases with the divergence (2.3), a transformation of the initial densities will generally decrease the asymptotic variance, although care must be taken to avoid transformations which alter the underlying normalizing

constants. In many cases, a good choice of $\mathcal{F}$ can provide substantial efficiency gains. In the multi-modal scenario, Wang et al. (2020) proposed a *stochastic* transformation that maps multi-modal densities to ones which are approximately uni-modal. Specifically, their Warp-U stochastic transformation makes use of the approximation ϕ_{mix} and is given by $\boldsymbol{\theta}^* = \mathcal{F}_\psi(\boldsymbol{\theta}) = \mathcal{S}_\psi^{-1}(\boldsymbol{\theta} - \boldsymbol{\mu}_\psi)$, where the random component index ψ is drawn from the conditional distribution under $\boldsymbol{\theta} \sim \phi_{\text{mix}}$, i.e.,

$$\varpi(\psi|\boldsymbol{\theta}) = \frac{\phi^{(\psi)}(\boldsymbol{\theta})}{\phi_{\text{mix}}(\boldsymbol{\theta})}. \tag{2.5}$$

The Warp-U transformation is based on a coupling argument whereby we *impose* the conditional distribution (2.5) when $\boldsymbol{\theta} \sim \pi$, so that the joint density function of $(\psi, \boldsymbol{\theta})$ is $p(\psi, \boldsymbol{\theta}) = \varpi(\psi|\boldsymbol{\theta})\pi(\boldsymbol{\theta})$. Applying (2.5) to ϕ_{mix} gives the standard Gaussian density ϕ, and since ϕ_{mix} approximates p, it is intuitive that applying (2.5) to p also gives an approximately uni-modal density. The analogy between ϕ_{mix} and q (or equivalently π) is further demonstrated by noting that $q^{(k)}(\boldsymbol{\theta}) = cw_k\pi_k = \varpi(k|\boldsymbol{\theta})q(\boldsymbol{\theta})$, viewed as a function of $\boldsymbol{\theta}$ only, can be interpreted as the k-th 'component' of q induced by the conditional distribution (2.5). Here π_k denotes the k-th *induced* component of π, i.e., $\varpi(k|\boldsymbol{\theta})\pi(\boldsymbol{\theta})/w_k$. See Wang et al. (2020) for full details and graphical illustrations. The transformed version of q is given by

$$\tilde{q}(\boldsymbol{\theta}^*) = \sum_{k=1}^{K} \phi(\boldsymbol{\theta}^*) \frac{q(S_k\boldsymbol{\theta}^* + \boldsymbol{\mu}_k)}{\phi_{\text{mix}}(S_k\boldsymbol{\theta}^* + \boldsymbol{\mu}_k)} w_k, \tag{2.6}$$

Wang et al. (2020) demonstrated that the normalizing constant of $\tilde{q}$ is c, i.e., the same as the normalizing constant of q, meaning that $\tilde{q}$ is potentially useful for estimating c, and $\tilde{\pi} = \tilde{q}/c$ is a normalized probability density.

To perform Warp-U bridge estimation one applies Algorithm 1 below. Wang et al. (2020) pointed out that the estimation in Step 3 of Algorithm 1 can be further improved by applying the simple warp transformations introduced by Meng and Schilling (2002) to $\tilde{q}$ and ϕ to further reduce the divergence (2.3), e.g., location shifts and scaling. These simple transformations can be effective for reducing the divergence after the Warp-U transformation because $\tilde{q}$ is approximately uni-modal

(and ϕ is uni-modal).

Algorithm 1 Warp-U estimation of c (Wang et al., 2020)

1: Input: parameters K and $w_k, S_k, \boldsymbol{\mu}_k$, for $k = 1, \ldots, K$, see (2.4), and samples $\boldsymbol{\theta}_{1,1}, \ldots, \boldsymbol{\theta}_{1,n_1} \sim \pi$ and $\boldsymbol{\theta}_{2,1}, \ldots, \boldsymbol{\theta}_{2,n_2} \sim \phi$.
2: Set $\boldsymbol{\theta}^*_{1,j} = \mathcal{F}_{\psi_j}(\boldsymbol{\theta}_{1,j})$, where $\mathcal{F}_{\psi_j}(\boldsymbol{\theta}_{1,j}) = \mathcal{S}^{-1}_{\psi_j}(\boldsymbol{\theta}_{1,j} - \boldsymbol{\mu}_{\psi_j})$, and ψ_j's are independently drawn according to (2.5), for $j = 1, \ldots, n_1$.
3: Apply (2.2) using the densities $q_1 = \tilde{\pi}$ in (2.6) and $q_2 = \phi$, and the samples $\boldsymbol{\theta}^*_{1,j}, \ldots, \boldsymbol{\theta}^*_{1,n_1} \sim \tilde{\pi}$ (Step 2) and $\boldsymbol{\theta}_{2,1}, \ldots, \boldsymbol{\theta}_{2,n_2} \sim \phi$ (Step 1).

2.2.2 Warp-U MCMC Method

The Warp-U transformation $\boldsymbol{\theta}^* = \mathcal{F}_\psi(\boldsymbol{\theta}) = \mathcal{S}^{-1}_\psi(\boldsymbol{\theta} - \boldsymbol{\mu}_\psi)$ maps the multi-modal target density p to an approximately uni-modal one $\tilde{p}$, which is the normalized version of (2.6). Here we also consider the inverse stochastic transformation $\mathcal{H}_{\psi^*}(\boldsymbol{\theta}^*) = \mathcal{F}^{-1}_{\psi^*}(\boldsymbol{\theta}^*) = \mathcal{S}_{\psi^*}\boldsymbol{\theta}^* + \boldsymbol{\mu}_{\psi^*}$, where to ensure that $(\psi^*, \mathcal{H}_{\psi^*}(\boldsymbol{\theta}^*))$ has the same distribution as $(\psi, \boldsymbol{\theta})$, the index ψ^* must be drawn from $P(\psi^*|\boldsymbol{\theta}^*) = \varpi(\psi^*|\mathcal{H}_{\psi^*}(\boldsymbol{\theta}^*))\pi(\mathcal{H}_{\psi^*}(\boldsymbol{\theta}^*))|\mathcal{H}'_{\psi^*}(\boldsymbol{\theta}^*)|$, where the final term on the right hand side is a Jacobian. By repeatedly applying $\mathcal{F}$ and then $\mathcal{H}$ we can generate a sequence of draws of $(\psi, \boldsymbol{\theta})$ which regularly switches between modes. As mentioned in Section 2.1, the key to this approach is that the Warp-U transformation maps each induced component π_k of the target to almost the same intermediate uni-modal density, meaning that the inverse Warp-U transformation maps to a random component of the target density, and therefore easily explores many modes.

We begin by assuming that a reasonable approximating mixture ϕ_{mix} has already been constructed, and return to identifying such an approximation at the end of this section. Algorithm 2 above specifies the basic version of our Warp-U MCMC sampler. In addition to the transformations already described, we include a MH step at the beginning of each iteration. This is necessary because otherwise our sampler sometimes gets stuck in a loop whereby it visits several modes and then returns to exactly the point where it started. This problem typically occurs when the number of components K is small and ϕ_{mix} is a poor approximation to π, because in that case the condi-

Algorithm 2 Basic Warp-U MCMC sampler

1: Input: parameters K and $w_k, S_k, \boldsymbol{\mu}_k$, for $k = 1, \ldots, K$, see (2.4), proposal variance σ^2, initial value $\boldsymbol{\theta}_0$, and number of samples to be collected T.
2: **for** $t = 1, \ldots, T$ **do**
3: (i) Generate $\boldsymbol{\theta}^{\text{MH}}$ using a MH step with proposal $\mathcal{N}(\boldsymbol{\theta}_{t-1}, \sigma^2\mathbf{I})$.
4: (ii) Sample ψ from

$$\varpi(\psi|\boldsymbol{\theta}^{\text{MH}}) = \frac{\phi^{(\psi)}(\boldsymbol{\theta}^{\text{MH}})}{\phi_{\text{mix}}(\boldsymbol{\theta}^{\text{MH}})}.$$

5: (iii) Set $\boldsymbol{\theta}^* = \mathcal{F}_\psi(\boldsymbol{\theta}^{\text{MH}})$, where $\mathcal{F}_\psi(\boldsymbol{\theta}^{\text{MH}}) = \mathcal{S}_\psi^{-1}(\boldsymbol{\theta}^{\text{MH}} - \boldsymbol{\mu}_\psi)$.
6: (iv) Sample ψ^* from

$$\nu(\psi^*|\boldsymbol{\theta}^*) \propto \underbrace{\varpi(\psi^*|\mathcal{H}_{\psi^*}(\boldsymbol{\theta}^*))q(\mathcal{H}_{\psi^*}(\boldsymbol{\theta}^*))|\mathcal{H}'_{\psi^*}(\boldsymbol{\theta}^*)|}_{cp(\psi^*, \mathcal{H}_{\psi^*}(\boldsymbol{\theta}^*))|\mathcal{H}'_{\psi^*}(\boldsymbol{\theta}^*)|}, \tag{2.7}$$

where $\mathcal{H}_{\psi^*}(\boldsymbol{\theta}^*) = \mathcal{F}_{\psi^*}^{-1}(\boldsymbol{\theta}^*) = \mathcal{S}_{\psi^*}\boldsymbol{\theta}^* + \boldsymbol{\mu}_{\psi^*}$.
7: (v) Set $\boldsymbol{\theta}_t = \mathcal{H}_{\psi^*}(\boldsymbol{\theta}^*)$.
8: **end for**

tional distribution $P(\psi^*|\boldsymbol{\theta}^*)$ may strongly favor particular components in each iteration, causing the components to be repeatedly sampled in a specific order. When the number of components is large, the chance of visiting the components in the same order multiple times is low, and so the problem is avoided. In any case, the issue is resolved by the MH step we include in Algorithm 2.

Algorithm 3 Adaptive Warp-U MCMC sampler

1: Input: proposal variance σ^2, initial value $\boldsymbol{\theta}_0$, number of within stage samples T, and number of stages M.
2: Obtain T samples Θ_0 from the uniform density with support Θ (or another initial density with support Θ).
3: Compute $\phi_{\text{mix}}^{(0)}$ by fitting ϕ_{mix} using the samples Θ_0 from Step 2.
4: **for** $s = 1, \ldots, M$ **do**
5: (i) Apply Algorithm 2 with the following inputs: parameters of $\phi_{\text{mix}}^{(s-1)}$, σ^2, $\boldsymbol{\theta}_0$, and T, we can obtain new samples Θ_{new}. Then we set $\Theta_s = \Theta_{s-1} \cup \Theta_{new}$.
6: (ii) Compute $\phi_{\text{mix}}^{(s)}$ by re-fitting ϕ_{mix} using the samples Θ_s from Step 5.
7: **end for**

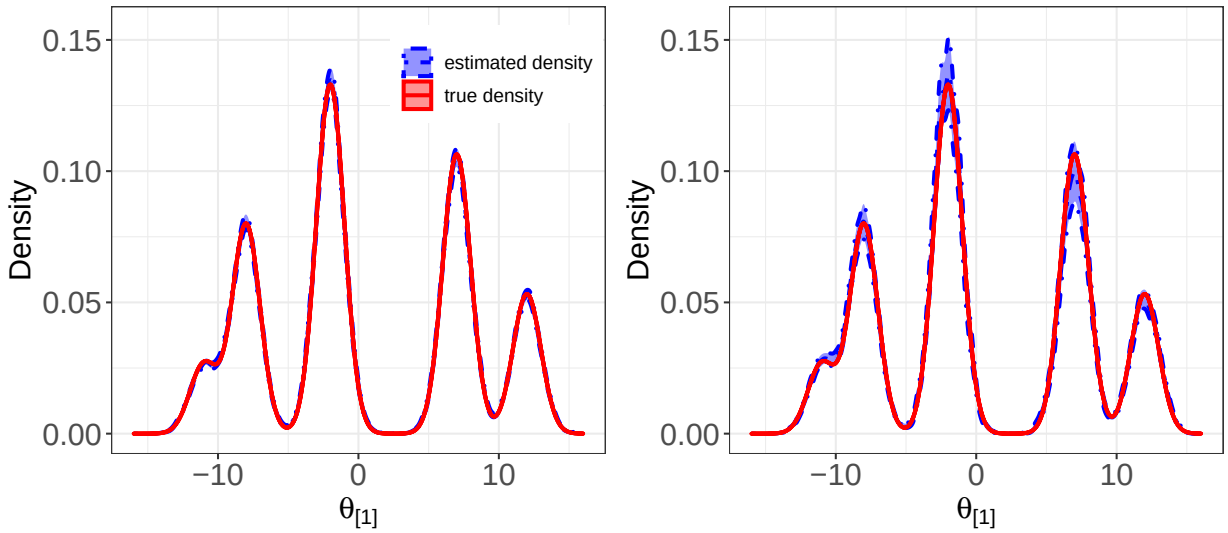

Figure 2.1: One marginal of the true target density (solid red line) and the estimated density (center dash-dot blue line) obtained by running Algorithm 2. The left and right panels show the case where ϕ_{mix} is set to be the target density and a two Gaussian mixture approximation, respectively. The shaded blue regions indicate pointwise 95% confidence intervals.

The performance of Algorithm 2 depends on the quality of the approximation ϕ_{mix}. We now consider the accuracy with which samples obtained from Algorithm 2 recover one marginal of the six-dimensional target density π, specified by (A.1) in Appendix A.1. The left panel of Figure 2.1 shows the true density (solid red line) and estimated density (center dash-dot blue line) in the case where ϕ_{mix} is the true target π, and the right panel shows the case where ϕ_{mix} is a two-mode Gaussian mixture distribution given by (A.4) in Appendix A.1. The blue regions show pointwise 95% confidence regions obtained by running Algorithm 2 fifty times with $T = 3000$ and then applying the kernel density estimation function `density` in `R` to each set of 3000 samples obtained. In both panels the confidence region is centered approximately on the true density, but as expected, it is much narrower in the left panel where the underlying ϕ_{mix} is a closer (exact) approximation to π.

In order to make our Warp-U MCMC sampler less dependent on the initial ϕ_{mix}, we introduce Algorithm 3 above, which is an adaptive version of Algorithm 2 and periodically updates ϕ_{mix}. The complexity of updating the Gaussian mixture approximation ϕ_{mix} is $\mathcal{O}(t_{\max}KTs)$, where $t_{\max}$ is

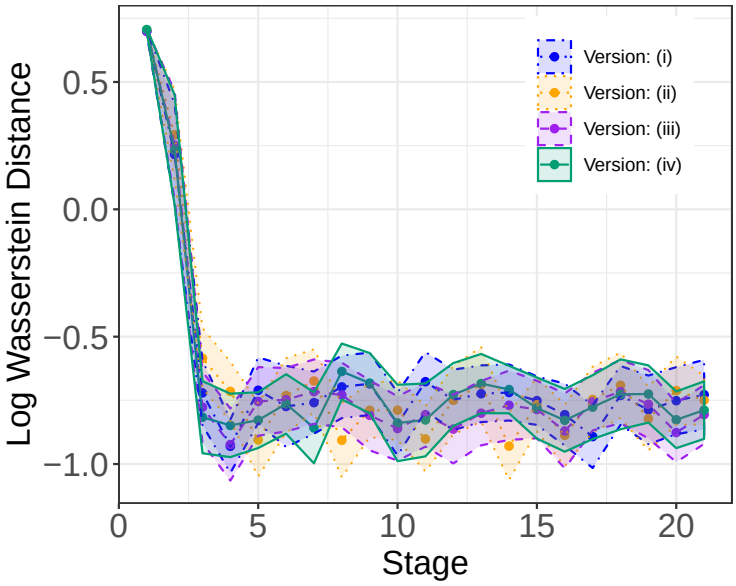

Figure 2.2: Log Wasserstein distance between the true target density (see (A.1) in Appendix A.1) and a kernel density estimate computed from samples generated via four versions of Algorithm 3, against the stage number of Algorithm 3. The four algorithm versions are described in the main text.

the maximum number of iterations of the EM algorithm to be used each time ϕ_{mix} is updated, and s is the current stage number of Algorithm 3. In practice, to save computational resources, the sT samples Θ_s in Step 5 of Algorithm 3 that are used to update ϕ_{mix} can be replaced by T samples randomly selected from Θ_s, or alternatively the updating of ϕ_{mix} can be eventually stopped, e.g., once $s > 10$. Figure 2.2 shows the log Wasserstein distance between the true target density and a kernel density estimate computed from samples generated via the following four versions of Algorithm 3: (i) Algorithm 3 as written above (dash-dot blue line), (ii) Step 6 is skipped for $s > 10$ (dotted orange line), (iii) the same as (i) except Step 6 uses T samples randomly selected from Θ_s to update ϕ_{mix}, rather than all the available samples (dashed purple line), and (iv) the same as (iii) except that 6 is skipped for $s > 10$ (solid green line). The log Wasserstein distances were computed using the `R` package '`transport`'. Figure 2.2 shows that all four procedures perform similarly as the stage number s increases.

2.2.3 Stochastic Warp-U Bridge Estimation

Having obtained samples from π using the Warp-U MCMC sampler introduced in Section 2.2.2, we then use these samples to estimate the target normalizing constant c. Here we propose a new estimation strategy which adapts and improves the Warp-U bridge estimator proposed by Wang et al. (2020). The main limitation of the Warp-U bridge sampling estimator is that it has high computational cost if q is expensive to evaluate, which is an important case because many estimation approaches can work well if q is inexpensive to evaluate (indeed, exact integration can be performed in the limit where q has no evaluation cost). The high computational cost is due to the fact that evaluation of $\tilde{q}$ in (2.6) requires K evaluations of q. Overall Algorithm 1 requires $K(n_1 + n_2)$ evaluations of q to obtain an estimate $\hat{c}$. Thus, although the Warp-U bridge estimator is statistically more efficient than standard bridge sampling with π and ϕ_{mix}, it is approximately K times more expensive. Consequently, Wang et al. (2020) found that Warp-U bridge sampling is only comparable to standard bridge sampling given fixed computational resources (and is sometimes slightly worse).

To propose a more computationally efficient strategy we exploit the expansion of $\tilde{q}$ given in (2.6). For the sake of generality, we allow $c_2 \neq 1$. To aid clarity we write $\tilde{q}_1 = \tilde{q}$, and then re-express (2.6) as

$$\tilde{q}_1(\tilde{\boldsymbol{\theta}}) = \sum_{k=1}^{K} \tilde{q}_{1k}(\tilde{\boldsymbol{\theta}}) w_k, \tag{2.8}$$

where

$$\tilde{q}_{1k}(\tilde{\boldsymbol{\theta}}) = \phi(\tilde{\boldsymbol{\theta}}) \frac{q(S_k \tilde{\boldsymbol{\theta}} + \boldsymbol{\mu}_k)}{\phi_{\text{mix}}(S_k \tilde{\boldsymbol{\theta}} + \boldsymbol{\mu}_k)}. \tag{2.9}$$

We can then split the estimand of interest into K parts, i.e.,

$$r = \frac{c_1}{c_2} = \frac{\int \sum_{k=1}^{K} \tilde{q}_{1k}(\boldsymbol{\theta}) w_k \, \mathrm{d}\boldsymbol{\theta}}{c_2} = \sum_{k=1}^{K} w_k \frac{c_{1k}}{c_2} = \sum_{k=1}^{K} w_k r_k. \tag{2.10}$$

Under this representation r can be interpreted as the expected value of a random variable R taking values $r_1, \ldots, r_K$, with probabilities $w_1, \ldots, w_K$, respectively. In other words, (2.10) suggests a

situation in which we want to estimate the normalizing constant of a *random* density divided by c_2. To estimate r we use the estimator

$$\hat{r}_{\text{SWB}} = \sum_{k=1}^{K} w_k \hat{r}_k, \tag{2.11}$$

where

$$\hat{r}_k = \frac{n_2^{-1} \sum_{j=1}^{n_2} \tilde{q}_{1k}(\boldsymbol{\theta}_{2,k,j}) \alpha(\boldsymbol{\theta}_{2,k,j})}{n_{1k}^{-1} \sum_{j=1}^{n_{1k}} q_2(\boldsymbol{\theta}^*_{1,k,j}) \alpha(\boldsymbol{\theta}^*_{1,k,j})}, \tag{2.12}$$

$\{\boldsymbol{\theta}^*_{1,k,1}, \ldots, \boldsymbol{\theta}^*_{1,k,n_{1k}}\} = \{\boldsymbol{\theta}^*_{1k} : \psi^*_j = k\}$, for $k = 1, \ldots, K$, and ψ^*_j is drawn from the conditional distribution $\varpi(\cdot|\boldsymbol{\theta}_{1,j})$ given by (2.5), for $j = 1, \ldots, n_1$. Here (2.12) is the classical bridge sampling estimator for r_k, and indeed the corresponding bridge identity holds, i.e.,

$$r_k = \frac{c_{1k}}{c_2} = \frac{E_{q_2}(\tilde{q}_{1k}(\boldsymbol{\theta})\alpha(\boldsymbol{\theta}))}{E_{\tilde{q}_{1k}}(q_2(\boldsymbol{\theta})\alpha(\boldsymbol{\theta}))}, \tag{2.13}$$

for $k = 1 \ldots, K$. We call our estimator (2.11) the *stochastic* Warp-U bridge (SWB) estimator in reference to the random density interpretation described above. Our stochastic bridge estimation procedure is detailed in Algorithm 4 below. It requires a total of $n_1 + Kn_2$ evaluations of q, which is $(K-1)n_1$ fewer than that needed for Warp-U bridge estimation. In Section 2.3.2, we prove that $\hat{r}_{\text{SWB}}$ is more efficient than Warp-U bridge estimation in terms of precision per CPU second.

2.3 Theoretical Justification

2.3.1 Ergodicity Property

In this section we establish that Algorithm 3 produces an ergodic Markov chain with stationary distribution π. We first consider Algorithm 2, which is non-adaptive. Let P_γ be the transition kernel for Algorithm 2, where $\gamma \in \mathcal{Y}$ denotes the parameters of ϕ_{mix}, and is included as a reminder that the kernel depends on these parameters, which are however treated as fixed in Algorithm 2. Lemma 1 confirms that the target density π is the stationary distribution of Algorithm 2. The proof is given in Appendix A.2.

Algorithm 4 Stochastic Bridge Sampling

1: Input: parameters K and $w_k, S_k, \boldsymbol{\mu}_k$, for $k = 1, \ldots, K$, see (2.4), and samples $\boldsymbol{\theta}_{1,1}, \ldots, \boldsymbol{\theta}_{1,n_1} \sim \pi$ and $\boldsymbol{\theta}_{2,1}, \ldots, \boldsymbol{\theta}_{2,Kn_2} \sim \phi$.
2: **for** $i = 1, \ldots, n_1$ **do**
3: (i) sample ψ^* from probability (2.5). Set $\boldsymbol{\theta}^*_{1\psi^*} = \mathcal{F}_{\psi^*}(\boldsymbol{\theta}_{1,i})$, where $\mathcal{F}_{\psi^*}(\boldsymbol{\theta}_{1,i}) = \mathcal{S}^{-1}_{\psi^*}(\boldsymbol{\theta}_{1,i} - \boldsymbol{\mu}_{\psi^*})$.
4: (ii) collect $\boldsymbol{\theta}^*_{1\psi^*}$ in set $Q_{1,k} = \{\boldsymbol{\theta}^*_{1,k,1}, \ldots, \boldsymbol{\theta}^*_{1,k,n_{1k}}\} = \{\boldsymbol{\theta}^*_{1k} : \psi^* = k\}$, where all samples in set $\{\boldsymbol{\theta}^*_{1k}\}$ follow $\tilde{q}_{1k}$ in (2.9), for $k = 1, \ldots, K$.
5: **end for**
6: **for** each component k from 1 to K **do**
7: Apply all transformed samples in set $Q_{1,k}$ and n_2 standard normal samples to classical Bridge estimation (2.2) with $q_1 = \tilde{q}_{1k}$ and $q_2 = \phi$ to get the estimation $\hat{r}_k$.
8: **end for**
9: Calculate the estimation $\hat{r} = \sum_{k=1}^{K} w_k \hat{r}_k$.

Lemma 1. *The transition kernel P_γ satisfies*

$$\int \pi(\boldsymbol{\theta}) P_\gamma(\boldsymbol{\theta}'|\boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{\theta} = \pi(\boldsymbol{\theta}'). \tag{2.14}$$

To further prove that π is also the stationary distribution of Algorithm 3, we need to establish that Algorithm 3 is ergodic, despite its adaptive nature. To achieve this we apply the framework proposed by Roberts and Rosenthal (2007), i.e., we prove the that the diminishing adaptation and simultaneous strongly aperiodic geometric ergodicity (SSAGE) conditions hold.

Lemma 2. *Suppose that $\Theta = \mathbb{R}^d$ and ϵ_s is the absolute tolerance used in the EM algorithm. If $\epsilon_s \to 0$ as $s \to \infty$, then the transition kernel of Algorithm 3 satisfies the diminishing adaptation condition, i.e.,*

$$\lim_{s\to\infty} \mathbb{P}(\sup_{\boldsymbol{\theta}\in\Theta} \| P_{\hat{\gamma}^s}(\,\cdot\,|\boldsymbol{\theta}) - P_{\hat{\gamma}^{s+1}}(\,\cdot\,|\boldsymbol{\theta})\|_{TV} \geq \delta_1) = 0 \tag{2.15}$$

for any $\delta_1 > 0$, where TV indicates the total variation distance. $\hat{\gamma}^s$ and $\hat{\gamma}^{s+1}$ is the estimated parameters of ϕ_{mix} used at s and $s+1$ stage.

Lemma 3. *For a state space $\boldsymbol{\theta} \in \Theta$, there exists a measurable set $C \in \mathcal{B}(\mathbb{R}^d)$, a drift function $V : \mathbb{R}^n \to [1, \infty]$ satisfying $\sup_{\boldsymbol{\theta}\in C} V(\boldsymbol{\theta}) < \infty$, and scalars $\delta > 0$, $\lambda < 1$, and $b < \infty$, such that*

the following two conditions hold:

C1. (Minorization). For each vector of map parameters $\hat{\gamma} \in \Gamma$, there is a probability measure $\omega_{\hat{\gamma}}$ defined on $C \subset \mathbb{R}^n$ with $P_{\hat{\gamma}}(\,\cdot\,|\boldsymbol{\theta}) \geq \delta\omega_{\hat{\gamma}}(\cdot)$.

C2. (Simultaneous drift). For $\hat{\gamma} \in \Gamma$ and $\boldsymbol{\theta} \in \mathbb{R}$, $\int_{\mathbb{R}^n} V(\boldsymbol{\beta}) P_{\hat{\gamma}}(\, d\boldsymbol{\beta}\,|\boldsymbol{\theta}) \leq \lambda V(\boldsymbol{\theta}) + bI_C(\boldsymbol{\theta})$.

The proofs of Lemmas 2 and 3 are given in Appendix A.2. Lemma 2 and 3 verify the diminishing adaptation and SSAGE conditions hold, respectively. Finally, Theorem 1 establishes that Algorithm 3 is ergodic. The proof is given in Appendix A.2.

Theorem 1. *Assume the absolute tolerance used in the EM algorithm ϵ_s satisfies $\epsilon_s \to 0$ as $s \to \infty$. The Warp-U sampling algorithm produces a Markov chain which is ergodic with the target distribution π as the stationary distribution.*

2.3.2 Asymptotic Variance and Precision Per Second of Stochastic Bridge Estimation

In this section, we demonstrate that our stochastic bridge estimator has lower asymptotic variance and greater Precision per Second (PpS) than Warp-U bridge estimation, where Pps is defined as $1/(\text{RMSE} \times \text{CPU seconds})$. We assume that all computational costs are negligible compare with evaluating q, and therefore CPU seconds are given by $eCg(q)$, where C is a constant, $g(q)$ is the time taken to evaluate q once, and e is the number of evaluations of q used by the method under consideration. The values of e for the different methods we compare are shown in the right three columns of Table 2.1, where n_1 and n_2 denote the number of the samples from the target and auxiliary distribution, respectively. The auxiliary distribution is the standard Gaussian for Warp-U bridge estimation and stochastic bridge estimation, and ϕ_{mix} for classical bridge sampling. The number of Warp-U bridge estimation target evaluations is lower in the case of Warp-U MCMC sampling because some of the necessary evaluations are computed during the sampling stage.

Meng and Wong (1996) showed that the asymptotic variance of the log bridge sampling estimator $\hat{\lambda} = \log(\hat{r})$ under the i.i.d. assumption is

$$\left(\frac{1}{n_1} + \frac{1}{n_2}\right)\left[(1 - H_A(p_1, p_2))^{-1} - 1\right] + o\left(\frac{1}{n_1 + n_2}\right), \tag{2.16}$$

Table 2.1: Number of evaluations of the unnormalized target density q for different sampling and estimation methods, where No. Iters represents the number of iterations (accept and reject), n_1 and $\tilde{n}_1$ denote the number of samples and average number of samples at each stage, respectively, M is the stage number, M_l is the number of temperature levels in PT, and n_2 is the number of samples from the auxiliary distribution.

Sampling	No. Iters	Sampling Evals.	Estimation Evals.		
			Bridge	Warp-U	S. Warp-U
Warp-U MCMC	$n_1 M$	$(K+1)n_1 M$	n_2	Kn_2	n_2
GWL	$\tilde{n}_1 M$	$\tilde{n}_1 M$	n_2	$(K-1)n_1 + Kn_2$	n_2
PT	$2n_1 M_l$	$n_1 M_l$	n_2	$(K-1)n_1 + Kn_2$	n_2

where $H_A(p_1, p_2)$ is the sample-size adjusted harmonic divergence defined in (2.3). Theorem 2 and 3 below establish our key result that the stochastics Warp-U bridge (SWB) estimation enjoys lower asymptotic variance and is computationally more efficient in terms of PpS under very mild conditions. The proof is given in Appendix A.3 and A.4.

Theorem 2. *Under the assumption of i.i.d. samples from π, ϕ and*

$$\sum_{k=1}^{K}\left[\left(\frac{1+\beta}{\beta}-\frac{w_k^2(\tilde{w}_k+\beta)}{\tilde{w}_k^2\beta}\right)\tilde{w}_k m(p_{1k})\right] \geq \sum_{k=1}^{K}\frac{1+\beta}{\beta}\tilde{w}_k\frac{m''(\tau_k)}{2}(p_{1k}-p_1)^2, \tag{2.17}$$

the asymptotic variances satisfy,

$$Var(\hat{\lambda}_{SWB}) \leq Var(\hat{\lambda}_{WB}), \tag{2.18}$$

where $\beta = \frac{n_2}{n_1}$, $\tilde{w}_k = \frac{n_{1k}}{n_1}$, $m(\tau_k) = (1 - H_A(\tau_k, p_2))^{-1} - 1$, and τ_k is some density between p_{1k} and p_1, for $k = 1, \ldots, K$.

Theorem 3. *Under the assumption of i.i.d. samples from π, ϕ and*

$$\sum_{k=1}^{K}\left[\left(K\frac{1+\beta}{\beta}-\frac{w_k^2(\tilde{w}_k+\beta)}{\tilde{w}_k^2\beta}\right)\tilde{w}_k m(p_{1k})\right] \geq \sum_{k=1}^{K}K\frac{1+\beta}{\beta}\tilde{w}_k\frac{m''(\tau_k)}{2}(p_{1k}-p_1)^2, \tag{2.19}$$

the precision per second satisfy,

$$PpS(\hat{\lambda}_{SWB}) \geq PpS(\hat{\lambda}_{WB}), \tag{2.20}$$

where $\beta = \frac{n_2}{n_1}$, $\tilde{w}_k = \frac{n_{1k}}{n_1}$, $m(\tau_k) = (1 - H_A(\tau_k, p_2))^{-1} - 1$, *and* τ_k *is some density between* p_{1k} *and* p_1, *for* $k = 1, \ldots, K$.

As mentioned in Section 2.2.3, stochastic bridge estimation requires $(1 - K)n_1$ fewer target evaluations than Warp-U bridge estimation, and therefore Theorem 3 does not necessarily imply that stochastic bridge estimation has lower asymptotic variance than Warp-U bridge estimation when only the condition (2.19) in Theorem 3 is satisfied. As stated in Appendix A.3 and A.4, both conditions are very milder as long as the Gaussian mixture approximate ϕ_{mix} is not too poor such that p_{1k} is too different from p_2 (usually set as standard Gaussian distribution). Since p_{1k} is the transformed distribution by Warp-U transformation, it will automatically follows a distribution that is close to standard Gaussian distribution, see equation (2.9), and this will lead to the conditions (2.17) or (2.19) can be easily satisfied, see details explained in Appendix A.3 and A.4. On the other hand, since the n_1 target samples are split between the K estimators $\hat{r}_k$ in (2.12), for $k = 1, \ldots, K$, meaning that small value of n_1 (relative to n_2) can lead to excessive variance of the individual estimators. To avoid this case, some criteria can be developed in the future to ensure each collection has lower bounded numbers of samples which mentioned in the Section 2.6. If the condition in Theorem 2 holds then we have

$$Var(\hat{\lambda}_{\text{SWB}}) \leq Var(\hat{\lambda}_{\text{WB}}) \leq Var(\hat{\lambda}_{\text{BS}}). \tag{2.21}$$

The final inequality states that the asymptotic variance of Warp-U bridge estimation is less than that of standard bridge sampling (BS), see Section 2.2.1 and Wang et al. (2020) for details.

2.4 Simulation Studies

2.4.1 Comparison of Sampling Methods

We now consider a six-dimensional multivariate Gaussian mixture target distribution π and compare the performance of our Warp-U MCMC sampling method to Parallel Tempering (PT) and Metropolis-Hastings (MH). PT (Geyer, 1991) is a widely used method designed to sample from multimodal target distributions, but it is necessary to tune the temperature levels, which can be time-consuming. For tuning the temperature level of PT, we apply Atchadé et al. (2011) to tune the temperature levels which iteratively selecting the inverse temperatures by a stochastic approximation algorithm described in Robbins and Monro (1951) and Andrieu and Robert (2001). For MH, we set the proposal density to be ϕ_{mix}, i.e., the same density used in the Warp-U transformations needed for our Warp-U MCMC sampler. The unnormalized target distribution is given by

$$q(\boldsymbol{\theta}) = \frac{1}{15}\exp(-\frac{1}{2}(\boldsymbol{\theta}+\mathbf{11})^T(\boldsymbol{\theta}+\mathbf{11})) + \frac{2}{15}\exp(-\frac{1}{2}(\boldsymbol{\theta}-\mathbf{12})^T(\boldsymbol{\theta}-\mathbf{12})) \tag{2.22}$$

$$+ \frac{3}{15}\exp(-\frac{1}{2}(\boldsymbol{\theta}+\mathbf{8})^T(\boldsymbol{\theta}+\mathbf{8})) + \frac{4}{15}\exp(-\frac{1}{2}(\boldsymbol{\theta}-\mathbf{7})^T(\boldsymbol{\theta}-\mathbf{7})) \tag{2.23}$$

$$+ \frac{5}{15}\exp(-\frac{1}{2}(\boldsymbol{\theta}+\mathbf{2})^T(\boldsymbol{\theta}+\mathbf{2})), \tag{2.24}$$

where bold numbers indicate the vectors in $\mathbb{R}^6$, e.g., $\mathbf{2} = (2,2,2,2,2,2)^T$. Note that some of the modes especially $-\mathbf{2}$ and $\mathbf{7}$ are well separated, and therefore a simple Metropolis-Hastings algorithm with a Gaussian proposal is not expected to work well.

We apply Algorithm 3 with $T = 4000$ (number of samples at each stage), $M = 11$ (number of stages), and $K = 10$ (number of components). Figure 2.3 shows the densities and pointwise 95% confidence intervals (middle dash-dot blue line and blue shaded regions) estimated from samples obtained using our adaptive Warp-U MCMC sampler at stage 1 (top left panel), 2 (top right panel), and 8 (bottom left panel), respectively. The solid red lines show the true target density. We can see that at stage 1 the confidence interval is highly inconsistent with the target density (since the

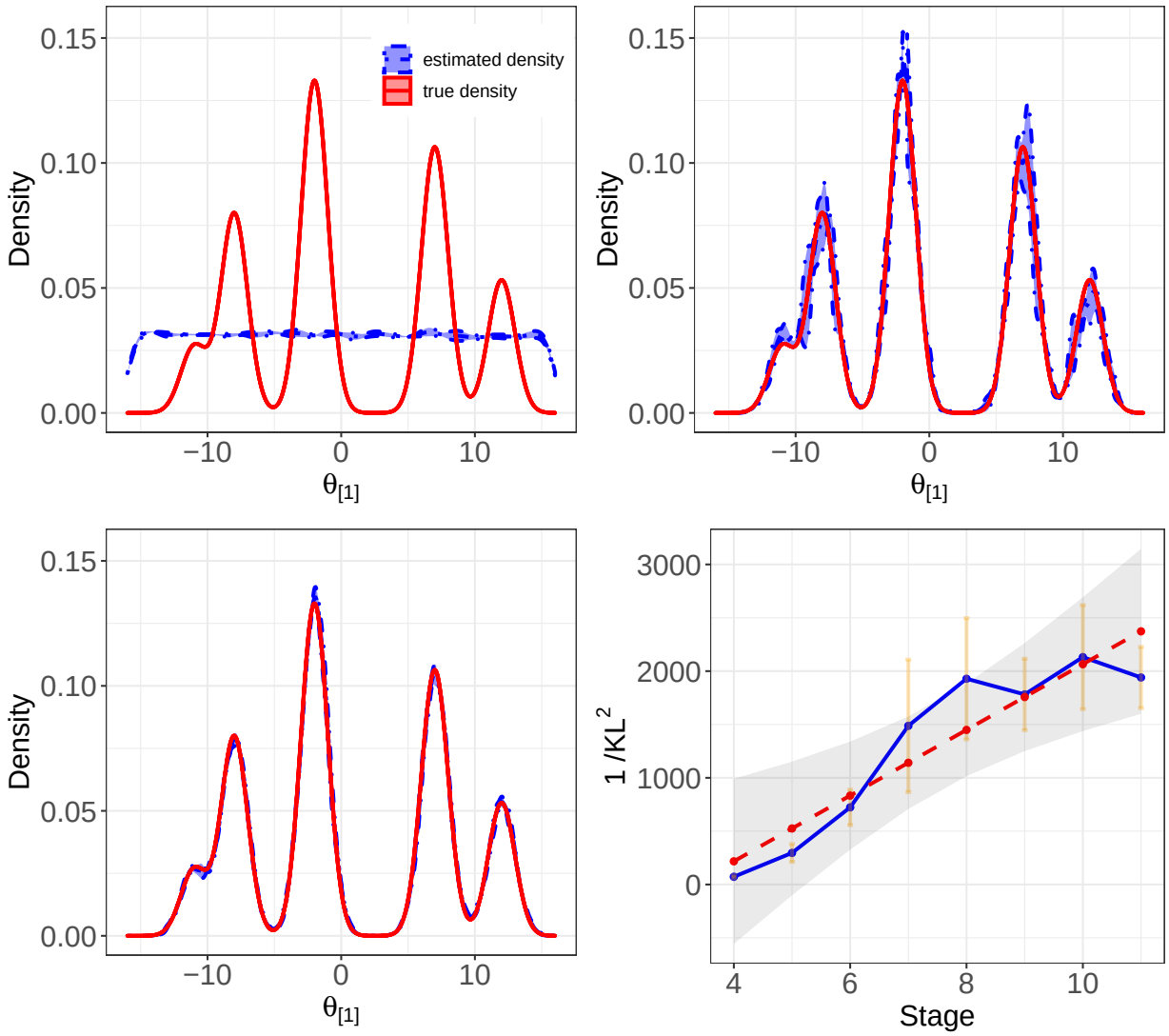

Figure 2.3: The top left, top right, and bottom left panels show the true target density (red line) and the density estimated from samples obtained using our adaptive Warp-U MCMC sampler (middle dash-dot blue line) after the first, second, and eighth stage of Algorithm 3, respectively. The blue shaded regions show pointwise 95% confidence intervals (marked by the upper and lower dash-dot blue lines). The bottom right panel shows the square of the 1/log KL divergence (solid blue line) and a 95% confidence interval, as well as a linear fit (dashed red line).

initial sampling is from a uniform distribution), but that the confidence interval at stage 8 covers the target density well (88.3% coverage). After trying different order of KL divergence, we find that square of $1/\log$ Kullback-Leibler (KL) divergence can be well fitted by a linear function of the stage number with 95% confidence interval (grey region) which is shown in the bottom right of Figure 2.3.

Figure 2.4 shows the log Wasserstein distance between the true target density and the density estimated from samples, as a function of target evaluations, when the sampling method is our adaptive Warp-U MCMC method (solid blue lines) or PT (dashed orange lines). Figure 2.4 shows that the samples obtained via our adaptive Warp-U MCMC sampler better approximate the true target density than those obtained via PT, for a given number of target evaluations. In other words, in this example, our approach has lower computational cost than PT for a given level of accuracy.

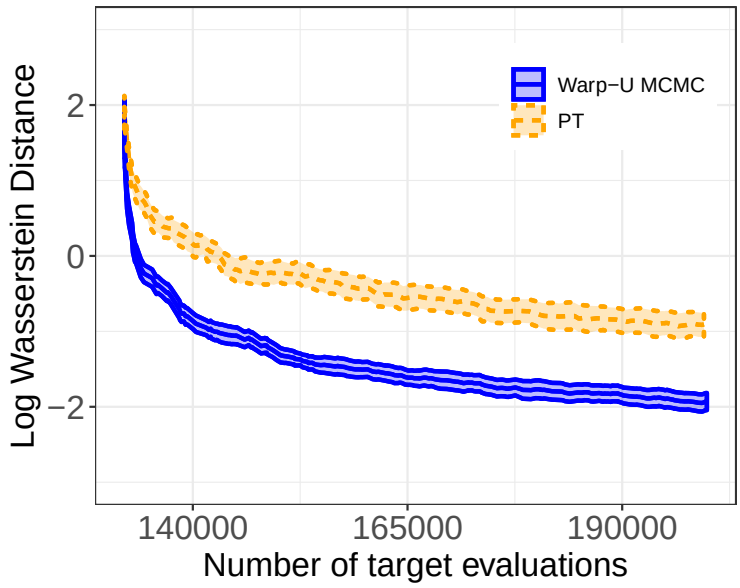

Figure 2.4: Log Wasserstein distance between the true target density and estimated density constructed using our adaptive Warp-U MCMC method (center solid blue line) and PT (center dashed orange line), as well as pointwise 95% confidence intervals (shaded regions).

Figure 2.5 shows the trace plot and auto-correlation function (ACF) for our non-adaptive Warp-

U MCMC sampler specified in Algorithm 2 (left column) and the Metropolis-Hastings (MH) algorithm with proposal density ϕ_{mix} (right column). We use the non-adaptive version of our sampler to avoid having to change the MH proposal density at each stage and simplify the comparison. The trace and ACF plots show that the Warp-U transformation used in Algorithm 2 greatly reduces the dependence between samples compared with the Metropolis-Hastings approach, and facilitates jumps between modes.

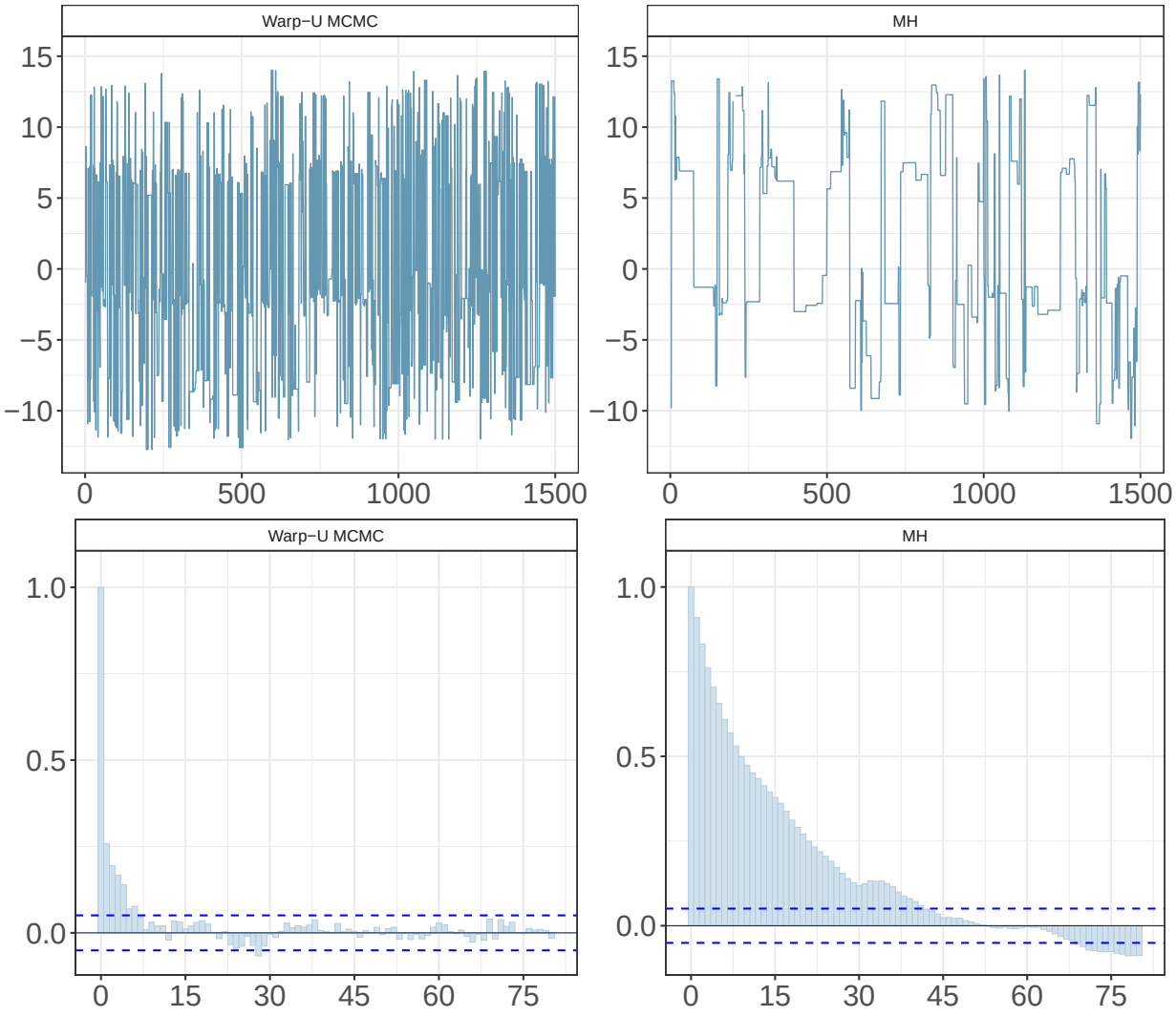

Figure 2.5: Trace and autocorrelation plots for our Warp-U MCMC method (left column) and MH with proposal density ϕ_{mix} (right column).

Table 2.2: Summary statistics of the log target density evaluated at 10^6 samples, where the target density is a 10-dimensional mixed skew-t distribution.

$\log(q)$	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
Target samples	-171.72	-27.86	-23.47	-24.67	-20.11	-10.26
Uniform samples	-53.07	-45.23	-43.71	-43.52	-42.01	-26.65

2.4.2 Comparison of Estimation Methods

We now compare our stochastic bridge estimation approach given by Algorithm 4 with bridge sampling (Meng and Wong, 1996) and Warp-U bridge estimation (Wang et al., 2020). The target density we consider is that used in Wang et al. (2020), and in particular is a 10 dimensional mixture of 25 multivariate skew-t distributions restricted to the support $[-20, 20]^{10}$. A similar target density was first introduced in a simulation study designed by Azzalini (2013). The target density presents a challenge in that large regions of its support have low density. To illustrate this, we generated 10^6 samples first from the target density itself and then from a uniform distribution on $[-20, 20]^{10}$. In the case of the uniform samples, we found that 99.99% had target density value lower than one-hundredth of the median target density value of the samples directly obtained from the target density, see Table 2.2.

The three estimation methods we consider are all designed to estimate normalizing constants *given* samples from the target density, and we investigate their performance when the samples are obtained via three different approaches: (i) our Warp-U MCMC sampler given by Algorithm 3; (ii) the GWL algorithm (Liang, 2005); and (iii) PT. The GWL algorithm does not directly sample from the target density and therefore in the case of (ii) we apply the additional sub-sampling step suggested in Section 5.2 of Wang et al. (2020). We set the number of samples from the target density to be the same, i.g., 25937 for all three sampling methods.

Figure 2.6 shows the root mean square error (RMSE) of the estimator as a function of the number of target evaluations for the classical bridge sampling (BS) estimator (green circles, and dash-dot line), the Warp-U bridge (WB) estimator (red triangles, and dotted line), and our stochas-

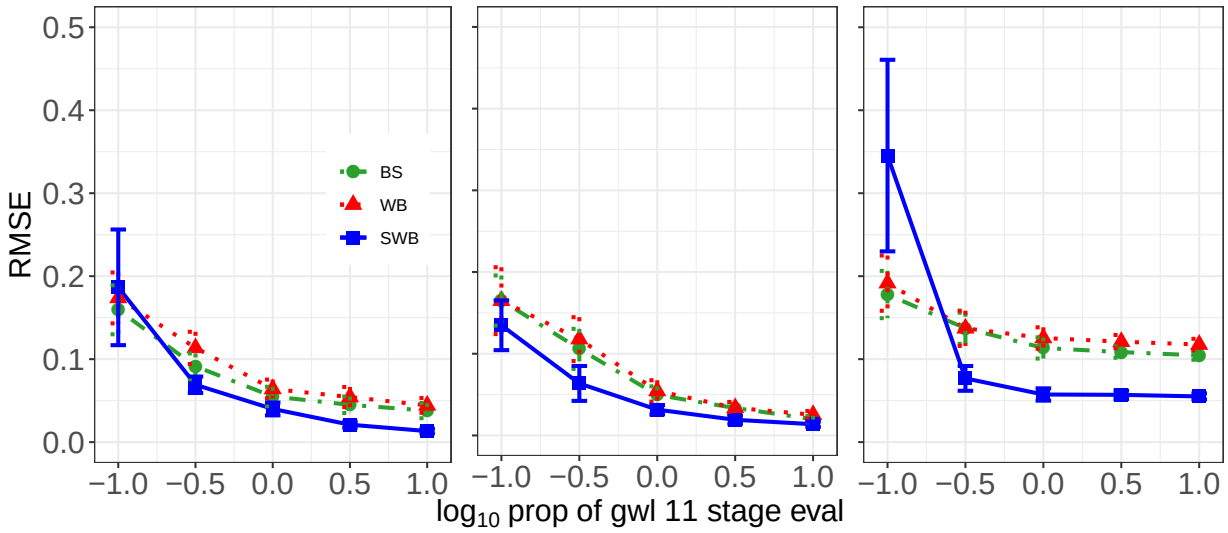

Figure 2.6: Root mean square error (RMSE) of three normalizing constant estimators as a function of the number of target evaluations, where the samples are obtained using our adaptive Warp-U MCMC sampler (left), GWL (middle), and PT (right). The three estimators are the classical bridge sampling (BS) estimator (green circles, and dash-dot line), the Warp-U bridge (WB) estimator (red triangles, and dotted line), and our stochastic Warp-U bridge (SWB) estimator (blue squares, and solid line). The error bars show 2 times the standard errors. The x-axis gives the number of estimation stage target evaluations as $\log_{10}$ of the proportion of target evaluations needed for an 11th stage of the GWL algorithm.

tic Warp-U bridge (SWB) estimator (blue squares, and solid line), when the samples are provided by our adaptive Warp-U MCMC sampler (left), GWL (middle), and PT (right). Our stochastic bridge estimator has the smallest RMSE regardless of the sampling method used, except when the number of target evaluations is small. Whether using GWL or Warp-U MCMC samples, the results are similar, and slightly better than with PT samples. On the other hand, GWL does not directly generate samples from target distribution, and additional sub-sampling steps (see Section 5.2 of Wang et al. (2020)) are needed to get the target samples, so the total computational cost is higher than for the other sampling methods (only the cost of estimation is illustrated in Figure 2.6).

We now return to Table 2.1 to compare the computational cost of the three estimation methods. In Table 2.1, M denote the number of stages in our adaptive Warp-U MCMC sampler and the GWL algorithm, and M_l denote the number of temperature levels for PT. For the GWL algorithm, the number of samples collected at each stage is different, so we use $\tilde{n}_1$ to denote the average number of samples across all stages. In addition, the burn-in are needed in all three cases. Note that the number of iterations given in Table 2.1 is not necessarily the number of unique samples, because some proposed samples are rejected. Table 2.1 shows that classical bridge sampling and stochastic Warp-U bridge estimation both require substantially less computation than Warp-U bridge estimation, even when some estimation stage computation is saved by use our Warp-U MCMC sampling method.

2.5 Bayesian Evidence: Explore New Planets Using Radial Velocity (RV) Data

In astronomy, plenty of types of data are explored to detecting the exoplanets. Among them, Radial velocity (RV) data is a popular one. Measuring from the earth, the astronomers can record the speed of the star moving away or towards us, i.e., the RV in meters per second. Since the planets orbit the stars, we can infer whether there is an exoplanet or not there by measuring the movement of the star. In the context of Bayesian statistics, we aim to compute the quantitative evidence, i.e., Bayesian evidence,

$$\mathcal{Z} \equiv \int p(\boldsymbol{d}|\boldsymbol{\theta}, \mathcal{M}) p(\boldsymbol{\theta}|\mathcal{M}) \, \mathrm{d}\boldsymbol{\theta}, \tag{2.25}$$

where $\boldsymbol{d}$ is the RV data, $\mathcal{M}$ represents the underlying physical and noise model and the $\boldsymbol{\theta}$ is the parameters of model $\mathcal{M}$. Here $p(\boldsymbol{d}|\boldsymbol{\theta}, \mathcal{M})$ is known as the likelihood and $p(\boldsymbol{\theta}|\mathcal{M})$ as the prior. The Bayesian evidence $\mathcal{Z}$ is the normalizing constant of the posterior. The statistics model of the likelihood is illustrated by

$$v_i = v_{\text{pred}}(t_i|\boldsymbol{\theta}) + \epsilon_i, \tag{2.26}$$

where v_{pred} is the physical model introduced in Loredo et al. (2012), with input time t and model parameters $\boldsymbol{\theta}$, see details in Appendix A.5. v_i is the component of the $\boldsymbol{d}$ and t_i is corresponding time. ϵ_i is the component of the noise $\boldsymbol{\epsilon}$ following from a multivariate Gaussian distribution, i.e., $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$. The covariance $\boldsymbol{\Sigma}$ can be captured by

$$\boldsymbol{\Sigma}_{ij} = \kappa_{ij} + \delta_{ij}(\sigma_i^2 + \sigma_J^2). \tag{2.27}$$

where κ_{ij} is a quasi-periodic kernel, δ_{ij} is the Kronecker delta, σ_J^2 is the amplitude of an additional unknown noise term and σ_i^2 is the measurement error. Here the quasi-periodic kernel is chosen as

$$\kappa_{ij} = \alpha^2 \exp\left[-\frac{1}{2}\left\{\frac{\sin[\pi(t_i - t_j)/\tau]}{\lambda_p^2} + \frac{t_i - t_j}{\lambda_e^2}\right\}\right], \tag{2.28}$$

where the hyperparameters are fixed as $\alpha = \sqrt{3}m/s$, $\lambda_e = 50.0$ days, $\lambda_p = 0.5$ (unit-less), and $\tau = 20$ (days).

Our data set consists of $n = 200$ observations from the RV community, see Nelson et al. (2018) for details. Each observation is accompanied by the time at which the RV is measured and a known measurement error (i.e., standard deviation). Left panel of Figure 2.7 shows the data of the RV varying with the time.

2.5.1 Comparison of the Density with Parallel Tempering and Hamiltonian Monte Carlo

To show the performance of our adaptive Warp-U MCMC sampler, we compare our method with two popular sampling method PT and Hamiltonian Monte Carlo. Although the PT is one of the popular sampling methods to samples from a multimodal target distribution, turning the

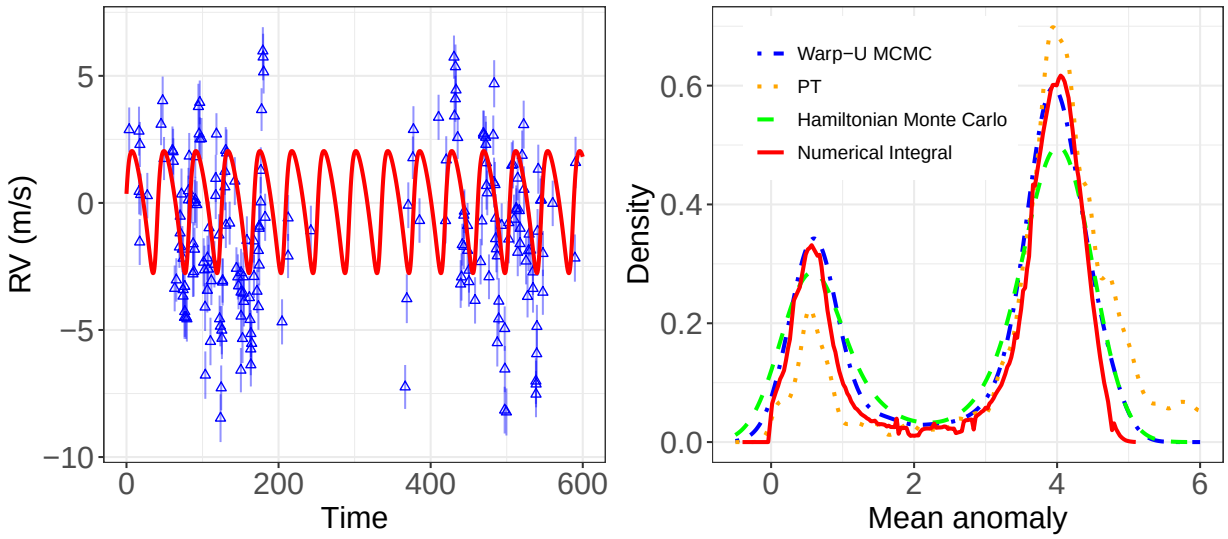

Figure 2.7: The left panel shows the radial velocity as a function of time, and the associated measurement errors. The right panel shows the marginal posterior distribution of the mean anomaly parameter. It compares the estimated densities using the samples obtained by Warp-U MCMC (dash-dot blue line), PT (dotted orange line) and Hamiltonian Monte Carlo(dashed green line). The solid red line is the estimated target density by numerical integral.

algorithm is not user friendly and much time-consuming.

The right panel of Figure 2.7 shows the plot of comparison between different sampling methods. The adaptive Warp-U MCMC sampler is more accurate with the numerical integral. The numerical integral is not a better choice when target density is multimodal because it has huge computation cost. Here we only treat it as a reasonable true target density to make all comparisons have a baseline, although the numerical integral costs huge time with the numerical error.

2.5.2 Estimation of the Bayesian Evidence

Now we compare the performance of stochastic Warp-U bridge estimation with classical bridge sampling estimation and Warp-U bridge estimation. We first apply PT and our adaptive Warp-U MCMC sampler to get target samples that is needed in estimation step. The temperature grid of PT is turned by the algorithm in Atchadé et al. (2011). Then we treat these samples as the input of bridge sampling, Warp-U bridge estimation, and stochastic Warp-U bridge estimation methods to see the performance of each method, separately. To make the comparison fair, we set the number of

the evaluation of the target density of all methods the same both in the sampling step and estimation step.

Table 2.3: RMSE (and associated SE) when estimating the $\log_{10}$ Bayesian evidence for a planet using bridge sampling, Warp-U bridge estimation, and stochastic Warp-U bridge estimation.

	Bridge		Warp-U		S. Warp-U	
Sampling	RMSE	SE	RMSE	SE	RMSE	SE
PT	0.238	0.015	0.239	0.017	0.115	0.008
Warp-U	0.136	0.004	0.137	0.008	0.059	0.002

Table 2.3 shows the comparison of root mean square error (RMSE) between bridge sampling, Warp-U bridge estimation, and stochastic Warp-U bridge estimation. The RMSE of our stochastic Warp-U bridge estimation is smaller than bridge sampling and Warp-U bridge estimation. It also shows that the RMSE is lower when the target samples are obtained by the adaptive Warp-U MCMC sampler than that obtained by PT which illustrates the better performance of our sampling method. Our stochastic Warp-U bridge's standard errors (SE) of RMSE are smaller than the other two estimation errors and this result is consistent with the smaller asymptotic variance compared with bridge sampling and Warp-U bridge estimation in Theorem 2. The best estimator $\log_{10}(-193.7386)$ is obtained by stochastic Warp-U bridge estimation using the samples from Warp-U MCMC sampler. It is closest to the median value in Nelson et al. (2018), which can be treated as the quasi true estimation.

2.6 Discussion

The component number of Gaussian mixture components in the approximating density ϕ_{mix} plays a role in the performance of the Warp-U MCMC sampler. In many settings, starting with a moderate number of components, e.g., $K = 20$, is practical. If the target density has fewer than K modes, this will not substantially negatively impact the performance of the sampling method, and the main price of specifying too many components is the computation cost of updating the

Gaussian mixture approximation ϕ_{mix}. Of course, users can also use a model selection criterion, e.g., BIC, to determine the number of components, and this may be particularly helpful when the target density has many modes.

In stochastic Warp-U bridge sampling estimation, when allocating the samples to different components in Steps 3-4 of Algorithm 4, there may be some components that have very few samples, which will lead to high variance of the bridge sampling estimator for those components. To address this problem, future work could develop a restriction to ensure that each component has a minimum number of samples.

3. FUNCTIONAL PCA WITH COVARIATE DEPENDENTMEAN AND COVARIANCE STRUCTURE

3.1 Introduction

Due to the prevalence and variety of functional data, and the computational challenges which arise through their analysis, many FDA methods have been developed. Among these approaches, the most fundamental is functional principal component analysis (FPCA), which we now briefly overview, with essential mathematical details deferred to Section 3.2.1. A comprehensive introduction of FPCA is given in Ramsay and Silverman (2005). In its canonical form, FPCA represents each functional observation as a mean function plus a linear combination of functional principal components (FPCs) and noise. The key to this method is that it is often possible to well capture the data using only a small number of FPCs, i.e., by imposing a low rank structure on the underlying covariance function. Furthermore, by constraining the inferred FPCs to be smooth, we can ensure that the corresponding covariance function is also smooth. Empirical studies have shown that this type of smoothness often improves estimation accuracy and predictive performance due to the usual bias-variance trade-off encountered in statistical inference.

One leading FPCA framework is based on the work of James et al. (2000), which interprets FPCA as a mixed effects model for sparse and irregularly sampled functional data. Their estimation strategy relies on a basis representation of the underlying eigenfuctions or FPCs, and imposes smoothness by limiting the number of basis functions in the expansion. Another popular FPCA approach is to compute the eigenvectors of the sample covariance matrix resulting from a locally smoothed approximation to the underlying covariance function, e.g., see Rice and Silverman (1991); Yao et al. (2005). A third strategy, proposed by Cai and Yuan (2010), relies on a covariance function expansion induced by a user-specified tensor product reproducing kernel Hilbert space (RKHS). This approach has some appealing features, but suffers from the fact that the number of parameters to be optimized grows linearly with the sample size (i.e., the number

of curves) and quadratically with the number of observations per curve. Furthermore, it uses least squares estimation which can be less efficient than maximum likelihood methods. In the Bayesian setting, Suarez and Ghosal (2017) proposed an FPCA method that uses a basis expansion of the FPCs (in a similar manner to James et al., 2000), and induces a prior on them via a prior on the basis coefficients. Van Der Linde (2008) suggested a variational inference approach to overcome the computational challenges that arise in Bayesian FPCA. Further Bayesian FDA methods include, e.g., Behseta et al. (2005); Rodríguez et al. (2009).

We propose an FPCA method that incorporates covariates in a computationally efficient manner, which is a key extension of the above approaches. Indeed, covariates are often available in practice, and their inclusion can facilitate low-rank representations of functional data and substantially improve predictive performance. There are a number of existing strategies for incorporating covariates, including early methods such as Capra and Müller (1997), and more recent approaches which primarily build on the work of Yao et al. (2005), e.g., Jiang and Wang (2010, 2011); Zhang et al. (2013); Zhang and Wang (2016). However, these methods all rely on local smoothing, which is computationally costly. Thus, in practice, they are slow and cannot be used for analyzing datasets which are large or have more than a few covariates. The literature also proposes a number of alternative strategies which achieve greater computational efficiency at the expense of simplifying assumptions, e.g., Li et al. (2016) assumes that only the FPC scores vary with the covariates, that they do so linearly, and that the sampling points are the same for each functional observation (often called balanced sampling). However, such assumptions are often violated in practice.

The approach we propose here overcomes the computational challenges of including covariates, while simultaneously allowing both the mean function and the covariance function to depend on the covariates in a non-linear way, as well as permitting unbalanced sampling patterns. We call our method *Covariate Dependent Functional PCA*, or CD-FPCA for short. Its computational efficiency results from a basis representation of the mean and covariance functions, similar to that used by James et al. (2000), and a modeling approach which carefully avoids costly optimization problems. Non-linear dependence of the covariance function on the covariates is captured by allowing

the underlying FPCs, as well as their scores, to depend flexibly on the covariates. Our CD-FPCA approach contains as a special case the Supervised Sparse and Functional PCA (SupSFPC) method proposed by Li et al. (2016), but avoids the simplifying assumptions mentioned earlier. As a result, our method outperforms SupSFPC in all our numerical studies, as well as James et al. (2000) (no covariate dependence) and Jiang and Wang (2010), the latter being representative of the linear smoothing approaches.

A challenge for our approach, and other FPCA methods, is ensuring that the underlying covariance function and FPCs have the required properties, e.g., positive semi-definiteness and orthogonality, respectively. In the James et al. (2000) framework, a low-rank covariance function is assumed, meaning that only a small number of FPCs are needed to represent it via the Karhunen-Loève expansion. This reduces the positive semi-definiteness constraint to ensuring that a small number of eigenvalues are positive. Peng and Paul (2009) addressed the orthogonality constraint by representing the FPCs with a finite basis expansion and using a restricted maximum likelihood (REML) method to fit the basis coefficients. Our approach also relies on a finite basis expansion, but considers the covariance function directly rather than the FPCs. In particular, we use the fact that the low-rank covariance function can be represented by $G(t, s|\boldsymbol{z}) \approx \boldsymbol{b}(t)^T \boldsymbol{\Sigma}(\boldsymbol{z}) \boldsymbol{b}(s)$, where $\boldsymbol{z}$ is a covariate vector and $\boldsymbol{b}(t)$ is a vector of the basis functions evaluated at t. Then we propose a model for $\boldsymbol{\Sigma}(\cdot)$ that maps Euclidean space to the symmetric positive semi-definite rank r matrix manifold, thereby automatically ensuring that $\boldsymbol{\Sigma}$ and G are positive semi-definite, and facilitating straightforward model fitting. Some recent papers have explored methods to model manifold-valued data, e.g., Lin et al. (2017). We choose the basis to be orthonormal because this makes it easy to obtain estimates of the underlying covariate dependent FPCs that satisfy the orthogonality constraint after fitting the model.

We additionally make use of roughness penalties to ensure smoothness of the mean and covariance function. This is preferable to controlling smoothness by choosing a small fixed number of basis functions, because the latter approach is discontinuous in nature. Our penalization method is based on the computationally efficient approach proposed by Wood (2006) for penalizing functions

with multiple arguments. Roughness penalties are a well-established tool in FDA, e.g., Pezzulli (1993); Silverman (1996); Cai and Yuan (2011).

This chapter related to supervised functional PCA is organized as follows. Section 3.2 briefly reviews classical FPCA, presents our covariate dependent model for the mean and covariance functions, and introduces a roughness penalty similar to that proposed by Wood (2006). Section 3.3 details our algorithm, which makes use of several approximations to reduce computational cost. In Sections 3.4 and 3.5, we compare our method with the approaches proposed by Li et al. (2016), James et al. (2000), and Jiang and Wang (2010) through a simulation study and an astronomical data analysis. Brief discussion is found in Section 3.6. Appendix B provide details of our roughness penalty, proofs, and key information about the competing method proposed by Li et al. (2016).

3.2 Covariate Dependent FPCA Model

3.2.1 Classical FPCA

Suppose that there are N latent functions of interest, denoted by x_n, for $n = 1, \ldots, N$. Let $x_n(t)$ denote the value of the n-th latent function at time t, for $n = 1, \ldots, N$. More generally, we could consider functions of other types of variable, such as location, but here restrict our attention to functions of time. The covariance function $\text{cov}(x_n(t), x_n(s)) = G(t, s)$ has eigendecomposition $G(t, s) = \sum_{j=1}^{\infty} d_j f_j(t) f_j(s)$, where f_j is the j-th eigenfunction or FPC and d_j is the corresponding eigenvalue. The eigenfunctions are orthonormal to each other, and the eigenvalues are ordered non-increasingly $d_1 \geq d_2 \geq d_3 \geq \ldots$ By the Karhunen-Loève theorem, the function $x_n(t)$ can be expressed by a linear combination of a mean function $\mu(t)$ and the above eigenfunctions, i.e.,

$$x_n(t) = \mu(t) + \sum_{j=1}^{\infty} \xi_n^{(j)} f_j(t), \tag{3.1}$$

where the $\xi_n^{(j)} = \int x_n(t) f_j(t) dt$ are uncorrelated random variables with mean 0 and variance d_j, for $j = 1, 2, \ldots$, e.g., often it is assumed that $\xi_n^{(j)} \sim \mathcal{N}(0, d_j)$, where $\mathcal{N}$ denotes a Gaussian

distribution. The low-rank approximation used by James et al. (2000) and others truncates the summation in (3.1), and hence also the eigen-decomposition for the covariance function G. This is generally a good approximation if $\sum_{j=r+1}^{\infty} d_j \ll \sum_{j=1}^{r} d_j$. Next, a second modification of (3.1) is needed because we typically only get to observe a noisy version of $x_n(t)$, which we denote by $y_n(t)$, for $n = 1, \ldots, N$. Putting these two points together, we write

$$y_n(t) = x_n(t) + \epsilon(t) \approx \mu(t) + \sum_{j=1}^{r} \xi_n^{(j)} f_j(t) + \epsilon(t) = \mu(t) + \boldsymbol{f}^T(t)\boldsymbol{\xi}^{(n)} + \epsilon(t), \tag{3.2}$$

where $\epsilon(t)$ denotes white noise with mean 0 and variance σ_e^2. The right-hand side of (3.2) uses the vector notation $\boldsymbol{f}(t) = (f_1(t), \ldots, f_r(t))^T$ and $\boldsymbol{\xi}^{(n)} = (\xi_1^{(n)}, \ldots, \xi_r^{(n)})^T$. The mixed effects interpretation introduced by James et al. (2000) is derived from (3.2) by replacing $\mu(t)$ and $f_j(t)$, for $j = 1, \ldots, r$, by basis expansions. The resulting expression then has some *fixed* basis coefficients and some *random* coefficients, hence the mixed effects interpretation.

3.2.2 Model extension to include covariates

Let $\boldsymbol{z}$ denote a vector of covariates, and suppose that $t \in \mathcal{T}$ and $\boldsymbol{z} \in \mathcal{Z}$, where $\mathcal{T} = [t_{\min}, t_{\max}]$ and $\mathcal{Z}$ are compact domains. To incorporate covariate dependence we replace (3.2) by

$$y_n(t, \boldsymbol{z}) \approx \mu(t, \boldsymbol{z}) + \sum_{j=1}^{r} f_j(t, \boldsymbol{z})\xi_j^{(n)} + \epsilon(t). \tag{3.3}$$

In this model, both the mean function $\mu(t, \boldsymbol{z})$ and the FPCs $f_j(t, \boldsymbol{z})$ are allowed to vary smoothly with the covariates $\boldsymbol{z}$, as will be explained further in Sections 3.2.3 and 3.2.4 below. In the rest of the work, we will particularly focus on the case when $\boldsymbol{z} \in \mathbb{R}$ is a scalar to avoid the curse of dimensionality.

The score vector $\boldsymbol{\xi}^{(n)}$ follows a multivariate Gaussian distribution with zero mean and covariance matrix $\mathbf{D}_{\boldsymbol{z}}$, which also depends on $\boldsymbol{z}$, i.e., $\boldsymbol{\xi}^{(n)} \sim \mathcal{N}_r(\mathbf{0}, \mathbf{D}_{\boldsymbol{z}})$, where $\mathcal{N}_r$ denotes a multivariate Gaussian distribution of dimension r. For a given covariate vector $\boldsymbol{z}$, the covariance function of the random latent functions is $G(t, s|\boldsymbol{z}) \approx \boldsymbol{f}^T(t, \boldsymbol{z})\mathbf{D}_{\boldsymbol{z}}\boldsymbol{f}(s, \boldsymbol{z})$, with the elements of the vector

$\boldsymbol{f}(s, \boldsymbol{z})$ being $f_j(t, \boldsymbol{z})$, for $j = 1, \ldots, r$. Thus, the rank of the underlying covariance function is r, regardless of $\boldsymbol{z}$.

3.2.3 Covariate dependent mean function

In our model (3.3), we allow the mean function $\mu(t, \boldsymbol{z})$ to vary smoothly with both time t and the covariates $\boldsymbol{z}$, which we achieve using a tensor product spline basis, as we now explain. Firstly, dependence on t is captured by an orthonormal cubic spline with l' equally spaced knots in the temporal domain $\mathcal{T}$, i.e., with $l = l' + 2$ degrees of freedom. Similarly, dependence on the covariates $\boldsymbol{z}$ is captured using an orthonormal cubic spline with p' knots, and therefore $p = p' + 2$ degrees of freedom. In our implementation, these splines are represented using orthonormal cubic B-spline bases, because computationally efficient methods for B-Splines and derivatives thereof are well developed, e.g., see Butterfield (1976) and De Boor (1977) and penalties can be calculated directly, see Section B.1 of Appendix B.

Let $\boldsymbol{a}(t) \in \mathbb{R}^l$ and $\boldsymbol{u}(\boldsymbol{z}) \in \mathbb{R}^p$ be the values of the orthonormal B-spline basis functions evaluated at t and $\boldsymbol{z}$, respectively, e.g., the i-th entry of $\boldsymbol{a}(t)$ is the value of the i-th temporal domain B-spline basis function evaluated at t and satisfy $\int_{\mathcal{T}} \boldsymbol{a}(t)\boldsymbol{a}(t)^T \mathrm{d}t = \mathbf{I}_l$, where $\mathbf{I}_l$ denotes the identity matrix and will be used throughout with dimensions implied by the context, e.g., here it has dimensions $l \times l$. With this notation, our proposed mean function is given by

$$\mu(t, \boldsymbol{z}) \approx \boldsymbol{a}(t)^T \boldsymbol{\Theta}_\mu \boldsymbol{u}(\boldsymbol{z}) = \sum_{i=1}^{l} \sum_{j=1}^{p} a_i(t) u_j(\boldsymbol{z}) \theta_{ij} = \mathbf{H}(t, \boldsymbol{z})^T \boldsymbol{\theta}_\mu, \tag{3.4}$$

where $\boldsymbol{\Theta}_\mu = \left(\theta_{ij}\right) \in \mathbb{R}^{l \times p}$ is the matrix of orthonormal B-spline basis coefficients, $\mathbf{H}(t, \boldsymbol{z}) = \boldsymbol{a}(t) \otimes \boldsymbol{u}(\boldsymbol{z})$ is a tensor product spline basis, and $\boldsymbol{\theta}_\mu = \mathrm{vec}(\boldsymbol{\Theta}_\mu)$, i.e., the vectorization of the coefficients matrix $\boldsymbol{\Theta}_\mu$.

3.2.4 Covariate dependent covariance function

To model the covariance $G(t, s|\boldsymbol{z})$, we rely on an orthonormal cubic B-spline basis with $w - 2$ knots in the temporal domain $\mathcal{T}$ and its function values at t are denoted $\boldsymbol{b}(t) \in \mathbb{R}^w$. We use

an orthonormal basis because this enables us to easily express the penalties and obtain estimates of the underlying FPCs after training our model. In general we set $w > r$ to facilitate accurate spline approximations of the true eigenfunctions, under the assumption of low rank structure. Our covariance function model can now be introduced as

$$G(t, s|\boldsymbol{z}) \approx \boldsymbol{b}(t)^T \boldsymbol{\Sigma}(\boldsymbol{z}; \boldsymbol{\beta}) \boldsymbol{b}(s), \tag{3.5}$$

where $\boldsymbol{\Sigma}(\boldsymbol{z}; \boldsymbol{\beta}) \in \mathbb{R}^{w \times w}$ is a matrix that depends on the covariates $\boldsymbol{z}$ and is parameterized by coefficients $\boldsymbol{\beta}$, as we now explain.

We must ensure that our covariance function G in (3.5) is symmetric positive semi-definite. This is equivalent to requiring $\boldsymbol{\Sigma}(\boldsymbol{z}\,; \boldsymbol{\beta})$ to be a symmetric positive semi-definite matrix for each $\boldsymbol{z}$. Based on the ideas in Zhu et al. (2009), we construct $\boldsymbol{\Sigma}(\cdot\,; \boldsymbol{\beta})$ using a map from $\mathcal{Z}$ to the symmetric positive semi-definite rank r matrix manifold, i.e.,

$$\boldsymbol{\Sigma}(\cdot; \boldsymbol{\beta}) : \ \boldsymbol{z} \mapsto \boldsymbol{\Sigma}(\boldsymbol{z}; \boldsymbol{\beta}) = \mathbf{C}(\boldsymbol{z}; \boldsymbol{\beta}) \mathbf{C}(\boldsymbol{z}; \boldsymbol{\beta})^T. \tag{3.6}$$

In the above, $\mathbf{C}(\boldsymbol{z}; \boldsymbol{\beta}) \in \mathbb{R}^{w \times r}$ depends on the covariates $\boldsymbol{z}$ and the unknown coefficients $\boldsymbol{\beta}$. The structure (3.6) is similar to that of Cholesky decomposition, except that the matrix $\mathbf{C}(\boldsymbol{z}; \boldsymbol{\beta})$ is not required to be lower triangular.

With the help of (3.6), the construction of a positive semi-definite $\boldsymbol{\Sigma}(\boldsymbol{z}; \boldsymbol{\beta})$ is reduced to the construction of a general matrix $\mathbf{C}(\boldsymbol{z}; \boldsymbol{\beta})$ without restriction. We set $C_{ij} = \boldsymbol{v}(\boldsymbol{z})^T \boldsymbol{\beta}_{ij}$, where $\boldsymbol{v}(\boldsymbol{z}) \in \mathbb{R}^q$ are the values of an orthonormal B-spline basis evaluated at $\boldsymbol{z}$ and $\boldsymbol{\beta}_{ij} \in \mathbb{R}^q$ are the corresponding coefficients, for $i = 1, \ldots, w$ and $j = 1, \ldots, r$. In summary, our model for $\mathbf{C}(\boldsymbol{z}; \boldsymbol{\beta})$ (and hence $\boldsymbol{\Sigma}(\boldsymbol{z}; \boldsymbol{\beta})$) has $w \times r \times q$ unknown parameters, which are collected in a matrix

$\boldsymbol{\Gamma} \in \mathbb{R}^{(qw)\times r}$:

$$\boldsymbol{\Gamma} = \begin{pmatrix} \boldsymbol{\beta}_{11} & \boldsymbol{\beta}_{12} & \cdots & \boldsymbol{\beta}_{1r} \\ \boldsymbol{\beta}_{21} & \boldsymbol{\beta}_{22} & \cdots & \boldsymbol{\beta}_{2r} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{\beta}_{w1} & \boldsymbol{\beta}_{w2} & \cdots & \boldsymbol{\beta}_{wr} \end{pmatrix}. \tag{3.7}$$

Let $\boldsymbol{\beta} = \text{vec}(\boldsymbol{\Gamma})$ denote the vectorized version of this matrix. It readily follows that the factor matrix $\mathbf{C}(\boldsymbol{z};\boldsymbol{\beta})$ in (3.6) can be written as $\mathbf{C}(\boldsymbol{z};\boldsymbol{\beta}) = (\mathbf{I}_w \otimes \boldsymbol{v}(\boldsymbol{z})^T)\boldsymbol{\Gamma}$.

3.2.5 Model negative log-likelihood

Our model for y_n can now be written as

$$y_n(t,\boldsymbol{z}) \approx \underbrace{\mathbf{H}(t,\boldsymbol{z})^T\boldsymbol{\theta}_\mu + \boldsymbol{b}(t)^T\mathbf{C}(\boldsymbol{z};\boldsymbol{\beta})\boldsymbol{\psi}^{(n)}}_{x_n(t,\boldsymbol{z})} + \epsilon(t), \tag{3.8}$$

where $\boldsymbol{\psi}^{(n)} \sim \mathcal{N}_r(\mathbf{0}, \mathbf{I}_r)$, with $\mathbf{0}$ being the zero vector of length r, and $\epsilon(t) \sim \mathcal{N}(0, \sigma_e^2)$ is white noise. It follows that $y_n(t,\boldsymbol{z})$ is Gaussion distributed with mean $\mathbf{H}(t,\boldsymbol{z})\boldsymbol{\theta}_\mu$ and covariance function $\boldsymbol{b}(t)^T\boldsymbol{\Sigma}(\boldsymbol{z};\boldsymbol{\beta})\boldsymbol{b}(s) + \sigma_e^2$. The model (3.8) is a direct extension of the simpler SupSFPC approach, which can be recovered as a special case, details are given in Section B.4 of Appendix B.

To recover the eigenfunction in (3.3), at a fixed $\boldsymbol{z}$, we take eigen-decomposition of the matrix $\boldsymbol{\Sigma}(\boldsymbol{z};\boldsymbol{\beta}) = \mathbf{C}(\boldsymbol{z};\boldsymbol{\beta})\mathbf{C}(\boldsymbol{z};\boldsymbol{\beta})^T$ to get $\boldsymbol{\Sigma}(\boldsymbol{z};\boldsymbol{\beta}) = \boldsymbol{\Theta}_{\boldsymbol{z}}\mathbf{D}_{\boldsymbol{z}}\boldsymbol{\Theta}_{\boldsymbol{z}}^T$, where $\boldsymbol{\Theta}_{\boldsymbol{z}}$ is an orthonormal matrix of eigenvectors and $\mathbf{D}_{\boldsymbol{z}}$ is a diagonal matrix with decreasing eigenvalues. Suppose $\boldsymbol{\Theta}_{\boldsymbol{z},j}$ is the jth column of $\boldsymbol{\Theta}_{\boldsymbol{z}}$, then model (3.8) has the correspondence relation $f_j(t,\boldsymbol{z}) = \boldsymbol{b}^T(t)\boldsymbol{\Theta}_{\boldsymbol{z},j}$ (ignoring the spline approximation error) for the j-th eigenfunction at this specific value of $\boldsymbol{z}$. Moreover, when $\mathbf{C}(\boldsymbol{z};\boldsymbol{\beta})$ has full column rank, we can set $\mathbf{V}_{\boldsymbol{z}} = [\mathbf{C}(\boldsymbol{z};\boldsymbol{\beta})^T\mathbf{C}(\boldsymbol{z};\boldsymbol{\beta})]^{-1}\mathbf{C}(\boldsymbol{z};\boldsymbol{\beta})^T\boldsymbol{\Theta}_{\boldsymbol{z}}\mathbf{D}_{\boldsymbol{z}}^{1/2}$ and verify $\mathbf{V}_{\boldsymbol{z}}\mathbf{V}_{\boldsymbol{z}}^T = \mathbf{I}_r$. The matrix $\mathbf{V}$ establishes a connection between $\boldsymbol{\psi}^{(n)}$ in (3.8) and $\boldsymbol{\xi}^{(n)}$ in (3.3) as follows: it holds that $\boldsymbol{b}(t)^T\mathbf{C}(\boldsymbol{z};\boldsymbol{\beta})\mathbf{V}_{\boldsymbol{z}} = \boldsymbol{b}^T(t)\boldsymbol{\Theta}_{\boldsymbol{z}}\mathbf{D}_{\boldsymbol{z}}^{1/2} = \boldsymbol{f}^T(t,\boldsymbol{z})\mathbf{D}_{\boldsymbol{z}}^{1/2}$ and that $\boldsymbol{\psi}^{(n)} = \mathbf{V}_{\boldsymbol{z}}\mathbf{D}_{\boldsymbol{z}}^{-1/2}\boldsymbol{\xi}^{(n)}$.

In practice, y_n is only observed at a finite collection of observation times $\boldsymbol{t}_n = (t_1^{(n)}, \ldots, t_{m_n}^{(n)})$, and has a specific value of the covariates associated with it, which we denote by $\boldsymbol{z}_n$. To simplify

notation we collect the basis evaluations $\boldsymbol{b}(t_i^{(n)})$ for the covariance function (see (3.5)) and the tensor product basis evaluations $\mathbf{H}(t_i^{(n)}, \boldsymbol{z}_n)$ for the mean function (see (3.4)) into matrices $\mathbf{B}_n$ and $\mathbf{H}_n$, respectively, i.e.,

$$\mathbf{B}_n = (\boldsymbol{b}(t_1^{(n)}), \boldsymbol{b}(t_2^{(n)}), \ldots, \boldsymbol{b}(t_{m_n}^{(n)}))^T, \tag{3.9}$$

and

$$\mathbf{H}_n = (\mathbf{H}(t_1^{(n)}, \boldsymbol{z}_n), \mathbf{H}(t_2^{(n)}, \boldsymbol{z}_n), \ldots, \mathbf{H}(t_{m_n}^{(n)}, \boldsymbol{z}_n))^T. \tag{3.10}$$

Thus, the observations $\boldsymbol{y}_n = (y_n(t_1^{(n)}), \ldots, y_n(t_{m_n}^{(n)}))^T$ follow a multivariate Gaussian with mean $\mathbf{H}_n\boldsymbol{\theta}_\mu$ and covariance matrix $\boldsymbol{\Sigma}_n = \mathbf{B}_n\mathbf{C}_n\mathbf{C}_n^T\mathbf{B}_n^T + \sigma_e^2\mathbf{I}_{m_n}$, where $\mathbf{C}_n = \mathbf{C}(\boldsymbol{z}_n, \boldsymbol{\beta})$. With this notation, the negative log-likelihood of the full dataset is proportional to

$$\mathcal{L}(\boldsymbol{\theta}_\mu, \boldsymbol{\beta}, \sigma_e^2) := \sum_{n=1}^{N} \log\det \boldsymbol{\Sigma}_n + \operatorname{tr}(\mathbf{S}_n\boldsymbol{\Sigma}_n^{-1}), \tag{3.11}$$

where $\mathbf{S}_n = (\boldsymbol{y}_n - \mathbf{H}_n\boldsymbol{\theta}_\mu)(\boldsymbol{y}_n - \mathbf{H}_n\boldsymbol{\theta}_\mu)^T$.

Suppose that $\hat{\boldsymbol{\theta}}_\mu$ and $\hat{\boldsymbol{\beta}}$ are the maximum likelihood estimates (MLEs) of the parameters obtained by minimizing (3.11). For a given value of $\boldsymbol{z}$, we can estimate the j-th FPC by $\hat{f}_j(t, \boldsymbol{z}) = \boldsymbol{b}(t)^T\hat{\boldsymbol{v}}_j$, where $\hat{\boldsymbol{v}}_j$ denotes the eigenvector corresponding to the j-th largest eigenvalue obtained from the eigen-decomposition of $\boldsymbol{\Sigma}(\boldsymbol{z}; \hat{\boldsymbol{\beta}}) = \mathbf{C}(\boldsymbol{z}; \hat{\boldsymbol{\beta}})\mathbf{C}(\boldsymbol{z}; \hat{\boldsymbol{\beta}})^T$. Note the eigenvectors $\hat{\boldsymbol{v}}_j$ implicitly depend on the covariates $\boldsymbol{z}$ through the decomposition of $\boldsymbol{\Sigma}(\boldsymbol{z}; \hat{\boldsymbol{\beta}})$. In addition, for a fixed $\boldsymbol{z}$, the estimated eigenfunctions are orthonormal to each other, because the eigenvectors ($\hat{\boldsymbol{v}}_j$'s) are orthonormal to each other and $\boldsymbol{b}(\cdot)$ is an orthonormal basis.

To encourage smoothness of the estimated mean and covariance functions, we follow the approach of roughness penalty of Wood (2006); Reiss et al. (2014). The details of the roughness penalty construction are given in Section B.1 of Appendix B. Combining the negative log-likelihood (3.11) with the roughness penalties given by (B.8) and (B.9) in Appendix B, we obtain

the following objective function to be minimized:

$$\mathcal{L} + \mathcal{P} = \sum_{n=1}^{N} \{\log\det \boldsymbol{\Sigma}_n + \mathrm{tr}(\mathbf{S}_n \boldsymbol{\Sigma}_n^{-1})\} + \boldsymbol{\theta}_\mu^T(\lambda_t^{(\mu)}\widetilde{\mathbf{S}}_t^{(\mu)} + \lambda_{\boldsymbol{z}}^{(\mu)}\widetilde{\mathbf{S}}_{\boldsymbol{z}}^{(\mu)})\boldsymbol{\theta}_\mu + \boldsymbol{\beta}^T(\lambda_t \mathbf{I}_r \otimes \widetilde{\mathbf{S}}_t + \lambda_{\boldsymbol{z}}\mathbf{I}_r \otimes \widetilde{\mathbf{S}}_{\boldsymbol{z}})\boldsymbol{\beta}, \tag{3.12}$$

where $\mathcal{L}$ and $\mathcal{P}$ denote the log-likelihood and penalty terms, respectively. We use cross-validation to choose the four tuning parameters $\lambda_t, \lambda_{\boldsymbol{z}}, \lambda_t^{(\mu)}$, and $\lambda_{\boldsymbol{z}}^{(\mu)}$. The next section presents an algorithm for minimizing (3.12).

3.3 Algorithm

3.3.1 Model training

Given noisy observations of a collection of latent functions, we can estimate the mean and covariance functions by optimizing (3.12). However, optimization is challenging because the objective function (3.12) is non-convex. Moreover, evaluating the log-likelihood $\mathcal{L}$ and its gradients involves computing the inverses of $\boldsymbol{\Sigma}_n \in \mathbb{R}^{m_n \times m_n}$, for $n = 1, \ldots, N$, which has a combined computational cost of $\mathcal{O}(\sum_n m_n^3)$. In practice, this latter issue is exacerbated by the need to perform exploratory evaluations of the objective function to select a good step size, i.e., the tuning parameter that controls the magnitude of changes in the parameters in each iteration of the optimization algorithm. We overcome these problems by proposing an initialization algorithm which identifies good initial parameter values, and by developing efficient ways to evaluate the objective function (3.12) and its gradient. Our gradient descent based optimization strategy is given in Algorithm 5 below. The algorithm iteratively updates the parameters $\boldsymbol{\theta}_\mu$, $\boldsymbol{\beta}$, and σ_e^2 until convergence.

In what follows, we repeatedly apply the matrix determinant lemma and Sherman-Morrison-Woodbury formula to reduce the per curve cost of computing the matrix inverse and log-determinant in Steps 4-6 of Algorithm 1 from $\mathcal{O}(m_n^3)$ to $\mathcal{O}(r^3)$. Initialization of $\boldsymbol{\beta}$ (Step 1) is discussed at the end of this subsection. Proofs for all the results below are given in Section B.2 of Appendix B. We begin with Lemma 4 which presents a more efficient expression for the log-likelihood $\mathcal{L}$ appearing

Algorithm 5 Modified gradient descent for optimizing CD-FPCA objective (3.12)

1: Initialize $\boldsymbol{\beta}$ using Algorithm 6 (below);
2: Initialize $\boldsymbol{\theta}_\mu$ to be the zero vector, and σ_e^2 to be an appropriate small positive value;
3: **repeat**
4: Update $\boldsymbol{\theta}_\mu$ by gradient descent until convergence, gradient is sum of (3.16) and (3.22);
5: Update $\boldsymbol{\beta}$ by gradient descent until convergence, gradient is sum of (3.19) and (3.23);
6: Update σ_e^2 by gradient descent until convergence, gradient is given by (3.18);
7: **until** convergence

in (3.12) (also see (3.11)).

Lemma 4. *The loss function $\mathcal{L}$ given by (3.11) is equal to*

$$2\sum_{n=1}^{N}\log\det(\mathbf{F}_n) - \sigma_e^{-4}\sum_{n=1}^{N}\|\boldsymbol{h}_n\|_2^2 + \sigma_e^{-2}\sum_{n=1}^{N}\|\boldsymbol{y}_n - \mathbf{H}_n\boldsymbol{\theta}_\mu\|_2^2 + \sum_{n=1}^{N} m_n \log\sigma_e^2. \tag{3.13}$$

where $\mathbf{F}_n$ is the Cholesky factor of $\mathbf{I}_r + \sigma_e^{-2}\mathbf{W}_n$ (i.e., $\mathbf{F}_n\mathbf{F}_n^T = \mathbf{I}_r + \sigma_e^{-2}\mathbf{W}_n$), with $\mathbf{W}_n = \mathbf{C}_n^T\mathbf{B}_n^T\mathbf{B}_n\mathbf{C}_n$, and $\boldsymbol{h}_n = \mathbf{F}_n^{-1}\mathbf{C}_n^T\mathbf{B}_n^T(\boldsymbol{y}_n - \mathbf{H}_n\boldsymbol{\theta}_\mu)$.

Next, it is straightforward to verify that the gradients of the log-likelihood (3.11) with respect to $\boldsymbol{\theta}_\mu$ and σ_e^2 are

$$\frac{\partial\mathcal{L}}{\partial\boldsymbol{\theta}_\mu} = \sum_{n=1}^{N} 2\mathbf{H}_n^T\boldsymbol{\Sigma}_n^{-1}(\mathbf{H}_n\boldsymbol{\theta}_\mu - \boldsymbol{y}_n), \tag{3.14}$$

and

$$\frac{\partial\mathcal{L}}{\partial\sigma_e^2} = \sum_{n=1}^{N}\mathrm{tr}(\boldsymbol{\Sigma}_n^{-1}) - \sum_{n=1}^{N}(\boldsymbol{y}_n - \mathbf{H}_n\boldsymbol{\theta}_\mu)^T\boldsymbol{\Sigma}_n^{-2}(\boldsymbol{y}_n - \mathbf{H}_n\boldsymbol{\theta}_\mu), \tag{3.15}$$

respectively. Following a similar approach as for Lemma 4, these gradients can be expressed in a computationally more efficient way.

Lemma 5. *Let $\mathbf{F}_n$ and $\boldsymbol{h}_n$ be defined as in Lemma 4 above. The gradient $\frac{\partial\mathcal{L}}{\partial\boldsymbol{\theta}_\mu}$ given by (3.14) can be expressed as*

$$\frac{\partial\mathcal{L}}{\partial\boldsymbol{\theta}_\mu} = \sum_{n=1}^{N} -2\sigma_e^{-2}(\mathbf{H}_n^T\boldsymbol{y}_n - \mathbf{H}_n^T\mathbf{H}_n\boldsymbol{\theta}_\mu) + 2\sigma_e^{-4}\mathbf{H}_n^T\mathbf{E}_n^T\boldsymbol{h}_n, \tag{3.16}$$

where $\mathbf{E}_n = \mathbf{F}_n^{-1}\mathbf{C}_n^T\mathbf{B}_n^T$.

Lemma 6. *The gradient* $\frac{\partial \mathcal{L}}{\partial \sigma_e^2}$ *given by (3.15) can be expressed as*

$$\frac{\partial \mathcal{L}}{\partial \sigma_e^2} = \sum_{n=1}^{N} \Big[m_n \sigma_e^{-2} - \sigma_e^{-4} \mathrm{tr}(\mathbf{E}_n^T \mathbf{E}_n) - \sigma_e^{-4} \|\boldsymbol{y}_n - \mathbf{H}_n \boldsymbol{\theta}_\mu\|_2^2 + 2\sigma_e^{-6} \|\boldsymbol{h}_n\|_2^2 \tag{3.17}$$

$$- \sigma_e^{-8} \boldsymbol{h}_n^T \mathbf{E}_n \mathbf{E}_n^T \boldsymbol{h}_n \Big], \tag{3.18}$$

where $\boldsymbol{h}_n$ *and* $\mathbf{E}_n$ *are as given in Lemma 4 and Lemma 5 above.*

We now consider the gradient of $\mathcal{L}$ with respect to $\boldsymbol{\beta}$. Let β_{ijk} denote the k-th element of the vector $\boldsymbol{\beta}_{ij}$. We have

$$\frac{\partial \mathcal{L}}{\partial \beta_{ijk}} = \sum_{n=1}^{N} \Big\langle \frac{\partial \mathcal{L}}{\partial \mathbf{C}_n}, \frac{\partial \mathbf{C}_n}{\partial \beta_{ijk}} \Big\rangle, \tag{3.19}$$

where the inner product is defined as $\langle \mathbf{A}, \mathbf{B} \rangle = \mathrm{tr}(\mathbf{A}^T\mathbf{B})$, $\frac{\partial \mathbf{C}_n}{\partial \beta_{ijk}}$ is the matrix of zeros except that its (i, j) element is $v_k(\boldsymbol{z})$, and

$$\frac{\partial \mathcal{L}}{\partial \mathbf{C}_n} = 2 \times \mathbf{B}_n^T \big[\boldsymbol{\Sigma}_n^{-1} - \boldsymbol{\Sigma}_n^{-1} \mathbf{S}_n \boldsymbol{\Sigma}_n^{-1} \big] \mathbf{B}_n \mathbf{C}_n. \tag{3.20}$$

Lemma 7 below gives a more computationally efficient expression for (3.20).

Lemma 7. *The gradient* $\frac{\partial \mathcal{L}}{\partial \mathbf{C}_n}$ *given by (3.20) can be expressed as*

$$\frac{\partial \mathcal{L}}{\partial \mathbf{C}_n} = 2\sigma_e^{-2}(\mathbf{B}_n^T \mathbf{B}_n \mathbf{C}_n - \mathbf{B}_n^T \mathbf{K}_n \mathbf{W}_n)$$

$$- 2\sigma_e^{-4}(\mathbf{B}_n^T - \mathbf{B}_n^T \mathbf{K}_n \mathbf{C}_n^T \mathbf{B}_n^T) \mathbf{S}_n (\mathbf{B}_n \mathbf{C}_n - \mathbf{K}_n \mathbf{W}_n), \tag{3.21}$$

where $\mathbf{W}_n = \mathbf{C}_n^T \mathbf{B}_n^T \mathbf{B}_n \mathbf{C}_n$ *and* $\mathbf{K}_n = \mathbf{B}_n \mathbf{C}_n \{\sigma_e^2 \mathbf{I}_r + \mathbf{W}_n\}^{-1}$.

Lastly, the gradients of the penalty term $\mathcal{P}$ in (3.12) with respect to $\boldsymbol{\theta}_\mu$ and $\boldsymbol{\beta}$ are

$$\frac{\partial \mathcal{P}}{\partial \boldsymbol{\theta}_\mu} = \lambda_t^{(\mu)} (\widetilde{\mathbf{S}}_t^{(\mu)} + \widetilde{\mathbf{S}}_t^{(\mu)T}) \boldsymbol{\theta}_\mu + \lambda_{\boldsymbol{z}}^{(\mu)} (\widetilde{\mathbf{S}}_{\boldsymbol{z}}^{(\mu)} + \widetilde{\mathbf{S}}_{\boldsymbol{z}}^{(\mu)T}) \boldsymbol{\theta}_\mu \tag{3.22}$$

and

$$\frac{\partial \mathcal{P}}{\partial \boldsymbol{\beta}} = \lambda_t[(\mathbf{I}_r \otimes \widetilde{\mathbf{S}}_t) + (\mathbf{I}_r \otimes \widetilde{\mathbf{S}}_t)^T]\boldsymbol{\beta} + \lambda_{\boldsymbol{z}}[(\mathbf{I}_r \otimes \widetilde{\mathbf{S}}_{\boldsymbol{z}}) + (\mathbf{I}_r \otimes \widetilde{\mathbf{S}}_{\boldsymbol{z}})^T]\boldsymbol{\beta}, \tag{3.23}$$

respectively. The gradients of the objective function (3.12) with respect to $\boldsymbol{\theta}_\mu$ and $\boldsymbol{\beta}$ are obtained by summing (3.14) and (3.22) and summing (3.19) and (3.23), respectively.

It remains to specify a procedure to initialize $\boldsymbol{\beta}$ in Step 1 of Algorithm 5, and our approach is summarized in Algorithm 6 below. Step 1 of Algorithm 6 divides the covariate domain into small bins (or regions) $\mathcal{Z}_1, \ldots, \mathcal{Z}_U$, and treats the observations in each bin as having a fixed covariates vector $\boldsymbol{z}_u$, for $u = 1 \ldots, U$. The covariate vectors $\boldsymbol{z}_u$, for $u = 1 \ldots, U$, are set to be the mean observed covariate vector in each bin, i.e., $\boldsymbol{z}_u = n_u^{-1} \sum_{n:\boldsymbol{z}_n \in \mathcal{Z}_u} \boldsymbol{z}_n$, where $n_u = |\mathcal{Z}_u|$ is the cardinality of the set $\mathcal{Z}_u$. In Step 2, for each bin, we fit the classical FPCA model introduced in (3.2) to the subset of functional observations falling in that particular bin, i.e., we fit it separately to $\{(\boldsymbol{t}_n, \boldsymbol{y}_n) : \boldsymbol{z}_n \in \mathcal{Z}_u\}$, for each $u \in \{1, \ldots, U\}$. Thus, we obtain an estimated covariance function $\hat{G}(t, s|\boldsymbol{z}_u) = \boldsymbol{b}(t)^T \widehat{\boldsymbol{\Sigma}}_{\boldsymbol{z}_u} \boldsymbol{b}(s)$ for each fixed $\boldsymbol{z}_u$. Finally, making use of (3.6), Step 3 initializes $\boldsymbol{\beta}$ by

$$\boldsymbol{\beta}^{(0)} := \arg\min_{\boldsymbol{\beta}} \sum_{u=1}^{U} \|\widehat{\boldsymbol{\Sigma}}_{\boldsymbol{z}_u}^{1/2} - \mathbf{C}(\boldsymbol{z}_u; \boldsymbol{\beta})\|_F^2, \tag{3.24}$$

where the subscript "F" denotes the Frobenius norm.

Algorithm 6 Initialization of $\boldsymbol{\beta}$

1: Divide the covariates domain into small bins (or regions) $\mathcal{Z}_1, \ldots, \mathcal{Z}_U$.
2: Fit the classical FPCA model (3.2) in each bin to obtain $\widehat{\boldsymbol{\Sigma}}_{\boldsymbol{z}_u}$, for $u = 1, \ldots, U$.
3: Initialize $\boldsymbol{\beta}$ by $\boldsymbol{\beta}^{(0)}$ in (3.24).

3.3.2 Prediction

Suppose that we have applied Algorithm 5 to a training dataset, and now obtain noisy observations $\boldsymbol{y}_* = (y_*(t_1^{(*)}), \ldots, y_*(t_{m_*}^{(*)}))^T$ of a new latent function $x_*(t)$ at the time points $t_1^{(*)}, \ldots, t_{m_*}^{(*)}$, together with a corresponding covariate vector $\boldsymbol{z}_*$. Suppose further that we want to estimate the

scores $\boldsymbol{\xi}^{(*)} = (\xi_1^{(*)}, \ldots, \xi_r^{(*)})^T$ for the function $x_*(t)$ (see (3.3)), and thereby predict the value of a new observation $y_*(t)$ at the time point $t \in \mathcal{T}$.

Following Yao et al. (2005), our approach to this prediction task is motivated by the empirical Bayes perspective. Let $\hat{\boldsymbol{\theta}}_\mu$, $\hat{\boldsymbol{\beta}}$ and $\hat{\sigma}_e^2$ denote the estimates of the model parameters (see (3.8)) obtained by applying Algorithm 5 to the training dataset. For the new observations, we define basis matrices $\mathbf{B}_* = (\boldsymbol{b}(t_1^{(*)}), \ldots, \boldsymbol{b}(t_{m^*}^{(*)}))^T$ and $\mathbf{H}_* = (\mathbf{H}(t_1^{(*)}, \boldsymbol{z}_*), \ldots, \mathbf{H}(t_{m^*}^{(*)}, \boldsymbol{z}_*))^T$, analogous to (3.9) and (3.10), respectively. Next, we compute the eigendecomposition $\boldsymbol{\Sigma}(\boldsymbol{z}_*; \hat{\boldsymbol{\beta}}) = \boldsymbol{\Theta}_* \mathbf{D}_* \boldsymbol{\Theta}_*^T$. Treating the parameter estimates $\hat{\boldsymbol{\theta}}_\mu$, $\hat{\boldsymbol{\beta}}$ and $\hat{\sigma}_e^2$ as if they were the the true values (i.e., the plug-in approach), the joint distribution of $\boldsymbol{y}_*$ and $\boldsymbol{\xi}^{(*)}$ is

$$\begin{pmatrix} \boldsymbol{y}_* \\ \boldsymbol{\xi}^{(*)} \end{pmatrix} \sim \mathcal{N}_{2m^*} \left(\begin{pmatrix} \mathbf{H}_* \hat{\boldsymbol{\theta}}_\mu \\ \mathbf{0} \end{pmatrix}, \begin{pmatrix} \mathbf{B}_* \boldsymbol{\Sigma}(\boldsymbol{z}_*; \hat{\boldsymbol{\beta}}) \mathbf{B}_*^T + \hat{\sigma}_e^2 \mathbf{I} & \mathbf{B}_* \boldsymbol{\Theta}_* \mathbf{D}_* \\ \mathbf{D}_* \boldsymbol{\Theta}_*^T \mathbf{B}_*^T & \mathbf{D}_* \end{pmatrix} \right). \tag{3.25}$$

This joint distribution results from assuming the prior $\boldsymbol{\xi}^{(*)} \sim \mathcal{N}(\mathbf{0}, \mathbf{D}_*)$, which is derived from the training data (and $\boldsymbol{z}^*$), hence the empirical Bayes connection. In a more complete empirical Bayes treatment $\boldsymbol{\theta}_\mu$, $\boldsymbol{\beta}$ and σ_e^2 would also be assigned priors, but this introduces additional complications and computation and is therefore avoided here. Based on (3.25), the posterior distribution of $\boldsymbol{\xi}^{(*)}$ is again a multivariate Gaussian whose mean and covariance matrix are given by

$$\mathbb{E}(\boldsymbol{\xi}^{(*)} | \boldsymbol{y}_*) = \mathbf{D}_* \boldsymbol{\Theta}_*^T \mathbf{B}_*^T (\mathbf{B}_* \boldsymbol{\Sigma}(\boldsymbol{z}_*; \hat{\boldsymbol{\beta}}) \mathbf{B}_*^T + \hat{\sigma}_e^2 \mathbf{I})^{-1} (\boldsymbol{y}_* - \mathbf{H}_* \hat{\boldsymbol{\theta}}_\mu) \tag{3.26}$$

and

$$\mathrm{Cov}(\boldsymbol{\xi}^{(*)} | \boldsymbol{y}_*) = \mathbf{D}_* - \mathbf{D}_* \boldsymbol{\Theta}_*^T \mathbf{B}_*^T (\mathbf{B}_* \boldsymbol{\Sigma}(\boldsymbol{z}_*; \hat{\boldsymbol{\beta}}) \mathbf{B}_*^T + \hat{\sigma}_e^2 \mathbf{I})^{-1} \mathbf{B}_* \boldsymbol{\Theta}_* \mathbf{D}_*, \tag{3.27}$$

respectively.

Finally, combining (3.26) and (3.27) with (3.8), the posterior predictive distribution of $y_*(t)$ at a new time t is a univariate Gaussian distribution with mean and variance given by

$$\mathbf{H}(t, \boldsymbol{z}_*) \hat{\boldsymbol{\theta}}_\mu + \boldsymbol{b}(t)^T \boldsymbol{\Theta}_* \mathbb{E}(\boldsymbol{\xi}^{(*)} | \boldsymbol{y}_*) \tag{3.28}$$

and

$$\boldsymbol{b}(t)^T \boldsymbol{\Theta}_* \mathrm{Cov}(\boldsymbol{\xi}^{(*)} | \boldsymbol{y}_*) \boldsymbol{\Theta}_*^T \boldsymbol{b}(t) + \hat{\sigma}_e^2, \tag{3.29}$$

respectively. If we are instead interested in the underlying latent function value $x_*(t)$, then the posterior predictive distribution will be the same except that the expression for the variance will not have the $\hat{\sigma}_e^2$ term. Sometimes, such as in astronomy, measurement errors are provided with each observed value of $y_*(t)$. In this case, we modify our predictions by replacing all instances of $\hat{\sigma}_e^2$ above by the actual measurement error value (including in the application of Algorithm 5 to the training data).

3.4 Simulation Study

We now compare our CD-FPCA approach (Algorithm 5) with the methods proposed by James et al. (2000), Jiang and Wang (2010), and Li et al. (2016). The James et al. (2000) method uses a spline basis approach to approximate the classical FPCA model (3.2), but does not incorporate covariates. Following Jiang and Wang (2010), we denote this approach by rFPCA, where the "r" stands for reduced rank. Jiang and Wang (2010) proposed two local linear smoother based methods, which do incorporate covariate information, but with high computational cost. Their methods are called fully adjusted FPCA (fFPCA) and mean adjusted FPCA (mFPCA); the former allows both the mean and covariance function to depend on covariates, whereas the latter only allows the mean function to do so. The supervised sparse and functional principal component (SupSFPC) method proposed by Li et al. (2016) allows the scores $\boldsymbol{\xi}^{(n)}$ in (3.2) (but not the mean function) to vary with the covariates, and is computationally more efficient than all the other methods considered here (including ours). It is a state-of-the-art approach, and therefore a key comparison. Nonetheless, it has a number of limitations including the linear assumption and a requirement that the data have balanced sampling and regular spacing.

3.4.1 Simulated datasets

We simulate two datasets of noisy realizations of $N = 100$ and $N = 7500$ latent functions, respectively. In both datasets, the n-th latent function $x_n(t, z)$ is a linear combination of a mean

function $\mu(t,z)$ and $r=3$ orthonormal eigenfunctions $f_j(t,z)$, $j=1,\ldots,r$. We set the covariate z to be univariate, the mean function to be $\mu(t,z)=30(t-z)^2$, and the three eigenfunctions to be $f_1(t,z)=\sqrt{2}\cos(\pi(t+z))$, $f_2(t,z)=\sqrt{2}\sin(\pi(t+z))$ and $f_3(t,z)=\sqrt{2}\cos(3\pi(t-z))$. To further impose dependence of the covariance structure on the covariate z we set the eigenvalues to be $\boldsymbol{d}=\big(2(z+20),z+10,z\big)$. The scores $\boldsymbol{\xi}^{(n)}$ are sampled from a Gaussian distribution with mean $\mathbf{0}$ and covariance matrix $\mathbf{D}_z=\mathrm{diag}(\boldsymbol{d})$, see (3.3).

For easy comparison, all data generated in this section lie on a regular grid, i.e., the time points $t_i=\frac{i-1}{m-1}$ for $i=1,\ldots,m=100$. This accommodates the SupSFPC method which cannot handle irregularly spaced functional data. The final simulated dataset contains m noisy realizations of each latent function x_n, i.e.,

$$y_n(t_i,z)=x_n(t_i,z)+\epsilon(t_i)=\mu(t_i\upsilon z)+\sum_{j=1}^{r}\xi_j^{(n)}f_j(t_i,z)+\epsilon(t_i) \tag{3.30}$$

$$=\mu(t_i,z)+\boldsymbol{f}^T(t_i,z)\boldsymbol{\xi}^{(n)}+\epsilon(t_i), \tag{3.31}$$

for $i=1,\ldots,m$, where ϵ is an independent white noise process with variance $\sigma_e^2=0.1$. We repeat the simulation 10 times.

3.4.2 Results

For our CD-FPCA method, we set the number of B-spline basis functions for capturing the dependence of the mean function on t and z to be to 10 and 5, respectively, i.e., $l=10$ and $p=5$, see (3.4). For the covariance function, we set the number of basis functions for capturing dependence on t and z to be 10 and 7, respectively, i.e., $w=10$ and $q=7$, see (3.5)-(3.8). Similarly, for rFPCA we again use two sets of 10 basis functions to capture the dependence of the mean and covariance function on t (the James et al. (2000) model does not incorporate covariates). All the splines are cubic, and the spline basis for capturing the temporal evolution of the covariance function $\boldsymbol{b}(t)\in\mathbb{R}^w$ is orthonormal, as mentioned in Section 3.2.4. In the case of the SupSFPC method, we use the average of all the observed curves as the mean function estimator, because

Li et al. (2016) do not specify an estimator. For the mFPCA and fFPCA methods (Jiang and Wang, 2010) we apply 10-fold cross-validation to the $N = 100$ dataset to select the smoothing bandwidths (cross-validation is too time-consuming to apply to the $N = 7500$ dataset so we use the same values for that dataset as well). In particular, to estimate the mean function we use the bandwidths 0.57 and 0.52 to smooth across time t and the covariate z, respectively. To estimate the eigenfunctions we use the bandwidths 0.57 and 0.66 to smooth across time t and the covariate z (fFPCA only), respectively.

Table 3.1: MSE of the mean and eigenfunction estimators under CD-FPCA, SupSFPC, fFPCA, mFPCA, and rFPCA.

		MSE (SD)			
		Mean Fun.	First Eigen.	Second Eigen.	Third Eigen.
$N = 100$	CD-FPCA	5.06 (1.86)	0.261 (0.097)	0.283 (0.106)	0.065 (0.050)
	SupSFPC	6.20 (1.34)	0.762 (0.069)	0.774 (0.069)	0.864 (0.068)
	fFPCA	5.79 (1.47)	0.266 (0.235)	0.305 (0.229)	1.857 (0.015)
	mFPCA	6.17(1.52)	0.745(0.060)	0.748(0.061)	1.866(0.014)
	rFPCA	30.12 (2.07)	0.746 (0.068)	0.771 (0.069)	0.867 (0.071)
$N = 7500$	CD-FPCA	0.14 (0.09)	0.001 (0.001)	0.001 (0.001)	0.002 (0.000)
	SupSFPC	5.14 (0.06)	0.723 (0.007)	0.736 (0.007)	0.872 (0.009)
	fFPCA	—	—	—	—
	mFPCA	4.35 (0.09)	0.736 (0.010)	0.740 (0.010)	1.879 (0.002)
	rFPCA	30.96 (0.32)	0.710 (0.007)	0.735 (0.007)	0.880 (0.009)

Table 3.1 gives the mean squared errors (MSE) of the estimators for the mean function and eigenfunctions under each of the five methods considered: CD-FPCA, SupSFPC, fFPCA, mFPCA, and rFPCA. For a function g and an estimate $\hat{g}$, we define the squared error to be

$$\frac{1}{Nm}\sum_{n=1}^{N}\sum_{i=1}^{m}(g(t_i, z_n) - \hat{g}(t_i, z_n))^2, \tag{3.32}$$

where m is the number of grid points at which each function is observed. We approximate the MSE by the mean of the squared error across the 10 replicate simulations. Table 3.1 also shows the standard deviation (SD) of the MSE across the 10 simulations. Our CD-FPCA method has lower

MSE than all the other approaches for all three eigenfunctions and the mean function. The second best method in terms of MSE is fFPCA, but this approach has prohibitively high computational cost, as we now illustrate. Figure 3.1 compares the computational cost of the five methods

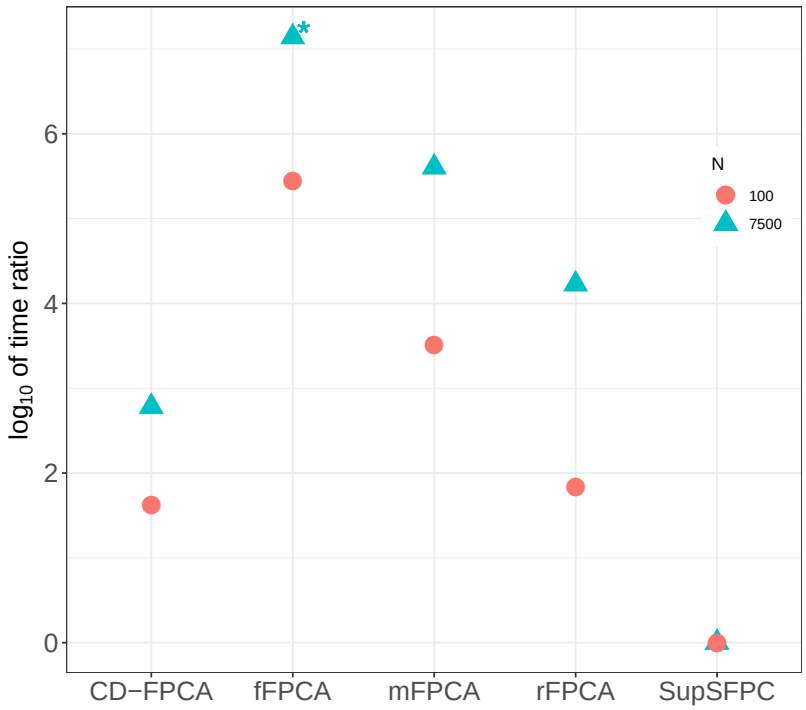

Figure 3.1: Log relative run time $\log_{10}(T_E/T_{\text{SupSFPC}})$, where T_E denotes the mean run time in seconds for $E \in \{\text{CD-FPCA}, \text{fFPCA}, \text{mFPCA}, \text{rFPCA}, \text{SupSFPC}\}$. The round points and triangle points represent different datasets, i.e., $N = 100$ and $N = 7500$, separately

and in particular shows $\log_{10}(T_E/T_{\text{SupSFPC}})$, where T_E denotes the mean run time in seconds for $E \in \{\text{CD-FPCA}, \text{SupSFPC}, \text{fFPCA}, \text{mFPCA}, \text{rFPCA}\}$. SupSFPC is used as the baseline because it is the fastest method. For reference, SupSFPC took 0.084 and 0.294 seconds for the $N = 100$ and $N = 7500$ cases, respectively. Our method is computationally more efficient than all the other methods except SupSFPC. Despite its speed, the SupSFPC approach has limitations, because it performs substantially worse than CD-FPCA (and fFPCA) in terms of MSE, e.g., Table 3.1 shows that under SupSFPC the MSE for the first eigenfunction is almost three times higher than under

CD-FPCA. Regarding the "$\star$" symbol in Figure 3.1, note that the fFPCA algorithm was not run for the case $N = 7500$ because it would have taken more than 1000 hours to complete. The time given in Figure 3.1 for this case (marked by "$\star$") was approximated by fitting a linear model to the run times for smaller values of N and extrapolating to $N = 7500$ (an optimistic estimate). The other local smoother based approach mFPCA was also computatiaonlly inefficient, although less so. The high accuracy and comparatively low computational cost of our method means that it is more suitable for application to large datasets than the other approaches.

3.4.3 Further comparison with SupSFPC

Here we more closely compare our CD-FPCA approach with SupSFPC because the latter is the only other method considered here which can be applied to large datasets and also incorporates covariate information. Firstly, to confirm the findings in Table 3.1, Figure 3.2 shows the estimated mean function and eigenfunctions under CD-FPCA (dot-dash lines) and SupSFPC (oragne dotted lines) as well as the true eigenfunctions (solid lines) given in Section 3.4.1, for a range of values of the covariate z. For each function (i.e., each large panel), the SupSPFC estimates (dotted lines) are all the same because the it is only the scores ($\boldsymbol{\xi}^{(n)}$ in (3.2)) that vary with the covariates under the SupSPFC method. The standard deviations associated with the estimated functions are all very small and are not plotted. In summary, Figure 3.2 shows that CD-FPCA recovers the true mean function and the three eigenfunctions almost exactly, whereas SupSPFC does not model any of the variation in these functions across covariate values.

Next, we compare the prediction accuracy of the two methods. We treat the $N = 7500$ dataset introduced in Section 3.4.1 as the training data, and generate noisy observations of an additional $N = 7500$ latent functions to serve as a test dataset. We fix the mean function and eigenfunctions using the training set introduced in Section 3.4.1. Using the fixed parameter estimates obtained in Section 3.4.2, we apply the method described in Section 3.3.2 to make predictions for a selection of the observations in the test dataset. In particular, for each function in the test dataset, we select first 20% of the time points to be observed ($\boldsymbol{y}_*$ in (3.26)), and make predictions for the remaining 80% by applying (3.28) for CD-FPCA and (B.47) in Section B.3 of Appendix B for SupSFPC. Some

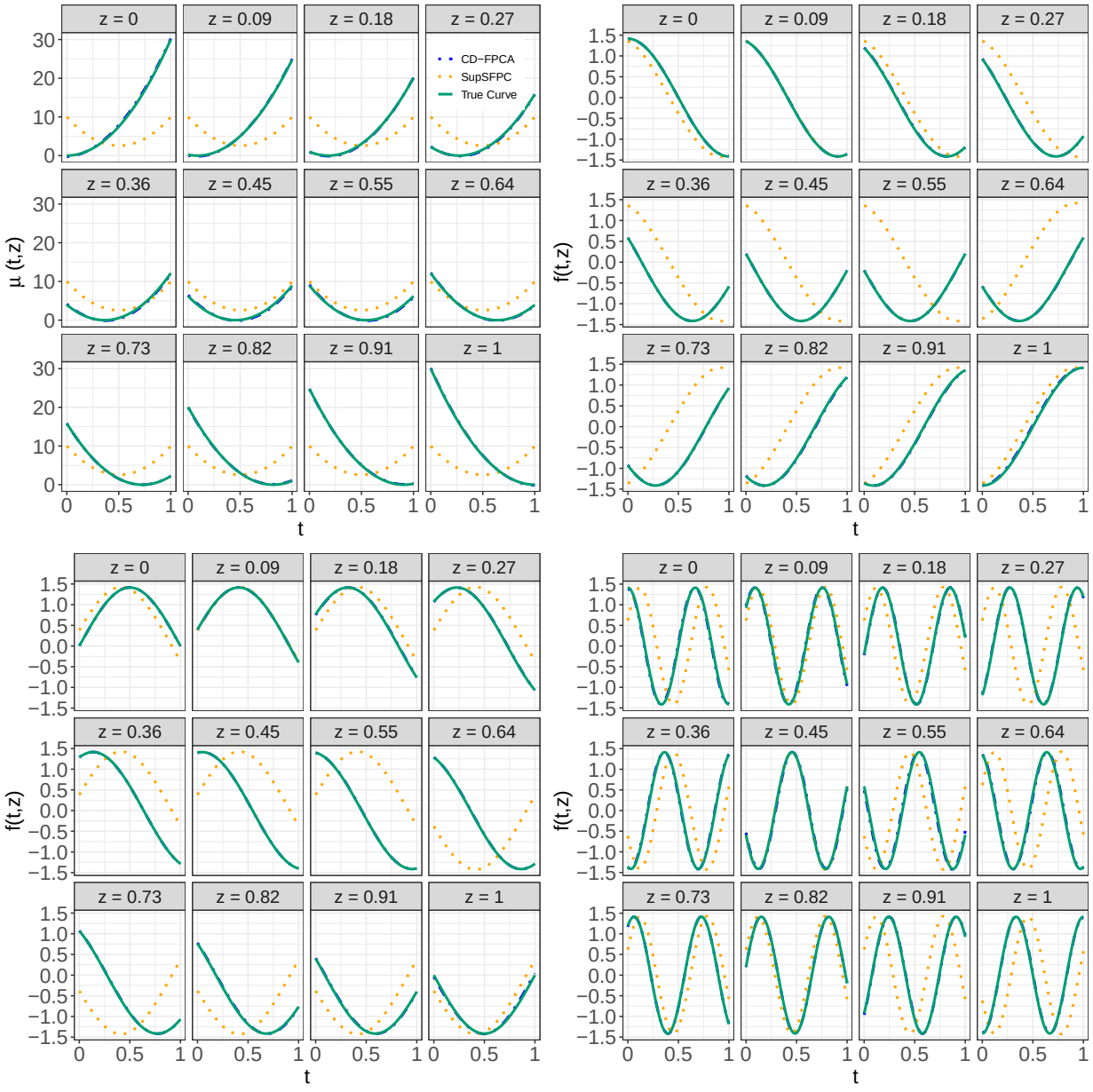

Figure 3.2: (Top left) estimates of the mean function under the CD-FPCA (dot-dash lines) and SupSFPC (dotted lines) methods, for a range of covariate values. (Top right, bottom left, bottom right) estimates of eigenfunctions 1, 2, and 3, respectively, under the CD-FPCA (dot-dash lines) and SupSFPC (dotted lines) methods, for a range of covariate values. The true functions are shown as solid lines

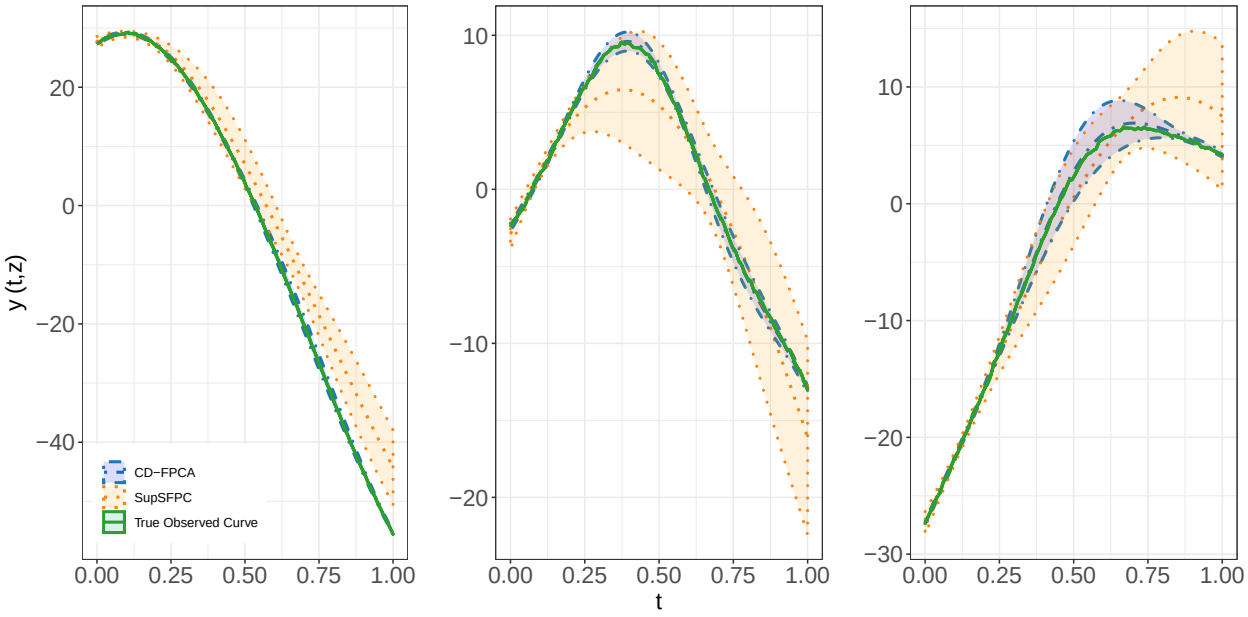

Figure 3.3: Example predictions and 95% predictive intervals under CD-FPCA (center dot-dash lines) and SupSFPC (center dotted lines). From left to right the panels correspond to covariate values 0.03, 0.41 and 0.87, respectively. The solid lines show the true observed curves

example predictions and 95% predictive intervals are shown in Figure 3.3. The CD-FPCA and SupSFPC predictive intervals were computed using (3.29) and (B.48) (Section B.3 of Appendix B), respectively.

To quantitatively compare the predictions we compute the mean squared fitting errors (MSFE) for the test set, i.e.,

$$\frac{1}{\widetilde{N}\widetilde{m}} \sum_{n=1}^{\widetilde{N}} \sum_{i=1}^{\widetilde{m}} \{y_n(\tilde{t}_i, \tilde{z}_n) - \hat{\mu}(\tilde{t}_i, \tilde{z}_n) - \sum_{j=1}^{r} \hat{\xi}_j^{(n)} \hat{f}_j(\tilde{t}_i, \tilde{z}_n)\}^2, \tag{3.33}$$

where $\widetilde{N}$ is the number of observed curves in the test dataset, $\tilde{t}_1 \ldots, \tilde{t}_{\widetilde{m}}$ are the times points at which predictions are made, $\tilde{z}_n$ is the covariate associated with curve n of the test dataset, and $\hat{\xi}_j^{(n)}$ is the estimate of the score for eigenfunction j and curve n (based on the first 20% of the observations for that curve). The mean function and eigenfunction estimates $\hat{\mu}$ and $\hat{f}_j$ are those from Section 3.4.1, and in the case of CD-FPCA they depend on both t and z, but in the case of

Table 3.2: Test set MSFE (3.33) under CD-FPCA and SupSFPC.

	CD-FPCA	SupSFPC
	MSFE	MSFE
r = 1	52.77 (5.03)	60.32 (0.90)
r = 2	10.59 (0.29)	19.76 (0.15)
r = 3	1.03 (0.03)	20.15 (0.18)

SupSFPC they only depend on t. Table 3.2 gives the MSFE under both methods for three values of r, i.e., the number of FPCs used, see (3.3). CD-FPCA has a lower MSFE for all three values of r. Furthermore, in the examples shown in Figure 3.3 when $r = 3$, the 95% predictive intervals under CD-FPCA (shaded regions marked by dot-dash lines) are generally much narrower than those under SupSFPC (shaded regions marked by dotted lines), and yet still have better coverage of the true observed curves (solid lines). The corresponding empirical coverage of CD-FPCA predictive intervals on all predictions is 93.23% compared to 69.69% of SupSFPC.

The superior performance of CD-FPCA seen in Figure 3.3 and Table 3.2 is due to fact that SupSFPC assumes that only the scores $\boldsymbol{\xi}^{(n)}$ in (3.2) depend on the covariate, and that they do so linearly, whereas these assumptions are clearly violated by the simulation settings described in Section 3.4.1. If the assumptions were met then SupSFPC would perform more comparably, but in practice, there is typically no reason to expect them to hold. Indeed, they do not seem to hold in the real data application in Section 3.5.

However, to further investigate the flexibility of our CD-FPCA approach, we now generate a dataset of noisy observations of $N = 7500$ latent functions using the SupSFPC model, which is specified by (B.42) in Section B.3 of Appendix B. We set the mean function to be $\mu(t) = 0$, the three eigenfunctions to be $f_1(t) = \sqrt{2}\cos(\pi(t))$, $f_2(t) = \sqrt{2}\sin(\pi(t))$ and $f_3(t) = \sqrt{2}\cos(3\pi(t))$. and the eigenvalues to be $\boldsymbol{d} = \big(0.3, 0.2, 0.1\big)$. The SupSFPC model assumes that the scores for each eigenfunction vary linearly with the covariate and we set the slope coefficients for these linear models to be $\mathbf{T} = \big(5, 8, 3\big)$, see (B.42) (the intercepts are zero, because Li et al. (2016) specify that the covariates should be centered). We sample covariates from a uniform distribution on $[-10, 10]$.

After training the models on this dataset, we compare CD-FPCA and SupSFPC predictions for a test dataset generated under the same settings, again using (3.28) and (B.47) (Section B.3 of Appendix B) to compute the CD-FPCA and SupSFPC predictions, respectively. Then, we select the first 20% of the time points to be observed and predict the remaining observations. The first row of Table 3.3 gives the prediction MSFE averaged across all time points in the test dataset, as well as the empirical coverage of the CD-FPCA and SupSPFC predictive intervals.

Table 3.3: Overall prediction MSFE (and MSFE standard deviation) and empirical coverage of the predictive interval under CD-FPCA and SupSFPC for the test dataset.

	CD-FPCA ($\times 10^{-2}$)		SupSFPC($\times 10^{-2}$)	
	MSFE	Emp. Coverage	MSFE	Emp. Coverage
Linear	$15.91\ (0.42)$	$94.65\ (1.19)$	$15.86\ (0.05)$	$96.21\ (0.07)$
Quadratic	$16.00\ (0.42)$	$94.62\ (1.11)$	$38.16\ (0.02)$	$95.39\ (0.04)$

These results show that CD-FPCA performs as well as SupSPFC, despite the data being generated under the SupSPFC setting. This is not surprising because the SupSFPC model is nested within the CD-FPCA model, as detailed in Section B.4 of Appendix B. Of course, SupSFPC could still benefit from the simplifying linear assumption, because here that assumption is correct, but the resulting advantage appears to be negligible in this example.

As a final simulation study, we generate a dataset from a modified version of the SupSFPC model in which the scores are assumed to have a quadratic relationship with the covariates, see (B.50) in Section B.3 of Appendix B. The second row of Table 3.3 gives the resulting prediction MSFE for CD-FPCA and SupSFPC (with the linear assumption), along with the empirical coverage of the corresponding predictive intervals. Figure 3.4 shows predictions for an example curve in the test dataset under the CD-FPCA (center dot-dash lines) and SupSPFC (center dotted line) methods, for simulations under the linear (left panel) and quadratic (right panel) score model. The solid lines show the true noisy curves and the shaded regions show 95% predictive intervals under CD-FPCA

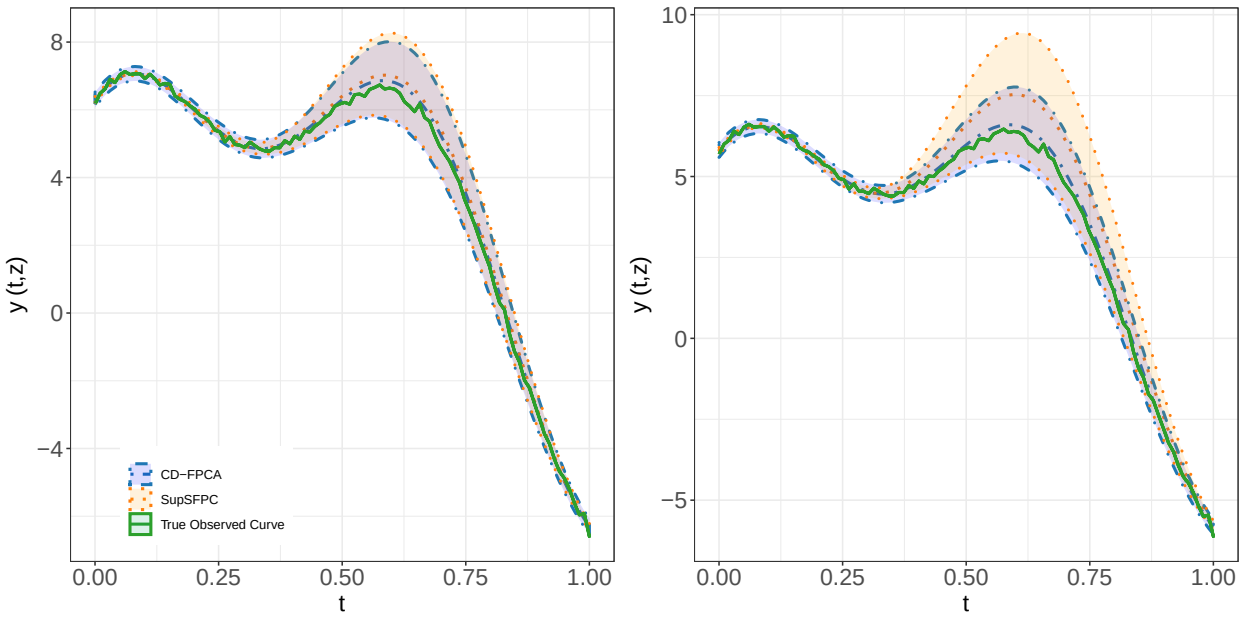

Figure 3.4: Example predictions under CD-FPCA (center dot-dash lines) and SupSFPC (center dotted lines) when the data are generated using a linear (left panel) and quadratic (right panel) score model. The shaded areas give 95% prediction regions and the true observed curves are shown as solid lines

(regions marked by dot-dash lines) and SupSPFC (regions marked by dotted lines). In summary, Table 3.3 and Figure 3.4 show that CD-FPCA performs comparably to SupSFPC when data are generated under the latter model, and performs substantially better when data are generated under a quadratic version of the SupSPFC model. This illustrates the greater flexibility of CD-FPCA compared with SupSFPC, and suggests that the former method should be preferred, except perhaps in the case of very large datasets for which we are confident that the SupSFPC assumptions are satisfied.

3.5 Modeling Astronomical Lightcurves

In astronomy, a lightcurve is a time series of the observed brightness of a source, e.g., a star or galaxy. Lightcurves are useful because some astronomical sources vary in brightness over time, and these variations can be used to classify the type of source or infer its properties, e.g., the period of star pulsations (from which additional physical insights can be gained). One type of variable

source is an eclipsing binary system, which is a system of two stars orbiting each other. Many stars visible to the eye are in fact eclipsing binary systems. If the orbits of the two stars lie in the plane that also contains our line of sight then the stars will alternately eclipse each other from our perspective. Binary stars cannot usually be resolved, but the eclipses block some of the light from reaching us and create periodic dips in the observed lightcurve. These characteristics can be used to distinguish eclipsing binary sources from other variable sources and help us to infer properties of the two stars, e.g., their relative masses.

Our data set consists of $N = 35615$ eclipsing binary lightcurves from the Catalina Real-Time Transient Survey (CRTS) (Drake et al., 2009) which were classified by the CRTS team in Drake et al. (2014). Each observed magnitude (brightness) measurement is accompanied by a known measurement error (i.e., standard deviation), that is determined by astronomers based on the properties of the telescope used. The data are publicly available from `http://crts.caltech.edu/`. The top left panel of Figure 3.5 shows 10 standardized eclipsing binary lightcurves from the dataset. The y-axis units are standardized magnitude: magnitude is an astronomical measure of the intensity of light from a star, with smaller numbers indicating greater intensity. Standardizing so that all the measurements fall in $[-0.5, 0.5]$ is necessary here because we are principally interested in modeling the similar shapes of the lightcurves. For visual purposes the measurement errors are not plotted; they have a median value of about 0.15 (in standardized magnitude units). The x-axes in Figure 3.5 are phase of oscillation (as opposed to time), because eclipsing binary lightcurves are periodic. The period of oscillation for each lightcurve was found by Drake et al. (2014) and is treated as known for the purposes of our analysis. In practice, the periods would have to be estimated, which is itself a challenging inference problem. It is worth noting that the improved modeling we present here could in turn facilitate improved period estimation accuracy in future.

An important feature in the top left panel of Figure 3.5 is that for some lightcurves the depth of the two eclipses are similar, and for others they are very different. This distinction is due to there being different types of eclipsing binary system. Eclipsing binaries are often divided into

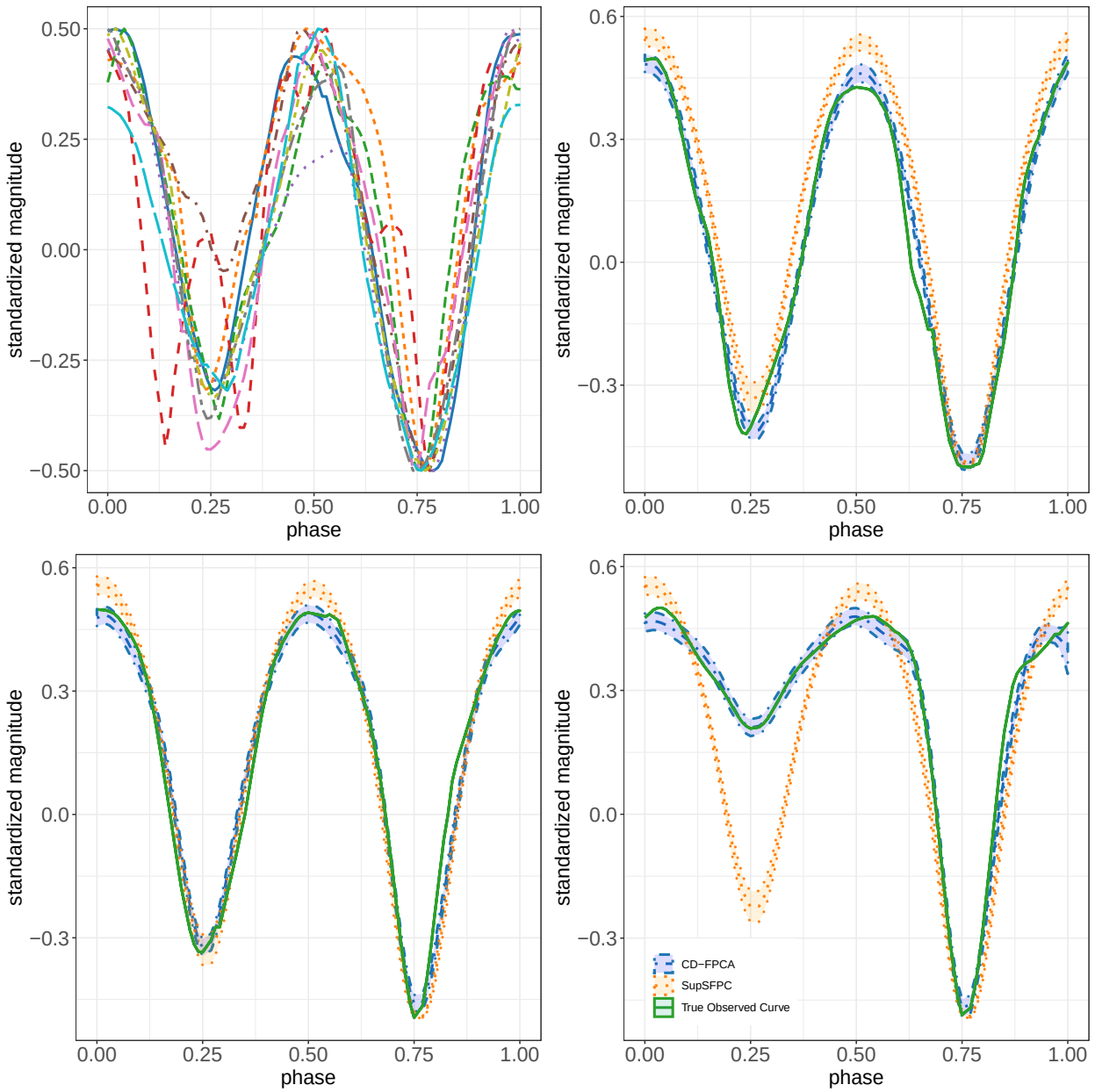

Figure 3.5: (Top left) standardized lightcurves of 10 eclipsing binary sources. (Top right, bottom left, bottom right) example lightcurve predictions and 95% predictive intervals for CD-FPCA (center dot-dash lines) and SupSFPC (center dotted lines). The solid lines show the true observed lightcurves

two classes, contact binaries which are sufficiently close to exchange mass, and detached binaries which are more separated. Contact binaries typically consist of two sources with similar properties (e.g., size and brightness), meaning that the two eclipses are similar. In contrast, detached binaries may have eclipses of any relative size, because the two sources can have completely different properties.

The above considerations raise an important modeling challenge: eclipsing binary lightcurves can be modeled using somewhat similar functions, because they have similar shapes and covariance structures, but it does not make sense to treat them as coming from a completely homogeneous distribution, as is typically assumed in FPCA. The current solution is to divide eclipsing binaries into contact and detached binaries and treat these groups as homogeneous, but this is still unsatisfactory because the detached binaries group is heterogeneous. Treating eclipsing binaries as homogeneous, means that any models we use to fit them will either be inaccurate or unnecessarily complicated, which in turn will reduce our ability to classify them, estimate their periods, and learn their other properties. Instead, we use our CD-FPCA method to learn a mean function and a set of covariance matrix eigenfunctions that smoothly vary with the relative depth of the two eclipses. This approach captures the fact that eclipsing binary lightcurves are similar, while also accounting for a physically interpretable difference.

To implement our approach we need a parameter or covariate z related to the relative depth of the two eclipses of each lightcurve. In practice, such information may sometimes be available from a separate observation of the eclipsing binary system, e.g., from another telescope targeting a different light wavelength range. However, in many cases a parameter capturing the relative depth would need to be inferred from the data. For the sake of simplicity, in this work we calculate an approximation of the relative depth of the eclipses of each lightcurve from the data and treat it as a covariate z. In particular, we use a simple cubic B-spline approximation to each lightcurve and compute the ratio of the change in standardized magnitude for the larger eclipse and that for the smaller eclipse. For our dataset this gives z values in the range $[1.00, 88.48]$. The key point is that this covariate is low dimensional (univariate) but still explains a great deal of the variation between

Table 3.4: Prediction MSFE under CD-FPCA and SupSFPC for the gridded lightcurve data.

	CD-FPCA ($\times 10^{-2}$)		SupSFPC($\times 10^{-2}$)	
No. FPCs	Training Set	Test Set	Training Set	Test Set
r = 1	2.06	2.08	2.05	2.18
r = 2	1.73	1.74	1.70	2.12
r = 3	0.42	0.42	1.64	1.75

the lightcurves (which have infinite dimension).

We now compare the modeling and prediction performance of CD-FPCA to that of SupSPFC for the lightcurve data. Since SupSFPC only handles regularly spaced data and does make use of measurement errors, we initially consider a processed version of the data in which all the observations lie on a regular grid in phase space. In particular, we use a cubic spline fit to each lightcurve to obtain observations at phases $t_i = \frac{i-1}{m-1}$ for $i = 1, \ldots, m = 101$, regardless of the number of observations in the original lightcurve. For this gridded data, we do not have measurement errors. After comparing the methods on the gridded data, we will also apply CD-FPCA to the raw data (including the measurement errors) to further demonstrate its applicability and performance.

We randomly divided the gridded dataset into a training set and a test set, composed of 80% and 20% of the total number of ligthcurves, respectively. Table 3.4 shows the training and test prediction MSFE for both CD-FPCA and SupSFPC. The predictions are computed in the same way as in Section 3.4.2, except that a random 25% of the observations in each lightcurve are used to estimate the lightcurve-specific scores $\boldsymbol{\xi}^{(n)}$ (see (3.3)), and predictions are made for the other 75%. The rows of Table 3.4 correspond to different values of r, the rank of the matrix $\mathbf{C}$ used in approximating the covariance matrix $\boldsymbol{\Sigma}$, see (3.6). Table 3.4 shows that the training set prediction MSFE is similar for both methods when $r = 2$, which suggests that the effective degrees of freedom of the two models are similar in this case. However, the test set prediction MSFE is much lower for our CD-FPCA method when $r = 2$, and in fact for all three choices of r.

The top right panel and two bottom panels of Figure 3.5 show estimates and 95% predictive intervals for three example lightcurves in the test set under the CD-FPCA (center dot-dash lines)

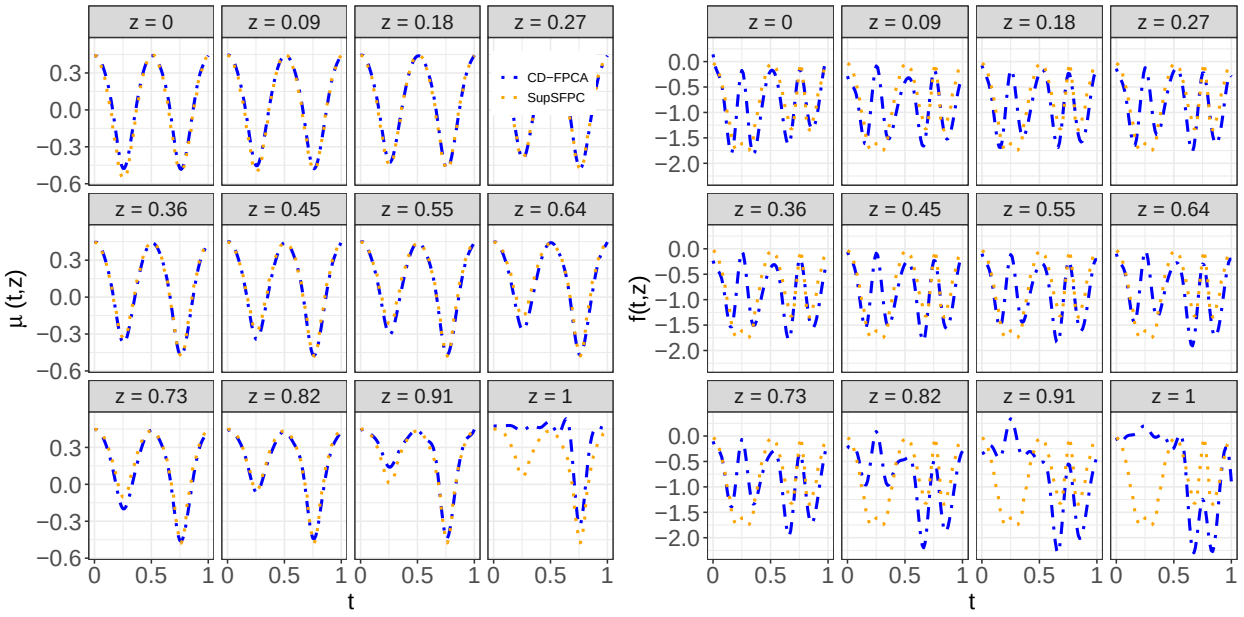

Figure 3.6: (Left panel) estimates of the mean function under the CD-FPCA (dot-dash lines) and SupSFPC (dotted lines) methods, for a range of covariate values. (Right panel) estimates of the first eigenfunction, under the CD-FPCA (dot-dash lines) and SupSFPC (dotted lines) methods, for a range of covariate value, where the covariates have been scaled to $[0, 1]$

and SupSFPCA (center dotted lines) methods. The true observed lightcurves are also shown (solid lines). CD-FPCA well captures the way the lightcurve shapes vary with the covariate, and also provides reasonable 95% predictive intervals. In contrast, the SupSFPC method does not capture the lightcurve shapes well, because its assumption that the scores vary linearly with the covariate is not valid. To further demonstrate this, we set the loss function to be the prediction mean squared fitting errors and plot it for both methods in the top left panel of Figure 3.7. CD-FPCA has similar loss for all values of z, but SupSFPC has much higher loss for larger values of z. In particular, since the SupSFPC method cannot properly capture the way the lightcurves change with the covariate, it focuses on fitting lightcurves with low covariate values, which constitute the majority of the data (and correspond to contact binaries).

Figure 3.6 shows the comparison of estimated mean and the first eigenfunction between CD-FPCA and SupSFPC. The estimated mean function of CD-FPCA is much different when the covariate is larger than 0.9, which corresponds to β Persei, also known as Algol eclipsing system.

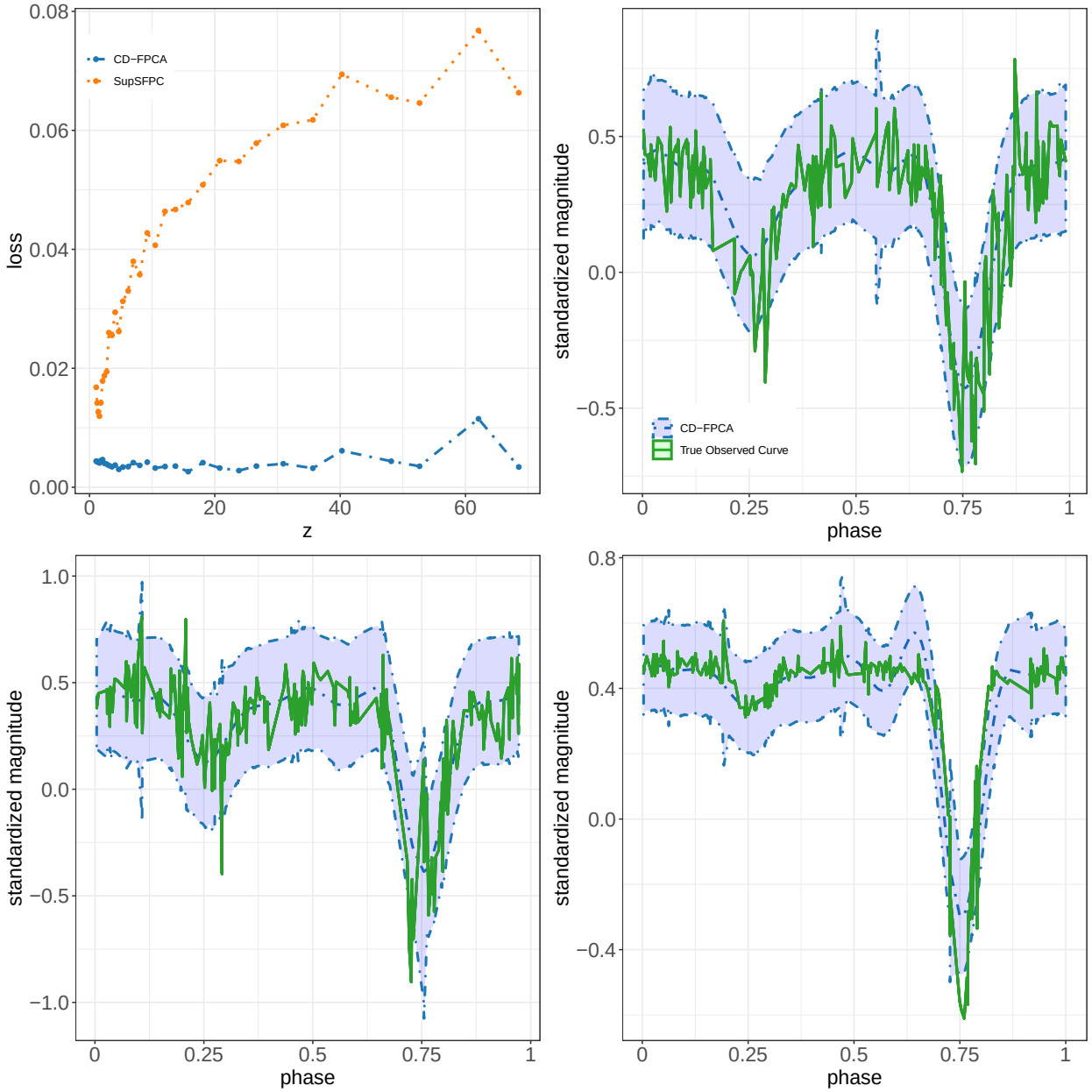

Figure 3.7: (Top left) prediction MSFE loss for CD-FPCA (dot-dash line) and SupSFPC (dotted line). (Top right, bottom left bottom right) CD-FPCA predictions and 95% predictive intervals (center dot-dash lines) for three example raw test data lightcurves. The solid line shows the raw lightcurves

As stated in Kallrath et al. (2009), the fall of Algol lightcurves occupies only a small fraction of the full light curve, typically less than approximately 15% for each minimum and the depth of the two minima are very different. The estimated eigenfunctions of CD-FPCA are more interpretable compared with SupSFPC. For different types of the eclipse system, e.g., contact binaries and detached binaries, the intensity patterns of the light are much different. Since the contact binaries typically consist of two similar sources (e.g., size and brightness), the intensity of the light may enjoy the same pattern varying with the phase. While the intensity patterns of detached binaries are much different because of their heterogeneous relative sizes. The estimated first eigenfunctions of CD-FPCA capture this feature by the location of the fluctuation varying with the covariates(related to the relative depth of the two eclipse binaries). When the covariate is small, the two periods of the estimated first eigenfunction almost the same which shows the pattern of the contact binaries. While the fluctuations become relatively larger around the second periodic dip as the covariate increase which shows the pattern of the detached binaries. Especially when $z = 0.73, 0.82, 0.91$, i.e., in the Algol eclipse binaries system where the hotter, bluer star dominates the light from the system and make the brightness of the system more stable during the first half phase, the small fall of the lightcurves may be undetectable, and this small fall near the first half phase is due to the "reflection effect" (a reprocessing of the hotter stars' radiation as it impinges on the atmosphere of its companion, increasing the cooler star's luminosity in the irradiated area, see Kallrath et al. (2009)). The estimated first eigenfunctions of CD-FPCA well capture this "reflection effect", showing that the fluctuations in the first half phase are much small compared to larger fluctuations in the last half phase.

Next, we apply CD-FPCA to the raw data, which cannot be analyzed using SupSFPC. In this case the training and test set prediction MSFE are almost identical to those in Table 3.4, indicating that the model still well captures the data. The top right and bottom two panels of Figure 3.7 show 95% predictive intervals for three example lightcurves in the test set. In this the case 95% predictive intervals actually appear more consistent with the data than in the gridded data case. This is partly due to chance and partly because, for the raw data, observation-specific measurement errors are

available both for fitting the model and for making predictions. In practice, reliable observation-specific measurement errors may not be available for predication (because measurement errors are typically calculated at the time of observation), but we can still obtain rough estimates of the measurement errors based on the training data, e.g., we could consider the median measurement error or measurement errors at phases nearby to the point where a prediction is made.

3.6 Discussion

Our CD-FPCA method offers an attractive option for analyzing large functional datasets in which both the mean and covariance function vary smoothly with covariates. It can flexibly incorporate this type of dependence and has substantially lower computational cost than popular local smoother based covariate adjustment approaches, e.g., Jiang and Wang (2010), Jiang and Wang (2011), Zhang et al. (2013), and Zhang and Wang (2016). While the SupSPFC approach proposed by Li et al. (2016) has even lower computational cost than CD-FPCA, it does not achieve the same levels of flexibility and accuracy. Indeed, in both our simulation study and our real data analysis, CD-FPCA performed better than SupSFPC in all the accuracy comparisons that we considered, e.g., mean and eigenfunction estimation, and prediction accuracy (except when the restrictive assumptions of SupSFPC were exactly satisfied, in which case the two methods were comparable, see the first row of Table 3.3). Furthermore, CD-FPCA can handle irregular and unbalanced observation times and incorporate measurement errors, whereas SupSFPC can not.

For simplicity we have focused on illustrations with univariate covariates, but our model formulation in Section 3.2 is general, and in future work it would be valuable to investigate the performance of our approach for datasets with higher dimensional covariates. Regarding further methodological development, one could construct a map from Euclidean space to the Stiefel manifold, as opposed to the symmetric positive semi-definite rank r matrix manifold (see (3.6)), because the underlying FPC coefficients matrix, denoted $\boldsymbol{\Theta}_{\boldsymbol{z}}$, lies on the Stiefel manifold (where, more specifically, $\boldsymbol{\Theta}_{\boldsymbol{z}}$ is such that $\boldsymbol{f}(t, \boldsymbol{z}) = \boldsymbol{\Theta}_{\boldsymbol{z}}^T \boldsymbol{b}(t)$). Indeed, the Stiefel manifold is a more frequently used structure than the symmetric positive semi-definite rank r matrix manifold, and optimization methods on the Stiefel manifold have been well developed, e.g., Boothby (1986), Balogh et al.

(2004), Nishimori and Akaho (2005), Wen and Yin (2013). There may be advantages to a Stiefel manifold based approach compared with our method proposed here, or vice versa, but this needs to be further investigated.

4. CONCLUSIONS

In my dissertation, we have proposed three novel methods. In the first chapter, we proposed a new adaptive Markov chain Monte Carlo (MCMC) method to sample from multi-modal target densities and simultaneously perform much of the computation needed for our complementary Bayesian evidence estimator. We also established the theorem of ergodicity for our Warp-U MCMC algorithm and offered theoretical proof of convergence of our algorithm. On the other hand, we proposed a new version of the Warp-U bridge estimator (Wang et al., 2020) which has lower asymptotic variance and offered theoretical proof. We also demonstrated the practical advantages of our approaches through a simulation study and an astronomical exoplanet data analysis.

In the second chapter, proposed a new supervised functional PCA framework in which both the mean and covariance structure depend on covariates. We also proposed a corresponding estimation algorithm, which makes use of spline basis representations and roughness penalties, and is substantially more computationally efficient than competing approaches of adequate estimation and prediction accuracy. We demonstrated the advantages of our methodology through a simulation study and an astronomical light curves dataset analysis.

REFERENCES

Ali, S. M. and Silvey, S. D. (1966). A general class of coefficients of divergence of one distribution from another. *Journal of the Royal Statistical Society: Series B (Methodological)*, 28(1):131–142.

Andrieu, C. and Robert, C. P. (2001). *Controlled MCMC for optimal sampling*. INSEE.

Atchadé, Y. F. (2006). An adaptive version for the metropolis adjusted langevin algorithm with a truncated drift. *Methodology and Computing in applied Probability*, 8(2):235–254.

Atchadé, Y. F., Roberts, G. O., and Rosenthal, J. S. (2011). Towards optimal scaling of metropolis-coupled markov chain monte carlo. *Statistics and Computing*, 21(4):555–568.

Azzalini, A. (2013). *The skew-normal and related families*, volume 3. Cambridge University Press.

Balogh, J., Csendes, T., and Rapcsák, T. (2004). Some global optimization problems on stiefel manifolds. *Journal of Global Optimization*, 30(1):91–101.

Behseta, S., Kass, R. E., and Wallstrom, G. L. (2005). Hierarchical models for assessing variability among functions. *Biometrika*, 92(2):419–434.

Bennett, C. H. (1976). Efficient estimation of free energy differences from monte carlo data. *Journal of Computational Physics*, 22(2):245–268.

Berg, B. A. and Neuhaus, T. (1991). Multicanonical algorithms for first order phase transitions. *Physics Letters B*, 267(2):249–253.

Boothby, W. M. (1986). *An introduction to differentiable manifolds and Riemannian geometry*, volume 120. Academic press.

Bornn, L., Jacob, P. E., Del Moral, P., and Doucet, A. (2013). An adaptive interacting Wang–Landau algorithm for automatic density exploration. *Journal of Computational and Graphical Statistics*, 22(3):749–773.

Butterfield, K. R. (1976). The computation of all the derivatives of a b-spline basis. *IMA Journal of Applied Mathematics*, 17(1):15–25.

Cai, T. and Yuan, M. (2010). Nonparametric covariance function estimation for functional and

longitudinal data. *University of Pennsylvania and Georgia inistitute of technology*.

Cai, T. and Yuan, M. (2011). Optimal estimation of the mean function based on discretely sampled functional data: Phase transition. *The annals of statistics*, 39(5):2330–2355.

Capra, W. B. and Müller, H.-G. (1997). An accelerated-time model for response curves. *Journal of the American Statistical Association*, 92(437):72–83.

Chib, S. and Jeliazkov, I. (2001). Marginal likelihood from the metropolis–hastings output. *Journal of the American Statistical Association*, 96(453):270–281.

De Boor, C. (1977). Package for calculating with b-splines. *SIAM Journal on Numerical Analysis*, 14(3):441–472.

Drake, A., Djorgovski, S., Mahabal, A., Beshore, E., Larson, S., Graham, M., Williams, R., Christensen, E., Catelan, M., Boattini, A., et al. (2009). First results from the catalina real-time transient survey. *The Astrophysical Journal*, 696(1):870.

Drake, A., Graham, M., Djorgovski, S., Catelan, M., Mahabal, A., Torrealba, G., García-Álvarez, D., Donalek, C., Prieto, J., Williams, R., et al. (2014). The catalina surveys periodic variable star catalog. *The Astrophysical Journal Supplement Series*, 213(1):9.

Geyer, C. J. (1991). Markov chain monte carlo maximum likelihood. *Computing science and statistics: Proceedings of 23rd Symposium on the Interface, Fairfax Station, 1991*, pages 156–163.

Jacob, P. E., Ryder, R. J., et al. (2014). The Wang–Landau algorithm reaches the flat histogram criterion in finite time. *The Annals of Applied Probability*, 24(1):34–53.

James, G. M., Hastie, T. J., and Sugar, C. A. (2000). Principal component models for sparse functional data. *Biometrika*, 87(3):587–602.

Jarner, S. F. and Hansen, E. (2000). Geometric ergodicity of metropolis algorithms. *Stochastic processes and their applications*, 85(2):341–361.

Jiang, C.-R. and Wang, J.-L. (2010). Covariate adjusted functional principal components analysis for longitudinal data. *The Annals of Statistics*, 38(2):1194–1226.

Jiang, C.-R. and Wang, J.-L. (2011). Functional single index models for longitudinal data. *The*

Annals of Statistics, 39(1):362–388.

Kallrath, J., Milone, E. F., and Wilson, R. (2009). *Eclipsing binary stars: modeling and analysis*. Springer.

Li, G., Shen, H., and Huang, J. Z. (2016). Supervised sparse and functional principal component analysis. *Journal of Computational and Graphical Statistics*, 25(3):859–878.

Liang, F. (2002). Dynamically weighted importance sampling in monte carlo computation. *Journal of the American Statistical Association*, 97(459):807–821.

Liang, F. (2005). A generalized Wang–Landau algorithm for Monte Carlo computation. *Journal of the American Statistical Association*, 100(472):1311–1327.

Liang, F., Liu, C., and Carroll, R. J. (2007). Stochastic approximation in monte carlo computation. *Journal of the American Statistical Association*, 102(477):305–320.

Lin, L., St. Thomas, B., Zhu, H., and Dunson, D. B. (2017). Extrinsic local regression on manifold-valued data. *Journal of the American Statistical Association*, 112(519):1261–1273.

Loredo, T. J., Berger, J. O., Chernoff, D. F., Clyde, M. A., and Liu, B. (2012). Bayesian methods for analysis and adaptive scheduling of exoplanet observations. *Statistical Methodology*, 9(1-2):101–114.

Meng, X.-L. and Schilling, S. (2002). Warp bridge sampling. *Journal of Computational and Graphical Statistics*, 11(3):552–586.

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.

Mira, A. and Nicholls, G. (2004). Bridge estimation of the probability density at a point. *Statistica Sinica*, pages 603–612.

Nelson, B. E., Ford, E. B., Buchner, J., Cloutier, R., Díaz, R. F., Faria, J. P., Rajpaul, V. M., and Rukdee, S. (2018). Quantifying the evidence for a planet in radial velocity data. *arXiv preprint arXiv:1806.04683*.

Nielsen, F. and Nock, R. (2013). On the chi square and higher-order chi distances for approximating f-divergences. *IEEE Signal Processing Letters*, 21(1):10–13.

Nishimori, Y. and Akaho, S. (2005). Learning algorithms utilizing quasi-geodesic flows on the stiefel manifold. *Neurocomputing*, 67:106–135.

Peng, J. and Paul, D. (2009). A geometric approach to maximum likelihood estimation of the functional principal components from sparse longitudinal data. *Journal of Computational and Graphical Statistics*, 18(4):995–1015.

Pezzulli, S. (1993). Some properties of smoothed principal components analysis for functional data. *Computational Statistics*, 8(1):1–16.

Pullen, N. and Morris, R. J. (2014). Bayesian model comparison and parameter inference in systems biology using nested sampling. *PloS one*, 9(2):e88419.

Ramsay, J. and Silverman, B. (2005). Principal components analysis for functional data. *Functional Data Analysis*, pages 147–172.

Reiss, P. T., Huang, L., Chen, H., and Colcombe, S. (2014). Varying-smoother models for functional responses. *arXiv preprint arXiv:1412.0778*.

Rice, J. A. and Silverman, B. W. (1991). Estimating the mean and covariance structure nonparametrically when the data are curves. *Journal of the Royal Statistical Society: Series B (Methodological)*, 53(1):233–243.

Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407.

Roberts, G. O. and Rosenthal, J. S. (2007). Coupling and ergodicity of adaptive markov chain monte carlo algorithms. *Journal of applied probability*, 44(2):458–475.

Rodríguez, A., Dunson, D. B., and Gelfand, A. E. (2009). Bayesian nonparametric functional data analysis through density estimation. *Biometrika*, 96(1):149–162.

Rubenstein, P. K., Bousquet, O., Djolonga, J., Riquelme, C., and Tolstikhin, I. (2019). Practical and consistent estimation of f-divergences. *arXiv preprint arXiv:1905.11112*.

Silverman, B. W. (1996). Smoothed functional principal components analysis by choice of norm. *The Annals of Statistics*, 24(1):1–24.

Suarez, A. J. and Ghosal, S. (2017). Bayesian estimation of principal components for functional

data. *Bayesian Analysis*, 12(2):311–333.

Van Der Linde, A. (2008). Variational bayesian functional pca. *Computational Statistics & Data Analysis*, 53(2):517–533.

Wang, F. and Landau, D. P. (2001). Efficient, multiple-range random walk algorithm to calculate the density of states. *Physical review letters*, 86(10):2050.

Wang, L., Jones, D. E., and Meng, X.-L. (2020). Warp bridge sampling: The next generation. *Journal of the American Statistical Association*.

Wen, Z. and Yin, W. (2013). A feasible method for optimization with orthogonality constraints. *Mathematical Programming*, 142(1-2):397–434.

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

Yao, F., Müller, H.-G., and Wang, J.-L. (2005). Functional data analysis for sparse longitudinal data. *Journal of the American Statistical Association*, 100(470):577–590.

Zhang, X., Park, B. U., and Wang, J.-l. (2013). Time-varying additive models for longitudinal data. *Journal of the American Statistical Association*, 108(503):983–998.

Zhang, X. and Wang, J.-L. (2016). From sparse to dense functional data and beyond. *The Annals of Statistics*, 44(5):2281–2321.

Zhu, H., Chen, Y., Ibrahim, J. G., Li, Y., Hall, C., and Lin, W. (2009). Intrinsic regression models for positive-definite matrices with applications to diffusion tensor imaging. *Journal of the American Statistical Association*, 104(487):1203–1212.

APPENDIX A

SETTING AND THEORETICAL PROOF OF WARP-U MCMC AND STOCHASTIC BRIDGE SAMPLING

A.1 Simulation Settings for Figure 2.1

For the simulation summarized in Figure 2.1, the unnormalized target density is given by

$$q = \frac{1}{15}\exp(-\frac{1}{2}(\boldsymbol{\theta}+\mathbf{11})^T(\boldsymbol{\theta}+\mathbf{11})) + \frac{2}{15}\exp(-\frac{1}{2}(\boldsymbol{\theta}-\mathbf{12})^T(\boldsymbol{\theta}-\mathbf{12})) \tag{A.1}$$

$$+ \frac{3}{15}\exp(-\frac{1}{2}(\boldsymbol{\theta}+\mathbf{8})^T(\boldsymbol{\theta}+\mathbf{8})) + \frac{4}{15}\exp(-\frac{1}{2}(\boldsymbol{\theta}-\mathbf{7})^T(\boldsymbol{\theta}-\mathbf{7})) \tag{A.2}$$

$$+ \frac{5}{15}\exp(-\frac{1}{2}(\boldsymbol{\theta}+\mathbf{2})^T(\boldsymbol{\theta}+\mathbf{2})). \tag{A.3}$$

The two-mode Gaussian mixture distribution used as ϕ_{mix} in right panel of Figure 2.1 is given by

$$\phi_{\text{mix}} \propto \frac{1}{2}\exp(-\frac{1}{2}(\boldsymbol{\theta}+\mathbf{11})^T(\boldsymbol{\theta}+\mathbf{11})) + \frac{1}{2}\exp(-\frac{1}{2}(\boldsymbol{\theta}-\mathbf{12})^T(\boldsymbol{\theta}-\mathbf{12})). \tag{A.4}$$

A.2 Proof of Ergodicity (Theorem 1)

Proof of Lemma 1. Let $\kappa(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)$ be the transition kernel of the MH update in Step 3 of Algorithm 2, and let $m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)$ be the transition kernel of Steps 4-7 of Algorithm 2. Then, the full transition kernel of Algorithm 2 is

$$P_\gamma(\boldsymbol{\theta}_2|\boldsymbol{\theta}_0) = \int m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\kappa(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)\,\mathrm{d}\boldsymbol{\theta}_1. \tag{A.5}$$

We want to verify that the transition kernel (A.5) preserves the target distribution π which is there-

fore the stationary distribution of Algorithm 2:

$$\int \pi(\boldsymbol{\theta}_0) P_\gamma(\boldsymbol{\theta}_2|\boldsymbol{\theta}_0)\,\mathrm{d}\boldsymbol{\theta}_0 = \int\int \pi(\boldsymbol{\theta}_0)\kappa(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0) m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1\,\mathrm{d}\boldsymbol{\theta}_0 \tag{A.6}$$

$$= \int \pi(\boldsymbol{\theta}_1)\; m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1. \tag{A.7}$$

The equality (A.7) holds due to the established result of the MH update.

To show the transition kernel $m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)$ preserves π, we denote $\mathcal{G}_{\psi^*,\psi} = \mathcal{H}_{\psi *} \circ \mathcal{F}_\psi$ as the composite of the stochastics transformation in Steps 4-7 of Algorithm 2. Set $\boldsymbol{\theta}_2 = \mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1) = \mathcal{H}_{\psi *} \circ \mathcal{F}_\psi(\boldsymbol{\theta}_1)$. When $\boldsymbol{\theta}_1 \sim \pi(\boldsymbol{\theta})$ and suppose the transformed variable $\boldsymbol{\theta}_2$ has density $f(\boldsymbol{\theta})$, our target is equivalent to show that $\pi(\boldsymbol{\theta}) \equiv f(\boldsymbol{\theta})$.

We firstly derive an explicit expression for the density f based on the transition probabilities in Step 4–7 of Algorithm 2. Denote $p(\psi^*, \psi|\boldsymbol{\theta}_1)$ as the probability of choosing transformation $\mathcal{F}_\psi$ and $\mathcal{H}_{\psi^*}$ given $\boldsymbol{\theta}_1$. Let A be an arbitrary Borel set of $\mathbb{R}^d$. Then, when $\boldsymbol{\theta}_1 \sim \pi(\boldsymbol{\theta})$, we have

$$P(\boldsymbol{\theta}_2 \in A) = \int \sum_{\psi,\psi^*=1}^{K} I(\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1) \in A) p(\psi^*, \psi|\boldsymbol{\theta}_1)\pi(\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1 \tag{A.8}$$

$$= \int \sum_{\psi,\psi^*=1}^{K} I(\boldsymbol{\theta}_2 \in A) p(\psi^*, \psi|\mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2))\pi(\mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2)) \times \left|\frac{\partial \mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2)}{\partial \boldsymbol{\theta}_2}\right| \mathrm{d}\boldsymbol{\theta}_2 \tag{A.9}$$

$$= \int_A \sum_{\psi,\psi^*=1}^{K} p(\psi^*, \psi|\mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2))\pi(\mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2)) \times \left|\frac{\partial \mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2)}{\partial \boldsymbol{\theta}_2}\right| \mathrm{d}\boldsymbol{\theta}_2. \tag{A.10}$$

In the above, the second equality applies the change of variable $\boldsymbol{\theta}_1 = \mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2)$ for the integral. Therefore, when $\boldsymbol{\theta}_1 \sim \pi(\boldsymbol{\theta})$, the probability density for $\boldsymbol{\theta}_2$ is

$$f(\boldsymbol{\theta}_2) = \sum_{\psi^*,\psi=1}^{K} \pi(\mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2)) \times \left|\frac{\partial \mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2)}{\partial \boldsymbol{\theta}_2}\right| \times p(\psi^*, \psi|\mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2)) \tag{A.11}$$

$$= \sum_{\psi^*,\psi=1}^{K} \pi(\mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2)) \times |\mathcal{S}_\psi| \times |\mathcal{S}_{\psi^*}|^{-1} \times p(\psi^*, \psi|\mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2)). \tag{A.12}$$

We continue to find an expression for $p(\psi^*, \psi|\mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2))$. Denote $\boldsymbol{\theta}^* = \mathcal{H}^{-1}_{\psi^*}(\boldsymbol{\theta}_2)$ as the in-

termediate Warp-U transformed sample (Algorithm 2 Step 5). The normalizing constant for the sampling probability in equation (2.7) is

$$b(\boldsymbol{\theta}^*) = \sum_{\widetilde{\psi}=1}^{K} \varpi(\widetilde{\psi}|\mathcal{H}_{\widetilde{\psi}}(\boldsymbol{\theta}^*)) \times q(\mathcal{H}_{\widetilde{\psi}}(\boldsymbol{\theta}^*)) \times |\mathcal{S}_{\widetilde{\psi}}| \tag{A.13}$$

$$= \sum_{\widetilde{\psi}=1}^{K} \varpi(\widetilde{\psi}|\mathcal{G}^{-1}_{\psi^*,\widetilde{\psi}}(\boldsymbol{\theta}_2)) \times q(\mathcal{G}^{-1}_{\psi^*,\widetilde{\psi}}(\boldsymbol{\theta}_2)) \times |\mathcal{S}_{\widetilde{\psi}}|, \tag{A.14}$$

where the last equality uses that $\mathcal{H}_{\widetilde{\psi}}(\boldsymbol{\theta}^*) = \mathcal{H}_{\widetilde{\psi}} \circ \mathcal{H}^{-1}_{\psi*}(\boldsymbol{\theta}_2) = \mathcal{F}^{-1}_{\widetilde{\psi}} \circ \mathcal{H}^{-1}_{\psi*}(\boldsymbol{\theta}_2) = \mathcal{G}^{-1}_{\psi^*,\widetilde{\psi}}(\boldsymbol{\theta}_2)$. Given the starting point $\mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2)$ in Step 4 of Algorithm 2, the probability of choosing $\mathcal{F}_\psi, \mathcal{H}_{\psi^*}$ in the subsequent steps is

$$p(\psi^*, \psi|\mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2)) = p(\psi|\mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2)) \times p(\psi^*|\mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2), \psi) \tag{A.15}$$

$$= \varpi(\psi|\mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2)) \times \frac{\varpi(\psi^*|\boldsymbol{\theta}_2)\pi(\boldsymbol{\theta}_2)|\mathcal{S}_{\psi^*}|}{\sum_{\widetilde{\psi}=1}^{K} \varpi(\widetilde{\psi}|\mathcal{G}^{-1}_{\psi^*,\widetilde{\psi}}(\boldsymbol{\theta}_2))\pi(\mathcal{G}^{-1}_{\psi^*,\widetilde{\psi}}(\boldsymbol{\theta}_2))|\mathcal{S}_{\widetilde{\psi}}|} \tag{A.16}$$

Consequently, combining (A.12) and (A.16), we can get

$$f(\boldsymbol{\theta}_2) = \sum_{\psi^*,\psi=1}^{K} \pi(\mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2))|\mathcal{S}_\psi| \times \frac{\varpi(\psi|\mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2)) \times \varpi(\psi^*|\boldsymbol{\theta}_2)\pi(\boldsymbol{\theta}_2)}{\sum_{\widetilde{\psi}=1}^{K} \varpi(\widetilde{\psi}|\mathcal{G}^{-1}_{\psi^*,\widetilde{\psi}}(\boldsymbol{\theta}_2))\pi(\mathcal{G}^{-1}_{\psi^*,\widetilde{\psi}}(\boldsymbol{\theta}_2))|\mathcal{S}_{\widetilde{\psi}}|} \tag{A.17}$$

$$= \pi(\boldsymbol{\theta}_2) \sum_{\psi^*=1}^{K} \varpi(\psi^*|\boldsymbol{\theta}_2) \sum_{\psi=1}^{K} \frac{\varpi(\psi|\mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2))\pi(\mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2))|\mathcal{S}_\psi|}{\sum_{\widetilde{\psi}=1}^{K} \varpi(\widetilde{\psi}|\mathcal{G}^{-1}_{\psi^*,\widetilde{\psi}}(\boldsymbol{\theta}_2))\pi(\mathcal{G}^{-1}_{\psi^*,\widetilde{\psi}}(\boldsymbol{\theta}_2))|\mathcal{S}_{\widetilde{\psi}}|} \tag{A.18}$$

$$= \pi(\boldsymbol{\theta}_2) \sum_{\psi^*=1}^{K} \varpi(\psi^*|\boldsymbol{\theta}_2) = \pi(\boldsymbol{\theta}_2). \tag{A.19}$$

From the above, we have proved the transition kernel $m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)$ in (A.7) preserves the stationary distribution π.

□

Proof of Lemma 2. We need to prove that

$$\lim_{s\to\infty} \mathbb{P}(\sup_{\boldsymbol{\theta}\in\mathbb{R}^d} \|P_{\hat{\gamma}^s}(\boldsymbol{\theta},\cdot) - P_{\hat{\gamma}^{s+1}}(\boldsymbol{\theta},\cdot)\|_{TV} \geq \delta_1) = 0 \tag{A.20}$$

for any $\delta_1 > 0$, where d is the dimension of $\boldsymbol{\theta}$. For fixed $\boldsymbol{\theta}$, the map from $\hat{\gamma}$ to $P_{\hat{\gamma}}(\boldsymbol{\theta}, A)$ is continuous and bounded for any A, and therefore it suffices to show that

$$\lim_{s\to\infty} \mathbb{P}(\|\hat{\gamma}^s - \hat{\gamma}^{s+1}\| > \delta) = 0, \tag{A.21}$$

for any $\delta > 0$. Recall that the estimate $\hat{\gamma}^s$ is obtained from the EM algorithm using the samples from all s stages run. To prove (A.21), we begin by using induction to show that for each iteration t of the EM algorithm we have

$$\lim_{s\to\infty} \mathbb{P}(\|\hat{\gamma}_k^{s,t} - \hat{\gamma}_k^{s+1,t}\| > \delta) = 0 \tag{A.22}$$

where $\gamma_k^{s,t} = (\boldsymbol{\mu}_1^{s,t} \ldots, \boldsymbol{\mu}_k^{s,t}, \boldsymbol{\Sigma}_k^{s,t}, \ldots, \boldsymbol{\Sigma}_k^{s,t}, w_k^{s,t}, \ldots, w_k^{s,t})$ and $\boldsymbol{\mu}_k^{s,t}$, $\boldsymbol{\Sigma}_k^{s,t}, w_k^{s,t}$ are parameters of the k-th component of the Gaussian mixture approximation (2.4) at iteration t and stage s.

For $t = 0$, we have $\lim_{s\to\infty} \mathbb{P}(\|\hat{\gamma}_k^{s,0} - \hat{\gamma}_k^{s+1,0}\| > \delta) = 0$, because we initialize the EM algorithm at the same values for each run. Next, assume that (A.22) holds for t iteration. We need to prove that

$$\lim_{s\to\infty} \mathbb{P}(\|\hat{\gamma}_k^{s,t+1} - \hat{\gamma}_k^{s+1,t+1}\| > \delta) = 0. \tag{A.23}$$

The E-step of EM algorithm is given by

$$\mathbb{P}(Z_i = k|\boldsymbol{\theta}_i) = \frac{w_k^{s,t}\mathcal{N}(\boldsymbol{\theta}_i^s|\boldsymbol{\mu}_k^{s,t}, \Sigma_k^{s,t})}{\sum_{k'=1}^K w_{k'}^{s,t}\mathcal{N}(\boldsymbol{\theta}_i^s|\boldsymbol{\mu}_{k'}^{s,t}, \Sigma_{k'}^{s,t})} = h_i^{s,t}(k), \tag{A.24}$$

where $\boldsymbol{\theta}_i^s$ is the i-th sample in s stage. The M-step updates are given by

$$\boldsymbol{\mu}_k^{s,t+1} = \frac{\sum_{i=1}^{N_s} h_i^{s,t}(k)\boldsymbol{\theta}_i^s}{N_k^{s,t}}, \tag{A.25}$$

$$\boldsymbol{\Sigma}_k^{s,t+1} = \frac{\sum_{i=1}^{N_s} h_i^{s,t}(k)(\boldsymbol{\theta}_i^s - \boldsymbol{\mu}_k^{s,t})(\boldsymbol{\theta}_i^s - \boldsymbol{\mu}_k^{s,t})^T}{N_k^{s,t}}, \tag{A.26}$$

$$w_k^{s,t+1} = \frac{N_k^{s,t}}{N_s}, \tag{A.27}$$

where $N_k^{s,t} = \sum_{i=1}^{N} h_i^{s,t}(k)$ can be interpreted as the effective number of samples assigned to component k. For any $\delta > 0$, we have

$$\mathbb{P}(\|\boldsymbol{\mu}_k^{s,t+1} - \boldsymbol{\mu}_k^{s+1,t+1}\| > \delta) = \mathbb{P}(\|\frac{1}{N_k^{s,t}}\sum_{i=1}^{N_s} h_i^{s,t}(k)\boldsymbol{\theta}_i^s - \frac{1}{N_k^{s+1,t}}\sum_{i=1}^{N_{s+1}} h_i^{s+1,t}(k)\boldsymbol{\theta}_i^{s+1}\| > \delta) \tag{A.28}$$

$$= \mathbb{P}(\|\frac{1}{N_k^{s,t}}\sum_{i=1}^{N_s} h_i^{s,t}(k)\boldsymbol{\theta}_i^s - \frac{1}{N_k^{s+1,t}}\sum_{i=1}^{N_s} h_i^{s+1,t}(k)\boldsymbol{\theta}_i^s - \frac{1}{N_k^{s+1,t}}\sum_{i=N_s+1}^{N_{s+1}} h_i^{s+1,t}(k)\boldsymbol{\theta}_i^s\| > \delta) \tag{A.29}$$

$$\leq \frac{1}{\delta}\mathbb{E}(\|\sum_{i=1}^{N_s}(\frac{1}{N_k^{s,t}}h_i^{s,t}(k) - \frac{1}{N_k^{s+1,t}}h_i^{s+1,t}(k))\boldsymbol{\theta}_i^s\|) + \frac{1}{\delta}\mathbb{E}(\|\frac{1}{N_k^{s+1,t}}\sum_{i=N_s+1}^{N_{s+1}} h_i^{s+1,t}(k)\boldsymbol{\theta}_i^s\|), \tag{A.30}$$

where (A.30) results from the triangle and Markov inequalities. Since $h_i^{s,t}(k)$ and $h_i^{s+1,t}(k)$ are continuous functions of $\gamma^{s,t}$ and $\gamma^{s+1,t}$, respectively, the continuous mapping theorem gives

$$\lim_{s\to\infty} \mathbb{P}(\|h_i^{s,t}(k) - h_i^{s+1,t}(k)\| > \delta) = 0. \tag{A.31}$$

Thus,

$$\mathbb{E}(\|\sum_{i=1}^{N_s}(\frac{1}{N_k^{s,t}}h_i^{s,t}(k) - \frac{1}{N_k^{s+1,t}}h_i^{s+1,t}(k))\boldsymbol{\theta}_i^s\|) \to 0, \tag{A.32}$$

and

$$\mathbb{E}(\|\frac{1}{N_k^{s+1,t}}\sum_{i=N_s+1}^{N_{s+1}} h_i^{s+1,t}(k)\boldsymbol{\theta}_i^s\|) \to 0, \tag{A.33}$$

as $s \to \infty$, $N_s \to \infty$. Therefore,

$$\lim_{s\to\infty} \mathbb{P}(\|\boldsymbol{\mu}_k^{s,t+1} - \boldsymbol{\mu}_k^{s+1,t+1}\| > \delta) = 0 \tag{A.34}$$

Analogous arguments apply for $\boldsymbol{\Sigma}_k$ and w_k, and hence

$$\lim_{s\to\infty} \mathbb{P}(\|\hat{\gamma}_k^{s,t+1} - \hat{\gamma}_k^{s+1,t+1}\| > \delta) = 0, \tag{A.35}$$

for $k = 1, \ldots, K$. Thus, by induction, (A.22) holds for all t, and all $\delta > 0$.

In any stage s, the EM algorithm is run until either $\|\gamma_k^{s,t+1} - \gamma_k^{s,t}\| < \epsilon_s$ or $t \geq t_{\max}$, where $t_{\max}$ is the maximum number of iterations allowed. Without loss of generality, suppose that it is run for t_1 and t_2 iterations in stages s and $s+1$, respectively, and that $t_1 \leq t_2$. Since (A.22) holds for all t, and all $\delta > 0$, we have

$$\lim_{s\to\infty} \mathbb{P}(\|\gamma_k^{s,t_2} - \gamma_k^{s+1,t_2}\| > \delta_1) = 0. \tag{A.36}$$

Thus,

$$\lim_{s\to\infty} \mathbb{P}(\|\gamma_k^{s,t_1} - \gamma_k^{s+1,t_2}\| > \delta) \tag{A.37}$$

$$\leq \lim_{s\to\infty} \mathbb{P}(\|\gamma_k^{s,t_1} - \gamma_k^{s,t_1+1}\| + \|\gamma_k^{s,t_1+1} - \gamma_k^{s,t_1+2}\| + \cdots + \|\gamma_k^{s,t_2} - \gamma_k^{s+1,t_2}\| > \delta) \tag{A.38}$$

$$\leq \lim_{s\to\infty} \mathbb{P}((t_2 - t_1)\epsilon + \|\gamma_k^{s,t_2} - \gamma_k^{s+1,t_2}\| > \delta) \tag{A.39}$$

$$= \lim_{s\to\infty} \mathbb{P}(\|\gamma_k^{s,t_2} - \gamma_k^{s+1,t_2}\| > \delta_1) = 0. \tag{A.40}$$

Therefore,

$$\lim_{s\to\infty} \mathbb{P}(\|\hat{\gamma}^s - \hat{\gamma}^{s+1}\| > \delta) = 0 \tag{A.41}$$

□

Proof of Lemma 3. We begin by proving that condition C1 (minorization) holds. Since

$$\int_A \int_{\mathbb{R}^d} \kappa(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0) m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1\,\mathrm{d}\boldsymbol{\theta}_2 = \int \sum_{\psi,\psi^*=1}^{K} \kappa(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0) I(\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1) \in A) p(\psi^*, \psi|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1 \tag{A.42}$$

$$= \int \sum_{\psi,\psi^*=1}^{K} \kappa(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0) I(\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1) \in A) \varpi(\psi|\boldsymbol{\theta}_1) \times \frac{\varpi(\psi^*|\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1))\pi(\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1))|\mathcal{S}_{\psi^*}|}{\sum_{\widetilde{\psi}=1}^{K} \varpi(\widetilde{\psi}|\mathcal{G}_{\widetilde{\psi},\psi}(\boldsymbol{\theta}_1))\pi(\mathcal{G}_{\widetilde{\psi},\psi}(\boldsymbol{\theta}_1))|\mathcal{S}_{\widetilde{\psi}}|}\,\mathrm{d}\boldsymbol{\theta}_1 \tag{A.43}$$

$$\geq \int_A \sum_{\psi,\psi^*=1}^{K} \kappa(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0) \varpi(\psi|\boldsymbol{\theta}_1) \times \frac{\varpi(\psi^*|\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1))\pi(\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1))|\mathcal{S}_{\psi^*}|}{\sum_{\widetilde{\psi}=1}^{K} \varpi(\widetilde{\psi}|\mathcal{G}_{\widetilde{\psi},\psi}(\boldsymbol{\theta}_1))\pi(\mathcal{G}_{\widetilde{\psi},\psi}(\boldsymbol{\theta}_1))|\mathcal{S}_{\widetilde{\psi}}|}\,\mathrm{d}\boldsymbol{\theta}_1, \tag{A.44}$$

where the last equality uses that $A \subset \mathcal{G}^{-1}_{\psi^*,\psi}(A)$ because for any $\boldsymbol{\theta}_1 \in \mathcal{G}^{-1}_{\psi^*,\psi}(A)$, $\boldsymbol{\theta}_1 = \mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1) \in A$ when $\psi = \psi^*$.

Let $J(\boldsymbol{\theta}_1) = \sum_{\psi,\psi^*=1}^{K} \varpi(\psi|\boldsymbol{\theta}_1) \times \frac{\varpi(\psi^*|\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1))\pi(\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1))|\mathcal{S}_{\psi^*}|}{\sum_{\widetilde{\psi}=1}^{K} \varpi(\widetilde{\psi}|\mathcal{G}_{\widetilde{\psi},\psi}(\boldsymbol{\theta}_1))\pi(\mathcal{G}_{\widetilde{\psi},\psi}(\boldsymbol{\theta}_1))|\mathcal{S}_{\widetilde{\psi}}|}$. We have $J(\boldsymbol{\theta}_1) > 0$ for all $\boldsymbol{\theta}_1 \in C$, where C is one closed ball that contains A. Then $\delta_1 = \inf_{\boldsymbol{\theta}_1 \in C} J(\boldsymbol{\theta}_1) > 0$ and

$$\int_A \int_{\mathbb{R}^d} \kappa(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0) m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1\,\mathrm{d}\boldsymbol{\theta}_2 \geq \int_A \delta_1 \kappa(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)\,\mathrm{d}\boldsymbol{\theta}_1. \tag{A.45}$$

Following lemma 4.1 of Atchadé (2006), there is a probability measure ν and $\delta_2 > 0$ such that

$$\int_A \kappa(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)\,\mathrm{d}\boldsymbol{\theta}_1 \geq \delta_2 \nu(A). \tag{A.46}$$

Hence, we have

$$\int_A \int_{\mathbb{R}^d} \kappa(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0) m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1\,\mathrm{d}\boldsymbol{\theta}_2 \geq \delta_1\delta_2 \nu(A). \tag{A.47}$$

Next, we prove that condition C2 (Simultaneous drift) holds. Define $V(\boldsymbol{\theta}) = c_v \pi^{-\alpha}(\boldsymbol{\theta})$, where $\alpha \in (0,1)$ and $c_v = \sup_{\boldsymbol{\theta}} \pi^{\alpha}(\boldsymbol{\theta})$, so that $\inf_{\boldsymbol{\theta}} V(\boldsymbol{\theta}) = 1$. Following Lemma 3.5 of Jarner and Hansen (2000), we only need to prove that the following two conditions hold:

$$\sup_{\boldsymbol{\theta}_0 \in \mathcal{X}} \frac{\int_{\mathbb{R}^n} V(\boldsymbol{\theta}_2) P_{\hat{\gamma}}(\boldsymbol{\theta}_0, \mathrm{d}\boldsymbol{\theta}_2)}{V(\boldsymbol{\theta}_0)} < \infty, \tag{A.48}$$

$$\limsup_{\|\boldsymbol{\theta}_0\|\to\infty}\sup_{\gamma\in\Gamma}\frac{\int_{\mathbb{R}^n}V(\boldsymbol{\theta}_2)P_\gamma(\boldsymbol{\theta}_0,\,\mathrm{d}\boldsymbol{\theta}_2)}{V(\boldsymbol{\theta}_0)}<1. \tag{A.49}$$

Recall that $\kappa(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0) = \alpha(\boldsymbol{\theta}_0,\boldsymbol{\theta}_1)q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0) + (1-r(\boldsymbol{\theta}_0))\delta(\boldsymbol{\theta}_0-\boldsymbol{\theta}_1)$, where $\alpha(\boldsymbol{\theta}_0,\boldsymbol{\theta}_1) = \min\{1,\frac{\pi(\boldsymbol{\theta}_1)q(\boldsymbol{\theta}_0|\boldsymbol{\theta}_1)}{\pi(\boldsymbol{\theta}_0)q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)}\}$ and $r(\boldsymbol{\theta}_0)=\int\alpha(\boldsymbol{\theta}_0,\boldsymbol{\theta}')q(\boldsymbol{\theta}'|\boldsymbol{\theta}_0)\,\mathrm{d}\boldsymbol{\theta}'$. To prove condition (A.48), we expand it as

$$\frac{\int_{\mathbb{R}^n}V(\boldsymbol{\theta}_2)P_{\hat{\gamma}}(\boldsymbol{\theta}_0,\,\mathrm{d}\boldsymbol{\theta}_2)}{V(\boldsymbol{\theta}_0)} \tag{A.50}$$

$$=\int\int_{A(\boldsymbol{\theta}_0)}\frac{\pi^{-\alpha}(\boldsymbol{\theta}_2)}{\pi^{-\alpha}(\boldsymbol{\theta}_0)}q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1\,\mathrm{d}\boldsymbol{\theta}_2 \tag{A.51}$$

$$+\int\int_{R(\boldsymbol{\theta}_0)}\frac{\pi^{-\alpha}(\boldsymbol{\theta}_2)}{\pi^{-\alpha}(\boldsymbol{\theta}_0)}\frac{\pi(\boldsymbol{\theta}_1)q(\boldsymbol{\theta}_0|\boldsymbol{\theta}_1)}{\pi(\boldsymbol{\theta}_0)q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)}q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1\,\mathrm{d}\boldsymbol{\theta}_2 \tag{A.52}$$

$$+\int_{R(\boldsymbol{\theta}_0)}\big(1-\frac{\pi(\boldsymbol{\theta}')q(\boldsymbol{\theta}_0|\boldsymbol{\theta}')}{\pi(\boldsymbol{\theta}_0)q(\boldsymbol{\theta}'|\boldsymbol{\theta}_0)}\big)q(\boldsymbol{\theta}'|\boldsymbol{\theta}_0)\,\mathrm{d}\boldsymbol{\theta}'\int m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_0)\frac{\pi^{-\alpha}(\boldsymbol{\theta}_2)}{\pi^{-\alpha}(\boldsymbol{\theta}_0)}\,\mathrm{d}\boldsymbol{\theta}_2 \tag{A.53}$$

where

$$A(\boldsymbol{\theta}_0)=\{\boldsymbol{\theta}_1\in\mathbb{R}^n,\pi(\boldsymbol{\theta}_1)q(\boldsymbol{\theta}_0|\boldsymbol{\theta}_1)\geq\pi(\boldsymbol{\theta}_0)q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)\}$$

and $R(\boldsymbol{\theta}_0)=A(\boldsymbol{\theta}_0)^c$.

Because $q(\boldsymbol{\theta}|\boldsymbol{\theta}_0)$ is Gaussian and for any $\boldsymbol{\theta}_0\in\mathcal{X}$, we can find $0<\epsilon_1<\epsilon_2<\infty$ and constant K_L and K_U such that

$$K_L g_{\epsilon_1}(\boldsymbol{\theta}_2-\boldsymbol{\theta}_0)\leq q(\boldsymbol{\theta}_2|\boldsymbol{\theta}_0)\leq K_U g_{\epsilon_2}(\boldsymbol{\theta}_2-\boldsymbol{\theta}_0), \tag{A.54}$$

where g_a is the probability density function of a d-dimensional Gaussian distribution with mean $\mathbf{0}$ and covariance matrix $a\mathbf{I}_d$, and $a>0$. Then we can bound $q^\alpha(\boldsymbol{\theta}_0|\boldsymbol{\theta}_1)q^{1-\alpha}(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)$ when $\boldsymbol{\theta}_1\in A(\boldsymbol{\theta}_0)$, i.e.,

$$\frac{\pi^{-\alpha}(\boldsymbol{\theta}_1)}{\pi^{-\alpha}(\boldsymbol{\theta}_0)}q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)\leq q^\alpha(\boldsymbol{\theta}_0|\boldsymbol{\theta}_1)q^{1-\alpha}(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)\leq K_U g_{\epsilon_2}(\boldsymbol{\theta}_1-\boldsymbol{\theta}_0) \tag{A.55}$$

Similarly, when $\boldsymbol{\theta}_1 \in R(\boldsymbol{\theta}_0)$,

$$\frac{\pi^{-\alpha}(\boldsymbol{\theta}_1)}{\pi^{-\alpha}(\boldsymbol{\theta}_0)}\frac{\pi(\boldsymbol{\theta}_1)q(\boldsymbol{\theta}_0|\boldsymbol{\theta}_1)}{\pi(\boldsymbol{\theta}_0)q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)}q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0) \leq q^{1-\alpha}(\boldsymbol{\theta}_0|\boldsymbol{\theta}_1)q^{\alpha}(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0) \leq K_U g_{\epsilon_2}(\boldsymbol{\theta}_1 - \boldsymbol{\theta}_0) \tag{A.56}$$

For the first term of (A.50),

$$\int\int_{A(\boldsymbol{\theta}_0)} \frac{\pi^{-\alpha}(\boldsymbol{\theta}_2)}{\pi^{-\alpha}(\boldsymbol{\theta}_0)} q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0) m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1\,\mathrm{d}\boldsymbol{\theta}_2 \tag{A.57}$$

$$= \int\int_{A(\boldsymbol{\theta}_0)} \frac{\pi^{-\alpha}(\boldsymbol{\theta}_1)}{\pi^{-\alpha}(\boldsymbol{\theta}_0)} q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0) \frac{\pi^{-\alpha}(\boldsymbol{\theta}_2)}{\pi^{-\alpha}(\boldsymbol{\theta}_1)} m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1\,\mathrm{d}\boldsymbol{\theta}_2 \tag{A.58}$$

$$\leq \int\int_{A(\boldsymbol{\theta}_0)} K_U g_{\epsilon_2}(\boldsymbol{\theta}_1 - \boldsymbol{\theta}_0)\pi^{\alpha}(\boldsymbol{\theta}_1)\pi^{-\alpha}(\boldsymbol{\theta}_2)\pi(\boldsymbol{\theta}_2)\tilde{m}(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1\,\mathrm{d}\boldsymbol{\theta}_2 \tag{A.59}$$

$$= \int\int_{A(\boldsymbol{\theta}_0)} K_U g_{\epsilon_2}(\boldsymbol{\theta}_1 - \boldsymbol{\theta}_0)\pi^{\alpha}(\boldsymbol{\theta}_1)\pi^{1-\alpha}(\boldsymbol{\theta}_2)\tilde{m}(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1\,\mathrm{d}\boldsymbol{\theta}_2 \tag{A.60}$$

where $m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1) = \pi(\boldsymbol{\theta}_2)\tilde{m}(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)$. Now, our goal is to prove (A.60) is smaller than ∞. We first prove that

$$\pi^{\alpha}(\boldsymbol{\theta}_1)\int \pi^{1-\alpha}(\boldsymbol{\theta}_2)\tilde{m}(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_2 < \infty \tag{A.61}$$

for any $\boldsymbol{\theta}_1$.

Let $g(\boldsymbol{\theta}_1) = \int \pi^{1-\alpha}(\boldsymbol{\theta}_2)\tilde{m}(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_2$. Similarly with the (A.8), for any Borel set A of $\mathbb{R}^d$, we have

$$\int_A g(\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1 = \int \sum_{\psi,\psi^*=1}^{K} I(\mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2) \in A)p(\psi^*,\psi|\mathcal{G}^{-1}_{\psi^*,\psi}(\boldsymbol{\theta}_2))\pi^{-\alpha}(\boldsymbol{\theta}_2)\,\mathrm{d}\boldsymbol{\theta}_2 \tag{A.62}$$

$$= \int \sum_{\psi,\psi^*=1}^{K} I(\boldsymbol{\theta}_1 \in A)p(\psi^*,\psi|\boldsymbol{\theta}_1)\pi^{-\alpha}(\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1)) \times \left|\frac{\partial \mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1)}{\partial \boldsymbol{\theta}_1}\right| \mathrm{d}\boldsymbol{\theta}_1 \tag{A.63}$$

$$= \int_A \sum_{\psi,\psi^*=1}^{K} p(\psi^*,\psi|\boldsymbol{\theta}_1)\pi^{-\alpha}(\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1)) \times |\mathcal{S}_{\psi^*}| \times |\mathcal{S}_{\psi}|^{-1}\,\mathrm{d}\boldsymbol{\theta}_1. \tag{A.64}$$

In the above, the second equality applies the change of variable $\boldsymbol{\theta}_2 = \mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1)$ for the integral.

Therefore,

$$g(\boldsymbol{\theta}_1) = \sum_{\psi,\psi^*=1}^{K} \pi^{-\alpha}(\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1))p(\psi^*,\psi|\boldsymbol{\theta}_1) \times |\mathcal{S}_{\psi^*}| \times |\mathcal{S}_{\psi}|^{-1} \tag{A.65}$$

$$= \sum_{\psi,\psi^*=1}^{K} \pi^{-\alpha}(\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1))p(\psi|\boldsymbol{\theta}_1)p(\psi^*|\boldsymbol{\theta}_1,\psi) \times |\mathcal{S}_{\psi^*}| \times |\mathcal{S}_{\psi}|^{-1} \tag{A.66}$$

$$= \sum_{\psi^*,\psi=1}^{K} \pi^{-\alpha}(\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1))\varpi(\psi|\boldsymbol{\theta}_1) \times \frac{\varpi(\psi^*|\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1))\pi(\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1))|\mathcal{S}_{\psi^*}|^2|\mathcal{S}_{\psi}|^{-1}}{\sum_{\widetilde{\psi}=1}^{K}\varpi(\widetilde{\psi}|\mathcal{G}_{\widetilde{\psi},\psi}(\boldsymbol{\theta}_1))\pi(\mathcal{G}_{\widetilde{\psi},\psi}(\boldsymbol{\theta}_1))|\mathcal{S}_{\widetilde{\psi}}|}. \tag{A.67}$$

Hence,

$$\pi^{\alpha}(\boldsymbol{\theta}_1)g(\boldsymbol{\theta}_1) = \sum_{\psi^*,\psi=1}^{K} \frac{\varpi(\psi|\boldsymbol{\theta}_1)\varpi(\psi^*|\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1))\pi^{\alpha}(\boldsymbol{\theta}_1)\pi^{1-\alpha}(\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1))|\mathcal{S}_{\psi^*}|^2|\mathcal{S}_{\psi}|^{-1}}{\sum_{\widetilde{\psi}=1}^{K}\varpi(\widetilde{\psi}|\mathcal{G}_{\widetilde{\psi},\psi}(\boldsymbol{\theta}_1))\pi(\mathcal{G}_{\widetilde{\psi},\psi}(\boldsymbol{\theta}_1))|\mathcal{S}_{\widetilde{\psi}}|}. \tag{A.68}$$

Let $|\mathcal{S}_{\max}| = \max_{\psi^*,\psi\in\{1...,K\}} \frac{|\mathcal{S}_{\psi^*}|^2}{|\mathcal{S}_{\psi}|}$ and $M_1 = \max_{\psi\in\{1...,K\}} \frac{|\mathcal{S}_{\max}|}{|\mathcal{S}_{\psi}|}$. Note that $\varpi(\psi|\boldsymbol{\theta}_1) \leq 1$ and $\varpi(\psi^*|\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1)) \leq 1$. Since there exist $\widetilde{\psi} = \psi$ such that $\mathcal{G}_{\widetilde{\psi},\psi}(\boldsymbol{\theta}_1) = \boldsymbol{\theta}_1$, when

$$\pi(\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1)) \leq \pi(\boldsymbol{\theta}_1), \tag{A.69}$$

we have

$$\pi^{\alpha}(\boldsymbol{\theta}_1)g(\boldsymbol{\theta}_1) \leq \sum_{\psi^*,\psi=1}^{K} \frac{\varpi(\psi|\boldsymbol{\theta}_1)\pi^{\alpha}(\boldsymbol{\theta}_1)\pi^{1-\alpha}(\boldsymbol{\theta}_1)|\mathcal{S}_{\psi^*}|^2|\mathcal{S}_{\psi}|^{-1}}{\sum_{\widetilde{\psi}=1}^{K}\varpi(\widetilde{\psi}|\mathcal{G}_{\widetilde{\psi},\psi}(\boldsymbol{\theta}_1))\pi(\mathcal{G}_{\widetilde{\psi},\psi}(\boldsymbol{\theta}_1))|\mathcal{S}_{\widetilde{\psi}}|} \tag{A.70}$$

$$\leq \sum_{\psi^*,\psi=1}^{K} \frac{\varpi(\psi|\boldsymbol{\theta}_1)\pi(\boldsymbol{\theta}_1)|\mathcal{S}_{\psi^*}|^2|\mathcal{S}_{\psi}|^{-1}}{\sum_{\widetilde{\psi}=1}^{K}\varpi(\widetilde{\psi}|\mathcal{G}_{\widetilde{\psi},\psi}(\boldsymbol{\theta}_1))\pi(\mathcal{G}_{\widetilde{\psi},\psi}(\boldsymbol{\theta}_1))|\mathcal{S}_{\widetilde{\psi}}|} \tag{A.71}$$

$$\leq \sum_{\psi^*,\psi=1}^{K} \frac{\varpi(\psi|\mathcal{G}_{\psi,\psi}(\boldsymbol{\theta}_1))\pi(\mathcal{G}_{\psi,\psi}(\boldsymbol{\theta}_1))|\mathcal{S}_{\psi}|M_1}{\sum_{\widetilde{\psi}=1}^{K}\varpi(\widetilde{\psi}|\mathcal{G}_{\widetilde{\psi},\psi}(\boldsymbol{\theta}_1))\pi(\mathcal{G}_{\widetilde{\psi},\psi}(\boldsymbol{\theta}_1))|\mathcal{S}_{\widetilde{\psi}}|} \leq K^2 M_1. \tag{A.72}$$

Similarly, when

$$\pi(\boldsymbol{\theta}_1) \leq \pi(\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1)), \tag{A.73}$$

we can also obtain that

$$\pi^{\alpha}(\boldsymbol{\theta}_1)g(\boldsymbol{\theta}_1) \leq \sum_{\psi^*,\psi=1}^{K} \frac{\varpi(\psi|\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1))\pi(\mathcal{G}_{\psi^*,\psi}(\boldsymbol{\theta}_1))|\mathcal{S}_{\psi^*}|M_1}{\sum_{\widetilde{\psi}=1}^{K}\varpi(\widetilde{\psi}|\mathcal{G}_{\widetilde{\psi},\psi}(\boldsymbol{\theta}_1))\pi(\mathcal{G}_{\widetilde{\psi},\psi}(\boldsymbol{\theta}_1))|\mathcal{S}_{\widetilde{\psi}}|} \leq K^2 M_1. \tag{A.74}$$

Therefore, we can conclude that

$$\int \pi^{\alpha}(\boldsymbol{\theta}_1)\pi^{1-\alpha}(\boldsymbol{\theta}_2)\tilde{m}(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_2 \leq K^2 M_1 < \infty. \tag{A.75}$$

Hence, (A.57) is bounded, i.e.,

$$\int\int_{A(\boldsymbol{\theta}_0)} \frac{\pi^{-\alpha}(\boldsymbol{\theta}_2)}{\pi^{-\alpha}(\boldsymbol{\theta}_0)} q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1\,\mathrm{d}\boldsymbol{\theta}_2 \tag{A.76}$$

$$\leq \int\int_{A(\boldsymbol{\theta}_0)} K_U g_{\epsilon_2}(\boldsymbol{\theta}_1 - \boldsymbol{\theta}_0)\pi^{\alpha}(\boldsymbol{\theta}_1)\pi^{1-\alpha}(\boldsymbol{\theta}_2)\tilde{m}(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1\,\mathrm{d}\boldsymbol{\theta}_2 \tag{A.77}$$

$$\leq \int_{A(\boldsymbol{\theta}_0)} K_U g_{\epsilon_2}(\boldsymbol{\theta}_1 - \boldsymbol{\theta}_0)K^2 M_1\,\mathrm{d}\boldsymbol{\theta}_1 < \infty. \tag{A.78}$$

Similarly for the second term of (A.50), with (A.75) holds,

$$\int\int_{R(\boldsymbol{\theta}_0)} \frac{\pi^{-\alpha}(\boldsymbol{\theta}_2)}{\pi^{-\alpha}(\boldsymbol{\theta}_0)} \frac{\pi(\boldsymbol{\theta}_1)q(\boldsymbol{\theta}_0|\boldsymbol{\theta}_1))}{\pi(\boldsymbol{\theta}_0)q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)} q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1\,\mathrm{d}\boldsymbol{\theta}_2 \tag{A.79}$$

$$= \int\int_{R(\boldsymbol{\theta}_0)} \frac{\pi^{-\alpha}(\boldsymbol{\theta}_1)}{\pi^{-\alpha}(\boldsymbol{\theta}_0)} \frac{\pi(\boldsymbol{\theta}_1)q(\boldsymbol{\theta}_0|\boldsymbol{\theta}_1))}{\pi(\boldsymbol{\theta}_0)q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)} q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0) \frac{\pi^{-\alpha}(\boldsymbol{\theta}_2)}{\pi^{-\alpha}(\boldsymbol{\theta}_1)} m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1\,\mathrm{d}\boldsymbol{\theta}_2 \tag{A.80}$$

$$\leq \int\int_{R(\boldsymbol{\theta}_0)} K_U g_{\epsilon_2}(\boldsymbol{\theta}_1 - \boldsymbol{\theta}_0)\pi^{\alpha}(\boldsymbol{\theta}_1)\pi^{-\alpha}(\boldsymbol{\theta}_2)\pi(\boldsymbol{\theta}_2)\tilde{m}(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1\,\mathrm{d}\boldsymbol{\theta}_2 \tag{A.81}$$

$$= \int\int_{R(\boldsymbol{\theta}_0)} K_U g_{\epsilon_2}(\boldsymbol{\theta}_1 - \boldsymbol{\theta}_0)\pi^{\alpha}(\boldsymbol{\theta}_1)\pi^{1-\alpha}(\boldsymbol{\theta}_2)\tilde{m}(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1\,\mathrm{d}\boldsymbol{\theta}_2 < \infty \tag{A.82}$$

For the third term of (A.50), following the equation (A.75), we have

$$\int \pi^{\alpha}(\boldsymbol{\theta}_0)\pi^{1-\alpha}(\boldsymbol{\theta}_2)\tilde{m}(\boldsymbol{\theta}_2|\boldsymbol{\theta}_0)\,\mathrm{d}\boldsymbol{\theta}_2 < \infty. \tag{A.83}$$

Then

$$\int_{R(\boldsymbol{\theta}_0)}(1-\frac{\pi(\boldsymbol{\theta}^{'})q(\boldsymbol{\theta}_0|\boldsymbol{\theta}^{'})}{\pi(\boldsymbol{\theta}_0)q(\boldsymbol{\theta}^{'}|\boldsymbol{\theta}_0)})q(\boldsymbol{\theta}^{'}|\boldsymbol{\theta}_0)\,\mathrm{d}\boldsymbol{\theta}^{'}\int m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_0)\frac{\pi^{-\alpha}(\boldsymbol{\theta}_2)}{\pi^{-\alpha}(\boldsymbol{\theta}_0)}\,\mathrm{d}\boldsymbol{\theta}_2 \tag{A.84}$$

$$\leq \int_{R(\boldsymbol{\theta}_0)} q(\boldsymbol{\theta}^{'}|\boldsymbol{\theta}_0)\,\mathrm{d}\boldsymbol{\theta}^{'}\int \pi^{\alpha}(\boldsymbol{\theta}_0)\pi^{1-\alpha}(\boldsymbol{\theta}_2)\tilde{m}(\boldsymbol{\theta}_2|\boldsymbol{\theta}_0)\,\mathrm{d}\boldsymbol{\theta}_2<\infty. \tag{A.85}$$

To prove the second condition (A.49), we first make the expansion of $\frac{\int_{\mathbb{R}^n}V(\boldsymbol{\theta}_2)P_{\hat{\gamma}}(\boldsymbol{\theta}_0,\mathrm{d}\boldsymbol{\theta}_2)}{V(\boldsymbol{\theta}_0)}$ same with (A.50). Then, following the proof of Lemma 6.2 in Atchadé (2006), we have

$$\limsup_{\|\boldsymbol{\theta}_0\|\to\infty}\sup_{\gamma\in\Gamma}\int_{A(\boldsymbol{\theta}_0)}\frac{\pi^{-\alpha}(\boldsymbol{\theta}_1)}{\pi^{-\alpha}(\boldsymbol{\theta}_0)}q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)\,\mathrm{d}\boldsymbol{\theta}_1 \tag{A.86}$$

$$+\limsup_{\|\boldsymbol{\theta}_0\|\to\infty}\sup_{\gamma\in\Gamma}\int_{R(\boldsymbol{\theta}_0)}\frac{\pi^{-\alpha}(\boldsymbol{\theta}_1)}{\pi^{-\alpha}(\boldsymbol{\theta}_0)}\frac{\pi(\boldsymbol{\theta}_1)q(\boldsymbol{\theta}_0|\boldsymbol{\theta}_1)}{\pi(\boldsymbol{\theta}_0)q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)}q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)\,\mathrm{d}\boldsymbol{\theta}_1=0. \tag{A.87}$$

With (A.75) holds, we can claim

$$\limsup_{\|\boldsymbol{\theta}_0\|\to\infty}\sup_{\gamma\in\Gamma}\int\int_{A(\boldsymbol{\theta}_0)}\frac{\pi^{-\alpha}(\boldsymbol{\theta}_1)}{\pi^{-\alpha}(\boldsymbol{\theta}_0)}q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)\pi^{\alpha}(\boldsymbol{\theta}_1)\pi^{1-\alpha}(\boldsymbol{\theta}_2)\tilde{m}(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1\,\mathrm{d}\boldsymbol{\theta}_2 \tag{A.88}$$

$$+\limsup_{\|\boldsymbol{\theta}_0\|\to\infty}\sup_{\gamma\in\Gamma}\int\int_{R(\boldsymbol{\theta}_0)}\frac{\pi^{-\alpha}(\boldsymbol{\theta}_1)}{\pi^{-\alpha}(\boldsymbol{\theta}_0)}\frac{\pi(\boldsymbol{\theta}_1)q(\boldsymbol{\theta}_0|\boldsymbol{\theta}_1)}{\pi(\boldsymbol{\theta}_0)q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)}q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0)\boldsymbol{\theta}_2\pi^{\alpha}(\boldsymbol{\theta}_1)\pi^{1-\alpha}(\boldsymbol{\theta}_2)\tilde{m}(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1\,\mathrm{d}\boldsymbol{\theta}_2=0 \tag{A.89}$$

For the third term of (A.50), we have

$$\int_{R(\boldsymbol{\theta}_0)}(1-\frac{\pi(\boldsymbol{\theta}^{'})q(\boldsymbol{\theta}_0|\boldsymbol{\theta}^{'})}{\pi(\boldsymbol{\theta}_0)q(\boldsymbol{\theta}^{'}|\boldsymbol{\theta}_0)})q(\boldsymbol{\theta}^{'}|\boldsymbol{\theta}_0)\,\mathrm{d}\boldsymbol{\theta}^{'}\int m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_0)\frac{\pi^{-\alpha}(\boldsymbol{\theta}_2)}{\pi^{-\alpha}(\boldsymbol{\theta}_0)}\,\mathrm{d}\boldsymbol{\theta}_2 \tag{A.90}$$

$$\leq \int_{R(\boldsymbol{\theta}_0)} q(\boldsymbol{\theta}^{'}|\boldsymbol{\theta}_0)\,\mathrm{d}\boldsymbol{\theta}^{'}\int \pi^{\alpha}(\boldsymbol{\theta}_0)\pi^{1-\alpha}(\boldsymbol{\theta}_2)\tilde{m}(\boldsymbol{\theta}_2|\boldsymbol{\theta}_0)\,\mathrm{d}\boldsymbol{\theta}_2 \tag{A.91}$$

$$\leq \pi^{\alpha}(\boldsymbol{\theta}_0)\int \pi^{1-\alpha}(\boldsymbol{\theta}_2)\tilde{m}(\boldsymbol{\theta}_2|\boldsymbol{\theta}_0)\,\mathrm{d}\boldsymbol{\theta}_2 \tag{A.92}$$

With (A.75) holds and when $\|\boldsymbol{\theta}_0\|\to\infty$, $\pi^{\alpha}(\boldsymbol{\theta}_0)$ will goes to 0. Therefore,

$$\limsup_{\|\boldsymbol{\theta}_0\|\to\infty}\sup_{\gamma\in\Gamma}\int_{R(\boldsymbol{\theta}_0)}(1-\frac{\pi(\boldsymbol{\theta}^{'})q(\boldsymbol{\theta}_0|\boldsymbol{\theta}^{'})}{\pi(\boldsymbol{\theta}_0)q(\boldsymbol{\theta}^{'}|\boldsymbol{\theta}_0)})q(\boldsymbol{\theta}^{'}|\boldsymbol{\theta}_0)\,\mathrm{d}\boldsymbol{\theta}^{'}\int m(\boldsymbol{\theta}_2|\boldsymbol{\theta}_0)\frac{\pi^{-\alpha}(\boldsymbol{\theta}_2)}{\pi^{-\alpha}(\boldsymbol{\theta}_0)}\,\mathrm{d}\boldsymbol{\theta}_2<1 \tag{A.93}$$

□

Proof of Theorem 1. Since Lemmas 2 ensure the diminishing adaptation condition and lemma 3 ensure SSAGE condition. With lemma 1, 2 and 3 in hand, we can claim the ergodicity of adaptive Warp-U algorithm by the framework of Roberts and Rosenthal (2007). □

A.3 Proof of Theorem on Relative Asymptotic Variance (Theorem 2)

Proof of Theorem 2. Meng and Wong (1996) showed that under the i.i.d. assumption, the asymptotic variance of $\hat{\lambda}_{\text{BS}} = \log \hat{r}_{\text{BS}}$ is

$$\left(\frac{1}{n_1} + \frac{1}{n_2}\right)\left[(1 - H_A(p_1, p_2))^{-1} - 1\right] + \mathbf{o}\left(\frac{1}{n_1} + \frac{1}{n_2}\right). \tag{A.94}$$

Following Nielsen and Nock (2013), we express $H_A(p_1, p_2)$ and $H_A(p_{1,k}, p_2)$ as $H_A(p_1, p_2) = \frac{n_1 n_2}{(n_1+n_2)^2}\chi^2_P(p_2, p_1)$ and $H_A(p_{1,k}, p_2) = \frac{n_{1k} n_2}{(n_{1k}+n_2)^2}\chi^2_P(p_2, p_{1,k})$, for $k = 1, \ldots, K$, where χ^2_P is Pearson Chi-type distances defined as

$$\chi^2_P(p_2, p_1) = \int_{\Theta_1 \cap \Theta_2} \frac{(p_2(\boldsymbol{\theta}) - p_1(\boldsymbol{\theta}))^2}{p_2(\boldsymbol{\theta})}\, \mathrm{d}\boldsymbol{\theta}. \tag{A.95}$$

Let $n_2 = \beta n_1$ and $\tilde{w}_k = \frac{n_{1k}}{n_1}$, $l(y) = \chi^2_P(p_2, y)$, and

$$m(y) = (1 - H_A(y, p_2))^{-1} - 1 = \left(1 - \frac{n_1 n_2}{(n_1 + n_2)^2}\chi^2_P(p_2, y)\right)^{-1} - 1 = \left(1 - \frac{\beta}{(1+\beta)^2} l(y)\right)^{-1} - 1. \tag{A.96}$$

We can obtain the first and second derivative of $m(y)$, i.e.,

$$m^{'}(y) = \frac{\beta}{(1+\beta)^2}\left(1 - \frac{\beta}{(1+\beta)^2} l(y)\right)^{-2} l^{'}(y), \tag{A.97}$$

and

$$m^{''}(y) = \frac{\frac{\beta}{(1+\beta)^2}\left(1 - \frac{\beta}{(1+\beta)^2} l(y)\right) l^{''}(y) + 2\left(\frac{\beta}{(1+\beta)^2} l^{'}(y)\right)^2}{\left(1 - \frac{\beta}{(1+\beta)^2} l(y)\right)^3}. \tag{A.98}$$

Since

$$\mathbb{E}(\frac{n_{1k}}{n_1}) = \int_{\mathbb{R}^d} \pi(\boldsymbol{\theta}) \frac{w_k \phi(\mathcal{S}_k^{-1}(\boldsymbol{\theta} - \boldsymbol{\mu}_k))|\mathcal{S}_k^{-1}|}{\phi_{\text{mix}}(\boldsymbol{\theta})} \, \mathrm{d}\boldsymbol{\theta} = \int \frac{1}{c_1} q(\boldsymbol{\theta}) \frac{w_k \phi(\mathcal{S}_k^{-1}(\boldsymbol{\theta} - \boldsymbol{\mu}_k))|\mathcal{S}_k^{-1}|}{\phi_{\text{mix}}(\boldsymbol{\theta})} \, \mathrm{d}\boldsymbol{\theta} \tag{A.99}$$

$$= \int \frac{1}{c_1} \phi(\tilde{\boldsymbol{\theta}}) \frac{w_k q(\mathcal{S}_k \tilde{\boldsymbol{\theta}} + \boldsymbol{\mu}_k)}{\phi_{\text{mix}}(\mathcal{S}_k \tilde{\boldsymbol{\theta}} + \boldsymbol{\mu}_k)} \, \mathrm{d}\tilde{\boldsymbol{\theta}} = \int \frac{w_k}{c_1} q_{1k}(\tilde{\boldsymbol{\theta}}) \, \mathrm{d}\tilde{\boldsymbol{\theta}} = \frac{w_k c_{1k}}{c_1}, \tag{A.100}$$

where the equality in (A.100) applies the change of variable $\tilde{\boldsymbol{\theta}} = \mathcal{S}_k^{-1}(\boldsymbol{\theta} - \boldsymbol{\mu}_k)$ for the integral, we approximate the weight $\frac{c_{1k} w_k}{c_1}$ by $\frac{n_{1k}}{n_1} = \tilde{w}_k$. Hence, $p_1(\boldsymbol{\theta}) = \sum_{k=1}^K \frac{c_{1k} w_k}{c_1} p_{1k}(\boldsymbol{\theta}) \approx \sum_{k=1}^K \tilde{w}_k p_{1k}(\boldsymbol{\theta})$. To obtain the expression of asymptotic variance, we first obtain the Taylor series for $m(p_{1k})$ at p_1, i.e.,

$$m(p_{1k}) = m(p_1) + m^{'}(p_1)(p_{1k} - p_1) + \frac{m^{''}(\tau_k)(p_{1k} - p_1)^2}{2}, \tag{A.101}$$

where τ_k is some density between p_{1k} and p_1, for $k = 1, \ldots, K$. Hence,

$$\sum_{k=1}^K \tilde{w}_k m(p_{1k}) = m(p_1) + \sum_{k=1}^K \tilde{w}_k \frac{m^{''}(\tau_k)(p_{1k} - p_1)^2}{2}. \tag{A.102}$$

Therefore, writing $\doteq$ to denote asymptotic equality, we express $Var(\hat{\lambda}_{\text{WB}})$ as,

$$Var(\hat{\lambda}_{\text{WB}}) \doteq (\frac{1}{n_1} + \frac{1}{n_2})[(1 - H_A(p_1, p_2))^{-1} - 1] = \frac{1 + \beta}{\beta} \frac{1}{n_1} m(p_1). \tag{A.103}$$

On the other hand, by the delta method, that is $Var[f(x)] = Var(x)(f'(E(x))^2$, we have

$$Var(\hat{\lambda}_{\text{SWB}}) = Var(\log(\hat{r}_{\text{SWB}})) \doteq \frac{Var(\hat{r}_{\text{SWB}})}{(E(\hat{r}_{\text{SWB}}))^2} = \frac{Var(\sum_{k=1}^K w_k \exp(\hat{\lambda}_{k,\text{SWB}}))}{r^2} \tag{A.104}$$

$$= \frac{1}{r^2} \sum_{k=1}^K w_k^2 Var(\hat{\lambda}_{k,\text{SWB}}) e^{2E(\hat{\lambda}_{k,\text{SWB}})}) = \sum_{k=1}^K w_k^2 Var(\hat{\lambda}_{k,\text{SWB}}), \tag{A.105}$$

where $Var(\hat{\lambda}_{k,\text{SWB}})$ is the variance of $\hat{\lambda}_k$, i.e.,

$$Var(\hat{\lambda}_{k,\text{SWB}}) \doteq (\frac{1}{n_{1k}} + \frac{1}{n_2})[(1 - H_A(p_{1k}, p_2))^{-1} - 1] = \frac{\tilde{w}_k + \beta}{\tilde{w}_k \beta} \frac{1}{n_1} m(p_{1k}). \tag{A.106}$$

Then

$$Var(\hat{\lambda}_{\text{WB}}) - Var(\hat{\lambda}_{\text{SWB}}) \tag{A.107}$$

$$= \frac{1+\beta}{\beta}\frac{1}{n_1}m(p_1) - \sum_{k=1}^{K} w_k^2 \frac{\tilde{w}_k+\beta}{\tilde{w}_k\beta}\frac{1}{n_1}m(p_{1k}) \tag{A.108}$$

$$= \frac{1+\beta}{\beta}\frac{1}{n_1}\left(\sum_{k=1}^{K}\tilde{w}_k m(p_{1k}) - \sum_{k=1}^{K}\tilde{w}_k\frac{m^{''}(\tau_k)(p_{1k}-p_1)^2}{2}\right) - \sum_{k=1}^{K} w_k^2 \frac{\tilde{w}_k+\beta}{\tilde{w}_k\beta}\frac{1}{n_1}m(p_{1k}) \tag{A.109}$$

$$= \frac{1}{n_1}\sum_{k=1}^{K}\left[\left(\frac{1+\beta}{\beta} - \frac{w_k^2(\tilde{w}_k+\beta)}{\tilde{w}_k^2\beta}\right)\tilde{w}_k m(p_{1k})\right] - \frac{1}{n_1}\sum_{k=1}^{K}\frac{1+\beta}{\beta}\tilde{w}_k\frac{m^{''}(\tau_k)}{2}(p_{1k}-p_1)^2. \tag{A.110}$$

Therefore, if

$$\sum_{k=1}^{K}\left[\left(\frac{1+\beta}{\beta} - \frac{w_k^2(\tilde{w}_k+\beta)}{\tilde{w}_k^2\beta}\right)\tilde{w}_k m(p_{1k})\right] \geq \sum_{k=1}^{K}\frac{1+\beta}{\beta}\tilde{w}_k\frac{m^{''}(\tau_k)}{2}(p_{1k}-p_1)^2, \tag{A.111}$$

$$Var(\hat{\lambda}_{\text{WB}}) \geq Var(\hat{\lambda}_{\text{SWB}}). \tag{A.112}$$

The condition (A.111) is very mild because the left term of the inequality is no less than 0 while the right term is near 0 for the following reasons. Following Rubenstein et al. (2019), when p_{1k} and p_2 are not much different, the plot of χ_p^2 varying with the difference is an almost horizontal line, i.e., $l^{'}(\tau_k) \approx 0$ and $l^{''}(\tau_k) \approx 0$. Then we have $m^{''}(\tau_k) \approx 0$ by equation (A.98). Meanwhile, $(p_{1k}-p_1)^2 \approx 0$, we have $\sum_{k=1}^{K}\frac{1+\beta}{\beta}\tilde{w}_k\frac{m^{''}(\tau_k)}{2}(p_{1k}-p_1)^2 \approx 0$. In addition, when p_{1k} is close to p_2, p_{1k} will also be close to p_1, then $\frac{c_{1k}}{c_1} \approx 1$ which means $w_k \approx \tilde{w}_k$. Since $\tilde{w}_k \leq 1$, we have $\sum_{k=1}^{K}[\left(\frac{1+\beta}{\beta} - \frac{w_k^2(\tilde{w}_k+\beta)}{\tilde{w}_k^2\beta}\right)\tilde{w}_k m(p_{1k})] \geq 0$. In conclusion, when p_{1k} is not too different from p_2 (usually satisfied because p_2 is set as standard Gaussian distribution and p_{1k} follows a distribution that is close to standard Gaussian since p_{1k} is the transformed distribution by Warp-U transformation, see equation (2.9)), $Var(\hat{\lambda}_{\text{WB}}) \geq Var(\hat{\lambda}_{\text{SWB}})$. □

A.4 Proof of Theorem on Relative Precision Per Second (Theorem 3)

Proof of Theorem 3. When $K = 1$, then $PpS(\hat{\lambda}_{\text{SWB}}) = PpS(\hat{\lambda}_{\text{WB}})$ because SWB is identical to WB in this case. When $K \geq 2$, the number of target evaluations performed by WB, denoted n_{WB}, is at least K times larger than that performed by SWB, denoted n_{SWB}, see Table 2.1 for details. Therefore, to prove that $PpS(\hat{\lambda}_{\text{SWB}}) \geq PpS(\hat{\lambda}_{\text{WB}})$, we need to show that the asymptotic variance satisfies

$$Var(\hat{\lambda}_{\text{SWB}}) \leq KVar(\hat{\lambda}_{\text{WB}}). \tag{A.113}$$

Similarly with (A.107),

$$KVar(\hat{\lambda}_{\text{WB}}) - Var(\hat{\lambda}_{\text{SWB}}) \tag{A.114}$$

$$= \frac{1}{n_1}\sum_{k=1}^{K}\left[\left(K\frac{1+\beta}{\beta} - \frac{w_k^2(\tilde{w}_k+\beta)}{\tilde{w}_k^2\beta}\right)\tilde{w}_k m(p_{1k})\right] - \frac{1}{n_1}\sum_{k=1}^{K} K\frac{1+\beta}{\beta}\tilde{w}_k\frac{m''(\tau_k)}{2}(p_{1k}-p_1)^2. \tag{A.115}$$

When

$$\sum_{k=1}^{K}\left[\left(K\frac{1+\beta}{\beta} - \frac{w_k^2(\tilde{w}_k+\beta)}{\tilde{w}_k^2\beta}\right)\tilde{w}_k m(p_{1k})\right] \geq \sum_{k=1}^{K} K\frac{1+\beta}{\beta}\tilde{w}_k\frac{m''(\tau_k)}{2}(p_{1k}-p_1)^2, \tag{A.116}$$

$$KVar(\hat{\lambda}_{\text{WB}}) \geq Var(\hat{\lambda}_{\text{SWB}}). \tag{A.117}$$

Therefore, for all $K \geq 1$, when (A.116) is satisfied, $PpS(\hat{\lambda}_{\text{SWB}}) \geq PpS(\hat{\lambda}_{\text{WB}})$. The condition (2.19) is milder compared with condition (2.17) of Theorem 2.

□

A.5 Physical Model of RV Prediction

Loredo et al. (2012) introduce the physical model to predict the RV. The prediction of RV is

$$v_{\text{pred}}(t) = K\left(e\cos\omega + \cos\left[\omega + \phi(t)\right]\right) + \gamma, \tag{A.118}$$

where K is the velocity semi-amplitude, γ is systemic velocity parameter which combines the projected center of mass (COM) velocity and other constant offsets in the measurement of the velocity, e is eccentricity, and ω is the argument of periapsis. $\phi(t)$ is defined as

$$\tan\frac{\phi}{2} = \sqrt{\left(\frac{1+e}{1-e}\right)}\tan\frac{E}{2}, \tag{A.119}$$

where E is an angle defined by Kepler's equation $E - e\sin E = \mathcal{M}$ and $\mathcal{M}$ is the mean anomaly $\mathcal{M} = 2\pi t/\tau + \mathcal{M}_0$, where $\mathcal{M}$ is the mean anomaly at epoch $t = 0$.

APPENDIX B

ROUGHNESS PENALTY, COMPUTATION SIMPLIFICATION OF CD-FPCA AND DETAILS OF SUPERVISED SPARSE FUNCTIONAL PCA

B.1 Roughness penalty

To encourage smoothness of the estimated mean function and covariance function, we add a roughness penalty to the negative log-likelihood (3.11). In the cases without covariates, the classical roughness penalty for an arbitrary function g is $J_t(g) = \int_a^b g''(t)^2 dt$. If g has the basis representation $g(t) = \sum_{k=1}^K \widetilde{\alpha}_k \phi_k(t) = \widetilde{\boldsymbol{\alpha}}^T \boldsymbol{\phi}(t)$, for some basis $\boldsymbol{\phi}(\cdot)$ and corresponding coefficients $\widetilde{\boldsymbol{\alpha}}$, then $J_t(g)$ is given by

$$J_t(g) = \widetilde{\boldsymbol{\alpha}}^T \int_a^b \left[\frac{\partial^2}{\partial t^2}\boldsymbol{\phi}(t)\right]\left[\frac{\partial^2}{\partial t^2}\boldsymbol{\phi}^T(t)\right] dt \widetilde{\boldsymbol{\alpha}} = \widetilde{\boldsymbol{\alpha}}^T \mathbf{S}_t \widetilde{\boldsymbol{\alpha}}, \tag{B.1}$$

where $\mathbf{S}_t = \int_a^b \left[\frac{\partial^2}{\partial t^2}\boldsymbol{\phi}(t)\right]\left[\frac{\partial^2}{\partial t^2}\boldsymbol{\phi}^T(t)\right] dt$.

This classical penalty is not directly applicable to our setting, because our mean and covariance functions depend on the unknown covariates $\boldsymbol{z}$, in addition to time t. We instead rely on the approach of Wood (2006); Reiss et al. (2014) for penalizing functions which take multiple arguments and can be expressed using tensor product bases. The expression $\mathbf{C}^T(\boldsymbol{z};\boldsymbol{\beta})\boldsymbol{b}(t)$, whose transpose appears in (3.8), is a vector of r functions evaluated at $(t, \boldsymbol{z})$, and we denote the j-th function by $h_j(t, \boldsymbol{z})$, for $j = 1, \ldots, r$. Thus, our covariance function is $G(t, s|\boldsymbol{z}) = \boldsymbol{b}(t)^T \mathbf{C}(\boldsymbol{z};\boldsymbol{\beta})\mathbf{C}^T(\boldsymbol{z};\boldsymbol{\beta})\boldsymbol{b}(s) = \sum_{j=1}^r h_j(t, \boldsymbol{z}) h_j(s, \boldsymbol{z})$, and smoothness can be imposed by penalizing the $h_j(t, \boldsymbol{z})$'s. Following Wood (2006), our penalty is

$$J(h) = \lambda_t \sum_{j=1}^r \int_{\mathcal{Z}} J_t(h^{(j)}_{t|\boldsymbol{z}}) d\boldsymbol{z} + \lambda_{\boldsymbol{z}} \sum_{j=1}^r \int_{\mathcal{T}} J_{\boldsymbol{z}}(h^{(j)}_{\boldsymbol{z}|t}) dt, \tag{B.2}$$

where $J_t(h^{(j)}_{t|\boldsymbol{z}})$ evaluates the roughness of $h^{(j)}_{t|\boldsymbol{z}} = h_j(t, \boldsymbol{z})$ viewed as a function of t, and similarly

$J_{\boldsymbol{z}}(h^{(j)}_{\boldsymbol{z}|t})$ treats $h^{(j)}_{\boldsymbol{z}|t}(\boldsymbol{z}) = h_j(t, \boldsymbol{z})$ as a function of $\boldsymbol{z}$. The penalty function $J_{\boldsymbol{z}}$ is analogous to (B.1) and evaluates the integrated squared value of the second order derivative of $h^{(j)}_{\boldsymbol{z}|t}$ with respect to $\boldsymbol{z}$. We determine the tuning parameters λ_t and $\lambda_{\boldsymbol{z}}$ by cross-validation.

In the interest of computational tractability of (B.2), we write the functions $h^{(j)}_{t|\boldsymbol{z}}(t)$ in terms of the temporal and covariate domain orthonormal B-spline bases introduced in Section 3.2.4. Denote $\boldsymbol{\beta}^{(j)}$ as the j-th column of the coefficients matrix $\boldsymbol{\Gamma}$ in (3.7). We can write $h^{(j)}_{t|\boldsymbol{z}}(t) = \widetilde{\boldsymbol{\alpha}}^{(j)}(\boldsymbol{z})^T \boldsymbol{b}(t)$, with $\widetilde{\boldsymbol{\alpha}}^{(j)}(\boldsymbol{z}) = \mathbf{M}_{\boldsymbol{z}} \boldsymbol{\beta}^{(j)}$ and $\mathbf{M}_{\boldsymbol{z}} = \mathbf{I}_w \otimes \boldsymbol{v}^T(\boldsymbol{z})$. By (B.1), the penalty $J_t(h^{(j)}_{t|\boldsymbol{z}})$ can then be expressed as $J_t(h^{(j)}_{t|\boldsymbol{z}}) = \widetilde{\boldsymbol{\alpha}}^{(j)}(\boldsymbol{z})^T \mathbf{S}_t \widetilde{\boldsymbol{\alpha}}^{(j)}(\boldsymbol{z}) = \big(\boldsymbol{\beta}^{(j)}\big)^T \mathbf{M}_{\boldsymbol{z}}^T \mathbf{S}_t \mathbf{M}_{\boldsymbol{z}} \boldsymbol{\beta}^{(j)}$, which yields

$$\int_{\boldsymbol{z}} J_t(h^{(j)}_{t|\boldsymbol{z}}) d\boldsymbol{z} = \big(\boldsymbol{\beta}^{(j)}\big)^T \Big[\int_{\mathcal{Z}} \mathbf{M}_{\boldsymbol{z}}^T \mathbf{S}_t \mathbf{M}_{\boldsymbol{z}} d\boldsymbol{z} \Big] \boldsymbol{\beta}^{(j)} \tag{B.3}$$

$$= \big(\boldsymbol{\beta}^{(j)}\big)^T \Big[\int_{\mathcal{Z}} \big(\mathbf{I}_w \otimes \boldsymbol{v}(\boldsymbol{z})\big) \big(\mathbf{S}_t \otimes 1\big) \big(\mathbf{I}_w \otimes \boldsymbol{v}^T(\boldsymbol{z})\big) d\boldsymbol{z} \Big] \boldsymbol{\beta}^{(j)} \tag{B.4}$$

$$= \big(\boldsymbol{\beta}^{(j)}\big)^T \Big[\int_{\mathcal{Z}} \big(\mathbf{S}_t \otimes \boldsymbol{v}(\boldsymbol{z}) \boldsymbol{v}^T(\boldsymbol{z})\big) d\boldsymbol{z} \Big] \boldsymbol{\beta}^{(j)} \tag{B.5}$$

Because that $\boldsymbol{v}(\boldsymbol{z})$ is orthonormal B-spline bases, i.e., $\int_{\mathcal{Z}} \boldsymbol{v}(\boldsymbol{z}) \boldsymbol{v}^T(\boldsymbol{z}) d\boldsymbol{z} = \mathbf{I}_q$, Eqn. (B.5) can be simplified as

$$\int_{\boldsymbol{z}} J_t(h^{(j)}_{t|\boldsymbol{z}}) d\boldsymbol{z} = \big(\boldsymbol{\beta}^{(j)}\big)^T \Big(\mathbf{S}_t \otimes \int_{\mathcal{Z}} \boldsymbol{v}(\boldsymbol{z}) \boldsymbol{v}^T(\boldsymbol{z}) d\boldsymbol{z} \Big) \boldsymbol{\beta}^{(j)} = \big(\boldsymbol{\beta}^{(j)}\big)^T \widetilde{\mathbf{S}}_t \boldsymbol{\beta}^{(j)}, \tag{B.6}$$

where $\widetilde{\mathbf{S}}_t = \mathbf{S}_t \otimes \mathbf{I}_q$. Note the result obtained hereby is similar but simpler than that of Reiss et al. (2014). Their penalty matrix $\widetilde{\mathbf{S}}_t$ is the Kronecker product of two dense matrices. By using orthonormal bases, one of our Kronecker product matrices become the identity matrix. When the numbers of knots q and w are large, our penalty matrix $\widetilde{\mathbf{S}}_t$ is sparse, making the computational cost smaller.

Similarly, with $\widetilde{\mathbf{S}}_{\boldsymbol{z}} = \mathbf{I}_w \otimes \mathbf{S}_{\boldsymbol{z}}$, the penalty for $h^{(j)}_{\boldsymbol{z}|t}$ is given by

$$\int_{\mathcal{T}} J_{\boldsymbol{z}}(h^{(j)}_{\boldsymbol{z}|t}) dt = \big(\boldsymbol{\beta}^{(j)}\big)^T \widetilde{\mathbf{S}}_{\boldsymbol{z}} \boldsymbol{\beta}^{(j)}, \tag{B.7}$$

Combining (B.6) and (B.7) above, the roughness penalty (B.2) has an explicit expression

$$J(h) = \sum_{j=1}^{r} \lambda_t \big(\boldsymbol{\beta}^{(j)}\big)^T \widetilde{\mathbf{S}}_t \boldsymbol{\beta}^{(j)} + \sum_{j=1}^{r} \lambda_{\boldsymbol{z}} \big(\boldsymbol{\beta}^{(j)}\big)^T \widetilde{\mathbf{S}}_{\boldsymbol{z}} \boldsymbol{\beta}^{(j)} = \boldsymbol{\beta}^T (\lambda_t \mathbf{I}_r \otimes \widetilde{\mathbf{S}}_t + \lambda_{\boldsymbol{z}} \mathbf{I}_r \otimes \widetilde{\mathbf{S}}_{\boldsymbol{z}}) \boldsymbol{\beta}, \quad \text{(B.8)}$$

where $\boldsymbol{\beta} = \mathrm{vec}(\boldsymbol{\Gamma})$.

We use an additional penalty, $J(\mu)$, to enforce smoothness of the mean function (3.4). The penalty is obtained by applying procedures analogous to those discussed above. In particular, we use the form

$$J(\mu) = \boldsymbol{\theta}_\mu^T (\lambda_t^{(\mu)} \widetilde{\mathbf{S}}_t^{(\mu)} + \lambda_{\boldsymbol{z}}^{(\mu)} \widetilde{\mathbf{S}}_{\boldsymbol{z}}^{(\mu)}) \boldsymbol{\theta}_\mu \quad \text{(B.9)}$$

where $\widetilde{\mathbf{S}}_t^{(\mu)} = \mathbf{S}_t^{(\mu)} \otimes \mathbf{I}_p$ and $\widetilde{\mathbf{S}}_{\boldsymbol{z}}^{(\mu)} = \mathbf{I}_l \otimes \mathbf{S}_{\boldsymbol{z}}^{(\mu)}$.

Combining the negative log-likelihood (3.11) with the roughness penalties (B.8) and (B.9), we obtain the objective function to be minimized:

$$\begin{aligned} \mathcal{L} + \mathcal{P} = & \sum_{n=1}^{N} \{\log\det \boldsymbol{\Sigma}_n + \mathrm{tr}(\mathbf{S}_n \boldsymbol{\Sigma}_n^{-1})\} \\ & + \boldsymbol{\theta}_\mu^T (\lambda_t^{(\mu)} \widetilde{\mathbf{S}}_t^{(\mu)} + \lambda_{\boldsymbol{z}}^{(\mu)} \widetilde{\mathbf{S}}_{\boldsymbol{z}}^{(\mu)}) \boldsymbol{\theta}_\mu + \boldsymbol{\beta}^T (\lambda_t \mathbf{I}_r \otimes \widetilde{\mathbf{S}}_t + \lambda_{\boldsymbol{z}} \mathbf{I}_r \otimes \widetilde{\mathbf{S}}_{\boldsymbol{z}}) \boldsymbol{\beta}, \end{aligned} \quad \text{(B.10)}$$

where $\mathcal{L}$ and $\mathcal{P}$ denote the log-likelihood and penalty terms, respectively. We use cross-validation to choose the four tuning parameters $\lambda_t, \lambda_{\boldsymbol{z}}, \lambda_t^{(\mu)}$, and $\lambda_{\boldsymbol{z}}^{(\mu)}$.

B.2 Proof of Lemmas

Proof of Lemma 4. With the help of the matrix determinant lemma and the Sherman-Morrison-Woodbury formula, we can obtain the determinant and inverse of $\boldsymbol{\Sigma}_n = \mathbf{B}_n \mathbf{C}_n \mathbf{C}_n^T \mathbf{B}_n^T + \sigma_e^2 \mathbf{I}$ (and reduce the computational cost at the same time), i.e.,

$$\log |\boldsymbol{\Sigma}_n| = \log\det(\mathbf{I}_r + \sigma_e^{-2} \mathbf{W}_n) + m_n \log \sigma_e^2 \quad \text{(B.11)}$$

and

$$\boldsymbol{\Sigma}_n^{-1} = \sigma_e^{-2}(\mathbf{I}_{m_n} - \mathbf{B}_n\mathbf{C}_n\{\sigma_e^2\mathbf{I}_r + \mathbf{W}_n\}^{-1}\mathbf{C}_n^T\mathbf{B}_n^T), \tag{B.12}$$

respectively, with $\mathbf{W}_n = \mathbf{C}_n^T\mathbf{B}_n^T\mathbf{B}_n\mathbf{C}_n$. Using the above results, the log likelihood in (3.11) can be simplified to

$$\mathcal{L} = \sum_{n=1}^{N} \mathrm{tr}\big(\sigma_e^{-2}\mathbf{S}_n[\mathbf{I}_{m_n} - \mathbf{B}_n\mathbf{C}_n\{\sigma_e^2\mathbf{I}_r + \mathbf{W}_n\}^{-1}\mathbf{C}_n^T\mathbf{B}_n^T]\big) \tag{B.13}$$

$$+ \sum_{n=1}^{N}\Big[\log\det(\mathbf{I}_r + \sigma_e^{-2}\mathbf{W}_n) + m_n\log\sigma_e^2\Big] \tag{B.14}$$

$$= \sum_{n=1}^{N}\Big[\log\det(\mathbf{I}_r + \sigma_e^{-2}\mathbf{W}_n) + m_n\log\sigma_e^2 + \mathrm{tr}(\sigma_e^{-2}\mathbf{S}_n)\Big] \tag{B.15}$$

$$- \sum_{n=1}^{N}\mathrm{tr}(\sigma_e^{-2}\mathbf{S}_n\mathbf{B}_n\mathbf{C}_n\{\sigma_e^2\mathbf{I}_r + \mathbf{W}_n\}^{-1}\mathbf{C}_n^T\mathbf{B}_n^T]) \tag{B.16}$$

$$= \sum_{n=1}^{N}\log\det(\mathbf{I}_r + \sigma_e^{-2}\mathbf{W}_n) + \sum_{n=1}^{N} m_n\log\sigma_e^2 + \sigma_e^{-2}\sum_{n=1}^{N}\|\boldsymbol{y}_n - \mathbf{H}_n\boldsymbol{\theta}_\mu\|_2^2 \tag{B.17}$$

$$- \sum_{n=1}^{N}(\sigma_e^{-2})^2\boldsymbol{g}_n^T\{\mathbf{I}_r + \sigma_e^{-2}\mathbf{W}_n\}^{-1}\boldsymbol{g}_n, \tag{B.18}$$

where $\boldsymbol{g}_n = \mathbf{C}_n^T\mathbf{B}_n^T(\boldsymbol{y}_n - \mathbf{H}_n\boldsymbol{\theta}_\mu)$. To further reduce computational cost, we use a Cholesky decomposition when evaluating the log-likelihood. If we compute the Cholesky factor of $\mathbf{I}_r + \sigma_e^{-2}\mathbf{W}_n$, that is $\mathbf{I}_r + \sigma_e^{-2}\mathbf{W}_n = \mathbf{F}_n\mathbf{F}_n^T$, and set $\boldsymbol{h}_n = \mathbf{F}_n^{-1}\boldsymbol{g}_n$, then the objective function can be expressed as

$$\mathcal{L} = 2\sum_{n=1}^{N}\log\det(\mathbf{F}_n) - (\sigma_e^{-2})^2\sum_{n=1}^{N}\|\boldsymbol{h}_n\|_2^2 + \sigma_e^{-2}\sum_{n=1}^{N}\|\boldsymbol{y}_n - \mathbf{H}_n\boldsymbol{\theta}_\mu\|_2^2 + \sum_{n=1}^{N} m_n\log\sigma_e^2. \tag{B.19}$$

□

Proof of Lemma 5. We use a similar approach as in the proof of Lemma 4 to reduce the computational cost of evaluating the log-likelihood gradients. As in Lemma 4, we set $\mathbf{F}_n$ as the Cholesky

factor of $\mathbf{I}_r + \sigma_e^{-2}\mathbf{W}_n$. Recall we have also set $\mathbf{W}_n = \mathbf{C}_n^T\mathbf{B}_n^T\mathbf{B}_n\mathbf{C}_n$ and $\boldsymbol{h}_n = \mathbf{F}_n^{-1}\mathbf{C}_n^T\mathbf{B}_n^T(\boldsymbol{y}_n - \mathbf{H}_n\boldsymbol{\theta}_\mu)$. Besides, denote $\mathbf{E}_n = \mathbf{F}_n^{-1}\mathbf{C}_n^T\mathbf{B}_n^T$. Then, based on (B.12), the steps to simplify the gradient (given in (3.14)) are as follows:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}_{\boldsymbol{\mu}}} = \sum_{n=1}^{N} 2\mathbf{H}_n^T\boldsymbol{\Sigma}_n^{-1}(\mathbf{H}_n\boldsymbol{\theta}_\mu - \boldsymbol{y}_n) \tag{B.20}$$

$$= \sum_{n=1}^{N} 2\sigma_e^{-2}\mathbf{H}_n^T[\mathbf{I}_{m_n} - \mathbf{B}_n\mathbf{C}_n(\sigma_e^2\mathbf{I}_r + \mathbf{W}_n)^{-1}\mathbf{C}_n^T\mathbf{B}_n^T](\mathbf{H}_n\boldsymbol{\theta}_{\boldsymbol{\mu}} - \boldsymbol{y}_n) \tag{B.21}$$

$$= \sum_{n=1}^{N} 2\sigma_e^{-2}\mathbf{H}_n^T[\mathbf{I}_{m_n} - \sigma_e^{-2}\mathbf{B}_n\mathbf{C}_n(\mathbf{F}_n\mathbf{F}_n^T)^{-1}\mathbf{C}_n^T\mathbf{B}_n^T](\mathbf{H}_n\boldsymbol{\theta}_{\boldsymbol{\mu}} - \boldsymbol{y}_n) \tag{B.22}$$

$$= \sum_{n=1}^{N} -2\sigma_e^{-2}(\mathbf{H}_n^T\boldsymbol{y}_n - \mathbf{H}_n^T\mathbf{H}_n\boldsymbol{\theta}_\mu) + 2\sigma_e^{-4}\mathbf{H}_n^T\mathbf{E}_n^T\boldsymbol{h}_n. \tag{B.23}$$

This leads to the simplified expression of the gradient. □

Proof of Lemma 6. First of all, based on (B.12), we have

$$\mathrm{tr}(\boldsymbol{\Sigma}_n^{-1}) = \sigma_e^{-2}\mathrm{tr}\left[\mathbf{I}_{m_n} - \sigma_e^{-2}\mathbf{B}_n\mathbf{C}_n(\mathbf{I}_r + \sigma_e^{-2}\mathbf{W}_n)^{-1}\mathbf{C}_n^T\mathbf{B}_n^T\right] \tag{B.24}$$

$$= \sigma_e^{-2}m_n - (\sigma_e^{-2})^2\mathrm{tr}(\mathbf{E}_n^T\mathbf{E}_n). \tag{B.25}$$

Eqn. (B.12) also implies that

$$\boldsymbol{\Sigma}_n^{-1}(\boldsymbol{y}_n - \mathbf{H}_n\boldsymbol{\theta}_\mu) = \sigma_e^{-2}(\boldsymbol{y}_n - \mathbf{H}_n\boldsymbol{\theta}_\mu) - (\sigma_e^{-2})^2\mathbf{E}_n^T\boldsymbol{h}_n, \tag{B.26}$$

where $\boldsymbol{h}_n$ and $\mathbf{E}_n$ are the same as defined in Lemma 4 and Lemma 5.

It follows that

$$\{\boldsymbol{\Sigma}_n^{-1}(\boldsymbol{y}_n - \mathbf{H}_n\boldsymbol{\theta}_\mu)\}^T\{\boldsymbol{\Sigma}_n^{-1}(\boldsymbol{y}_n - \mathbf{H}_n\boldsymbol{\theta}_\mu)\} \tag{B.27}$$

$$=\sigma_e^{-4}\|\boldsymbol{y}_n - \mathbf{H}_n\boldsymbol{\theta}_\mu\|_2^2 - 2\sigma_e^{-6}\|\boldsymbol{h}_n\|_2^2 + \sigma_e^{-8}\boldsymbol{h}_n^T\mathbf{E}_n\mathbf{E}_n^T\boldsymbol{h}_n \tag{B.28}$$

Combining (B.25) and (B.28), we can express the gradient of the log likelihood with respect to σ_e^2

(given in (3.15)) as

$$\frac{\partial \mathcal{L}}{\partial \sigma_e^2} = \sum_{n=1}^{N} \mathrm{tr}(\boldsymbol{\Sigma}_n^{-1}) - \{\boldsymbol{\Sigma}_n^{-1}(\boldsymbol{y}_n - \mathbf{H}_n\boldsymbol{\theta}_\mu)\}^T\{\boldsymbol{\Sigma}_n^{-1}\{(\boldsymbol{y}_n - \mathbf{H}_n\boldsymbol{\theta}_\mu)\} \tag{B.29}$$

$$= \sum_{n=1}^{N} \big[m_n\sigma_e^{-2} - (\sigma_e^{-2})^2\mathrm{tr}(\mathbf{E}_n^T\mathbf{E}_n) - \{\boldsymbol{\Sigma}_n^{-1}(\boldsymbol{y}_n - \mathbf{H}_n\boldsymbol{\theta}_\mu)\}^T\{\boldsymbol{\Sigma}_n^{-1}(\boldsymbol{y}_n - \mathbf{H}_n\boldsymbol{\theta}_\mu)\}\big] \tag{B.30}$$

$$= \sum_{n=1}^{N} \Big[m_n\sigma_e^{-2} - \sigma_e^{-4}\mathrm{tr}(\mathbf{E}_n^T\mathbf{E}_n) - \sigma_e^{-4}\|\boldsymbol{y}_n - \mathbf{H}_n\boldsymbol{\theta}_\mu\|_2^2 + 2\sigma_e^{-6}\|\boldsymbol{h}_n\|_2^2 \tag{B.31}$$

$$- \sigma_e^{-8}\boldsymbol{h}_n^T\mathbf{E}_n\mathbf{E}_n^T\boldsymbol{h}_n\Big]. \tag{B.32}$$

□

Proof of Lemma 7. Based on the Sherman-Morrison-Woodbury formula, we have

$$\boldsymbol{\Sigma}_n^{-1} = \sigma_e^{-2}(\mathbf{I}_{m_n} - \mathbf{B}_n\mathbf{C}_n\{\sigma_e^2\mathbf{I}_r + \mathbf{W}_n\}^{-1}\mathbf{C}_n^T\mathbf{B}_n^T) \tag{B.33}$$

$$= \sigma_e^{-2}(\mathbf{I}_{m_n} - \mathbf{K}_n\mathbf{C}_n^T\mathbf{B}_n^T), \tag{B.34}$$

where $\mathbf{W}_n = \mathbf{C}_n^T\mathbf{B}_n^T\mathbf{B}_n\mathbf{C}_n$ and $\mathbf{K}_n = \mathbf{B}_n\mathbf{C}_n\{\sigma_e^2\mathbf{I}_r + \mathbf{W}_n\}^{-1}$. Then, $\frac{\partial \mathcal{L}}{\partial \mathbf{C}_n}$ in (3.20) can be equivalently expressed as

$$\frac{\partial \mathcal{L}}{\partial \mathbf{C}_n} = 2 \times \mathbf{B}_n^T\big[\boldsymbol{\Sigma}_n^{-1} - \boldsymbol{\Sigma}_n^{-1}\mathbf{S}_n\boldsymbol{\Sigma}_n^{-1}\big]\mathbf{B}_n\mathbf{C}_n \tag{B.35}$$

$$= 2\sigma_e^{-2}(\mathbf{B}_n^T\mathbf{B}_n\mathbf{C}_n - \mathbf{B}_n^T\mathbf{K}_n\mathbf{W}_n) \tag{B.36}$$

$$- 2(\sigma_e^2)^{-2}(\mathbf{B}_n^T - \mathbf{B}_n^T\mathbf{K}_n\mathbf{C}_n^T\mathbf{B}_n^T)\mathbf{S}_n(\mathbf{B}_n\mathbf{C}_n - \mathbf{K}_n\mathbf{W}_n). \tag{B.37}$$

□

B.3 Details of Supervised Sparse Functional PCA

The main assumption of the Supervised Sparse and Functional PCA (SupSFPC) method proposed by Li et al. (2016) is that the scores $\boldsymbol{\xi}^{(n)}$ in (3.2) are linearly related to the covariates $\boldsymbol{z}_n$,

i.e.,

$$\boldsymbol{\xi}^{(n)} = \boldsymbol{\tau}_0 + \mathbf{T}^T \boldsymbol{z}_n + \boldsymbol{\gamma}^{(n)}, \tag{B.38}$$

where $\boldsymbol{\tau}_0 \in \mathbb{R}^r$ is an intercept vector, $\mathbf{T}$ is a coefficient matrix with rows corresponding to the covariates and columns corresponding to the scores. The vector $\boldsymbol{\gamma}^{(n)} \in \mathbb{R}^r$ is an independent Gaussian realization with mean zero and diagonal covariance matrix, denoted $\boldsymbol{\Sigma}_{\boldsymbol{\gamma}}$, with positive decreasing eigenvalues along the diagonal. Under this assumption Li et al. (2016) write (3.2) as

$$\begin{aligned} y_n(t) &= \mu(t) + \boldsymbol{f}^T(t)\boldsymbol{\xi}^{(n)} + \epsilon^{(n)}(t) && \text{(B.39)} \\ &= \mu(t) + \boldsymbol{f}^T(t)(\boldsymbol{\tau}_0 + \mathbf{T}^T\boldsymbol{z}_n + \boldsymbol{\gamma}^{(n)}) + \epsilon^{(n)}(t) && \text{(B.40)} \\ &= \big[\mu(t) + \boldsymbol{\tau}_0^T\boldsymbol{f}(t)\big] + (\boldsymbol{z}_n)^T\mathbf{T}\boldsymbol{f}(t) + \big[(\boldsymbol{\gamma}^{(n)})^T\boldsymbol{f}(t) + \epsilon^{(n)}(t)\big]. && \text{(B.41)} \end{aligned}$$

In Li et al. (2016), the intercept term $\mu(t) + \boldsymbol{\tau}_0^T\boldsymbol{f}(t)$ is omitted since they assume that $y_n(t)$ and $\boldsymbol{z}_n$ are centered. Hence, the final SupSFPC model is

$$y_n(t) = \boldsymbol{z}_n^T\mathbf{T}\boldsymbol{f}(t) + \big[(\boldsymbol{\gamma}^{(n)})^T\boldsymbol{f}(t) + \epsilon^{(n)}(t)\big]. \tag{B.42}$$

The corresponding discretized version is

$$\mathbf{Y} = \mathbf{ZTF}^T + \mathbf{QF}^{\mathbf{T}} + \mathbf{E}, \tag{B.43}$$

where $\mathbf{Y}$ is an $N \times m$ matrix with $\mathbf{Y}_{ij} = y_i(t_j)$, $\mathbf{Z}$ is the matrix $(\boldsymbol{z}_1, \ldots \boldsymbol{z}_N)^T$, $\mathbf{F}$ is an $m \times r$ matrix with $\mathbf{F}_{jk} = f_k(t_j)$, $\mathbf{Q}$ is the matrix $(\boldsymbol{\gamma}^{(1)}, \ldots, \boldsymbol{\gamma}^{(N)})^T$, and $\mathbf{E}$ is an $N \times m$ error matrix $\mathbf{E}$ with $\mathbf{E}_{ij} = \epsilon^{(i)}(t_j)$.

Li et al. (2016) use penalized maximum likelihood to obtain estimates $\hat{\mathbf{T}}$, $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}}$, and $\hat{\mathbf{F}}$ of the coefficient matrix, covariance matrix, and eigenfunction matrix, respectively. Predictions for new curves can then be made using an approach analogous to that in Section 3.3.2, as we now explain. Suppose we observe noisy observations of a new curve $\boldsymbol{y}_* = (y_*(t_1^{(*)}), \ldots, y_*(t_{m_*}^{(*)}))^T$ together

with the corresponding covariate vector $\boldsymbol{z}_*$. We require that the time points $t_1^{(*)}, \ldots, t_{m_*}^{(*)}$ belong the collection of time points in the training data, i.e., $\{t_1, \ldots, t_m\}$, because SupSFPC is a discretized FPCA method. Let $\hat{\mathbf{F}}_*$ be the sub-matrix of $\hat{\mathbf{F}}$ whose rows correspond to $t_1^{(*)}, \ldots, t_{m_*}^{(*)}$. Then, using a plug-in approach, the joint distribution of $\boldsymbol{y}_*$ and $\boldsymbol{\xi}^{(*)}$ is approximated by

$$\begin{pmatrix} \boldsymbol{y}_* \\ \boldsymbol{\xi}^{(*)} \end{pmatrix} \sim \mathcal{N}_{2m^*} \left(\begin{pmatrix} \hat{\mathbf{F}}_* \hat{\mathbf{T}}^T \boldsymbol{z}_* \\ \hat{\mathbf{T}}^T \boldsymbol{z}_* \end{pmatrix}, \begin{pmatrix} \hat{\mathbf{F}}_* \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}} \hat{\mathbf{F}}_*^T + \hat{\sigma}_e^2 \mathbf{I} & \hat{\mathbf{F}}_* \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}} \\ \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}} \hat{\mathbf{F}}_*^T & \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}} \end{pmatrix} \right). \tag{B.44}$$

The "posterior distribution" of $\boldsymbol{\xi}^{(*)}$ is then a multivariate Gaussian whose mean and covariance matrix are given by

$$\hat{\mathbf{T}}^T \boldsymbol{z}_* + \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}} \hat{\mathbf{F}}_*^T (\hat{\mathbf{F}}_* \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}} \hat{\mathbf{F}}_*^T + \hat{\sigma}_e^2 \mathbf{I})^{-1} (\boldsymbol{y}_* - \hat{\mathbf{F}}_* \hat{\mathbf{T}}^T \boldsymbol{z}_*), \tag{B.45}$$

and

$$\hat{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}} - \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}} \hat{\mathbf{F}}_*^T (\hat{\mathbf{F}}_* \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}} \hat{\mathbf{F}}_*^T + \hat{\sigma}_e^2 \mathbf{I})^{-1} \hat{\mathbf{F}}_* \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}}, \tag{B.46}$$

respectively.

Finally, combining (B.45) and (B.46) with (B.39), the posterior predictive distribution of $y_*(t_j)$ at a new time t_j is a univariate Gaussian distribution with mean and variance given by

$$\hat{\boldsymbol{f}}(t_j)^T \mathbb{E}(\boldsymbol{\xi}^{(*)} | \boldsymbol{y}_*) \tag{B.47}$$

and

$$\hat{\boldsymbol{f}}(t_j)^T \mathrm{Cov}(\boldsymbol{\xi}^{(*)} | \boldsymbol{y}_*) \hat{\boldsymbol{f}}(t_j) + \hat{\sigma}_e^2, \tag{B.48}$$

respectively.

In Section 3.4.3, we also generate from an extended version of SupFPCS where the mean of the scores $\boldsymbol{\xi}^{(n)}$ is a quadratic function of the covariates $\boldsymbol{z}_n$, i.e.,

$$\boldsymbol{\xi}^{(n)} = \boldsymbol{\tau}_0 + \mathbf{T}^T \boldsymbol{z}_n + \mathbf{U}^T \boldsymbol{z}_n^2 + \boldsymbol{\gamma}^{(n)}, \tag{B.49}$$

where $\boldsymbol{z}_n^2$ is a vector whose elements are the squares of those of $\boldsymbol{z}_n$, and $\mathbf{U}$ is a coefficients matrix. Thus, the model (B.42) is modified to

$$y_n(t) = \boldsymbol{z}_n^T \mathbf{T} \boldsymbol{f}(t) + (\boldsymbol{z}_n^2)^T \mathbf{U} \boldsymbol{f}(t) + \left[(\boldsymbol{\gamma}^{(n)})^T \boldsymbol{f}(t) + \epsilon^{(n)}(t) \right]. \tag{B.50}$$

B.4 The Connection between SupSFPC and CD-FPCA

Here we show that SupSFPC is a special case of our CD-FPCA method. Under the SupSFPC model (B.42), the observed noisy function at time t, i.e., $y(t)$, follows the Gaussian distribution $\mathcal{N}\left(\boldsymbol{z}^T \mathbf{T} f(t), \boldsymbol{f}(t)^T \boldsymbol{\Sigma}_{\boldsymbol{\gamma}} \boldsymbol{f}(t) + \sigma_e^2\right)$. On the other hand, under the CD-FPCA model (3.8) we have $y(t) \sim \mathcal{N}\left(\mathbf{H}(t, \boldsymbol{z})^T \boldsymbol{\theta}_\mu, \boldsymbol{b}(t)^T \boldsymbol{\Sigma}(\boldsymbol{z}; \boldsymbol{\beta}) \boldsymbol{b}(t) + \sigma_e^2\right)$. Therefore, SupSFPC falls under the CD-FPCA framework, because the equations

$$\boldsymbol{f}(t)^T \mathbf{T}^T \boldsymbol{z} = \mathbf{H}(t, \boldsymbol{z})^T \boldsymbol{\theta}_\mu = \boldsymbol{a}(t)^T \boldsymbol{\Theta}_\mu \boldsymbol{u}(\boldsymbol{z}) \tag{B.51}$$

and

$$\boldsymbol{f}(t)^T \boldsymbol{\Sigma}_{\boldsymbol{\gamma}} \boldsymbol{f}(t) = \boldsymbol{b}(t)^T \mathbf{C}(\boldsymbol{z}; \boldsymbol{\beta}) \mathbf{C}(\boldsymbol{z}; \boldsymbol{\beta})^T \boldsymbol{b}(t), \tag{B.52}$$

can always be satisfied for suitable bases $\boldsymbol{a}(t), \boldsymbol{b}(t), \boldsymbol{u}(\boldsymbol{z})$, and $\boldsymbol{v}(\boldsymbol{z})$. In particular, regarding (B.51), note that we can choose $\boldsymbol{u}(\boldsymbol{z}) = \boldsymbol{z}$, and that $\boldsymbol{f}(t)^T \mathbf{T}^T$ is just another function of t, say $\boldsymbol{g}(t)^T$. Thus, since $\boldsymbol{\Theta}_\mu$ is a matrix of unconstrained coefficients, we can choose $\boldsymbol{\Theta}_\mu$ such that $\boldsymbol{f}(t)^T \mathbf{T}^T = \boldsymbol{g}(t)^T = \boldsymbol{a}(t)^T \boldsymbol{\Theta}_\mu$, provided that $\boldsymbol{a}(t)$ is a suitable basis, and thus (B.51) can always be satisfied. Similarly, regarding (B.52), let $\boldsymbol{\Theta}$ be the coefficients matrix such that $\boldsymbol{f}(t) = \boldsymbol{\Theta}^T \boldsymbol{b}(t)$, for a suitable basis $\boldsymbol{b}(t)$. Since $\mathbf{C}$ is unconstrained, it can be chosen such that $\mathbf{C} = \boldsymbol{\Theta} \Sigma_\gamma^{1/2}$, and hence (B.52) can also always be satisfied mathematically. Of course, CD-FPCA is more flexible than SupSFPC and does not typically reduce to the latter in practice. In particular, if we replace the unknowns in (B.51) and (B.52) by their estimates then the equalities will typically no longer hold. Whether they approximately hold obviously depends on the specific data and the penalizations implemented.