UNIVERSITY of CALIFORNIA

SANTA BARBARA

Scalable Latent Factor Models and Forecasting for Dynamical Systems

A dissertation submitted in partial satisfaction

of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

STATISTICS AND APPLIED PROBABILITY

by

Yizi Lin

Committee in charge:

Professor Mengyang Gu, Chair

Professor Alexander Franks

Professor Yuedong Wang

June 2026

The Dissertation of Yizi Lin is approved.

Professor Alexander Franks

Professor Yuedong Wang

Professor Mengyang Gu, Committee Chair

June 2026

Scalable Latent Factor Models and Forecasting for Dynamical Systems

by

Yizi Lin

To my family.

Acknowledgements

Looking back on this journey, I feel incredibly fortunate to have been accompanied by so many brilliant and inspiring people, each of whom has shaped both this work and who I am.

First and foremost, I am deeply grateful to my advisor, Professor Mengyang Gu, whose mentorship has been invaluable throughout my doctoral journey. Working under his guidance has fundamentally shaped the way I approach research, teaching me not only the technical skills needed to tackle complex problems, but more importantly, the importance of always seeking to understand the "why" behind every result. His insight, enthusiasm, and patience for research have been a constant source of inspiration.

I would also like to thank my dissertation committee members, Professor Yuedong Wang and Professor Alexander Franks, for their thoughtful feedback, constructive suggestions, and continued support throughout this process.

I am grateful to my collaborators, Professor Diana Y. Qiu and Professor Paul Segall, whose expertise and dedication broadened the scope and depth of this work. I have learned a great deal from each of them.

The research presented in this dissertation was supported in part by the National Science Foundation under grants DMS-2053423, OAC-2411043, ITE-2236021, and CNS-2220417. This work also received support from the University of California Multicampus Research Programs and Initiatives grant M23PL5990, and the University of California, Santa Barbara Faculty Acceleration Grant.

I would also like to thank my peers, Xinyi Fang, Xubo Liu, and Yaxuan Wang, for the many insightful discussions we shared and for their companionship throughout this journey. Their support and friendship made graduate school a much brighter and more enjoyable experience.

Finally, and most profoundly, I thank my parents and maternal grandparents for their unconditional love and unwavering support. None of this would have been possible without the foundation they provided.

Curriculum Vitæ

Yizi Lin

Education

2026	Ph.D. in Statistics, University of California, Santa Barbara
2020	M.S. in Statistical Science, Duke University
2018	B.S. in Economics Statistics, Jinan University

Experience

06/2024 - 09/2024	Graduate Data Scientist Intern, Amgen.
09/2021 - 06/2026	Research and Teaching Assistant, Department of Statistics and Applied Probability, University of California, Santa Barbara.

Publications

1. Lin, Y., Liu, X., Segall, P., & Gu, M. (2026). Fast data inversion for high-dimensional dynamical systems from noisy measurements. *Accepted by SIAM/ASA Journal of Uncertainty Quantification, arXiv preprint arXiv:2501.01324.*

2. Baracaldo, L., King, B., Yan, H., Lin, Y., Miolane, N., & Gu, M. (2026). Unsupervised Cell Segmentation by Fast Gaussian Processes. *The New England Journal of Statistics in Data Science, 1-16.* doi:10.51387/26-NEJSDS97.

3. Gu, M., Lin, Y., Lee, V. C., & Qiu, D. Y. (2024). Probabilistic forecast of nonlinear dynamical systems with uncertainty quantification. *Physica D: Nonlinear Phenomena, 457*, 133938.

4. Xu, Y., Lin, Y., Bell, R. P., Towe, S. L., Pearson, J. M., Nadeem, T., ... & Meade, C. S. (2021). Machine learning prediction of neurocognitive impairment among people with HIV using clinical and multimodal magnetic resonance imaging data. *Journal of neurovirology, 27(1)*, 1-11.

Software

1. Gu M, Fang X, Lin Y (2025). FastGaSP: Fast and Exact Computation of Gaussian Stochastic Process. R package version 0.6.2.
https://CRAN.R-project.org/package=FastGaSP

Presentations

03/2025	Contributed poster, Kernel Methods in Uncertainty Quantification and Experimental Design Workshop in Statistical and Mathematical Innovation (IMSI) at the University of Chicago, Chicago, IL.
08/2024	Contributed poster, Joint Statistical Meetings, Portland, OR.
03/2024	Oral presentation, American Physical Society's Meeting, Minneapolis, MN.
02/2026, 2025, 2024	Oral presentation, Graduate Student Research Lightning talks at UC Santa Barbara, Santa Barbara, CA.

Abstract

Scalable Latent Factor Models and Forecasting for Dynamical Systems

by

Yizi Lin

High-dimensional dynamical systems arise in many scientific and engineering applications, yet scalable probabilistic methods for estimating latent processes and forecasting future states with reliable uncertainty quantification remain limited. Latent factor models with Gaussian process (GP) priors on the latent factors offer a flexible probabilistic framework that naturally captures spatio-temporal dependence, quantifies uncertainty, and performs well in data-limited settings. However, existing approaches face two fundamental challenges. First, high-dimensional dynamical systems require estimating a large number of parameters, while GP priors on latent factors incur cubic computational complexity, making scalable inference prohibitive. Second, forecasting nonlinear dynamics with uncertainty quantification remains challenging when the governing equations are unknown.

This dissertation addresses both obstacles through three contributions. First, in Chapter 2, we introduce a latent factor model with orthogonal factor loadings, where each latent factor independently follows an Ornstein-Uhlenbeck process with distinct correlation and variance parameters to capture heterogeneous underlying dynamics. To enable scalable inference, we develop a fast Expectation-Maximization (EM) algorithm that leverages the orthogonal loading structure together with the Kalman filter and Rauch-Tung-Striebel smoother to derive closed-form parameter updates. The resulting algorithm scales linearly with both the output dimension and the number of observations. Extensive simulated studies and a real-world application to slip estimation in the

Cascadia region demonstrate the accuracy and scalability of the proposed approach.

Second, in Chapter 3, we present three new modules in the `FastGaSP` R package for scalable Gaussian process and latent factor model inference. The `fgasp` module enables fast and exact computation for Gaussian processes with Matérn kernel. The `gppca` module performs parameter estimation for latent factor models with orthogonal factor loadings and GP priors on the latent factors. The `fmou` module implements the scalable algorithm developed in Chapter 2. Together, these modules provide practical and computationally efficient tools for large-scale spatio-temporal modeling and uncertainty quantification.

Finally, in Chapter 4, we develop a probabilistic forecasting framework for nonlinear dynamical systems by extending the parallel-partial Gaussian process method to estimate the unknown transition function and generate uncertainty-aware forecasts through posterior sampling. We further establish the equivalence between dynamic mode decomposition (DMD) and the maximum likelihood estimator of the linear mapping matrix in a linear state space model, providing a probabilistic interpretation of DMD that enables uncertainty quantification of prediction. Numerical examples demonstrate the role of uncertainty quantification in assessing forecast reliability and highlight the importance of correctly specifying model inputs.

Together, these contributions provide probabilistic tools for scalable parameter estimation and forecasting with uncertainty quantification in high-dimensional dynamical systems.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

High-dimensional dynamical data arise naturally in many scientific and engineering fields. In geophysics, networks of Global Positioning System (GPS) stations record ground deformation time series across hundreds of spatial locations to monitor seismic activity and slow slip events [2]. In condensed matter physics, simulating quantum many-body systems far from equilibrium generates high-dimensional density matrices whose time evolution requires millions of Central Processing Unit (CPU) hours even for systems of only a few atoms [3, 4, 5]. In climate science, atmospheric quantities are tracked simultaneously at thousands of spatial grid points [6]. A common property across these applications is that the observed data are both high-dimensional and correlated, and that the scientific goal is not merely to fit the data but to recover latent physical processes, forecast future states, and quantify uncertainty.

Modern data-driven methods, such as recurrent neural networks [7, 8] and transformer [9], have demonstrated remarkable success in modeling complex, high-dimensional sequence data. However, their applications in science face several fundamental challenges. First, these methods typically require large amounts of training data to accurately estimate the model parameters, whereas in many scientific problems, the sample size is

limited by practical constraints such as the cost of repeated experiments or the time required to collect each observation. Second, training these models requires substantial computational resources, such as large-scale GPU clusters and extensive computation time, which may be prohibitive in scientific settings where budgets are constrained. Third, most of these methods provide only point estimates of the quantities of interest without any measurement of uncertainty. Yet uncertainty quantification is crucial for assessing the reliability and confidence of predictions, particularly for tasks that inform decision-making, experimental design, and scientific inference.

This dissertation addresses the problem of modeling high-dimensional dynamical systems within a probabilistic framework that explicitly quantifies uncertainty. We focus on two classes of statistical models: Gaussian processes (GPs) and latent factor models. Unlike neural network-based approaches, GPs provide a Bayesian framework for modeling correlated data that naturally offer uncertainty quantification, requiring relatively few observations for accurate inference. GPs have been widely applied to spatial statistics [10], emulation of expensive computer simulations [11, 12, 13], calibration of physical models [14, 15, 16], and particle trajectories [17, 18]. Latent factor models offer a framework for reducing the dimensionality of k-dimensional observations by mapping them into a d-dimensional latent space ($k \geq d$). In the setting considered in this dissertation, each latent factor is independently modeled by a GP, enabling the capture the dependency between observations and inheriting the same advantages of GPs. The latent factor models have broad applications in climate science [19], microscopy video [20], and multivariate time series analysis [21].

Despite their strengths, both approaches face computational bottlenecks that limit their applicability to large-scale, high-dimensional problems. For scalar-valued GPs, evaluating the marginal log-likelihood and computing the predictive distribution require $\mathcal{O}(n^3)$ computational order in the general case. Extending it to multivariate output in-

troduces additional challenges. For example, the many single-output emulator approach [22, 23, 24], which fits an independent GP for each output coordinate, has a computational cost of $\mathcal{O}(kn^3)$. Moreover, the number of kernel parameters grows with the output dimension, making marginal likelihood optimization high-dimensional and numerically unstable. For latent factor models, the $k \times d$ factor loading matrix introduces a large number of parameters when both k and d are large, creating challenges for both estimation and storage. When the latent factor model has a state-space representation, the conventional Expectation-Maximization (EM) algorithm for marginal likelihood evaluation requires inversion of a $k \times k$ covariance matrix at each time step, resulting in $\mathcal{O}(nk^3)$ complexity, which becomes prohibitive when both the number of time steps n and the output dimension k are large. In addition, estimating the factor loading matrix under the orthogonality constraint requires optimization on the Stiefel manifold, which is computationally non-trivial. Finally, forecasting a nonlinear dynamical system with uncertainty quantification, where the true transition function is unknown, remains an open challenge. Developing scalable and numerically stable methods that overcome these bottlenecks is therefore the central computational motivation of this dissertation.

In this dissertation, we make three main contributions that address these limitations. In Chapter 2, we develop a scalable algorithm, called the Fast latent factor model with Multivariate Ornstein-Uhlenbeck (OU) processes (FMOU), for parameter estimation in high-dimensional latent factor models with distinct OU processes for each latent factor. By introducing an orthogonal factor loading matrix and leveraging Kalman filter (KF) [25] and Rauch-Tung-Striebel (RTS) [26] smoother, we derive closed-form expressions for all parameters in each EM iteration, reducing the per-iteration computational cost to linear complexity in both the number of inputs (n) and the dimension of the output dimension (k). In Chapter 3, we present three modules in the `FastGaSP` R package: `fgasp`, `gppca` and `fmou`. The `fgasp` module provides fast and exact computation of

profile log-likelihood and predictive distribution for GPs with one-dimensional inputs, scalar outputs, and Matérn kernel with roughness parameters $\nu \in \{\frac{1}{2}, \frac{3}{2}, \frac{5}{2}\}$, at a cost of $\mathcal{O}(n)$ by the Kalman filter and RTS smoother. This module serves as the computational foundation for the other two modules. The `gppca` module efficiently estimates parameters in generalized probabilistic principal component analysis (GPPCA) [19], a latent factor model with an orthogonal factor loading matrix, where each latent factor is modeled as an independent GP. The `fmou` module implements the algorithm developed in Chapter 2. As a special case of GPPCA, it allows a faster estimation procedure with closed-form parameter updates. In Chapter 4, we address uncertainty quantification for forecasting nonlinear dynamical systems when the true data-generating process is unknown. We extend the parallel-partial Gaussian process (PP-GP) [27] method to predict the one-step-ahead transition function of high-dimensional dynamical systems, enabling probabilistic forecasting via posterior sampling. A key advantage of this approach is that the uncertainty quantification from PP-GP can serve as an alert when the forecast reliability degrades, enabling timely intervention. Together, these contributions advance the practical applications of GPs and latent factor models for high-dimensional dynamical systems, with improved computational efficiency and uncertainty quantification.

The remainder of this chapter introduces the methodological background of this dissertation. Section 1.1 reviews Gaussian process models for both scalar- and vector-valued outputs, including correlation structures, parameter estimation, and predictive distribution. Section 1.2 introduces latent factor models and establishes their connection to GP-based covariance structures. Section 1.3 describes the dynamic linear model (DLM), which provides a state-space representation connecting to GPs and latent factor models. The Kalman filter and RTS smoother are introduced as computationally efficient algorithms for sequential state estimation within this framework. Section 1.4 presents the Expectation-Maximization algorithm as a general framework for parameter estimation

in models with unobservable variables, such as latent factor models and DLM. Section 1.5 reviews dynamic mode decomposition [28, 29], a data-driven method which is popular in applied mathematics for efficiently extracting the low-rank representation of high-dimensional dynamical systems. Together, these topics form the methodological foundation for the contributions developed in this dissertation.

1.1 Gaussian process with scalar- and vector-values

The Gaussian stochastic process (GaSP) or Gaussian process (GP) model is widely used for modeling correlated data. Let $y(\mathbf{x}) \in \mathbb{R}$ denote a noisy measurement associated with an input vector $\mathbf{x} \in \mathbb{R}^{p_x}$. For example, in climate science, $\mathbf{x}$ can be the spatial coordinate and $y(\mathbf{x})$ can be the climate variables, such as the temperature or precipitation which is crucial for vegetation condition and crop yield [30, 31]. Under the GP model, the noisy measurement is represented as $y(\mathbf{x}) = f(\mathbf{x}) + \epsilon(\mathbf{x})$, where $f(\cdot)$ is the latent Gaussian process and $\epsilon(\mathbf{x}) \overset{\text{i.i.d}}{\sim} \mathcal{N}(0, \sigma_0^2)$ is the measurement error. Since f and ϵ are independent, after integrating out the latent Gaussian process $f(\cdot)$, $y(\cdot)$ follows a GP:

$$y(\cdot) \sim \text{GP}\left(\mu(\cdot), \sigma^2 c(\cdot,\cdot) + \sigma_0^2 \delta(\cdot,\cdot)\right), \tag{1.1}$$

where $\mu(\cdot)$ is the mean function, σ^2 is the signal variance, $c(\cdot,\cdot)$ is the correlation function, and $\delta(\cdot,\cdot)$ is the Kronecker delta function, which equals 1 if $\mathbf{x} = \mathbf{x}'$ and 0 otherwise. Given a set of n input $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, the output vector follows a multivariate normal distribution:

$$((y(\mathbf{x}_1), y(\mathbf{x}_2), \ldots, y(\mathbf{x}_n))^\top \mid \boldsymbol{\mu}, \sigma^2, \mathbf{R}, \sigma_0^2 \sim \mathcal{MN}(\boldsymbol{\mu}, \sigma^2 \mathbf{R} + \sigma_0^2 \mathbf{I}_n), \tag{1.2}$$

where $\boldsymbol{\mu} = (\mu(\mathbf{x}_1), \mu(\mathbf{x}_2), \ldots, \mu(\mathbf{x}_n))^\top$ is the mean vector based on the mean function $\mu(\cdot)$, $\mathbf{R}$ is the correlation matrix where the (i, i') entry is computed as $R_{i,i'} = c(\mathbf{x}_i, \mathbf{x}_{i'})$ for $i, i' = 1, 2, \ldots, n$.

Typically, a linear additive structure is applied to the mean function, such that $\mu(\mathbf{x}) = \mathbf{h}(\mathbf{x})\mathbf{b}$, where $\mathbf{h}(\mathbf{x}) = (h_1(\mathbf{x}), \ldots, h_{p_\mu}(\mathbf{x})) \in \mathbb{R}^{p_\mu}$ is a row vector of basis functions and $\mathbf{b} = (b_1, \ldots, b_{p_\mu})$ is a vector of trend parameters. In practice, a constant mean (e.g., $\mathbf{h}(\mathbf{x}) = 1$) is often used as a default.

Correlation functions are typically constructed using either an isotropic or a product structure. An isotropic correlation considers only the Euclidean distance between two inputs, such that $c(\mathbf{x}_i, \mathbf{x}_{i'}) = K(\|\mathbf{x}_i - \mathbf{x}_{i'}\|)$, where $K(\cdot)$ denotes a one-dimensional kernel function and $\|\cdot\|$ represents the Euclidean distance. The underlying assumption for this approach is that all input coordinates share the same scale and smoothness, so that a one-unit change in any coordinate has an identical effect on the correlation. Therefore, it is suitable when all input coordinates operate on the same scale. However, this assumption may be unrealistic when the input coordinates represent different physical quantities or operate on different scales, such as $\mathbf{x}$ = (temperature, time). In such cases, a product kernel is more appropriate, as it allows each coordinate to have its own correlation structure: $c(\mathbf{x}_i, \mathbf{x}_{i'}) = \prod_{p=1}^{p_x} K_p(\|x_{i,p} - x_{i',p}\|)$, where each K_p has its own roughness and range parameter to account for the scale and smoothness in the p-th coordinate. By utilizing the product kernel, the correlation matrix can be decomposed as $\mathbf{R} = \mathbf{R}_1 \circ \cdots \circ \mathbf{R}_p \circ \cdots \circ \mathbf{R}_{p_x}$, where $\mathbf{R}_p$ is the correlation matrix for the p-th coordinate constructed via $K_p(\cdot)$ for $p = 1, \ldots, p_x$, and $\circ$ is the element-wise product.

Table 1.1 summarizes some popular choices for the kernel function, where $r_p = |x_{i,p} - x_{i',p}|$ denotes the distance between two inputs along the p-th coordinate. Each kernel is characterized by two parameters: a roughness parameter ν and a range parameter γ_p. The roughness parameter controls the differentiability of the process, where increasing

its value generally yields a smoother sample path. The range parameter γ_p controls the correlation decay between two inputs. A larger value γ_p implies a slower decay and therefore a stronger correlation between inputs that are farther apart. In practice, the roughness parameter is often fixed based on the prior assumption about the system, while the range parameter is estimated from data.

Within the power exponential family, when $\nu = 1$, the kernel becomes the exponential kernel, introducing a continuous but nondifferentiable process. At the other extreme, $\nu = 2$ gives the Gaussian kernel, also known as the radial basis function (RBF) kernel, which produces an infinitely mean-square differentiable and thus extremely smooth process. However, the resulting correlation matrix is prone to singularity, making it difficult to invert as required for likelihood evaluation and predictive inference. To overcome this limitation, a roughness parameter close to 2 (e.g. $\nu = 1.9$) is often used instead. More broadly, any choice of $\nu < 2$ within the power exponential family introduces a process that is not mean-square differentiable, which may be improper when the underlying system is known to exhibit a finite degree of smoothness.

The Matérn family addresses this limitation by offering explicit control over the differentiability of the process. Specifically, a Matérn kernel with roughness parameter ν yields a process that is $\lceil \nu - 1 \rceil$-times mean-square differentiable. In particular, when $\nu = \frac{1}{2}$, the Matérn kernel reduces to the exponential kernel, while it converges to the Gaussian kernel as $\nu \to \infty$. A common choice in practice is $\nu = \frac{5}{2}$, which serves as the default setting in several software packages for Gaussian process surrogate modeling [32, 33]. This parameterization produces a twice mean-square differentiable process and demonstrates good empirical performance.

There are four sets of parameters that need to be estimated in a GP model: the signal variance σ^2, the noise variance σ_0^2, the trend parameters $\mathbf{b}$ in the mean function, and the range parameters $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_{\tilde{p}_x})$ in the kernel, where $\tilde{p}_x = 1$ is for isotropic

Kernel function	Definition
Power Exponential	$\exp\left(-(\frac{r_p}{\gamma_p})^{\nu}\right), \quad \nu \in (0, 2]$
Rational Quadratic	$\left(1+\frac{r_p}{\gamma_p}^2\right)^{-\nu}, \quad \nu \in (0, +\infty)$
Matérn	$\frac{1}{2^{\nu-1}\Gamma(\nu)}(\frac{r_p}{\gamma_p})^{\nu}\mathcal{K}_{\nu}(\frac{r_p}{\gamma_p}), \quad \nu \in (0, +\infty)$

Table 1.1: Common choice of kernel function, where $r_p = |x_{i,p} - x_{i',p}|$. γ_p and ν are the range and roughness parameters, respectively. $\Gamma(\cdot)$ is the gamma function and $\mathcal{K}_{\nu}$ is the modified Bessel function of second kind of order ν.

kernel and $\tilde{p}_x = p_x$ is for product kernel . The likelihood function of the output vector $\mathbf{y} = (y(\mathbf{x}_1), \ldots, y(\mathbf{x}_n))^{\top} \in \mathbb{R}^n$ in the GP model follows a multivariate normal density:

$$p(\mathbf{y} \mid \sigma^2, \eta, \mathbf{b}, \boldsymbol{\gamma}) = (2\pi\sigma^2)^{-\frac{n}{2}} |\tilde{\mathbf{R}}|^{-\frac{1}{2}} \exp\left(-\frac{(\mathbf{y}-\mathbf{H}\mathbf{b})^{\top}\tilde{\mathbf{R}}^{-1}(\mathbf{y}-\mathbf{H}\mathbf{b})}{2\sigma^2}\right), \tag{1.3}$$

where $\mathbf{H} = (\mathbf{h}^{\top}(\mathbf{x}_1), \ldots, \mathbf{h}^{\top}(\mathbf{x}_n))$ is a $n \times p_{\mu}$ basis function matrix, and $\tilde{\mathbf{R}} = \mathbf{R} + \eta\mathbf{I}_n$ with $\eta = \frac{\sigma_0^2}{\sigma^2}$ being the nugget variance ratio.

In practice, evaluating the likelihood requires computing $|\tilde{\mathbf{R}}|$ and $\tilde{\mathbf{R}}^{-1}$, which is typically done via the Cholesky decomposition $\tilde{\mathbf{R}} = \mathbf{L}\mathbf{L}^{\top}$, where $\mathbf{L}$ is a lower triangular matrix with the ith diagonal element being L_{ii}. We then have $\log(|\tilde{\mathbf{R}}|) = 2\sum_{i=1}^{n}\log(L_{ii})$ and the quadratic form can be computed via forward substitution. Compared to directly computing the matrix inverse and determinant, although the computational orders are both $\mathcal{O}(n^3)$, the Cholesky decomposition is numerically more stable. However, this decomposition must be recomputed at every evaluation of the likelihood during parameter optimization, making it the dominant computational bottleneck in GP inference.

Although the maximum likelihood estimator (MLE) of the parameters can be obtained by maximizing the logarithm of the likelihood in Equation (1.3), it is well known to be numerically unstable [34, 35, 36, 37]. To improve robustness, we instead utilize formal objective priors, namely reference priors, which yield more stable parameter esti-

mates. Specifically, we assume the following reference prior for $(\mathbf{b}, \sigma^2)$ [38]:

$$\pi^R(\mathbf{b}, \sigma^2) \propto \frac{1}{\sigma^2}. \tag{1.4}$$

By integrating out $(\mathbf{b}, \sigma^2)$, the marginal likelihood of $\mathbf{y}$ given $\boldsymbol{\gamma}$ and η has the expression below [38]:

$$p(\mathbf{y} \mid \boldsymbol{\gamma}, \eta) \propto \int p(\mathbf{y} \mid \mathbf{b}, \sigma^2, \boldsymbol{\gamma}, \eta)\pi^R(\mathbf{b}, \sigma^2)d\mathbf{b}d\sigma^2 \tag{1.5}$$

$$\propto |\tilde{\mathbf{R}}|^{-\frac{1}{2}}|\mathbf{H}^\top\tilde{\mathbf{R}}^{-1}\mathbf{H}|^{-\frac{1}{2}}(S^2)^{-\frac{n-p_\mu}{2}}, \tag{1.6}$$

where $S^2 = (\mathbf{y} - \mathbf{H}\hat{\mathbf{b}})^\top\tilde{\mathbf{R}}^{-1}(\mathbf{y} - \mathbf{H}\hat{\mathbf{b}})$ with $\hat{\mathbf{b}} = (\mathbf{H}^\top\tilde{\mathbf{R}}^{-1}\mathbf{H})^{-1}\mathbf{H}^\top\tilde{\mathbf{R}}^{-1}\mathbf{y}$ being the generalized least squares estimator of $\mathbf{b}$ given $\boldsymbol{\gamma}$ and η.

Since the marginal posterior distribution of $(\boldsymbol{\gamma}, \eta)$ is usually intractable, these parameters are estimated using their posterior mode via numerical optimization, such that

$$(\hat{\boldsymbol{\gamma}}, \hat{\eta}) = \arg\max_{\boldsymbol{\gamma},\eta}(p(\mathbf{y} \mid \boldsymbol{\gamma}, \eta)\pi(\boldsymbol{\gamma}, \eta)), \tag{1.7}$$

where $\pi(\boldsymbol{\gamma}, \eta)$ is the reference prior distribution for the range and nugget parameters [39] with robust parameterization. Due to the computational cost of the gradient of the reference prior, the jointly robust parameter of the inverse range parameter is often used as an approximation in practice [40]. Quasi-Newton optimization methods, such as the Broyden–Fletcher–Goldfarb–Shanno (BFGS) [41] algorithm and Limited-memory BFGS [42], are commonly used.

After obtaining the estimated range and nugget parameters, for a new input $\mathbf{x}^*$, the predictive distribution of $y(\mathbf{x}^*)$ can be computed by plugging in $(\hat{\boldsymbol{\gamma}}, \hat{\eta})$, and integrating

out the trend parameter and signal variance by using the reference prior,

$$p(y(\mathbf{x}^*) \mid \mathbf{y}, \hat{\boldsymbol{\gamma}}, \hat{\eta}) = \int p(y(\mathbf{x}^*) \mid \mathbf{y}, \mathbf{b}, \sigma^2, \hat{\boldsymbol{\gamma}}, \hat{\eta}) \pi^R(\mathbf{b}, \sigma^2) d\mathbf{b} d\sigma^2. \tag{1.8}$$

Under the prior in Equation (1.4), the predictive distribution of $(y(\mathbf{x}^*) \mid \mathbf{y}, \hat{\boldsymbol{\gamma}}, \hat{\eta})$ follows a Student t distribution with $n - p_\mu$ degrees of freedom [33]:

$$y(\mathbf{x}^*) \mid \mathbf{y}, \hat{\boldsymbol{\gamma}}, \hat{\eta} \sim \mathcal{T}(\hat{y}(\mathbf{x}^*), \hat{\sigma}^2 R^{**}, n - p_\mu), \tag{1.9}$$

where

$$\hat{y}(\mathbf{x}^*) = \mathbf{h}(\mathbf{x}^*)\hat{\mathbf{b}} + \mathbf{r}^\top(\mathbf{x}^*)\tilde{\mathbf{R}}^{-1}(\mathbf{y} - \mathbf{H}\hat{\mathbf{b}}), \tag{1.10}$$

$$\hat{\sigma}^2 = \frac{1}{n - p_\mu}(\mathbf{y} - \mathbf{H}\hat{\mathbf{b}})^\top \tilde{\mathbf{R}}^{-1}(\mathbf{y} - \mathbf{H}\hat{\mathbf{b}}), \tag{1.11}$$

$$\begin{aligned} R^{**} = {} & c(\mathbf{x}^*, \mathbf{x}^*) + \hat{\eta} - \mathbf{r}^\top(\mathbf{x}^*)\tilde{\mathbf{R}}^{-1}\mathbf{r}(\mathbf{x}^*) + \left(\mathbf{h}(\mathbf{x}^*) - \mathbf{r}^\top(\mathbf{x}^*)\tilde{\mathbf{R}}^{-1}\mathbf{H}\right) \\ & \times (\mathbf{H}^\top \tilde{\mathbf{R}}^{-1}\mathbf{H})^{-1} \left(\mathbf{h}(\mathbf{x}^*) - \mathbf{r}^\top(\mathbf{x}^*)\tilde{\mathbf{R}}^{-1}\mathbf{H}\right)^\top, \end{aligned} \tag{1.12}$$

with $\mathbf{r}(\mathbf{x}^*) = (c(\mathbf{x}_1, \mathbf{x}^*), \ldots, c(\mathbf{x}_n, \mathbf{x}^*))^\top$ being the correlation between the new input and training inputs, obtained by plugging in the estimated range parameters $\hat{\boldsymbol{\gamma}}$. Parameter estimations of a GP model using a reference prior, together with the predictive distribution, are implemented in the `RobustGaSP` R package [33].

While a standard GP models a single scalar output $y(\mathbf{x}) \in \mathbb{R}$ for a given input $\mathbf{x}$, many computer simulations and physical experiments generate high-dimensional vector outputs. A classic example is the Lorenz 96 system [6], a dynamical system commonly used as a benchmark for data assimilation. In this system, the output at any given time stamp is typically a highly correlated 40-dimensional vector. As another example, the TITAN2D computer model simulates pyroclastic flows, producing outputs such as

flow height over a large number of spatio-temporal grid points [43, 44]. To accurately model these systems with massive output at each run, the standard framework must be extended to vector-valued GPs, which learn a mapping from a p_x-dimensional input $\mathbf{x}$ to a k-dimensional output $\mathbf{y}(\mathbf{x}) = (y_1(\mathbf{x}), y_2(\mathbf{x}), \ldots, y_k(\mathbf{x}))^\top$: $\mathbb{R}^{p_x} \to \mathbb{R}^k$.

A straightforward approach for modeling vector-valued output is to build an independent GP model for each output coordinate, known as the Many Single-output (MS) emulator [22, 23, 24]. However, this approach has a computational order of $\mathcal{O}(kn^3)$, which can be computationally expensive when k is large, such as 10^9. Moreover, the MS emulator ignores dependence among output coordinates, which may be inappropriate for systems such as the Lorenz 96, where strong cross-coordinate correlations are naturally present.

An alternative is to use a multi-output GPs with a separable kernel [45], which directly models correlations across both inputs and outputs, and has been used for modeling computer models [22]. Let $\mathbf{Y} = [\mathbf{y}(\mathbf{x}_1), \ldots, \mathbf{y}(\mathbf{x}_n)]$ denote the $k \times n$ observation matrix, and let $\mathbf{Y}_{\text{vec}} = \text{Vec}(\mathbf{Y}^\top)$ denote its vectorization. Under the separable kernel setting, the covariance matrix of the vectorized output is given by

$$\text{Cov}(\mathbf{Y}_{\text{vec}}) = \mathbf{\Sigma}_y \otimes \mathbf{R} + \sigma_0^2 \mathbf{I}_{nk}, \tag{1.13}$$

where $\mathbf{\Sigma}_y$ is a $k \times k$ positive semi-definite matrix that captures between-output dependence, and $\mathbf{R}$ is a $n \times n$ correlation matrix capturing the input dependence. Here, $\otimes$ represents the Kronecker product. Although this separable structure offers substantial computational advantages by decoupling input and output correlations, directly estimating $\mathbf{\Sigma}_y$ requires estimating $\mathcal{O}(k^2)$ parameters. Hence, in high-dimensional settings, accurately estimating $\mathbf{\Sigma}_y$ remains challenging and computationally expensive.

The parallel partial Gaussian process (PP-GP) model offers a computationally feasible

and robust alternative, and has shown success for high-dimensional output vector [27]. Under this framework, each output coordinate has a distinct variance parameter σ_j^2 and mean function $\mu_j(\mathbf{x}) = \mathbf{h}(\mathbf{x})\mathbf{b}_j$, where $\mathbf{h}(\mathbf{x}) \in \mathbb{R}^{p_\mu}$ is a row vector of basis functions and $\mathbf{b}_j = (b_{j,1}, \ldots, b_{j,p_\mu})$ is the trend parameter for the jth coordinate, for $j = 1, \ldots, k$. The range parameters are assumed to be shared across output coordinates. This implies that the correlation of output at any two coordinates j and j' is assumed to be the same, i.e. $\mathrm{Cor}(y_j(\mathbf{x}), y_j(\mathbf{x}')) = \mathrm{Cor}(y_{j'}(\mathbf{x}), y_{j'}(\mathbf{x}'))$, which significantly simplifies the computation. Therefore, for any input $\mathbf{x}$, the jth output coordinate is defined below:

$$y_j(\mathbf{x}) = f_j(\mathbf{x}) + \epsilon_j(\mathbf{x}), \tag{1.14}$$

where $f_j(\cdot) \sim \mathrm{GP}(\mu_j(\cdot), \sigma_j^2 c(\cdot,\cdot))$ with $c(\cdot,\cdot)$ being a correlation function, and $\epsilon_j(\mathbf{x})$ is an independent Gaussian noise with variance σ_0^2. Given a set of n inputs $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, by integrating out the latent Gaussian process $f_j(\cdot)$, the marginal distribution $\mathbf{y}_j = (y_j(\mathbf{x}_1), \ldots, y_j(\mathbf{x}_n))^\top$ follows a multivariate normal distribution:

$$\mathbf{y}_j \mid \mathbf{b}_j, \sigma_j^2, \eta, \boldsymbol{\gamma} \sim \mathcal{MN}(\mathbf{H}\boldsymbol{\mu}_j, \sigma_j^2 \tilde{\mathbf{R}}), \tag{1.15}$$

where $\mathbf{H} = (\mathbf{h}^\top(\mathbf{x}_1), \ldots, \mathbf{h}^\top(\mathbf{x}_n))^\top$ is an $n \times p_\mu$ basis function matrix, and σ_j^2 is the variance parameter. $\tilde{\mathbf{R}} = \mathbf{R} + \eta \mathbf{I}_n$ with $\mathbf{R}$ being a correlation matrix and η being a nugget parameter due to noise in observations. The (i, i') entry of $\mathbf{R}$ follows $R_{i,i'} = c(\mathbf{x}_i, \mathbf{x}_{i'})$ where $c(\cdot,\cdot)$ is a correlation function parameterized by range parameters $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_{\tilde{p}_x})$. $\tilde{p}_x = 1$ when we use an isotropic kernel and $\tilde{p}_x = p_x$ when we use a product kernel.

In the PP-GP model, we have four sets of unknown parameters: kp_μ trend parameters $(\mathbf{b}_1, \ldots, \mathbf{b}_k)$ in the mean function, k variance parameters $(\sigma_1^2, \ldots, \sigma_k^2)$, $\tilde{p}_x$ range parameters $\boldsymbol{\gamma}$, and nugget parameters η. We assume a reference prior of mean and vari-

ance parameters [27, 38] such that $\pi^R(\mathbf{b}_1, \ldots, \mathbf{b}_k, \sigma_1^2, \ldots, \sigma_k^2) \propto \frac{1}{\prod_{j=1}^k \sigma_j^2}$. The range and nugget parameters of the PP-GP model can be estimated from a mode estimator, such as the maximum likelihood estimator (MLE) or maximum marginal posterior estimator (MMPE). However, the MLE can be unstable for estimating these parameters when the sample size is small. Instead, we define the inverse range parameters $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_{\tilde{p}_x})$ with $\beta_{\tilde{p}} = 1/\gamma_{\tilde{p}}$ for $\tilde{p} = 1, \cdots \tilde{p}_x$, and use the MMPE for parameter estimation:

$$(\hat{\boldsymbol{\beta}}, \hat{\eta}) = \arg\max_{\boldsymbol{\beta}, \eta} \Big\{ \log(\mathcal{L}(\boldsymbol{\beta}, \eta)) + \log(\pi(\boldsymbol{\beta}, \eta)) \Big\}, \tag{1.16}$$

where the logarithm of the marginal likelihood after integrating out the mean and variance is

$$\log(\mathcal{L}(\boldsymbol{\beta}, \eta)) = c_1 - \frac{k}{2} \log |\tilde{\mathbf{R}}| - \frac{k}{2} \log |\mathbf{1}_n^\top \tilde{\mathbf{R}}^{-1} \mathbf{1}_n| - \frac{n-1}{2} \sum_{j=1}^{k} \log(S_j^2), \tag{1.17}$$

with $\mathbf{1}_n \in \mathbb{R}^n$ being an all-ones vector, $S_j^2 = (\mathbf{y}_j - \mathbf{H}\hat{\mathbf{b}}_j)^\top \tilde{\mathbf{R}}^{-1} (\mathbf{y}_j - \mathbf{H}\hat{\mathbf{b}}_j)$ and c_1 is a normalizing constant not related to $(\boldsymbol{\beta}, \eta)$.

The jointly robust prior [40] is used as a default choice for prior in the RobustGaSP package [33]

$$\log\big(\pi(\boldsymbol{\beta}, \eta)\big) = c_2 + a \log\left(\sum_{\tilde{p}=1}^{\tilde{p}_x} C_{\tilde{p}} \beta_{\tilde{p}} + \eta \right) - b \left(\sum_{\tilde{p}=1}^{\tilde{p}_x} C_{\tilde{p}} \beta_{\tilde{p}} + \eta \right), \tag{1.18}$$

where c_2 is a normalizing constant not relevant to $(\boldsymbol{\beta}, \boldsymbol{\eta})$ and the default choice of prior parameters in the `RobustGaSP` package is $a = 0.2$, $b = n^{-1/\tilde{p}_x}(a + \tilde{p}_x)$ and $C_{\tilde{p}} = n^{-1/\tilde{p}_x} |x_{\tilde{p}}^{\max} - x_{\tilde{p}}^{\min}|$ with $x_{\tilde{p}}^{\max}$ and $x_{\tilde{p}}^{\min}$ being the largest and smallest input values in the $\tilde{p}$th coordinate, respectively. For deterministic output values, including the cases where numerical error of the simulations is negligible, the nugget η may be set to be zero.

Denote $\mathbf{Y} = [\mathbf{y}(\mathbf{x}_1), \ldots, \mathbf{y}(\mathbf{x}_n)]$ as a $k \times n$ observation matrix. By integrating out the mean and variance parameters, the predictive distribution for $\mathbf{x}^*$ follows a noncentral Student's t distribution with $n - p_\mu$ degrees of freedom [27]:

$$y_j(x^*) \mid \mathbf{Y}, \hat{\boldsymbol{\gamma}}, \hat{\eta} \sim \mathcal{T}(\hat{y}_j(\mathbf{x}^*), \hat{\sigma}_j^2 R^{**}, n - p_\mu), \tag{1.19}$$

where the predictive mean and scale parameters follow

$$\hat{y}_j(\mathbf{x}^*) = \mathbf{h}(\mathbf{x}^*)\hat{\mathbf{b}}_j + \mathbf{r}^\top(\mathbf{x}^*)\tilde{\mathbf{R}}^{-1}(\mathbf{y}_j - \mathbf{H}\hat{\mathbf{b}}_j), \tag{1.20}$$

$$\hat{\sigma}_j^2 = \frac{1}{n - p_\mu}(\mathbf{y}_j - \mathbf{H}\hat{\mathbf{b}}_j)^\top \tilde{\mathbf{R}}^{-1}(\mathbf{y}_j - \mathbf{H}\hat{\mathbf{b}}_j), \tag{1.21}$$

with $\hat{\mathbf{b}}_j = (\mathbf{H}^\top \tilde{\mathbf{R}}^{-1}\mathbf{H})^{-1}\mathbf{H}^\top \tilde{\mathbf{R}}^{-1}\mathbf{y}_j$ being the generalized least squares estimator of the mean, $\mathbf{r}(\mathbf{x}^*) = (c(\mathbf{x}_1, \mathbf{x}^*), \ldots, c(\mathbf{x}_n, \mathbf{x}^*))^\top$ being the correlation between the training inputs and the new input, and R^{**} being defined in Equation (1.12).

The predictive mean in Equation (1.20) is typically used for prediction. The uncertainty of the prediction can be quantified by the predictive interval. The PP-GP model has been implemented in different computational platforms such as MATLAB, Python and R [33].

Compared with other GP approaches for emulating vectorized output, one advantage of the PP-GP model comes with the computational scalability when the output dimension k is large. Computing the predictive mean for k coordinates in Equation (1.20) requires $\mathcal{O}(nk) + \mathcal{O}(n^3)$ operations. The highest cost of PP-GP typically comes from estimating the range and nugget parameters, which requires $\mathcal{O}(M_{pp}n^3 + M_{pp}n^2k)$ operations for M_{pp} iterations in numerical optimization. When the number of time points n is large, approximation methods, such as the inducing point method [46] and the Vecchia approach [47], may be used for approximating the likelihood function of Gaussian processes. The

PP-GP is equivalent to the separable GPs with a diagonal output correlation matrix, i.e., $\mathbf{\Sigma}_y = \text{diag}(\sigma_1^2, \ldots, \sigma_k^2)$, while PP-GP has a smaller computational order.

1.2 Latent factor models of correlated data

The latent factor model provides a statistical framework for modeling vector-valued outputs by mapping high-dimensional observations into a low-dimensional latent space. By representing the output vector as a linear combination of unobservable variables, these models capture underlying dynamics with scientific meaning. For example, in geophysical applications, the latent factors correspond to unobservable slip processes on fault segments, which can be recovered from deformation measurements. Moreover, the low-rank representation of the observations is particularly useful for dimensionality reduction in large-scale data. Furthermore, by separating the signal from noise, latent factor models also serve as an effective denoising tool.

Let $\mathbf{y}(\mathbf{x})$ denote a k-dimensional observation with a p_x-dimensional input $\mathbf{x}$. The vector-valued data can be modeled by a latent factor model:

$$\mathbf{y}(\mathbf{x}) = \mathbf{A}\mathbf{z}(\mathbf{x}) + \boldsymbol{\epsilon}(\mathbf{x}), \tag{1.22}$$

where $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_d]$ is a $k \times d$ factor loading matrix mapping the latent factor to the observation, $\mathbf{z}(\mathbf{x}) = [z_1(\mathbf{x}), \ldots, z_d(\mathbf{x})]^\top$ is a d-dimensional latent factor ($d \leq k$), and $\boldsymbol{\epsilon}(\mathbf{x}) \in \mathbb{R}^k$ is independent Gaussian noise. Given a set of input vectors $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, denote $\mathbf{Y} = [\mathbf{y}(\mathbf{x}_1), \ldots, \mathbf{y}(\mathbf{x}_n)]$ as the observation matrix, and $\mathbf{z}_l = [z_l(\mathbf{x}_1), \ldots, z_l(\mathbf{x}_n)]^\top$ as the l-th latent process, for $l = 1, \ldots, d$.

One classic example of the latent factor model is the linear model of coregionalization (LMC) [48, 49], also known as a semi-parametric latent factor model [50], where each

latent factor is assumed to independently follow a zero-mean GP with correlation function $c_l(\cdot,\cdot)$, indicating that $\mathbf{z}_l = [z_l(\mathbf{x}_1),\ldots,z_l(\mathbf{x}_n)]^\top \sim \mathcal{MN}(\mathbf{0},\mathbf{\Sigma}_l)$, where $\mathbf{\Sigma}_l = \sigma_l^2\mathbf{R}_l$ and $\mathbf{R}_l$ is the correlation matrix constructed by $c_l(\cdot,\cdot)$.

The GP model with a separable kernel defined in Equation (1.13) is a special case of LMC. Specifically, when all latent processes share the same correlation function, i.e., $\mathbf{R}_1 = \cdots = \mathbf{R}_d = \mathbf{R}$, the covariance matrix of the vectorized output matrix is

$$\mathrm{Cov}(\mathrm{Vec}(\mathbf{Y}^\top)) = \left(\mathbf{A}\mathrm{diag}(\sigma_1^2,\ldots,\sigma_d^2)\mathbf{A}^\top\right)\otimes\mathbf{R} + \sigma_0^2\mathbf{I}_{nk}, \tag{1.23}$$

which has the same structure as Equation (1.13) where $\mathbf{\Sigma}_y = \mathbf{A}\mathrm{diag}(\sigma_1^2,\ldots,\sigma_d^2)\mathbf{A}^\top$. Furthermore, consider a spatial-temporal setting, where the input is $\mathbf{x} = (s,t)$, with s and t denoting the spatial location and time points, respectively. Suppose $\mathbf{A}$ is orthogonal and $d = k$. Defining the $\mathbf{R}_s := \mathbf{A}\mathrm{diag}(\sigma_1^2,\ldots,\sigma_d^2)\mathbf{A}^\top$, the covariance of the vectorized output can be written as $\mathrm{Cov}(\mathrm{Vec}(\mathbf{Y}^\top)) = \mathbf{R}_s\otimes\mathbf{R} + \sigma_0^2\mathbf{I}_{nk}$, which is the separable spatial-temporal covariance structure, where $\mathbf{R}_s$ and $\mathbf{R}$ represent the spatial and temporal covariance matrices, respectively. Under this formulation, the columns of $\mathbf{A}$ correspond to the eigenvectors of the spatial covariance matrix, while $(\sigma_1^2,\ldots,\sigma_d^2)$ are the associated eigenvalues. This interpretation suggests that the factor loading matrix may be specified through the eigenvectors of a kernel-parameterized spatial covariance. Such an approach is explored in [51], which leverages the Kalman filter to allow scalable computation in large-scale spatial and spatial-temporal data, even when the observations have irregular missing patterns.

Estimating range parameters is often a key challenge in GPs, as it can be computationally demanding and numerically unstable. The shared correlation assumption in Equation (1.23) simplifies this process by introducing only a single range parameter. However, this simplification is restrictive, as it forces all latent processes to share the

same smoothness. In practice, vector-valued outputs often arise from latent processes operating at different scales. For example, in financial time series data, a large range parameter represents a persistent long-term trend, while a small range parameter captures transient, high-frequency fluctuations. To allow for such heterogeneity, each latent process can be assigned its own correlation function, yielding the covariance structure:

$$\mathrm{Cov}(\mathrm{Vec}(\mathbf{Y}^\top)) = \sum_{l=1}^{d} \sigma_l^2(\mathbf{a}_l\mathbf{a}_l^\top) \otimes \mathbf{R}_l + \sigma_0^2\mathbf{I}_{nk}, \tag{1.24}$$

which is a sum of separable kernel structures.

The factor loading matrix $\mathbf{A}$ can be either fixed or estimated from data. In many applications, physically motivated basis functions, such as discrete Fourier basis and Green's functions, are commonly used as fixed loading matrices. One example is the differential dynamic microscopy (DDM) [52, 53], a method for extracting rheological and mechanical properties of materials from microscopy video. As shown in [20], the underlying DDM model can be expressed as the latent factor model in Equation (1.22), where the loading matrix $\mathbf{A}$ is specified as a two-dimensional discrete Fourier basis. Within this framework, parameter estimation is equivalent to minimizing the temporal autocorrelation of the projected observations in Fourier space, enabling accurate inference of material properties without explicit particle segmentation [54, 55], which is often challenging in microscopy images of optically dense cells [56]. Another example arises from the use of Green's functions to link field data to latent physical processes across spatial sites, such as slip estimation [57] and volcanology [58, 59]. In particular, the Network Inverse Filter method [2] employs Green's function to connect the unobservable slip process to ground deformation measurement, therefore enabling volcanic hazard quantification. This model can be written as a latent factor model [60], in which the factor loading matrix is derived from the eigenvectors of a matrix constructed by the Green's function.

Alternatively, the factor loading matrix can be estimated directly from the data. A classic example is principal component analysis (PCA) [61], in which the loading matrix is given by the leading eigenvectors of the sample covariance matrix. The spanned subspace is the same as that obtained from the maximum marginal likelihood estimator (MMLE) in probabilistic principal component analysis (PPCA). In PPCA, however, the latent factors are assumed to be independently distributed as standard Gaussian distributions, which is restrictive for modeling correlated data. To address this limitation, the generalized probabilistic principal component analysis (GPPCA) [19] extends PPCA by modeling each latent process as a Gaussian process, thereby allowing dependence across outputs. Under the orthogonality assumption on $\mathbf{A}$, when $\mathbf{\Sigma}_1 = \cdots = \mathbf{\Sigma}_d = \mathbf{\Sigma}$, the MMLE of the factor loading matrix admits a closed-form solution, as established in Theorem 2 of [19]. In a more general setting where the covariance matrices differ across latent processes, no closed-form estimator is available for $\mathbf{A}$, and numerical optimization in the Stiefel manifold is required, which can be computationally expensive and unstable. Another data-driven approach for estimating the factor loading matrix from data is proposed in [62, 21], where $\hat{\mathbf{A}}$ is derived from the eigenvectors of a nonnegative definite matrix constructed by aggregating the sample autocovariance matrices across multiple lags. However, in a high-dimensional setting, estimating the $k \times d$ factor loading matrix can be computationally expensive and memory-intensive for large k. One approach to overcome this challenge is to impose sparsity constraints on the loading matrix, such as the spike-and-slab priors [63, 64]. More recently, [65] proposes a tensor structure on the factor loading matrix, which also acts as a parameter reduction tool. However, this approach does not place a probabilistic model on the latent factors. Consequently, it only provides point estimates of the latent factors, without uncertainty quantification. Moreover, interpolation and forecasting are not supported, and the underlying dynamical process cannot be learned.

1.3 Dynamic linear model, the Kalman Filter and the Rauch-Tung-Striebel Smoother

The Dynamic Linear Model (DLM) [66] is another widely used method for modeling multivariate time series and dynamic data. Consider a sequence of continuous time points $\{t_1, t_2, \ldots, t_n\}$, with $t_i \in \mathbb{R}^+$ and the corresponding time-dependent observations $\mathbf{y}(t_1), \mathbf{y}(t_2), \ldots, \mathbf{y}(t_n) \in \mathbb{R}^k$. A DLM models the time series through the observation and state equations:

$$\mathbf{y}(t_i) = \mathbf{F}(t_i)\boldsymbol{\theta}(t_i) + \boldsymbol{\epsilon}(t_i), \tag{1.25}$$

$$\boldsymbol{\theta}(t_i) = \mathbf{G}(t_i)\boldsymbol{\theta}(t_{i-1}) + \mathbf{w}(t_i) \tag{1.26}$$

where $\boldsymbol{\theta}(t_i)$ is a p_θ-dimensional latent state process with $\boldsymbol{\theta}(t_1) \sim \mathcal{MN}(0, \mathbf{W}(t_1))$, $\mathbf{F}(t_i) \in \mathbb{R}^{k \times p_\theta}$ links the latent state to the observation, $\mathbf{G}(t_i) \in \mathbb{R}^{p_\theta \times p_\theta}$ is a transition matrix describing the evolution of the latent state over time. The state innovation term follows $\mathbf{w}(t_i) \sim \mathcal{MN}(\mathbf{0}, \mathbf{W}(t_i))$, while the observation noise is given by $\boldsymbol{\epsilon}(t_i) \sim \mathcal{MN}(\mathbf{0}, \mathbf{E}(t_i))$.

The DLM can be viewed as a special case of the latent factor model in Equation (1.22) when the input is one-dimensional, i.e.,$\mathbf{x}_i := t_i$. In this case, the latent state $\boldsymbol{\theta}(t_i)$ plays the role of the latent factor $\mathbf{z}(\mathbf{x})$. When $\mathbf{F}(t_i)$ is time-invariant, i.e., $\mathbf{F}(t_i) = \mathbf{F}$ for all i, it is analogous to the factor loading matrix $\mathbf{A}$ in Equation (1.22).

DLMs are also closely connected to GP models. Specifically, a scalar-valued GP model with a Matérn kernel and a half-integer roughness parameter ν admits an exact DLM representation [67, 68]. For example, the Matérn kernel with $\nu = \frac{1}{2}$, also known as the exponential kernel,

$$K(t_i, t_{i'}) = \sigma^2 \exp\left(-\frac{|t_i - t_{i'}|}{\gamma}\right). \tag{1.27}$$

A GP with this kernel is equivalent to an Ornstein-Uhlenbeck process, which can be represented as a DLM with a scalar observation and scalar state, i.e., $k = p_\theta = 1$. The corresponding model parameters are:

$$F(t_i) = 1, G(t_i) = \exp\left(-\frac{\Delta t_i}{\gamma}\right), W(t_i) = \sigma^2(1-G^2(t_i)), W(t_1) = \sigma^2, E(t_i) = \sigma_0^2, \quad (1.28)$$

where $\Delta t_i = t_i - t_{i-1}$ for $i = 2, \ldots, n$.

Another example is the Matérn kernel with $\nu = \frac{5}{2}$, given by

$$K(t_i, t_{i'}) = \left(1 + \frac{\sqrt{5}|t_i - t_{i'}|}{\gamma} + \frac{5|t_i - t_{i'}|^2}{3\gamma^2}\right) \exp\left(-\frac{\sqrt{5}|t_i - t_{i'}|}{\gamma}\right). \quad (1.29)$$

Let $\lambda = \frac{\sqrt{5}}{\gamma}$. As shown in [69], a GP with the Matérn kernel in Equation (1.29) has a DLM representation such that

$$\mathbf{F}(t_i) = [1, 0, 0], \quad (1.30)$$

$$\mathbf{G}(t_i) = e^{-\frac{\lambda\Delta t_i}{2}} \begin{pmatrix} (\lambda\Delta t_i)^2 + 2\lambda\Delta t + 2 & 2\lambda(\Delta t_i)^2 + 2\Delta t_i & (\Delta t_i)^2 \\ -\lambda^3(\Delta t_i)^2 & -2((\lambda\Delta t_i)^2 - \lambda\Delta t_i - 1) & 2 - \lambda(\Delta t_i)^2 \\ \lambda^4(\Delta t_i)^2 - 2\lambda^3\Delta t_i & 2(\lambda^3(\Delta t_i)^2 - 3\lambda^2\Delta t_i) & (\lambda\Delta t_i)^2 - 4\lambda\Delta t_i + 2 \end{pmatrix}, \quad (1.31)$$

$$\mathbf{W}(t_1) = \sigma^2 \begin{pmatrix} 1 & 0 & -\lambda^2/3 \\ 0 & \lambda^2/3 & 0 \\ -\lambda^2/3 & 0 & \lambda^4 \end{pmatrix}, \quad (1.32)$$

$$\mathbf{W}(t_i) = \frac{4\sigma^2\lambda^5}{3} \begin{pmatrix} W_{1,1}(t_i) & W_{1,2}(t_i) & W_{1,3}(t_i) \\ W_{2,1}(t_i) & W_{2,2}(t_i) & W_{2,3}(t_i) \\ W_{3,1}(t_i) & W_{3,2}(t_i) & W_{3,3}(t_i) \end{pmatrix}, \quad (1.33)$$

where

$$
\begin{aligned}
W_{1,1}(t_i) &= \frac{e^{-2\lambda\Delta t_i}(3 + 6\lambda\Delta t_i + 6(\lambda\Delta t_i)^2 + 4(\lambda\Delta t_i)^3 + 2(\lambda\Delta t_i)^4) - 3}{-4\lambda^5}, \\
W_{1,2}(t_i) &= W_{2,1}(t_i) = \frac{e^{-2\lambda\Delta t_i}}{2}, \\
W_{1,3}(t_i) &= W_{3,1}(t_i) = \frac{e^{-2\lambda\Delta t_i}(1 + 2\lambda\Delta t_i + 2(\lambda\Delta t_i)^2 + 4(\lambda\Delta t_i)^3 - 2(\lambda\Delta t_i)^4) - 1}{4\lambda^3}, \\
W_{2,2}(t_i) &= \frac{e^{-2\lambda\Delta t_i}(1 + 2\lambda\Delta t_i + 2(\lambda\Delta t_i)^2 - 4(\lambda\Delta t_i)^3 + 2(\lambda\Delta t_i)^4) - 1}{-4\lambda^3}, \\
W_{2,3}(t_i) &= W_{3,2}(t_i) = \frac{e^{-2\lambda\Delta t_i}(4 - 4\lambda\Delta t_i + (\lambda\Delta t_i)^2)}{2}, \\
W_{3,3}(t_i) &= \frac{e^{-2\lambda\Delta t_i}(-3 + 10\lambda\Delta t_i - 22(\lambda\Delta t_i)^2 + 12(\lambda\Delta t_i)^3 - 2(\lambda\Delta t_i)^4) + 3}{4\lambda}.
\end{aligned}
$$

Directly evaluating the log-marginal likelihood and computing the predictive distribution in a GP model usually requires $\mathcal{O}(n^3)$ computation order due to the inversion of an $n \times n$ covariance matrix, as shown in Equation (1.6)-(1.10). However, when a GP kernel has a DLM representation, both the log-marginal likelihood and the conditional distribution of the latent function can be computed in $\mathcal{O}(n)$ time by Kalman filtering (KF) [25] and Rauch–Tung–Striebel (RTS) smoother [26]. Lemma 1-2 provides the details on how to implement the algorithms for the case of scalar observations. Proofs can be found in Appendix A.1 and A.2.

Lemma 1 (Kalman Filter) *Denote $\mathbf{y}_{1:i} = [y(t_1), \ldots, y(t_i)]$ as the first i observations. For $i = 1, 2, \ldots, n$, given $\boldsymbol{\theta}(t_{i-1}) \mid \mathbf{y}_{1:i-1} \sim \mathcal{MN}(\mathbf{m}(t_{i-1}), \mathbf{C}(t_{i-1}))$, one can iteratively compute the conditional distribution $\boldsymbol{\theta}(t_i) \mid \mathbf{y}_{1:i}$ by the following three steps:*

1. *The one-step-ahead prediction of $\boldsymbol{\theta}(t_i)$ given $\mathbf{y}_{1:(i-1)}$ is*

$$
\boldsymbol{\theta}(t_i) \mid \mathbf{y}_{1:(i-1)} \sim \mathcal{MN}(\mathbf{a}(t_i), \mathbf{R}(t_i)), \tag{1.34}
$$

where $\mathbf{a}(t_i) = \mathbf{G}(t_i)\mathbf{m}(t_{i-1})$ and $\mathbf{R}(t_i) = \mathbf{G}(t_i)\mathbf{C}(t_{i-1})\mathbf{G}^\top(t_i) + \mathbf{W}(t_i)$.

2. *The one-step-ahead predictive distribution of* $y(t_i)$ *given* $\mathbf{y}_{1:(i-1)}$ *is*

$$y(t_i) \mid \mathbf{y}_{1:(i-1)} \sim \mathcal{N}(f(t_i), Q(t_i)), \tag{1.35}$$

where $f(t_i) = \mathbf{F}(t_i)\mathbf{a}(t_i)$ *and* $Q(t_i) = \mathbf{F}(t_i)\mathbf{R}(t_i)\mathbf{F}^\top(t_i) + E(t_i)$.

3. *The filtering distribution of* $\boldsymbol{\theta}(t_i)$ *given* $\mathbf{y}_{1:i}$ *is*

$$\boldsymbol{\theta}(t_i) \mid \mathbf{y}_{1:i} \sim \mathcal{MN}(\mathbf{m}(t_i), \mathbf{C}(t_i)), \tag{1.36}$$

where

$$\begin{aligned}\mathbf{m}(t_i) &= \mathbf{a}(t_i) + \mathbf{R}(t_i)\mathbf{F}^\top(t_i)Q^{-1}(t_i)(y(t_i) - f(t_i)) \\ \mathbf{C}(t_i) &= \mathbf{R}(t_i) - \mathbf{R}(t_i)\mathbf{F}^\top(t_i)Q^{-1}(t_i)\mathbf{F}(t_i)\mathbf{R}(t_i).\end{aligned}$$

Performing the KF on a p_θ-dimensional state vector $\boldsymbol{\theta}(t_i)$ with scalar observation $y(t_i)$ requires $\mathcal{O}(np_\theta^2)$ operations in total, which enables efficient evaluation of the marginal likelihood of $\mathbf{y}_{1:n}$ when p_θ is small. Specifically, the likelihood in Equation (1.3) can be computed via the chain rule of conditional distribution: $p(\mathbf{y}_{1:n}) = p(y(t_1)) \prod_{i=2}^n p(y(t_i) \mid \mathbf{y}_{1:(i-1)})$, which yields

$$p(\mathbf{y}_{1:n} \mid \sigma^2, \gamma, \eta) = \prod_{i=1}^n (2\pi Q(t_i))^{-\frac{1}{2}} \exp\left(-\sum_{i=1}^n \frac{(y(t_i) - f(t_i))^2}{2Q(t_i)}\right). \tag{1.37}$$

Equations (1.3) and (1.37) build the equivalence between the following quantities [18]:

$$|\tilde{\mathbf{R}}| = \prod_{i=1}^n Q(t_i), \quad \frac{\mathbf{y}^\top \tilde{\mathbf{R}}^{-1}\mathbf{y}}{\sigma^2} = \sum_{i=1}^n \frac{(y(t_i) - f(t_i))^2}{Q(t_i)}. \tag{1.38}$$

Since $Q(t_i)$ is a scalar, evaluating this likelihood does not require any matrix inversion,

so the total computational cost is $\mathcal{O}(np_\theta^2)$. For the exponential kernel in Equation (1.27), this reduces further to $\mathcal{O}(n)$. In addition, Lemma 1 can be extended to k-dimensional observations, in which case $\mathbf{Q}(t_i)$ becomes a $k \times k$ matrix that must be inverted at each time step, incurring a cost of $\mathcal{O}(k^3)$, which becomes prohibitive for high-dimensional observations.

Lemma 2 provides a sequential algorithm to compute the posterior distribution of $\boldsymbol{\theta}(t_i)$ given the full data $\mathbf{y}_{1:n}$, with a total computational order of $\mathcal{O}(np_\theta^3)$. For the exponential kernel in Equation (1.27), this complexity simplifies to $\mathcal{O}(n)$.

Lemma 2 (Rauch–Tung–Striebel Smoother) *Suppose the posterior distribution of* $\boldsymbol{\theta}(t_{i+1})$ *conditional on full data* $\mathbf{y}_{1:n}$ *is* $\boldsymbol{\theta}(t_{i+1}) \mid \mathbf{y}_{1:n} \sim \mathcal{MN}(\mathbf{s}(t_{i+1}), \mathbf{S}(t_{i+1}))$ *for* $i = n-1, n-2, ..., 1$, *then the conditional distribution* $\boldsymbol{\theta}(t_i) \mid \mathbf{y}_{1:n}$ *is*

$$\boldsymbol{\theta}(t_i) \mid \mathbf{y}_{1:n} \sim \mathcal{MN}(\mathbf{s}(t_i), \mathbf{S}(t_i)), \tag{1.39}$$

where

$$\begin{aligned}
\mathbf{s}(t_i) &= \mathbf{m}(t_i) + \mathbf{C}(t_i)\mathbf{G}^\top(t_{i+1})\mathbf{R}^{-1}(t_{i+1})(\mathbf{s}(t_{i+1}) - \mathbf{a}(t_{i+1})) \\
\mathbf{S}(t_i) &= \mathbf{C}(t_i) - \mathbf{C}(t_i)\mathbf{G}^\top(t_{i+1})\mathbf{R}^{-1}(t_{i+1})(\mathbf{R}(t_{i+1}) - \mathbf{S}(t_{i+1}))\mathbf{R}^{-1}(t_{i+1})\mathbf{G}^\top(t_{i+1})\mathbf{C}(t_i).
\end{aligned}$$

Furthermore, the covariance between $\boldsymbol{\theta}(t_i)$ *and* $\boldsymbol{\theta}(t_{i+1})$ *conditional on* $\mathbf{y}_{1:n}$ *is*

$$\tilde{\mathbf{S}}(t_i) = Cov[\boldsymbol{\theta}(t_i), \boldsymbol{\theta}(t_{i+1}) \mid \mathbf{y}_{1:n}] = \mathbf{C}(t_i)\mathbf{G}^\top(t_{i+1})\mathbf{R}^{-1}(t_{i+1})\mathbf{S}(t_{i+1}), \tag{1.40}$$

for $i = 1, 2, \ldots, n-1$.

1.4 EM algorithm

The KF and RTS smoother introduced in Chapter 1.3 compute the conditional distribution of the state vector $\boldsymbol{\theta}(t_i)$ given the observations, assuming the parameters $\{\mathbf{F}(t_i)\}_{i=1}^n, \{\mathbf{E}(t_i)\}_{i=1}^n, \{\mathbf{G}(t_i)\}_{i=1}^n, \{\mathbf{W}(t_i)\}_{i=1}^n$ in DLM are known. In practice, however, these parameters are unknown and must be estimated from observed data.

The expectation-maximization (EM) algorithm [70] provides an elegant and general method for parameter estimation in models with latent variables or missing data. Since the latent factor model and DLM contain latent variables, they naturally fit within this framework. The key idea is to avoid directly maximizing the marginal likelihood of the observations, which can be computationally expensive and numerically unstable, by instead working with the joint likelihood of the observations and latent factors (or missing data), and iteratively optimizing its conditional expectation. Another well-known example of the EM algorithm is the Gaussian mixture model (GMM) [70], which is described below.

Example 1 (Gaussian Mixture Model) *Consider the generative process for a Gaussian Mixture Model (GMM):*

1. *Generate the cluster l from some discrete distribution characterized by parameters $\mathbf{w}$, such as $P(z = l \mid \mathbf{w}) = w_l$, where w_l is the mixture weight for cluster l, satisfying $0 \leq w_l \leq 1$ and $\sum_l w_l = 1$.*

2. *Given the cluster assignment, generate the observation $\mathbf{y}$ from the corresponding Gaussian distribution $(\mathbf{y} \mid z = l) \sim \mathcal{MN}(\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)$.*

GMM is a classic example of latent factor models and is commonly used for unsupervised clustering, where observations are assumed to arise from a mixture of several latent groups, each represented by a Gaussian distribution.

Let $\mathbf{y}_i$ denote the i-th observation and $\mathbf{z}_i$ the corresponding latent variable, with $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_n]$ and $\mathbf{Z} = [\mathbf{z}_1, \cdots \mathbf{z}_n]$ being the observation and latent factor matrices, respectively, and $\boldsymbol{\Theta}$ the set of model parameters. The EM algorithm starts from the joint log-likelihood between observations $\mathbf{Y}$ and latent factor $\mathbf{Z}$:

$$l(\boldsymbol{\Theta}; \mathbf{Y}, \mathbf{Z}) = \log(p(\mathbf{Y} \mid \mathbf{Z}, \boldsymbol{\Theta})) + \log(p(\mathbf{Z} \mid \boldsymbol{\Theta})), \tag{1.41}$$

which is more tractable than the marginal likelihood obtained by integrating out $\mathbf{Z}$. The EM algorithm maximizes the marginal likelihood indirectly by iterating between two steps:

1. **Expectation step (E-step)**: Compute the expected value of $l(\boldsymbol{\Theta}; \mathbf{Y}, \mathbf{Z})$ with respect to the conditional distribution of the latent variables or missing data, given the current parameter estimates $\boldsymbol{\Theta}^{(m)}$:

$$Q(\boldsymbol{\Theta} \mid \boldsymbol{\Theta}^{(m)}) = \mathbb{E}_{\mathbf{Z}|\mathbf{Y}, \boldsymbol{\Theta}^{(m)}}[l(\boldsymbol{\Theta}; \mathbf{Y}, \mathbf{Z})]. \tag{1.42}$$

   The intuition behind this step is that, because the latent variables are unobserved, we cannot directly evaluate the joint log-likelihood. Instead, we use our current estimate of the parameters to "fill in" the missing information by replacing the latent variables with their conditional expectation under the current parameter estimates. In DLM, this step is carried out efficiently using the Kalman filter and RTS smoother.

2. **Maximization step (M-step)**: Maximize $Q(\boldsymbol{\Theta} \mid \boldsymbol{\Theta}^{(m)})$ with respect to $\boldsymbol{\Theta}$ to obtain updated parameter estimates.

$$\boldsymbol{\Theta}^{(m+1)} = \arg\max_{\boldsymbol{\Theta}} Q(\boldsymbol{\Theta} \mid \boldsymbol{\Theta}^{(m)}). \tag{1.43}$$

When using the EM algorithm to estimate parameters, we typically assume the auxiliary function $Q(\boldsymbol{\Theta} \mid \boldsymbol{\Theta}^{(m)})$ is easy to optimize. In models such as GMM and PPCA, the M-step admits a closed-form solution, making each iteration computationally efficient.

When the M-step cannot be solved in closed-form, iterative optimization methods such as gradient descent may be employed, although this introduces nested iterations and additional computational cost. The generalized EM algorithm (GEM) [70] offers a useful relaxation: rather than requiring the global maximizer in the M-step, GEM only requires finding $\boldsymbol{\Theta}^{(m+1)}$ such that $Q(\boldsymbol{\Theta}^{(m+1)}|\boldsymbol{\Theta}^{(m)}) \geq Q(\boldsymbol{\Theta}^{(m)}|\boldsymbol{\Theta}^{(m)})$. That is, any update that does not decrease $Q(\cdot \mid \cdot)$ is acceptable. This relaxation is sufficient to preserve the monotone ascent property of the marginal likelihood, which is important for establishing convergence to a stationary point. Lemma 3 formalizes the monotone ascent property [70].

Lemma 3 (Ascending property of EM) *The log marginal likelihood is guaranteed not to decrease after each EM iteration. That is,*

$$l(\boldsymbol{\Theta}^{(m+1)}; \mathbf{Y}) \geq l(\boldsymbol{\Theta}^{(m)}; \mathbf{Y}), \tag{1.44}$$

where $l(\boldsymbol{\Theta}; \mathbf{Y}) = \log(p(\mathbf{Y} \mid \boldsymbol{\Theta})) = \log \int p(\mathbf{Y}, \mathbf{Z} \mid \boldsymbol{\Theta}) d\mathbf{Z}$.

This monotone ascent property ensures that the EM algorithm does not decrease the marginal log-likelihood at any iteration. Under regular conditions, it further guarantees convergence to a stationary point of the marginal likelihood, as shown in Theorem 1 [71].

Theorem 1 (Global convergence of the EM-algorithm) *Suppose*

1. $\boldsymbol{\Theta} \in \boldsymbol{\Omega}$, *where* $\boldsymbol{\Omega}$ *is a subset in a finite-dimensional Euclidean space.*

2. $\boldsymbol{\Omega}_{\boldsymbol{\Theta}_0} = \{\boldsymbol{\Theta} \in \boldsymbol{\Omega} : l(\boldsymbol{\Theta}; \mathbf{Y}) \geq l(\boldsymbol{\Theta}_0; \mathbf{Y})\}$ *is compact for any* $l(\boldsymbol{\Theta}_0; \mathbf{Y}) > -\infty$.

3. $l(\boldsymbol{\Theta}; \mathbf{Y})$ *is continuous in* $\boldsymbol{\Omega}$ *and differentiable in the interior of* $\boldsymbol{\Omega}$.

4. $Q(\boldsymbol{\Theta} \mid \boldsymbol{\Theta}^{(m)})$ *is continuous in both* $\boldsymbol{\Theta}$ *and* $\boldsymbol{\Theta}^{(m)}$.

Then by the global convergence theorem [72], all the limit points of an EM algorithm are stationary points of marginal log-likelihood and $l(\boldsymbol{\Theta}^{(m)}; \mathbf{Y})$ *converges monotonically to* $l(\boldsymbol{\Theta}^*)$ *for some stationary point* $\boldsymbol{\Theta}^*$.

Theorem 1 only guarantees convergence to a stationary point of the marginal likelihood, which may be a local maximum or a saddle point. In practice, one strategy is to run the algorithm from multiple random initializations and select the solution with the largest marginal likelihood. For the latent factor model in Equation (1.22), the factor loading matrix can be initialized as the left singular vectors of the observation matrix $\mathbf{Y}$, which often leads to faster convergence and better solutions [73, 19].

1.5 Dynamic mode decomposition

Dynamic mode decomposition (DMD) [28] is a data-driven approach to obtain a reduced-rank representation of complex, high-dimensional dynamical systems, which has quickly gained popularity for approximating and forecasting nonlinear dynamic systems [74]. DMD is also related to the Koopman operator theory [75], where the evolution of a nonlinear dynamical system is represented through a linear operator acting on observable functions of the system. Specifically, DMD seeks the best-fit linear transition matrix in the space spanned by observations, yielding a finite-dimensional approximation of the Koopman operator. Hence, the DMD eigenvalues and modes approximate the Koopman eigenvalues and Koopman modes, respectively.

DMD relies on a linear transition matrix assumption such that $\mathbf{y}(t_{i+1}) \approx \mathbf{A}\mathbf{y}(t_i)$. Given a sequences of observations $[\mathbf{y}(t_1), \mathbf{y}(t_2), \ldots, \mathbf{y}(t_n)]$ uniformly sampled in time, with $\mathbf{y}(t_i) \in \mathbb{R}^k$ for $i = 1, \ldots, n$, we define two data matrices $\mathbf{Y}_1 = [\mathbf{y}(t_1), \ldots, \mathbf{y}(t_{n-1})]$ and $\mathbf{Y}_2 = [\mathbf{y}(t_2), \ldots, \mathbf{y}(t_n)]$. By minimizing the discrepancy between the current observation and the one reconstructed from the previous observation, the linear transition matrix can be estimated by

$$\hat{\mathbf{A}} = \arg\min_{\mathbf{A}} \|\mathbf{Y}_2 - \mathbf{A}\mathbf{Y}_1\| = \mathbf{Y}_2(\mathbf{Y}_1)^+, \tag{1.45}$$

where $\|\cdot\|$ is the L_2 norm or Frobenius norm, and $(\mathbf{Y}_1)^+$ is the Moore-Penrose pseudo-inverse of $\mathbf{Y}_2$.

For high-dimensional systems, such as $k = 10^7$, $\mathbf{A} \in \mathbb{R}^{k \times k}$ is too large to compute or store directly. Instead, [29] provides the exact DMD algorithm to efficiently compute the eigenvalues and eigenvectors of $\hat{\mathbf{A}}$. The algorithm is described below:

1. Compute the singular value decomposition $\mathbf{Y}_1 = \mathbf{U}_1\mathbf{D}_1\mathbf{V}_1^\top$ and keep the r largest singular values and their associated singular vectors to obtain $\mathbf{D}_1^{(r)}$, $\mathbf{U}_1^{(r)}$ and $\mathbf{V}_1^{(r)}$. The estimated transition matrix in Equation (1.45) can be approximated by

$$\hat{\mathbf{A}}^{(r)} := \mathbf{Y}_2\mathbf{V}_1^{(r)}(\mathbf{D}_1^{(r)})^{-1}\mathbf{U}_1^{(r)\top}, \tag{1.46}$$

   where $r \leq \min(k, n-1)$.

2. Project the $\hat{\mathbf{A}}^{(r)} \in \mathbb{R}^{k \times k}$ onto the column space of $\mathbf{U}_1^{(r)}$ and form the $r \times r$ reduced matrix

$$\tilde{\mathbf{A}} := \mathbf{U}_1^{(r)\top}\hat{\mathbf{A}}^{(r)}\mathbf{U}_1^{(r)} \tag{1.47}$$

3. Let $(\lambda_1, \ldots, \lambda_r)$ and $\boldsymbol{\Omega} := (\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_r)$ denote the eigenvalues and eigenvectors of $\tilde{\mathbf{A}}$. The eigenvalues are also the eigenvalues of $\hat{\mathbf{A}}$ [29], with the associated eigenvectors,

also known as DMD modes, being

$$\mathbf{\Phi} = \mathbf{Y}_2 \mathbf{V}_1^{(r)} (\mathbf{D}_1^{(r)})^{-1} \mathbf{\Omega}, \tag{1.48}$$

It has been shown that $(\lambda_1, \ldots, \lambda_r)$ approximate the Koopman eigenvalues, while the DMD modes $\mathbf{\Phi}$ approximates the Koopman modes [76].

The observation at any time step t_i can be approximated by DMD modes and eigenvalues:

$$\hat{\mathbf{y}}(t_i) \approx \mathbf{\Phi} \mathbf{\Lambda}^{i-1} \mathbf{a}, \tag{1.49}$$

where $\mathbf{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_r)$, and $\mathbf{a} = \mathbf{\Phi}^+ \mathbf{y}(t_1)$ represents the mode amplitudes with $\mathbf{\Phi}^+$ being the pseudo-inverse of $\mathbf{\Phi}$. An alternative way is to minimize the squared error loss such that $\hat{\mathbf{a}} = \arg\min_{\mathbf{a}} \sum_{i=1}^{n} \|\mathbf{\Lambda}^{i-1}\mathbf{a} - \mathbf{y}(t_i)\|^2$.

Since Equation (1.49) applies to any time step t_i, including $i > n$, it can be used for out-of-sample forecasting beyond the observed data. When the observations are noise-free, a more straightforward way is to let $\hat{\mathbf{y}}(t_n) = \mathbf{y}(t_n)$ and forecast the output vector on any $i > n$ by

$$\hat{\mathbf{y}}(t_i) = \hat{\mathbf{A}}^{i-n} \mathbf{y}(t_n). \tag{1.50}$$

There are two limitations of DMD. First, DMD assumes noise-free observations, so when measurement error is present in the data, model performance could degrade. Second, DMD lacks a probabilistic framework, which makes it impossible to quantify the uncertainty of the forecast. In scientific applications, uncertainty quantification is crucial for assessing the reliability of the predictions and determining whether the model produces accurate results for unseen scenarios from the system under study.

1.6 Outline

Chapter 2 is based on the material from Lin, Y., Liu, X., Segall, P., & Gu, M. (2025). Fast data inversion for high-dimensional Ornstein-Uhlenbeck Processes from noisy measurements. *SIAM/ASA Journal on Uncertainty Quantification*, to appear. Reproduced with permission of the Society for Industrial and Applied Mathematics. This chapter introduces the Fast algorithm for latent factor models with Multivariate Ornstein-Uhlenbeck (O-U) processes (FMOU), a scalable and computationally efficient algorithm for parameter estimation in high-dimensional noisy dynamical systems. In this model, we assume an orthogonal factor loading matrix, and each latent process is modeled by a distinct O-U process with its own correlation and variance parameters. We begin with an overview of latent factor models, discussing both fixed and estimated factor loading matrices as well as the associated computational bottlenecks. Then we present a motivating example: geological slip estimation from geodetic measurements using the network inversion filter (NIF). Building on this example, we then introduce a flexible latent factor model and derive a fast EM algorithm that integrates the KF and RTS smoother with orthogonal projection to avoid costly matrix inversions, yielding fast parameter updates with closed-form expressions. We further establish connections between FMOU and several related approaches, including DMD, vector autoregressive models, generalized probabilistic principal component analysis, and NIF, highlighting the computational scalability and efficiency of our approach. Finally, simulated experiments and a real-world application in estimating slip propagation in Cascadia are presented to demonstrate the speed and accuracy of our algorithm.

Chapter 3 presents three modules in the `FastGaSP` R package: `fgasp`, `gppca` and `fmou`, each providing fast computation for Gaussian processes with the Matérn kernel having half-integer roughness parameters and one-dimensional inputs. Under this set-

ting, an equivalent dynamic linear model representation exists, which enables the use of the Kalman filter and RTS smoother to speed up the computation. The `fgasp` module forms the computational foundation, handling scalar-valued output and enabling $\mathcal{O}(n)$ evaluation of the profile log-likelihood and predictive distribution via forward filtering and backward smoothing, in contrast to the $\mathcal{O}(n^3)$ cost of direct computation. Building on this, the `gppca` module handles the correlated vector-valued output through the generalized probabilistic principal component analysis (GPPCA) model, a latent factor model with an orthogonal factor loading matrix in which each latent factor is independently modeled by a GP. We provide two functions in this module to efficiently estimate the parameters and compute the predictive distribution. Finally, the `fmou` module implements the algorithm presented in Chapter 2 as a special case of GPPCA, where inputs are equally spaced, and each latent factor follows an Ornstein-Uhlenbeck process. Under this structure, the computational complexity of parameter estimation is further reduced, and all estimators have closed-form expressions. For each module, the main functions are described and illustrated with simulated or real-data examples.

Chapter 4 is based on material from: Gu, M., Lin, Y., Lee, V. C., & Qiu, D. Y. (2024). Probabilistic forecast of nonlinear dynamical systems with uncertainty quantification. *Physica D: Nonlinear Phenomena*, 457, 133938. This chapter addresses probabilistic forecasting of nonlinear dynamical systems with uncertainty quantification. We begin by reviewing popular data-driven approaches for modeling dynamical systems and identify their shared limitation: the lack of uncertainty quantification. We then extend the parallel partial Gaussian process (PP-GP) to forecast vector-valued observations by approximating the one-step-ahead transition function, with predictive uncertainty quantified through posterior sampling. We further develop a probabilistic framework for DMD that enables uncertainty quantification for forecasts. We also show that the maximum likelihood estimator of the linear mapping matrix in a DLM is equivalent to the transition

matrix in DMD. Two numerical examples are presented: one in which the inputs to the dynamics are correctly specified, and one in which the inputs are misspecified. The results demonstrate that uncertainty can be correctly quantified under correct specification, while input misspecification leads to unreliable credible intervals.

Chapter 5 summarizes the contributions of the dissertation and outlines several directions for future work. For latent factor models, future work includes extending the algorithm to handle irregular missing patterns in noisy observations, modeling each latent factor independently with a distinct Matérn 5/2 kernel, and incorporating kernel-induced structure into the factor loading matrix. For probabilistic forecasting of nonlinear dynamical systems, several extensions are worth exploring. Dimensionality reduction tools may be applied to high-dimensional data prior to fitting the surrogate model, reducing computational cost while preserving the key dynamics. Additionally, uncertainty quantification could be incorporated into transformer-based models, which are naturally suited for modeling sequential data, enabling reliable predictive intervals for forecasts.

Chapter 2

Fast Data Inversion for High-dimensional Ornstein-Uhlenbeck Processes from Noisy Measurements

This chapter is based on the material from Lin, Y., Liu, X., Segall, P., & Gu, M. (2025). Fast data inversion for high-dimensional Ornstein-Uhlenbeck Processes from noisy measurements. *SIAM/ASA Journal on Uncertainty Quantification*, to appear. Reproduced with permission of the Society for Industrial and Applied Mathematics.

In this chapter, we develop a scalable approach for a flexible latent factor model for high-dimensional dynamical systems. Each latent factor process has its own correlation and variance parameters, and the orthogonal factor loading matrix can be either fixed or estimated. We utilize an orthogonal factor loading matrix that avoids computing the inversion of the posterior covariance matrix at each time of the Kalman filter, and derive closed-form expressions in an expectation-maximization algorithm for parameter estima-

tion, which substantially reduces the computational complexity without approximation. Our approach has several applications, including noise filtering for high-dimensional time series, estimating nonseparable covariance structure between different time series, and estimating latent physical processes from real-world measurements. Extensive simulated studies illustrate higher accuracy and scalability of our approach compared to alternatives. Furthermore, by applying our method to geodetic measurements to estimate slow slip events from geodetic data in the Cascadia region, our estimated slip better agrees with independently measured seismic data of tremor events. The substantial acceleration from our method enables the use of massive noisy data for geological hazard quantification and other applications.

2.1 Introduction

Latent factor models of time-dependent systems have wide applications, such as estimating unobservable geophysical processes [2], model calibration for computer simulation of multivariate outputs [77], and inferring multiple time series in economics [21]. Though applications differ, latent factor models can be broadly classified into two categories with either fixed or data-dependent factor loading matrices.

Basis functions, such as discrete Fourier basis and Green's functions, are widely used for inverse estimation of experimental or field observations. Differential dynamic microscopy [52], for instance, is a physical approach for estimating the rheological properties of the materials by microscopy videos. The estimation corresponds to minimizing the temporal autocorrelation in the Fourier space by a latent factor model with the complex conjugate of the discrete Fourier basis [78]. Green's functions, as another example, relate field observations to unobserved displacement at different spatial scales, such as cellular force estimation by traction force microscopy [79] and geologic slip estimation by geodetic

data [57].

On the other hand, the factor loading matrix can be estimated from the data. Probabilistic models were built to understand the underlying model assumptions made by these estimations. The loading matrix estimated by the principal component analysis [61], for instance, is shown to be the maximum marginal likelihood estimator of a latent factor model with random factors independently distributed as standard Gaussian distributions [73]. Various approaches extend the independent assumption for correlated data. In [80], for instance, the authors show the estimation of the dynamic mode decomposition is equivalent to the maximum likelihood estimator of the linear mapping matrix in a vector autoregressive model of noise-free observations. As another example, each latent factor is modeled by a Gaussian process in coregionalization models [81].

In this chapter, we develop a scalable and efficient approach for latent factor models with correlated factors and an orthogonal factor loading matrix, either fixed or estimated by data. Our contributions are threefold. First, estimating a large number of correlation parameters in multivariate Gaussian processes is a fundamental challenge. When each latent factor process has distinct parameters, conventional strategies, such as posterior sampling or numerical optimization, can be prohibitively slow for estimating a large number of parameters. We developed a novel expectation-maximization (EM) algorithm for fast parameter estimation, which avoids expensive matrix inversion at each time in the Kalman filter required in the conventional EM algorithm for state space models [82], based on the orthogonal projection through the latent factor loading matrix. We surprisingly found that the estimation of the factor loading matrix, correlation, and variance parameters all have a closed-form expression in the algorithm. In particular, the correlation parameters of latent processes can be solved by cubic equations of order three. These key results are provided in Theorem 2. The log likelihood in the EM algorithm typically converges in less than 20 iterations in most of the scenarios studied

in this work, an example of which is shown in Figure 2.4. The fast convergence makes our fast approach an almost exact solution for many problems. Second, we model latent factors by a multivariate Ornstein-Uhlenbeck process that contains distinct correlation and variance parameters, which provide a flexible way to capture dependence across output coordinates. Third, we develop a new way to estimate the number of factors with either a fixed or an estimated factor loading matrix by matching the estimated noise variance with its measurement.

Our approach has advantages over alternative approaches developed in computational mathematics, statistics, and geophysics communities. In computational mathematics, dynamic mode decomposition is a popular approach that linearizes the one-step-ahead transition operator of nonlinear dynamical systems to reconstruct the dynamics by the eigenpairs of the linear mapping matrix [28, 29], which produces a finite-dimensional approximation of the Koopman modes and eigenvalues [83]. Our model extends the probabilistic model of the dynamic mode decomposition by including the noise model and utilizing a symmetric factor loading matrix to reduce the space in estimation. Significant improvements against dynamic mode decomposition will be shown for noisy observations in Section 2.4.

In the statistics community, the EM algorithm and the Kalman filter have long been used for estimating parameters in the vector autoregressive models [82, 84], and they are implemented in [1]. Our approach contains three advantages over the conventional EM algorithm for autoregressive models. First, our approach enables both the estimated factor loading matrix and dimension reduction of latent factors, while the EM algorithm typically assumes that the dimension of the latent factor is the same as the number of time series. Second, assuming models with d latent factor processes at n time points, our new algorithm only requires $\mathcal{O}(nd)$ operations in Kalman filter without approximation due to the orthogonality of the latent factor loading matrix, while the conventional EM

algorithm for vector autoregressive models requires $\mathcal{O}(nk^3)$ operations for inverting a $k \times k$ one-step-ahead predictive covariance matrix of the observations at each time step in Kalman filter. Third, our model can be set to be either stationary or non-stationary, yet the estimation from the conventional EM algorithm in [82] cannot guarantee stationarity.

In the geophysics community, the network inverse filter [2, 85] is widely used for estimating geologic slips from ground deformation measurements, such as the GPS time series. It assumes the vector autoregressive processes with the latent states following integrated Brownian motions, with shared parameters across different processes. Our method is both faster and more flexible with distinct correlation and variance parameters for each latent process, as will be shown in Section 2.4 and Section 2.5.

The rest of the paper is organized as follows. In Section 2.2, we introduce a motivating example for estimating geologic slip and introduce a flexible latent factor model for high-dimensional dynamical systems with a fast EM algorithm given in Theorem 2. We show our model substantially accelerates the computation compared to other approaches in Section 2.3. Extensive experiments in Section 2.4 show high accuracy and computational scalability of our approach. In Section 2.5, we compare our approach with existing ways to estimate the slip migration in the Cascadia region using the GPS data and demonstrate the higher detection rate of tremor events from an independent source of seismic data not used in estimation. The FMOU algorithm is coded in the `FastGaSP` 0.6.2 package available on CRAN [86]. The data and code are made publicly available in GitHub: `https://github.com/UncertaintyQuantification/FMOU/`.

2.2 Fast and efficient estimation of high-dimensional dynamical systems

2.2.1 Motivating example: Geologic slip estimation by ground deformation data

Our study is motivated by geologic slip estimation from geodetic measurements, but the new approach has broad applications for problems in science and engineering. Ground deformation observations, such as continuous Global Positioning System (GPS) observations [2] and interferometric synthetic-aperture radar interferograms [87], have been widely used to quantify geological hazards, including volcanic eruptions and earthquakes. The goal of our application is to estimate the slip that quantifies the relative movement velocities of two sides of the faults [88]. Since slip cannot be directly observed, time-dependent ground deformation information from GPS data has been used for slip estimation [89].

The Cascadia subduction zone is known to experience aseismic, transient slip events known as Slow Slip Events [90]. Slow slip events, recorded by high precision GPS data, have been found to be spatially and temporally associated with 'tectonic tremor', composed of low-frequency earthquakes recorded on seismic stations. We utilized the GPS data in the Cascadia subduction zone, publicly available at the Plate Boundary Observatory, for estimating the slip on the megathrust fault from June 2011 to August 2011 [85]. Denote the p_y-dimensional observations $\mathbf{y}(\mathbf{x}, t) \in \mathbb{R}^{p_y}$ at location $\mathbf{x} \in \mathbb{R}^{p_x}$ and time $t \in \mathbb{R}$. In our application, as the vertical displacement measurements contain little information for slip migration, the ground displacement measurement $\mathbf{y}(\mathbf{x}, t)$ is a two-dimensional vector in the East-West and North-South directions observed at equally spaced time t at a GPS station, with spatial coordinates $\mathbf{x} = (x_1, x_2)$, and hence $p_x = p_y = 2$. The

network inversion filter introduced in [2] is a popular approach for modeling the displacement at the Earth's surface by: $\mathbf{y}(\mathbf{x},t) = \int \mathbf{G}(\mathbf{x},\boldsymbol{\xi})\mathbf{z}_s(\boldsymbol{\xi},t)d\boldsymbol{\xi} + \boldsymbol{\mu}(\mathbf{x},t) + \boldsymbol{\epsilon}(t)$, where $\mathbf{G}(\mathbf{x},\boldsymbol{\xi})$ is a quasi-static elastic Green's function [91] that relates the observations to the unobservable slip $\mathbf{z}_s(\boldsymbol{\xi},t)$, and $\boldsymbol{\mu}(\mathbf{x},t)$, the local benchmark motion, captures the site-specific local motion of the GPS antenna. For data measured over a short period, such as the transient deformations from 2011 in the Cascadia subduction zone considered in this work, the impact of local motion is negligible, and hence we let $\boldsymbol{\mu}(\mathbf{x},t) = \mathbf{0}$. Furthermore, the reference frame motion is often integrated into this equation by augmented latent processes. The Gaussian noise of the measurement is denoted by $\boldsymbol{\epsilon}(t)$ and the variance of the noise is typically available *a priori* from the GPS measurements.

Suppose we collect GPS observations at $\tilde{k}$ locations, resulting in a $k = p_y\tilde{k}$ vector of observations $\mathbf{y}(t) = [\mathbf{y}(\mathbf{x}_1,t), \mathbf{y}(\mathbf{x}_2,t), \ldots, \mathbf{y}(\mathbf{x}_{\tilde{k}},t)]^\top$. In [2], the output vector is modeled as:

$$\mathbf{y}(t) = \mathbf{G}\mathbf{z}_s(t) + \boldsymbol{\epsilon}(t), \tag{2.1}$$

where $\mathbf{G}$ is a $k \times k'$ matrix of discretized Green's function, $\mathbf{z}_s(t) = [z_s(\boldsymbol{\xi}_1,t), \ldots, z_s(\boldsymbol{\xi}_{k'},t)]^\top$ is a k'-vector of unobservable geologic slip, with subscript 's' meaning the slip, and the Gaussian noises follow $\boldsymbol{\epsilon}(t) \sim \mathcal{MN}(\mathbf{0}, \sigma_0^2\mathbf{I}_k)$ with variance σ_0^2. The (I,H) block of $\mathbf{G}$ is $\mathbf{G}_{I,H} = \mathbf{G}(\mathbf{x}_I, \boldsymbol{\xi}_H)\Delta$ and Δ is the area size in discretization, for $I = 1, ..., \tilde{k}$ and $H = 1, ..., k'$ with $\tilde{k} = 100$ GPS stations and $k' = 1978$ discretization points used in estimating the slip propagation in Cascadia [85]. We found that increasing the number of discretization points has almost no impact on estimates of the slip.

Let $\mathbf{U}_0$ be $k \times d$ left singular vector matrix corresponding to the largest d singular values in the singular value decomposition (SVD) of the Green's function $\mathbf{G} \approx \mathbf{U}_0\mathbf{D}_0\mathbf{V}_0^\top$, where $\mathbf{D}_0$ is a $d \times d$ diagonal matrix and $\mathbf{V}_0$ is a $k' \times d$ matrix of d right singular vectors with $d \leq k$. As the number of discretization points k' is larger than k, the latent slip

vector is modeled by a linear combination of the latent processes [2] to obtain a reduced order representation:

$$\mathbf{z}_s(t) = \mathbf{G}^\top \mathbf{U}_0 \tilde{\mathbf{z}}(t), \tag{2.2}$$

with $\tilde{\mathbf{z}}(t) = [\tilde{z}_1(t), \ldots, \tilde{z}_d(t)]^\top$ being a d-vector of latent processes. The latent process $\tilde{z}_l(t)$ is assumed to follow an integrated Brownian motion in [2], for $l = 1, \ldots, d$. The Kalman filter is used for computing the posterior distribution of the latent process $\tilde{\mathbf{z}}(\cdot)$ and slip estimation. There are several restrictions in this approach. First, the latent process $\tilde{z}_l(\cdot)$ has the same covariance across $l = 1, ..., d$, yet the random latent factors can have different correlation length scales. Second, one needs to invert a $k \times k$ covariance matrix at each time in the Kalman filter, which is prohibitively slow when the number of GPS stations is moderately large. Third, selecting the number of latent factors is not discussed in this approach.

The goal of this chapter is to develop a generally applicable approach and scalable algorithm for latent models for estimating the mean of the data and the posterior distribution of the latent variables. For estimating the slip rates in Cascadia between June and August in 2011, we found that the estimated slips had higher spatial correlation with independently detected tremor events not used in estimation. Furthermore, our method is $500 - 1600$ times faster than the network inversion filter in [2] in our real application. The high scalability of our method enables abundant data from GPS networks to be jointly used for hazard quantification.

2.2.2 A multivariate Ornstein-Uhlenbeck process with an orthogonal basis matrix

We follow the assumption in (2.2), which indicates that the mean in (2.1) is approximated by $\mathbf{G}\mathbf{z}_s(t) \approx \mathbf{U}_0\mathbf{z}(t)$, where $\mathbf{z}(t) = \mathbf{D}_0^2\tilde{\mathbf{z}}(t)$ with the lth entry being $z_l(t)$, and

there is no approximation when $k = d$. The d-dimensional latent factor processes $\mathbf{z}(\cdot)$ are modeled by independent Ornstein-Uhlenbeck (OU) processes, where each process has its own correlation and variance parameters. The OU processes correspond to autoregressive models of order 1, widely used for modeling time series. Furthermore, the proposed model is closely related to the dynamic mode decomposition, a popular dimension reduction tool to approximate dynamical systems [28], as will be compared in Section 2.3.1. Together, Equations (2.1)-(2.2) lead to

$$\mathbf{y}(t) = \mathbf{U}_0\mathbf{z}(t) + \boldsymbol{\epsilon}(t), \tag{2.3}$$

$$z_l(t) = \rho_l z_l(t-1) + w_l(t), \tag{2.4}$$

where $\mathbf{U}_0$ is a $k \times d$ orthogonal factor loading matrix, either fixed or estimated, which relates the d-dimensional latent factors to k-dimensional measurements with $d \leq k$. Here, the parameter ρ_l controls the temporal correlation length of the lth latent process, and $w_l(t) \sim \mathcal{N}(0, \sigma_l^2)$ is the innovation of the lth latent process for $t = 2, ..., n$, with the initial state being $z_l(1) \sim \mathcal{N}(0, \tau_l^2)$ for $l = 1, \ldots, d$. The d sets of parameters $\boldsymbol{\rho} = (\rho_1, \rho_2, \ldots, \rho_d)$ and $\boldsymbol{\sigma}^2 = (\sigma_1^2, \sigma_2^2, \ldots, \sigma_d^2)$ make the model flexible. We postpone the discussion of estimating d in Section 2.2.4.

We highlight that the orthogonal factor loading matrix $\mathbf{U}_0$ in Equation (2.3) and OU processes in Equation (2.4) bring substantial computational advantages, as will be elaborated soon. The assumption of orthogonality of $\mathbf{U}_0$ originates from several previous approaches. First, the matrix $\mathbf{G}$ in Equation (2.1) is typically neither symmetric nor orthogonal, but with the assumption in Equation (2.2) used in [2], the model in (2.1) reduces to the Model (2.3) with an orthogonal $\mathbf{U}_0$ matrix. Second, due to the non-identifiability between factor loadings and factors, the orthogonal assumption of factor loadings was also used in the past [62]. In our experience, when the dimension k is

large, the orthogonal assumption typically works well, as the orthogonal matrix $\mathbf{U}_0$ has a large number of parameters that can be estimated, making the model flexible to capture dependence across different factor processes. In this work, we develop novel algorithms for both fixed and estimated $\mathbf{U}_0$, which are broadly applicable to many applications. Furthermore, we assume the mean of the factor processes to be zero for simplicity, though additional mean structures can be included in the model.

Lemma 4 connects the model in (2.3)-(2.4) to the continuous-time multivariate OU process [92, 93] with an orthogonal basis matrix. The derivations of all lemmas in this subsection are provided in Section B.1 in the appendix.

Lemma 4 *Denote the mean of the k-dimensional processes in (2.3) by* $\mathbf{m}(t) = \mathbf{U}_0\mathbf{z}(t)$. *When* $\rho_l \in (0,1)$ *in Equation (2.4) for* $l = 1, \ldots, d$, *the mean process* $\mathbf{m}(t)$ *is a discretized process of the continuous-time multivariate Ornstein-Uhlenbeck process, defined by the stochastic differential equation:*

$$d\mathbf{m}(t) = -\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top\mathbf{m}(t)dt + \mathbf{U}_0\tilde{\mathbf{D}}d\mathbf{B}_t, \tag{2.5}$$

where $\mathbf{D} = diag(-\log(\rho_1), \ldots, -\log(\rho_d))$, $\tilde{\mathbf{D}}$ *is a diagonal matrix with the* l*th element being* $\sqrt{\frac{-2\sigma_l^2\log(\rho_l)}{1-\rho_l^2}}$ *for* $l = 1, \ldots, d$, *and* $\mathbf{B}_t$ *is a vector of independent Brownian motions.*

We study the model in (2.3)-(2.4) because of its flexibility due to distinct parameters (ρ_l, σ_l^2) for each latent factor l, for $l = 1, ..., d$ and the computational scalability when the number of latent factors, d, is large. The latent process $z_l(\cdot)$ can be either stationary or nonstationary, depending on the assumption of the initial state, stated in the Lemma 5.

Lemma 5 *For* $l = 1, ..., d$, *the covariance of the process* $z_l(\cdot)$ *in Equation (2.4) follows*

$$Cov[z_l(t), z_l(t')] = \rho_l^{t'-t}\left\{\rho_l^{2(t-1)}\tau_l^2 + \frac{1-\rho_l^{2(t-1)}}{(1-\rho_l^2)}\sigma_l^2\right\}, \quad \text{for } t' \geq t. \tag{2.6}$$

In particular, when the variance of initial state is $\tau_l^2 = \frac{\sigma_l^2}{1-\rho_l^2}$, $z_l(\cdot)$ *is stationary with*

$$Cov[z_l(t), z_l(t')] = \frac{\sigma_l^2}{1-\rho_l^2}\rho_l^{|t-t'|}. \tag{2.7}$$

When $\tau_l^2 = \frac{\sigma_l^2}{1-\rho_l^2}$, the latent process $z_l(\cdot)$ is constrained to be stationary in Lemma 5, and when τ_l^2 is estimated as a separate parameter in the EM algorithm, the latent factor process $z_l(\cdot)$ may not be stationary. In the rest of the chapter, we assume that each latent process is stationary by letting $\tau_l^2 = \frac{\sigma_l^2}{1-\rho_l^2}$. Lemma 6 outlines key properties of the covariance and precision matrices, which enable us to modify Kalman filter (KF) [25] and Rauch–Tung–Striebel (RTS) smoother [26] to scalably compute expensive quantities in our fast algorithm.

Lemma 6 *Assume* $\tau_l^2 = \frac{\sigma_l^2}{1-\rho_l^2}$ *and denote* $\mathbf{\Sigma}_l = \frac{\sigma_l^2}{1-\rho_l^2}\mathbf{R}_l$ *as the covariance matrix of* $\mathbf{z}_l = (z_l(1), \ldots, z_l(n))^\top$ *defined in Equation (2.4). The* (t, t') *entry of* $\mathbf{R}_l$ *is* $\rho_l^{|t-t'|}$, *for* $t, t' = 1, 2, \ldots, n$. *The inverse of* $\mathbf{R}_l$ *has a tri-diagonal structure, with diagonal entries of* $\frac{1}{1-\rho_l^2}$ *at the first and last positions, and* $\frac{1+\rho_l^2}{1-\rho_l^2}$ *at the remaining positions. The primary off-diagonal entries are* $-\frac{\rho_l}{1-\rho_l^2}$. *Additionally, the determinant of* $\mathbf{\Sigma}_l$ *follows* $|\mathbf{\Sigma}_l| = \frac{\sigma_l^{2n}}{1-\rho_l^2}$.

2.2.3 A fast EM algorithm with closed-form expressions

In this section, we derive a fast EM algorithm, named a Fast algorithm of Multivariate Ornstein-Uhlenbeck processes (FMOU), for a general scenario where the parameters are $\mathbf{\Theta} = (\mathbf{U}_0, \sigma_0^2, \boldsymbol{\rho}, \boldsymbol{\sigma}^2)$ with $\boldsymbol{\rho} = (\rho_1, \rho_2, \ldots, \rho_d)$ and $\boldsymbol{\sigma}^2 = (\sigma_1^2, \sigma_2^2, \ldots, \sigma_d^2)$. The algorithm leverages the KF and RTS smoother for computing required quantities with orthogonal projection, to bypass costly matrix inversion, conventionally required in each step of the KF. The proofs in this subsection are given in Appendix B.2.

Let $\mathbf{Y} = [\mathbf{y}(1), \ldots, \mathbf{y}(n)]$ represents an observation matrix and $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_d]^\top$

denotes a latent factor matrix. After integrating out the latent factors, the parameters $\boldsymbol{\Theta}$ are estimated by the maximum marginal likelihood estimator (MMLE):

$$\boldsymbol{\Theta}^{\text{MMLE}} = \arg\max_{\boldsymbol{\Theta}} \int p(\mathbf{Y} \mid \mathbf{Z}, \boldsymbol{\Theta}) p(\mathbf{Z} \mid \boldsymbol{\Theta}) d\mathbf{Z}. \tag{2.8}$$

Direct optimization of the marginal likelihood of the model in Equations (2.3) and (2.4) can be unstable due to optimizing the latent factor matrix in the Stiefel manifold in each step of the optimization [94] and numerical optimization in high-dimensional parameter space. We develop an EM algorithm that has closed-form expressions in each iteration.

First, we show in Appendix B.2.1 that the natural logarithm of the joint likelihood of $(\mathbf{Y}, \mathbf{Z} \mid \boldsymbol{\Theta})$ follows

$$\ell(\boldsymbol{\Theta}) = C - \frac{nk}{2}\log(\sigma_0^2) - \frac{\operatorname{tr}(\mathbf{Y}^\top\mathbf{Y} - 2\mathbf{Y}^\top\mathbf{U}_0\mathbf{Z})}{2\sigma_0^2} - \sum_{l=1}^{d}\Big(\frac{\mathbf{z}_l^\top\mathbf{z}_l}{2\sigma_0^2} + \frac{\log|\boldsymbol{\Sigma}_l| + \mathbf{z}_l^\top\boldsymbol{\Sigma}_l^{-1}\mathbf{z}_l}{2}\Big), \tag{2.9}$$

where $C = -\frac{(nk+nd)}{2}\log(2\pi)$ is a constant.

In the E step, we calculate the expectation of the joint log-likelihood function with respect to the distribution of $\mathbf{Z}$ conditional on observations $\mathbf{Y}$, and the current estimate of parameters, denoted as $\hat{\boldsymbol{\Theta}} = (\hat{\mathbf{U}}_0, \hat{\sigma}_0^2, \hat{\boldsymbol{\rho}}, \hat{\boldsymbol{\sigma}}^2)$. Our model in (2.3)-(2.4) is a specific case of the model in [19] with the latent factor processes assumed to be OU processes, and this specification enables our model to be computed much faster, as will be compared in Section 2.3.3. The posterior distribution of factors given $\hat{\boldsymbol{\Theta}}$ follows a multivariate Gaussian distribution shown in [19]:

$$(\mathbf{z}_l \mid \mathbf{Y}, \hat{\boldsymbol{\Theta}}) \sim \mathcal{MN}\left(\hat{\mathbf{z}}_l, \hat{\sigma}_0^2\hat{\boldsymbol{\Sigma}}_l(\hat{\boldsymbol{\Sigma}}_l + \hat{\sigma}_0^2\mathbf{I}_n)^{-1}\right), \tag{2.10}$$

where $\hat{\mathbf{z}}_l = \hat{\mathbf{\Sigma}}_l(\hat{\mathbf{\Sigma}}_l + \hat{\sigma}_0^2\mathbf{I}_n)^{-1}\tilde{\mathbf{y}}_l$ with $\tilde{\mathbf{Y}} = \mathbf{U}_0^\top\mathbf{Y} = [\tilde{\mathbf{y}}_1, \cdots, \tilde{\mathbf{y}}_d]^\top$ being the projected observation matrix, and the matrix $\hat{\mathbf{\Sigma}}_l$ is formed by the same kernel function as $\mathbf{\Sigma}_l$, with the current estimate of the parameters σ_l^2 and ρ_l plugged in, for $l = 1, \ldots, d$. Denote $\hat{\mathbf{Z}} = [\hat{\mathbf{z}}_1, \ldots, \hat{\mathbf{z}}_d]^\top$, a $d \times n$ matrix of the posterior mean of latent factors and $\bar{\ell}(\mathbf{\Theta}) = \mathbb{E}_{\mathbf{Z}|\mathbf{Y},\hat{\mathbf{\Theta}}}[\ell(\mathbf{\Theta})]$. Utilizing Equation (2.10), we show in Appendix B.2.2 the E step follows

$$\bar{\ell}(\mathbf{\Theta}) = C + \frac{\mathrm{tr}(\mathbf{Y}^\top\mathbf{U}_0\hat{\mathbf{Z}})}{\sigma_0^2} - \sum_{l=1}^{d}\left\{\frac{\log|\mathbf{\Sigma}_l|}{2} + \frac{\hat{\mathbf{z}}_l^\top\hat{\mathbf{z}}_l + \mathrm{tr}[\hat{\sigma}_0^2\hat{\mathbf{\Sigma}}_l(\hat{\mathbf{\Sigma}}_l + \hat{\sigma}_0^2\mathbf{I}_n)^{-1}]}{2\sigma_0^2}\right\}$$
$$- \sum_{l=1}^{d}\frac{\hat{\mathbf{z}}_l^\top\mathbf{\Sigma}_l^{-1}\hat{\mathbf{z}}_l + \mathrm{tr}[\hat{\sigma}_0^2\mathbf{\Sigma}_l^{-1}\hat{\mathbf{\Sigma}}_l(\hat{\mathbf{\Sigma}}_l + \hat{\sigma}_0^2\mathbf{I}_n)^{-1}]}{2} - \frac{nk}{2}\log(\sigma_0^2) - \frac{\mathrm{tr}(\mathbf{Y}^\top\mathbf{Y})}{2\sigma_0^2}, \quad (2.11)$$

where $\mathbf{\Sigma}_l$ depends on unknown $(\boldsymbol{\rho}, \boldsymbol{\sigma}^2)$ to be optimized, and $\hat{\mathbf{\Sigma}}_l$ is obtained based on the current estimate of the parameters $(\hat{\boldsymbol{\rho}}, \hat{\boldsymbol{\sigma}}^2)$.

Directly computing the conditional distribution in Equation (2.10) requires $\mathcal{O}(dn^3)$ operations due to inverting d matrices of size $n \times n$, which can be computationally expensive. Instead, by treating the projected observations $\tilde{\mathbf{y}}_l = (\tilde{y}_l(1), \ldots, \tilde{y}_l(n))^\top$ as the noisy observations of the lth latent process, we can apply KF and RTS smoother independently to each latent factor. This approach efficiently computes the required quantities with $\mathcal{O}(dn)$ operations, avoiding the need for matrix inversion. Lemma 7 demonstrates their roles in simplifying computations in the E step. The proof can be found in Appendix B.2.3.

Lemma 7 *Consider a dynamic linear model $\tilde{y}_l(t) = z_l(t) + \epsilon$ with latent factor process defined in (2.4) and ϵ being an independent noise with variance σ_0^2, for $l = 1, ..., d$. Denote $s_l(t) = \mathbb{E}[z_l(t) \mid \tilde{\mathbf{y}}_l, \hat{\mathbf{\Theta}}]$, $S_l(t) = \mathbb{V}[z_l(t) \mid \tilde{\mathbf{y}}_l, \hat{\mathbf{\Theta}}]$ and $\tilde{S}_l(t) = Cov[z_l(t), z_l(t+1) \mid \tilde{\mathbf{y}}_l, \hat{\mathbf{\Theta}}]$,*

which are computed by the KF and RTS smoother. We have

$$tr[\hat{\sigma}_0^2\hat{\mathbf{\Sigma}}_l(\hat{\mathbf{\Sigma}}_l+\hat{\sigma}_0^2\mathbf{I}_n)^{-1}]=\sum_{t=1}^{n}\mathbb{V}[z_l(t)\mid\mathbf{Y},\hat{\mathbf{\Theta}}]=\sum_{t=1}^{n}S_l(t), \tag{2.12}$$

$$\hat{\mathbf{z}}_l^\top\mathbf{\Sigma}_l^{-1}\hat{\mathbf{z}}_l=\frac{1}{\sigma_l^2}\left((1-\rho_l^2)s_l^2(1)+\sum_{t=2}^{n}(s_l(t)-\rho_l s_l(t-1))^2\right), \tag{2.13}$$

$$tr[\hat{\sigma}_0^2\mathbf{\Sigma}_l^{-1}\hat{\mathbf{\Sigma}}_l(\hat{\mathbf{\Sigma}}_l+\hat{\sigma}_0^2\mathbf{I}_n)^{-1}]=\frac{1}{\sigma_l^2}\left(\sum_{t=1}^{n}S_l(t)+\rho_l^2\sum_{t=2}^{n-1}S_l(t)-2\rho_l\sum_{t=1}^{n-1}\tilde{S}_l(t)\right). \tag{2.14}$$

In the M step, the parameters $\hat{\mathbf{\Theta}}^{\text{new}}=(\hat{\mathbf{U}}_0^{\text{new}},(\hat{\sigma}_0^2)^{\text{new}},\hat{\boldsymbol{\rho}}^{\text{new}},(\hat{\boldsymbol{\sigma}}^2)^{\text{new}})$ are obtained by

$$\hat{\mathbf{\Theta}}^{\text{new}}=\text{argmax}_{\mathbf{\Theta}}\bar{\ell}(\mathbf{\Theta}). \tag{2.15}$$

In Theorem 2, we show that all parameters have closed-form expressions in the M-step in (2.15), thus avoiding numerically optimizing high-dimensional parameters in the algorithm. The proof can be found in Appendix B.2.4. Since the M-step is solved exactly via these closed-form expressions, the log-likelihood function is guaranteed to be nondecreasing at each iteration, and the algorithm converges to a stationary point [71].

Theorem 2 *Given the parameters estimated from the last iteration, one can compute* $\{s_l(t)\}_{t=1}^n,\{S_l(t)\}_{t=1}^n$ *and* $\{\tilde{S}_l(t)\}_{t=1}^{n-1}$ *by the KF and RTS smoother, for* $l=1,\ldots,d$*. The closed-form expressions of* $\hat{\mathbf{\Theta}}^{new}$ *from the M step, as defined in (2.15), are given below.*

1. *Update* $\hat{\mathbf{U}}_0^{new}$ *by*

$$\hat{\mathbf{U}}_0^{new}=\tilde{\mathbf{V}}\tilde{\mathbf{U}}^\top, \tag{2.16}$$

where $\tilde{\mathbf{U}}$ *and* $\tilde{\mathbf{V}}$ *being the left and right singular vectors of the* $\hat{\mathbf{Z}}\mathbf{Y}^\top$*, respectively.*

2. *Update* $(\hat{\sigma}_0^2)^{new}$ *by*

$$(\hat{\sigma}_0^2)^{new} = \frac{tr(\mathbf{Y}^\top \mathbf{Y}) - 2tr(\mathbf{Y}^\top \hat{\mathbf{U}}_0^{new} \hat{\mathbf{Z}}) + \sum_{l=1}^{d} \sum_{t=1}^{n} (s_l^2(t) + S_l(t))}{nk}. \tag{2.17}$$

3. *Update* $\hat{\rho}_l^{new}$ *by solving the following cubic equation in the interval* $(-1, 1)$ *which has a unique root in* $(-1, 1)$ *if* $\hat{\sigma}_0^2 > 0$ *and* $\hat{\sigma}_l^2 > 0$*:*

$$\beta_0 + \beta_1 \rho_l + \beta_2 \rho_l^2 + \beta_3 \rho_l^3 = 0, \tag{2.18}$$

where $\beta_0 = n\Big(\sum_{t=2}^{n} s_l(t-1)s_l(t) + \sum_{t=1}^{n-1} \tilde{S}_l(t)\Big)$, $\beta_1 = -\sum_{t=1}^{n} s_l^2(t) - \sum_{t=1}^{n} S_l(t) - n\sum_{t=2}^{n-1} s_l^2(t) - n\sum_{t=2}^{n-1} S_l(t)$, $\beta_2 = (2-n)(\sum_{t=2}^{n} s_l(t-1)s_l(t) + \sum_{t=1}^{n-1} \tilde{S}_l(t))$, $\beta_3 = (n-1)(\sum_{t=2}^{n-1} s_l^2(t) + \sum_{t=2}^{n-1} S_l(t))$.

4. *Update* $(\hat{\sigma}_l^2)^{new}$ *by*

$$(\hat{\sigma}_l^2)^{new} = \frac{1}{n}\left\{(1 - (\hat{\rho}_l^{new})^2)s_l^2(1) + \sum_{t=2}^{n}(s_l(t) - \hat{\rho}_l^{new} s_l(t-1))^2\right\} + \frac{1}{n}\left\{\sum_{t=1}^{n} S_l(t) + (\hat{\rho}_l^{new})^2 \sum_{t=2}^{n-1} S_l(t) - 2\hat{\rho}_l^{new} \sum_{t=1}^{n-1} \tilde{S}_l(t)\right\}. \tag{2.19}$$

5. *Update* $\{s_l^{new}(t)\}_{t=1}^n, \{S_l^{new}(t)\}_{t=1}^n$ *and* $\{\tilde{S}_l^{new}(t)\}_{t=1}^{n-1}$ *by KF and RTS smoother using* $\hat{\mathbf{\Theta}}^{new}$*. Set* $\hat{\mathbf{z}}_l^{new} = [s_l^{new}(1), \ldots, s_l^{new}(n)]^\top$ *and* $\hat{\mathbf{Z}}^{new} = [\hat{\mathbf{z}}_1^{new}, \ldots, \hat{\mathbf{z}}_d^{new}]^\top$.

The FMOU in Theorem 2 is nontrivial to derive, and it differs from the conventional EM algorithm for vector autoregressive (VAR) models [82]. First, FMOU avoids the costly matrix inversion required in each step in the KF due to the use of an orthogonal latent factor loading matrix that can be estimated from data. Second, it enables dimension reduction. Third, according to Lemma 5, the FMOU algorithm can be set to be stationary or nonstationary, while the conventional EM for VAR models cannot

guarantee stationarity in estimation.

Algorithm 1 summarizes the proposed FMOU approach. The output of Algorithm 1 is the MMLE of the parameters $\mathbf{\Theta}^{\text{MMLE}}$ in (2.8) and posterior mean of the latent factors given $\mathbf{\Theta}^{\text{MMLE}}$, denoted as $\mathbf{Z}^{\text{post}} = [\mathbf{z}^{\text{post}}(1), ..., \mathbf{z}^{\text{post}}(n)] = \mathbb{E}[\mathbf{Z} \mid \mathbf{Y}, \mathbf{\Theta}^{\text{MMLE}}]$. In our application of slip estimation, other quantities, such as the posterior distribution of the mean of the observations and slips at time t, follow

$$\left(\mathbf{m}(t) \mid \mathbf{Y}, \mathbf{\Theta}^{\text{MMLE}}\right) \sim \mathcal{MN}\left(\mathbf{U}_0 \mathbf{z}^{\text{post}}(t), \mathbf{\Sigma}_m^{\text{MMLE}}(t)\right), \tag{2.20}$$

$$\left(\mathbf{z}_s(t) \mid \mathbf{Y}, \mathbf{\Theta}^{\text{MMLE}}\right) \sim \mathcal{MN}\left(\mathbf{G}^\top \mathbf{U}_0 \mathbf{D}_0^{-2} \mathbf{z}^{\text{post}}(t), \mathbf{\Sigma}_s^{\text{MMLE}}(t)\right), \tag{2.21}$$

where $\mathbf{\Sigma}_m^{\text{MMLE}}(t) = \mathbf{U}_0 \mathbf{D}_S(t) \mathbf{U}_0^\top$ and $\mathbf{\Sigma}_s^{\text{MMLE}}(t) = \mathbf{G}^\top \mathbf{U}_0 \mathbf{D}_0^{-2} \mathbf{D}_S(t) \mathbf{D}_0^{-2} \mathbf{U}_0^\top \mathbf{G}$, with $\mathbf{D}_S(t)$ being a $d \times d$ diagonal matrix where the lth element is $S_l(t)$ computed by plugging in the MMLE of the parameters for $l = 1, 2, \ldots, d$. Here $\mathbf{U}_0$ can be either fixed or estimated. In the application of slip estimation, $\mathbf{U}_0$ and $\mathbf{D}_0$ are the first d left singular vectors and the largest d singular values of the matrix of Green's function $\mathbf{G}$, respectively, and the variance of the noise is usually available from GPS measurements, with details discussed in Appendix B.2.5. Furthermore, in our default setting the initialization of $\mathbf{U}_0$ and σ_0^2 are initialized as the first d left singular vectors of the output matrix and $\log(1.5)$, respectively, as they do not have a large impact on estimating the mean due to the flexibility of the models with a large number of parameters. Other initialization ways can be used. For instance, the initialization of $\mathbf{U}_0$ can be sampled from the Stiefel manifold or specified as a $k \times d$ matrix where the diagonal elements of the first d columns are 1, and 0 otherwise, with $d \leq k$.

Several advantages of this EM algorithm are worth mentioning. First, the kernel parameters in Gaussian processes are notoriously hard to estimate numerically, and we have d sets of kernel and scale parameters, where d can be on the order of 10^3 or even 10^6 in

Algorithm 1 Fast EM algorithm for multivariate Ornstein-Uhlenbeck process

Input: $k \times n$ observation matrix $\mathbf{Y}$, time step $1, 2, \ldots, n$ and selected number of latent processes d. The representer $\mathbf{G}$ and noise level σ_0^2 should be provided if they are fixed.

1: (Optional) Initialize $\mathbf{U}_0$ and σ_0^2 if they are unspecified.
2: Initialize $(\hat{\boldsymbol{\rho}}, \hat{\boldsymbol{\sigma}}^2)$ and then apply KF and RTS smoother to get $\{s_l(t)\}_{t=1}^n, \{S_l(t)\}_{t=1}^n$ and $\{\tilde{S}_l(t)\}_{t=1}^{n-1}$ for $l = 1, \ldots, d$.
3: **while** The convergence criteria were not met and the number of iterations is less than the upper bound **do**
4: (Optional.) If $\mathbf{U}_0$ is unknown, update $\hat{\mathbf{U}}_0$ using Equation (2.16).
5: (Optional.) If σ_0^2 is unknown, update $\hat{\sigma}_0^2$ using Equation (2.17).
6: Update $\hat{\rho}_l$ in the correlation matrix $\mathbf{R}_l$ for $l = 1, \ldots, d$ using Equation (2.18).
7: Update $\hat{\sigma}_l^2$ for $l = 1, \ldots, d$ using Equation (2.19).
8: Update $\{s_l(t)\}_{t=1}^n, \{S_l(t)\}_{t=1}^n$, $\{\tilde{S}_l(t)\}_{t=1}^{n-1}$ and $\hat{\mathbf{Z}}$ by KF and RTS smoother.
9: **end while**

Output: $\boldsymbol{\Theta}^{\mathrm{MMLE}}$ and $\mathbf{Z}^{\mathrm{post}}$.

some applications. The estimates of these parameters have closed-form expressions in our EM algorithm, which had not been derived before. Our algorithm substantially improves estimation stability compared to optimization algorithms that numerically optimize in a high-dimensional parameter space. Second, when $\mathbf{U}_0$ is unknown and the parameters are distinct in each latent process, originally one needs to solve a constrained optimization problem in the Stiefel manifold [94] to optimize $\mathbf{U}_0$ in each of the numerical iteration for estimating the parameters of the model in (2.3) and (2.4) [19]. In contrast, we derived closed-form expressions for estimating $\mathbf{U}_0$ in each iteration of the EM algorithm, and hence the algorithm is much faster. Third, we integrate orthogonal projection in the KF and RTS smoother to bypass the matrix inversion operation at each time step. For M iterations of the EM algorithms, the overall complexity of the parameter estimation process is $\mathcal{O}(Mknd) + \mathcal{O}(Mkd^2)$ when the coefficient $\mathbf{U}_0$ is unknown, as shown in Table 2.1. For the scenario where $\mathbf{U}_0$ is derived from the Green's function, we need to perform SVD to obtain $\mathbf{U}_0$ only once, which has the order $\mathcal{O}(min(k^2k', k(k')^2))$.

2.2.4 Estimation of the number of latent processes

Estimating a suitable number of factors is crucial to distinguish signals from noise. Various approaches have been developed based on, for instance, information criteria [95], the cumulative percentage of variance explained by the factors [96], and the ratio of neighboring eigenvalues of the empirical autocovariance matrix [62]. In our scientific application, the goal is not to select the number of factors, but to provide a suitable model for linking the complex slip propagation process to ground deformation. To enable the variability in the signal to be properly explained by our model, we select the number of latent factors, d, by minimizing the difference between the estimated and measured noise variance:

$$\hat{d} = \arg\min_{d} |(\sigma_0^{\mathrm{MMLE}}(d))^2 - \sigma_0^2|, \tag{2.22}$$

where σ_0^2 is the measured variance of the noise from GPS, and $(\sigma_0^{\mathrm{MMLE}}(d))^2$ is the estimated variance of the noise with d latent factors. We call this approach variance matching (VM).

When the variance of the noise is unknown, we follow [95] to use the information criterion (IC) in Equation (2.23) to select the number of factors, which has the form below:

$$\hat{d} = \arg\min_{d} \left\{ \log(||\mathbf{Y} - \tilde{\mathbf{U}}_{1:d}\tilde{\mathbf{U}}_{1:d}^{\top}\mathbf{Y}||_F) + C(k,n) \right\}, \tag{2.23}$$

where $||\cdot||_F$ denotes the Frobenius matrix norm, $\tilde{\mathbf{U}}_{1:d}$ is the first d columns from the left unitary matrix of the singular value decomposition of $\mathbf{Y}$ and $C(k,n)$ is a penalty term, such as $C(k,n) = d\left(\frac{k+n}{kn}\right)\log(\frac{kn}{k+n})$. This approach will be compared in Section 2.4.

methods	computational order	noise models	explicit estimators of $\mathbf{U}_0$
FMOU	$\mathcal{O}(Mknd + Mkd^2)$	Yes	Yes
DMD	$\min(\mathcal{O}(kn^2), \mathcal{O}(k^2 n))$	No	Yes
EM-VAR(1)	$\mathcal{O}(M^* k^3 n)$	Yes	/
GPPCA	$\mathcal{O}(M_1 M_2 knd + M_1 M_2 kd^2)$	Yes	No
NIF	$\mathcal{O}(\tilde{M} k^3 n)$	Yes	/

Table 2.1: Comparison of different approaches by the computational order, whether a noise model is included, and whether a closed-form expression in estimating the factor loading matrix is available when the factor processes have distinct correlation parameters. For FMOU, we assume M iterations in EM, where $M \approx 10$s often achieve good performance. For EM-VAR(1), M^* is the number of E-M iterations, and the order is when $d = k$, which is the default setting in [1]. For DMD, we only consider the cost of obtaining the first d eigenvectors, whereas obtaining the transition coefficient matrix requires extra $O(k^2 d)$ operations. When each factor contains distinct correlation parameters, GPPCA requires M_1 iterations and each requires M_2 steps to optimize the factor loading matrix in the Stiefel manifold. To estimate the slip, an additional SVD operation of a $k \times k'$ matrix $\mathbf{G}$ is needed, and it only needs to be done once. Thus, the order of FMOU is $\mathcal{O}(min(k^2 k', k(k')^2)) + \mathcal{O}(Mknd)$ and the order of NIF is $\mathcal{O}(min(k^2 k', k(k')^2)) + \mathcal{O}(\tilde{M} k^3 n)$ with $\tilde{M}$ being the number of iterations in NIF for slip estimation.

2.3 Improved computational scalability and efficiency

We discuss the connection to other approaches, including the dynamic mode decomposition [28], the conventional EM algorithm for vector autoregressive models [82], and the generalized probabilistic principal component analysis [19] in Sections 2.3.1-2.3.3, respectively. We also compare the network inversion filter (NIF) [2] and its modification [85] for slip estimation in Appendix B.3. The computational order of different approaches is provided in Table 2.1.

2.3.1 The FMOU approach extends the dynamic mode decomposition with a symmetric transition matrix for noisy data

The dynamic mode decomposition (DMD) is a popular approach in computational mathematics to obtain the reduced-rank representation of nonlinear dynamical systems [28, 29], which is reviewed in Chapter 1.5.

However, the DMD assumes noise-free observation, which is restrictive in practice. The FMOU model includes an additional level of noise modeling for DMD with a symmetric transition matrix. To see this, for the FMOU model in (2.3) and (2.4), denoting $\bar{\mathbf{z}}(t) = \mathbf{U}_0\mathbf{z}(t)$, the model can be equivalently written as

$$\mathbf{y}(t) = \bar{\mathbf{z}}(t) + \boldsymbol{\epsilon}(t), \tag{2.24}$$

$$\bar{\mathbf{z}}(t) = \mathbf{A}\bar{\mathbf{z}}(t-1) + \bar{\mathbf{w}}(t), \tag{2.25}$$

where $\mathbf{A} = \mathbf{U}_0\boldsymbol{\Lambda}_\rho\mathbf{U}_0^\top$ with $\mathbf{U}_0$ being an $k \times d$ orthogonal matrix and $\boldsymbol{\Lambda}_\rho$ being an $d \times d$ diagonal matrix having the lth diagonal entry ρ_l, and $\bar{\mathbf{w}}(t) \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\bar{\mathbf{w}}})$ with $\boldsymbol{\Sigma}_{\bar{\mathbf{w}}} = \mathbf{U}_0\mathbf{D}_\sigma\mathbf{U}_0^\top$ being a $k \times k$ matrix, and $\mathbf{D}_\sigma = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_d^2)$. By multiplying $\mathbf{U}_0^\top$ on both sides of Equations (2.24)-(2.25), we obtain the FMOU model.

Thus, the FMOU approach provides a fast solution of a noise-inclusive, probabilistic DMD model with a symmetric transition matrix. This extension drastically improves the DMD when the data contain noise, as will be numerically demonstrated in Section 2.4. The computational order of DMD and FMOU is summarized in Table 2.1, and both are fast in practice. When both k and n are large, the FMOU can be faster than the DMD.

2.3.2 The FMOU approach accelerates the EM algorithm for vector autoregressive models and enables dimension reduction

The EM algorithm was developed for vector autoregressive models [82]:

$$\mathbf{y}(t) = \mathbf{G}(t)\mathbf{z}(t) + \boldsymbol{\epsilon}(t), \tag{2.26}$$

$$\mathbf{z}(t) = \boldsymbol{\Phi}\mathbf{z}(t-1) + \mathbf{w}(t), \tag{2.27}$$

where $\mathbf{G}(t)$ is a $k \times d$ known factor loading matrix, $\boldsymbol{\epsilon}(t) \sim \mathcal{MN}(\mathbf{0}, \boldsymbol{\Sigma}_\epsilon)$ with covariance $\boldsymbol{\Sigma}_\epsilon$, $\boldsymbol{\Phi}$ is an $d \times d$ coefficient matrix and $\mathbf{w}(t) \sim \mathcal{MN}(\mathbf{0}, \boldsymbol{\Sigma}_w)$. The EM algorithm was derived for estimating parameters $\{\boldsymbol{\Sigma}_\epsilon, \boldsymbol{\Phi}, \boldsymbol{\Sigma}_w\}$ along with the KF. We call this approach the EM algorithm of the vector autoregressive model of order 1 (EM-VAR(1)).

The proposed FMOU approach has three advantages. First, we developed fast algorithms to estimate the factor loading matrix and enable selecting a reduced dimension d, while an identity factor loading matrix is often assumed with $d = k$ in the EM-VAR(1) approach, when $\mathbf{G}(t)$ is unknown [1]. Second, the KF and RTS smoothers in our FMOU approach only take $\mathcal{O}(nd)$ operations, which is substantially faster than the EM algorithm, as it requires $\mathcal{O}(nk^3)$ for inverting a $k \times k$ matrix at each time step in KF. Third, each factor process of the FMOU model can be set to be stationary using Lemma 5, yet the EM-VAR(1) approach cannot guarantee that the latent process is stationary. The computational complexity of the EM-VAR(1) model [82, 1] is given in Table 2.1. We will numerically compare FMOU and EM-VAR(1) in Experiment 3.

2.3.3 The FMOU approach enables scalable and robust estimation for the generalized probabilistic principal component analysis

The FMOU model is connected to the generalized probabilistic principal component analysis (GPPCA) approach [19], which assumes each latent factor follows a Gaussian process. However, when the variance and range parameters of the latent processes are distinct, in each iteration of the parameter estimation, one needs to apply an iterative algorithm to estimate the factor loading matrix on the Stiefel manifold [94]. In total, one needs $M_1 M_2$ iterations, where M_1 is the number of iterations in numerical optimization of the parameters, each requiring M_2 iterations to optimize the factor loading matrix. This process can be prohibitively expensive. Furthermore, numerical optimization of d sets of range and variance parameters can also be unstable. In FMOU, we only need M EM iterations, as the estimation of parameters have closed-form expressions. We use Experiment 1 to numerically compare the stability and computational cost of FMOU and GPPCA.

Experiment 1 *The data are generated by Equations (2.3)-(2.4). The orthogonal matrix* $\mathbf{U}_0 \in \mathbb{R}^{k\times d}$ *is sampled from the Stiefel manifold with two configurations with* $(k = 20, d = 5)$ *and* $(k = 40, d = 10)$*, where* d *is assumed to be known. The latent factors are generated with inputs* $t = 1, \ldots, n$ *for three scenarios with* $n \in \{100, 200, 400\}$*. The parameters* ρ_l *and* σ_l^2 *are sampled from Unif* $(0.95, 1)$ *and Unif* $(0.5, 1)$ *for* $l = 1, ..., d$*, respectively, and the variance of the noise is* $\sigma_0^2 = 0.2$*. We repeat the simulation* $N = 20$ *times under each configuration.*

We use two criteria to evaluate the estimation of the latent factor loadings and the mean. First, we compute the largest principal angle ϕ_d between $\mathcal{M}(\hat{\mathbf{U}}_0)$ and $\mathcal{M}(\mathbf{U}_0)$,

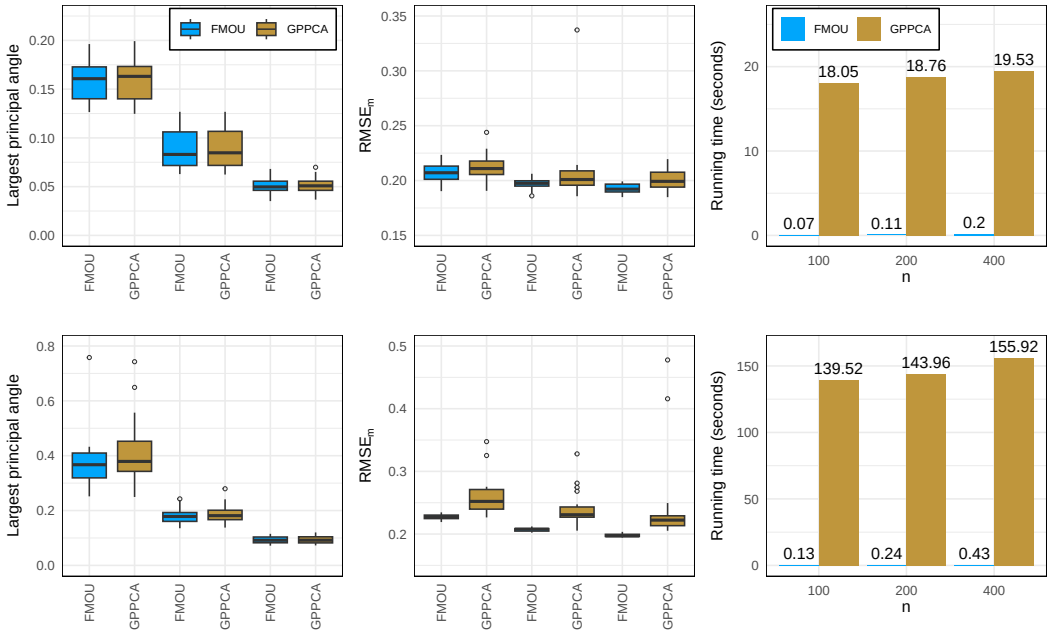

Figure 2.1: The largest principal angle between the true loading matrix $\mathbf{U}_0$ and its estimates, the RMSE_m and the running time of performing models in Experiment 1 with a correctly specified number of latent factors in FMOU and GPPCA. The first and second rows show the scenarios with $(k = 20, d = 5)$ and $(k = 40, d = 10)$, respectively. The first 2, middle 2, and last 2 boxes are associated with $n = 100$, $n = 200$, and $n = 400$, respectively, in each figure.

the linear subspaces spanned by the estimation $\hat{\mathbf{U}}_0$ and by the truth $\mathbf{U}_0$, respectively, to measure the closeness of two linear subspaces. Let $0 \leq \phi_1 \leq \ldots \leq \phi_d \leq \pi/2$ be the principal angles:

$$\phi_l = \arccos\left(\max_{\mathbf{u}\in\mathcal{M}(\mathbf{U}_0),\hat{\mathbf{u}}\in\mathcal{M}(\hat{\mathbf{U}}_0)} |\mathbf{u}^\top \hat{\mathbf{u}}|\right) = \arccos(|\mathbf{u}_l^\top \hat{\mathbf{u}}_l|), \tag{2.28}$$

such that $\|\mathbf{u}\| = \|\hat{\mathbf{u}}\| = 1, \mathbf{u}^\top \mathbf{u}_l = \hat{\mathbf{u}}^\top \hat{\mathbf{u}}_l = 0$, with $\mathbf{u}_l$ and $\hat{\mathbf{u}}_l$ being the lth column of $\mathbf{U}_0$ and $\hat{\mathbf{U}}_0$ for $l = 1, \ldots, d-1$. A smaller largest principal angle indicates a better estimation.

Second, we compute the root of the mean squared error for the mean of the observa-

tions:

$$\text{RMSE}_m = \frac{1}{N}\sqrt{\frac{\sum_{i=1}^{k}\sum_{t=1}^{n}\left(\hat{y}_i(t) - \mathbb{E}[y_i(t)]\right)^2}{kn}},$$

where $\hat{y}_i(t)$ is the predictive mean of $y_i(t)$ and N is the number of replications.

Figure 2.1 compares FMOU with GPPCA. The largest principal angles between the exact loading matrix and its estimations from FMOU and GPPCA are close. The FMOU is consistently better than the GPPCA in estimating the mean, and the effect is more pronounced for $d = 10$, a larger dimension of the latent space. This is because numerically optimizing a large number of parameters and factor loading matrix leads to a larger error from GPPCA, whereas such a problem is overcome by closed-form expressions in each EM iteration in FMOU. Furthermore, the FMOU is much faster than the GPPCA, as shown in the right panels in Figure 2.1, as the estimation of factor loadings has a closed-form solution in FMOU, yet numerical optimization in the Stiefel manifold is required in each optimization step in GPPCA.

Hence, FMOU achieves tremendous improvements in terms of scalability and efficiency for the model in (2.3)-(2.4), compared to GPPCA. In Sections 2.4 and 2.5, we use extensive simulated and real examples with correctly specified or misspecified scenarios to compare FMOU with alternative approaches. The numerical results are computed by the macOS Mojave system with an 8-core Intel i9 processor running at 3.60 GHz and 32 GB of RAM.

2.4 Simulated experiments

In this section, we numerically compare our method with alternative approaches by simulation. The experiments are split into two subsections. The scenarios with an

estimated factor loading matrix are considered in Section 2.4.1, whereas Section 2.4.2 focuses on cases with a fixed factor loading matrix. Both correctly specified models and misspecified models are considered. In Section 2.4.1, we compared with DMD and two data-driven latent factor models, denoted as LY1 and LY5, introduced in [21]. In the LY1 and LY5 methods, the number of latent factors $\hat{d}$ is first estimated by minimizing the ratio of neighboring eigenvalues of $\mathbf{C} := \sum_{p=1}^{p_0} \hat{\mathbf{\Sigma}}_y(p)\hat{\mathbf{\Sigma}}_y^\top(p)$, with $p_0 = 1$ for LY1 and $p_0 = 5$ for LY5, where $\hat{\mathbf{\Sigma}}_y(p)$ denotes the sample covariance matrix with a lag time p. In both methods, the factor loading matrix $\mathbf{U}_0$ is estimated by the first d eigenvectors of $\mathbf{C}$ corresponding to the largest d eigenvalues. Results with an estimated latent factor loading matrix $\hat{d}$ are shown in Appendix B.4. For Experiment 3 in Section 2.4.1, we also include EM-VAR(1) [82] with the default setting $\mathbf{G}(t) = \mathbf{I}_k$ in Equation (2.26) for estimating the mean from noisy observations. The non-orthogonal eigenvectors of the $\mathbf{\Phi}$ in Equation (2.27) play a similar role as the factor loading matrix in the FMOU model, which is estimated by the data. We do not include EM-VAR(1) in other simulated examples as they are too expensive to compute. In Section 2.4.2, we construct two additional experiments where the factor loadings are either sampled from the Stiefel manifold or derived from the singular vectors of the Green's function used in real data, mimicking the slip propagation process. We compare FMOU with NIF [2] for these two experiments. Other than estimation error and computational time, we also record the proportion of the signals and slips covered in 95% posterior credible intervals and the average length of the intervals of the FMOU approach in Tables B.1-B.6 in the Appendix, yet these uncertainty measures are not available for other methods, such as DMD.

2.4.1 Simulated experiments with estimated factor loadings

We first study Experiment 2, where all parameters, $(\mathbf{U}_0, \boldsymbol{\sigma}_0^2, \boldsymbol{\sigma}^2, \boldsymbol{\rho})$, are estimated.

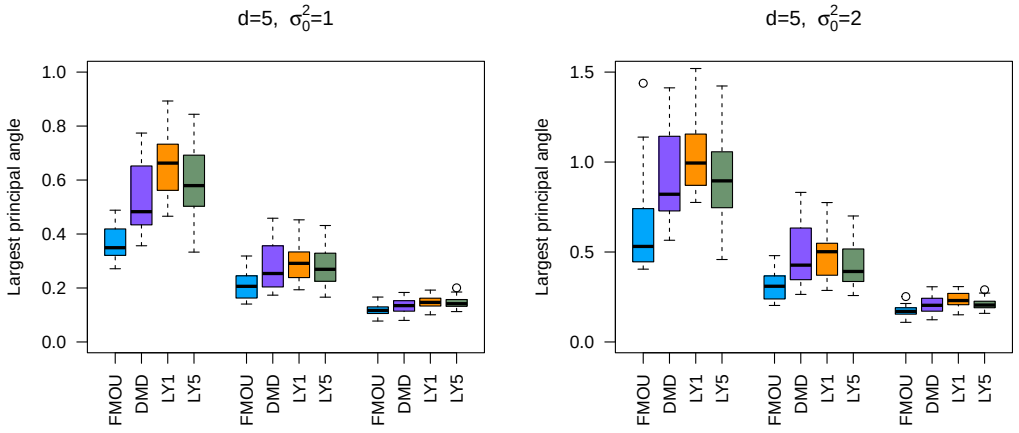

Figure 2.2: The largest principal angle (from 0 to $\pi/2$) between the true loading matrix $\mathbf{U}_0$ and its estimates from 4 methods in Experiment 2 with correctly specified latent factors. The variance of the noise is assumed to be $\sigma_0^2 = 1$ and $\sigma_0^2 = 2$ for the left and right panels, respectively. In each panel, the first 4, middle 4, and last 4 boxes are associated with a different number of time points, $n = 100$, $n = 200$, and $n = 400$, respectively.

Experiment 2 (Correctly specified models with estimated factor loading) *The data are generated by Equations (2.3)-(2.4) with the orthogonal matrix* $\mathbf{U}_0$ *sampled from the Stiefel manifold with* $k = 20$ *and* $d = 5$. *Here* ρ_l *and* σ_l^2 *are sampled from Unif* $(0.95, 1)$ *and Unif* $(0.5, 1)$ *for* $l = 1, ..., d$, *respectively. Six scenarios with three choices of time points* $n \in \{100, 200, 400\}$ *and two noise variances* $\sigma_0^2 \in \{1, 2\}$ *are considered. We repeat the simulation* $N = 20$ *times for each scenario.*

In Figure 2.2, we show the largest principal angle of factor loading matrix $\mathbf{U}_0$ for all approaches with a correctly specified d. Across all scenarios, the estimation by the FMOU approach has the smallest principal angles between the estimated and true factor loading matrix. In Figure B.1 in the Appendix, we show that the number of latent factors can be correctly estimated by the IC in Equation (2.23), and the estimation is more accurate than the alternatives.

The RMSE_m in estimating the mean of the data of Experiment 2 is shown in Table

$d = 5$	$\sigma_0^2 = 1$			$\sigma_0^2 = 2$		
	$n = 100$	$n = 200$	$n = 400$	$n = 100$	$n = 200$	$n = 400$
FMOU	**0.38**	**0.35**	**0.33**	**0.50**	**0.44**	**0.41**
DMD	0.63	0.63	0.63	0.77	0.76	0.77
LY1	0.91	0.58	0.57	1.3	0.83	0.81
LY5	0.89	0.57	0.58	1.3	0.81	0.81

Table 2.2: Average of RMSE_m over N repeats of Experiment 2 with the truth $d = 5$. The average standard deviations of the output across all sample sizes and repeats are 2.90 and 3.10 for $\sigma_0^2 = 1$ and $\sigma_0^2 = 2$, respectively.

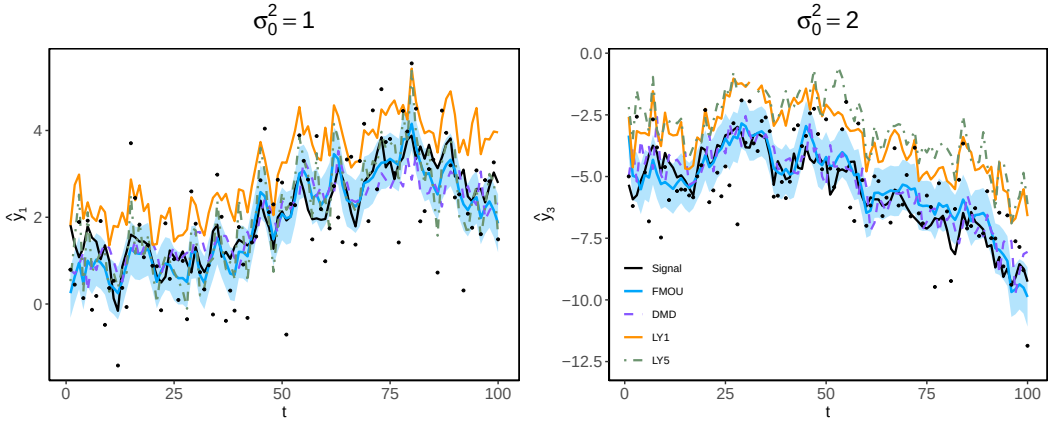

Figure 2.3: Predictive mean by FMOU (solid blue curves), DMD (dashed purple curve), LY1 (solid orange curves), and LY5 (dashed green curves), from one repetition in Experiment 2. The observations and means of the observations are plotted by the black circles and curves, respectively. The blue-shaded area is the 95% posterior credible interval given by FMOU.

2.2. The FMOU achieves higher accuracy under all combinations with different n and σ_0^2. In Figure 2.3, we plot the observations, the mean, and estimation by different methods for $\sigma_0^2 = 1$ and $\sigma_0^2 = 2$. The predictive mean from FMOU is close to the truth, plotted as black curves, and the 95% posterior credible interval of the mean covers the truth most of the time. Based on Equation (2.20), the 95% posterior credible interval of the jth observation mean at time t is given by $[(\mathbf{U}_0\mathbf{z}^{\text{post}}(t))_j - 1.96\Sigma_{m,jj}^{\text{MMLE}}(t), (\mathbf{U}_0\mathbf{z}^{\text{post}}(t))_j + 1.96\Sigma_{m,jj}^{\text{MMLE}}(t)]$, where $(\mathbf{U}_0\mathbf{z}^{\text{post}}(t))_j$ is the j-th element of $\mathbf{U}_0\mathbf{z}^{\text{post}}(t)$ and $\Sigma_{m,jj}^{\text{MMLE}}(t)$ is the j-th diagonal element of $\mathbf{\Sigma}_{m,jj}^{\text{MMLE}}(t)$.

Experiment 3 (Misspecified models with estimated factor loading)

We consider two test cases where signals are generated by physical processes different from our models.

(3a) ***Linear diffusion*** *[97]. The signal is governed by the partial differential equation* $\frac{\partial u(x,t)}{\partial t} = D\frac{\partial^2 u(x,t)}{\partial x^2}$*, where* $u(x,t)$ *represents the concentration of the diffusing material at location* x *and time* t*, and* D *is the diffusion coefficient. We follow [98] to let* $D = 1$*, discretize the spatial domain* $[0,1]$ *into* 300 *equally spaced grid points,* $u(x,0) = 0$*, and let a boundary condition be applied at one end with a constant external concentration of 1. The signal is generated over a time interval* $t \in [0, 0.2]$ *using* $n = 300$ *with a numerical solver [99]. Three noise variances* $\sigma_0^2 \in \{0.01^2, 0.05^2, 0.3^2\}$ *are tested.*

(3b) ***Branin function*** *[100]. The underlying signal is generated by the Branin function, which takes a two-dimensional input* (x_1, x_2) *and yields a scalar output given by* $f(x_1, x_2) = a(x_2 - bx_1^2 + cx_1 - r)^2 + s(1-t)\cos(x_1) + s$*, where* $x_1 \in [-5, 10]$ *and* $x_2 \in [0, 15]$*. The recommended parameter values are used:* $a = 1, b = \frac{5.1}{4\pi^2}, c = \frac{5}{\pi}, r = 6, s = 10$ *and* $t = \frac{1}{8\pi}$*. When generating the signal, the input domain is uniformly discretized into a* 300×300 *grid. Noisy observations are then obtained via* $y(x_1, x_2) = f(x_1, x_2) + \epsilon$*, where* ϵ *represents independent Gaussian noise with variance* σ_0^2*. We test the effects of varying noise variances, specifically* $\sigma_0^2 \in \{1, 5^2, 20^2\}$*.*

Table 2.3 presents the average RMSE_m and computational time for signal estimation over $N = 20$ simulations across various approaches. The results show that FMOU consistently achieves better accuracy under all noise levels, particularly excelling in large-noise scenarios. Moreover, FMOU is much faster over the EM-VAR(1) method with the default setting [1] as the FMOU enables dimension reduction of the latent factors and avoids inverting the covariance matrix of the observations at each time in the Kalman

	Linear diffusion				Branin function			
	RMSE_m $\sigma_0^2 = 0.01^2$	RMSE_m $\sigma_0^2 = 0.05^2$	RMSE_m $\sigma_0^2 = 0.3^2$	Avg Time	RMSE_m $\sigma_0^2 = 1$	RMSE_m $\sigma_0^2 = 5^2$	RMSE_m $\sigma_0^2 = 20^2$	Avg Time
FMOU	**0.0024**	**0.0083**	**0.038**	0.56	**0.14**	**0.60**	**2.2**	**0.23**
DMD	0.010	0.047	0.29	**0.44**	0.79	4.5	19	0.41
LY1	0.10	0.050	0.30	3.6	2.4	11	20	3.5
LY5	0.010	0.050	0.30	18	5.9	5.8	20	17
EM-VAR(1)	0.0098	0.050	0.30	672	4.1	5.5	60	807

Table 2.3: Average RMSE_m and computational time (in seconds) for Experiment 3. RMSE_m is averaged over $N = 20$ repeats per noise level. The computational time is averaged over three noise levels and $N = 20$ repeats. For linear diffusion, the average standard deviations of the output are 0.29 ($\sigma_0^2 = 0.01^2$), 0.30 ($\sigma_0^2 = 0.05^2$) and 0.42 ($\sigma_0^2 = 0.3^2$). For Branin function, the average standard deviations of output are 51.57 ($\sigma_0^2 = 1$), 51.81 ($\sigma_0^2 = 5^2$) and 55.32 ($\sigma_0^2 = 20^2$).

filter. Figure 2.4 provides the comparison between FMOU and DMD in estimating the signal from noisy observations generated by the linear diffusion equation and Branin function, respectively. The signal estimated by FMOU closely aligns with the ground truth, while DMD exhibits larger deviations. We highlight that the convergence of the log likelihood in the FMOU approach only takes around 5-10 iterations in the EM algorithm, shown in Figure 2.4 (d) and (h).

2.4.2 Simulated experiments with known factor loadings

In this section, we assume that $\mathbf{U}_0$ is given by the SVD of Green's function. The observations are related to the unobserved slip by Equation (2.2). We quantify the estimated error of slips by RMSE_s:

$$\text{RMSE}_s = \frac{1}{N}\sqrt{\frac{\sum_{t=1}^{n}\left(\hat{\mathbf{z}}_s(t) - \mathbf{z}_s(t)\right)^\top\left(\hat{\mathbf{z}}_s(t) - \mathbf{z}_s(t)\right)}{k'n}}, \tag{2.29}$$

where $\hat{\mathbf{z}}_s(t) = (\hat{z}_{s,1}(t), \ldots, \hat{z}_{s,k'}(t))^\top$ is the estimated slips at time t, computed by $\hat{\mathbf{z}}_s(t) = \mathbf{G}^\top\mathbf{U}_0\mathbf{D}_0^{-2}\hat{\mathbf{z}}(t)$ with $\hat{\mathbf{z}}(t)$ being the estimation of latent factors.

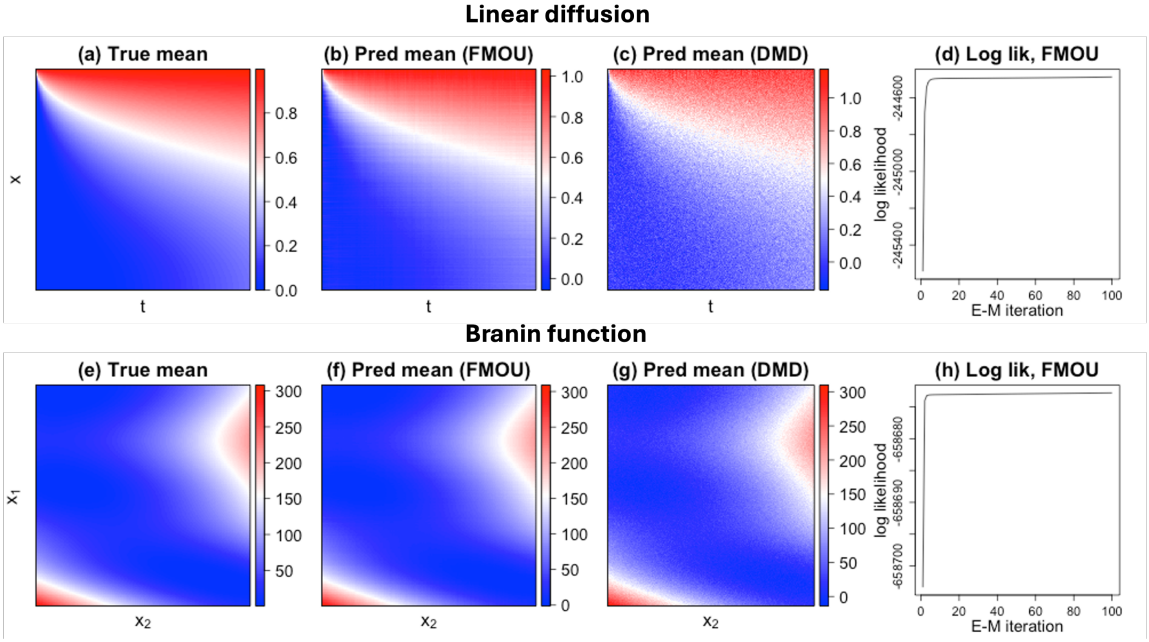

Figure 2.4: (a) True mean generated by linear diffusion. (b) Predictive mean by FMOU, where observations are generated with $\sigma_0^2 = 0.05^2$. (c) Predictive mean by DMD (i.e., $\hat{u}_{\text{DMD}}(x,t)$) where observations are generated with $\sigma_0^2 = 0.05^2$. (d) The convergence of the log likelihood in the EM algorithm of the FMOU approach. (e) True mean generated by the Branin function. (f) Predictive mean by FMOU, where observations are generated with $\sigma_0^2 = 25$.(g) Predictive mean by DMD, where observations are generated with $\sigma_0^2 = 25$. (h) The convergence of the log likelihood in the EM algorithm of the FMOU approach.

Experiment 4 (Correctly specified models with a known factor loading)

Data are generated by Equations (2.3)-(2.4), and the slips are computed by Equation (2.2), which approximates the data-generating system in Equation (2.1). The orthogonal matrices $\mathbf{U}_0 \in \mathbb{R}^{k\times k}$, $\mathbf{V}_0 \in \mathbb{R}^{k'\times k}$ *are generated from the Stiefel manifold with two configurations: (1)* $k = 25, k' = 150, d = 6$ *and (2)* $k = 32, k' = 100, d = 8$. *The* l*th latent factor* $\mathbf{z}_l$ *is generated with inputs* $t = 1, \ldots, n$ *for* $n \in \{100, 200, 300\}$. *We have* $\mathbf{D}_0 = diag(d_1, \ldots, d_k)$ *with* d_j *sampled from* $Unif(0,1)$ *for* $j = 1, ..., k$ *and rearranged decreasingly. Parameters* ρ_l *and* σ_l^2 *are sampled from* $Unif(0.95, 1)$ *and* $Unif(1,2)$*, respectively, for* $l = 1, ..., d$. *The matrix of the Green's function is constructed by* $\mathbf{G} = \mathbf{U}_0\mathbf{D}_0\mathbf{V}_0^\top$. *The variance of the noise is assumed to be* $\sigma_0^2 = 1.5$. *We repeated the simulation* $N = 20$

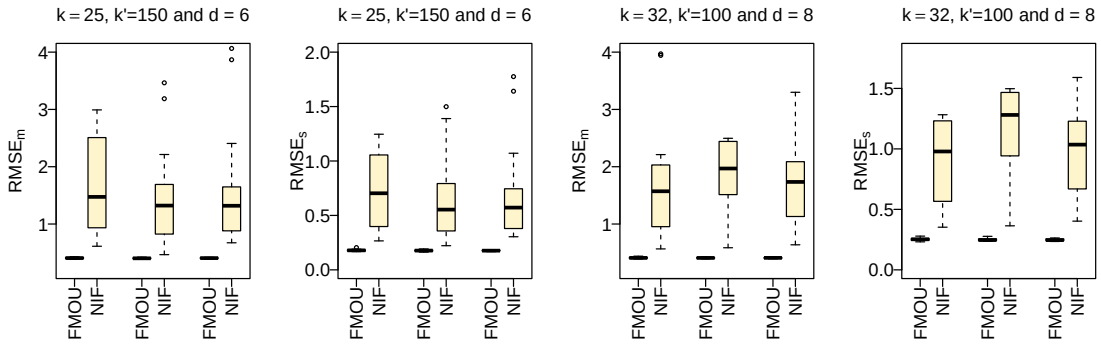

Figure 2.5: Box plots of RMSE_m and RMSE_s by FMOU and NIF in Experiment 4. The number of factors d is correctly specified for both FMOU and NIF. In each subfigure, the first 2, middle 2, and the last 2 boxes are based on $n = 100$, $n = 200$ and $n = 300$, respectively.

times for each configuration.

We compare FMOU with NIF, as other previously compared methods, such as DMD, LY1 and LY5, estimate the factor loading matrix. The RMSE_m and RMSE_s by FMOU and NIF are provided in Figure 2.5. As NIF does not involve the selection of the number of factors d, to eliminate the effect of selecting d, we assume d is known for both FMOU and NIF. Across all configurations, the FMOU model consistently outperforms NIF in estimating the mean of the observations and the slip. Additionally, Figure B.2 in the Appendix compares the estimation of d in Experiment 4 and it demonstrates that IC in Equation (2.23) can correctly identify the number of factors with a moderately large number of time points. The improvement by FMOU for Experiment 4 is not surprising, as data are generated by FMOU model. Next we use a misspecified model to illustrate the flexibility of FMOU estimation.

Experiment 5 (Misspecified model for estimating a 2D elliptical slip)

This simulation uses the Green's tensor $\mathbf{G}$ *inherited from the real data for inferring the slip propagation in the Cascadia zone, where* $k = 200$, $k' = 1978$, $n = 88$. *An elliptical slip region is generated on a triangular mesh with a fixed center at* $(\xi_{0,1}, \xi_{0,2}) =$

$(123.2^\circ W, 46.0^\circ N)$ and a fixed semi-minor axis extending from $(123.9^\circ W,\ 46.0^\circ N)$ to $(122.5^\circ W,\ 46.0^\circ N)$ with ground distance $r_0 = 108km$. The semi-major axis grows along the longitude as $r(t) = r_0/3 + vt$ with the growth rate $v = 8km/day$. Data is generated through model (2.1) and the slip at the hth fault patch $(\xi_{h,1}, \xi_{h,2})$ and time t is given by

$$z_{s,h}(t) = z_{s,max}(1 - E(\boldsymbol{\xi}_h, t)^2) \times 1_{\{E(\boldsymbol{\xi}_h,t)\leq 1\}}, \tag{2.30}$$

where $E(\boldsymbol{\xi}_h, t) = (\xi_{h,1}-\xi_{0,1})^2/r_0^2+(\xi_{h,2}-\xi_{0,2})^2/r(t)^2$, $h = 1, \ldots, k', t = 1, \ldots, n$. The slips have a peak value of $z_{s,max} = 3cm$ at the center and drop to zero outside the ellipse. We consider three configurations with known noise variances $\sigma_0^2 \in \{0.01^2, 0.05^2, 0.2^2\}cm^2$.

In Experiment 5, both $\mathbf{U}_0$ and σ_0^2 are known to mimic the application of the real data. We use the VM method in (2.22) to estimate the number of latent factors. Table 2.4 records the numerical comparisons between FMOU and NIF for Experiment 5. We consider two variants of NIF: one with $d = k$ and the other one with an estimated $\hat{d}$ from VM. The FMOU achieves better accuracy in estimating both the mean of the observations and slips. Additionally, FMOU is considerably faster than the NIF model, as computing the likelihood function in FMOU only requires $\mathcal{O}(knd)$ operations. These orders do not consider the SVD of Green's function, which only needs to be done once. Furthermore, as each step of the EM algorithm has closed-form expressions, FMOU is also robust in estimating a large number of parameters.

Figure 2.6 graphs the 7-day averaged slips in centimeters, i.e., $\bar{z}_{s,h}(t) = \sum_{t'=0}^{6} \hat{z}_{s,h}(t + t')/7$ for $h = 1, \ldots, k'$. Both FMOU and NIF detect the front of ellipses and estimate slips' propagation within the elliptical regions. The slips estimated by the FMOU model align more closely with the truth, which demonstrates that FMOU is more accurate in estimation.

$\sigma_0^2 = 0.01^2$	$\hat{d}$	Avg. RMSE$_m$	Avg. RMSE$_s$	Running time (seconds)
FMOU	105	**0.0039**	**0.23**	**10**
NIF	105	0.11	0.64	549
NIF	200	0.11	0.64	2271
$\sigma_0^2 = 0.05^2$	$\hat{d}$	Avg. RMSE$_m$	Avg. RMSE$_s$	Running time (seconds)
FMOU	67	**0.013**	**0.34**	**6.0**
NIF	67	0.12	0.66	251
NIF	200	0.12	0.66	2247
$\sigma_0^2 = 0.2^2$	$\hat{d}$	Avg. RMSE$_m$	Avg. RMSE$_s$	Running time (seconds)
FMOU	30	**0.034**	**0.48**	**3.6**
NIF	30	0.12	0.67	100
NIF	200	0.12	0.66	2243

Table 2.4: Results for Experiment 5. The standard deviation of the mean of the output and the slip are 0.19 and 0.85, respectively. The major cost in FMOU lies in estimating the number of latent factors. FMOU takes 9.2, 5.3 and 3.2 seconds for $\sigma_0^2 = 0.01^2, 0.05^2, 0.2^2$ to estimate the number of latent factors, d, respectively.

2.5 Estimating slip propagation in Cascadia

2.5.1 Data and methods

We employ GPS measurements to estimate slip rates within the Cascadia region, situated along the western edge of North America. The Cascadia region is characterized by the subduction of the Juan de Fuca plate beneath the North American plate. This geological interaction triggers slow slip events that last for weeks to months. The dataset used in this study is publicly available at the Plate Boundary Observatory, which contains observations from $\tilde{k} = 100$ GPS stations over $n = 88$ days in 2011 [85]. Each observation represents the daily average geographical position recorded in three directions: East-West, North-South and Up-Down. We follow [85] to use GPS measurements in the East-West and North-South directions, as the vertical displacement measurements contain little information for slip estimation. Figure 2.7(a) shows the cumulative displacements in

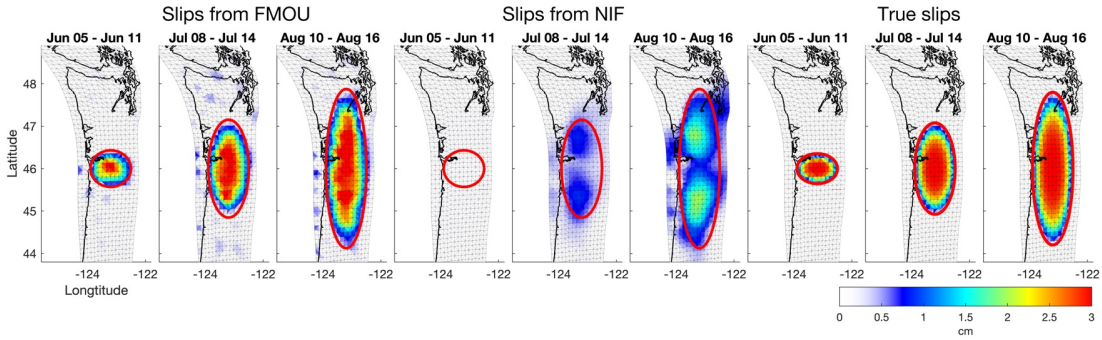

Figure 2.6: Seven-day averages of slips estimated by FMOU (left 3 panels) and NIF (middle 3 panels) in Experiment 5 when $\sigma_0^2 = 0.01^2\ cm^2$. The true slips (right 3 panels) propagate as a growing ellipse. The red boundary shows the front of the slip region.

the Cascadia region between June 3, 2011 to August 30, 2011, while the displacements in the East-West and North-South directions of six GPS stations are shown in Figure 2.7(b). As identified in [101], the observations encompass secular signals, such as annual and semiannual components, which are typically removed before the analysis [85]. Details of preprocessing GPS measurements before modeling are provided in Appendix B.5. Apart from the GPS measurements, we utilize determinations of tectonic tremor locations from the same region and period [102] as an independent assessment of the location of geologic slip at a given time.

The Green's functions here are built on a mesh consisting of $k' = 1978$ triangular patches [103] located on a fault plane beneath the ground and they are calculated by assuming triangular dislocations in a homogeneous and elastic half-space [104]. Observations from 100 GPS stations in two directions yield a 200×1978 Green's function matrix $\mathbf{G}$. Furthermore, measurements are typically influenced by assigning the GPS-derived displacements in a North American Plate fixed reference frame, and local displacements of the GPS monuments. Thus, it is common to include the frame motion, a time-dependent trend of each direction shared by each GPS station [85]. Consequently, the augmented model can be written as $\mathbf{y}(t) = \mathbf{G}^{aug}\mathbf{z}_s^{aug}(t) + \boldsymbol{\epsilon}(t)$, where $\mathbf{G}^{aug} = (\mathbf{G}, \mathbf{I}_{k\times 2})$

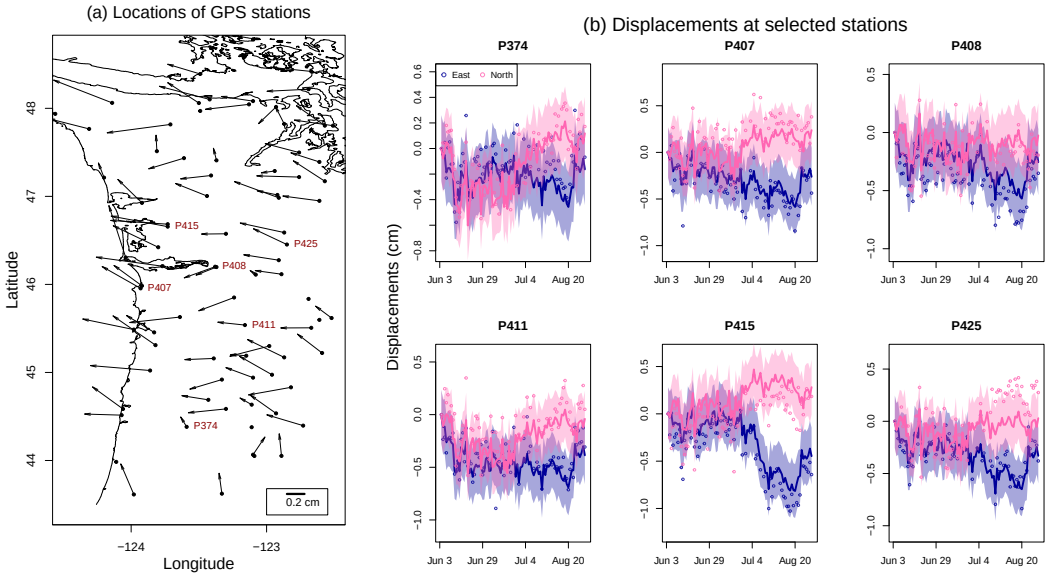

Figure 2.7: (a) The cumulative displacements from June 3, 2011, to August 30, 2011, in the Cascadia region are plotted by dark arrows with the unit of the measurements given in the inset. (b) The displacements by six GPS stations in East-West and North-South directions during the 2011 episodic tremor and slip event in centimeters (cm). The solid lines and corresponding shaded regions represent the predictive mean of the data and the 95% posterior credible intervals by FMOU, respectively.

with $\mathbf{I}_{k\times 2} := (\mathbf{I}_2, \ldots, \mathbf{I}_2)^\top$, and $\mathbf{z}_s^{aug}(t) = [\mathbf{z}_s(t)^\top, f_E(t), f_N(t)]^\top$ with $f_E(t)$ and $f_N(t)$ being the frame motion in the East-West and North-South directions, respectively, for $t = 1, \ldots, n$. The Gaussian noise vector follows $\boldsymbol{\epsilon}(t) \sim N(0, \sigma_0^2 \mathbf{I}_k)$ with σ_0^2 being the observed variance. Following [85], we estimate the magnitude of the geologic slips projected onto a horizontal plane at 52 degrees clockwise from North along with fault direction [101].

We compare three methods. For FMOU, we use the VM method in (2.22) for estimating the number of latent factors and the parameters are estimated by Algorithm 1. We use the posterior mean of slip $\hat{\mathbf{z}}_s(t)$ from (2.21) for estimation. Then the slip rates are calculated by

$$\hat{\mathbf{z}}_{s,rate}(t) = \hat{\mathbf{z}}_s(t+1) - \hat{\mathbf{z}}_s(t), \tag{2.31}$$

for $t = 1, \ldots, n-1$. Following [85], we truncate the negative slip rates to zero, as it is

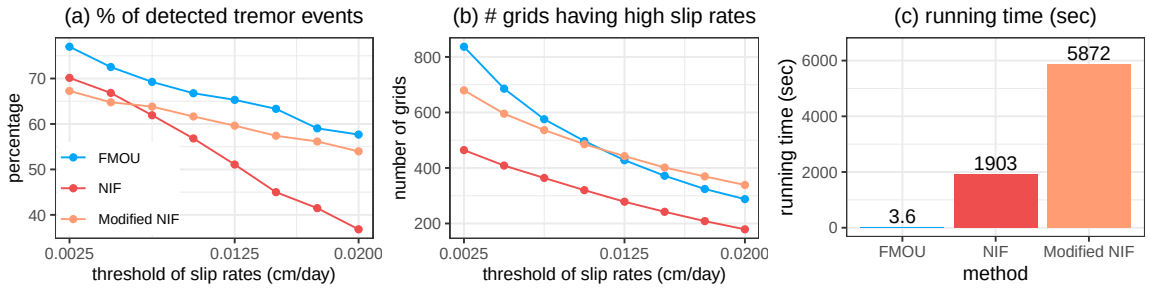

Figure 2.8: Model performance in 2011 Cascadia data. We compare three methods: FMOU (blue), NIF (red), and modified NIF (orange). (a) Proportion of identified tremor. (b) The number of grids containing large average slip rates. (c) Running time (seconds).

deemed unlikely that the fault would slip backward in the prevailing tectonic stress state. For comparison, we include the NIF model [2] and the modified NIF model [85], which are summarized in Appendix B.3. We do not include EM-VAR(1) in the real example as it requires inverting a $k' \times k'$ matrix for each time in the RTS smoother in the EM algorithm, which is too costly to compute.

2.5.2 Results

To evaluate the model performance, we use the held-out tremor locations to validate the estimation, as tremor is well associated in both space and time with geologic slip. The tremors are monitored continuously in the Cascadia zone [102, 105], available from the Pacific Northwest Seismic Network catalog. We consider two metrics: the proportion of tremors identified by large slip rates with overlapping grids (the true positive rate), and the total grids containing large slip rates (the positive rate).

Panels (a)-(b) in Figure 2.8 show the proportion of detected tremor events and the number of grids that contain high slip rates. The FMOU model detected the highest number of tremor events among all methods across all thresholds of the slip rates. A slip rate larger than 0.0125 cm/day is considered to be high. The FMOU model has a

smaller number of spatial grids detected to have a high slip rate and it detects around 5% more tremors than the modified NIF model, shown in panel (b) of Figure 2.8. The NIF model has a substantially lower true positive rate compared to the modified NIF and FMOU, as it estimates slip rates to be negative or close to zero for more grids than the other two models. Because the noise in GPS measurement is relatively large compared to the slip-generated signal, a flexible model is preferred for capturing the heterogeneous slip changes. The FMOU model, with distinct correlation and variance parameters for each latent process, is more flexible than the NIF and modified NIF for modeling the slip propagation.

The computational time of the three methods is given in panel (c) in Figure 2.8. Notably, the computational time of the FMOU is more than 528 times and 1631 times faster than the NIF model and the modified NIF model, respectively. The most computationally intensive step for the FMOU is estimating the number of factors, which takes 3.03 seconds. After selecting the number of latent processes, estimating all parameters and slip rates requires only 0.54 seconds. This dramatic acceleration in computation enables the use of massive datasets from large GPS networks, and provides new opportunities to jointly estimate the hazards across larger regions, which could otherwise be prohibitive due to the large computational expense.

Figure 2.9 shows the seven-day average of the estimated slip rates of three periods by the FMOU, NIF and modified NIF models, where the solid dots represent the tremor epicenters. Estimation of slip rates of other periods is provided in Figure B.4 in Appendix. The estimates from FMOU align well with tremor dataset events, even though the GPS measurements and tremor datasets were collected independently from distinct sources. Compared to the modified NIF, the FMOU model provides better agreement between high-slip regions and tremor epicenters during August 10–16. In contrast, the modified NIF model detects a much larger area with relatively high slip rates, particularly in the

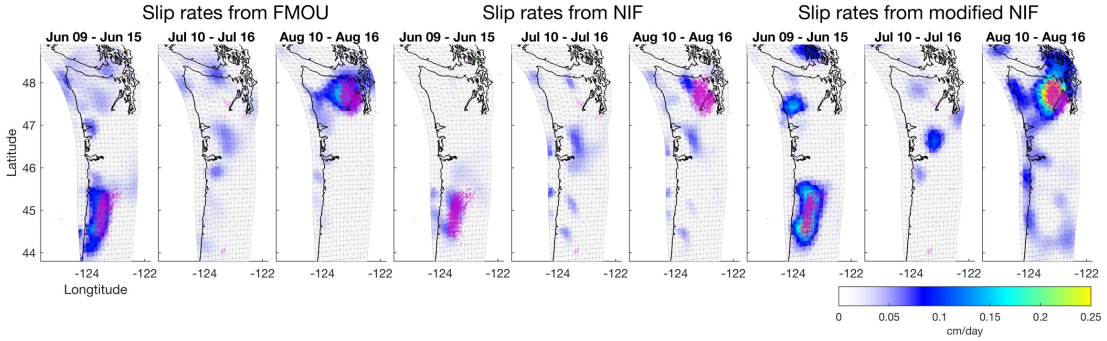

Figure 2.9: The seven-day averages of slip rates estimated by the FMOU, NIF and modified NIF are plotted in the left 3, middle and right 3 panels, respectively, for the Cascadia displacement data in 2011. The tremor epicenters are plotted as magenta dots.

southeast region during August 10–16, resulting in a higher false positive rate. Both FMOU and modified NIF reveal an area of high slip rates centered around 45°N and 123.5°W in early June, and another region with high slip rates centered around 47.5°N and 123°W in early August. However, the migration of the estimated high slip rate region is less apparent in the NIF model and its modified version. The finding by the FMOU model is consistent with the previous results by [85], which illustrate the physical mechanism of the coincidence between tremor centers and the regions with high slip rates.

Chapter 3

Fast Gaussian Process computation with `FastGaSP`

Statistical inference in scalar-valued Gaussian processes (GPs) with the Matérn kernel can be computationally expensive, as evaluating the marginal log-likelihood for parameter estimation in Equation (1.3) and computing the predictive distribution in Equation (1.8) requires a Cholesky decomposition of an $n \times n$ correlation matrix, at a cost of $\mathcal{O}(n^3)$, where n is the number of observations. However, as introduced in Chapter 1.3, when the input is one-dimensional and the Matérn kernel has a half-integer roughness parameter, the corresponding Gaussian process admits an equivalent representation as a discrete-time dynamic linear model with a Markov structure. This representation allows the use of forward filtering and backward smoothing, reducing the overall computational cost to $\mathcal{O}(n)$, as illustrated in Lemmas 1-2.

Motivated by this computational advantage, in this chapter, we introduce three modules in the `FastGaSP` R package that provide fast and exact computation for Gaussian processes with the Matérn kernel and one-dimensional input. We begin with the `fgasp` module, which handles scalar-valued output. For Matérn kernels with roughness pa-

rameter $\nu \in \{0.5, 1.5, 2.5\}$, given the range parameter and the nugget-to-variance ratio (when noise is present), it evaluates the natural logarithm of the profile likelihood and computes the predictive distribution via forward filtering and backward smoothing, at a cost of $\mathcal{O}(n)$. We then present the `GPPCA` module, which models correlated vector-valued output via the generalized probabilistic principal component analysis (GPPCA) [19]. In this framework, the observations are modeled by a latent factor model with an orthogonal factor loading matrix, where each latent process is independently modeled by a Matérn GP. The computation of predictive distributions in GPPCA builds directly on the `fgasp` module. Finally, we introduce the `FMOU` module as a special case of GPPCA, in which the inputs are equally spaced and each latent process follows an Ornstein-Uhlenbeck (OU) process. The model specification and parameter estimation are detailed in Chapter 2.

3.1 The `fgasp` module

The main purpose of the `fgasp` module is to provide fast and exact computation for scalar-valued GPs. In particular, we focus on observations with one-dimensional inputs and the Matérn kernel with a half-integer roughness parameter. Under this setting, the GPs can be represented by a dynamical linear model, enabling forward filtering and backward smoothing, which significantly reduces the computational complexity from $\mathcal{O}(n^3)$ to $\mathcal{O}(n)$, without introducing approximation error.

3.1.1 Main functions of `fgasp` module

Figure 3.1 illustrates three main functions in the `fgasp` module. The first function `fgasp` accepts the training input as well as output, and serves as the model setup step. Given a set of range and nugget (if any) parameters, the second function `log_lik` computes the natural logarithm of the profile likelihood, enabling efficient parameter es-

timation. Finally, the `predict()` takes the estimated or user-specified parameters along with a vector of testing inputs, and returns the predictive mean and variance at those locations.

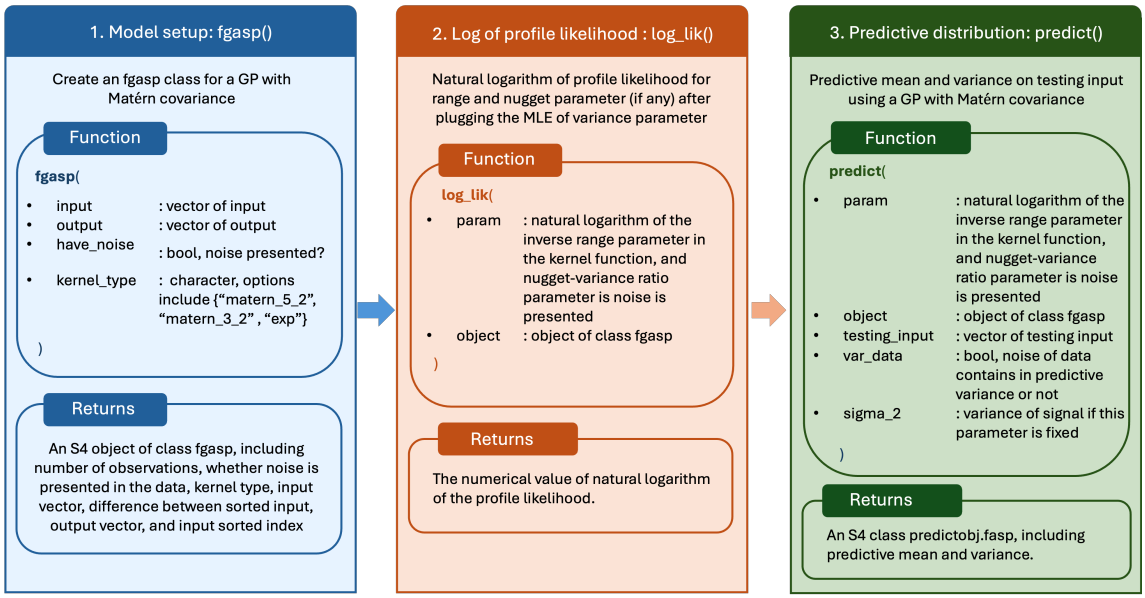

Figure 3.1: Main functions of `fgasp` module in R package `FastGaSP`: `fgasp()`, `log_lik` and `predict()`, along with functions' arguments and returned values.

The `fgasp` function

The `fgasp()` serves as the foundation of the module, responsible for GP model specification and gathering all data needed for computation. It requires two primary arguments: the n-dimensional input vector and the corresponding output vector, where the outputs are assumed to have zero mean. By default, the observations contain measurement errors and are modeled by a Matérn 5/2 kernel. Users can also specify a noise-free GP by setting `have_noise=F`, and may select among three Matérn kernel types with different roughness parameters via the `kernel_type` argument, with options {`"exp"`, `"matern_3_2"`, `"matern_5_2"`}, corresponding to roughness parameters $\nu = \frac{1}{2}, \frac{3}{2}$, and $\frac{5}{2}$, respectively.

The function returns an S4 object of class `fgasp`, which contains the model speci-

fication, sorted input and output data, and auxiliary quantities used for computation. The `num_obs` stores the number of observations, while `have_noise` and `kernel_type` store the corresponding values from the function input. In addition, the slots `input` and `output` contain the sorted input vector and the associated output vector, respectively, and `delta_x` stores the differences between consecutive sorted inputs, which are precomputed here and used in forward filtering. All slots are passed to `log_lik()` and `predict()` in the subsequent steps.

The `log_lik` function

The `log_lik` function evaluates the exact natural logarithm of the profile likelihood in $\mathcal{O}(n)$ operations, which is particularly important when the profile likelihood is repeatedly evaluated during numerical optimization over the range and nugget parameter. Below, we provide the derivation and implementation details.

Consider a scalar-valued GP with one-dimensional input and zero-mean observations. The marginal likelihood in Equation (1.3) involves three parameters: the range parameter γ in the kernel function, the signal variance σ^2, and the nugget variance ratio η. Here, the maximum likelihood estimator of the signal variance can be derived in closed form by taking the first derivative of Equation (1.3) with respect to σ^2 and setting it to zero:

$$\hat{\sigma}^2 = \frac{\mathbf{y}^\top \tilde{\mathbf{R}}^{-1}\mathbf{y}}{n}. \tag{3.1}$$

Substituting $\hat{\sigma}^2$ back into the marginal likelihood in Equation (1.3), taking the natural logarithm, and dropping constants gives the profile log-likelihood, which depends only on (γ, η):

$$\log\left((p_f(\mathbf{y} \mid \eta, \gamma)\right) \propto -\frac{n}{2}\log(\mathbf{y}^\top \tilde{\mathbf{R}}^{-1}\mathbf{y}) - \frac{1}{2}\log|\tilde{\mathbf{R}}|. \tag{3.2}$$

Optimizing the natural logarithm of the profile likelihood over the two-dimensional

space is both faster and more numerically stable than jointly optimizing the full marginal likelihood over (γ, η, σ^2). However, directly evaluating Equation (3.2) still requires computing the inverse and determinant of $\tilde{\mathbf{R}}$, which costs $\mathcal{O}(n^3)$. In the implementation of `log_lik`, we overcome this bottleneck by utilizing the equivalence in Equation (1.38), which allows both $|\tilde{\mathbf{R}}|$ and $\mathbf{y}^\top \tilde{\mathbf{R}} \mathbf{y}$ to be computed in $\mathcal{O}(n)$ via the Kalman filter.

Since the profile likelihood does not contain σ^2, it must be factored out before performing the Kalman filter. For example, for the dynamical linear model associated with the exponential kernel, the model has signal variance $\tilde{W}(t_i) = 1 - G^2(t_i)$, variance of the initial state $\tilde{\Sigma}_1 = 1$ and noise variance $\tilde{V}(t_i) = \eta$. Other parameters are the same as those in (1.28). The Kalman filter then produces the normalized one-step-ahead predictive distribution $y(t_i) \mid \mathbf{y}_{1:(i-1)} \sim \mathcal{N}(f(t_i), \tilde{Q}(t_i))$, from which the profile log-likelihood is evaluated as

$$\log(p_f(\mathbf{y} \mid \eta, \gamma)) = C_f - \frac{n}{2} \log \left(\sum_{i=1}^{n} \frac{(y(t_i) - f(t_i))^2}{\tilde{Q}(t_i)} \right) - \frac{1}{2} \sum_{i=1}^{n} \log(\tilde{Q}(t_i)), \tag{3.3}$$

where C_f is a constant. In practice, one can use the L-BFGS-B quasi-Newton algorithm to optimize the range and nugget parameters in the profile log-likelihood.

Figure 3.2 (a) compares the computational time (in seconds) of evaluating the profile log-likelihood directly versus via `log_lik()` function. The direct method grows substantially with sample size, while the `log_lik()` remains nearly flat, demonstrating the efficiency of the `log_lik()`.

The `predict` function

The `predict()` function computes the predictive distribution of n' testing inputs with computational complexity $\mathcal{O}(n + n')$ via the forward filtering and backward smoothing, avoiding the $\mathcal{O}(n^3 + n^2 n')$ cost of directly computing Equation (1.8). The function

accepts the following arguments. `param` is a vector containing the natural logarithm of the inverse range parameter. If the data contains noise, it also includes the logarithm of the nugget-variance ratio parameter. `object` accepts an S4 object of class `fgasp` created by `fgasp()`. `testing_input` is an n'-dimensional vector of testing inputs at which predictions are made. `var_data` is a boolean argument that controls whether the noise variance $\hat{\sigma}^2\hat{\eta}$ is included in the posterior predictive variance. Setting `var_data = T` returns the variance of the noise observation, while `var_data = F` returns the variance of the predictive mean. If `sigma_2` is not specified by the user, the signal variance is estimated by the closed-form MLE in Equation (3.1). The function returns an S4 object containing the predictive mean and variance at the testing inputs.

Next, we describe the computation of the predictive mean and variance at testing inputs. Let $\{x_1, x_2, \ldots, x_n\}$ and $\{x_1^*, x_2^*, \ldots, x_{n'}^*\}$ denote the training and testing inputs, respectively. The `predict()` function first merges and sorts them into a single sequence, such that $x_{(1)} \leq x_{(2)} \leq \cdots \leq x_{(n+n')}$. The differences $\Delta x_{(i)} = x_{(i+1)} - x_{(i)}$ are precomputed to obtain the transition matrix and signal covariance matrix in the dynamic linear model at each step. The Kalman filter is then performed over all $n + n'$ inputs with σ^2 factored out. At a training input $x_{(i)}$ where an observation is available, both the predictive and filtered distributions described in Lemma 1 are applied. At a testing input $x_{(i)}$ where no observation is available, only the predictive distributions in Equations (1.34)-(1.35) are applied. The filtering distribution is set to be $\mathbf{m}(x_{(i)}) = \mathbf{a}(x_{(i)})$ and $\mathbf{C}(x_{(i)}) = \mathbf{R}(x_{(i)})$. After the forward pass, the RTS smoother runs backward over all $n + n'$ inputs following Lemma 2, producing the smoothed mean $\mathbf{s}(x_{(i)})$ and variance $\mathbf{S}(x_{(i)})$ at every input, including the testing points. Finally, for each testing input x^*, the predictive distribution

is given by

$$\mathbb{E}[y(x^*) \mid \mathbf{y}] = \mathbf{F}(x^*)\mathbf{s}(x^*) \tag{3.4}$$

$$\mathbb{V}[y(x^*) \mid \mathbf{y}] = \hat{\sigma}^2(\mathbf{F}(x^*)\mathbf{S}(x^*)\mathbf{F}^\top(x^*) + \hat{\eta}). \tag{3.5}$$

Equation (3.5) corresponds to the predictive variance of the noisy observation and is returned when `var_data=T`. When `var_data=F`, the noise variance is subtracted and the function returns $\hat{\sigma}^2\mathbf{F}(x^*)\mathbf{S}(x^*)\mathbf{F}^\top(x^*)$ as the variance.

In Figure 3.2 (b), we compare the computational time of obtaining the predictive mean at 10 testing locations, between direct computation and via the `predict()` function. Again, the `predict()` function demonstrates clear efficiency gains by leveraging the Kalman filter and RTS smoother, with the advantage becoming significant as the sample size grows.

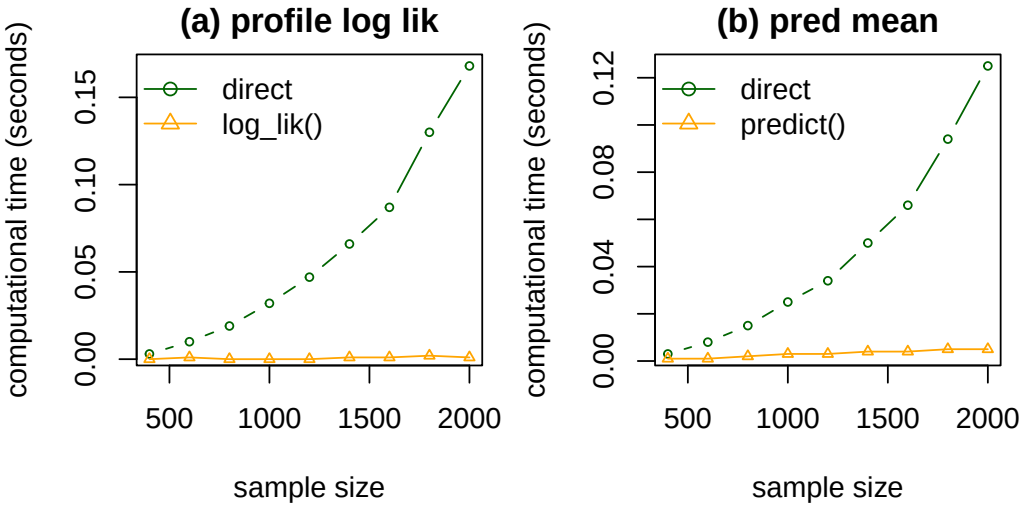

Figure 3.2: Computational time (in seconds) versus sample size for the profile log–likelihood (a) and predictive mean (b), comparing the direct computation against the `log_lik()` and `predict()`, respectively.

3.1.2 Example of fgasp module

In this section, we illustrate the use of `fgasp` to define a GP model, `log_lik` to evaluate the profile likelihood and optimize the range and nugget parameters, and `predict()` to compute the predictive distribution at the testing inputs.

We first generate a simulated dataset from a Matérn 5/2 kernel with inverse range parameter $\frac{1}{\gamma} = 0.01$, signal variance $\sigma^2 = 1$, noise variance $\sigma_0^2 = 0.01$, and sample size $n = 200$, where the inputs are equally spaced on the interval $[1, n]$.

```
R> library(FastGaSP)
R> library(RobustGaSP)
R> beta <- 0.01
R> sigma2 = 1
R> sigma0_2 = 0.01
R> eta <- sigma0_2/sigma2
R> n=200
R> input = as.numeric(1:n)
R> R00 = abs(outer(input, input, "-"))
R> R = matern_5_2_funct(R00, beta)
R> R_tilde = R + diag(eta, n)
R> L = t(chol(R))
R> true_mean = sigma2*L%*%rnorm(n)
R> output = matrix(true_mean + rnorm(n, sd=sqrt(sigma0_2)), ncol=1)
```

The `fgasp()` function takes the input and output vectors to construct a `fgasp` object, which by default assumes a Matérn 5/2 covariance kernel with noise. Calling `show()` on the fitted object prints a summary, displaying the number of observations, whether a nugget term is included, and the kernel type.

```
R> gp_model <- fgasp(input, output)
R> show(gp_model)
Number of observations:  200
Have noise:  TRUE
Type of kernel:  matern_5_2
```

For the `log_lik()` function, we first initialize the inverse range and nugget parameters on the log scale and evaluate the profile log-likelihood via the `log_lik()` at this point, yielding a value of -139.68. The `optim()` function with the L-BFGS-B method is then used to maximize the profile log-likelihood. Evaluating the `log_lik()` at the optimized parameters confirms a substantially improved profile log-likelihood.

```
R> ini_param <- c(log(0.1),log(0.1))
R> log_lik(ini_param, gp_model)
[1] -139.6808
R> opt <- optim(ini_param, log_lik, object=gp_model, method="L-BFGS-B",
                control = list(fnscale=-1))
R> opt$par
[1] -5.529775 -5.849935
R> log_lik(opt$par, gp_model)
[1] -80.10677
```

Finally, we generate 10 testing inputs uniformly over $[1, n]$ and call `predict()` with the optimized parameters to obtain the predictive distributions at the testing inputs. The predictive mean and variance are then used to construct 95% credible intervals of the predictive output.

```
R> testing_input = runif(10, 1, n)
R> pred_gp <- predict(param=opt$par, object=gp_model,
```

```
                    testing_input=testing_input)
R> pred_lb <- pred_gp@mean + qnorm(0.025)*sqrt(pred_gp@var)
R> pred_ub <- pred_gp@mean + qnorm(0.975)*sqrt(pred_gp@var)
```

3.2 The gppca module

The gppca module in the R package FastGaSP implements parameter estimation and forecasting for Generalized Probabilistic Principal Component Analysis (GPPCA) [19], a latent factor model for correlated vector-valued outputs. Specifically, given a p_x-dimensional input $\mathbf{x}$, the corresponding output vector $\mathbf{y}(\mathbf{x}) \in \mathbb{R}^k$ is modeled by $\mathbf{y}(\mathbf{x}) = \mathbf{A}\mathbf{z}(\mathbf{x}) + \boldsymbol{\epsilon}(\mathbf{x})$, as shown in Equation (1.22). In particular, the factor loading matrix $\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_d] \in \mathbb{R}^{k\times d}$ is assumed to be orthogonal, satisfying $\mathbf{A}^\top\mathbf{A} = \mathbf{I}_d$, and each latent factor is independently modeled by a zero-mean Gaussian process (GP). Therefore, given a set of inputs $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, the l-th latent process $\mathbf{z}_l = [z_l(\mathbf{x}_1), \ldots, z_l(\mathbf{x}_n)]^\top$ follows a multivariate normal distribution

$$\mathbf{z}_l \sim \mathcal{MN}(\mathbf{0}, \boldsymbol{\Sigma}_l). \tag{3.6}$$

where $\boldsymbol{\Sigma}_l = \sigma_l^2\mathbf{R}_l$ with $\mathbf{R}_l$ being a correlation matrix constructed from a correlation function for $l = 1, \ldots, d$.

The gppca module focuses on the one-dimensional input setting, i.e., $\mathbf{x} = x$, and the correlation matrices are constructed by Matérn kernel with roughness parameter $\nu = \frac{1}{2}$ or $\nu = \frac{5}{2}$. Under this framework, the model involves four sets of unknown parameters: the factor loading matrix $\mathbf{A}$, the kernel range parameters $\boldsymbol{\gamma} := (\gamma_1, \ldots, \gamma_d)$, the signal variance $\boldsymbol{\sigma}^2 := (\sigma_1^2, \ldots, \sigma_d^2)$, and the noise variance σ_0^2. In the general setting, each latent factor has its own range and variance parameters, enabling the model to capture different

scales among the output coordinates. As a simplification, the module also supports a shared covariance setting in which all latent factors have the same covariance structure, such that $\gamma_1 = \cdots = \gamma_d = \gamma$, $\sigma_1^2 = \cdots = \sigma_d^2 = \sigma^2$ and thus $\mathbf{\Sigma}_1 = \cdots = \mathbf{\Sigma}_d = \mathbf{\Sigma}$. In the rest of this section, we will introduce three main functions in the `gppca` module and give an illustrative example of the developed codes.

3.2.1 Main functions of `gppca` module

Figure 3.3 displays the workflow of the `gppca` module, which consists of three stages: model specification, parameter estimation, and forecasting. The `gppca()` function first takes the input vector and output matrix as arguments and specifies the model settings, including the number of latent factors d (if known, or whether d should be estimated), the kernel type, and whether the latent factors share a covariance matrix. Next, the `fit.gppca()` function estimates the four sets of unknown parameters in the model. Finally, the `predict.gppca()` function offers the multi-step-ahead forecasts based on the estimated parameters.

The `gppca` function

We begin by introducing the `gppca()` function, which specifies the structure of the GPPCA model. Its first argument is a sorted n-dimensional vector whose elements represent one-dimensional inputs, such as spatial locations or time points. The second argument is the corresponding $k \times n$ observations matrix, where the ith column contains the observation at the ith input, for $i = 1, \ldots, n$. If the number of latent factors d is known, it should be passed as a positive integer to the `d` argument. Otherwise, users should set `est_d=TRUE` to estimate it from data. When the noise variance is known, d is selected via the variance matching approach in Equation (2.22). When it is un-

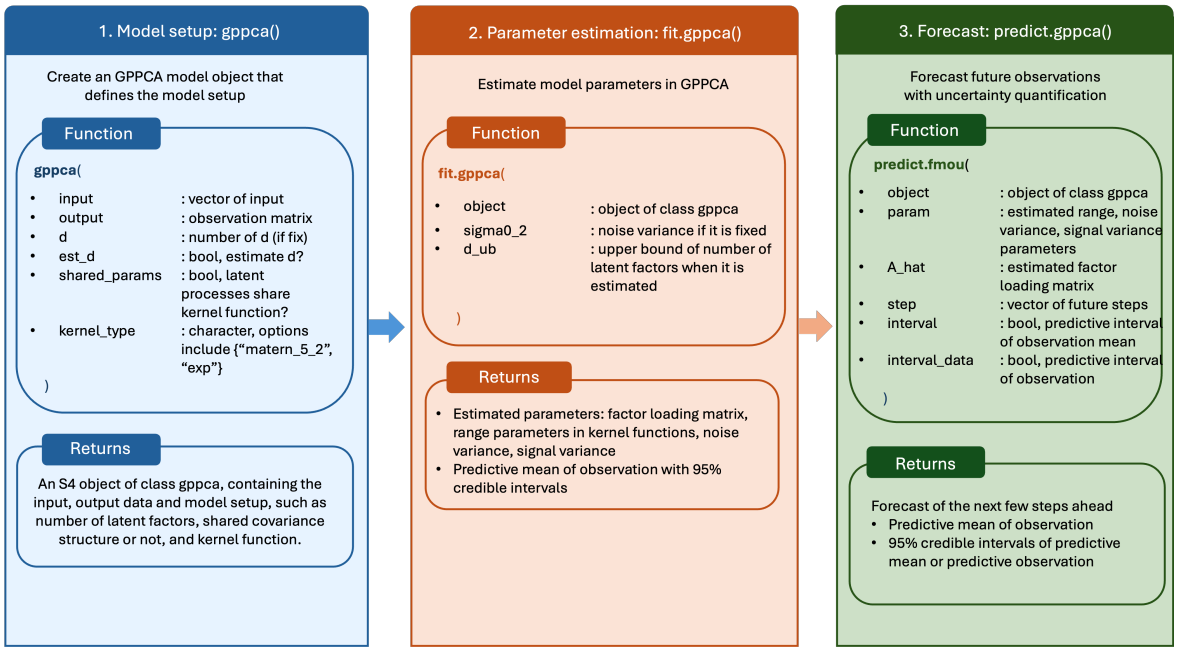

Figure 3.3: Workflow of GPPCA module in R package `FastGaSP`, illustrating the three main functions: `gppca()`, `fit.gppca()`, `predict.gppca()`, along with their inputs and outputs for model setup, parameter estimation, and forecasting with uncertainty quantification.

known, the information criterion in Equation (2.23) is used instead. In addition, the `shared_params` argument takes a boolean value indicating whether all latent factors share the same covariance matrix, i.e., $\mathbf{\Sigma}_1 = \cdots = \mathbf{\Sigma}_d = \mathbf{\Sigma}$. Finally, users must specify a kernel type, either the exponential kernel (`kernel_type="exp"`) or the Matérn 5/2 kernel (`kernel_type="matern_5_2"`). By default, users should assign a value to `d`, and the latent factor model has a shared covariance matrix constructed by Matérn 5/2 kernel.

The `gppca()` function returns an S4 object of class `gppca`, which contains the input vector, observation matrix, and the model setup, such as the number of latent factors, shared covariance matrix or not, and the kernel function. It serves as the first argument for the `fit.gppca()` and `predict.gppca()` functions.

The fit.gppca function

The fit.gppca() is the core function as it estimates all parameters in the GPPCA model. Its first argument is an S4 object created by the gppca() function. If est_d=TRUE and the noise variance is known, a positive scalar should be passed to the argument sigma0_2, which is then used in variance matching to determine the number of latent factors. When the information criterion is used, since it may tend to overestimate d, users can assign a positive integer to d_ub to impose an upper bound on the estimate.

Let $\mathbf{Y} = [\mathbf{y}(x_1), \ldots, \mathbf{y}(x_n)]$ denote the observation matrix, $\tau_l = \frac{\sigma_l^2}{\sigma_0^2}$ denote the signal-to-noise ratio (SNR) for the lth latent factor, with $\boldsymbol{\tau} := (\tau_1, \ldots, \tau_d)$, and $\beta_l = \frac{1}{\gamma_l}$ denote the inverse range parameter for the lth latent factor, with $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_d)$.

The GPPCA framework estimates the parameters $(\mathbf{A}, \boldsymbol{\beta}, \boldsymbol{\tau}, \sigma_0^2)$ through the maximum marginal likelihood estimator. The procedure begins by marginalizing out the latent factors to derive the marginal likelihood for the observation $\mathbf{Y}$. Based on this marginal likelihood, the maximum likelihood estimator for the factor loading $\mathbf{A}$ depends on whether the covariance matrix of the latent factors are shared. Under the orthonormality constraint, when all latent factors share a common covariance matrix (i.e., $\boldsymbol{\Sigma}_1 = \cdots = \boldsymbol{\Sigma}_d = \boldsymbol{\Sigma}$), the maximum marginal likelihood estimator $\hat{\mathbf{A}}$ has a closed-form solution, as shown in Theorem 2 of [19]:

$$\hat{\mathbf{A}} = \mathbf{U}_{\mathbf{G}} \bar{\mathbf{R}}, \tag{3.7}$$

where $\mathbf{U}_{\mathbf{G}}$ is a $k \times d$ matrix containing the first d eigenvectors of $\mathbf{G}_{\text{shared}} = \mathbf{Y}(\sigma_0^2 \boldsymbol{\Sigma}^{-1} + \mathbf{I}_n)^{-1} \mathbf{Y}^\top$, and $\bar{\mathbf{R}}$ is a $d \times d$ rotation matrix.

In the more general case where each latent factor has its own range and variance parameters, $\hat{\mathbf{A}}$ is obtained by solving an optimization problem with orthogonal constraints

on the Stiefel manifold [94], as shown in Theorem 3 of [19]:

$$\hat{\mathbf{A}} = \arg\max_{\mathbf{A}} \sum_{l=1}^{d} \mathbf{a}_l^\top \mathbf{G}_l \mathbf{a}_l, \quad \text{s.t. } \mathbf{A}^\top \mathbf{A} = \mathbf{I}_d, \tag{3.8}$$

where $\mathbf{G}_l = \mathbf{Y}(\sigma_0^2 \boldsymbol{\Sigma}_l^{-1} + \mathbf{I}_n)^{-1}\mathbf{Y}^\top$.

Given $\mathbf{A}, \boldsymbol{\tau}$ and $\boldsymbol{\beta}$, the maximum likelihood estimator of the noise variance σ_0^2 is expressed in closed form:

$$\hat{\sigma}_0^2 = \frac{S^2}{nk}, \tag{3.9}$$

where $S^2 = \mathrm{tr}(\mathbf{Y}^\top \mathbf{Y}) - \sum_{l=1}^{d} \mathbf{a}_l^\top \mathbf{Y}(\tau_l^{-1}\mathbf{R}_l + \mathbf{I}_n)^{-1}\mathbf{Y}^\top \mathbf{a}_l$.

Given $\hat{\mathbf{A}}$ and $\hat{\sigma}_0^2$, the SNR and range parameters $(\boldsymbol{\tau}, \boldsymbol{\beta})$ are estimated by maximizing the following objective function [19]:

$$(\hat{\boldsymbol{\tau}}, \hat{\boldsymbol{\beta}}) := \arg\max_{(\boldsymbol{\tau}, \boldsymbol{\beta})} \Big\{ \prod_{l=1}^{d} |\tau_l \mathbf{R}_l + \mathbf{I}_n|^{-\frac{1}{2}} \Big\} |S^2|^{-\frac{nk}{2}}. \tag{3.10}$$

In the implementation, we utilize the L-BFGS-B quasi-Newton algorithm to optimize Equation (3.10). Within each iteration of the $(\boldsymbol{\tau}, \boldsymbol{\beta})$ update, $\hat{\mathbf{A}}$ is re-estimated by Equation (3.7) or (3.8), depending on whether a shared covariance matrix is used. After obtaining $(\hat{\mathbf{A}}, \hat{\boldsymbol{\tau}}, \hat{\boldsymbol{\beta}})$, the estimated noise variance is obtained via Equation (3.9).

The `fit.gppca()` function returns the estimated parameters $(\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}}^2, \hat{\boldsymbol{\beta}}, \hat{\sigma}_0^2)$, which can be used for forecasting. The function also returns the predictive mean of the observations with 95% credible intervals.

The `predict.gppca` function

The `predict.gppca()` performs the forecasting task. It accepts the S4 object returned by the `gppca()` as its first argument, while `param` and `A_hat` takes $(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\sigma}}^2, \hat{\sigma}_0^2)$ and $\hat{\mathbf{A}}$, respectively. The `step` argument accepts a vector specifying the future steps.

For example, `step=c(1,1.2, 2)` returns predictive mean at $x = n+1, n+1.2, n+2$, where n is the number of observations used to fit the model. Uncertainty quantification is also provided. Setting `interval=T` and `interval_data=F` returns the 95% credible intervals for the predictive mean, whereas `interval=T` and `interval_data=T` return the 95% credible intervals for future observations. Both predictive mean and variance are computed efficiently via the `predict()` function from the `fgasp` module. By default, the function only returns the predictive mean of the new inputs.

3.2.2 Example of gppca module

In this section, we present an illustrative example of the `gppca` module using a simulated dataset. We generate the data with $n = 200$, $k = 3$ and $d = 2$, where the input is equally spaced over {1,2,...,n}, the noise variance is $\sigma_0^2 = 0.01$, and the Matérn 5/2 kernel is used for latent factors. We consider the distinct covariance setting with $\boldsymbol{\beta} = (0.01, 0.05)$ and $\boldsymbol{\sigma}^2 = (1, 1)$.

```
R> library(FastGaSP)
R> library(RobustGaSP)
R> library(rstiefel)
R> # simulate data
R> set.seed(2)
R> n = 200
R> k = 3
R> d = 2
R> beta_real=c(0.01, 0.05)
R> sigma_real=c(1,1)
R> sigma_0_real=sqrt(.01)
R> input=seq(1,n,1)
```

```
R> R0_00=as.matrix(abs(outer(input,input,'-')))
R> A=rustiefel(k, d)
R> Factor=matrix(0,d,n)
R> for(l in 1: d){
R>   L_sample <- t(chol(matern_5_2_funct(R0_00, beta_real[l])))
R>   Factor[l,]=sigma_real[l]^2*L_sample%*%rnorm(n)
R> }
R> true_mean <- A %*% Factor
R> output=true_mean + matrix(rnorm(k*n, sd=sigma_0_real),k,n)
```

We first call `gppca()` to specify the model inputs, outputs, and structure. Since the true number of latent factors is assumed known, we set `d=2`. Because each latent factor has a distinct covariance matrix, we have `shared_params=FALSE`. The function returns an object including both the dataset and the model configuration.

```
R> gppca_obj <- gppca(input, output, d=d, shared_params = FALSE)
R> str(gppca_obj)
Formal class 'gppca' [package "FastGaSP"] with 6 slots
  ..@ input        : num [1:200] 1 2 3 4 5 6 7 8 9 10 ...
  ..@ output       : num [1:3, 1:200] -0.148 -0.134 0.8203 -0.0552 0.1198 ...
  ..@ d            : num 2
  ..@ est_d        : logi FALSE
  ..@ shared_params: logi FALSE
  ..@ kernel_type  : chr "matern_5_2"
```

Parameter estimation is then implemented via `fit.gppca()`, which returns the estimated parameters $(\hat{\mathbf{A}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\sigma}}^2, \hat{\sigma}_0^2)$, the predictive mean of the observations, and the corresponding 95% credible intervals. Figure 3.4 visualizes the predictive mean (orange circle)

with the 95% intervals. The GPPCA model recovers the true mean accurately despite the presence of observation noise.

```
R> fit_obj <- fit.gppca(gppca_obj)
R> names(fit_obj)
[1] "est_A"          "est_beta"       "est_sigma0_2"   "est_sigma2"
[5] "mean_obs"       "mean_obs_95lb" "mean_obs_95ub"
```

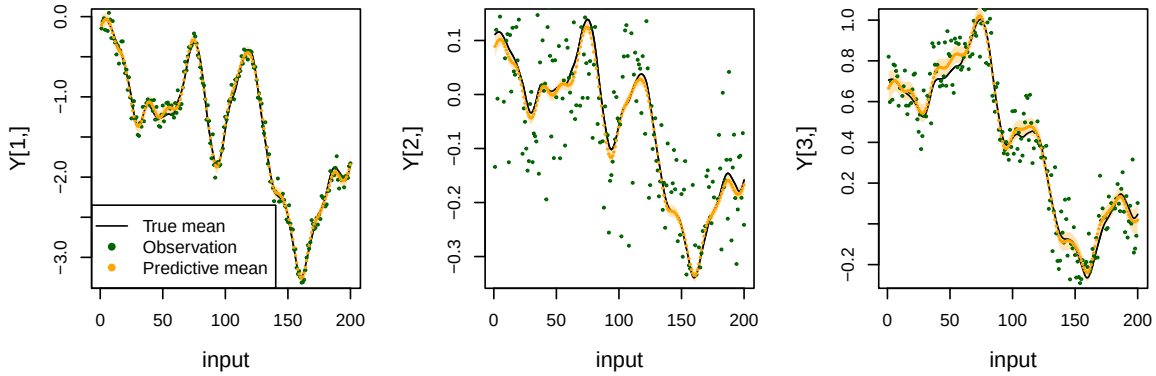

Figure 3.4: Illustrative example of results from `fit.gppca()`. Data are simulated with $n = 200$, $k = 3$, $d = 2$, and $\sigma_0^2 = 0.01$, where the latent factors have distinct Matérn 5/2 kernels with $\boldsymbol{\beta} = (0.01, 0.05)$ and $\boldsymbol{\sigma}^2 = (1, 1)$. The true mean is shown as a black curve, noisy observations as green circles, the GPPCA predictive mean as orange circles, and the shaded area as the 95% credible intervals.

Finally, `predict.gppca()` is used to forecast at future inputs $x = n+0.3, n+0.8, n+1, n+1.5$. Setting `interval=T` and `interval_Data=F` returns the predictive mean with its 95% credible intervals at each forecast location.

```
R> est_params <- c(fit_obj$est_beta, fit_obj$est_sigma2, fit_obj$est_sigma0_2)
R> forecast_obj <- predict.gppca(gppca_obj,
                    param = est_params, A_hat = fit_obj$est_A,
                    step = c(0.3, 0.8, 1,1.5),
                    interval=T, interval_data=F)
R> str(forecast_obj)
```

```
List of 3
 $ pred_mean         : num [1:3, 1:4] -1.8255 -0.1653 0.0199 -1.792 ...
 $ pred_interval_95lb: num [1:3, 1:4] -1.9728 -0.1868 -0.0665 -1.9677 ...
 $ pred_interval_95ub: num [1:3, 1:4] -1.678 -0.144 0.106 -1.616 ...
```

3.3 The fmou module

Figure 3.5 illustrates the workflow of the FMOU module in the FastGaSP R package, covering three stages: model setup, parameter estimation, and forecasting. The fmou() function defines the structure of the latent factor model, including the observation matrix and set of parameters to be estimated. The function returns an S4 object of class fmou, which served as the first input to the subsequent estimation and forecasting steps. The fit.fmou() accepts this fmou object and estimates the unknown model parameters via the EM algorithm, leveraging the KF and RTS smoother. Finally, the predict.fmou() function takes the fmou object together with the estimated parameters from fit.fmou() to produce multi-step-ahead forecasts.

3.3.1 Main functions of fmou module

The fmou function

The fmou() function initializes the FMOU model by constructing its setup from the observed data and user-specified model configurations. Specifically, the primary input output is a $k \times n$ observation matrix, where k denotes the output dimension and n the number of time steps. The factor loading matrix $\mathbf{U}_0$ and noise variance σ_0^2 can be either estimated or fixed. Setting est_U0=TRUE or est_sigma0_2=TRUE instructs the function to estimate these quantities. Otherwise, setting them to FALSE and providing a $k \times d$ orthogonal matrix to U0 or a nonnegative scalar to sigma0_2 fixes them at the specific

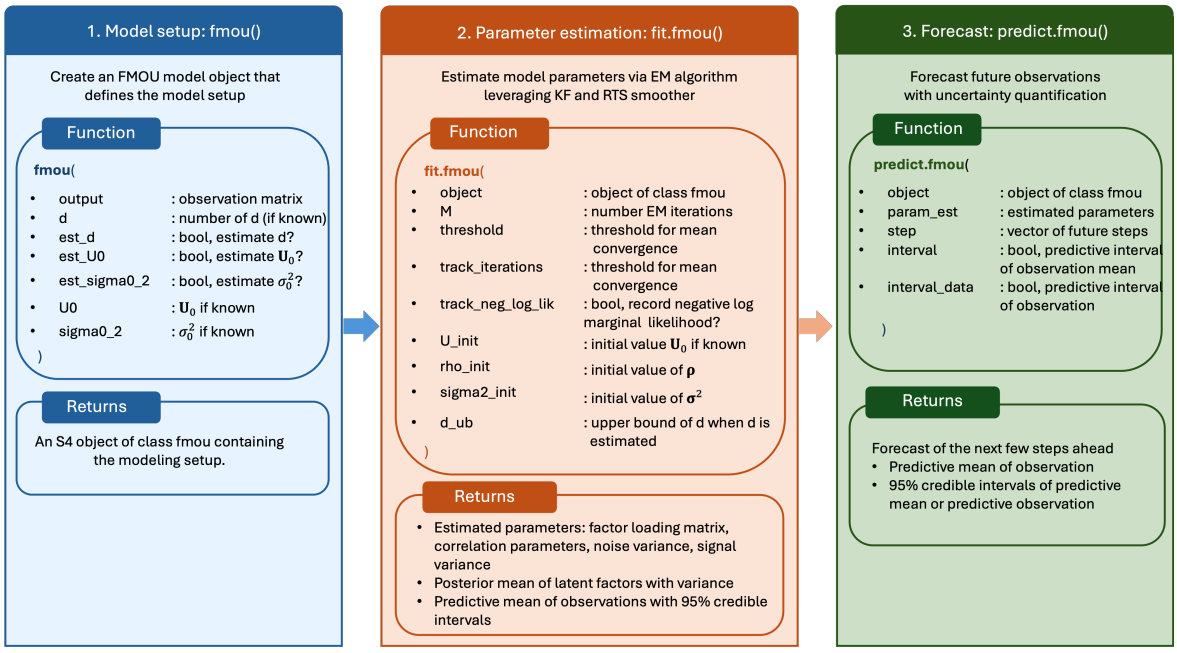

Figure 3.5: Workflow of FMOU module in R package `FastGaSP`, illustrating the three main functions: `fmou()`, `fit.fmou()`, `predict.fmou()`, along with their inputs and outputs for model setup, parameter estimation, and forecasting with uncertainty quantification.

values. The number of latent factors d is handled similarly. When `est_d=FALSE`, the value of `d` must be provided and is treated as fixed. When `est_d=FALSE`, the function estimates d using either the variance matching criterion in Equation (2.22) when the noise variance is known, or the information criterion in Equation (2.23) otherwise. The default setting of this function is to estimate all parameters $(\mathbf{U}_0, \boldsymbol{\rho}, \boldsymbol{\sigma}^2, \sigma_0^2)$ with a user-specific d.

The function returns an S4 object of class `fmou`, which served as the first input to the subsequent estimation and forecasting steps.

The `fit.fmou` function

The `fit.fmou()` is the core function of the FMOU module, implementing an efficient EM algorithm that leverages the KF and RTS smoother for fast parameter estimation

in latent factor models with high-dimensional noisy dynamic data. Its primary input is an object of class `fmou`, constructed via the `fmou()` function. The EM algorithm terminates under one of two stopping criteria: either the maximum number of iterations specified by `M` is reached, or the predictive mean of the observations converges within the tolerance specified by `threshold`. Users may provide initial values for model parameters, including the factor loading matrix, correlation parameters, and signal variance. In the absence of user-specified initialization, the factor loading matrix is initialized using the top d left singular vectors of the observation matrix, the correlation parameters are initialized as equally spaced values between 0.8 and 0.99, and the signal variances are initialized as equally spaced values between 0.5 and 1. In addition, the function provides options to record parameter estimates and the negative log marginal likelihood at each iteration, enabling detailed monitoring of the algorithm's convergence behavior. The default settings of this function are `M=50` and `threshold=1e-4`. By default, estimated parameters at each EM iteration and the negative logarithm marginal likelihood are not recorded to reduce storage and improve computational efficiency.

The `fit.fmou()` function returns the estimated model parameters $(\mathbf{U}_0, \boldsymbol{\sigma}^2, \boldsymbol{\rho}, \sigma_0^2)$, the posterior means and variances of the latent factors, and the predictive mean of the observations with 95% credible intervals.

The `predict.fmou` function

The `predict.fmou()` function generates multi-step-ahead forecasts using the fitted FMOU model. It takes as input an object of class `fmou` together with the estimated parameters obtained from `fit.fmou()`. Users specify the desired forecast horizon via the `step` argument. The function returns the predictive mean of future observations and optionally, corresponding 95% credible intervals. Specifically, setting `interval = TRUE` and `interval_data = FALSE` produces credible intervals for the predictive mean, while

setting `interval = TRUE` and `interval_data = TRUE` yields credible intervals for the observations.

3.3.2 Example of `fmou` module

In this section, we present an illustrative example of model initialization, parameter estimation, and forecasting using the FMOU module in the `FastGaSP` R package. The observation is a nuclei cell image [56, 106], represented as a 250×374 matrix. We use the first 370 time steps to fit the model and reserve the remaining 4 time steps for out-of-sample evaluation. The training data is shown in Figure 3.6 (a).

Below we provide example code for initializing the model. In this setup, we use the default settings where all parameters $(\mathbf{U}_0, \boldsymbol{\rho}, \boldsymbol{\sigma}^2, \sigma_0^2)$ are estimated. Besides, the number of latent factors d is estimated. Since the noise variance is unknown, the IC criterion in Equation (2.23) is used to select d.

```
R> library(FastGaSP)
R> library(plot3D)
# initialize the setup of FMOU model, estimate all parameters
R> fmou_obj <- fmou(nuclei_figure[,1:370], est_d = T)
```

Next, the model parameters are estimated using the `fit.fmou()` function. As shown in the code, we limit the EM algorithm to a maximum of 20 iterations (`M=20`) and set `threshold`$=10^{-6}$ as the convergence criterion for the predictive mean. Besides, we restrict the upper bound of d by setting `d_ub=nrow(nuclei_figure)*2/3`. Finally, we record the negative log-likelihood at each EM iteration by setting `track_neg_log_lik = TRUE`. The estimated parameters are returned in the function output. For this nuclei cell example, the number of latent factors is selected as $d = 42$ based on the information criterion, and the corresponding correlation and variance parameters are obtained accord-

ingly. Figure 3.6 (b) and (c) visualize the predictive mean and negative log-likelihood from the function output.

```
# estimate the parameters
R> fmou_fit <- fit.fmou(fmou_obj, M=20, threshold=10^{-4},
                    d_ub=nrow(nuclei_figure)*2/3, track_neg_log_lik = T)
R> str(fmou_fit)
List of 14
 $ output            : num [1:250, 1:370] 0.471 0.459 0.478 0.478 ...
 $ d                 : int 42
 $ U                 : num [1:250, 1:42] -0.0592 -0.0591 -0.0591 -0.0592 ...
 $ post_z_mean       : num [1:42, 1:370] -7.8684 0.0133 -0.0912 -0.1968 ...
 $ post_z_var        : num [1:42, 1:370] 0.000237 0.000244 0.000244  ...
 $ post_z_cov        : num [1:42, 1:369] 1.69e-05 9.62e-06 1.05e-05 ...
 $ mean_obs          : num [1:250, 1:370] 0.493 0.491 0.484 0.479 ...
 $ mean_obs_95lb     : num [1:250, 1:370] 0.487 0.486 0.48 0.475 ...
 $ mean_obs_95ub     : num [1:250, 1:370] 0.498 0.496 0.488 0.484 ...
 $ sigma0_2          : num 0.000255
 $ rho               : num [1:42] 1 0.974 0.976 0.969 0.974 ...
 $ sigma2            : num [1:42] 0.00308 0.00582 0.00529 0.0058 ...
 $ num_iterations    : int 8
 $ record_neg_log_lik: num [1:8] 191255 178181 166445 163926 163547 ...
```

Finally, we use the `predict.fmou()` function to generate four-step-ahead forecasts, along with 95% credible intervals for the observations. As shown in the code below, the resulting credible intervals achieve a coverage rate of 95.1% on the hold-out data.

```
# forecast the last 4 steps with 95% intervals of future obs
R> fmou_forecast <- predict.fmou(fmou_obj,fmou_fit, step=1:4,
```

```
                                interval=T, interval_data=T)
# compute the coverage
R> mean((fmou_forecast$pred_interval_95lb<nuclei_figure[,371:374]) &
       (fmou_forecast$pred_interval_95ub>nuclei_figure[,371:374]))
[1] 0.951
```

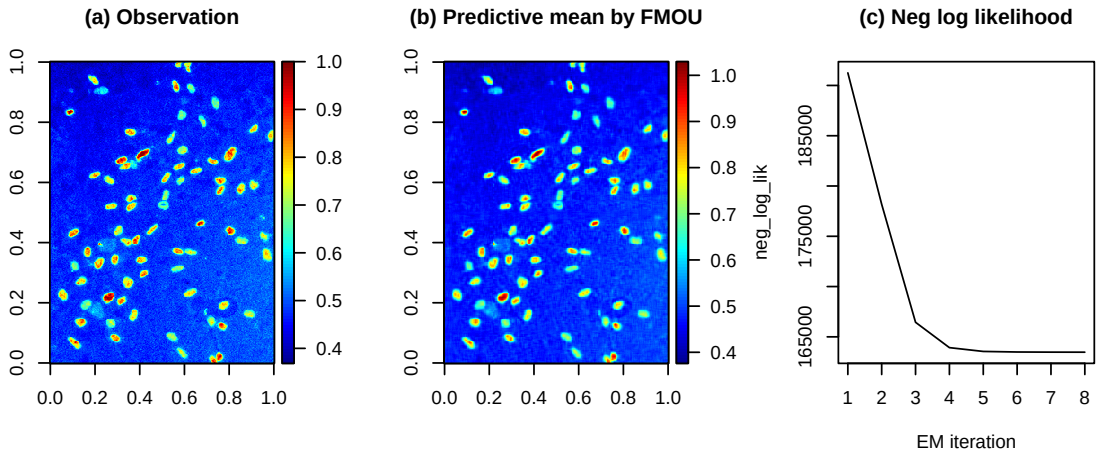

Figure 3.6: (a) Image of nuclei cells. (b) Predictive mean by FMOU. (c) Negative logarithm marginal likelihood v.s. EM iterations.

Chapter 4

Probabilistic forecast of nonlinear dynamical systems with uncertainty quantification

This chapter is based on material from: Gu, M., Lin, Y., Lee, V. C., & Qiu, D. Y. (2024). Probabilistic forecast of nonlinear dynamical systems with uncertainty quantification. *Physica D: Nonlinear Phenomena*, 457, 133938.

Data-driven modeling is useful for reconstructing nonlinear dynamical systems when the underlying process is unknown or too expensive to compute. Having a reliable uncertainty assessment of the forecast enables tools to be deployed to predict new scenarios that have not been observed before. In this chapter, we first extend parallel partial Gaussian processes for predicting the vector-valued transition function that links the observations between the current and next time points, and quantify the uncertainty of predictions by posterior sampling. Second, we show the equivalence between the dynamic mode decomposition and the maximum likelihood estimator of the linear mapping matrix in the linear state space model. The connection provides a probabilistic generative model

of dynamic mode decomposition and thus, uncertainty of predictions can be obtained. Furthermore, we draw close connections between different data-driven models for approximating nonlinear dynamics through a unified view of generative models. We study two numerical examples, where the inputs of the dynamics are assumed to be known in the first example and the inputs are unknown in the second example. The examples indicate that uncertainty of forecast can be properly quantified, whereas model or input misspecification can degrade the accuracy of uncertainty quantification.

4.1 Introduction

Dynamical systems are ubiquitously used for describing natural phenomena, such as passive motions driven by thermodynamics [107] and phase transition from flocking [108, 109], and social behaviors, such as epidemiological processes [110]. As mathematical models typically contain unknown parameters, observations are often used for calibrating the models and filtering the noise to estimate the latent state of the dynamical system. Kalman filter [25] and Rauch–Tung–Striebel smoother [26], for instance, produce fast estimation of latent states for linear dynamical systems with additive Gaussian fluctuations and noises, at a computational cost linearly increasing with the number of time points. When dynamical systems are nonlinear or non-Gaussian, approximate approaches, such as the extended Kalman filter [111], particle filters [112] and ensemble Kalman filter [113], were developed to approximate the posterior distributions of latent states. However, these approaches require underlying data-generating models to be known, whereas models that exactly reproduce the reality may be unavailable or too costly to compute in some applications.

Data-driven approaches become useful for estimating dynamical systems when the true data-generating mechanism is unknown. For instance, an orthogonal basis is esti-

mated in proper orthogonal decomposition [114, 61] to reconstruct the covariance between each of the output coordinates by treating temporal observations as independent measurements. Dynamic mode decomposition [28, 29] reconstructs the output vector through linearizing the one-step-ahead transition operator between the input and output pairs, where the eigenpairs of the linear mapping matrix produce a finite-dimensional approximation of the Koopman modes and eigenvalues [76, 115]. Extensive variants of the Koopman operator have been proposed, such as utilizing longer temporal lag of observations through the Hankel method or higher order dynamic mode decomposition [116], and utilizing nonlinear basis functions for lifting the process by the extended dynamic mode decomposition [117]. A few recent techniques, such as sparse regression [118, 119], model predictive control [120], and Koopman eigenfunctions [121], were studied for designing the nonlinear basis and estimating the lifted state in extended dynamic mode decomposition. Other nonlinear methods employ neural networks to model differential equations [122, 123, 124]. The uncertainty of the estimation by the dynamic mode decomposition and other machine learning approaches, however, may not be available, as the generative models were not well-studied.

Deploying data-driven models to forecast or extrapolate the input space requires reliable uncertainty assessments of predictions. We evaluate the precision of uncertainty quantification by the percentage of held-out observations covered by the $1-\alpha$ predictive intervals, with $0 < \alpha < 1$. An efficient approach should have around $1-\alpha$ of the held-out samples covered by a short predictive interval. Gaussian processes have been applied in emulating expensive computer simulations [11, 12, 13], and calibrating computer models [14, 15, 16]. However, uncertainty assessment of these approaches is typically assessed for interpolating physical input space, while forecast or extrapolation is required in various real-world applications, such as optimizing chemical reaction conditions through Bayesian optimization [125] and controlling the predictive error by active learning [126].

The goal of this paper is to quantify the uncertainty from probabilistic forecasts by different approaches. Our contributions are threefold. First, we extend a recent approach, called the parallel partial Gaussian process (PP-GP) [27], to forecast dynamical systems with multivariate outputs through predicting the one-step-ahead transition function, and the uncertainty of the forecast can be assessed through posterior sampling. Predicting the one-step-ahead transition function with a finite-dimensional space allows one to transform the forecast problem to an interpolation problem, and under regularity conditions, Gaussian process regression converges to the truth with a minimax rate [127]. Furthermore, the assessed uncertainty from the PP-GP can alert when the forecast becomes less accurate, which allows for timely interventions to control predictive error. The PP-GP is particularly suitable for dynamical systems with massive outputs, as the computational complexity of the PP-GP is linear to the number of outputs at each time point.

Second, we introduce a general two-step approach to derive a probabilistic generative model. Based on this approach, we show that the estimation of dynamic mode decomposition is equivalent to the maximum likelihood estimator of a linear mapping matrix in a linear state space model, which produces uncertainty assessment from the sampling model. Third, we draw connections between different approaches, including Gaussian processes, proper orthogonal decomposition, and dynamic mode decomposition. These connections allow one to examine the inherent generative models of different approaches, and to develop a suitable predictive model for real-world problems.

We compare the approaches for forecasting and uncertainty quantification by two numerical examples. In the first example, we assume the inputs of the process are known, whereas we do not have prior knowledge of the functional form of the process. Hence, we cannot use the exact form of the function to form the nonlinear basis. Rather, we aim to test default or generic kernels or bases, which can be used for other scenarios. We test this scenario by the Lorenz 96 system [6], a benchmark approach of modeling

atmospheric quantities at equally spaced locations along a cycle that induces chaotic behaviors. The GPs can detect the time when the predictive error becomes large, based on their internal uncertainty assessment of the forecast.

In the second example, we do not assume the true inputs are known, which is unconventional in designing data-driven approaches, but not uncommon in practice [128]. We consider one of the most challenging problems in condensed matter physics: simulating quantum many-body systems far from equilibrium. Many problems in quantum dynamics, such as the motion of atoms or charge carriers, cannot be modeled by well-established equilibrium methods, including density functional theory (DFT) for the electronic ground state or the GW plus Bethe–Salpeter equation [129, 130] which describes excited-state properties in the equilibrium linear response regime. This limits, for example, the understanding of systems under irradiation by ultrafast or intense laser pulses for obtaining information about the electronic structure of a system [131]. Therefore, nonequilibrium simulations that describe how a system responds to an external perturbation and how it evolves from one configuration to another under these circumstances are crucial for a complete understanding of the electronic and optical properties of molecules and solids.

A rigorous approach to simulating materials' nonequilibrium dynamics lies in propagating the nonequilibrium Green's function as a two-point correlator of the creation and annihilation field operators on the Keldysh contour [3, 4, 5]. This approach has been recently applied to compute various nonlinear and nonequilibrium optical responses from first principles in the adiabatic limit, which limits the time-evolution to a single average time, neglecting memory effects [132, 133, 134, 135, 136]. However, even in the adiabatic approximation, the numerical evaluation is far from trivial, requiring millions of CPU hours for systems of only a few atoms. Thus, models that can forecast the time evolution without the need for the simulator are urgently needed. Recent work [137, 138] uses dynamic mode decomposition to approximate the Green's functions where the system is

assumed to start from a known non-interacting state, and it is driven by an arbitrary external electromagnetic field. Different representations of the Green's function encode spectroscopic information, which may be measured in experiments. In this work, inputs such as the external field and many-electron interactions are not used in constructing data-driven models to test the uncertainty quantification of the forecast for the scenario when inputs are misspecified.

This chapter is organized as follows. In Section 4.2, we extended the PP-GP for forecasting nonlinear dynamical processes. We introduce a general approach to derive a generative model, and apply this approach to derive a sampling model of the dynamic mode decomposition and its predictive distribution in Section 4.2.3. PP-GP, dynamic mode decomposition, and proper orthogonal decomposition are compared in Section 4.3, focusing on generative models of these approaches. In Section 4.4, we numerically compare different approaches for forecasting, and discuss the scenarios when a reliable forecast can be constructed even at a reasonably long trajectory. The data and code used in this work are publicly available (`https://github.com/UncertaintyQuantification/forecast_dynamical_systems`).

4.2 Probabilistic forecast and uncertainty quantification by parallel partial Gaussian processes

We review the parallel partial Gaussian processes (PP-GP) in Section 1.1, including the model assumption, parameter estimation, and predictive distribution of testing input. Building on this, in this section, we first interpret the predictive mean in Equation (1.20) as weighted averages of basis functions and output vectors. We then extend the framework to time series forecasting via posterior sampling. In the rest of this chapter,

we assume the observations are time-dependent, meaning that the tth output vector is denoted as $\mathbf{y}(\mathbf{x}_t) \in \mathbb{R}^k$, where $\mathbf{x}_t$ is a p_x-dimensional input vector containing the observations in the prior time points and additional physics input, for $t = 1, 2, \ldots, n$. When the output vector in the previous time point is used as input, we have $\mathbf{x}_t = \mathbf{y}(\mathbf{x}_{t-1})$ and thus $p_x = k$. In general, the dimensions of the input and output vectors can be different. Furthermore, we assume a constant mean in Gaussian processes, i.e., $\mathbf{h}(\mathbf{x}_t) = 1$, and thus the mean for each output coordinate is denoted as μ_j, for $j = 1, \ldots, k$.

4.2.1 Predictions as weighted averages of basis functions and output vectors

The predictive mean or median $\hat{y}_j(\mathbf{x}_{t^*})$ is often used for one-step-ahead prediction of output coordinate j for any test input $\mathbf{x}_{t^*}$, for $j = 1, ..., k$. The corollary below shows that the prediction of the PP-GP model can be written as a weighted average of the observations and kernel functions.

Corollary 1 *The predictive mean vector* $\hat{\mathbf{y}}(\mathbf{x}_{t^*}) = (\hat{y}_1(\mathbf{x}_{t^*}), ..., \hat{y}_k(\mathbf{x}_{t^*}))^\top$ *from Equation (1.20) follows*

$$\hat{\mathbf{y}}(\mathbf{x}_{t^*}) = \mathbf{Y}\mathbf{v}^\top = \hat{\boldsymbol{\mu}} + \mathbf{W}\mathbf{r}(\mathbf{x}_{t^*}), \tag{4.1}$$

where $\hat{\boldsymbol{\mu}} = [\hat{\mu}_1, \ldots, \hat{\mu}_k]^\top$ *is the mean vector,* $\mathbf{v}$ *is an* n*-dimensional row vector and* $\mathbf{W} = [\mathbf{w}_1^\top, ..., \mathbf{w}_k^\top]^\top$ *is a* $k \times n$ *matrix with* $\mathbf{w}_j$ *being an* n*-dimensional row vector defined below:*

$$\mathbf{v} = \frac{(1 - \mathbf{r}^\top(\mathbf{x}_{t^*})\tilde{\mathbf{R}}^{-1}\mathbf{1}_n)\mathbf{1}_n^\top\tilde{\mathbf{R}}^{-1}}{\left(\mathbf{1}_n^\top\tilde{\mathbf{R}}^{-1}\mathbf{1}_n\right)} + \mathbf{r}^\top(\mathbf{x}_{t^*})\tilde{\mathbf{R}}^{-1}, \tag{4.2}$$

$$\mathbf{w}_j = (\mathbf{y}_j - \hat{\mu}_j\mathbf{1}_n)^\top\tilde{\mathbf{R}}^{-1}, \tag{4.3}$$

for $j = 1, ..., k$.

From Corollary 1, the prediction of a testing input at the jth coordinate of the output from the PP-GP model can be written as a weighted average of the observations at the jth coordinate, $\hat{y}_j(\mathbf{x}_{t^*}) = \mathbf{v}\mathbf{y}_j$. Furthermore, the residuals can be written as a weighted average of the kernel function between the test input and training input set, $\hat{y}_j(\mathbf{x}_{t^*}) - \hat{\mu}_j = \mathbf{w}_j\mathbf{r}(\mathbf{x}_{t^*})$, as outlined by the second equality in Equation (4.1). When $\hat{\mu}_j = 0$, the predictive mean estimator in Equation (1.20) at each output coordinate is equivalent to the kernel ridge regression separately for each coordinate j [139]:

$$\hat{y}_j(\cdot) = \text{argmin}_{f_j \in \mathcal{H}} \left\{ \frac{1}{n} \sum_{t=1}^{n} (y_j(\mathbf{x}_t) - f_j(\mathbf{x}_t))^2 + \frac{\eta}{n} ||f_j||^2_{\mathcal{H}} \right\}, \tag{4.4}$$

where $\mathcal{H}$ is the reproducing kernel Hilbert space (RKHS) [140] attached to the kernel $K(\cdot, \cdot)$ and $|| \cdot ||_{\mathcal{H}}$ is the associated native norm. The loss function in Equation (4.4) penalizes the fitting error and complexity of the model simultaneously, which helps avoid the overfitting problem automatically. Compared to the kernel ridge regression in Equation (4.4), the uncertainty of the prediction of PP-GP can be quantified based on the predictive distribution in Equation (1.19).

Here the uncertainty of the mean and variance parameters is taken into account in the analysis, whereas the uncertainty in estimating the range and nugget parameters was not considered due to computational feasibility, and confounding issues between kernel parameters [141]. Sampling the parameters from the posterior distribution or the residual bootstrap approach [142] can be used for estimating the uncertainty of these kernel parameters.

4.2.2 Forecast by parallel partial Gaussian processes

Here we focus on estimating the one-step-ahead transition function that maps the input $\mathbf{x}_t$ to the output $\mathbf{y}(\mathbf{x}_t)$ at any time point t. Consider a simple scenario, where the

previous snapshot is used for predicting the output at the current time step: $\mathbf{x}_t = \mathbf{y}(\mathbf{x}_{t-1})$ for any $t \geq 1$. Since the function is nonlinear, we may iteratively use the predictive distribution to sample M_c chains for forecasting from $t^* = n+1, ..., n+n^*$. Let the input be $\mathbf{x}_{n+1}^{(m_c)} = \mathbf{y}(\mathbf{x}_n)$ for any chain m_c, $m_c = 1, ..., M_c$. For each of the chain m_c, we simulate a new output from the predictive distribution sequentially for $t^* = n+1, ..., n+n^*$:

$$\mathbf{y}^{(m_c)}(\mathbf{x}_{t^*+1}^{(m_c)}) \sim p(\mathbf{y}^{(m_c)}(\mathbf{x}_{t^*+1}^{(m_c)}) \mid \mathbf{Y}, \mathbf{X}, \mathbf{x}_{t^*+1}^{(m_c)}, \boldsymbol{\gamma}, \eta), \tag{4.5}$$

where $p(\mathbf{y}^{(m_c)}(\mathbf{x}_{t^*+1}^{(m_c)}) \mid \mathbf{Y}, \mathbf{X}, \mathbf{x}_{t^*+1}^{(m_c)}, \boldsymbol{\gamma}, \eta)$ is the predictive distribution of $\mathbf{y}^{(m_c)}(\mathbf{x}_{t^*+1})$.

As directly sampling from the joint predictive distribution at all output coordinates can be computationally intensive, one may sample the jth coordinate of the output from the marginal predictive distribution $p(y_j^{(m_c)}(\mathbf{x}_{t^*+1}) \mid \mathbf{Y}, \mathbf{X}, \mathbf{x}_{t^*+1}^{(m_c)}, \boldsymbol{\gamma}, \eta)$ in Equation (1.19) as an approximation. After we obtain predictive samples $y_{t^*,j}^{(m_c)}$ for $m_c = 1, ..., M_c$, we can use mean or median for predictions, and the lower and upper α quantiles of the samples for constructing the $1-\alpha$ predictive interval for any $0 < \alpha < 1$. Furthermore, the predictive mean $\hat{\mathbf{y}}(\mathbf{x}_{t^*+1})$ may be approximated by using a plug-in estimator of the input $\mathbf{x}_{t^*+1} \approx \hat{\mathbf{y}}(\mathbf{x}_{t^*})$ by Equation (1.20). The assessed uncertainty from PP-GP can alert when the forecast becomes less accurate, which allows for timely interventions to control the predictive error, whereas the uncertainty assessment may not be available in some other machine learning approaches [118, 122, 123].

Here we transform the forecast problem to predict the one-step-ahead transition function with a finite-dimensional input space. Under regularity conditions, the minimax convergence rate of Gaussian process regression to the truth was studied [127]. We should note that the convergence is obtained when the training inputs fill a bounded input domain, whereas sequentially sampled inputs in the forecast could move outside the training input domain in practice. When the inputs approach the boundary of the

input space, the quantified uncertainty becomes large, which can give an alert for refining the prediction by expanding the input domain and training the PP-GP emulator with inputs in the expanded input domain. It is of interest to study the convergence of the PP-GP forecast in different dynamical systems.

4.2.3 Computational complexity

Compared with other GP approaches for emulating vectorized output, one advantage of the PP-GP model comes with the computational scalability when the number of output coordinates k is large. Computing the predictive mean of k output coordinates in Equation (1.20) requires $\mathcal{O}(nk) + \mathcal{O}(n^3)$ operations, and to obtain M_c predictive samples of k output vectors at n^* time points for uncertainty quantification requires $\mathcal{O}(n^2 n^* M_c) + \mathcal{O}(n^2 k)$ operations. The highest cost of PP-GP typically comes from estimating the range and nugget parameters, which requires $\mathcal{O}(M_{pp} n^3 + M_{pp} n^2 k)$ operations for M_{pp} iterations in numerical optimization. When the number of time points n is large, approximation methods, such as the inducing point method [46] and the Vecchia approach [47], may be used for approximating the likelihood function of Gaussian processes.

The computational advantage of PP-GP comes from two assumptions. First, the outputs at different coordinates are assumed to be independent. In Theorem 1 in [27], the authors show that the predictive mean of PP-GP is exactly the same as the predictive mean of a separable Gaussian process of the vectorized output, with the covariance $\mathbf{\Sigma}_y \otimes \mathbf{R}$, where $\mathbf{\Sigma}_y$ is the covariance of output at different coordinates, and the variance between the two models is similar. The inverse of covariance between output coordinates $\mathbf{\Sigma}_y$ generally takes $\mathcal{O}(k^3)$ operations in computing the likelihood function of a separable Gaussian process, whereas the complexity of predictions by PP-GP is linear to the number of coordinates k. Second, the correlation of the output at different inputs $\mathbf{x}$ is

shared across output coordinates. Allowing the correlation parameters to differ at each spatial coordinate makes the computational complexity become $\mathcal{O}(n^3k)$ for computing the predictive mean, which is higher than $\mathcal{O}(nk) + \mathcal{O}(n^3)$. Furthermore, separably esti

Mathematical and machine learning approaches often minimize a loss function for estimating parameters. Many loss-minimization approaches can be shown to be equivalent to statistical estimations from probabilistic generative models. We summarize a two-step procedure of building a generative model.

1. Build a probabilistic generative model of untransformed data with parameters that can be identified from data. Generalize the model as much as possible to the extent that the parameters can still be identified from the data.

2. Show equivalence between the estimator that minimizes a loss function and a statistical estimator, such as the maximum likelihood estimator or maximum marginal likelihood estimator, of the probabilistic generative model.

The generative model indicates the underlying assumptions of the loss-minimization estimator. A more efficient estimator can be derived based on the generative model if the loss-minimization estimator is not optimal. We follow this procedure to derive a generative model of the dynamic mode decomposition that enables uncertainty assessment of the estimation.

4.2.4 Dynamic mode decomposition

Dynamic mode decomposition (DMD) is a data-driven approach to obtain a reduced rank representation of data from high-dimensional dynamical systems [28], which is reviewed in Section 1.5.

We first introduce the lemma that connects the DMD estimation to the maximum

likelihood estimator (MLE) of the linear mapping matrix in a dynamic linear model [66] or linear state space model [143].

Lemma 8 *The DMD estimator* $\hat{\mathbf{A}}$ *in Equation (1.45) is the MLE of* $\mathbf{A}$ *of the following linear state space model*

$$\mathbf{y}(\mathbf{x}_{t+1}) = \mathbf{A}\mathbf{y}(\mathbf{x}_t) + \boldsymbol{\varepsilon}_{t+1}, \tag{4.6}$$

where $\boldsymbol{\varepsilon}_{t+1} \sim \mathcal{MN}(\mathbf{0}, \boldsymbol{\Sigma}_\varepsilon)$ *is a vector of Gaussian distributions with a positive definite covariance matrix* $\boldsymbol{\Sigma}_\varepsilon$*, for any* $t = 1, 2, ...n$ *and we assume the marginal distribution of initial state* $\mathbf{y}(\mathbf{x}_1)$ *does not depend on* $\mathbf{A}$*. We refer the linear state model by Equation (4.6) the DMD-induced process.*

Proof: The log-likelihood of $\mathbf{A}$ and $\boldsymbol{\Sigma}_\varepsilon$ is:

$$\begin{aligned}
\mathcal{L}(\mathbf{A}, \boldsymbol{\Sigma}_\varepsilon) &= \log\left\{ p(\mathbf{y}(\mathbf{x}_1)) \prod_{t=2}^{n} p(\mathbf{y}(\mathbf{x}_t) \mid \mathbf{y}(\mathbf{x}_{t-1}), \mathbf{A}, \boldsymbol{\Sigma}_\varepsilon) \right\} \\
&\propto -\frac{n}{2}\log(|\boldsymbol{\Sigma}_\varepsilon|) - \frac{\mathbf{y}(\mathbf{x}_1)^\top \boldsymbol{\Sigma}_\varepsilon^{-1} \mathbf{y}(\mathbf{x}_1)}{2} \\
&\quad - \frac{\sum_{t=2}^{n} (\mathbf{y}(\mathbf{x}_t) - \mathbf{A}\mathbf{y}(\mathbf{x}_{t-1}))^\top \boldsymbol{\Sigma}_\varepsilon^{-1} (\mathbf{y}(\mathbf{x}_t) - \mathbf{A}\mathbf{y}(\mathbf{x}_{t-1}))}{2}.
\end{aligned}$$

Taking the derivative of log-likelihood with respect to $\mathbf{A}$ and $\boldsymbol{\Sigma}_\varepsilon$, we obtain the maximum likelihood estimator of $\mathbf{A}$ and $\boldsymbol{\Sigma}_\varepsilon$:

$$\begin{aligned}
\hat{\mathbf{A}} &= \left(\sum_{t=2}^{n} \mathbf{y}(\mathbf{x}_t) \mathbf{y}(\mathbf{x}_{t-1})^\top \right) \left(\sum_{t=2}^{n} \mathbf{y}(\mathbf{x}_{t-1}) \mathbf{y}(\mathbf{x}_{t-1})^\top \right)^+ \\
&= \mathbf{Y}_2 \mathbf{Y}_1^\top (\mathbf{Y}_1 \mathbf{Y}_1^\top)^+ \\
&= \mathbf{Y}_2 (\mathbf{Y}_1)^+, \\
\hat{\boldsymbol{\Sigma}}_\varepsilon &= \frac{\mathbf{y}(\mathbf{x}_1)\mathbf{y}(\mathbf{x}_1)^\top + \sum_{t=2}^{n} (\mathbf{y}(\mathbf{x}_t) - \hat{\mathbf{A}}\mathbf{y}(\mathbf{x}_{t-1}))(\mathbf{y}(\mathbf{x}_t) - \hat{\mathbf{A}}\mathbf{y}(\mathbf{x}_{t-1}))^\top}{n}.
\end{aligned}$$

The last equality in the equation of $\hat{\mathbf{A}}$ can be derived by performing the singular value

decomposition (SVD) on $\mathbf{Y}_1$ and connecting the SVD with the Moore–Penrose pseudo-inverse. ■

We notice that the maximum likelihood estimator of $\mathbf{A}$ is equivalent to the solution of DMD shown in Equation (1.45). Here $\mathbf{Y}_1^+$ can be computed by the SVD of $\mathbf{Y}_1 = \mathbf{U}_1\mathbf{D}_1\mathbf{V}_1^\top$ as $\mathbf{Y}_1^+ = \mathbf{V}_1\mathbf{D}_1^{-1}\mathbf{U}_1^\top$, where $\mathbf{D}_1 \in \mathbb{R}^{k\times(n-1)}$ is a rectangular diagonal matrix of non-negative singular values.

From the DMD-induced process in Equation (4.6), we have the following lemma, which gives the posterior distribution for the forecast.

Lemma 9 *Conditional on the observations* $\mathbf{Y}$ *with plug-in estimators of* $\hat{\mathbf{A}}$ *and* $\hat{\mathbf{\Sigma}}_\varepsilon$*, the posterior distribution of the output vector of DMD-induced process in Equation (4.6) at any* $\mathbf{x}_{t^*}$ *follows a multivariate normal distribution*

$$\left(\mathbf{y}(\mathbf{x}_{t^*}) \mid \mathbf{Y}, \hat{\mathbf{A}}, \hat{\mathbf{\Sigma}}_\varepsilon\right) \sim \mathcal{MN}\left(\hat{\mathbf{y}}(\mathbf{x}_{t^*}), \sum_{i^*=0}^{t^*-n-1} \hat{\mathbf{A}}^{i^*}\hat{\mathbf{\Sigma}}_\varepsilon(\hat{\mathbf{A}}^\top)^{i^*}\right), \tag{4.7}$$

where $\hat{\mathbf{y}}(\mathbf{x}_{t^*})$ *follows Equation (1.50) for any* $t^* > n$*.*

Proof: We prove this by induction. For $t^* = n+1$, we have

$$\begin{aligned}
\mathbb{E}[\mathbf{y}(\mathbf{x}_{t^*}) \mid \mathbf{Y}, \hat{\mathbf{A}}, \hat{\mathbf{\Sigma}}_\varepsilon] &= \hat{\mathbf{A}}\mathbf{y}(\mathbf{x}_n),\\
\mathbb{V}[\mathbf{y}(\mathbf{x}_{t^*}) \mid \mathbf{Y}, \hat{\mathbf{A}}, \hat{\mathbf{\Sigma}}_\varepsilon] &= \hat{\mathbf{\Sigma}}_\varepsilon.
\end{aligned}$$

Assume for $t^* > n+1$, we have

$$\begin{aligned}
\mathbb{E}[\mathbf{y}(\mathbf{x}_{t^*}) \mid \mathbf{Y}, \hat{\mathbf{A}}, \hat{\mathbf{\Sigma}}_\varepsilon] &= \hat{\mathbf{A}}^{t^*-n}\mathbf{y}(\mathbf{x}_n),\\
\mathbb{V}[\mathbf{y}(\mathbf{x}_{t^*}) \mid \mathbf{Y}, \hat{\mathbf{A}}, \hat{\mathbf{\Sigma}}_\varepsilon] &= \sum_{i^*=0}^{t^*-n-1} \hat{\mathbf{A}}^{i^*}\hat{\mathbf{\Sigma}}_\varepsilon(\hat{\mathbf{A}}^\top)^{i^*}.
\end{aligned}$$

For any t^*+1, by the sampling model in Equation (4.6) and the law of total expec-

tation, the posterior mean follows:

$$\begin{aligned}\mathbb{E}[\mathbf{y}(\mathbf{x}_{t^*+1}) \mid \mathbf{Y}, \hat{\mathbf{A}}, \hat{\boldsymbol{\Sigma}}_\varepsilon] =& \mathbb{E}[\mathbb{E}[\mathbf{y}(\mathbf{x}_{t^*+1}) \mid \mathbf{Y}, \mathbf{y}(\mathbf{x}_{t^*}), \hat{\mathbf{A}}, \hat{\boldsymbol{\Sigma}}_\varepsilon]] \\ =& \mathbb{E}[\hat{\mathbf{A}}\mathbf{y}(\mathbf{x}_{t^*}) \mid \mathbf{Y}, \hat{\mathbf{A}}, \hat{\boldsymbol{\Sigma}}_\varepsilon] \\ =& \hat{\mathbf{A}}\hat{\mathbf{y}}(\mathbf{x}_{t^*}) = \hat{\mathbf{A}}^{t^*+1-n}\mathbf{y}(\mathbf{x}_n).\end{aligned}$$

By the sampling model in Equation (4.6) and the law of total covariance, for any $t^* + 1$, the posterior covariance follows

$$\begin{aligned}&\mathbb{V}[\mathbf{y}(\mathbf{x}_{t^*+1}) \mid \mathbf{Y}, \hat{\mathbf{A}}, \hat{\boldsymbol{\Sigma}}_\varepsilon] \\ =& \mathbb{V}[\mathbb{E}[\mathbf{y}(\mathbf{x}_{t^*+1}) \mid \mathbf{Y}, \mathbf{y}(\mathbf{x}_{t^*}), \hat{\mathbf{A}}, \hat{\boldsymbol{\Sigma}}_\varepsilon]] + \mathbb{E}[\mathbb{V}[\mathbf{y}(\mathbf{x}_{t^*+1}) \mid \mathbf{Y}, \mathbf{y}(\mathbf{x}_{t^*}), \hat{\mathbf{A}}, \hat{\boldsymbol{\Sigma}}_\varepsilon]] \\ =& \mathbb{V}[\hat{\mathbf{A}}\mathbf{y}(\mathbf{x}_{t^*}) \mid \mathbf{Y}, \hat{\mathbf{A}}, \hat{\boldsymbol{\Sigma}}_\varepsilon] + \hat{\boldsymbol{\Sigma}}_\varepsilon \\ =& \hat{\mathbf{A}} \left(\sum_{i^*=0}^{t^*-n-1} \hat{\mathbf{A}}^{i^*} \hat{\boldsymbol{\Sigma}}_\varepsilon (\hat{\mathbf{A}}^\top)^{i^*} \right) \hat{\mathbf{A}}^\top + \hat{\boldsymbol{\Sigma}}_\varepsilon = \sum_{i^*=0}^{t^*-n} \hat{\mathbf{A}}^{i^*} \hat{\boldsymbol{\Sigma}}_\varepsilon (\hat{\mathbf{A}}^\top)^{i^*}.\end{aligned}$$

■

Although we can assess the uncertainty of the forecast by DMD from the predictive distribution in Equation (4.7), the linear state space model can be restrictive in approximating nonlinear dynamical systems. Furthermore, the uncertainty in estimating $\mathbf{A}$ and $\boldsymbol{\Sigma}_\varepsilon$ is not propagated for predictions. Finally, choosing the rank r in DMD is an open problem as it represents one's belief in the degree to which the model is misspecified, which could be hard to quantify precisely. Typical ways of choosing r include letting the summation of DMD eigenvalues explain a large proportion of the output variability, while this choice could potentially misfit the data, as minimizing the L_2 loss in Equation (1.45) cannot avoid overfitting the data.

4.2.5 Higher order dynamic mode decomposition

One limitation of the DMD approach is that only the observation from the prior time point is used, equivalently inducing a first-order Markov model in Equation (4.6). Variants of DMD approaches, such as Higher Order Dynamic Mode Decomposition (HODMD) [116] or Hankel DMD [144], use more observations from longer time lag to construct the dynamics:

$$\mathbf{y}(\mathbf{x}_{t+q}) = \mathbf{A}_1\mathbf{y}(\mathbf{x}_t) + \mathbf{A}_2\mathbf{y}(\mathbf{x}_{t+1}) + \cdots + \mathbf{A}_q\mathbf{y}(\mathbf{x}_{t+q-1}), \tag{4.8}$$

where $q \geq 1$ is a tunable parameter that determines the number of time-lagged snapshots to be included in the model.

The estimation accuracy from HODMD can be higher than that of conventional DMD, as multiple time-lagged snapshots are used. For scenarios where the number of time points is larger than the number of output coordinates, including more time-lagged snapshots can increase the upper bound of the number of nonzero singular values, thus potentially capturing complex dynamics in a higher-dimensional space. From the theoretical point of view, the eigenfunctions and eigenvalues of HODMD are guaranteed to converge to the Koopman eigenfunctions and eigenvalues for ergodic systems [144].

Let us define $\mathbf{y}^{\text{aug}}(\mathbf{x}_t) = (\mathbf{y}(\mathbf{x}_t)^\top, \mathbf{y}(\mathbf{x}_{t+1})^\top, ..., \mathbf{y}(\mathbf{x}_{t+q-1})^\top)^\top$, an augmented vector of kq dimensions that contain q snapshots. The linear mapping matrix $\hat{\mathbf{A}}^{\text{HODMD}}$ in HODMD can be obtained by minimizing the Frobenius or L_2 norm between the observations and linear dynamics constructed from the previous time steps: $\hat{\mathbf{A}}^{\text{HODMD}} = \arg\min_{\mathbf{A}^{\text{HODMD}}} \|\mathbf{Y}_2^{\text{aug}} - \mathbf{A}^{\text{HODMD}}\mathbf{Y}_1^{\text{aug}}\|$, where $\mathbf{Y}_2^{\text{aug}} = [\mathbf{y}^{\text{aug}}(\mathbf{x}_2), \ldots, \mathbf{y}^{\text{aug}}(\mathbf{x}_{n+1-q})]$ and $\mathbf{Y}_1^{\text{aug}} = [\mathbf{y}^{\text{aug}}(\mathbf{x}_1), \ldots, \mathbf{y}^{\text{aug}}(\mathbf{x}_{n-q})]$.

However, concatenating q consecutive snapshots increases the number of rows in $\mathbf{Y}_1^{\text{aug}}$ from k to kq, and the cost of a singular value decomposition for $\mathbf{Y}_1^{\text{aug}}$, leading to higher

computational cost. To overcome this limitation, one can use a subsampled version of the data instead of the entire dataset. For instance, one might skip Δt time steps when constructing the data matrices, i.e., $\tilde{\mathbf{Y}}_1^{\text{aug}} = [\mathbf{y}^{\text{aug}}(\mathbf{x}_1), \mathbf{y}^{\text{aug}}(\mathbf{x}_{1+\Delta t}), \ldots, \mathbf{y}^{\text{aug}}(\mathbf{x}_{1+\lfloor \frac{n-q}{\Delta t} \rfloor \Delta t})]$ and $\tilde{\mathbf{Y}}_2^{\text{aug}} = [\mathbf{y}^{\text{aug}}(\mathbf{x}_2), \mathbf{y}^{\text{aug}}(\mathbf{x}_{2+\Delta t}), \ldots, \mathbf{y}^{\text{aug}}(\mathbf{x}_{2+\lfloor \frac{n-q+1}{\Delta t} \rfloor \Delta t})]$. The estimator of $\mathbf{A}^{\text{HODMD}}$ can be computed below

$$\hat{\mathbf{A}}^{\text{HODMD}} = \arg \min_{\mathbf{A}^{\text{HODMD}}} \|\tilde{\mathbf{Y}}_2^{\text{aug}} - \mathbf{A}^{\text{HODMD}} \tilde{\mathbf{Y}}_1^{\text{aug}}\|. \tag{4.9}$$

The HODMD contains two prespecified parameters: the number of time-lagged snapshots q to be included in any given time, and the number of skipped time steps Δt in estimation. Note that the estimated $\mathbf{A}^{\text{HODMD}}$ does not preserve the model structure in Equation (4.8). Instead, let us consider the following generative model for HODMD with parameters $(q, \Delta t)$

$$\mathbf{y}^{\text{aug}}(\mathbf{x}_{2+\tilde{i}\Delta t}) = \mathbf{A}^{\text{HODMD}} \mathbf{y}^{\text{aug}}(\mathbf{x}_{1+\tilde{i}\Delta t}) + \boldsymbol{\varepsilon}_{2+\tilde{i}\Delta t}^{\text{aug}}, \tag{4.10}$$

for $\tilde{i} = 1, ..., \lfloor (n-q)/\Delta t \rfloor$ with $\boldsymbol{\varepsilon}_{2+\tilde{i}\Delta t}^{\text{aug}} \sim \mathcal{MN}(\mathbf{0}, \boldsymbol{\Sigma}_{\varepsilon}^{aug})$. In practice, the model performance depends on the choice of $(q, \Delta t)$, since the underlying data-generating model may rely on multiple time-lagged snapshots and using observations from longer time lag makes the model more accurate. The HODMD estimator in Equation (4.9) is equivalent to the MLE of $\mathbf{A}^{\text{HODMD}}$ in the generative model in Equation (4.10).

4.2.6 Extended dynamic mode decomposition

The generative model of the DMD algorithm in Equation (4.6) is a linear state space model, while some dynamical systems cannot be accurately approximated by linear dynamics. The extended dynamic mode decomposition (EDMD) [117] aims to define a dictionary of $\tilde{k}$ nonlinear basis functions $\boldsymbol{\psi}(\cdot) = (\psi_1(\cdot), \ldots, \psi_{\tilde{m}}(\cdot))^\top$ to lift the observations

to a system that can be approximated by linear dynamics. Denote the linear mapping matrix $\mathbf{A}^{\text{EDMD}}$ to be an approximation of the Koopman operator. In EDMD, the linear mapping matrix $\mathbf{A}^{\text{EDMD}}$ is obtained by minimizing the squared error loss function

$$\hat{\mathbf{A}}^{\text{EDMD}} = \arg\min_{\mathbf{A}^{\text{EDMD}}} \sum_{t=1}^{n-1} \|\boldsymbol{\psi}(\mathbf{y}(\mathbf{x}_{t+1})) - \mathbf{A}^{\text{EDMD}}\boldsymbol{\psi}(\mathbf{y}(\mathbf{x}_t))\|_2^2. \tag{4.11}$$

Similar to the DMD-induced process, the estimator of EDMD is equivalent to the maximum likelihood estimator of the linear mapping matrix in a linear state space model defined in the lifted space for $t = 1, \ldots, n-1$:

$$\boldsymbol{\psi}(\mathbf{y}(\mathbf{x}_{t+1})) = \mathbf{A}^{\text{EDMD}}\boldsymbol{\psi}(\mathbf{y}(\mathbf{x}_t)) + \boldsymbol{\varepsilon}_{t+1}^{\text{EDMD}}, \tag{4.12}$$

with $\boldsymbol{\varepsilon}_{t+1}^{\text{EDMD}} \sim \mathcal{MN}(\mathbf{0}, \boldsymbol{\Sigma}_{\varepsilon}^{\text{EDMD}})$, where $\boldsymbol{\Sigma}_{\varepsilon}^{\text{EDMD}}$ is a positive definite matrix.

After estimating the linear mapping matrix between the linear state space model, we need to transform it back to predict the future states [120], which may be achieved by defining $\hat{\mathbf{y}}(\mathbf{x}_t) = \mathbf{P}\boldsymbol{\psi}(\mathbf{y}(\mathbf{x}_t))$, where $\mathbf{P}$ is a $k \times \tilde{k}$ matrix and can be estimated by $\hat{\mathbf{P}} = \arg\min_{\mathbf{P}} \sum_{t=1}^{n} \|\mathbf{y}(\mathbf{x}_t) - \mathbf{P}\boldsymbol{\psi}(\mathbf{y}(\mathbf{x}_t))\|^2$.

Choosing an appropriate set of basis functions is crucial for the EDMD method. A few generic basis functions, such as Hermite polynomials, radial basis functions, and discontinuous spectral elements, were suggested in [117]. The selection of basis functions depends on the context of the problem and domain knowledge may be used as well, whereas misspecifying basis functions can degrade the estimation efficiency of the model. PP-GP is closely connected to EDMD. Assuming the mean is zero, we can write the predictive mean of PP-GP in a matrix form $\hat{\mathbf{Y}} = \mathbf{WK}$, where $\mathbf{Y}$ is a $m \times n$ observational matrix of n snapshots, and $\mathbf{W} = [\mathbf{w}_1^\top, ..., \mathbf{w}_m^\top]^\top$ is an $m \times n$ weight matrix given from Corollary 1. This means the prediction from PP-GP uses the same kernel basis to

represent the output for each coordinate, whereas the weights are estimated by solving the linear system of equations with shared coefficients, separately for the output at each coordinate. Compared to EDMD, the PP-GP does not project the output onto the lifted space, and hence we do not need to transform the lifted states back for forecasting. The PP-GP is a flexible model, as the mean and variance parameters are distinct for each coordinate, which can be marginalized out by computing the predictive distribution.

4.2.7 Computational complexity

The computational complexity of estimating the transition and covariance matrices in DMD is $\mathcal{O}(\min(k^2n, kn^2))$ and $\mathcal{O}(k^2n)$, respectively. Obtaining the n^*-step forecast with uncertainty quantification by DMD requires $\mathcal{O}(k^2rn^*)$, where r is the rank of the observation matrix $\mathbf{Y}_1$. Similarly, for HODMD with time lag q and thining parameter Δt, the computational complexity of parameter estimation and n^*-step forecast with uncertainty quantification are $\mathcal{O}(\min((kq)^2\lfloor\frac{n}{\Delta t}\rfloor, kq(\lfloor\frac{n}{\Delta t}\rfloor)^2)) + \mathcal{O}((kq)^2\lfloor\frac{n}{\Delta t}\rfloor)$ and $\mathcal{O}((kq)^2rn^*)$, respectively. For EDMD using $\tilde{k}$ radial basis functions to lift the data where the centers are determined by the K-means clustering approach, transforming the data and estimating the parameters in the lifted space requires $\mathcal{O}(kn\tilde{k}T)$, where T is the number of iterations in K-means. Obtaining the predictions needs $\mathcal{O}(k\tilde{k}n^*)$. For simplicity, here we assume $k > n > \tilde{k}$. The computational complexity for estimating the parameters as well as providing forecasts with uncertainty assessment by DMD, HODMD, EDMD, and PP-GP is summarized in Table 4.1.

	Parameter estimation	Forecast and UQ
DMD	$\mathcal{O}(k^2 n)$	$\mathcal{O}(k^2 r n^*)$
HODMD	$\mathcal{O}((kq)^2 \lfloor \frac{n}{\Delta t} \rfloor)$	$\mathcal{O}(k^2 q^2 r n^*)$
EDMD	$\mathcal{O}(kn\tilde{k}T)$	$\mathcal{O}(k\tilde{k}n^*)$
PP-GP	$\mathcal{O}(M_{pp} n^3 + M_{pp} n^2 k)$	$\mathcal{O}(M_c k n^2 n^* + n^3)$

Table 4.1: Computational complexity for parameter estimation and forecast of n^* time points in DMD, HODMD, EDMD, and PP-GP, where k is the dimension of an output, n is the number of observations, r is the rank of data matrix, q is the number of snapshots stacked in HODMD, Δt is the number of skipped time points in HODMD, $\tilde{k}$ is the number of basis functions in EDMD, T is the number of iterations in K-means, M_{pp} and M_c are the iterations of numerical optimization and samples for forecast in PP-GP.

4.3 Connection of different data-driven approaches of modeling dynamical systems with respect to the generative models

Here we compare three classes of data-driven models, namely the proper orthogonal decomposition (POD) [61], DMD, and PP-GP approaches. To simplify the notations, we assume the data are properly centered, meaning that the $k \times n$ real-valued output matrix $\mathbf{Y}$ has zero mean.

In POD, the data are decomposed by SVD $\mathbf{Y} = \mathbf{U}_y \mathbf{D}_y \mathbf{V}_y^\top$, where $\mathbf{U}_y$ and $\mathbf{V}_y$ are $k \times k$ and $n \times n$ unitary matrix, respectively, and $\mathbf{D}_y$ is a $k \times n$ rectangle diagonal matrix with non-negative singular values in the diagonals. The first $r \leq k$ columns of $\mathbf{U}_y$ associated with the largest r singular values provide the orthogonal basis of a linear subspace to reconstruct the covariance of output at different coordinates by treating the temporal observations as independent measurements: $\mathbf{Y}\mathbf{Y}^\top/(n-1) = \mathbf{U}_y \mathbf{D}_y^2 \mathbf{U}_y^\top/(n-1)$. This approach is known as the principal component analysis (PCA) [145], which is widely used in unsupervised learning and dimension reduction. The SVD basis from the PCA can be shown to have the same linear subspace to the maximum marginal likelihood

estimator of the linear mapping matrix $\mathbf{A}$ after marginalizing out $\mathbf{z}(\mathbf{x}_t)$ in the following generative model [73]:

$$\mathbf{y}(\mathbf{x}_t) = \mathbf{A}\mathbf{z}(\mathbf{x}_t) + \boldsymbol{\epsilon}_t, \tag{4.13}$$

where $\boldsymbol{\epsilon}_t \sim \mathcal{MN}(\mathbf{0}, \sigma_0^2\mathbf{I}_m)$ and $\mathbf{z}(\mathbf{x}_t) \sim \mathcal{MN}(\mathbf{0}, \mathbf{I}_r)$ is a r-dimensional latent factors independently following standard normal distributions. Under such a model, the covariance of the data at each time follows $\mathbb{V}[\mathbf{y}(\mathbf{x}_t)] = \mathbf{A}\mathbf{A}^\top + \sigma_0^2\mathbf{I}_m$. However, the generative model assumes independence between the observations at different time points, which is restrictive. In [19], $z_l(\cdot)$ is modeled as a GP for each $l = 1, ..., r$, and the maximum marginal likelihood estimator of $\mathbf{A}$ under the assumption $\mathbf{A}^\top\mathbf{A} = \mathbf{I}_r$ is derived.

Second, the generative model of DMD is given in Equation (4.6), where the noise of the data is not modeled. Assuming the initial states follow a multivariate normal distribution with zero mean and covariance $\boldsymbol{\Sigma}_\varepsilon$. It is not hard to show that $\mathbb{E}[\mathbf{y}(\mathbf{x}_t)] = \mathbf{0}$ and the covariance between any two output vectors at two time points follows: $\mathrm{Cov}[\mathbf{y}(\mathbf{x}_{t'}), \mathbf{y}(\mathbf{x}_t)] = \mathbf{A}^{t'-t}\sum_{i=0}^{t-1}\mathbf{A}^i\boldsymbol{\Sigma}_\varepsilon(\mathbf{A}^\top)^i$ for $t' \geq t$. Specifically, the covariance of output at any time point $t \geq 1$ follows $\mathbb{V}[\mathbf{y}(\mathbf{x}_t)] = \sum_{i=0}^{t-1}\mathbf{A}^i\boldsymbol{\Sigma}_\varepsilon(\mathbf{A}^\top)^i$. Compared with the sampling model in POD, the generative model in the DMD-induced process in Equation (4.6) is correlated over time, and the strength of the correlation is captured by the linear mapping matrix $\mathbf{A}$ between output vectors at the consecutive time points. The DMD-induced process may not be differentiable with respect to time, and the assumption of homogeneous variance at each output coordinate may also be restrictive for applications where the output has different scales.

Third, the PP-GP model in Equation (1.20) has the same predictive mean as modeling the output matrix $\mathbf{Y}$ by a matrix-normal distribution [27], with a separable covariance $\mathbb{V}[\mathbf{Y}] = \boldsymbol{\Sigma}_y \otimes \tilde{\mathbf{R}}$, where $\boldsymbol{\Sigma}_y$ is the covariance between output coordinates with the jth diagonal term being σ_j^2, for $j = 1, ..., k$ and $\tilde{\mathbf{R}}$ is the correlation matrix between inputs

with $\otimes$ denoting the Kronecker product. In comparison, the generative model by DMD in Equation (4.6) is a linear state space model, which has a semi-separable covariance structure. The PP-GP induces nonlinear dynamics when using the observations from previous time points as the inputs, and the differentiability of the nonlinear processes induced by PP-GP can be controlled by the choice of kernel function. When the underlying dynamic is smooth, a differentiable GP prior of the nonlinear dynamics by PP-GP may be preferred to have a better convergence rate compared to a nondifferentiable GP prior [127]. Another advantage of PP-GP is that the range parameters can be estimated by the MLE or maximum marginal posterior mode, which is more flexible than using fixed nonlinear basis functions in EDMD. Lastly, the variance of the output coordinate is distinct and the variance estimator of PP-GP has a closed-form expression in Equation (1.21), whereas the induced processes by DMD and its variants typically have homogeneous variance. The different variance terms make PP-GP particularly suitable when the output has different scales, which are common in practice.

4.4 Numerical results

We compare different data-driven forecast approaches for nonlinear dynamical systems, focusing on uncertainty quantification of the forecast. We consider two scenarios. In the first scenario, we assume the underlying dynamical system is modeled by a map from $\mathbb{R}^p \to \mathbb{R}^k$: $d\mathbf{y}/dt = \mathbf{f}(\mathbf{x}_t)$, where the input variables $\mathbf{x}_t$ is a subset of $\mathbf{y}_t$. Here, the vector-valued function $\mathbf{f}$ is treated as unknown and required to be approximated, whereas the inputs $\mathbf{x}_t$ are known. For all approaches, we do not include the vector-valued function $\mathbf{f}(\cdot)$ in nonlinear basis functions. Instead, we test uncertainty quantification with generic kernels or nonlinear basis functions that provide default ways of approximation. In the second scenario, the dynamical system is described by $d\mathbf{y}/dt = \mathbf{f}(\mathbf{x}_t, \mathbf{u}_t)$, where we can

only observe $\mathbf{x}_t$, whereas the external inputs $\mathbf{u}_t$ and the vector-valued function $\mathbf{f}(\cdot)$ are unobserved.

For both scenarios, we forecast held-out data $\mathbf{y}(\mathbf{x}_{t^*}) = (y_1(\mathbf{x}_{t^*}), ..., y_m(\mathbf{x}_{t^*}))^\top$ at $t^* = n+1, n+2, ..., n+n^*$. The autoregressive model with lag order 1 (AR(1)) separately fitted for each coordinate is used as a benchmark prediction model [146]. Also included are DMD, HODMD and PP-GP. Note that the generative model of the DMD method is equivalent to a noise-free vector autoregressive (VAR) model with lag order 1 [147] and hence we do not include other VAR models for comparison. The criteria include the predictive root of mean squared error (RMSE), the average length of the 95% predictive intervals ($L(95\%)$), and the proportion of the samples covered in the 95% predictive interval ($P(95\%)$):

$$\text{RMSE} = \left(\sum_{j=1}^{k} \sum_{t=n+1}^{n+n^*} (\hat{y}_j(\mathbf{x}_{t^*}) - y_j(\mathbf{x}_{t^*}))^2 \right)^{\frac{1}{2}}, \tag{4.14}$$

$$L(95\%) = \frac{1}{kn^*} \sum_{j=1}^{k} \sum_{t^*=n+1}^{n^*+n} \text{length}\left\{CI_{j,t^*}(95\%)\right\}, \tag{4.15}$$

$$P(95\%) = \frac{1}{kn^*} \sum_{j=1}^{k} \sum_{t^*=n+1}^{n^*+n} 1_{y_j(\mathbf{x}_{t^*}) \in CI_{j,t^*}(95\%)}, \tag{4.16}$$

where $\hat{y}_j(\mathbf{x}_{t^*})$ is the prediction of the output at coordinate j with input $\mathbf{x}_{t^*}$, $CI_{j,t^*}(95\%)$ is the 95% predictive interval of the output at coordinate j and time t^*, length $\{CI_{j,t^*}(95\%)\}$ denotes the length of the predictive interval, and $\bar{y} = \sum_{j=1}^{k} \sum_{t=1}^{n} y_j(\mathbf{x}_t)/(kn)$ is the mean of the observations in the training data set. An accurate method should have a small predictive error quantified by RMSE, a short average length of 95% predictive interval ($L(95\%)$) and the proportion of the sample covered by the 95% predictive interval ($P(95\%)$) should be close to the 95% nominal level.

4.4.1 Lorenz 96 system

We first discuss the Lorenz 96 system for modeling the atmospheric quantities at equally spaced locations along a cycle [6]:

$$\frac{dy_j(t)}{dt} = (y_{j+1}(t) - y_{j-2}(t))y_{j-1}(t) - y_j(t) + F, \tag{4.17}$$

for $j = 1, ..., k$, where $k = 40$ and $F = 8$ are typically used for testing. Here $f_j(\mathbf{x}_t) = (y_{j+1}(t) - y_{j-2}(t))y_{j-1}(t) - y_j(t)$, and the input is $\mathbf{x}_t = [y_{j-2}(t), y_{j-1}(t), y_j(t), y_{j+1}(t)]$. The Lorenz 96 system is often used for demonstrating the effectiveness of nonlinear filtering approaches, such as ensemble Kalman filter in data assimilation [148], where the function $f_j(\cdot)$ is typically assumed to be known. Here we assume the underlying dynamics from $f_j(\cdot)$ is unknown.

We test a few methods and compare their performance on uncertainty quantification. We assume both the derivative and output values are available. The data are obtained by the Runge-Kutta method of order 4 with step size $h = 0.01$ for 1000 steps. The initial values of the states are sampled from a zero-mean multivariate normal distribution where the covariance matrix is sampled from a Wishart distribution with the scale matrix being identity and k degrees of freedom [149]. Any method can use the $kn = 4,000$ observations from the first $n = 100$ time points as training observations, whereas the rest of the $36,000$ observations at later $n^* = 900$ time points are held out as test data. For DMD and HODMD, we try both the observed output values and derivatives for forecasting. Since neither way works well, we only present results based on the observed output values. When constructing the data matrices for HODMD, 6 observed snapshots ($q = 6$) are concatenated and 3 time points are skipped in the augmented data ($\Delta t = 3$). For PP-GP, we uniformly subsample $n_{training} = 500$ observations from $4,000$ observations of derivatives in the training time period to estimate the parameters and construct predictive

distributions in Equation (1.19), because of the high computational cost when n is large. We use the default product Matérn covariance function with roughness parameter being 2.5 in PP-GP. As PP-GP can be considered as an extended version of DMD on projecting the data onto the kernel space discussed in Section 4.2.6, we do not include any other EDMD approach. The range parameters of the kernel in PP-GP are estimated by the default marginal posterior mode estimation [33], which is more flexible than assuming a fixed nonlinear basis function in EDMD.

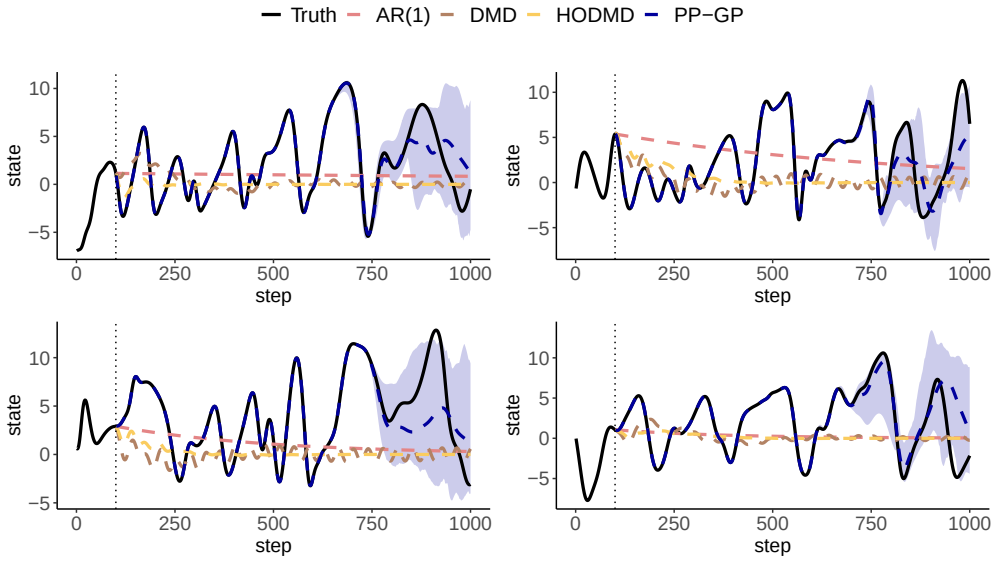

Figure 4.1: Forecast of the Lorenz 96 systems for 900 steps by AR(1) (pink dashed curves), DMD (brown dashed curves), HODMD (yellow dashed curves) and PP-GP (blue dashed curve). The 95% predictive interval by PP-GP is graphed as a blue shaded area. The blue dashed curves (PP-GP) and black curves (held-out truth) overlap for around the first 500 held-out time steps.

Figure 4.1 gives the 900-step forecast by AR(1), DMD, HODMD and PP-GP for the 10th, 20th, 30th, and 40th states. The uncertainty of the forecast by PP-GP is graphed as the blue shaded area in all plots. With the default kernel function and estimation [33], the forecast of PP-GP is reasonably accurate for the first 500 time steps and 95% predictive interval by PP-GP (graphed as the blue shaded area) is almost indistinguishable in this

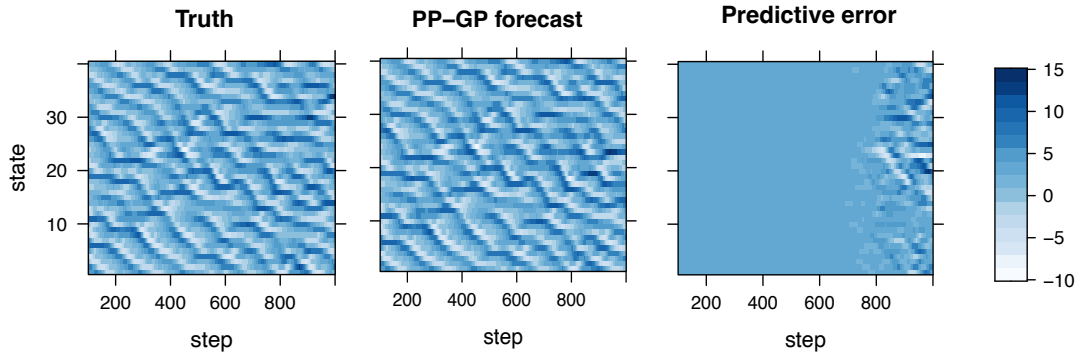

Figure 4.2: The truth, 900-step forecast by PP-GP, and their difference for the Lorenz 96 system.

500-step forecast	RMSE	$P(95\%)$	$L(95\%)$
AR(1)	4.60	69.8%	10.8
DMD	4.51	91.6%	20.2
HODMD	4.24	98.8%	39.4
PP-GP	0.0126	93.9%	0.0352
900-step forecast	RMSE	$P(95\%)$	$L(95\%)$
AR(1)	4.63	75.4%	12.6
DMD	4.55	93.5%	21.9
HODMD	4.37	99.3%	43.6
PP-GP	1.52	94.6%	2.64

Table 4.2: Forecast accuracy and uncertainty assessment on the held-out data. The standard deviation is 3.54 and 3.62 for the 500-step test data and 900-step test data, respectively.

time domain. As the average Lyapunov time for our simulation is around 1.58, the PP-GP can make a precise forecast for more than 3 Lyapunov times. The 95% predictive interval becomes noticeably wider at later time steps, and simultaneously, the difference between PP-GP and held-out truth becomes large. The internal uncertainty assessment quantified by the 95% predictive interval of PP-GP provides a time range of reliable forecasts without knowing the held-out truth. Furthermore, Figure 4.2 compares the truth to the forecast by PP-GP for all states, which confirms that the forecast by PP-GP for the first 500 steps is accurate for all states.

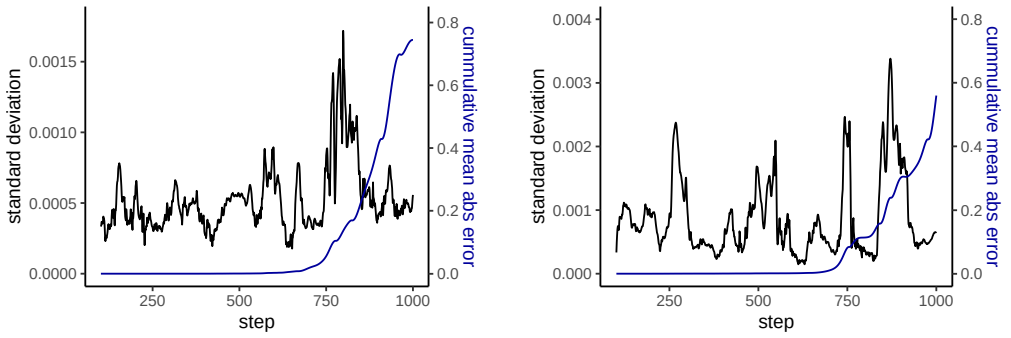

Figure 4.3: The predictive standard deviation from the PP-GP model at each step and cumulative mean absolute error $SE_j(t^*) = \sum_{s^*=n+1}^{t^*} |\hat{y}_j(s^*) - y_j(s^*)|/(t^* - n)$ of 900-step forecast of state $j = 1$ and $j = 21$ for $t^* = 101, ..., 1000$ and $n = 100$.

Table 4.2 summarizes the performance of forecast and uncertainty assessment by different approaches for the Lorenz 96 system. The RMSE by the PP-GP for the first 500 steps is much smaller than the standard deviation of the test data and the length of the 95% predictive interval by PP-GP is also substantially smaller than the variability in the held-out observations. Even if the 95% predictive interval by PP-GP is short, it covers 93.9% of the observations, indicating the uncertainty of the forecast is properly quantified. The predictive error by PP-GP becomes large at later steps due to the accumulation of the approximation error, and the overall predictive error of the 900-step forecast is dominated by the large error at later time points. Note that the length of the 95% predictive interval from PP-GP increases automatically in PP-GP, enabling 94.6% of the held-out data to be covered by the 95% predictive interval. In comparison, although the proportion of the samples covered by the 95% predictive interval by DMD and HODMD is also close to the 95% nominal level, the average length of intervals is substantially larger, as the underlying dynamics cannot be written as a linear combination of previous outputs. It is worth mentioning that the uncertainty of DMD and HODMD is affected by the selected rank to represent the data, here chosen to be the smallest value such that the summation of the eigenvalues explains at least 99% of the variability.

A principled way to model the noise and select the rank may improve the uncertainty assessment of these methods.

Figure 4.3 presents the predictive standard deviation of PP-GP and the corresponding cumulative mean absolute error between the true values and PP-GP forecasts, for the 1st and 21st states. In the initial 500-step forecasts, both the standard deviation computed by Equation (1.19) and the cumulative mean absolute error are relatively small. As the number of forecast steps increases, the cumulative mean absolute error increases. The 95% predictive interval in Figure 4.1 can be used to quantify the time when the forecast becomes inaccurate.

The uncertainty assessment by the PP-GP model is reasonably accurate as we convert the challenging problem of forecasting chaotic systems to the problem of predicting the one-step-ahead function in a 4-dimensional input space. In practice, reducing the inputs to a low-dimensional space is helpful for producing reliable forecasts and uncertainty assessments.

4.4.2 Time-dependent Green's function

In the second example, we apply the data-driven approach to forecast the simulation of the nonequilibrium dynamics of interacting electrons in materials exposed to intense time-varying electric fields, which is one of the most challenging problems in condensed matter physics. Modeling nonequilibrium dynamics from the basic principles of quantum mechanics is exceptionally challenging in computation, and has only been extended to the *ab initio* simulation of real materials in the last decade through the development of a new time-domain approach based on a formalism of Keldysh, Kadanoff and Baym [3, 4, 5] for the dynamics of the nonequilibrium Green's function [132, 133, 150]. We refer to the new computational approach as the time-dependent adiabatic GW (TD-

aGW) method, where G stands for the interacting single-particle Green's function and W stands for the screened Coulomb interaction. Details about our implementation of the method can be found in [133], which is built on approximations developed in [132]. In principle, the TD-aGW approach gives quantitatively accurate descriptions of materials' response to electric fields, allowing for the simulation of experiments with femtosecond resolution and the development of new classes of quantum materials whose properties can be switched with laser pulses [134, 135, 151]. However, the scaling with respect to the size of the problem is computationally prohibitive, and thus, the TD-aGW method is currently limited to systems containing a few atoms and for short timescales.

In the context of many-body perturbation theory and second quantization, the non-equilibrium Green's function is a two-point correlation function comprised of the creation and annihilation field operators that increase and decrease the number of particles in a given state, respectively. In general, it is a function of two time variables, but following the work in [132, 133], the change in the energy of an electron due to interactions with its environment (i.e., the electron self-energy) can be approximated as the equilibrium self-energy plus a static nonequilibrium contribution. In this approximation, the key equation that we solve in TD-aGW is an equation of motion for a matrix of single-particle density $\boldsymbol{\rho}(t)$ with index $(m_1 m_2, \mathbf{k})$

$$i\hbar \frac{\partial}{\partial t} \rho_{m_1 m_2, \mathbf{k}}(t) = [\mathbf{H}(t), \boldsymbol{\rho}(t)]_{m_1 m_2, \mathbf{k}}, \tag{4.18}$$

where $\mathbf{H}(t)$ is a matrix of the Hamiltonian of the system having the same size of $\boldsymbol{\rho}(t)$, the square bracket denotes the commutator of two matrices $\mathbf{M}_1$ and $\mathbf{M}_2$ such that $[\mathbf{M}_1, \mathbf{M}_2] = \mathbf{M}_1\mathbf{M}_2 - \mathbf{M}_2\mathbf{M}_1$, and $\hbar$ is the Planck constant.

The particle density matrix is written in a basis of electron quantum states with a band index m and a wave vector (or crystal momentum) $\mathbf{k}$. Hence, the matrix element

$\rho_{m_1 m_2,\mathbf{k}}(t)$ encodes the entanglement of a state $(m_1, \mathbf{k})$ with another state $(m_2, \mathbf{k})$. The Hamiltonian of the system is calculated as

$$\mathbf{H}(t) = \mathbf{H}_0 - e\mathbf{E}(t) \cdot \mathbf{r} + \mathbf{\Sigma}^{GW} + \delta\mathbf{\Sigma}(t). \tag{4.19}$$

Here, $\mathbf{H}_0$ is the system's mean-field Hamiltonian at equilibrium; $\mathbf{\Sigma}^{GW}$ is the electron self-energy at equilibrium computed within the GW approximation; and $\delta\mathbf{\Sigma}$ is the nonequilibrium correction to the electron self-energy. $e\mathbf{E}(t) \cdot \mathbf{r}$ describes the coupling of the system with the external electric field, such that $\mathbf{E}(t)$ is a spatially-uniform, time-varying electric field; $\mathbf{r}$ is the position operator in quantum mechanics; and e is the fundamental charge of the electron. $\mathbf{H}_0$ and $\mathbf{\Sigma}^{GW}$ describe equilibrium properties and thus have no time-dependence. $\delta\mathbf{\Sigma}(t)$ is a functional of $\boldsymbol{\rho}(t)$:

$$\delta\Sigma_{m_1 m_2,\mathbf{k}}(t) = \sum_{m_1', m_2', \mathbf{k}'} \rho_{m_1 m_2, \mathbf{k}-\mathbf{k}'}(t) W_{m_1 m_1' m_2 m_2', \mathbf{k}-\mathbf{k}'}, \tag{4.20}$$

where $W_{m_1 m_1' m_2 m_2', \mathbf{k}-\mathbf{k}'}$ is the equilibrium screened Coulomb interaction computed in the random-phase approximation (RPA); m_1, m_1', m_2, m_2' are band indices, and $\mathbf{k}$ and $\mathbf{k}'$ are vectors in reciprocal space.

We use monolayer MoS_2–a material where the electron self-energy is known to be large [152, 153, 154, 155, 156]–as our test system. In order to perform the time evolution in Equation (4.18), we first need to compute the equilibrium solution before the electric field is turned on to obtain $\boldsymbol{\rho}(t = 0)$, as well as the time-independent matrices $\mathbf{H}_0$ and $\mathbf{\Sigma}^{GW}$ in Equation (4.18) and $\mathbf{W}$ in Equation (4.20). We do this within the one-shot GW approximation [129, 130, 157] using the following calculation parameters. Our basis includes 4 bands (the top 2 valence bands and the bottom 2 conduction bands) and a uniform grid of $36 \times 36 \times 1$ $\mathbf{k}$-points in reciprocal space. Density functional theory

(DFT) calculations with spin-orbit coupling are performed using the Quantum Espresso package [158]. We use norm-conserving fully relativistic PBE pseudopotentials from the SG15 ONCV potential library [159] and a plane wave cutoff energy of 80 Ry. A GW plus Bethe Salpeter equation (BSE) calculation is done as a one-shot calculation on top of DFT using the BerkeleyGW package [157]. A dielectric cutoff of 10 Ry and 6000 bands is used in the GW and BSE calculations. We use the results of the equilibrium calculations to set up Equation (4.18) at time t=0, and then we propagate the equation in time with an external electric field that mimics a laser pulse polarized along the r_1 direction. The field is $E_{r_1}(t) = A \sin\left(\frac{\pi t}{T}\right)^2 \sin(\omega t)$, where the constants (in Ryd) $A = 0.006$ is the amplitude of the electric field, $\omega = 0.022$ is the frequency of the light, and $T = 160$ fs is the duration of the light pulse, all in Rydberg atomic units. $E_{r_2}(t) = 0$ and $E_{r_3}(t) = 0$. The electric field parameters are values consistent with typical experimental setups in high harmonic generation (HHG) experiments [160]. The 4^{th} order Runge-Kutta method is then used to update $\rho_{m_1 m_2, \mathbf{k}}(t)$ and $\delta\Sigma(t)$ at each time step, and the matrix element $\rho_{m_1 m_2, \mathbf{k}}(t)$ is saved for the single $\mathbf{k}$-point, $\mathbf{k} = \mathbf{0}$, to be used as the test and training data for our data-driven models.

Our focus in this example is on forecasting the real part of the density matrix from Equation (4.18), which in turn is related to the two-time Green's function through the Generalized Kadanoff-Baym ansatz (GKBA) [5, 161]. Each snapshot is a 16-dimensional vector: $\mathbf{y}_t = \text{Vec}(\rho_{\mathbf{k}}(t))$, where $\rho_{\mathbf{k}}(t)$ is a 4×4 matrix with the (m_1, m_2)th entry being $\rho_{m_1 m_2, \mathbf{k}}(t)$ for $m_1 = 1, ..., 4$ and $m_2 = 1, ..., 4$, and $\mathbf{k} = \mathbf{0}$. To solve Equation (4.18), one needs to compute $\mathbf{E}(t)$, which is a known analytic function, and $\delta\mathbf{\Sigma}(t)$, which depends on the density matrix $\rho_{m_1 m_2, \mathbf{k}-\mathbf{k}'}(t)$ at other reciprocal lattice vectors $\mathbf{k} \neq \mathbf{0}$. To evaluate the performance of that data-driven approaches, we only utilize the observations $\mathbf{y}_t$ to construct the model, which only contains the local information at $\mathbf{k} = \mathbf{0}$, whereas the external field $\mathbf{E}(t)$ and interactions terms $\delta\mathbf{\Sigma}(t)$ from other lesser Green's function $G^{<}_{m_1 m_2, \mathbf{k}-\mathbf{k}'}(t)$

1000-step forecast	RMSE	$P(95\%)$	$L(95\%)$
AR(1)	6.20×10^{-6}	23.7%	3.97×10^{-6}
DMD	2.87×10^{-5}	8.65%	2.03×10^{-6}
HODMD	1.30×10^{-5}	9.52%	6.56×10^{-7}
PP-GP	3.53×10^{-6}	65.2%	2.98×10^{-6}
2000-step forecast	RMSE	$P(95\%)$	$L(95\%)$
AR(1)	6.23×10^{-6}	35.8%	5.57×10^{-6}
DMD	2.01×10^{-3}	4.35%	7.29×10^{-5}
HODMD	6.22×10^{-5}	9.22%	1.94×10^{-6}
PP-GP	3.42×10^{-6}	75.3%	3.97×10^{-6}

Table 4.3: Forecast and uncertainty assessment for the density matrix using the held-out data. Observations at the first 2500 time steps are used to fit the model. The standard deviation is 1.56×10^{-5} and 1.48×10^{-5} of the 1000-step test data and the 2000-step test data, respectively.

are not used. For all methods, we test two scenarios with training time steps of 2500 and 3500, respectively. For AR(1) and DMD, all training data are used. For HODMD, we use the same setting as the previous example with $q = 6$ and $\Delta t = 3$. For PP-GP, we use an isotropic kernel and 800 pairs of observations, uniformly sampled from the training data for constructing the model, and the output vector of $m = m_1 m_2 = 16$ dimensions from the previous time point is used as the input for predicting the one-step-ahead transition function.

Table 4.3 and Table 4.4 provide the performance of each method when using observations from the first 2500 time steps and first 3500 time steps as the training data, respectively. The PP-GP has the smallest RMSE for the 2000-step forecast among the three approaches in both scenarios, and the smallest RMSE for the 1000-step forecast in the first scenario.

The misspecification of the input variables, however, degrades the accuracy of predictions and uncertainty quantification. The predictive error differs when using two output regimes as the training data, as the trend is different, and neither represents the trend in the held-out test data. The coverage of held-out data by the 95% predictive interval

1000-step forecast	RMSE	$P(95\%)$	$L(95\%)$
AR(1)	1.13×10^{-5}	31.7%	7.12×10^{-6}
DMD	6.00×10^{-6}	15.9%	1.00×10^{-6}
HODMD	9.29×10^{-6}	29.6%	1.05×10^{-6}
PP-GP	3.93×10^{-6}	81.7%	1.89×10^{-5}
2000-step forecast	RMSE	$P(95\%)$	$L(95\%)$
AR(1)	1.50×10^{-5}	28.5%	9.93×10^{-6}
DMD	8.83×10^{-6}	16.3%	2.98×10^{-6}
HODMD	2.24×10^{-5}	20.2%	1.62×10^{-6}
PP-GP	9.13×10^{-6}	88.8%	4.36×10^{-5}

Table 4.4: Forecast and uncertainty assessment of the density using the held-out data. Observations from the first 3500-time steps are used to fit the model. The standard deviation is 1.40×10^{-5} and 1.06×10^{-5} of the 1000-step test data and the 2000-step test data, respectively.

from the PP-GP is the highest among all methods, and having observations from a longer training time period seems to improve the overall coverage of the held-out data.

Figure 4.4 and Figure 4.5 display the 1000-step forecast by AR(1), DMD, HODMD and PP-GP model using 2500 and 3500 timesteps as training data, ,respectively. The PP-GP model can make an accurate prediction for the first few cycles with short predictive intervals. The prediction error accumulates, and the model automatically detects the inaccuracy of the prediction, leading to large 95% predictive intervals at later time points. Note that the scale of the held-out truth is decreasing and the overall trend is not captured by any method. This is because for all four methods, some inputs such as $\mathbf{H}(t)$ and $\boldsymbol{\rho}(t)$ at $\mathbf{k} \neq \mathbf{0}$, are assumed to be unknown. The large predictive intervals from PP-GP indicate substantial differences between the outputs in the training and forecast period, suggesting that more information is required to obtain an accurate prediction.

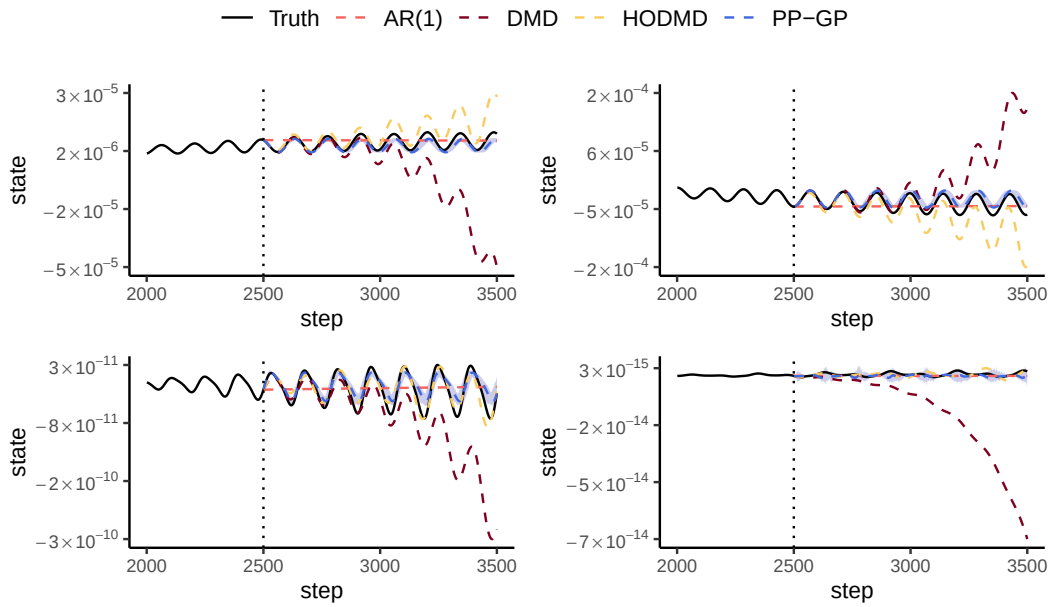

Figure 4.4: Forecast of two time Green's function for 1000 steps by AR(1) (orange dashed curves), DMD (brown dashed curves), HODMD (yellow dashed curves) and PP-GP (blue dashed curve). 2500-time steps are used as training data. The 95% predictive interval by PP-GP is graphed as blue shaded area.

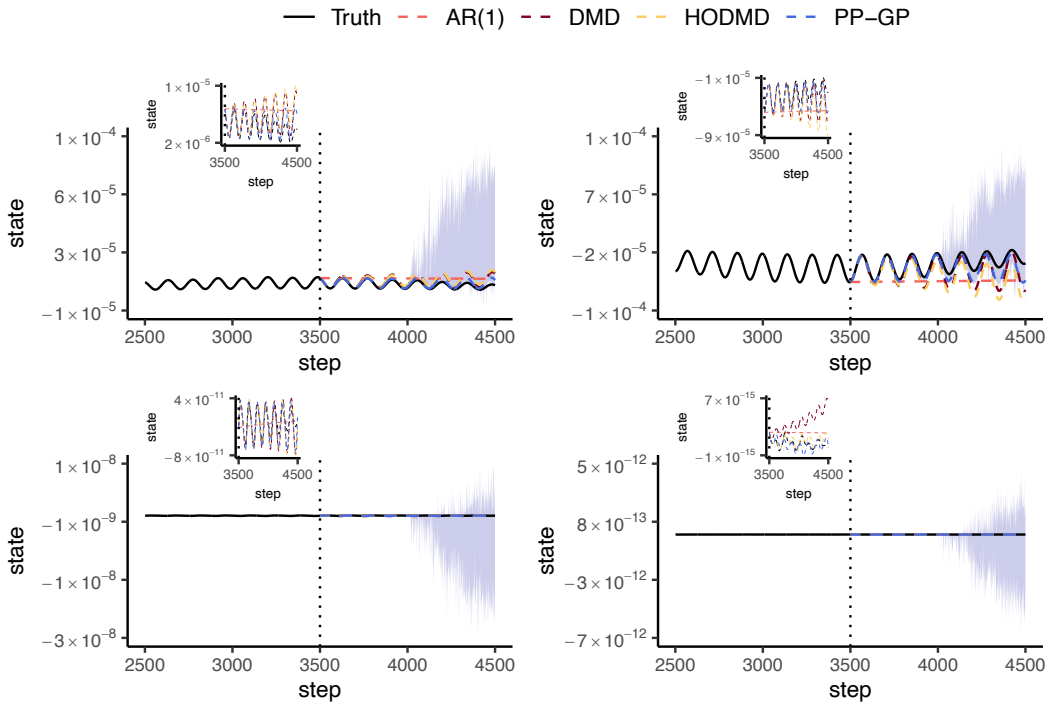

Figure 4.5: Forecast of two-time Green's function for 1000 steps by AR(1) (orange dashed curves), DMD (brown dashed curves), HODMD (yellow dashed curves) and PP-GP (blue dashed curve). 3500 time steps are used as training data. The 95% predictive interval by PP-GP is graphed as blue shaded area.

Chapter 5

Conclusion and future work

This dissertation develops latent factor models and Gaussian process methods for high-dimensional dynamical systems. The proposed approaches provide scalable tools for parameter estimation and forecasting. They collectively extend the applicability of probabilistic modeling to large-scale scientific and engineering problems.

In Chapter 2, we present a scalable algorithm for parameter estimation in a latent factor model for high-dimensional observations. We first specify a latent factor model with an orthogonal factor loading matrix, where each latent factor follows an Ornstein-Uhlenbeck process with distinct correlation and variance parameters. We then develop a fast EM algorithm, named the Fast algorithm for Multivariate Ornstein-Uhlenbeck process (FMOU), that leverages the orthogonal loading structure, KF, and RTS smoother, achieving linear computational complexity in both the output dimension and the number of observations. Simulation studies and a real application to fault slip estimation demonstrate the accuracy and computational efficiency of the proposed approach.

In Chapter 3, we introduce three modules in the open-source R package `FastGaSP` for scalable GP and latent factor model inference. The `fgasp` module leverages forward filtering and backward smoothing to evaluate the profile log-likelihood and compute the

predictive distribution in $\mathcal{O}(n)$ times. Built on this module, the `gppca` module provides parameter estimation for the Generalized Probabilistic Principal Component Analysis (GPPCA), a latent factor model with an orthogonal loading matrix where each factor follows a GP. The `fmou` module implements the FMOU algorithm as a computationally efficient case of GPPCA, in which each latent factor is modeled by a distinct OU process. Together, these modules provide practical tools for high-dimensional dynamical system modeling.

In Chapter 4, we develop probabilistic forecasting frameworks for nonlinear dynamical systems. We establish that dynamic mode decomposition (DMD) is equivalent to the maximum likelihood estimator of the linear mapping matrix in a linear state-space model. This probabilistic interpretation of DMD enables rigorous uncertainty quantification of forecasts. We further extend the parallel partial Gaussian process to approximate the one-step-ahead transition function and propagate uncertainty through posterior sampling for multi-step forecasting. Numerical examples demonstrate that when model inputs are correctly specified, forecasts are accurate and uncertainty quantification reflects forecast reliability. However, misspecified inputs degrade forecast accuracy and inflate predictive uncertainty.

5.1 Future work on latent factor model

5.1.1 FMOU with irregular missing observations

Although the FMOU framework was originally developed for complete data, extending it to handle irregularly missing observations would significantly broaden its applicability to real-world datasets. One practical approach is to initialize the missing entries using nearby available data and then apply the FMOU algorithm to the completed data. At

each EM iteration, the imputed values are updated using the posterior predictive distribution in Equation (2.20), and the procedure repeats until the convergence of predictive mean or the maximum number of iterations is reached. We summarize this extension in Algorithm 2.

Algorithm 2 FMOU with irregular missing observations

Input: $k \times n$ observation matrix $\mathbf{Y}$, time step $1, 2, \ldots, n$ and selected number of latent processes d. The representer $\mathbf{G}$ and noise level σ_0^2 should be provided if they are fixed.

1: Impute the incomplete data matrix $\mathbf{Y}$ to obtain the completed matrix $\mathbf{Y}^c$.
2: (Optional) Initialize $\mathbf{U}_0$ and σ_0^2 if they are unspecified.
3: Initialize $(\hat{\boldsymbol{\rho}}, \hat{\boldsymbol{\sigma}}^2)$ and then apply KF and RTS smoother to get $\{\mathbf{s}_l(t)\}_{t=1}^n, \{\mathbf{S}_l(t)\}_{t=1}^n$ and $\{\tilde{\mathbf{S}}_l(t)\}_{t=1}^{n-1}$ for $l = 1, \ldots, d$.
4: **while** The convergence criteria were not met and the number of iterations is less than the upper bound **do**
5: (Optional.) If $\mathbf{U}_0$ is unknown, update $\hat{\mathbf{U}}_0$ using Equation (2.16).
6: (Optional.) If σ_0^2 is unknown, update $\hat{\sigma}_0^2$ using Equation (2.17).
7: Update $\hat{\rho}_l$ for $l = 1, \ldots, d$ by solving Equation (2.18).
8: Update $\hat{\sigma}_l^2$ for $l = 1, \ldots, d$ using Equation (2.19).
9: Update $\{\mathbf{s}_l(t)\}_{t=1}^n, \{\mathbf{S}_l(t)\}_{t=1}^n$, $\{\tilde{\mathbf{S}}_l(t)\}_{t=1}^{n-1}$ and $\hat{\mathbf{Z}}$ by KF and RTS smoother.
10: Update $\mathbf{Y}^c$. Specifically, denote $y_j(t)$, as the j-th observation at time t, $j = 1, \ldots, k$ and $t = 1, \ldots, n$. If it is missing, then it is estimated by $m_j(t) + \hat{\epsilon}(t)$, where $m_j(t)$ is the j-th element of $\mathbf{m}(t)$ in Equation (2.20), and $\hat{\epsilon}(t) \sim \mathcal{N}(0, \hat{\sigma}_0^2)$.
11: **end while**

Output: $\boldsymbol{\Theta}^{\text{MMLE}}$, $\mathbf{Z}^{\text{post}}$ and $\mathbf{Y}^c$.

We evaluate the proposed algorithm on the Branin function described in Example 3 (b) of Section 2.4.1. Figure 5.1(a) displays the true mean of the observation. We create the observation by adding independent Gaussian noise of variance $\sigma_0^2 = 1$ to the signal, and removing all observations in the top-left corner, which constitute 16% of the data. Figure 5.1 (b) visualizes the observed data used to fit the model.

We apply Algorithm 2 to this simulated data. Since the noise variance is unknown and the information criterion in Equation (2.23) can not be directly applied to incomplete data, we instead select the number of latent factors via a validation set. Specifically, we randomly split the observed data into training (80%), validation (10%), and testing

(10%) sets. For each candidate value of d, we first fit the models on the training set and then compute the root mean squared error (RMSE) on the validation set. We select the value of d that yields the smallest RMSE. We then refit the model with the selected d using both the training and validation data, and compute the MSE on the testing set to measure the predictive accuracy. Figure 5.1 (c) displays the predictive mean obtained from Algorithm 2 with selected $d = 3$. The test RMSE is 0.174, against a standard deviation of 51 on the true mean, indicating that the algorithm recovers the underlying signal accurately despite the irregular missingness.

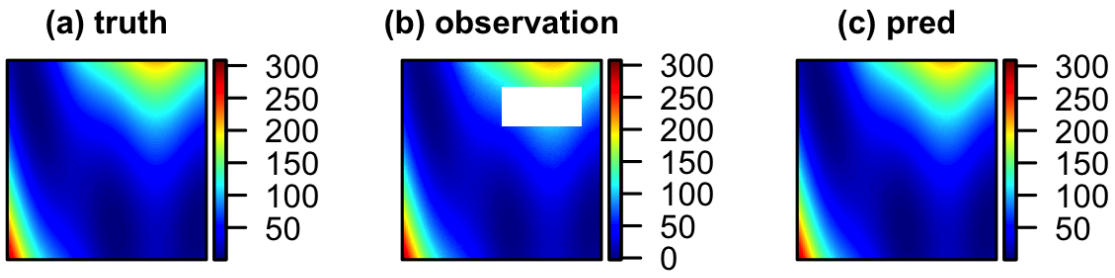

Figure 5.1: (a) True mean of data. (b) Observed data. (c) Predictive mean by Algorithm 2.

5.1.2 Model the latent processes by Matérn kernel with roughness parameter being 2.5

In Chapter 2, the model assumes that each latent process follows a distinct Ornstein-Uhlenbeck (OU) process, which is not mean-squared differentiable. A natural extension is to replace the OU process with a Matérn kernel with $\nu = \frac{5}{2}$, allowing smoother latent

trajectories. In this case, the corresponding dynamical linear model is written as

$$\mathbf{y}(t) = \mathbf{APv}(t) + \boldsymbol{\epsilon}(t) \tag{5.1}$$

$$\mathbf{v}_l(t) = \mathbf{G}_l^* \mathbf{v}_l(t-1) + \mathbf{w}_l^*(t). \tag{5.2}$$

In Equation (5.1), $\mathbf{y}(t) \in \mathbb{R}^k$ is a vector observation at input t, for $t = 1, 2, \ldots, n$. $\mathbf{A}$ is a $k \times d$ orthogonal factor loading matrix. The latent state $\mathbf{v}(t) = [\mathbf{v}_1^\top(t), \ldots, \mathbf{v}_d^\top(t)]^\top \in \mathbb{R}^{3d}$ stacks d independent latent factors. $\mathbf{P}$ is a $d \times 3d$ selection matrix where the $(l, 3l-2)$ entries are 1 for $l = 1, \ldots, d$, and all other entries are zero, meaning that only the first component of each latent factor contributes directly to the observation. $\boldsymbol{\epsilon}(t) \sim \mathcal{MN}(\mathbf{0}, \sigma_0^2 \mathbf{I}_k)$ is the independent Gaussian noise. Equation (5.2) gives the evolution of the lth latent factor, which has a higher-order structure, ensuring that the latent factor is twice differentiable. Specifically, $\mathbf{v}_l(t) = [z_l(t), z_l^{(1)}(t), z_l^{(2)}(t)]^\top$ contains the lth latent factor, its first derivative, and its second derivative at time t, respectively. The dynamics of the latent process are governed by a 3×3 transition matrix $\mathbf{G}_l^*$, given by

$$\mathbf{G}_l^* = \mathbf{L}_{\mathbf{W}_l}^{-1} \mathbf{G}_l \mathbf{L}_{\mathbf{W}_l}, \tag{5.3}$$

where $\mathbf{G}_l$ and $\mathbf{W}_l$ are the special cases of the transition matrix $\mathbf{G}(t_i)$ and the innovation variance $\mathbf{W}(t_i)$ in Equation (1.31) and (1.33), respectively, where $\Delta t = 1$. Both $\mathbf{G}_l$ and $\mathbf{W}_l$ are parameterized by the range parameter γ_l. $\mathbf{L}_{\mathbf{W}}$ is the Cholesky decomposition of $\mathbf{W}_l$. Besides, the distribution matrix of the initial state follows $\mathbf{v}_l(1) \sim \mathcal{MN}(\mathbf{0}, \mathbf{W}_l^*(1))$, where $\mathbf{W}_l^*(1) = \mathbf{L}_{\mathbf{W}_l}^{-1} \mathbf{W}_l(1) \mathbf{L}_{\mathbf{W}_l}$ and $\mathbf{W}_l(1)$ is given in Equation (1.32). Finally, we assume $\mathbf{w}_l(t) \sim \mathcal{MN}(0, \sigma_l^2 \mathbf{I}_3)$ for $t = 2, 3, \ldots, n$.

By using the selection matrix $\mathbf{P}$, Equation (5.1) can be expressed as Equation (1.22), where the factor loading matrix is orthogonal, and the inputs are one-dimensional and

equally spaced. Specifically, $\mathbf{x} = t$ and $\mathbf{z}(t) = \mathbf{P}\mathbf{v}(t) = [z_1(t), \ldots, z_d(t)]^\top$.

Denote $\mathbf{Y} = [\mathbf{y}(1), \ldots, \mathbf{y}(n)]$ as a $k \times n$ observation matrix, $\mathbf{v}_l = [\mathbf{v}_l^\top(1), \ldots \mathbf{v}_l^\top(n)]^\top \in \mathbb{R}^{3n}$ as the lth latent process, and $\mathbf{V} = [\mathbf{v}(1), \ldots, \mathbf{v}(t)]$ as the latent factor matrix. Let $\boldsymbol{\sigma}^2 = (\sigma_1^2, \ldots, \sigma_d^2)$ and $\tilde{\boldsymbol{\gamma}} = (\tilde{\gamma}_1, \ldots, \tilde{\gamma}_d)$, with $\tilde{\gamma}_l = \frac{\sqrt{5}}{\gamma_l}$. The parameters needed to be estimated in Equation (5.1)-(5.2) are denoted as $\boldsymbol{\Theta} = (\mathbf{A}, \sigma_0^2, \boldsymbol{\sigma}^2, \tilde{\boldsymbol{\gamma}})$.

Similar to FMOU, the Expectation-Maximization (EM) algorithm is used in finding the maximum marginal likelihood estimator (MMLE) in this model. It starts with the logarithm of the joint likelihood of $(\mathbf{Y}, \mathbf{V})$:

$$\ell(\boldsymbol{\Theta}) = C - \frac{nk}{2}\log(\sigma_0^2) - \frac{\operatorname{tr}(\mathbf{Y}^\top\mathbf{Y} - 2\mathbf{Y}^\top\mathbf{A}\mathbf{Z})}{2\sigma_0^2} - \sum_{l=1}^{d}\Big(\frac{\mathbf{z}_l^\top\mathbf{z}_l}{2\sigma_0^2} + \frac{\log|\boldsymbol{\Sigma}_l| + \mathbf{v}_l^\top\boldsymbol{\Sigma}_l^{-1}\mathbf{v}_l}{2}\Big), \tag{5.4}$$

where $C = -\frac{(nk+3nd)}{2}\log(2\pi)$ is a constant, $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_d]^\top$ with $\mathbf{z}_l = [z_l(1), \ldots, z_l(n)]^\top$ for $l = 1, \ldots, d$. $\boldsymbol{\Sigma}_l \in \mathbb{R}^{3n\times 3n}$ is the covariance matrix of $\mathbf{v}_l$. By Equation (5.2), $\boldsymbol{\Sigma}_l^{-1}$ has a block tri-diagonal structure and $\log|\boldsymbol{\Sigma}_l| = 3n\log(\sigma_l^2) + \log(|\mathbf{W}_l^*(1)|)$.

In the E step, we calculate the expectation of the joint log-likelihood function with respect to the distribution of $\mathbf{V}$ conditional on observations $\mathbf{Y}$ and the current estimate of parameters, denoted as $\hat{\boldsymbol{\Theta}}$. The posterior distribution of latent factors given $\hat{\boldsymbol{\Theta}}$ follows a multivariate Gaussian distribution such that:

$$(\mathbf{v}_l \mid \mathbf{Y}, \hat{\boldsymbol{\Theta}}) \sim \mathcal{MN}\left(\hat{\mathbf{v}}_l, \hat{\sigma}_0^2\hat{\boldsymbol{\Sigma}}_l(\hat{\sigma}_0^2\mathbf{I}_{3n\times 3n} + \mathbf{P}^*\hat{\boldsymbol{\Sigma}}_l)^{-1}\right), \tag{5.5}$$

where $\hat{\mathbf{v}}_l = \hat{\sigma}_0^2\hat{\boldsymbol{\Sigma}}_l(\hat{\sigma}_0^2\mathbf{I}_{3n\times 3n} + \mathbf{P}^*\hat{\boldsymbol{\Sigma}}_l)^{-1}\tilde{\mathbf{P}}^\top\tilde{\mathbf{y}}_l$ with $\tilde{\mathbf{Y}} = \mathbf{A}^\top\mathbf{Y} = [\tilde{\mathbf{y}}_1, \cdots, \tilde{\mathbf{y}}_d]^\top$ being the projected observation matrix, for $l = 1, \ldots, d$. $\tilde{\mathbf{P}}$ is a $n \times 3n$ matrix with the $(t, 3t-2)$ element being one and others are zero, for $t = 1, \ldots, n$. $\mathbf{P}^* = \tilde{\mathbf{P}}^\top\tilde{\mathbf{P}}$ is a $3n \times 3n$ diagonal matrix where only the $3t-2$ terms in the diagonal is 1 for $t = 1, \ldots, n$, others

are all zeros. We further define $\hat{\mathbf{Z}} = [\hat{\mathbf{z}}_1, \ldots, \hat{\mathbf{z}}_d]^\top$ where $\hat{\mathbf{z}}_l = [\hat{z}_l(1), \ldots, \hat{z}_l(n)]^\top$ with $\hat{z}_l(n) = \mathbb{E}[z_l(t) \mid \mathbf{Y}, \hat{\boldsymbol{\Theta}}]$, and $\hat{\boldsymbol{\Sigma}}_l^{(z)} = \mathrm{Cov}[\mathbf{z}_l \mid \mathbf{Y}, \hat{\boldsymbol{\Theta}}]$ for $l = 1, \ldots, d$. Both $\hat{\mathbf{Z}}$ and $\hat{\boldsymbol{\Sigma}}_l^{(z)}$ can be obtained from Equation (5.5).

Utilizing Equation (5.5), the E-step follows

$$\begin{aligned}
\bar{\ell}(\boldsymbol{\Theta}) &= \mathbb{E}_{\mathbf{V}|\mathbf{Y},\hat{\boldsymbol{\Theta}}}[\ell(\boldsymbol{\Theta})] \\
&= C - \frac{nk}{2}\log(\sigma_0^2) + \frac{\mathrm{tr}(\mathbf{Y}^\top \mathbf{A}\hat{\mathbf{Z}})}{\sigma_0^2} - \sum_{l=1}^{d}\left\{\frac{\log|\boldsymbol{\Sigma}_l|}{2} + \frac{\hat{\mathbf{z}}_l^\top \hat{\mathbf{z}}_l + \mathrm{tr}[\hat{\boldsymbol{\Sigma}}_l^{(z)}]}{2\sigma_0^2}\right\} \\
&\quad - \sum_{l=1}^{d}\frac{\hat{\mathbf{v}}_l^\top \boldsymbol{\Sigma}_l^{-1}\hat{\mathbf{v}}_l + \mathrm{tr}[\hat{\sigma}_0^2\boldsymbol{\Sigma}_l^{-1}\hat{\boldsymbol{\Sigma}}_l(\hat{\sigma}_0^2\mathbf{I}_{3n\times 3n} + \mathbf{P}^*\hat{\boldsymbol{\Sigma}}_l)^{-1}]}{2} - \frac{\mathrm{tr}(\mathbf{Y}^\top\mathbf{Y})}{2\sigma_0^2}.
\end{aligned} \tag{5.6}$$

Directly evaluating Equation (5.6) can be computationally intensive, requiring $\mathcal{O}(dn^3)$ operations due to the need to invert d covariance matrices of size $3n \times 3n$. To overcome this, we treat $\tilde{\mathbf{y}}_l$ as the observations for the l-th latent process. This allow us to apply the Kalman filter (KF) and Rauch–Tung–Striebel (RTS) smoother to each latent factor independently, significantly reducing the complexity to $\mathcal{O}(dn)$. The expressions for this fast computation are provided in Lemma 10.

Lemma 10 *Denote* $\mathbf{s}_l(t) = \mathbb{E}[\mathbf{v}_l(t) \mid \mathbf{Y}, \hat{\boldsymbol{\Theta}}]$, $\mathbf{S}_l(t) = \mathbb{V}[\mathbf{v}_l(t) \mid \mathbf{Y}, \hat{\boldsymbol{\Theta}}]$ *and* $\tilde{\mathbf{S}}_l(t) = Cov[\mathbf{v}_l(t), \mathbf{v}_l(t+1) \mid \mathbf{Y}, \hat{\boldsymbol{\Theta}}]$, *which are obtained by KF and RTS with* $\mathcal{O}(n)$ *for each*

latent process.

$$tr[\hat{\boldsymbol{\Sigma}}_l^{(z)}] = \sum_{t=1}^{n} \mathbb{V}[z_l(t) \mid \mathbf{Y}, \hat{\boldsymbol{\Theta}}], \tag{5.7}$$

$$\begin{aligned} &\hat{\mathbf{v}}_l^\top \boldsymbol{\Sigma}_l^{-1} \hat{\mathbf{v}}_l \\ =&\frac{\mathbf{s}_l^\top(1)\mathbf{W}_l^{-1}(1)\mathbf{s}_l(1) + \sum_{t=2}^{n}(\mathbf{s}_l(t) - \mathbf{G}_l\mathbf{s}_l(t-1))^\top(\mathbf{s}_l(t) - \mathbf{G}_l\mathbf{s}_l(t-1))}{\sigma_l^2}, \end{aligned} \tag{5.8}$$

$$\begin{aligned} &tr[\hat{\sigma}_0^2 \boldsymbol{\Sigma}_l^{-1} \hat{\boldsymbol{\Sigma}}_l (\hat{\sigma}_0^2 \mathbf{I}_{3n\times 3n} + \boldsymbol{\Lambda}\hat{\boldsymbol{\Sigma}}_l)^{-1}] \\ =&\frac{tr[\mathbf{W}_l^{-1}(1)\mathbf{S}_l(1) + \mathbf{G}_l^\top\mathbf{G}_l \sum_{t=1}^{n-1}\mathbf{S}_l(t) + \sum_{t=2}^{n}\mathbf{S}_l(t) - 2\sum_{t=1}^{n-1}\tilde{\mathbf{S}}_l(t)\mathbf{G}_l]}{\sigma_l^2}. \end{aligned} \tag{5.9}$$

In the M-step, $\boldsymbol{\Theta}$ is estimated by maximizing Equation (5.6). Lemma 11 provides the expressions for updating the parameters, where $\mathbf{A}$, σ_0^2 and $\boldsymbol{\sigma}^2$ have closed-form expressions, and $\tilde{\boldsymbol{\gamma}}$ requires numerical optimization.

Lemma 11 *1. Update* $\mathbf{A}$ *by*

$$\hat{\mathbf{A}}^{new} = \tilde{\mathbf{V}}\tilde{\mathbf{U}}^\top, \tag{5.10}$$

where $\tilde{\mathbf{U}}$ *and* $\tilde{\mathbf{V}}$ *are the left and right singular vectors of* $\hat{\mathbf{Z}}\mathbf{Y}^\top$.

2. Update σ_0^2 *by*

$$(\hat{\sigma}_0^2)^{new} = \frac{tr(\mathbf{Y}^\top\mathbf{Y}) - 2tr(\mathbf{Y}^\top\hat{\mathbf{A}}^{new}\hat{\mathbf{Z}}) + \sum_{l=1}^{d}\hat{\mathbf{z}}_l^\top\hat{\mathbf{z}}_l + \sum_{t=1}^{n}\mathbb{V}[z_l(t) \mid \mathbf{Y}, \hat{\boldsymbol{\Theta}}]}{nk}. \tag{5.11}$$

3. Update $\tilde{\boldsymbol{\gamma}}$ by numerical maximizing the objective function below:

$$L(\lambda_l) = -\frac{3n}{2}\log\Big\{ tr\left(\mathbf{W}_l^{*-1}(1)\mathbf{C}_{l,1}\right) + C_{l,4} + tr\left(\mathbf{G}_l^{*\top}\mathbf{G}_l^{*}\mathbf{C}_{l,2}\right) - 2tr[\mathbf{G}_l^{*}\mathbf{C}_{l,3}]\Big\} - \frac{\log(|\mathbf{W}_l^{*}(1)|)}{2} \tag{5.12}$$

where $\mathbf{C}_{l,1} := \mathbf{s}_l(1)\mathbf{s}_l^{\top}(1) + \mathbf{S}_l(1)$, $\mathbf{C}_{l,2} := \sum_{t=2}^{n}\mathbf{s}_l(t-1)\mathbf{s}_l^{\top}(t-1) + \sum_{t=1}^{n-1}\mathbf{S}_l(t)$, $\mathbf{C}_{l,3} := \sum_{t=2}^{n}\mathbf{s}_l(t-1)\mathbf{s}_l^{\top}(t) + \sum_{t=1}^{n-1}\tilde{\mathbf{S}}_l(t)$, $C_{l,4} := \sum_{t=2}^{n}\mathbf{s}_l^{\top}(t)\mathbf{s}_l(t) + tr[\sum_{t=2}^{n}\mathbf{S}_l(t)]$.

4. Update $\boldsymbol{\sigma}^2$ by:

$$\begin{aligned}(\hat{\sigma}_l^2)^{new} = &\frac{\mathbf{s}_l^{\top}(1)(\mathbf{W}_l^{*-1}(1))^{new}\mathbf{s}_l(1) + (\mathbf{W}_l^{*-1}(1))^{new}\mathbf{S}_l(1)}{3n} \\ &+ \frac{(\mathbf{G}_l^{*\top})^{new}(\mathbf{G}_l^{*})^{new}\sum_{t=1}^{n-1}\mathbf{S}_l(t) + \sum_{t=2}^{n}\mathbf{S}_l(t) - 2\sum_{t=1}^{n-1}\tilde{\mathbf{S}}_l(t)(\mathbf{G}_l^{*})^{new}}{3n} \\ &+ \frac{\sum_{t=2}^{n}\Big(\mathbf{s}_l(t) - (\mathbf{G}_l^{*})^{new}\mathbf{s}_l(t-1)\Big)^{\top}\Big(\mathbf{s}_l(t) - (\mathbf{G}_l^{*})^{new}\mathbf{s}_l(t-1)\Big)}{3n}\end{aligned} \tag{5.13}$$

Although the M-step requires numerical optimization for $\tilde{\boldsymbol{\gamma}}$, we emphasize that the coefficients $\mathbf{C}_{l,1}$, $\mathbf{C}_{l,2}$, $\mathbf{C}_{l,3}$ and $C_{l,4}$ remain fixed throughout the iteration. These can be precomputed with computational complexity of $\mathcal{O}(n)$. Therefore, each evaluation of $L(\tilde{\gamma}_l)$ incurs only $\mathcal{O}(1)$ cost, bringing the total complexity for updating $\tilde{\gamma}_l$ to $\mathcal{O}(n + M_\gamma)$, where M_γ denotes the number of iterations for numerical convergence.

5.1.3 Kernel-induced factor loading matrix

In FMOU, the factor loading matrix $\mathbf{U}_0$ is estimated as a collection of free parameters, leading to a storage of $\mathcal{O}(kd)$, which is expensive for high-dimensional observations. To mitigate this challenge, we may consider a constraint where the columns of $\mathbf{U}_0$ are the first d leading eigenvectors of a spatial correlation matrix $\mathbf{R}_s(\lambda_s) \in \mathbb{R}^{k\times k}$. $\mathbf{R}_s$ is constructed

using a spatial kernel function such as exponential or Matérn 5/2, and it is parameterized by a range parameter λ_s, which characterizes the spatial correlation between k output coordinates. This assumption effectively reduces the number of parameters required to represent the loading matrix from kd to a single scalar. We refer to this model as FMOU with Kernel-induced loading (FMOU-K).

Under this construction, the joint log-likelihood and the resulting E-step remain mathematically identical to Equation (2.9) and (2.11). The quantities obtained from the posterior distribution of $\mathbf{z}_l$ are still effectively computed via the KF and RTS smoother in $\mathcal{O}(dn)$, as introduced in Lemma 7. The primary difference lies in the M-step update for $\mathbf{U}_0$. Instead of the SVD-based update shown in Equation (2.16), we optimize the expected log-likelihood with respect to λ_s. Isolating the terms in Equation (2.11) that depend on $\mathbf{U}_0$, the optimization problem becomes:

$$\hat{\lambda}_s = \arg\max_{\lambda_s} \operatorname{tr}\left(\mathbf{Y}^\top \mathbf{U}_0(\lambda_s)\hat{\mathbf{Z}}\right), \tag{5.14}$$

subject to the eigendecomposition $\mathbf{R}_s(\lambda_s) = \tilde{\mathbf{U}}_R \tilde{\mathbf{\Lambda}}_R \tilde{\mathbf{U}}_R^\top$, where $\mathbf{U}_0(\lambda_s)$ comprises the first d columns of $\tilde{\mathbf{U}}_R$.

Although Equation (5.14) requires numerical optimization, $\mathbf{U}_0(\lambda_s)$ can be reconstructed by the spatial coordinates and λ_s. Therefore, the system no longer needs to store a $k \times d$ matrix. This is particularly beneficial in high-dimensional settings.

5.2 Future work on probabilistic forecast of nonlinear dynamical systems with uncertainty quantification

In chapter 4, we demonstrate that the PP-GP can accurately approximate the one-step-ahead transition function and provide reliable probabilistic forecasts for vector-valued outputs when the inputs are correctly specified. However, when the input dimensions are large, the model faces both the curse of dimensionality and a computational bottleneck. A promising future direction is therefore to develop data reduction methods, such as the variational autoencoder, to construct a low-dimensional representation of the inputs, along with a suitable distance metric on the reduced space that preserves the predictive power of the PP-GP.

Another limitation of the PP-GP framework in chapter 4 is its Markov assumption, which restricts the input to the immediately preceding time step and may be insufficient for dynamical systems with long-range temporal dependencies. Although one can stack observations from several previous time steps to form an augmented input, it may suffer from the curse of dimensionality. An alternative is to leverage the transformer-based models [9] for forecasting nonlinear dynamical systems. Since both natural language and time series are sequence data, Large Language Models (LLMs) are well-suited to model correlated data. For example, GPT4TS [162] utilizes pretrained transformer parameters and only fine-tunes the positional embeddings and layer normalization layer, achieving comparable performance across a range of time series tasks, including short/long-term forecasting. Another example is Mamba [163], which is a selective state space model that captures long-range dependencies in sequential data while maintaining near-linear computational complexity. Its computational efficiency and ability to model long sequences

make it a promising approach for time series forecasting. A natural and important extension of these approaches is to incorporate uncertainty quantification into the forecasting frameworks, such as by modeling the parameters of predictive distributions alongside point forecasts.

Integration of multiple heterogeneous data sources to enrich the available information for nonlinear dynamical systems is also a promising direction. In many real-world applications, the dynamics of a system are governed not only by its historical data but also by external signals. For example, in weather forecasting, future temperature or precipitation may depend not only on historical weather data, but also on satellite images and radar maps. Future work could develop multi-source fusion frameworks that combine physical observations with auxiliary data that can improve forecasting accuracy and uncertainty quantification.

Appendix A

Appendix of Chapter 1

A.1 Proof of Lemma 1

Proof: For the one-step-ahead predictive distribution of $\mathbf{z}(t_i)$ given $\mathbf{y}_{1:(i-1)}$,

$$\begin{aligned}
\mathbf{a}(t_i) &= \mathbb{E}[\mathbf{z}(t_i) \mid \mathbf{y}_{1:(i-1)}] = \mathbb{E}[\mathbb{E}[\mathbf{z}(t_i) \mid \mathbf{z}(t_{i-1}), \mathbf{y}_{1:(i-1)}]] = \mathbf{G}(t_i)\mathbf{m}(t_{i-1}), \\
\mathbf{R}(t_i) &= \mathbb{V}[\mathbf{z}(t_i) \mid \mathbf{y}_{1:(i-1)}] \\
&= \mathbb{V}[\mathbb{E}[\mathbf{z}(t_i) \mid \mathbf{z}(t_{i-1}), \mathbf{y}_{1:(i-1)}]] + \mathbb{E}[\mathbb{V}[\mathbf{z}(t_i) \mid \mathbf{z}(t_{i-1}), \mathbf{y}_{1:(i-1)}]] \\
&= \mathbf{G}(t_i)\mathbf{C}(t_{i-1})\mathbf{G}^\top(t_i) + \mathbf{W}(t_i).
\end{aligned}$$

For the one-step-ahead predictive distribution of $\tilde{y}(t)$ given $\tilde{\mathbf{y}}_{1:t-1}$,

$$\begin{aligned}
f(t_i) &= \mathbb{E}[y(t_i) \mid \mathbf{y}_{1:(i-1)}] = \mathbb{E}[\mathbb{E}[y(t_i) \mid \mathbf{z}(t_i), \mathbf{y}_{1:(i-1)}]] = \mathbf{F}(t_i)\mathbf{a}(t_i) \\
Q(t_i) &= \mathbb{V}[y(t_i) \mid \mathbf{y}_{1:(i-1)}] \\
&= \mathbb{V}[\mathbb{E}[y(t_i) \mid \mathbf{z}(t_i), \mathbf{y}_{1:(i-1)}]] + \mathbb{E}[\mathbb{V}[y(t) \mid \mathbf{z}(t_i), \mathbf{y}_{1:(i-1)}]] \\
&= \mathbf{F}(t_i)\mathbf{R}(t_i)\mathbf{F}^\top(t_i) + V(t_i).
\end{aligned}$$

For the filtering distribution of $\mathbf{z}(t_i)$ given $\mathbf{y}_{1:(i-1)}$, we start from the conditional joint distribution of $(y(t_i), \mathbf{z}(t_i))^\top$ given $\mathbf{y}_{1:(i-1)}$:

$$\begin{pmatrix} y(t_i) \\ \mathbf{z}(t_i) \end{pmatrix} \mid \mathbf{y}_{1:(i-1)} \sim \mathcal{MN}\left(\begin{pmatrix} f(t_i) \\ \mathbf{a}(t_i) \end{pmatrix}, \begin{pmatrix} Q(t_i) & \mathbf{F}(t_i)\mathbf{R}(t_i) \\ \mathbf{R}(t_i)\mathbf{F}^\top(t_i) & \mathbf{R}(t_i) \end{pmatrix} \right).$$

Thus, by the conditional distributions of the normal distribution, we have

$$\begin{aligned} \mathbf{m}(t_i) &= \mathbb{E}[\mathbf{z}(t_i) \mid \mathbf{y}_{1:(i-1)}] = \mathbf{a}(t_i) + \mathbf{R}(t_i)Q^{-1}(t_i)(y(t_i) - f(t_i)), \\ \mathbf{C}(t_i) &= \mathbb{V}[\mathbf{z}(t_i) \mid \mathbf{y}_{1:(i-1)}] = \mathbf{R}(t_i) - \mathbf{R}(t_i)Q^{-1}(t_i)\mathbf{R}(t_i). \end{aligned}$$

■

A.2 Proof of Lemma 2

Proof: We first show the derivation of $\mathbf{s}(t_i)$ and $\mathbf{S}(t_i)$. The conditional distribution of $(\mathbf{z}(t_{i+1}), \mathbf{z}(t_i))^\top$ given $\mathbf{y}_{1:i}$ is:

$$\begin{pmatrix} \mathbf{z}(t_{i+1}) \\ \mathbf{z}(t_i) \end{pmatrix} \mid \mathbf{y}_{1:i} \sim \mathcal{MN}\left(\begin{pmatrix} \mathbf{a}(t_{i+1}) \\ \mathbf{m}(t_i) \end{pmatrix}, \begin{pmatrix} \mathbf{R}(t_{i+1}) & \mathbf{G}(t_{i+1})\mathbf{C}(t) \\ \mathbf{C}(t_i)\mathbf{G}^\top(t_{i+1}) & \mathbf{C}(t_i) \end{pmatrix} \right).$$

Therefore,

$$\begin{aligned} \mathbb{E}[\mathbf{z}(t_i) \mid \mathbf{z}(t_{i+1}), \mathbf{y}_{1:i}] &= \mathbf{m}(t_i) + \mathbf{C}(t_i)\mathbf{G}^\top(t_{i+1})\mathbf{R}^{-1}(t_{i+1})(\mathbf{z}(t_{i+1}) - \mathbf{a}(t_{i+1})) \\ \mathbb{V}[\mathbf{z}(t_i) \mid \mathbf{z}(t_{i+1}), \mathbf{y}_{1:i}] &= \mathbf{C}(t_i) - \mathbf{C}(t_i)\mathbf{G}^\top(t_{i+1})\mathbf{R}^{-1}(t_{i+1})\mathbf{C}(t_i), \end{aligned}$$

from which one has

$$\begin{aligned}\mathbf{s}(t_i) &= \mathbb{E}[\mathbf{z}(t_i) \mid \mathbf{y}_{1:n}] \\ &= \mathbb{E}[\mathbb{E}[\mathbf{z}(t_i) \mid \mathbf{z}(t_{i+1}), \mathbf{y}_{1:n}]] \\ &= \mathbf{m}(t_i) + \mathbf{C}(t_i)\mathbf{G}^\top(t_{i+1})\mathbf{R}^{-1}(t_{i+1})(\mathbf{s}(t_{i+1}) - \mathbf{a}(t_{i+1})),\end{aligned}$$

and

$$\begin{aligned}\mathbf{S}(t_i) &= \mathbb{V}[\mathbf{z}(t_i) \mid \mathbf{y}_{1:n}] \\ &= \mathbb{V}[\mathbb{E}[\mathbf{z}(t_i) \mid \mathbf{z}(t_{i+1}), \mathbf{y}_{1:n}]] + \mathbb{E}[\mathbb{V}[\mathbf{z}(t_i) \mid \mathbf{z}(t_{i+1}), \mathbf{y}_{1:n}]] \\ &= \mathbf{C}(t_i) - \mathbf{C}(t_i)\mathbf{G}^\top(t_{i+1})\mathbf{R}^{-1}(t_{i+1})(\mathbf{R}(t_{i+1}) - \mathbf{S}(t_{i+1})\mathbf{R}^{-1}(t_{i+1})\mathbf{G}(t_{i+1})\mathbf{C}(t_i).\end{aligned}$$

For the covariance between $\mathbf{z}(t_i)$ and $\mathbf{z}(t_{i+1})$ conditional on $\mathbf{y}_{1:n}$,

$$\begin{aligned}\tilde{\mathbf{S}}(t_i) &= \mathrm{Cov}[\mathbf{z}(t_i), \mathbf{z}(t_{i+1}) \mid \mathbf{y}_{1:n}] \\ &= \mathbb{E}[\mathrm{Cov}[\mathbf{z}(t_i), \mathbf{z}(t_{i+1}) \mid \mathbf{z}(t_{i+1}), \mathbf{y}_{1:n}]] \\ &\quad + \mathrm{Cov}[\mathbb{E}[\mathbf{z}(t_i) \mid \mathbf{z}(t_{i+1}), \mathbf{y}_{1:n}], \mathbb{E}[\mathbf{z}(t_{i+1}) \mid \mathbf{z}(t_{i+1}), \mathbf{y}_{1:n}]] \\ &= \mathbf{C}(t_i)\mathbf{G}^\top(t_{i+1})\mathbf{R}^{-1}(t_{i+1})\mathbf{S}(t_{i+1}).\end{aligned}$$

We finish the proof. ■

Appendix B

Appendix of Chapter 2

B.1 Derivations for Lemmas 4-6

B.1.1 Proof of Lemma 4

Before proving Lemma 2.1, we introduce three auxiliary facts.

1. Let $\mathbf{A}$ and $\mathbf{B}$ be $a \times a$ and $b \times b$ square matrices, respectively. Then the Kronecker sum is defined as

$$\mathbf{A} \oplus \mathbf{B} = \mathbf{A} \otimes \mathbf{I}_a + \mathbf{I}_b \otimes \mathbf{B},$$

   where $\otimes$ denotes the Kronecker product, $\mathbf{I}_a$ and $\mathbf{I}_b$ are identity matrices with sizes $a \times a$ and $b \times b$, respectively.

2. Let $\mathbf{A}$ be a matrix,

$$e^{-(\mathbf{A} \oplus \mathbf{A})} = e^{(-\mathbf{A}) \oplus (-\mathbf{A})} = e^{-\mathbf{A}} \otimes e^{-\mathbf{A}}.$$

3. For matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$,

$$(\mathbf{C}^\top \otimes \mathbf{A})\text{vec}(\mathbf{B}) = \text{vec}(\mathbf{ABC}).$$

Now we start to prove Lemma 2.1.

Proof: The discretization of the multivariate Ornstein-Uhlenbeck process described in Equation (2.5) can be written as [93]

$$\mathbf{m}(t+1) = e^{-\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top}\mathbf{m}(t) + \boldsymbol{\varepsilon}(t+1), \tag{B.1}$$

where $\boldsymbol{\varepsilon}(t+1) \sim \mathcal{MN}(\mathbf{0}, \boldsymbol{\Sigma}_\varepsilon)$ with $\boldsymbol{\Sigma}_\varepsilon$ satisfying

$$\left((\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top)\right) \oplus (\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top)\right) \text{vec}(\boldsymbol{\Sigma}_\varepsilon) = \left(\mathbf{I} - e^{-\left((\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top)\right)\oplus(\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top)\right)}\right) \text{vec}(\mathbf{U}_0\tilde{\mathbf{D}}^2\mathbf{U}_0^\top).$$

According to Equations (2.3)-(2.4), the evolution of the mean of the observation in our model can be expressed in a vector form:

$$\mathbf{m}(t+1) = \mathbf{C}\mathbf{m}(t) + \tilde{\mathbf{w}}(t+1), \tag{B.2}$$

where $\mathbf{C} = \mathbf{U}_0\text{diag}(\rho_1, \ldots, \rho_d)\mathbf{U}_0^\top$ and $\tilde{\mathbf{w}}(t+1) = \mathbf{U}_0\text{diag}(w_1(t+1), \ldots, w_d(t+1)) \sim \mathcal{MN}(\mathbf{0}, \boldsymbol{\Sigma}_{\tilde{w}})$, with $w_l(t+1) \sim \mathcal{N}(0, \sigma_l^2)$ being the innovation of the lth latent factor and $\boldsymbol{\Sigma}_{\tilde{w}} = \mathbf{U}_0\text{diag}(\sigma_1^2, \ldots, \sigma_d^2)\mathbf{U}_0^\top$.

We will show that Equation (B.1) and Equation (B.2) are equivalent. First of all, as $\mathbf{D} = \text{diag}(-\log(\rho_1), \ldots, -\log(\rho_d))$, we have

$$e^{-\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top} = \sum_{q=0}^{\infty} \frac{1}{q!}\left(-\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top\right)^q = \mathbf{U}_0\left(\sum_{q=0}^{\infty}\frac{(-\mathbf{D})^q}{q!}\right)\mathbf{U}_0^\top = \mathbf{U}_0\text{diag}(\rho_1, \ldots, \rho_d)\mathbf{U}_0^\top = \mathbf{C},$$

where the second equality holds as $\mathbf{U}_0^\top \mathbf{U}_0 = \mathbf{I}_d$ and the third equality holds because $\mathbf{D}$ is a diagonal matrix, and $\sum_{q=0}^{\infty} \frac{\log(\rho_l)^q}{q!} = e^{\log(\rho_l)} = \rho_l$ by the Taylor expansion.

Second, we show $\boldsymbol{\Sigma}_\varepsilon = \boldsymbol{\Sigma}_{\tilde{w}}$ when $\tilde{\mathbf{D}} = \text{diag}\left(\sqrt{-2\frac{\sigma_1^2 \log(\rho_1)}{1-\rho_1^2}}, \ldots, \sqrt{-2\frac{\sigma_d^2 \log(\rho_d)}{1-\rho_d^2}}\right)$. This is equivalent to proving the following:

$$\left((\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top)) \oplus (\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top)\right) \text{vec}\left(\boldsymbol{\Sigma}_{\tilde{w}}\right) = \left(\mathbf{I} - e^{-\left((\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top)) \oplus (\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top)\right)}\right) \text{vec}(\mathbf{U}_0\tilde{\mathbf{D}}\tilde{\mathbf{D}}^\top\mathbf{U}_0^\top). \tag{B.3}$$

The left-hand side of Equation (B.3) can be expressed as:

$$\begin{aligned}
&\left((\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top)) \oplus (\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top)\right) \text{vec}\left(\boldsymbol{\Sigma}_{\tilde{w}}\right) \\
=&\left((\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top) \otimes \mathbf{I}_k + \mathbf{I}_k \otimes (\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top)\right) \text{vec}\left(\mathbf{U}_0 \text{diag}(\sigma_1^2, \ldots, \sigma_d^2)\mathbf{U}_0^\top\right) \\
=&\text{vec}\left(\mathbf{U}_0\Big(\mathbf{D}\text{diag}(\sigma_1^2, \ldots, \sigma_d^2) + \mathbf{D}\text{diag}(\sigma_1^2, \ldots, \sigma_d^2)\Big)\mathbf{U}_0^\top\right) \\
=&\text{vec}\left(\mathbf{U}_0 \text{diag}(-2\log(\rho_1)\sigma_1^2, \ldots, -2\log(\rho_d)\sigma_d^2)\mathbf{U}_0^\top\right),
\end{aligned}$$

where the first and second equalities use auxiliary Facts 1 and 3, respectively.

The right-hand side of Equation (B.3) can be expressed as:

$$\begin{aligned}
&\left(\mathbf{I} - e^{-\left((\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top)) \oplus (\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top)\right)}\right) \text{vec}(\mathbf{U}_0\tilde{\mathbf{D}}\tilde{\mathbf{D}}^\top\mathbf{U}_0^\top) \\
=&\text{vec}(\mathbf{U}_0\tilde{\mathbf{D}}\tilde{\mathbf{D}}^\top\mathbf{U}_0^\top) - \left(e^{-\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top} \otimes e^{-\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top}\right) \text{vec}(\mathbf{U}_0\tilde{\mathbf{D}}\tilde{\mathbf{D}}^\top\mathbf{U}_0^\top) \\
=&\text{vec}(\mathbf{U}_0\tilde{\mathbf{D}}\tilde{\mathbf{D}}^\top\mathbf{U}_0^\top) - \text{vec}(\mathbf{C}\mathbf{U}_0\tilde{\mathbf{D}}\tilde{\mathbf{D}}^\top\mathbf{U}_0^\top\mathbf{C}) \\
=&\text{vec}\left(\mathbf{U}_0\Big(\tilde{\mathbf{D}}\tilde{\mathbf{D}}^\top - \text{diag}(\rho_1, \ldots, \rho_d)\tilde{\mathbf{D}}\tilde{\mathbf{D}}^\top\text{diag}(\rho_1, \ldots, \rho_d)\Big)\mathbf{U}_0^\top\right) \\
=&\text{vec}\left(\mathbf{U}_0 \text{diag}(-2\log(\rho_1)\sigma_1^2, \ldots, -2\log(\rho_d)\sigma_d^2)\mathbf{U}_0^\top\right),
\end{aligned}$$

where the first equality utilizes auxiliary Fact 2, and the second equality is by Fact 3 and $\mathbf{C} = e^{-\mathbf{U}_0\mathbf{D}\mathbf{U}_0^\top}$. Therefore, we have shown that the left and right-hand sides in Equation (B.3) are the same. As a result, Equations (B.1) and (B.2) are equivalent. We finish the proof. ■

B.1.2 Proof of Lemma 5

Proof: We first show that $\mathbb{V}[z_l(t)] = \rho_l^{2(t-1)}\tau_l^2 + \frac{1-\rho_l^{2(t-1)}}{1-\rho_l^2}\sigma_l^2$ for $t = 1, 2, \ldots, n$. We prove this by induction. For t=1, we have:

$$\mathbb{V}[z_l(1)] = \rho_l^0\tau_l^2 + \frac{1-\rho_l^2}{1-\rho_l^2}\sigma_l^2 = \tau_l^2.$$

Assume for $t > 1$, we have

$$\mathbb{V}[z_l(t)] = \rho_l^{2(t-1)}\tau_l^2 + \frac{1-\rho_l^{2(t-1)}}{1-\rho_l^2}\sigma_l^2. \tag{B.4}$$

Note for $t' > t$, Equation (B.4) leads to

$$\begin{aligned}
\mathrm{Cov}[z_l(t), z_l(t')] &= \mathrm{Cov}[z_l(t), \rho_l z_l(t'-1) + w_l(t')] \\
&= \rho_l \mathrm{Cov}[z_l(t), z_l(t'-1)] \\
&= \rho_l^{t'-t}\mathrm{Cov}[z_l(t), z_l(t)] \\
&= \rho_l^{t'-t}\left(\rho_l^{2(t-1)}\tau_l^2 + \frac{1-\rho_l^{2(t-1)}}{1-\rho_l^2}\sigma_l^2\right).
\end{aligned}$$

For any $t+1$, by Equation (2.4) and the law of total variance, the variance of $z_l(t+1)$

follows

$$\begin{aligned}\mathbb{V}[z_l(t+1)] &= \mathbb{V}[\mathbb{E}(z_l(t+1)|z_l(t))] + \mathbb{E}[\mathbb{V}(z_l(t+1)|z_l(t))] \\ &= \rho_l^2\left(\rho_l^{2(t-1)}\tau_l^2 + \frac{1-\rho_l^{2(t-1)}}{1-\rho_l^2}\sigma_l^2\right) + \sigma_l^2 \\ &= \rho_l^{2t}\tau_l^2 + \frac{1-\rho_l^{2t}}{1-\rho_l^2}\sigma_l^2.\end{aligned}$$

For $t' > t+1$,

$$\begin{aligned}\mathrm{Cov}[z_l(t+1), z_l(t')] &= \mathrm{Cov}[z_l(t+1), \rho_l z_l(t'-1) + w_l(t')] \\ &= \rho_l \mathrm{Cov}[z_l(t+1), z_l(t'-1)] \\ &= \rho_l^{t'-t-1}\mathrm{Cov}[z_l(t+1), z_l(t+1)] \\ &= \rho_l^{t'-t-1}\left(\rho_l^{2t}\tau_l^2 + \frac{1-\rho_l^{2t}}{1-\rho_l^2}\sigma_l^2\right),\end{aligned}$$

satisfying the format of Equation (2.6). Since we have shown both the base case and the inductive step, Lemma 2.2 is true. We finish the proof.

When $\tau_l^2 = \frac{\sigma_l^2}{1-\rho_l^2}$, one can plug it into Equation (2.6) to obtain Equation (2.7). ■

B.1.3 Proof of Lemma 6

Proof: The tri-diagonal structure of $\mathbf{R}_l^{-1}$ can be proved by showing $\mathbf{R}_l\mathbf{R}_l^{-1} = \mathbf{I}_n$, where $\mathbf{I}_n$ is a $n \times n$ identity matrix.

To compute $|\mathbf{R}_l|$, we use the Cholesky decomposition of $\mathbf{R}_l$ given by $\mathbf{R}_l = \mathbf{L}_l\mathbf{L}_l^\top$,

where the Cholesky factor $\mathbf{L}_l$ has the form

$$\mathbf{L}_l = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ \rho_l & \sqrt{1-\rho_l^2} & 0 & \dots & 0 \\ \rho_l^2 & \rho_l\sqrt{1-\rho_l^2} & \sqrt{1-\rho_l^2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \rho_l^{n-1} & \rho_l^{n-2}\sqrt{1-\rho_l^2} & \rho_l^{n-3}\sqrt{1-\rho_l^3} & \dots & \sqrt{1-\rho_l^2} \end{pmatrix}.$$

Since $\mathbf{L}_l$ is a lower triangular matrix, $|\mathbf{L}_l| = |\mathbf{L}_l^\top| = (1-\rho_l^2)^{(n-1)/2}$. Therefore, $|\mathbf{R}_l| = |\mathbf{L}_l||\mathbf{L}_l^\top| = (1-\rho_l^2)^{(n-1)}$ and $|\mathbf{R}_l^{-1}| = \frac{1}{(1-\rho_l^2)^{n-1}}$. By utilizing the determinant of $\mathbf{R}_l$, we have $|\mathbf{\Sigma}_l| = (\frac{\sigma_l^2}{1-\rho_l^2})^n|\mathbf{R}_l| = \frac{\sigma_l^{2n}}{1-\rho_l^2}$. We finish the proof. ■

B.2 Derivation for Section 2.2.3

We are ready to show the derivation of the Expectation-Maximization (EM) algorithm. We start from a general scenario where the parameters $\mathbf{\Theta} = (\mathbf{U}_0, \sigma_0^2, \boldsymbol{\sigma}^2, \boldsymbol{\rho})$ are unknown. Then we modify our algorithm to special applications where $\mathbf{U}_0$ and σ_0^2 are given.

B.2.1 Proof of Equation 2.9

The EM algorithm begins with the natural logarithm of the joint likelihood of the observations $\mathbf{Y}$ and the latent factors $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_d]^\top$.

$$
\begin{aligned}
\ell(\boldsymbol{\Theta}) =& \log\left(p(\mathbf{Y}, \mathbf{Z} \mid \boldsymbol{\Theta})\right) \\
=& \log\left(p(\mathbf{Y} \mid \mathbf{Z}, \boldsymbol{\Theta})\right) + \log\left(p(\mathbf{Z} \mid \boldsymbol{\Theta})\right) \\
=& C - \frac{nk}{2}\log(\sigma_0^2) - \sum_{t=1}^{n} \frac{(\mathbf{y}(t) - \mathbf{U}_0\mathbf{z}(t))^\top(\mathbf{y}(t) - \mathbf{U}_0\mathbf{z}(t))}{2\sigma_0^2} \\
& - \sum_{l=1}^{d}\left(\frac{\log|\boldsymbol{\Sigma}_l| + \mathbf{z}_l^\top\boldsymbol{\Sigma}_l^{-1}\mathbf{z}_l}{2}\right) \\
=& C - \frac{nk}{2}\log(\sigma_0^2) - \frac{\mathrm{tr}(\mathbf{Y}^\top\mathbf{Y} - 2\mathbf{Y}^\top\mathbf{U}_0\mathbf{Z})}{2\sigma_0^2} - \sum_{l=1}^{d}\Big(\frac{\mathbf{z}_l^\top\mathbf{z}_l}{2\sigma_0^2} + \frac{\log|\boldsymbol{\Sigma}_l| + \mathbf{z}_l^\top\boldsymbol{\Sigma}_l^{-1}\mathbf{z}_l}{2}\Big),
\end{aligned}
$$

where $C = -\frac{(nk+nd)}{2}\log(2\pi)$ is a constant.

B.2.2 Proof of Equation 2.11

In the Expectation (E) step we calculate the expectation of Equation (2.9) with respect to the distribution of latent factor $\mathbf{Z}$, conditional on data $\mathbf{Y}$ and the current estimated parameters $\hat{\boldsymbol{\Theta}} = (\hat{\mathbf{U}}_0, \hat{\sigma}_0^2, \hat{\boldsymbol{\rho}}, \hat{\boldsymbol{\sigma}}^2)$. Making use of the expectation of quadratic

forms, we obtain the expectation of $\ell(\mathbf{\Theta})$ as follows:

$$\begin{aligned}
\bar{\ell}(\mathbf{\Theta}) :&= \mathbb{E}_{\mathbf{Z}|\mathbf{Y},\hat{\mathbf{\Theta}}}[\ell(\mathbf{\Theta})] \\
&= C - \frac{nk}{2}\log(\sigma_0^2) - \frac{\mathrm{tr}(\mathbf{Y}^\top\mathbf{Y})}{2\sigma_0^2} + \frac{\mathbb{E}_{\mathbf{Z}|\mathbf{Y},\hat{\mathbf{\Theta}}}[\mathrm{tr}(\mathbf{Y}^\top\mathbf{U}_0\mathbf{Z})]}{\sigma_0^2} - \sum_{l=1}^{d}\frac{\mathbb{E}_{\mathbf{Z}|\mathbf{Y},\hat{\mathbf{\Theta}}}[\mathbf{z}_l^\top\mathbf{z}_l]}{2\sigma_0^2} \\
&\quad - \sum_{l=1}^{d}\frac{\log|\mathbf{\Sigma}_l| + \mathbb{E}_{\mathbf{Z}|\mathbf{Y},\hat{\mathbf{\Theta}}}[\mathbf{z}_l^\top\mathbf{\Sigma}_l^{-1}\mathbf{z}_l]}{2} \\
&= C - \frac{nk}{2}\log(\sigma_0^2) - \frac{\mathrm{tr}(\mathbf{Y}^\top\mathbf{Y})}{2\sigma_0^2} + \frac{\mathrm{tr}(\mathbf{Y}^\top\mathbf{U}_0\hat{\mathbf{Z}})}{\sigma_0^2} - \sum_{l=1}^{d}\frac{\hat{\mathbf{z}}_l^\top\hat{\mathbf{z}}_l + \mathrm{tr}[\hat{\sigma}_0^2\hat{\mathbf{\Sigma}}_l(\hat{\mathbf{\Sigma}}_l + \hat{\sigma}_0^2\mathbf{I}_n)^{-1}]}{2\sigma_0^2} \\
&\quad - \sum_{l=1}^{d}\frac{\log|\mathbf{\Sigma}_l| + \hat{\mathbf{z}}_l^\top\mathbf{\Sigma}_l^{-1}\hat{\mathbf{z}}_l + \mathrm{tr}[\hat{\sigma}_0^2\mathbf{\Sigma}_l^{-1}\hat{\mathbf{\Sigma}}_l(\hat{\mathbf{\Sigma}}_l + \hat{\sigma}_0^2\mathbf{I}_n)^{-1}]}{2},
\end{aligned}$$

where $\hat{\mathbf{z}}_l = \mathbb{E}[\mathbf{z}_l \mid \mathbf{Y}, \hat{\mathbf{\Theta}}]$, $\hat{\mathbf{Z}} = [\hat{\mathbf{z}}_1, \ldots, \hat{\mathbf{z}}_d]^\top$.

B.2.3 Proof of Lemma 7

Proof: First, for $l = 1, 2, \ldots, d$, by definition of KF and RTS smoother, one has

$$\begin{aligned}
\hat{\mathbf{Z}} = [\hat{\mathbf{z}}_1, \ldots, \hat{\mathbf{z}}_d]^\top, \quad \text{with } \hat{\mathbf{z}}_l = [s_l(1), s_l(2), \ldots, s_l(n)]^\top, \\
\mathrm{tr}[\hat{\sigma}_0^2\hat{\mathbf{\Sigma}}_l(\hat{\mathbf{\Sigma}}_l + \hat{\sigma}_0^2\mathbf{I}_n)^{-1}] = \sum_{t=1}^{n}\mathbb{V}[\mathbf{z}_l(t) \mid \mathbf{Y}, \hat{\mathbf{\Theta}}] = \sum_{t=1}^{n} S_l(t),
\end{aligned}$$

where $s_l(t) = \mathbb{E}[z_l(t) \mid \mathbf{Y}, \hat{\mathbf{\Theta}}]$ and $S_l(t) = \mathbb{V}[z_l(t) \mid \mathbf{Y}, \hat{\mathbf{\Theta}}]$ for $t = 1, \ldots, n$ can be obtained by applying KF and RTS smoother to the l-th latent process with complexity $\mathcal{O}(n)$.

Second, by utilizing the tri-diagonal structure of $\mathbf{\Sigma}_l^{-1}$, as shown in Lemma 7, one has

$$\begin{aligned}\hat{\mathbf{z}}_l^\top \mathbf{\Sigma}_l^{-1} \hat{\mathbf{z}}_l &= \frac{1}{\sigma_l^2} \begin{pmatrix} s_l(1) \\ s_l(2) \\ \vdots \\ \vdots \\ s_l(n) \end{pmatrix}^\top \begin{pmatrix} 1 & -\rho_l & 0 & 0 & \cdots \\ -\rho_l & 1+\rho_l^2 & -\rho_l & 0 & \cdots \\ 0 & -\rho_l & 1+\rho_l^2 & \ddots & \\ \vdots & \vdots & \ddots & \ddots & -\rho_l \\ 0 & 0 & \cdots & -\rho_l & 1 \end{pmatrix} \begin{pmatrix} s_l(1) \\ s_l(2) \\ \vdots \\ \vdots \\ s_l(n) \end{pmatrix} \\ &= \frac{1}{\sigma_l^2} \left((1-\rho_l^2) s_l^2(1) + \sum_{t=2}^{n} (s_l(t) - \rho_l s_l(t-1))^2 \right).\end{aligned}$$

$$\begin{aligned}&\mathrm{tr}[\hat{\sigma}_0^2 \mathbf{\Sigma}_l^{-1} \hat{\mathbf{\Sigma}}_l (\hat{\mathbf{\Sigma}}_l + \hat{\sigma}_0^2 \mathbf{I}_n)^{-1}] \\ =&\mathrm{tr}\left(\begin{pmatrix} \frac{1}{\sigma_l^2} & -\frac{\rho_l}{\sigma_l^2} & 0 & 0 & \cdots \\ -\frac{\rho_l}{\sigma_l^2} & \frac{1+\rho_l^2}{\sigma_l^2} & \frac{-\rho_l}{\sigma_l^2} & 0 & \cdots \\ 0 & -\frac{\rho_l}{\sigma_l^2} & \frac{1+\rho_l^2}{\sigma_l^2} & \ddots & \cdots \\ \vdots & \vdots & \ddots & \ddots & -\frac{\rho_l}{\sigma_l^2} \\ 0 & 0 & \cdots & -\frac{\rho_l}{\sigma_l^2} & \frac{1}{\sigma_l^2} \end{pmatrix} \begin{pmatrix} S_l(1) & \tilde{S}_l(1) & \cdot & \cdot & \cdot \\ \tilde{S}_l(1) & S_l(2) & \tilde{S}_l(2) & \cdot & \cdot \\ \cdot & \tilde{S}_l(2) & S_l(3) & \tilde{S}_l(3) & \cdot \\ & & \ddots & \ddots & \\ \cdot & \cdot & \cdot & \cdot & S_l(n) \end{pmatrix} \right) \\ =&\frac{1}{\sigma_l^2} \left(\sum_{t=1}^{n} S_l(t) + \rho_l^2 \sum_{t=2}^{n-1} S_l(t) - 2\rho_l \sum_{t=1}^{n-1} \tilde{S}_l(t) \right),\end{aligned}$$

where $\tilde{S}_l(t) = \mathrm{Cov}[z_l(t), z_l(t+1) \mid \mathbf{Y}, \hat{\mathbf{\Theta}}]$ for $t = 1, \ldots, n-1$ can be obtained by applying KF and RTS smoother to the l-th latent process with computational complexity $\mathcal{O}(n)$.

Therefore, Equation (2.11) can be written as

$$
\begin{aligned}
&\bar{\ell}(\mathbf{\Theta}) \\
=&C - \frac{nk}{2}\log(\sigma_0^2) + \frac{\mathrm{tr}(\mathbf{Y}^\top \mathbf{U}_0 \hat{\mathbf{Z}})}{\sigma_0^2} - \sum_{l=1}^{d}\left\{\frac{1}{2}\log\frac{\sigma_l^{2n}}{1-\rho_l^2} + \frac{\sum_{t=1}^{n}(s_l^2(t)+S_l(t))}{2\sigma_0^2}\right\} - \frac{\mathrm{tr}(\mathbf{Y}^\top\mathbf{Y})}{2\sigma_0^2} \\
&- \sum_{l=1}^{d}\frac{1}{2\sigma_l^2}\Bigg[(1-\rho_l^2)s_l^2(1) + \sum_{t=2}^{n}(s_l(t)-\rho_l s_l(t-1))^2 \\
&\qquad + \sum_{t=1}^{n} S_l(t) + \rho_l^2\sum_{t=2}^{n-1} S_l(t) - 2\rho_l\sum_{t=1}^{n-1}\tilde{S}_l(t)\Bigg].
\end{aligned}
$$

We finish the proof. ■

B.2.4 Proof of Theorem 2

Proof: By leveraging Equation (2.11) and Lemma 7, estimating $\mathbf{U}_0$ is equivalent to maximizing $\mathrm{tr}(\hat{\mathbf{Z}}\mathbf{Y}^\top\mathbf{U}_0)$ with the constraint $\mathbf{U}_0^\top\mathbf{U}_0 = \mathbf{I}_d$. This is a linear optimization on Stiefel manifold (see Proposition 2.5 in [164]). Denote the SVD of $\mathbf{Z}\mathbf{Y}^\top = \tilde{\mathbf{U}}\tilde{\mathbf{D}}\tilde{\mathbf{V}}^\top$. The objective function is then

$$
\mathrm{tr}\left[\hat{\mathbf{Z}}\mathbf{Y}^\top\mathbf{U}_0\right] = \mathrm{tr}\left[\tilde{\mathbf{U}}\tilde{\mathbf{D}}\tilde{\mathbf{V}}^\top\mathbf{U}_0\right] = \mathrm{tr}\left[\tilde{\mathbf{D}}\tilde{\mathbf{V}}^\top\mathbf{U}_0\tilde{\mathbf{U}}\right]. \tag{B.5}
$$

Denote $\tilde{\mathbf{V}}^\top\mathbf{U}_0\tilde{\mathbf{U}} := [\mathbf{m}_1, ..., \mathbf{m}_d] := \mathbf{M}$ a $d \times d$ orthogonal matrix. As $||\mathbf{m}_l|| = 1$ for $l = 1, \ldots, d$, the diagonal terms of $\mathbf{M}$ is not larger than one. As the diagonal term in $\tilde{\mathbf{D}}$ is nonnegative, the maximum values are obtained when diagonal values of $\mathbf{M}$ are 1 or equivalently when

$$
\tilde{\mathbf{V}}^\top\hat{\mathbf{U}}_0^{\mathrm{new}}\tilde{\mathbf{U}} = \mathbf{I}_d,
$$

from which Equation (2.16) is proved.

After obtaining $\mathbf{U}_0^{\text{new}}$, compute $\tilde{\mathbf{Y}} = (\hat{\mathbf{U}}_{\mathbf{0}}^{\text{new}})^\top \mathbf{Y}$ and denote $\tilde{\mathbf{y}}_l = (\tilde{y}_l(1), \ldots, \tilde{y}_l(n))^\top$ for $l = 1, 2, \ldots, d$.

For $((\hat{\sigma}_0^2)^{\text{new}}, \hat{\boldsymbol{\rho}}^{\text{new}}, (\hat{\boldsymbol{\sigma}}^2)^{\text{new}})$, we take partial derivatives of Equation (2.11) with respect to σ_0^2, $\boldsymbol{\sigma}^2$ and $\boldsymbol{\rho}$, respectively, and set the equations to be zero:

$$\frac{\partial}{\partial \rho_l} \bar{\ell}(\boldsymbol{\Theta}) = \beta_0 + \beta_1 \rho_l + \beta_2 \rho_l^2 + \beta_3 \rho_l^3 = 0, \tag{B.6}$$

$$\frac{\partial}{\partial \sigma_0^2} \bar{\ell}(\boldsymbol{\Theta}) = -\frac{nk}{2\sigma_0^2} - \frac{\operatorname{tr}(\mathbf{Y}^\top \hat{\mathbf{U}}_0^{\text{new}} \hat{\mathbf{Z}})}{\sigma_0^4} + \frac{\operatorname{tr}(\mathbf{Y}^\top \mathbf{Y}) + \sum_{l=1}^d \sum_{t=1}^n (s_l^2(t) + S_l(t))}{2\sigma_0^4} = 0, \tag{B.7}$$

$$\begin{aligned}\frac{\partial}{\partial \sigma_l^2} \bar{\ell}(\boldsymbol{\Theta}) = &-\frac{n}{2\sigma_l^2} + \frac{(1-\rho_l^2) s_l(1) + \sum_{t=2}^n (s_l(t) - \rho_l s_l(t-1))^2}{2\sigma_l^4} \\ &+ \frac{\sum_{t=1}^n S_l(t) + \rho_l^2 \sum_{t=2}^{n-1} S_l(t) - 2\rho_l \sum_{t=1}^{n-1} \tilde{S}_l(t)}{2\sigma_l^4} = 0,\end{aligned} \tag{B.8}$$

where $\beta_0 = n\Big(\sum_{t=2}^n s_l(t-1) s_l(t) + \sum_{t=1}^{n-1} \tilde{S}_l(t)\Big)$, $\beta_1 = -\sum_{t=1}^n s_l^2(t) - \sum_{t=1}^n S_l(t) - n\sum_{t=2}^{n-1} s_l^2(t) - n\sum_{t=2}^{n-1} S_l(t)$, $\beta_2 = (2-n)(\sum_{t=2}^n s_l(t-1) s_l(t) + \sum_{t=1}^{n-1} \tilde{S}_l(t))$, $\beta_3 = (n-1)(\sum_{t=2}^{n-1} s_l^2(t) + \sum_{t=2}^{n-1} S_l(t))$, and the superscript "new" is omitted for simplicity.

Solving Equation (B.6) in $(-1, 1)$ provides the updated estimation of ρ_l. We show that there is a unique root in $(-1, 1)$ for this cubic equation. Consider a function of ρ_l such that $h(\rho_l) = \beta_0 + \beta_1 \rho_l + \beta_2 \rho_l^2 + \beta_3 \rho_l^3$. Since $\beta_3 = (n-1)(\sum_{t=2}^{n-1} s_l^2(t) + \sum_{t=2}^{n-1} S_l(t)) > 0$ for $n > 1$, $h(\rho_l) > 0$ as $\rho_l \to \infty$ and $h(\rho_l) < 0$ as $\rho_l \to -\infty$.

First we substitute $\rho_l = -1$:

$$\begin{aligned}h(-1) =& \beta_0 - \beta_1 + \beta_2 - \beta_3 \\ =& \sum_{t=1}^n s_l^2(t) + \sum_{t=1}^n S_l(t) + 2\sum_{t=2}^n s_l(t-1) s_l(t) + 2\sum_{t=1}^{n-1} \tilde{S}_l(t) + \sum_{t=2}^{n-1} s_l^2(t) + \sum_{t=2}^{n-1} S_l(t) \\ =& \sum_{t=1}^{n-1} (s_l(t) + s_l(t+1))^2 + \sum_{t=1}^{n-1} \mathbb{V}\left(z_l(t) + z_l(t+1) \mid \mathbf{Y}, \hat{\boldsymbol{\Theta}}\right) \\ >& 0,\end{aligned}$$

where strict inequality is held as the second term is strictly larger than zero. This is because

$$
\begin{aligned}
&\mathbb{V}\left(z_l(t)+z_l(t+1)\mid \mathbf{Y},\hat{\boldsymbol{\Theta}}\right)\\
=&\mathbb{V}\left(z_l(t)\mid \mathbf{Y},\hat{\boldsymbol{\Theta}}\right)+\mathbb{V}\left(z_l(t+1)\mid \mathbf{Y},\hat{\boldsymbol{\Theta}}\right)+2\mathrm{Cov}\left(z_l(t),z_l(t+1)\mid \mathbf{Y},\hat{\boldsymbol{\Theta}}\right)\\
\geq&\mathbb{V}\left(z_l(t)\mid \mathbf{Y},\hat{\boldsymbol{\Theta}}\right)+\mathbb{V}\left(z_l(t+1)\mid \mathbf{Y},\hat{\boldsymbol{\Theta}}\right)-2\mathbb{V}\left(z_l(t)\mid \mathbf{Y},\hat{\boldsymbol{\Theta}}\right)^{1/2}\mathbb{V}\left(z_l(t+1)\mid \mathbf{Y},\hat{\boldsymbol{\Theta}}\right)^{1/2}\\
=&\left\{\mathbb{V}\left[z_l(t)\mid \mathbf{Y},\hat{\boldsymbol{\Theta}}\right]^{1/2}-\mathbb{V}\left[z_l(t+1)\mid \mathbf{Y},\hat{\boldsymbol{\Theta}}\right]^{1/2}\right\}^2,
\end{aligned}
$$

where the last equation is due to

$$
\mathrm{Cov}[z_l(t),z_l(t+1)\mid \mathbf{Y},\hat{\boldsymbol{\Theta}}]<\left(\mathbb{V}[z_l(t)\mid \mathbf{Y},\hat{\boldsymbol{\Theta}}]\mathbb{V}[z_l(t+1)\mid \mathbf{Y},\hat{\boldsymbol{\Theta}}]\right)^{1/2}.
$$

We only need to show there exists one t, $1\leq t<n-1$,

$$
\left\{\mathbb{V}\left[z_l(t)\mid \mathbf{Y},\hat{\boldsymbol{\Theta}}\right]^{1/2}-\mathbb{V}\left[z_l(t+1)\mid \mathbf{Y},\hat{\boldsymbol{\Theta}}\right]^{1/2}\right\}^2>0.
$$

First, as $\hat{\sigma}_0^2>0$, we have $\mathbb{V}\left[z_l(t)\mid \mathbf{Y},\hat{\boldsymbol{\Theta}}\right]^{1/2}>0$. Second, as $\hat{\sigma}_l>0$, we have

$$
\mathbb{V}\left[z_l(n)\mid \mathbf{Y},\hat{\boldsymbol{\Theta}}\right]^{1/2}<\mathbb{V}\left[z_l(n-1)\mid \mathbf{Y},\hat{\boldsymbol{\Theta}}\right]^{1/2}.
$$

These two facts lead to $\sum_{t=1}^{n-1}\mathbb{V}\left(z_l(t)+z_l(t+1)\mid \mathbf{Y},\hat{\boldsymbol{\Theta}}\right)>0$. Therefore, there exists at least one root in $(-\infty,-1)$.

Next we substitute $\rho_l = 1$:

$$\begin{aligned}
h(1) =& \beta_0 + \beta_1 + \beta_2 + \beta_3 \\
=& -\sum_{t=1}^{n} s_l^2(t) - \sum_{t=1}^{n} S_l(t) + 2\sum_{t=2}^{n} s_l(t-1)s_l(t) + 2\sum_{t=1}^{n-1} \tilde{S}_l(t) - \sum_{t=2}^{n-1} s_l^2(t) - \sum_{t=2}^{n-1} S_l(t) \\
=& -\sum_{t=1}^{n-1} \left(s_l(t) - s_l(t+1)\right)^2 - \sum_{t=1}^{n-1} \mathbb{V}\left(z_l(t) - z_l(t+1) \mid \mathbf{Y}, \hat{\boldsymbol{\Theta}}\right) < 0.
\end{aligned}$$

The strict inequality is due to $\hat{\sigma}_0^2 > 0$ and $\hat{\sigma}_l^2 > 0$. Hence, there exists at least one root in $(1, \infty)$.

Since $h(-1) > 0$ and $h(1) < 0$, by the intermediate value theorem, there exists at least one root in $\hat{\rho}_l \in (-1, 1)$ such that $h(\hat{\rho}_l) = 0$. Furthermore, since $h(\rho_l)$ is cubic, it has at most three roots, with one root in $(-\infty, -1)$, one in $(1, \infty)$, and the root in $(-1, 1)$. Hence, for Equation (B.6), the root in $(-1, 1)$ exists and is unique.

Furthermore, solving Equations (B.7) and (B.8) gives the analytical expression of $(\hat{\sigma}_0^2)^{\text{new}}$ and $(\hat{\boldsymbol{\sigma}}^2)^{\text{new}}$ which are given in Equation (2.17) and Equation (2.19), respectively.

Now we prove that $((\hat{\sigma}_0^2)^{\text{new}}, (\hat{\boldsymbol{\sigma}}^2)^{\text{new}})$ maximize Equation (2.11) by showing that the second derivative is negative at the updated estimators.

$$\begin{aligned}
&\frac{\partial^2}{\partial (\sigma_0^2)^2} \bar{\ell}(\boldsymbol{\Theta}) \bigg|_{\sigma_0^2 = (\hat{\sigma}_0^2)^{\text{new}}} \\
&= \frac{1}{2(\hat{\sigma}_0^6)^{\text{new}}} \left(-\text{tr}(\mathbf{Y}^\top \mathbf{Y}) - \text{tr}(\hat{\mathbf{Z}}^\top \hat{\mathbf{Z}}) + 2\text{tr}(\mathbf{Y}^\top \hat{\mathbf{U}}_0^{\text{new}} \hat{\mathbf{Z}}) \right) - \sum_{l=1}^{d} \sum_{t=1}^{n} S_l(t) \Bigg).
\end{aligned}$$

Since $-\text{tr}(\mathbf{Y}^\top \mathbf{Y}) - \text{tr}(\hat{\mathbf{Z}}^\top \hat{\mathbf{Z}}) + 2\text{tr}(\mathbf{Y}^\top \hat{\mathbf{U}}_0^{\text{new}} \hat{\mathbf{Z}})) = -\text{tr}\left(\|\mathbf{Y} - \hat{\mathbf{U}}_0^{\text{new}} \hat{\mathbf{Z}}\|_F^2 \right) < 0$ for $\mathbf{Y} \neq \hat{\mathbf{U}}_0^{\text{new}} \hat{\mathbf{Z}}$, $S_l(t) \geq 0$ for any l, t, and $(\hat{\sigma}_0^2)^{\text{new}} > 0$, the second derivative of σ_0^2 at $(\hat{\sigma}_0^2)^{\text{new}}$ is negative. Therefore, $(\hat{\sigma}_0^2)^{\text{new}}$ maximizes Equation (2.11).

Besides, since

$$\frac{\partial^2}{\partial(\sigma_l^2)^2}\bar{\ell}(\boldsymbol{\Theta})\bigg|_{\sigma_l^2=(\hat{\sigma}_l^2)^{\text{new}}} = -\frac{n}{2(\hat{\sigma}_l^4)^{\text{new}}} < 0,$$

$(\hat{\sigma}_l^2)^{\text{new}}$ maximizes Equation (2.11). ■

B.2.5 EM algorithm with known $\mathbf{U}_0$ and σ_0^2

When employing our FMOU model in deformation data inversion, the loading matrix $\mathbf{U}_0$ is derived from the matrix of Green's function, and the noise intensity σ_0^2 is usually available as the observation uncertainty. In this case, we only need to estimate $\boldsymbol{\rho}$ and $\boldsymbol{\sigma}^2$. The MMLEs of the parameters can be obtained by maximizing the marginal likelihood of $\mathbf{Y}$:

$$\begin{aligned}(\boldsymbol{\rho}^{\text{MMLE}}, (\boldsymbol{\sigma}^2)^{\text{MMLE}}) &= \arg\max_{\boldsymbol{\rho},\boldsymbol{\sigma}^2} p(\mathbf{Y} \mid \boldsymbol{\rho}, \boldsymbol{\sigma}^2) \\ &= \arg\max_{\boldsymbol{\rho},\boldsymbol{\sigma}^2} \int p(\mathbf{Y} \mid \mathbf{Z}, \mathbf{U}_0, \sigma_0^2, \boldsymbol{\rho}, \boldsymbol{\sigma}^2) p(\mathbf{Z} \mid \boldsymbol{\rho}, \boldsymbol{\sigma}^2) d\mathbf{Z},\end{aligned}$$

The joint log-likelihood of $(\mathbf{Y}, \mathbf{Z} \mid \mathbf{U}_0, \sigma_0^2, \boldsymbol{\rho}, \boldsymbol{\sigma}^2)$ can be written as:

$$\begin{aligned}\ell(\boldsymbol{\rho}, \boldsymbol{\sigma}^2) &= \log\big(p(\mathbf{Y}, \mathbf{Z} \mid \mathbf{U}_0, \sigma_0^2, \boldsymbol{\rho}, \boldsymbol{\sigma}^2)\big) \\ &= \tilde{C} + \sum_{l=1}^{d}\Big(\frac{2\mathbf{z}_l^\top \tilde{\mathbf{y}}_l - \mathbf{z}_l^\top \mathbf{z}_l}{2\sigma_0^2} - \frac{\log|\boldsymbol{\Sigma}_l|}{2} - \frac{\mathbf{z}_l^\top \boldsymbol{\Sigma}_l^{-1} \mathbf{z}_l}{2}\Big),\end{aligned} \tag{B.9}$$

where $\tilde{C} = -\frac{(nk+nd)}{2}\log(2\pi) - \frac{nk}{2}\log(\sigma_0^2) - \frac{\text{tr}(\mathbf{Y}^\top\mathbf{Y})}{2\sigma_0^2}$ is constant since σ_0^2 is known, and $\tilde{\mathbf{Y}} = \mathbf{U}_0^\top \mathbf{Y} = [\tilde{\mathbf{y}}_1, \ldots, \tilde{\mathbf{y}}_d]^\top$.

In the E-step, we take the expectation of Equation (B.9) with respect to the conditional distribution of $\mathbf{Z}$ based on the current estimation of $(\hat{\boldsymbol{\rho}}, \hat{\boldsymbol{\sigma}}^2)$ and observations

$\mathbf{Y}$:

$$\begin{aligned}
\bar{\ell}(\boldsymbol{\rho}, \boldsymbol{\sigma}^2) :&= \mathbb{E}_{\mathbf{Z}|\mathbf{Y},\mathbf{U}_0,\sigma_0^2,\hat{\boldsymbol{\rho}},\hat{\boldsymbol{\sigma}}^2}[\ell(\boldsymbol{\rho}, \boldsymbol{\sigma}^2)] \\
&= \tilde{C} - \frac{1}{2}\sum_{l=1}^{d} \log|\boldsymbol{\Sigma}_l| + \frac{\sum_{l=1}^{d} \mathbb{E}_{\mathbf{Z}|\mathbf{Y},\mathbf{U}_0,\sigma_0^2,\hat{\boldsymbol{\rho}},\hat{\boldsymbol{\sigma}}^2}[\mathbf{z}_l^\top \tilde{\mathbf{y}}_l]}{\sigma_0^2} - \sum_{l=1}^{d} \frac{\mathbb{E}_{\mathbf{Z}|\mathbf{Y},\mathbf{U}_0,\sigma_0^2,\hat{\boldsymbol{\rho}},\hat{\boldsymbol{\sigma}}^2}[\mathbf{z}_l^\top \mathbf{z}_l]}{2\sigma_0^2} \\
&\quad - \sum_{l=1}^{d} \frac{\mathbb{E}_{\mathbf{Z}|\mathbf{Y},\mathbf{U}_0,\sigma_0^2,\hat{\boldsymbol{\rho}},\hat{\boldsymbol{\sigma}}^2}[\mathbf{z}_l^\top \boldsymbol{\Sigma}_l^{-1} \mathbf{z}_l]}{2} \\
&= \tilde{C} - \frac{1}{2}\sum_{l=1}^{d} \log|\boldsymbol{\Sigma}_l| + \frac{\sum_{l=1}^{d} \hat{\mathbf{z}}_l^\top \tilde{\mathbf{y}}_l}{\sigma_0^2} - \sum_{l=1}^{d} \frac{\hat{\mathbf{z}}_l^\top \hat{\mathbf{z}}_l + \mathrm{tr}[\sigma_0^2 \hat{\boldsymbol{\Sigma}}_l (\hat{\boldsymbol{\Sigma}}_l + \sigma_0^2 \mathbf{I}_n)^{-1}]}{2\sigma_0^2} \\
&\quad - \sum_{l=1}^{d} \frac{\hat{\mathbf{z}}_l^\top \boldsymbol{\Sigma}_l^{-1} \hat{\mathbf{z}}_l + \mathrm{tr}[\sigma_0^2 \boldsymbol{\Sigma}_l^{-1} \hat{\boldsymbol{\Sigma}}_l (\hat{\boldsymbol{\Sigma}}_l + \sigma_0^2 \mathbf{I}_n)^{-1}]}{2}. \qquad \text{(B.10)}
\end{aligned}$$

To estimate $\boldsymbol{\rho}$ and $\boldsymbol{\sigma}^2$, one only needs to force $\hat{\mathbf{U}}_0^{\text{new}} \equiv \mathbf{U}_0$ and $(\hat{\sigma}_0^2)^{\text{new}} \equiv \sigma_0^2$ throughout the iterations. The analytical forms of updating ρ_l and σ_l^2 are provided in Equations (2.18) and (2.19).

B.3 Summary of the network inversion filter

The network inversion filter (NIF) proposed by [2] is a routinely used approach for inverse estimation of geodetic data. The slips of interest are assumed to be a linear combination of d orthogonal basis functions in Equation (2.2) with coefficient vectors $\tilde{\mathbf{z}}_l = [\tilde{z}_l(1), \ldots, \tilde{z}_l(n)]^\top$ constructed by $\tilde{z}_l(t) = v_l t + W_l(t)$ for $l = 1, \ldots, d$ and $t = 1, \ldots, n$, where v_l is a time-independent velocity and $W_l(t) = \int_0^t B(t')dt'$ is an integrated Brownian motion with a scale parameter α. The orthogonal basis functions are chosen as $\mathbf{U}_0\boldsymbol{\Lambda}_0$, where the $k \times d$ orthogonal matrix $\mathbf{U}_0$ and the $d \times d$ diagonal matrix $\boldsymbol{\Lambda}_0$ contain the first d eigenvectors and eigenvalues of $\mathbf{G}\mathbf{G}^\top$, respectively. Thus $\boldsymbol{\Lambda}_0 = \mathbf{D}_0^2$. For equally spaced

time points, the model below is assumed in NIF for GPS measurements

$$\begin{aligned}
\mathbf{y}(t) &= \mathbf{U}_0\mathbf{\Lambda}_0\mathbf{X}(t) + \boldsymbol{\epsilon}(t), \quad \boldsymbol{\epsilon}(t) \sim \mathcal{MN}(\mathbf{0}, \sigma_0^2\mathbf{I}_k), \\
W_l(t) &= W_l(t-1) + \dot{W}_l(t-1) + \omega_l(t), \quad \omega_l(t) \sim \mathcal{N}\left(0, \frac{\alpha^2}{3}\right), \\
\dot{W}_l(t) &= \dot{W}_l(t-1) + \dot{\omega}_l(t), \quad \dot{\omega}_l(t) \sim \mathcal{N}(0, \alpha^2),
\end{aligned}$$

where the state vector $\mathbf{X}(t) = [v_1 t + W_1(t), \ldots, v_d t + W_d(t)]^\top$, $\dot{W}_l(t)$ represents the derivative of $W_l(t)$, and $\mathrm{Cov}(\omega_l(t), \dot{\omega}_l(t)) = \frac{\alpha^2}{2}$, for $l = 1, \ldots, d$. When using the KF for computing the likelihood function, one needs to directly invert some $k \times k$ matrices in each step, which can be computationally expensive for large k, as the overall computational complexity of NIF is $\mathcal{O}(\min(k^2k', k(k')^2)) + \mathcal{O}(M_0nk^3)$, where M_0 is the number of optimization steps taken by NIF. In comparison, the computational operation of our EM algorithm for the FMOU model using a given Green's function only requires $\mathcal{O}(\min(k^2k', k(k')^2)) + \mathcal{O}(Mknd)$ operations. As the second term is dominating the computation, the FMOU is at the order of k^2/d times faster than the NIF in each iteration of the optimization.

In the modified NIF proposed in [85], the GPS data in Cascadia at any time t follows

$$\mathbf{y}(t) = \tilde{\mathbf{G}}\tilde{\mathbf{z}}_s(t) + \boldsymbol{\epsilon}(t) = \mathbf{G}\mathbf{z}_s(t) + \begin{pmatrix} \mathbf{I}_2 \\ \vdots \\ \mathbf{I}_2 \end{pmatrix} \mathbf{f}_2 + \boldsymbol{\epsilon}(t). \tag{B.11}$$

Here $\mathbf{f}_2 = (f_E, f_N)^\top$ denotes the reference frame motions in the East-West and North-South directions, assumed to follow $\mathcal{MN}(\mathbf{0}, \sigma_f^2\mathbf{I}_2)$, where σ_f^2 measures the amplitude of external global motions. One can rewrite the modified NIF model as a dynamic linear

model:

$$\mathbf{y}(t) = \mathbf{H}(t)\mathbf{X}(t) + \boldsymbol{\epsilon}(t) \quad \boldsymbol{\epsilon}(t) \sim \mathcal{MN}(\mathbf{0}, \sigma_0^2 \mathbf{I}_k), \tag{B.12}$$

$$\mathbf{X}(t) = \mathbf{T}(t)\mathbf{X}(t-1) + \boldsymbol{\delta}(t), \quad \boldsymbol{\delta}(t) \sim \mathcal{MN}(\mathbf{0}, \boldsymbol{\Omega}(t)). \tag{B.13}$$

The terms in Equations (B.12) and (B.13) are specified as follows.

1. The latent states $\mathbf{X}(t)$ includes slips and slip rates at time t:

$$\mathbf{X}(t) = [z_s(\boldsymbol{\xi}_1, t), \dot{z}_s(\boldsymbol{\xi}_1, t), ..., z_s(\boldsymbol{\xi}_{k'}, t), \dot{z}_s(\boldsymbol{\xi}_{k'}, t), \mathbf{f}_2]^\top \in \mathbb{R}^{2k'+2},$$

   where $\dot{z}_s(\boldsymbol{\xi}_h, t)$ represents the slip rate at $\boldsymbol{\xi}_h$ and time t, for $h = 1, \ldots, k'$ and $t = 1, \ldots, n$.

2. $\mathbf{H}(t) = \left(\mathbf{G}^*, \mathbf{I}_k\right) \in \mathbb{R}^{k \times (k+2k')}$, where $\mathbf{G}^* = \mathbf{G} \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & \cdots & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & \cdots & 1 & 0 \\ \vdots & & \vdots & & \vdots & & \vdots & \\ 1 & 0 & 1 & 0 & 1 & \cdots & 1 & 0 \end{pmatrix}_{k' \times 2k'}$ is a matrix of Green's function calculated for the Cascadia region.

3. $\mathbf{T}(t) = \begin{pmatrix} \mathbf{T}^0(t) & & & \\ & \ddots & & \\ & & \mathbf{T}^0(t) & \\ & & & \mathbf{0} \end{pmatrix} \in \mathbb{R}^{2k'+1}$, where $\mathbf{T}^0(t) = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$.

4. $\boldsymbol{\Omega}(t) = \begin{pmatrix} \boldsymbol{\Omega}^0(t) & & & \\ & \ddots & & \\ & & \boldsymbol{\Omega}^0(t) & \\ & & & \sigma_f^2 \mathbf{I}_2 \end{pmatrix}$, where $\boldsymbol{\Omega}^0(t) = \begin{pmatrix} \frac{\alpha^2}{3} & \frac{\alpha^2}{2} \\ \frac{\alpha^2}{2} & \alpha^2 \end{pmatrix}$.

The parameters α and σ_f^2 can be estimated by the MLE, where KF is used to compute the likelihood function and RTS smoother is used for computing the posterior distribution of the slip. Due to the high computational cost of inverting the covariance matrices at each time point, only a small number of parameter values at a grid is computed in modified NIF to approximate MLE.

As the dimension of state spaces depends entirely on k and k', the MATLAB code proposed in [85] requires more time for the computation compared to both the FMOU model and NIF model due to matrix inversions of a $k \times k$ matrix during forward filtering, and a $(2(k'+1)) \times (2(k'+1))$ matrix during backward smoothing. The Woodbury matrix identity can be used to simplify the computation cost of the modified NIF so that it is closer to the NIF model.

B.4 Supplement to simulation results

B.4.1 Uncertainty quantification of the predictive signal and slip

We consider two additional criteria to evaluate the model, including the average length of the 95% posterior credible intervals of the observation mean ($\mathrm{L}^m(95\%)$) and the slip ($\mathrm{L}^s(95\%)$), and the proportion of the observation mean ($\mathrm{P}^m(95\%)$) and the slip

($\mathrm{P}^s(95\%)$)covered by the 95% posterior credible intervals:

$$\mathrm{L}^m(95\%) = \frac{1}{kn}\sum_{t=1}^{n}\sum_{i=1}^{k}\mathrm{length}\{\mathrm{CI}_{i,t}^m(95\%)\}, \tag{B.14}$$

$$\mathrm{L}^s(95\%) = \frac{1}{k'n}\sum_{t=1}^{n}\sum_{h=1}^{k'}\mathrm{length}\{\mathrm{CI}_{h,t}^s(95\%)\}, \tag{B.15}$$

$$\mathrm{P}^m(95\%) = \frac{1}{kn}\sum_{t=1}^{n}\sum_{i=1}^{k}1_{m_i(t)} \in \mathrm{CI}_{i,t}^m(95\%), \tag{B.16}$$

$$\mathrm{P}^s(95\%) = \frac{1}{k'n}\sum_{t=1}^{n}\sum_{h=1}^{k'}1_{(z_{s,h}(t))\in \mathrm{CI}_{h,t}^s(95\%)}, \tag{B.17}$$

where the superscript 'm' and 's' denote the mean and slip, respectively.

B.4.2 Additional results in Experiment 2

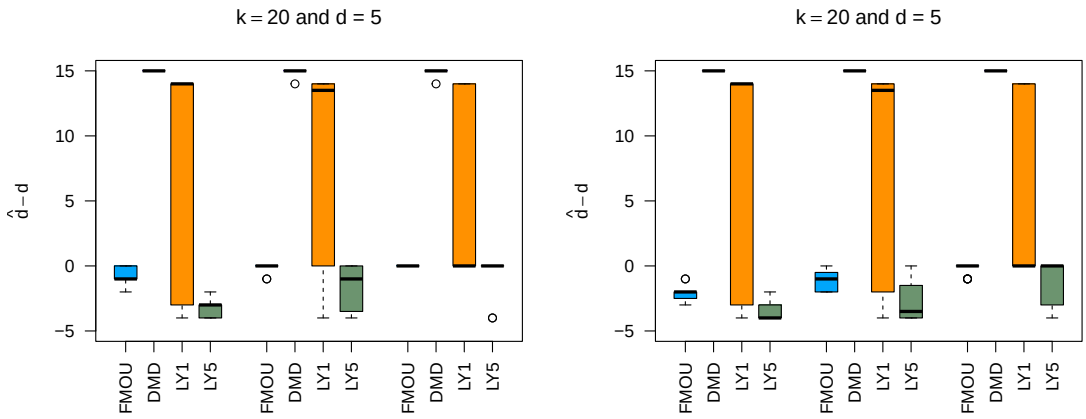

Figure B.1: Differences between the selected number of factors and the truth by different methods in Experiment 2. In each subfigure, the first 4, middle 4, and the last 4 boxes are based on $n = 100$, $n = 200$, and $n = 300$, respectively. The noise variance is $\sigma_0^2 = 1$ and $\sigma_0^2 = 2$ for the left and right subfigures, respectively.

In Figure B.1, we plot the difference between the estimated number of latent factors $\hat{d}$ and the true latent factor using four different methods. The estimation accuracy based on IC in Equation (2.23), LY1 and LY5 in [21] improves with larger sample sizes. Among

Estimated d	$\sigma_0^2 = 1$			$\sigma_0^2 = 2$		
	$n = 100$	$n = 200$	$n = 400$	$n = 100$	$n = 200$	$n = 400$
FMOU	**0.41**	**0.35**	**0.33**	**0.62**	**0.53**	**0.44**
DMD	0.71	0.67	0.65	0.91	0.83	0.80
LY1	1.3	1.1	0.71	2.0	1.5	1.0
LY5	1.7	0.95	0.65	2.0	1.4	1.2

Table B.1: Average of RMSE_m over N simulated configurations in Experiment 2 with d estimated by 4 different methods.

True d	$\sigma_0^2 = 1$			$\sigma_0^2 = 2$		
	$n = 100$	$n = 200$	$n = 400$	$n = 100$	$n = 200$	$n = 400$
$\text{L}^m(95\%)$	1.2	1.2	1.2	1.5	1.5	1.5
$\text{P}^m(95\%)$	88%	91%	93%	87%	90%	92%
SD_{signal}	5.1	2.2	2.9	5.1	2.2	2.9
Estimated d	$\sigma_0^2 = 1$			$\sigma_0^2 = 2$		
	$n = 100$	$n = 200$	$n = 400$	$n = 100$	$n = 200$	$n = 400$
$\text{L}^m(95\%)$	1.1	1.2	1.2	1.1	1.3	1.4
$\text{P}^m(95\%)$	82%	90%	93%	64%	77%	90%
SD_{signal}	5.1	2.2	2.9	5.1	2.2	2.9

Table B.2: The average lengths of 95% posterior credible intervals, the percentages of the signal covered by the intervals from FMOU, and the standard deviation of signals (SD_{signal}) for Experiment 3.

these methods, IC consistently outperforms the other methods in accuracy across all configurations. Table B.1 displays the RMSE_m by the four methods using the estimated number of latent factors. The FMOU method achieves the most accurate estimation of the signal from noise observations.

Table B.2 summarizes the average length and proportion of the signal covered in 95% posterior credible intervals of FMOU. The lengths of the intervals are relatively short compared to the standard deviation of the signal, indicating precise estimation. Note that this simulation represents a challenging scenario, as the model contains a large number of parameters that cannot be integrated out, and the variance of noise is

relatively large. Consequently, the true values covered by the 95% credible interval are shorter than 95% for a small sample size. When the sample size grows to moderately large, the percentage of the true outputs covered by the 95% posterior credible interval increases to be close to 95%.

B.4.3 Additional results in Experiment 3

Table B.3 shows the average length of the 95% posterior credible intervals and the percentage of the signal covered in the 95% posterior credible intervals. The length of the 95% posterior credible intervals is short compared to the standard deviation of the signal, indicating the method is precise. In both the linear diffusion and Branin function examples, the percentage of the signal covered in the 95% posterior credible interval is smaller than 95%, particularly when the noise variance is large. This discrepancy arises because the signal is governed by a smooth process, while the FMOU process is not differentiable.

	Linear diffusion			Branin function		
	$\sigma_0^2 = 0.0001$	$\sigma_0^2 = 0.0025$	$\sigma_0^2 = 0.09$	$\sigma_0^2 = 1$	$\sigma_0^2 = 25$	$\sigma_0^2 = 400$
$\mathrm{L}^m(95\%)$	0.0046	0.016	0.038	0.35	1.5	4.2
$\mathrm{P}^m(95\%)$	75%	72%	42%	80%	77%	67%
SD_{signal}	0.30	0.30	0.30	52	52	52

Table B.3: The average lengths of 95% posterior credible intervals, the percentages of the samples covered by the interval from FMOU, and the standard deviation of signals (SD_{signal}) for Experiment 3.

B.4.4 Additional results in Experiment 4

In Experiment 4, we assess the accuracy of $\hat{d}$ selected by three approaches. First, assuming σ_0^2 is unknown, we apply the IC in Equation (2.23) based on the left singular vectors of observations and the known factor loading matrix $\mathbf{U}$. We also employ the

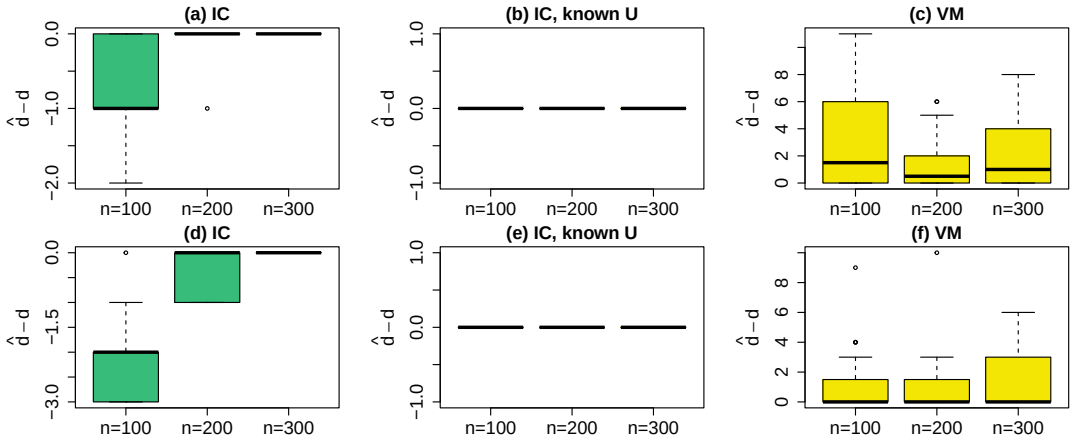

Figure B.2: Difference between $\hat{d}$ and d under different combinations of parameters in Experiment 4. Three approaches are compared: 1) the 'IC' method denotes the IC criterion in Equation (2.23) with factor loading matrix constructed by left singular vectors of observations; 2) the 'IC, known U' method denotes the IC criterion using the known $\mathbf{U}$ as factor loading matrix; 3) the 'VM' method denotes the variance matching method. The results for scenarios with $(k = 25, k' = 150, d = 6)$ are shown in panels (a)-(c), and the results for scenarios with $(k = 32, k' = 100, d = 8)$ are shown in panels (d)-(f).

true d	$k = 25, k' = 150, d = 6$			$k = 32, k' = 100, d = 8$		
	$n = 100$	$n = 200$	$n = 300$	$n = 100$	$n = 200$	$n = 300$
$\mathrm{L}^m(95\%)$	1.5	1.5	1.5	1.6	1.6	1.6
$\mathrm{P}^m(95\%)$	95%	95%	95%	95%	95%	95%
SD_{signal}	2.4	2.5	2.4	3.6	2.7	2.6
Estimated d	$k = 25, k' = 150, d = 6$			$k = 32, k' = 100, d = 8$		
	$n = 100$	$n = 200$	$n = 300$	$n = 100$	$n = 200$	$n = 300$
$\mathrm{L}^m(95\%)$	1.6	1.5	1.5	1.6	1.6	1.6
$\mathrm{P}^m(95\%)$	69%	92%	95%	45%	84%	95%
SD_{signal}	2.4	2.5	2.4	3.6	2.7	2.6

Table B.4: The average lengths of the 95% posterior credible intervals, the percentages of the signal covered by the intervals from FMOU, and the standard deviation of the signals (SD_{signal}) for Experiment 4. The IC in Equation (2.23) is used for estimating the number of factors d.

variance matching (VM) method to estimate the number of factors. Since d can take values from 1 to k, estimating the noise levels for all possible values of d can be slow when k is large. As $\hat{\sigma}_0^2(d)$ is approximately nonincreasing over d, we use the binary search

true d	$k = 25, k' = 150, d = 6$			$k = 32, k' = 100, d = 8$		
	$n = 100$	$n = 200$	$n = 300$	$n = 100$	$n = 200$	$n = 300$
$L^s(95\%)$	0.67	0.66	0.66	0.95	0.95	0.95
$P^s(95\%)$	95%	95%	95%	95%	95%	95%
SD_{slip}	1.1	1.1	1.0	2.4	1.7	1.6
Estimated d	$k = 25, k' = 150, d = 6$			$k = 32, k' = 100, d = 8$		
	$n = 100$	$n = 200$	$n = 300$	$n = 100$	$n = 200$	$n = 300$
$L^s(95\%)$	0.68	0.67	0.66	0.95	0.94	0.95
$P^s(95\%)$	68%	92%	95%	45%	84%	95%
SD_{slip}	1.1	1.1	1.0	2.4	1.7	1.6

Table B.5: The average lengths of the 95% posterior credible intervals of the slips by FMOU, the true slip covered by the intervals, and the standard deviation of the slip (SD_{slip}) in Experiment 4. The IC in Equation (2.23) is used for estimating the number of factors d.

algorithm [165] to find a d that yields a $\hat{\sigma}_0^2$ closest to the known noise level. The results are summarized in Figure B.2. The IC in Equation (2.23) with the known factor loading matrix accurately estimates d for both small and moderately large sample sizes. The VM method tends to slightly overestimate the number of factors in some simulation scenarios. Note that the goal of the VM is to select a model to approximate the signal. We found that the selected models with a slightly larger number of factors and more parameters tend to have better performance when the model is misspecified.

Figure B.3 compares $RMSE_m$ with different methods of selecting d. The predictive accuracy of the signal aligns with the trends observed in Figure B.2, where all methods improve as the sample size increases. Among these methods, the VM and the IC with a known $\mathbf{U}$ are more precise than the other approaches.

As shown in Table B.4 and B.5, the FMOU method provides relatively short 95% posterior credible intervals for both signals and slips, indicating precise estimation. The proportion of the samples covered in 95% of the posterior credible interval is close to 95% when the sample size is moderately large.

	$L^m(95\%)$	$P^m(95\%)$	SD_{signal}	$L^s(95\%)$	$P^s(95\%)$	SD_{slip}
$\sigma_0^2 = 0.01^2$	0.014	92%	0.19	0.25	50%	0.85
$\sigma_0^2 = 0.05^2$	0.044	90%	0.19	0.51	61%	0.85
$\sigma_0^2 = 0.2^2$	0.12	89%	0.19	0.87	73%	0.85

Table B.6: The average length of the 95% posterior credible intervals of signals by FMOU, the percentages of the samples covered by the intervals, and standard deviation of the signals and slips in Experiment 4.

B.4.5 Additional results in Experiment 5

Table B.6 provides the average length of the 95% posterior credible intervals of signals and slips by FMOU, and the percentages of the truth covered by the intervals for Experiment 5. The percentage of the signal covered by the interval is close to 95%, yet the percentage of the slip covered by the interval is smaller than 95%. This is because we only have 88 time points which are small, and the signal is misspecified. Obtaining more observations and extending the FMOU model with differentiable latent processes can increase the percentage of true slips covered by the model.

B.5 Data preprocessing and model fitting for the Cascadia data

The original observations, denoted as $\mathbf{y}^*(\mathbf{x}_i, t) \in \mathbb{R}^2$, represent the measurements in East-West and North-South directions at the ith GPS station and time point t, for $i = 1, \ldots, 100$, $t = 1, \ldots, 88$ indicating the total of $k = 100$ GPS stations and $n = 88$ days of observations. As identified by [101], the observed data include internal signals related to plate convergence, which are at a scale comparable to the slip-related signals. Following [101, 85], we first remove the seasonal components and get the processed data

$\mathbf{y}(\mathbf{x}_i, t)$ by

$$\mathbf{y}(\mathbf{x}_i, t) = \mathbf{y}^*(\mathbf{x}_i, t) - \mathbf{c}_1 t - \mathbf{c}_2 sin(2\pi t) - \mathbf{c}_3 cos(2\pi t) - \mathbf{c}_4 sin(4\pi t) - \mathbf{c}_5 cos(4\pi t), \quad \text{(B.18)}$$

where $\mathbf{c}_1 = (c_{1,E}, c_{1,N})^\top$ represents the secular velocity and $\mathbf{c}_2 = (c_{2,E}, c_{2,N})^\top$, $\mathbf{c}_3 = (c_{3,E}, c_{3,N})^\top$, $\mathbf{c}_4 = (c_{4,E}, c_{4,N})^\top$, $\mathbf{c}_5 = (c_{5,E}, c_{5,N})^\top$ represent the rates of four seasonal components in East-West and North-South directions. For $t = 1, \ldots, 88$, after obtaining $\mathbf{y}(t) = [\mathbf{y}^\top(\mathbf{x}_1, t), \ldots, \mathbf{y}^\top(\mathbf{x}_{100}, t)]^\top$ by Equation (B.18), we impute 15% of missing values using a Gaussian process model with an exponential covariance function separately for each station. The parameters are estimated by the MLE. We use the `FastGaSP R` package [86] to compute these predictive distributions and obtain posterior samples of the missing observations for imputation.

In the Cascadia dataset, the measurement noise variance are provided. As the noise variance at each location and time point is approximately the same, we use the average to estimate the variance of the noise in observations. We implement the variance matching method with a binary search algorithm to estimate the number of latent processes, $\hat{d} = 30$, in the FMOU model.

To compute the proportion of tremors detected by the large slip rate and the total areas containing a large slip rate shown in Figure 8, we partition the Cascadia subduction zone into 2,280 grids of equal area, with the temporal domain segmented into 16 intervals, each spanning 5 days. Given a threshold, we identify grids containing stations with larger average slip rates in 5 consecutive days. These grids are flagged as potential anomalies, and we count the number of grids having high slip rates. The proportion of identified tremors is defined as the ratio of grids containing both tremors and potential anomalies to the total number of grids with tremors.

Figure B.4 shows the estimated slip rates from FMOU, NIF and the modified NIF

over other periods, in addition to Figure 10. The results are consistent with the findings discussed in the manuscript.

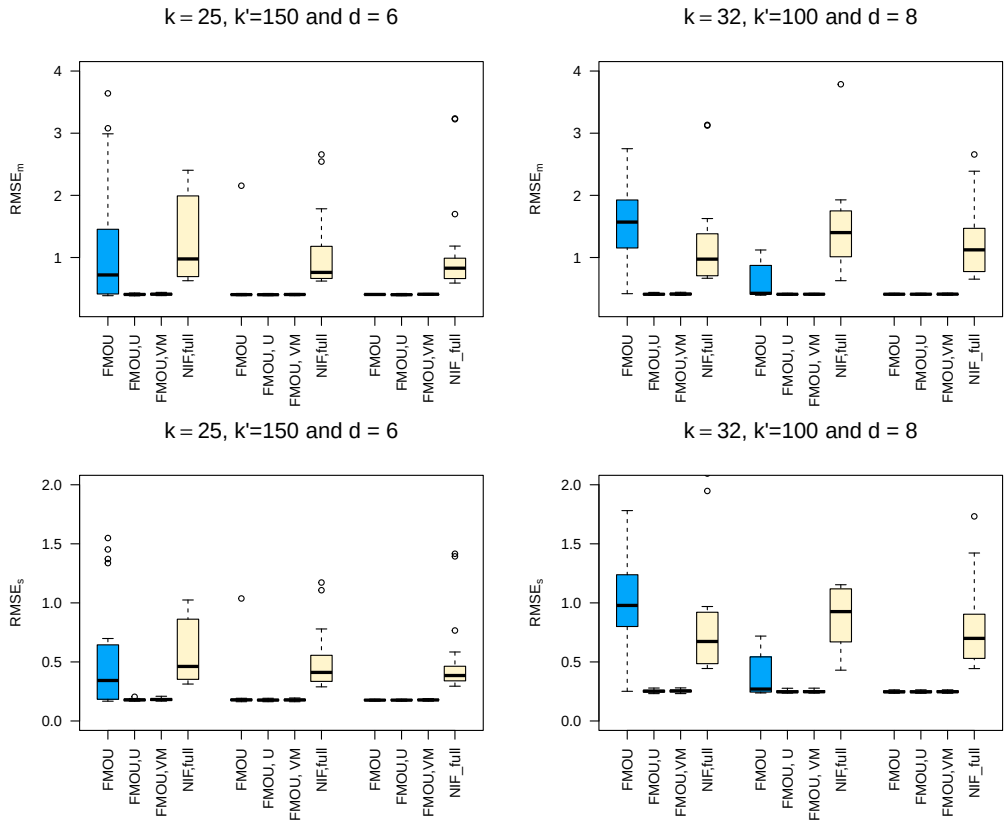

Figure B.3: Box plots of RMSE_m and RMSE_s from 4 methods in Experiment 4. 1) 'FMOU': the estimated latent factor, $\hat{d}$ is selected by IC in Equation (2.23) with factor loading matrix given by left singular vectors of observations. 2) 'FMOU, U': $\hat{d}$ selected by IC with a known factor loading $\mathbf{U}$; 3) 'FMOU, VM': d is estimated by variance matching; 4) 'NIF,full': NIF with full rank ($d = k$). In each subfigure, the first 4, middle 4, and the last 4 boxes are based on $n = 100$, $n = 200$ and $n = 300$, respectively.

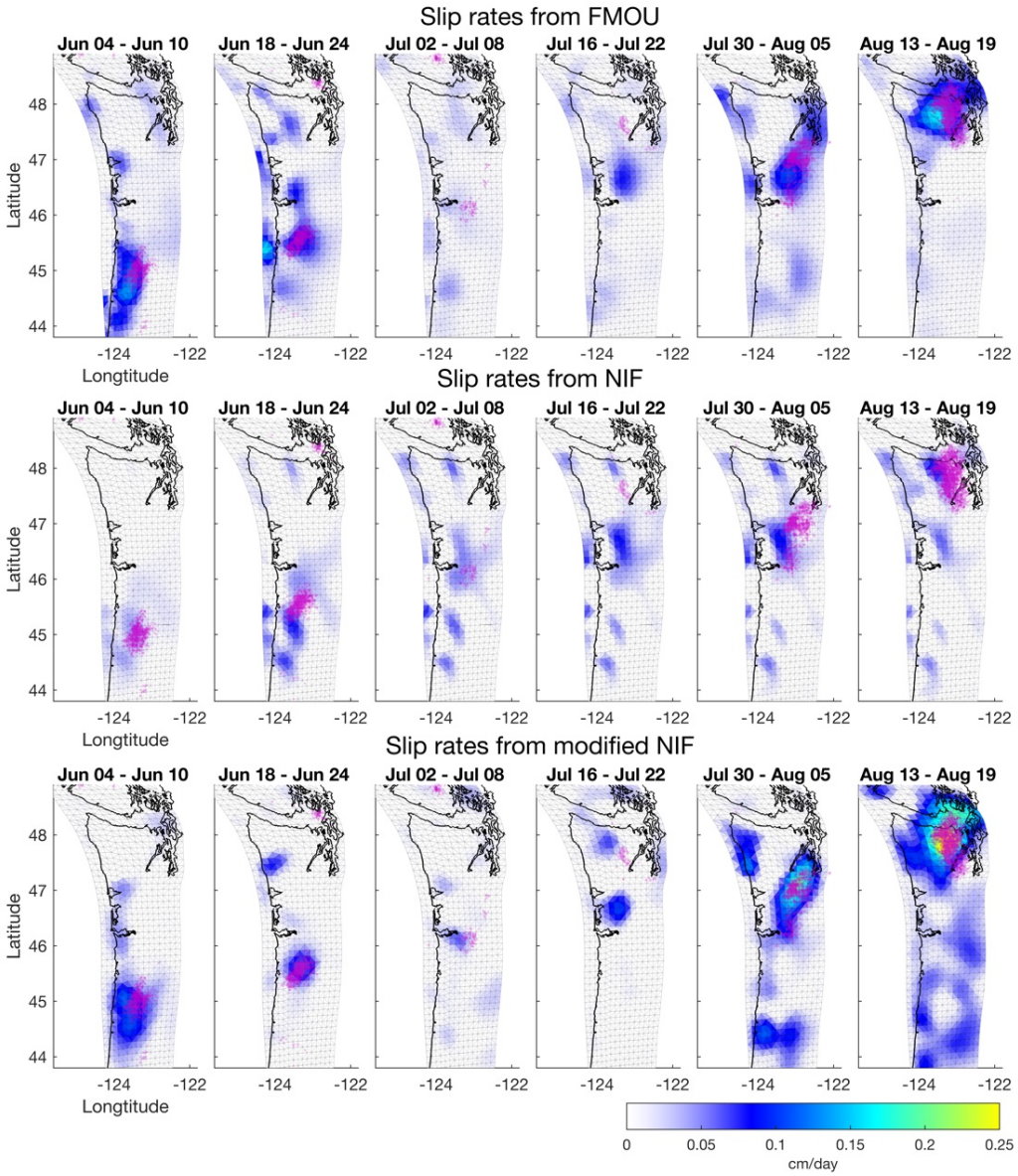

Figure B.4: The seven-day averages of slip rates over the observed dates. Slip rates estimated by the FMOU, NIF and modified NIF are plotted in upper row, middle row and bottom row, respectively, for the Cascadia displacement data in 2011. The tremor epicenters are plotted as magenta dots.

Bibliography

[1] D. Stoffer, *astsa: Applied Statistical Time Series Analysis*, 2025. R package version 2.2.

[2] P. Segall and M. Matthews, "Time dependent inversion of geodetic data," *Journal of Geophysical Research: Solid Earth*, vol. 102, no. B10, pp. 22391–22409, 1997.

[3] L. V. Keldysh, "Diagram technique for nonequilibrium processes," in *Selected Papers of Leonid V Keldysh*, pp. 47–55, World Scientific, 2024.

[4] L. P. Kadanoff, *Quantum statistical mechanics*. CRC Press, 2018.

[5] G. Stefanucci and R. Van Leeuwen, *Nonequilibrium many-body theory of quantum systems: a modern introduction*. Cambridge University Press, 2013.

[6] E. N. Lorenz, "Predictability: A problem partly solved," in *Proc. Seminar on predictability*, vol. 1, 1996.

[7] K. Cho, B. Van Merriënboer, Ç. Gulçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder–decoder for statistical machine translation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1724–1734, 2014.

[8] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[10] N. A. Cressie, *Statistics for spatial data*. Wiley, New York, 1993.

[11] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, "Design and analysis of computer experiments," *Statistical science*, vol. 4, no. 4, pp. 409–423, 1989.

[12] M. J. Bayarri, J. O. Berger, R. Paulo, J. Sacks, J. A. Cafeo, J. Cavendish, C.-H. Lin, and J. Tu, "A framework for validation of computer models," *Technometrics*, vol. 49, no. 2, pp. 138–154, 2007.

[13] H. Li, M. Zhou, J. Sebastian, J. Wu, and M. Gu, "Efficient force field and energy emulation through partition of permutationally equivalent atoms," *The Journal of Chemical Physics*, vol. 156, no. 18, p. 184304, 2022.

[14] M. C. Kennedy and A. O'Hagan, "Bayesian calibration of computer models," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 63, no. 3, pp. 425–464, 2001.

[15] H. Zhao and J. Kowalski, "Bayesian active learning for parameter calibration of landslide run-out models," *Landslides*, vol. 19, no. 8, pp. 2033–2045, 2022.

[16] W. Chang, B. A. Konomi, G. Karagiannis, Y. Guan, and M. Haran, "Ice model calibration using semicontinuous spatial data," *The Annals of Applied Statistics*, vol. 16, no. 3, pp. 1937–1961, 2022.

[17] M. Gu, X. Liu, X. Fang, and S. Tang, "Scalable marginalization of correlated latent variables with applications to learning particle interaction kernels," *Accepted in the New England Journal of Statistics in Data Science, arXiv preprint arXiv:2203.08389*, 2022.

[18] X. Fang and M. Gu, "The inverse kalman filter," *Biometrika*, vol. 112, no. 4, p. asaf054, 2025.

[19] M. Gu and W. Shen, "Generalized probabilistic principal component analysis of correlated data," *Journal of Machine Learning Research*, vol. 21, no. 13, 2020.

[20] M. Gu, Y. He, X. Liu, and Y. Luo, "Ab initio uncertainty quantification in scattering analysis of microscopy," *Accepted in Physical Review E, arXiv preprint arXiv:2309.02468*, 2024.

[21] C. Lam and Q. Yao, "Factor modeling for high-dimensional time series: inference for the number of factors," *The Annals of Statistics*, vol. 40, no. 2, pp. 694–726, 2012.

[22] S. Conti and A. O'Hagan, "Bayesian emulation of complex multi-output and dynamic computer models," *Journal of statistical planning and inference*, vol. 140, no. 3, pp. 640–651, 2010.

[23] L. Lee, K. Carslaw, K. Pringle, G. Mann, and D. Spracklen, "Emulation of a complex global aerosol model to quantify sensitivity to uncertain parameters," *Atmospheric Chemistry and Physics*, vol. 11, no. 23, pp. 12253–12273, 2011.

[24] L. Lee, K. Carslaw, K. Pringle, and G. Mann, "Mapping the uncertainty in global ccn using emulation," *Atmospheric Chemistry and Physics*, vol. 12, no. 20, pp. 9739–9751, 2012.

[25] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[26] H. E. Rauch, F. Tung, and C. T. Striebel, "Maximum likelihood estimates of linear dynamic systems," *AIAA journal*, vol. 3, no. 8, pp. 1445–1450, 1965.

[27] M. Gu and J. O. Berger, "Parallel partial Gaussian process emulation for computer models with massive output," *The Annals of Applied Statistics*, vol. 10, no. 3, pp. 1317–1347, 2016.

[28] P. J. Schmid, "Dynamic mode decomposition of numerical and experimental data," *Journal of fluid mechanics*, vol. 656, pp. 5–28, 2010.

[29] J. H. Tu, C. W. Rowley, D. M. Luchtenburg, S. L. Brunton, and J. N. Kutz, "On dynamic mode decomposition: Theory and applications," *Journal of Computational Dynamics*, vol. 1, no. 2, pp. 391–421, 2014.

[30] M. D. Hartman, W. J. Parton, J. D. Derner, D. K. Schulte, W. K. Smith, D. E. Peck, K. A. Day, S. J. Del Grosso, S. Lutz, B. A. Fuchs, *et al.*, "Seasonal grassland productivity forecast for the US Great Plains using Grass-Cast," *Ecosphere*, vol. 11, no. 11, p. e03280, 2020.

[31] E. McPhillips, H. Lee, X. Xie, K. Baylis, C. Funk, and M. Gu, "Long-term probabilistic forecast of vegetation conditions using climate attributes in the four corners region," *Remote Sensing*, vol. 18, no. 6, p. 850, 2026.

[32] O. Roustant, D. Ginsbourger, and Y. Deville, "Dicekriging, diceoptim: Two r packages for the analysis of computer experiments by kriging-based metamodeling and optimization," *Journal of Statistical Software*, vol. 51, no. 1, pp. 1–55, 2012.

[33] M. Gu, J. Palomo, and J. O. Berger, "RobustGaSP: Robust Gaussian Stochastic Process Emulation in R," *The R Journal*, vol. 11, no. 1, pp. 112–136, 2019.

[34] R. Li and A. Sudjianto, "Analysis of computer experiments using penalized likelihood in Gaussian Kriging models," *Technometrics*, vol. 47, no. 2, pp. 111–120, 2005.

[35] D. Lopes, *Development and implementation of Bayesian computer model emulators*. PhD thesis, Duke University, 2011.

[36] J. Oakley, *Bayesian uncertainty analysis for complex computer codes*. PhD thesis, University of Sheffield, 1999.

[37] P. Ranjan, R. Haynes, and R. Karsten, "A computationally stable approach to Gaussian process interpolation of deterministic computer simulation data," *Technometrics*, vol. 53, no. 4, pp. 366–378, 2011.

[38] J. O. Berger, V. De Oliveira, and B. Sansó, "Objective Bayesian analysis of spatially correlated data," *Journal of the American Statistical Association*, vol. 96, no. 456, pp. 1361–1374, 2001.

[39] M. Gu, X. Wang, and J. O. Berger, "Robust Gaussian stochastic process emulation," *Annals of Statistics*, vol. 46, no. 6A, pp. 3038–3066, 2018.

[40] M. Gu, "Jointly robust prior for Gaussian stochastic process in emulation, calibration and variable selection," *Bayesian Analysis*, vol. 14, no. 1, 2018.

[41] J. Nocedal, "Updating quasi-newton matrices with limited storage," *Mathematics of computation*, vol. 35, no. 151, pp. 773–782, 1980.

[42] D. C. Liu and J. Nocedal, "On the limited memory bfgs method for large scale optimization," *Mathematical programming*, vol. 45, no. 1, pp. 503–528, 1989.

[43] A. Patra, A. Bauer, C. Nichita, E. B. Pitman, M. Sheridan, M. Bursik, B. Rupp, A. Webber, A. Stinton, L. Namikawa, and C. Renschler, "Parallel adaptive numerical simulation of dry avalanches over natural terrain," *Journal of Volcanology and Geothermal Research*, vol. 139, no. 1, pp. 1–21, 2005.

[44] M. J. Bayarri, J. O. Berger, E. S. Calder, K. Dalbey, S. Lunagomez, A. K. Patra, E. B. Pitman, E. T. Spiller, and R. L. Wolpert, "Using statistical and computer models to quantify volcanic hazards," *Technometrics*, vol. 51, pp. 402–413, 2009.

[45] M. A. Alvarez, L. Rosasco, and N. D. Lawrence, "Kernels for vector-valued functions: A review," *Foundations and Trends® in Machine Learning*, vol. 4, no. 3, pp. 195–266, 2012.

[46] E. Snelson and Z. Ghahramani, "Sparse Gaussian processes using pseudo-inputs," *Advances in neural information processing systems*, vol. 18, p. 1257, 2006.

[47] A. V. Vecchia, "Estimation and model identification for continuous spatial processes," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 50, no. 2, pp. 297–312, 1988.

[48] A. E. Gelfand, S. Banerjee, and D. Gamerman, "Spatial process modelling for univariate and multivariate dynamic spatial data," *Environmetrics: The official journal of the International Environmetrics Society*, vol. 16, no. 5, pp. 465–479, 2005.

[49] R. Paulo, G. García-Donato, and J. Palomo, "Calibration of computer models with multivariate output," *Computational Statistics and Data Analysis*, vol. 56, no. 12, pp. 3959–3974, 2012.

[50] Y. W. Teh, M. Seeger, and M. I. Jordan, "Semiparametric latent factor models," in *International Workshop on Artificial Intelligence and Statistics*, pp. 333–340, PMLR, 2005.

[51] M. Gu and H. Li, "Gaussian orthogonal latent factor processes for large incomplete matrices of correlated data," *Bayesian Analysis*, vol. 17, no. 4, pp. 1219 – 1244, 2022.

[52] R. Cerbino and V. Trappe, "Differential dynamic microscopy: probing wave vector dependent dynamics with a microscope," *Phys. Rev. Lett.*, vol. 100, no. 18, p. 188102, 2008.

[53] M. Gu, Y. Luo, Y. He, M. E. Helgeson, and M. T. Valentine, "Uncertainty quantification and estimation in differential dynamic microscopy," *Physical Review E*, vol. 104, no. 3, p. 034610, 2021.

[54] J. C. Crocker and D. G. Grier, "Methods of digital video microscopy for colloidal studies," *J. Colloid Interface Sci.*, vol. 179, no. 1, pp. 298–310, 1996.

[55] C. A. Schneider, W. S. Rasband, and K. W. Eliceiri, "NIH image to ImageJ: 25 years of image analysis," *Nature methods*, vol. 9, no. 7, pp. 671–675, 2012.

[56] Y. Luo, M. Gu, M. Park, X. Fang, Y. Kwon, J. M. Urueña, J. Read de Alaniz, M. E. Helgeson, C. M. Marchetti, and M. T. Valentine, "Molecular-scale substrate anisotropy, crowding and division drive collective behaviours in cell monolayers," *Journal of the Royal Society Interface*, vol. 20, no. 204, 2023.

[57] T. Yabuki and M. Matsu'Ura, "Geodetic data inversion using a bayesian information criterion for spatial distribution of fault slip," *Geophysical Journal International*, vol. 109, no. 2, pp. 363–375, 1992.

[58] Y. Aoki, P. Segall, T. Kato, P. Cervelli, and S. Shimada, "Imaging magma transport during the 1997 seismic swarm off the izu peninsula, japan," *Science*, vol. 286, no. 5441, pp. 927–930, 1999.

[59] D. Dzurisin, "A comprehensive approach to monitoring volcano deformation as a window on the eruption cycle," *Reviews of Geophysics*, vol. 41, no. 1, 2003.

[60] Y. Lin, X. Liu, P. Segall, and M. Gu, "Fast data inversion for high-dimensional dynamical systems from noisy measurements," *arXiv preprint arXiv:2501.01324*, 2025.

[61] G. Berkooz, P. Holmes, and J. L. Lumley, "The proper orthogonal decomposition in the analysis of turbulent flows," *Annual review of fluid mechanics*, vol. 25, no. 1, pp. 539–575, 1993.

[62] C. Lam, Q. Yao, and N. Bathia, "Estimation of latent factors for high-dimensional time series," *Biometrika*, vol. 98, no. 4, pp. 901–918, 2011.

[63] T. J. Mitchell and J. J. Beauchamp, "Bayesian variable selection in linear regression," *Journal of the american statistical association*, vol. 83, no. 404, pp. 1023–1032, 1988.

[64] J. Bernardo, M. Bayarri, J. Berger, A. Dawid, D. Heckerman, A. Smith, and M. West, "Bayesian factor regression models in the "large p, small n" paradigm," *Bayesian statistics*, vol. 7, pp. 733–742, 2003.

[65] R. Chen, D. Yang, and C.-H. Zhang, "Factor models for high-dimensional tensor time series," *Journal of the American Statistical Association*, vol. 117, no. 537, pp. 94–116, 2022.

[66] M. West and P. J. Harrison, *Bayesian Forecasting & Dynamic Models*. Springer Verlag, 2nd ed., 1997.

[67] P. Whittle, "On stationary processes in the plane," *Biometrika*, pp. 434–449, 1954.

[68] J. Hartikainen and S. Sarkka, "Kalman filtering and smoothing solutions to temporal Gaussian process regression models," in *Machine Learning for Signal Processing (MLSP), 2010 IEEE International Workshop on*, pp. 379–384, IEEE, 2010.

[69] M. Gu and Y. Xu, "Fast nonseparable Gaussian stochastic process with application to methylation level interpolation," *Journal of Computational and Graphical Statistics*, vol. 29, no. 2, pp. 250–260, 2020.

[70] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the royal statistical society: series B (methodological)*, vol. 39, no. 1, pp. 1–22, 1977.

[71] C. J. Wu, "On the convergence properties of the em algorithm," *The Annals of statistics*, pp. 95–103, 1983.

[72] W. I. Zangwill, "Nonlinear programming: a unified approach," *(No Title)*, 1969.

[73] M. E. Tipping and C. M. Bishop, "Probabilistic principal component analysis," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 61, no. 3, pp. 611–622, 1999.

[74] P. J. Schmid, "Dynamic mode decomposition and its variants," *Annual Review of Fluid Mechanics*, vol. 54, no. 1, pp. 225–254, 2022.

[75] B. O. Koopman, "Hamiltonian systems and transformation in hilbert space," *Proceedings of the National Academy of Sciences*, vol. 17, no. 5, pp. 315–318, 1931.

[76] C. W. Rowley, I. Mezić, S. Bagheri, P. Schlatter, and D. S. Henningson, "Spectral analysis of nonlinear flows," *Journal of fluid mechanics*, vol. 641, pp. 115–127, 2009.

[77] D. Higdon, J. Gattiker, B. Williams, and M. Rightley, "Computer model calibration using high-dimensional output," *Journal of the American Statistical Association*, vol. 103, no. 482, pp. 570–583, 2008.

[78] T. Lin, J. Lee, M. Helgeson, M. T. Valentine, Y. Luo, and M. Gu, "Model-free estimation in scattering analysis of microscopy," *arXiv preprint arXiv:2605.29424*, 2026.

[79] B. Sabass, M. L. Gardel, C. M. Waterman, and U. S. Schwarz, "High resolution traction force microscopy based on experimental and computational advances," *Biophysical journal*, vol. 94, no. 1, pp. 207–220, 2008.

[80] M. Gu, Y. Lin, V. C. Lee, and D. Y. Qiu, "Probabilistic forecast of nonlinear dynamical systems with uncertainty quantification," *Physica D: Nonlinear Phenomena*, vol. 457, p. 133938, 2024.

[81] A. E. Gelfand, A. M. Schmidt, S. Banerjee, and C. Sirmans, "Nonstationary multivariate process modeling through spatially varying coregionalization," *Test*, vol. 13, no. 2, pp. 263–312, 2004.

[82] R. H. Shumway and D. S. Stoffer, "An approach to time series smoothing and forecasting using the em algorithm," *Journal of time series analysis*, vol. 3, no. 4, pp. 253–264, 1982.

[83] I. Mezić, "Analysis of fluid flows via spectral properties of the Koopman operator," *Annual Review of Fluid Mechanics*, vol. 45, pp. 357–378, 2013.

[84] K. Metaxoglou and A. Smith, "Maximum likelihood estimation of varma models using a state-space em algorithm," *Journal of Time Series Analysis*, vol. 28, no. 5, pp. 666–685, 2007.

[85] N. M. Bartlow, S. Miyazaki, A. M. Bradley, and P. Segall, "Space-time correlation of slip and tremor during the 2009 cascadia slow slip event," *Geophysical Research Letters*, vol. 38, no. 18, 2011.

[86] M. Gu, X. Fang, and Y. Lin, *FastGaSP: Fast and Exact Computation of Gaussian Stochastic Process*, 2025. R package version 0.6.1.

[87] K. R. Anderson, I. A. Johanson, M. R. Patrick, M. Gu, P. Segall, M. P. Poland, E. K. Montgomery-Brown, and A. Miklius, "Magma reservoir failure and the onset of caldera collapse at Kīlauea volcano in 2018," *Science*, vol. 366, no. 6470, 2019.

[88] R. Bürgmann, "The geophysics, geology and mechanics of slow fault slip," *Earth and Planetary Science Letters*, vol. 495, pp. 112–134, 2018.

[89] Y. Bock, D. C. Agnew, P. Fang, J. F. Genrich, B. H. Hager, T. A. Herring, K. W. Hudnut, R. W. King, S. Larsen, J.-B. Minster, *et al.*, "Detection of crustal deformation from the landers earthquake sequence using continuous geodetic measurements," *Nature*, vol. 361, no. 6410, pp. 337–340, 1993.

[90] J. Gomberg, C. 2007, and B. W. Group, "Slow-slip phenomena in cascadia from 2007 and beyond: A review," *Bulletin*, vol. 122, no. 7-8, pp. 963–978, 2010.

[91] K. Aki, "Quantitative seismology," *Theory and models.*, vol. 1, p. 932p, 1980.

[92] C. W. Gardiner, "Handbook of stochastic methods for physics, chemistry and the natural sciences," *Springer series in synergetics*, 1985.

[93] A. Meucci, "Review of statistical arbitrage, cointegration, and multivariate Ornstein-Uhlenbeck," *Cointegration, and Multivariate Ornstein-Uhlenbeck (May 14, 2009)*, 2009.

[94] Z. Wen and W. Yin, "A feasible method for optimization with orthogonality constraints," *Mathematical Programming*, vol. 142, no. 1-2, pp. 397–434, 2013.

[95] J. Bai and S. Ng, "Determining the number of factors in approximate factor models," *Econometrica*, vol. 70, no. 1, pp. 191–221, 2002.

[96] H. Hotelling, "Analysis of a complex of statistical variables into principal components.," *Journal of educational psychology*, vol. 24, no. 6, p. 417, 1933.

[97] H. S. Carslaw, *Introduction to the Mathematical Theory of the Conduction of Heat in Solids*. Macmillan and Company, limited, 1906.

[98] H. Lu and D. M. Tartakovsky, "Prediction accuracy of dynamic mode decomposition," *SIAM Journal on Scientific Computing*, vol. 42, no. 3, pp. A1639–A1662, 2020.

[99] K. E. Soetaert, T. Petzoldt, and R. W. Setzer, "Solving differential equations in r: package desolve," *Journal of statistical software*, vol. 33, no. 9, 2010.

[100] V. Picheny, T. Wagner, and D. Ginsbourger, "A benchmark of kriging-based infill criteria for noisy optimization," *Structural and Multidisciplinary Optimization*, vol. 48, no. 3, pp. 607–626, 2013.

[101] C. DeMets, R. G. Gordon, and D. F. Argus, "Geologically current plate motions," *Geophysical journal international*, vol. 181, no. 1, pp. 1–80, 2010.

[102] A. G. Wech and K. C. Creager, "Automated detection and location of Cascadia tremor," *Geophysical Research Letters*, vol. 35, no. 20, 2008.

[103] Y. Fukushima, V. Cayol, and P. Durand, "Finding realistic dike models from interferometric synthetic aperture radar data: The february 2000 eruption at piton de la fournaise," *Journal of Geophysical Research: Solid Earth*, vol. 110, no. B3, 2005.

[104] A. L. Thomas, "Polu3d: A three-dimensional, polygonal element, displacement discontinuity boundary element computer program with applications to fractures, faults, and cavities in the earth's crust," *Master Thesis at Stanford University*, 1993.

[105] A. G. Wech, "Interactive tremor monitoring," *Seismological Research Letters*, vol. 81, no. 4, pp. 664–669, 2010.

[106] L. Baracaldo, B. King, H. Yan, Y. Lin, N. Miolane, and M. Gu, "Unsupervised cell segmentation by fast gaussian processes," *arXiv preprint arXiv:2505.18902*, 2025.

[107] W. Coffey and Y. P. Kalmykov, *The Langevin equation: with applications to stochastic problems in physics, chemistry and electrical engineering*, vol. 27. World Scientific, 2012.

[108] T. Vicsek, A. Czirók, E. Ben-Jacob, I. Cohen, and O. Shochet, "Novel type of phase transition in a system of self-driven particles," *Physical review letters*, vol. 75, no. 6, p. 1226, 1995.

[109] J. Toner and Y. Tu, "Flocks, herds, and schools: A quantitative theory of flocking," *Physical review E*, vol. 58, no. 4, p. 4828, 1998.

[110] H. W. Hethcote, "The mathematics of infectious diseases," *SIAM review*, vol. 42, no. 4, pp. 599–653, 2000.

[111] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, 2004.

[112] G. Kitagawa, "Monte carlo filter and smoother for non-gaussian nonlinear state space models," *Journal of computational and graphical statistics*, vol. 5, no. 1, pp. 1–25, 1996.

[113] G. Evensen, "Sequential data assimilation with a nonlinear quasi-geostrophic model using monte carlo methods to forecast error statistics," *Journal of Geophysical Research: Oceans*, vol. 99, no. C5, pp. 10143–10162, 1994.

[114] L. Sirovich, "Turbulence and the dynamics of coherent structures, parts i, ii and iii," *Quart. Appl. Math.*, pp. 561–590, 1987.

[115] S. L. Brunton, M. Budišić, E. Kaiser, and J. N. Kutz, "Modern koopman theory for dynamical systems," *arXiv preprint arXiv:2102.12086*, 2021.

[116] S. Le Clainche and J. M. Vega, "Higher order dynamic mode decomposition," *SIAM Journal on Applied Dynamical Systems*, vol. 16, no. 2, pp. 882–925, 2017.

[117] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley, "A data–driven approximation of the koopman operator: Extending dynamic mode decomposition," *Journal of Nonlinear Science*, vol. 25, no. 6, pp. 1307–1346, 2015.

[118] E. Kaiser, J. N. Kutz, and S. L. Brunton, "Sparse identification of nonlinear dynamics for model predictive control in the low-data limit," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 474, no. 2219, 2018.

[119] S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proceedings of the national academy of sciences*, vol. 113, no. 15, pp. 3932–3937, 2016.

[120] M. Korda and I. Mezić, "Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control," *Automatica*, vol. 93, pp. 149–160, 2018.

[121] C. Folkestad, D. Pastor, I. Mezic, R. Mohr, M. Fonoberova, and J. Burdick, "Extended dynamic mode decomposition with learned koopman eigenfunctions for prediction and control," in *2020 american control conference (acc)*, pp. 3906–3913, IEEE, 2020.

[122] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," *Advances in neural information processing systems*, vol. 31, 2018.

[123] Z. Li, "Neural operator: Learning maps between function spaces," in *2021 Fall Western Sectional Meeting*, AMS, 2021.

[124] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Fourier neural operator for parametric partial differential equations," *arXiv preprint arXiv:2010.08895*, 2020.

[125] B. J. Shields, J. Stevens, J. Li, M. Parasram, F. Damani, J. I. M. Alvarado, J. M. Janey, R. P. Adams, and A. G. Doyle, "Bayesian reaction optimization as a tool for chemical synthesis," *Nature*, vol. 590, no. 7844, pp. 89–96, 2021.

[126] X. Fang, M. Gu, and J. Wu, "Reliable emulation of complex functionals by active learning with error control," *The Journal of Chemical Physics*, vol. 157, no. 21, 2022.

[127] A. W. van der Vaart and J. H. van Zanten, "Rates of contraction of posterior distributions based on Gaussian process priors," *The Annals of Statistics*, vol. 36, no. 3, pp. 1435–1463, 2008.

[128] M. Gu, X. Fang, and Y. Luo, "Data-driven model construction for anisotropic dynamics of active matter," *PRX Life*, vol. 1, no. 1, p. 013009, 2023.

[129] M. S. Hybertsen and S. G. Louie, "Electron correlation in semiconductors and insulators: Band gaps and quasiparticle energies," *Physical Review B*, vol. 34, no. 8, p. 5390, 1986.

[130] M. Rohlfing and S. G. Louie, "Electron-hole excitations and optical spectra from first principles," *Physical Review B*, vol. 62, no. 8, p. 4927, 2000.

[131] D. Sangalli, S. Dal Conte, C. Manzoni, G. Cerullo, and A. Marini, "Nonequilibrium optical properties in semiconductors from first principles: A combined theoretical and experimental study of bulk silicon," *Physical Review B*, vol. 93, no. 19, p. 195205, 2016.

[132] C. Attaccalite, M. Grüning, and A. Marini, "Real-time approach to the optical properties of solids and nanostructures: Time-dependent bethe-salpeter equation," *Physical Review B—Condensed Matter and Materials Physics*, vol. 84, no. 24, p. 245110, 2011.

[133] Y.-H. Chan, D. Y. Qiu, F. H. da Jornada, and S. G. Louie, "Giant exciton-enhanced shift currents and direct current conduction with subbandgap photo excitations produced by many-electron interactions," *Proceedings of the National Academy of Sciences*, vol. 118, no. 25, p. e1906938118, 2021.

[134] E. Perfetto, D. Sangalli, A. Marini, and G. Stefanucci, "First-principles approach to excitons in time-resolved and angle-resolved photoemission spectra," *Physical Review B*, vol. 94, no. 24, p. 245303, 2016.

[135] E. Perfetto, D. Sangalli, M. Palummo, A. Marini, and G. Stefanucci, "First-principles nonequilibrium green's function approach to ultrafast charge migration in glycine," *Journal of Chemical Theory and Computation*, vol. 15, no. 8, pp. 4526–4534, 2019.

[136] M. Iskin and C. Sá de Melo, "Bcs-bec crossover of collective excitations in two-band superfluids," *Physical Review B—Condensed Matter and Materials Physics*, vol. 72, no. 2, p. 024512, 2005.

[137] J. Yin, Y.-h. Chan, F. H. da Jornada, D. Y. Qiu, S. G. Louie, and C. Yang, "Using dynamic mode decomposition to predict the dynamics of a two-time non-equilibrium green's function," *Journal of Computational Science*, vol. 64, p. 101843, 2022.

[138] J. Yin, Y.-h. Chan, F. H. da Jornada, D. Y. Qiu, C. Yang, and S. G. Louie, "Analyzing and predicting non-equilibrium many-body dynamics via dynamic mode decomposition," *Journal of Computational Physics*, vol. 477, p. 111909, 2023.

[139] M. Gu, F. Xie, and L. Wang, "A theoretical framework of the scaled gaussian stochastic process in prediction and calibration," *SIAM/ASA Journal on Uncertainty Quantification*, vol. 10, no. 4, pp. 1435–1460, 2022.

[140] H. Wendland, *Scattered data approximation*, vol. 17. Cambridge university press, 2004.

[141] H. Zhang, "Inconsistent estimation and asymptotically equal interpolations in model-based geostatistics," *Journal of the American Statistical Association*, vol. 99, no. 465, pp. 250–261, 2004.

[142] A. C. Davison and D. V. Hinkley, *Bootstrap methods and their application*. No. 1, Cambridge university press, 1997.

[143] J. Durbin and S. J. Koopman, *Time series analysis by state space methods*, vol. 38. OUP Oxford, 2012.

[144] H. Arbabi and I. Mezic, "Ergodic theory, dynamic mode decomposition, and computation of spectral properties of the Koopman operator," *SIAM Journal on Applied Dynamical Systems*, vol. 16, no. 4, pp. 2096–2126, 2017.

[145] I. T. Jolliffe, *Principal component analysis for special types of data*. Springer, 2002.

[146] G. Petris, S. Petrone, and P. Campagnoli, *Dynamic linear models*. Springer, 2009.

[147] R. Prado and M. West, *Time series: modeling, computation, and inference*. Chapman and Hall/CRC, 2010.

[148] G. Evensen, *Data assimilation: the ensemble Kalman filter*. Springer Science & Business Media, 2009.

[149] M. Roth, G. Hendeby, C. Fritsche, and F. Gustafsson, "The ensemble kalman filter: a signal processing perspective," *EURASIP Journal on Advances in Signal Processing*, vol. 2017, no. 1, pp. 1–16, 2017.

[150] P. Maioli, T. Meunier, S. Gleyzes, A. Auffeves, G. Nogues, M. Brune, J. Raimond, and S. Haroche, "Nondestructive rydberg atom counting with mesoscopic fields in a cavity," *Physical review letters*, vol. 94, no. 11, p. 113601, 2005.

[151] Y.-H. Chan, D. Y. Qiu, F. H. da Jornada, and S. G. Louie, "Giant self-driven exciton-floquet signatures in time-resolved photoemission spectroscopy of mos2 from time-dependent gw approach," *Proceedings of the National Academy of Sciences*, vol. 120, no. 32, p. e2301957120, 2023.

[152] D. Y. Qiu, F. H. Da Jornada, and S. G. Louie, "Optical spectrum of mos 2: many-body effects and diversity of exciton states," *Physical review letters*, vol. 111, no. 21, p. 216805, 2013.

[153] D. Y. Qiu, F. H. Da Jornada, and S. G. Louie, "Screening and many-body effects in two-dimensional crystals: Monolayer mos 2," *Physical Review B*, vol. 93, no. 23, p. 235435, 2016.

[154] M. M. Ugeda, A. J. Bradley, S.-F. Shi, F. H. Da Jornada, Y. Zhang, D. Y. Qiu, W. Ruan, S.-K. Mo, Z. Hussain, Z.-X. Shen, *et al.*, "Giant bandgap renormalization and excitonic effects in a monolayer transition metal dichalcogenide semiconductor," *Nature materials*, vol. 13, no. 12, pp. 1091–1095, 2014.

[155] A. Chernikov, T. C. Berkelbach, H. M. Hill, A. Rigosi, Y. Li, B. Aslan, D. R. Reichman, M. S. Hybertsen, and T. F. Heinz, "Exciton binding energy and nonhydrogenic rydberg series in monolayer ws 2," *Physical review letters*, vol. 113, no. 7, p. 076802, 2014.

[156] G. Wang, A. Chernikov, M. M. Glazov, T. F. Heinz, X. Marie, T. Amand, and B. Urbaszek, "Colloquium: Excitons in atomically thin transition metal dichalcogenides," *Reviews of Modern Physics*, vol. 90, no. 2, p. 021001, 2018.

[157] J. Deslippe, G. Samsonidze, D. A. Strubbe, M. Jain, M. L. Cohen, and S. G. Louie, "Berkeleygw: A massively parallel computer package for the calculation of the quasiparticle and optical properties of materials and nanostructures," *Computer Physics Communications*, vol. 183, no. 6, pp. 1269–1289, 2012.

[158] P. Giannozzi, S. Baroni, N. Bonini, M. Calandra, R. Car, C. Cavazzoni, D. Ceresoli, G. L. Chiarotti, M. Cococcioni, I. Dabo, *et al.*, "Quantum espresso: a modular and open-source software project for quantum simulations of materials," *Journal of physics: Condensed matter*, vol. 21, no. 39, p. 395502, 2009.

[159] P. Scherpelz, M. Govoni, I. Hamada, and G. Galli, "Implementation and validation of fully relativistic gw calculations: spin–orbit coupling in molecules, nanocrystals, and solids," *Journal of chemical theory and computation*, vol. 12, no. 8, pp. 3523–3544, 2016.

[160] H. Liu, Y. Li, Y. S. You, S. Ghimire, T. F. Heinz, and D. A. Reis, "High-harmonic generation from an atomically thin semiconductor," *Nature Physics*, vol. 13, no. 3, pp. 262–265, 2017.

[161] P. Lipavskỳ, V. Špička, and B. Velickỳ, "Generalized kadanoff-baym ansatz for deriving quantum transport equations," *Physical Review B*, vol. 34, no. 10, p. 6933, 1986.

[162] T. Zhou, P. Niu, L. Sun, R. Jin, *et al.*, "One fits all: Power general time series analysis by pretrained lm," *Advances in neural information processing systems*, vol. 36, pp. 43322–43355, 2023.

[163] A. Gu and T. Dao, "Mamba: Linear-time sequence modeling with selective state spaces," *arXiv preprint arXiv:2312.00752*, 2023.

[164] H. Wolkowicz, "A note on lack of strong duality for quadratic problems with orthogonal constraints," *European Journal of Operational Research*, vol. 143, no. 2, pp. 356–364, 2002.

[165] E. K. Donald *et al.*, "The art of computer programming," *Sorting and searching*, vol. 3, no. 426-458, p. 4, 1999.