GAUSSIAN PROCESS REGRESSION IN HIGH-DIMENSIONAL AND STRUCTURED DOMAINS
VARIATIONAL INFERENCE AND ACTIVE LEARNING ON MANIFOLDS

BY

YUANXING CHENG

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Applied Mathematics
at the Illinois Institute of Technology

Approved _____
Adviser

Approved _____
Co-Adviser

Chicago, Illinois
August 2025

ACKNOWLEDGMENT

I would like to express my deepest gratitude to my advisors, Prof. Lulu Kang and Prof. Chun Liu, for their continuous support, guidance, and encouragement throughout my Ph.D. journey. Their expertise and mentorship have been invaluable.

I also thank my committee members, Prof. Yiwei Wang, Prof. Shuwang Li, Prof. Xiaofan Li and Prof. Binghui Wang, for their thoughtful feedback and kind support during the ups and downs of my entire graduate study. Their insights and patience have been greatly appreciated.

Without the dedication and care of my advisors and committee members, this thesis would not have been possible.

AUTHORSHIP STATEMENT

I, Yuanxing Cheng, attest that the work in this thesis is substantially my own.

In accordance with the disciplinary norms of authorship in Applied Mathematics, the following collaborations occurred in this thesis. (Please consult Appendix S of the IIT Faculty Handbook for more details.)

Portions of this dissertation are adapted from the following collaborative papers, co-authored with Lulu Kang, Chun Liu, and Yiwei Wang.

- *Energetic Variational Inference for Gaussian Process Regression* [1]
- *Active Learning of Manifold Gaussian Processes via ALC Acquisition* [2]

My advisor, Prof. Lulu Kang, formerly of IIT, now at University of Massachusetts Amherst, contributed to the conceptualization, guidance, and overall progress of both research projects presented in this thesis, as is typical for a Ph.D. supervisor. My advisor Prof. Chun Liu, who served as chair of my committee and Prof. Yiwei Wang, contributed to the theoretical and coding guidance for both research projects. Their help is fundamental and gratefully acknowledged.

Chapter 2 includes material adapted from both papers. I synthesized, reorganized, and expanded the material to provide a unified and comprehensive overview tailored for this thesis. The content has been revised and extended beyond the original manuscripts for clarity and completeness in the context of this dissertation.

Chapter 3 is based on material from [1]. My co-authors contributed to the development of the EVI method and theoretical framework. I co-developed the codebase, conducted all numerical experiments and analysis, and revised and extended the content for the thesis.

Chapter 4 includes material based on the second paper [2], in which I developed the methodology, carried out all theoretical and experimental components, and wrote the manuscript. Prof. Lulu Kang provided valuable guidance and feedback. The chapter has been substantially revised and expanded beyond the original manuscript.

All co-authors have been appropriately acknowledged, and their contributions are clearly attributed above. All publications are cited in the References section. Portions of this dissertation include material from the following IEEE-copyrighted publications. Only the accepted versions of these publications have been included in this dissertation in compliance with IEEE reuse policy:

1. Lulu Kang, Yuanxing Cheng, Yiwei Wang, and Chun Liu, "Energetic Variational Gaussian Process Regression," *Proc. 2024 Winter Simulation Conference (WSC)*, IEEE.

2. Yuanxing Cheng, Lulu Kang, Yiwei Wang, and Chun Liu, "Active Learning for Manifold Gaussian Process Regression," *Proc. 2025 Winter Simulation Conference (WSC)*, IEEE.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Gaussian process (GP) regression offers a useful framework for surrogate modeling, providing both predictive flexibility and rigorous uncertainty quantification. As computational models grow in complexity, two fundamental challenges arise: how to perform reliable inference with limited data, and how to select data points effectively when evaluations are expensive. This thesis addresses both challenges through two complementary approaches.

First, we develop an inference framework called Energetic Variational Inference for Gaussian Processes (EVI-GP). Instead of relying on variational bounds or sampling-based methods, EVI-GP constructs an energy-based functional grounded in thermodynamic principles. Using a particle-based implicit solver, the method captures the posterior shape and mode while incorporating shrinkage priors for variable selection. This results in a stable, interpretable inference mechanism that is effective in high-dimensional or data-scarce settings.

Second, we propose an active learning framework called Active Learning for Manifold Gaussian Processes (ALmGP). This method uncovers a low-dimensional latent structure using an autoencoder trained together with the GP model, and places the GP within this space. Data acquisition is guided by the Active Learning Cohn (ALC) criterion, which selects new samples to reduce global predictive uncertainty. This approach improves sample efficiency and enhances robustness in complex, nonlinear modeling tasks.

CHAPTER 1

INTRODUCTION

1.1 Background and Motivation

Computer experiments refer to simulations of physical, biological, or engineering systems using numerical models. Unlike analytical solutions or empirical studies, these simulations are often high-fidelity and computationally expensive, requiring significant resources for each evaluation. Applications range from climate modeling to aerospace design and biomedical analysis. For instance, in computational fluid dynamics, running a high-resolution Navier-Stokes simulation for each design iteration may require hours or even days of computation [3, 4].

To reduce computational cost, surrogate models are widely adopted to approximate the output of these complex simulations. Among them, Gaussian Process (GP) regression has become a popular choice due to its flexibility, non-parametric structure, and the ability to provide uncertainty quantification [5]. A GP model predicts outputs at unobserved inputs by learning correlations from observed data, and quantifies predictive confidence through its posterior variance. Despite these strengths, GP regression faces significant challenges when applied to complex, high-dimensional scientific systems.

1.2 Problem Statement and Key Challenges

In such settings, several interrelated difficulties arise that hold back the scalability and effectiveness of GP models.

First, GP regression suffers from the curse of dimensionality: as the number of input variables grows, the amount of data required to maintain predictive accuracy

increases exponentially [6]. In practice, acquiring such data is often prohibitively expensive. Each new observation may require a costly simulation or physical experiment [7], making sample efficiency a central concern.

Even when data is available, inference remains a bottleneck. Standard GP models involve matrix operations with cubic time complexity [8], rendering them inefficient for large datasets or repeated updates. Approximate inference techniques—such as Markov chain Monte Carlo (MCMC) and variational inference (VI)—are widely used to alleviate these issues, but they come with their own limitations: MCMC is computationally expensive, and VI often suffers from bias and convergence instability [6].

These challenges collectively limit the scalability of GP regression. They motivate the need for new inference and learning strategies that are both data-efficient and computationally tractable—especially in high-dimensional, resource-constrained modeling scenarios.

1.3 Research Questions

In response to these constraints, this thesis focus on two research questions.

The first is how to improve the inference. To be specific, we want to have a fast and stable method to sample the posterior distribution, especially in Bayesian Gaussian Process framework.

The second question focuses on improving data acquisition. As in some high dimensional experiments, a good active learning strategy can improve the overall performance of the model.

These two questions are developed independently but share a common goal: to make GP regression more efficient, robust, and applicable to real-world scientific

modeling tasks.

1.4 Contributions

My contributions in answering these two questions are summarized below.

We first apply a novel variational method based on the energetic variational inference (EVI) framework to sample from the posterior distribution. By reformulating Bayesian posterior inference as a dynamic energy minimization problem, the method enables a fast, stable, and guaranteed posterior approximation, supports automatic variable selection, and avoids the high computational cost of sampling.

We propose an active learning strategy that embeds high-dimensional data into a latent manifold using a new manifold Gaussian Process model, whose training process rely on both the negative log likelihood of the manifold Gaussian process and the reconstruction loss from an autoencoder architecture. New samples are selected based on the Active Learning Cohn (ALC) criterion with a pre-screening step, so as to achieve global uncertainty reduction with fewer data points and fast computation. We found that applying this new active learning method greatly reduced the training data size needed to reach the error tolerance.

1.5 Thesis Outline

The remainder of this thesis is organized into four chapters, each corresponding to a specific aspect of the research:

- **Chapter 2: Background and Related Work** introduces the theoretical foundations of Gaussian Process regression, including kernel design, Bayesian inference, and active learning. It also reviews related work in surrogate modeling and manifold learning.
- **Chapter 3: Energetic Variational Inference for GP Regression** presents

a variational inference framework based on energy minimization. This method enables stable posterior approximation and automatic variable selection via particle-based dynamics.

- **Chapter 4: Active Learning on Latent Manifolds** proposes an active learning strategy that embeds high-dimensional data into a low-dimensional latent space using an autoencoder. Sample selection is guided by the ALC criterion to reduce global uncertainty.

- **Chapter 5: Discussion and Conclusion** summarizes the key contributions, discusses current limitations, and outlines future research directions.

CHAPTER 2

BACKGROUND AND RELATED WORK

2.1 Gaussian Process Regression

[1]This section provides a detailed exploration of the fundamental principles underlying GPR, followed by an in-depth analysis of the challenges posed by the curse of dimensionality specifically within the context of GPR.

2.1.1 Fundamentals of Gaussian Process Regression

Gaussian Processes (GPs) are nonparametric stochastic processes that define a distribution over functions. In regression tasks, GPs are used to model unknown functions $f(x)$ by placing a GP prior over f and updating it using observed data via Bayes' rule. The result is a posterior distribution that provides both a predictive mean and uncertainty at each input location. This framework enables interpolation with principled uncertainty quantification and has been widely adopted in surrogate modeling and Bayesian optimization [5].

Mathematically, a Gaussian Process Regression model can be defined by: Given n training inputs $\boldsymbol{x}_i \in \mathcal{X} \subset \mathbb{R}^D$ and corresponding outputs $y_i \in \mathcal{Y} \subset \mathbb{R}$, the Gaussian Process Regression model assumes that the outputs satisfy:

$$y_i = \boldsymbol{g}(\boldsymbol{x}_i)^\top \boldsymbol{\beta} + Z(\boldsymbol{x}_i) + \epsilon_i, i = 1, 2, \ldots, n \tag{2.1.1}$$

Where $\boldsymbol{g}(\boldsymbol{x}_i)^\top \boldsymbol{\beta}$ is the mean function. $\boldsymbol{\beta}$ are the coefficients and $\boldsymbol{g}(\boldsymbol{x}_i)$ are the effects, which can include zero order effect, 1, first order effects $x_1, x_2, \ldots, x_D$ and second order

[1]This chapter is adapted from [2, 1]. See authorship statement for more details.

effects, $x_1^2, x_1x_2, x_1x_3, \ldots, x_D^2$, etc, and for other choices see [1, 9]. The mean function is often sufficient to be set to $\mathbf{0}$ [10]. In the remainder of this thesis, we will keep the zero mean assumption unless otherwise stated. $Z(\cdot)$ is assumed to be a stationary Gaussian Process, i.e., $\mathbb{E}\left[Z(\boldsymbol{x})\right] = 0$ and $\text{Cov}\left[Z(\boldsymbol{x}_i), Z(\boldsymbol{x}_j)\right] = \tau^2 k(\boldsymbol{x}_i, \boldsymbol{x}_j; \boldsymbol{\omega})$. The last term is the random noise ϵ_i, and they are independently and identically distributed following $\mathcal{N}(0, \sigma_\epsilon^2)$.

For the correlation function, τ^2 is a scaling parameter, $k(\boldsymbol{x}_i, \boldsymbol{x}_j; \boldsymbol{\omega}) : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$ is the correlation of the stochastic process and $\boldsymbol{\omega} = (\omega_1, \omega_2, \ldots, \omega_D) \in \mathbb{R}^D$. The stationary assumption also requires that k to be positive (semi)definite function. For the kernel function, one of the most commonly used kernels is the Gaussian kernel, a specific example of a radial basis function (RBF) kernel. RBF kernels are a general class of positive-definite kernels that depend only on the distance between inputs. The Gaussian kernel takes the form:

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j; \boldsymbol{\omega}) = \exp\left\{-\sum_{k=1}^{D} \omega_k (x_{ik} - x_{jk})^2\right\}.$$

Other common choices of k include Linear kernel,

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j; \boldsymbol{\omega}) = \exp\left\{-\sum_{k=1}^{D} \omega_k x_{ik} x_{jk}\right\}.$$

and the Matérn kernel families [11]

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j; \boldsymbol{\omega}) = \prod_{k=1}^{D} \frac{1}{\Gamma(\nu) 2^{\nu-1}} \left(\frac{2\sqrt{\nu}|x_{ik} - x_{jk}|}{\theta_k}\right)^{\nu} K_\nu \left(\frac{2\sqrt{\nu}|x_{ik} - x_{jk}|}{\theta_k}\right),$$

where $\nu > 0$ is a smoothness parameter, θ_k is the length-scale along the k-th input dimension, and $K_\nu(\cdot)$ is the modified Bessel function of the second kind. They are proposed to encode extra data structure [5].

Under the above model assumption, the outputs $y_i(\boldsymbol{x}_i)$ is a Gaussian Process with mean $\mathbb{E}[y_i] = 0$ and correlation

$$\begin{aligned}\operatorname{Cov}[y_i, y_j] &= \tau^2 k(\boldsymbol{x}_i, \boldsymbol{x}_j; \boldsymbol{\omega}) + \sigma_\epsilon^2 \delta(i,j), \qquad \forall \boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{X} \\ &= \tau^2 \left(k(\boldsymbol{x}_i, \boldsymbol{x}_j; \boldsymbol{\omega}) + \sigma_\epsilon^2 / \tau^2 \delta(i,j) \right) \\ &\triangleq \tau^2 \left(k_{ij} + \eta \delta_{ij} \right)\end{aligned}$$

where $\delta(i,j)$ is the kronecker delta, and we denote $\eta \triangleq \sigma_\epsilon^2/\tau^2$ the noise-to-signal ratio. Denote $\boldsymbol{X}_n \in \mathbb{R}^{n \times D}$ and $\boldsymbol{Y}_n \in \mathbb{R}^n$ as all the training inputs and outputs. $\boldsymbol{Y}_n$ is a multivariate normal $\boldsymbol{Y}_n \sim \mathcal{N}\left(\boldsymbol{G}\boldsymbol{\beta}, \tau^2 \left(\boldsymbol{K}_n + \eta \boldsymbol{I}_n\right)\right)$ where $\boldsymbol{G}$ is a matrix of row vectors $\boldsymbol{g}(\boldsymbol{x}_i)^\top$'s and $\boldsymbol{I}_n$ is an identity matrix. A small η indicating the nugget effect is usually included here to ensure that the covariance matrix is positive definite [12].

Proposition 2.1.1. *Given the Gaussian Process model* (2.1.1)*, the predictive distribution at any test point $\boldsymbol{x} \in \mathcal{X}$ is a Gaussian random variable with the following distribution.*

$$y(\boldsymbol{x}) \mid \boldsymbol{X}_n, \boldsymbol{Y}_n, \tau^2, \boldsymbol{\omega}, \eta \sim \mathcal{N}(\hat{\mu}(\boldsymbol{x}), \sigma^2(\boldsymbol{x})),$$

where

$$\begin{aligned}\hat{\mu}(\boldsymbol{x}) &= k(\boldsymbol{x}, \boldsymbol{X}_n)^\top \left(\boldsymbol{K}_n + \eta \boldsymbol{I}_n\right)^{-1} \tilde{\boldsymbol{Y}}_n \in \mathbb{R} \\ \tilde{\boldsymbol{Y}} &\triangleq \boldsymbol{Y}_n - \boldsymbol{G}\boldsymbol{\beta} \in \mathbb{R}^n \\ \sigma^2(\boldsymbol{x}) &= \tau^2 \left(1 + \eta - k(\boldsymbol{x}, \boldsymbol{X}_n)^\top \left(\boldsymbol{K}_n + \eta \boldsymbol{I}_n\right)^{-1} k(\boldsymbol{x}, \boldsymbol{X}_n)\right) \in \mathbb{R} \\ k(\boldsymbol{x}, \boldsymbol{X}_n) &\triangleq \left[k(\boldsymbol{x}, \boldsymbol{x}_1), k(\boldsymbol{x}, \boldsymbol{x}_2), \ldots, k(\boldsymbol{x}, \boldsymbol{x}_n)\right]^\top \in \mathbb{R}^n\end{aligned}$$

This is based on the distribution of joint multivariate normal distribution and the proof is in appendix. These equations are central to GPR. The predictive mean $\hat{\mu}(\boldsymbol{x})$ provides the best estimate of the function values at the test points, while the predictive covariance matrix $\sigma^2(x)$ quantifies the uncertainty associated with these

predictions. These equations provide closed-form predictions with uncertainty, making GPR appealing for scientific modeling under limited data. However, exact inference requires $\mathcal{O}(n^3)$ time because of matrix inversion, and the model may degrade in high dimensions.

2.1.2 Model Selection and Hyperparameter Learning

To complete the specification of the GP model, the key hyperparameters must be estimated from data, including the length-scales, process variance, and nugget. In practice, these parameters control the overall scale and smoothness of the model, and their values significantly influence the posterior predictive behavior. A widely adopted approach for hyperparameter learning is to maximize the marginal likelihood of the observed data under the GP prior.

Under the assumption that $\boldsymbol{Y}_n \sim \mathcal{N}(\boldsymbol{G}\boldsymbol{\beta}, \tau^2(\boldsymbol{K}_n+\eta\boldsymbol{I}_n))$, a multivariate normal distribution. The likelihood function of the parameters admits the form:

$$\begin{aligned}\mathcal{L}(\tau^2, \boldsymbol{\omega}, \eta) &= (2\pi)^{-n/2} \det(\tau^2(\boldsymbol{K}_n + \eta\boldsymbol{I}_n))^{-1/2} \\ &\qquad \exp\left(-\frac{1}{2\tau^2}\left(\boldsymbol{Y}_n - \boldsymbol{G}\boldsymbol{\beta}\right)^\top \left(\boldsymbol{K}_n + \eta\boldsymbol{I}_n\right)^{-1}\left(\boldsymbol{Y}_n - \boldsymbol{G}\boldsymbol{\beta}\right)\right) \\ &= (2\pi\tau^2)^{-n/2} \det(\boldsymbol{K}_n + \eta\boldsymbol{I}_n)^{-1/2} \\ &\qquad \exp\left(-\frac{1}{2\tau^2}\tilde{\boldsymbol{Y}}_n^\top(\boldsymbol{K}_n + \eta\boldsymbol{I}_n)^{-1}\tilde{\boldsymbol{Y}}_n\right).\end{aligned}$$

Taking the logarithm yields the log-marginal likelihood:

$$\begin{aligned}\ell(\tau^2, \boldsymbol{\omega}, \eta) &= \log \mathcal{L}(\tau^2, \boldsymbol{\omega}, \eta) \\ &= -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log\tau^2 - \frac{1}{2}\log\det(\boldsymbol{K}_n + \eta\boldsymbol{I}_n) - \frac{1}{2\tau^2}\tilde{\boldsymbol{Y}}_n^\top(\boldsymbol{K}_n + \eta\boldsymbol{I}_n)^{-1}\tilde{\boldsymbol{Y}}_n.\end{aligned} \tag{2.1.2}$$

We proceed to derive the closed-form expression for the noise-to-signal parameter.

When η is held fixed, differentiating ℓ with respect to τ^2 gives

$$\frac{\mathrm{d}\ell(\tau^2, \boldsymbol{\omega}, \eta)}{\mathrm{d}\tau^2} = 0 - \frac{n}{2}\frac{1}{\tau^2} - 0 + \frac{1}{2}(\tau^2)^{-2}\tilde{\boldsymbol{Y}}_n^\top(\boldsymbol{K}_n + \eta\boldsymbol{I}_n)^{-1}\tilde{\boldsymbol{Y}}_n$$

and setting this derivative to zero yields:

$$\frac{n}{2}\frac{1}{\tau^2} = \frac{1}{2}(\tau^2)^{-2}\tilde{\boldsymbol{Y}}_n^\top(\boldsymbol{K}_n + \eta\boldsymbol{I}_n)^{-1}\tilde{\boldsymbol{Y}}_n$$

and solving this we have the estimator for τ^2:

$$\hat{\tau}^2 = \frac{1}{n}\tilde{\boldsymbol{Y}}_n^\top(\boldsymbol{K}_n + \eta\boldsymbol{I}_n)^{-1}\tilde{\boldsymbol{Y}}_n. \tag{2.1.3}$$

Substituting (2.1.3) back into (2.1.2), we obtain a profiled log-likelihood function dependent solely on η and length-scales:

$$\begin{aligned}
\ell(\boldsymbol{\omega}, \eta) &= -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log\left(\frac{1}{n}\tilde{\boldsymbol{Y}}_n^\top(\boldsymbol{K}_n + \eta\boldsymbol{I}_n)^{-1}\tilde{\boldsymbol{Y}}_n\right) \\
&\quad - \frac{1}{2}\log\det(\boldsymbol{K}_n + \eta\boldsymbol{I}_n) - \frac{\tilde{\boldsymbol{Y}}_n^\top(\boldsymbol{K}_n + \eta\boldsymbol{I}_n)^{-1}\tilde{\boldsymbol{Y}}_n}{2\left(\frac{1}{n}\tilde{\boldsymbol{Y}}_n^\top(\boldsymbol{K}_n + \eta\boldsymbol{I}_n)^{-1}\tilde{\boldsymbol{Y}}_n\right)} \\
&= -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log\left(\frac{1}{n}\right) - \frac{n}{2}\log\left(\tilde{\boldsymbol{Y}}_n^\top(\boldsymbol{K}_n + \eta\boldsymbol{I}_n)^{-1}\tilde{\boldsymbol{Y}}_n\right) \\
&\quad - \frac{1}{2}\log\det(\boldsymbol{K}_n + \eta\boldsymbol{I}_n) - \frac{n}{2}
\end{aligned}$$

$$\ell(\boldsymbol{\omega}, \eta) = -\frac{n}{2}\log\left(\tilde{\boldsymbol{Y}}_n^\top(\boldsymbol{K}_n + \eta\boldsymbol{I}_n)^{-1}\tilde{\boldsymbol{Y}}_n\right) - \frac{1}{2}\log\det(\boldsymbol{K}_n + \eta\boldsymbol{I}_n) + C, \tag{2.1.4}$$

where C is a constant independent of η.

The function $\ell(\boldsymbol{\omega}, \eta)$ encapsulates a trade-off between data fit and model complexity. Minimizing this profiled objective allows for consistent estimation of the nugget parameter even in the absence of replication. In the implementation that follows, the minimization of (2.1.4) is carried out using PyTorch's L-BFGS optimizer

[13], as discussed in next two chapters.

2.1.3 Curse of Dimensionality in Gaussian Process Regression

As introduced in Chapter 1, the curse of dimensionality broadly refers to the challenges that arise as the dimensionality of input data increases [14]. In the context of GPR, these challenges manifest in both computational burden and degraded predictive performance.

From a computational perspective, GPR requires inversion of an $N \times N$ kernel matrix $\boldsymbol{K}_n$, with time complexity $\mathcal{O}(N^3)$ via Cholesky decomposition. Although this cost depends directly on the sample size N, higher-dimensional spaces require exponentially more data to maintain predictive accuracy, exacerbating the computational load [8]. Approximation techniques such as sparse GPs and variational methods [15, 16, 17] alleviate the cost to some extent, yet remain sensitive to dimensional scaling.

Model accuracy also suffers. In high dimensions, pairwise distances become increasingly similar [18], making kernels like the Gaussian kernel less discriminative. The kernel value approaches a constant for most point pairs, leading to oversmoothing and loss of expressivity. As a result, the model struggles to capture local variations or high-frequency components, and may generalize poorly beyond the training data.

Moreover, the training data tends to occupy a vanishingly small portion of the input space, making the model overly sensitive to noise and reducing robustness. These limitations motivate the use of structure-aware priors and low-dimensional embeddings explored in later chapters.

These issues underscore the necessity of incorporating inductive biases that

exploit low-dimensional structure, a theme we pursue through variational inference and manifold learning in subsequent chapters.

2.2 Manifold Learning

As discussed above, the curse of dimensionality undermines both the computational tractability and predictive reliability of Gaussian process regression in high-dimensional spaces. A natural remedy is to seek a lower-dimensional representation of the data before applying GPR. Manifold learning—a class of non-linear dimensionality reduction techniques—is especially suitable when the data are believed to lie near a lower-dimensional manifold embedded in the ambient space [19]. In this section, we review general manifold learning principles, examine Manifold Gaussian Process regression and its related work, and finally present our variant of mGP tailored to active learning.

2.2.1 Principles of Manifold Learning

The manifold hypothesis posits that high-dimensional data often concentrate around a lower-dimensional manifold. Techniques such as Isomap, LLE, and Laplacian Eigenmaps construct neighborhood graphs to preserve local relationships in a lower-dimensional embedding [20, 21, 22]. Such embeddings "unroll" nonlinear structure—e.g., a Swiss roll—enabling subsequent learning stages to focus on meaningful variation within data [19]. Unlike PCA, which captures only global linear structure, manifold methods preserve local geometric relationships, a property essential for complex-feature modeling in GPR.

2.2.2 Manifold Gaussian Process Regression

Manifold Gaussian Process (mGP) extends standard GPR by learning an ex-

plicit mapping

$$M : \mathcal{X} \subset \mathbb{R}^D \to \mathbb{R}^Q$$

such that the GP operates in this latent manifold space [23]. Specifically,

$$\boldsymbol{Y}_n \sim \mathcal{N}\big(\boldsymbol{G}\boldsymbol{\beta},\, \tau^2(\tilde{\boldsymbol{K}}_n + \eta \boldsymbol{I}_n)\big),$$

where $\tilde{K}_{n,ij} = k(M(\boldsymbol{x}_i), M(\boldsymbol{x}_j))$. Mapping M can be parameterized via a neural net with parameters θ_M, optimized jointly with GP hyperparameters [23].

[24] has generalized GP models to Riemannian domains via wrapped GPs. More recent efforts—such as implicit mGP—learn manifold structure end-to-end through variational schemes [25]. These methods maintain uncertainty estimates while adapting GP priors to latent geometry. However, existing mGP approaches lack principled strategies for selecting data points under limited budgets. Next section introduces an active learning strategy to accomplish this.

2.3 Active Learning Strategies

Active learning (AL) refers to a family of data-efficient sampling strategies that aim to optimize model performance by selectively querying the most informative data points. Rather than relying on uniform or random sampling, AL methods iteratively update the model and strategically acquire new inputs, improving predictive accuracy under limited evaluation budgets [26]. This is particularly crucial in surrogate modeling scenarios such as Gaussian Process Regression (GPR), where evaluating the ground truth model often requires expensive simulations or experiments.

In the context of GPR, active learning is important for reducing the number of labeled data points needed to achieve a target level of accuracy. The core idea is to identify where the model is most uncertain or where adding a new observa-

tion would most improve global performance. Several acquisition criteria have been proposed, including predictive variance, expected improvement, and information-theoretic metrics. Among these, strategies that directly minimize integrated predictive uncertainty—such as the Integrated Mean Squared Prediction Error (IMSPE)—are particularly well-suited for regression tasks.

2.3.1 The Active Learning Cohn (ALC) Criterion

The Active Learning Cohn (ALC) criterion [26, 27] is a foundational information theoretic strategy for sequential experimental design in Gaussian Process Regression (GPR). It selects new data points $\boldsymbol{x}_{n+1}$ that minimizes the model's integrated mean-squared error (IMSE) over the input space. Formally, given current observations, the ALC score at a candidate point $\boldsymbol{x}_{n+1}$ is defined as:

$$\mathrm{ALC}(\boldsymbol{x}_{n+1}) = \int_{\mathcal{X}} \sigma_n^2(\boldsymbol{x}) - \sigma_{n+1}^2(\boldsymbol{x}|\boldsymbol{x}_{n+1})\, d\boldsymbol{x},$$

where $\sigma_n^2(\boldsymbol{x})$ and $\sigma_{n+1}^2(\boldsymbol{x}|\boldsymbol{x}_{n+1})$ denote the predictive variances before and after including $\boldsymbol{x}_{n+1}$. In practice, this integral is approximated using a fixed reference set $\mathcal{X}_{\mathrm{ref}}$. The full formulation and implementation details are deferred to Chapter 4.

This formulation draws conceptual parallels to Bayesian Optimization (BO), where acquisition functions such as Expected Improvement (EI) or Upper Confidence Bound [28] are used to select informative points. Both ALC and BO rely on model-based acquisition functions to guide data selection. However, their objectives differ: BO seeks to find the global maximum of an unknown function, often focusing on exploitation and exploration trade-offs, while ALC is tailored for reducing global predictive error in regression problems.

ALC has been shown to be effective in reducing prediction error in surrogate modeling settings, but its computational cost—involving integration over the input

space—can be prohibitive in high dimensions. To address this, we incorporates a variance-based screening step to reduce the number of full ALC evaluations, detailed in Chapter 4.

CHAPTER 3

EVI-GP FRAMEWORK

3.1 GP under Bayesian Framework

[2]We assume the following prior distributions for the parameters $\boldsymbol{\theta} = (\boldsymbol{\beta}, \boldsymbol{\omega}, \tau^2, \eta)$,

$$\begin{aligned} &\boldsymbol{\beta} \sim MVN_p(\mathbf{0}, \nu^2 \boldsymbol{R}), \qquad \omega_i \overset{\text{i.i.d.}}{\sim} \text{Gamma}(a_\omega, b_\omega), \text{ for } i = 1, \ldots, d \\ &\tau^2 \sim \text{Inverse-}\chi^2(df_{\tau^2}), \qquad \eta \sim \text{Gamma}(a_\eta, b_\eta). \end{aligned} \tag{3.1.1}$$

These distribution families are commonly used in the literature such as [29] and [30].

However, the choice of parameters of the prior distributions may require fine-tuning using testing data or cross-validation procedures. In some literature, parameters $\boldsymbol{\omega}$, τ^2, and η are considered to be hyperparameters. The conditional posterior distribution of $\boldsymbol{\beta}$ given data and $(\boldsymbol{\omega}, \tau^2, \eta)$ is a multivariate normal distribution, which is shown later.

Next, we derive the posterior distributions and some conditional posterior distributions. Based on the data, the sampling distribution is

$$\boldsymbol{y}_n | \boldsymbol{\theta} \sim MVN_n\left(\boldsymbol{G}\boldsymbol{\beta}, \tau^2(\boldsymbol{K}_n + \eta \boldsymbol{I}_n)\right),$$

where $\boldsymbol{y}_n$ is the vector of y_i's and $\boldsymbol{G}$ is a matrix of row vectors $\boldsymbol{g}(\boldsymbol{x}_i)^\top$'s. The matrix $\boldsymbol{K}_n$ is the $n \times n$ kernel matrix with entries $K_n[i, j] = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and is a symmetric

[2]This chapter is adapted from [1]. See authorship statement for more details.

and positive definite matrix, and $\boldsymbol{I}_n$ is an $n \times n$ identity matrix. The probability density function of $\boldsymbol{y}_n|\boldsymbol{\theta}$ is

$$p(\boldsymbol{y}_n|\boldsymbol{\theta}) \propto (\tau^2)^{-\frac{n}{2}} \det(\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1/2} \exp\left(-\frac{1}{2\tau^2}(\boldsymbol{y}_n - \boldsymbol{G}\boldsymbol{\beta})^\top (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1}(\boldsymbol{y}_n - \boldsymbol{G}\boldsymbol{\beta})\right).$$

Following Bayes' Theorem, the joint posterior distribution of all parameters is

$$p(\boldsymbol{\theta}|\boldsymbol{y}_n) \propto p(\boldsymbol{\theta})p(\boldsymbol{y}_n|\boldsymbol{\theta}) \propto p(\boldsymbol{\beta}) \left(\prod_{j=1}^{d} p(\omega_i)\right) p(\tau^2)p(\eta)p(\boldsymbol{y}_n|\boldsymbol{\theta}).$$

The conditional posterior distribution of $\boldsymbol{\beta}$ can be easily obtained through conjugacy. It is also straightforward to obtain the posterior distribution $p(\boldsymbol{\omega}, \tau^2, \eta|\boldsymbol{y}_n)$. The results are summarized in Proposition 3.1.1. The posterior distribution of τ^2 can be significantly simplified if a non-informative prior is used for $\boldsymbol{\beta}$, as described in Proposition 3.1.2. All the proofs can be found in the appendix.

Proposition 3.1.1. *Using the prior distribution of $\boldsymbol{\theta} = (\boldsymbol{\beta}, \boldsymbol{\omega}, \tau^2, \eta)$ in (3.1.1), the conditional posterior distribution of $\boldsymbol{\beta}$ is $\boldsymbol{\beta}|\boldsymbol{y}_n, \boldsymbol{\omega}, \tau^2, \eta \sim MVN_p\left(\hat{\boldsymbol{\beta}}_n, \boldsymbol{\Sigma}_{\boldsymbol{\beta}|n}\right)$, where*

$$\boldsymbol{\Sigma}_{\boldsymbol{\beta}|n} = \left[\frac{1}{\tau^2}\boldsymbol{G}^\top(\boldsymbol{K}_n + \eta\boldsymbol{I}_n)^{-1}\boldsymbol{G} + \frac{1}{\nu^2}\boldsymbol{R}^{-1}\right]^{-1}, \quad \hat{\boldsymbol{\beta}}_n = \tau^{-2}\boldsymbol{\Sigma}_{\boldsymbol{\beta}|n}\left[\boldsymbol{G}^\top(\boldsymbol{K}_n + \eta\boldsymbol{I}_n)^{-1}\right]\boldsymbol{y}_n.$$

The marginal posterior distribution of $(\boldsymbol{\omega}, \tau^2, \eta)$ is

$$p(\boldsymbol{\omega}, \tau^2, \eta|\boldsymbol{y}_n) \propto \det(\boldsymbol{\Sigma}_{\boldsymbol{\beta}|n})^{1/2} \exp\left[-\frac{1}{2}\hat{\boldsymbol{\beta}}_n^\top \boldsymbol{\Sigma}_{\boldsymbol{\beta}|n}^{-1}\hat{\boldsymbol{\beta}}_n - \frac{1}{2\tau^2}\boldsymbol{y}_n^\top(\boldsymbol{K}_n + \eta\boldsymbol{I}_n)^{-1}\boldsymbol{y}_n\right] \times (\tau^2)^{-n/2}\det(\boldsymbol{K}_n + \eta\boldsymbol{I}_n)^{-1/2}p(\tau^2)p(\boldsymbol{\omega})p(\eta). \tag{3.1.2}$$

Proposition 3.1.2. *If using a non-informative prior distribution for $\boldsymbol{\beta}$, i.e., $p(\boldsymbol{\beta}) \propto 1$, and the same prior distributions for $(\boldsymbol{\omega}, \tau^2, \eta)$ in (3.1.1), the conditional posterior*

distribution of $\boldsymbol{\beta}$ *is still* $MVN_p\left(\hat{\boldsymbol{\beta}}_n, \boldsymbol{\Sigma}_{\boldsymbol{\beta}|n}\right)$, *but the covariance and mean are*

$$\boldsymbol{\Sigma}_{\boldsymbol{\beta}|n} = \tau^2 \left[\boldsymbol{G}^\top (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1} \boldsymbol{G}\right]^{-1},$$
$$\hat{\boldsymbol{\beta}}_n = \left[\boldsymbol{G}^\top (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1} \boldsymbol{G}\right]^{-1} \left[\boldsymbol{G}^\top (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1}\right] \boldsymbol{y}_n.$$

The conditional posterior distribution for τ^2 *is*

$$\tau^2 | \boldsymbol{\omega}, \eta, \boldsymbol{y}_n \sim \textit{Scaled Inverse-}\chi^2(df_{\tau^2} + n - p, \hat{\tau}^2),$$

where $\hat{\tau}^2 = (1 + s_n^2)/\left(df_{\tau^2} + n - p\right)$, $s_n^2 = \tau^{-2} \hat{\boldsymbol{\beta}}_n^\top \boldsymbol{\Sigma}_{\boldsymbol{\beta}|n}^{-1} \hat{\boldsymbol{\beta}}_n + \boldsymbol{y}_n^\top (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1} \boldsymbol{y}_n$. *The marginal posterior of* $(\boldsymbol{\omega}, \eta)$ *is*

$$p(\boldsymbol{\omega}, \eta | \boldsymbol{y}_n) \propto (\hat{\tau}^2)^{-\frac{1}{2}(df_{\tau^2} + n - p)} \det(\boldsymbol{G}^\top (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1} \boldsymbol{G})^{-1/2} \det(\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1/2} p(\boldsymbol{\omega}) p(\eta). \tag{3.1.3}$$

If we use non-informative prior distributions for all the parameters, i.e., $p(\boldsymbol{\beta}) \propto 1$ and $p(\omega_i) \overset{\text{i.i.d.}}{\sim} \text{Uniform}[a_\omega, b_\omega]$ for $i = 1, \ldots, D$, $p(\tau^2) \propto \tau^{-2}$, and $p(\eta) \propto \text{Uniform}[a_\eta, b_\eta]$, the Bayesian framework is equivalent to the empirical Bayesian or maximum likelihood estimation method. The GP regression model estimated via this frequentist approach is common in both methodology research and application [9, 10]. Here we consider the empirical Bayesian estimation as a special case of the Bayesian GP model. The choice between the two different types of prior distributions for $\boldsymbol{\beta}$, informative or non-informative, is subject to the dimension of the input variables, the assumption on basis functions $\boldsymbol{g}(\boldsymbol{x})$, the goal of GP modeling (accurate prediction v.s. interpretation), and sometimes the application of the computer experiment. Both types have their unique advantages and shortcomings. The non-informative prior distribution for $\boldsymbol{\beta}$ reduces the computation involved in the posterior sampling for τ^2, but we would lose the l_2 regularization effect on $\boldsymbol{\beta}$ brought by the informative

prior $\boldsymbol{\beta}$.

One issue with the informative prior distribution is to choose its parameters, i.e., the constant variance ν^2 and the correlation matrix $\boldsymbol{R}$. Here we recommend using a cross-validation procedure to select ν^2. If the mean function $\boldsymbol{g}(\boldsymbol{x})^\top\boldsymbol{\beta}$ is a polynomial function of the input variables, we specify the matrix $\boldsymbol{R}$ to be a diagonal matrix $\boldsymbol{R} = \text{diag}\{1, r, \ldots, r, r^2, \ldots, r^2, \ldots\}$, where $r \in (0, 1)$ is a user-specified parameter. The power index of r is the same as the order of the corresponding polynomial term. For example, if $\boldsymbol{g}(\boldsymbol{x})^\top\boldsymbol{\beta}$ with $\boldsymbol{x} \in \mathbb{R}^2$ is a full quadratic model and contains the terms $\boldsymbol{g}(\boldsymbol{x}) = [1, x_1, x_2, x_1^2, x_2^2, x_1x_2]^\top$, the corresponding prior correlation matrix should be specified as $\boldsymbol{R} = \text{diag}\{1, r, r, r^2, r^2, r^2\}$. In this way, the prior variance of the effect decreases exponentially as the order of effect increases, following the effect hierarchy principle [31, 32]. It states that lower-order effects are more important than higher-order effects, and the effects of the same order are equally important. The hierarchy ordering principle can reduce the size of the model and avoid including higher-order and less significant model terms. Such prior distribution was firstly proposed by [33], and later used in [34, 35, 36, 37, 38, 39].

Proposition 3.1.3. *Given the parameters $(\boldsymbol{\omega}, \tau^2, \eta)$, the posterior predictive distribution of $y(\boldsymbol{x})$ at any query point $\boldsymbol{x}$ is the following normal distribution.*

$$y(\boldsymbol{x})|\boldsymbol{y}_n, \boldsymbol{\omega}, \tau^2, \eta \sim N(\hat{\mu}(\boldsymbol{x}), \sigma_n^2(\boldsymbol{x})),$$

where

$$\begin{aligned}
\hat{\mu}(\boldsymbol{x}) &= \boldsymbol{g}(\boldsymbol{x})^\top\hat{\boldsymbol{\beta}}_n + K(\boldsymbol{x}, \mathcal{X}_n)(\boldsymbol{K}_n + \eta\boldsymbol{I}_n)^{-1}(\boldsymbol{y}_n - \boldsymbol{G}\hat{\boldsymbol{\beta}}_n), \\
\sigma_n^2(\boldsymbol{x}) &= \tau^2\Big\{1 - K(\boldsymbol{x}, \mathcal{X}_n)(\boldsymbol{K}_n + \eta\boldsymbol{I}_n)^{-1}K(\mathcal{X}_n, \boldsymbol{x}) + \\
&\qquad \boldsymbol{c}(\boldsymbol{x})^\top\left[\boldsymbol{G}^\top(\boldsymbol{K}_n + \eta\boldsymbol{I}_n)^{-1}\boldsymbol{G}\right]^{-1}\boldsymbol{c}(\boldsymbol{x})\Big\},
\end{aligned}$$

where $\boldsymbol{c}(\boldsymbol{x}) = \boldsymbol{g}(\boldsymbol{x}) - \boldsymbol{G}^\top(\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1} K(\mathcal{X}_n, \boldsymbol{x})$ *and* $K(\boldsymbol{x}, \mathcal{X}_n) = K(\mathcal{X}_n, \boldsymbol{x})^\top = [K(\boldsymbol{x}, \boldsymbol{x}_1), \ldots, K(\boldsymbol{x}, \boldsymbol{x}_n)]$. *For non-informative prior, the posterior predictive distribution* $y(\boldsymbol{x})|\boldsymbol{y}_n, \boldsymbol{\omega}, \eta$ *is the same except that* τ^2 *is replaced by* $\hat{\tau}^2$.

The proof is similar to the proof of proposition 2.1.1. Thanks to the Gaussian process assumption and the conditional conjugate prior distributions for $\boldsymbol{\beta}$ and τ^2, Proposition 3.1.1, 3.1.2, and 3.1.3 give the explicit and easy to generate conditional posterior distribution of $\boldsymbol{\beta}$ (and τ^2) and posterior predictive distribution. Therefore, how to generate samples from $p(\boldsymbol{\omega}, \tau^2, \eta|\boldsymbol{y}_n)$ in (3.1.2) or $p(\boldsymbol{\omega}, \eta|\boldsymbol{y}_n)$ in (3.1.3) is the bottleneck of the computation for GP models. Since $p(\boldsymbol{\omega}, \tau^2, \eta|\boldsymbol{y}_n)$ and $p(\boldsymbol{\omega}, \eta|\boldsymbol{y}_n)$ are not from any known distribution families, Metropolis-Hastings (MH) algorithm [40, 41], Hamiltonian Monte Carlo (HMC) [42], or Metropolis-adjusted Langevin algorithm (MALA) can be used for sampling [43]. Here we introduce an alternative computational tool, namely, a variational inference approach to approximate the posterior distribution $p(\boldsymbol{\omega}, \tau^2, \eta|\boldsymbol{y}_n)$ or $p(\boldsymbol{\omega}, \eta|\boldsymbol{y}_n)$. More specifically, we plan to use energetic variational inference, a particle method, to generate posterior samples.

3.2 Energetic Variational Inference Gaussian Process

Variational inference-based GP models have been explored in prior works such as [16], [44], and [45]. Despite sharing the variational inference idea, the proposed Energetic Variational Inference (EVI) GP differs significantly from these existing methods regarding the specific variational techniques employed. [16] utilized auto-encoding, while [44] and [45] employed the mean-field variational inference [6].

The EVI approach offers simplicity in implementation across diverse applications without the need for training any neural networks. Notably, EVI establishes a connection between the Maximum A Posteriori (MAP) procedure and posterior sampling through a user-specified number of particles. In contrast to the complexity

of auto-encoding variational methods, the particle-based approach is much simpler, devoid of any neural network intricacies. Moreover, in comparison to mean-field methods, both particle-based and MAP-based approaches (auto-encoding falls into the MAP-based category) can exhibit enhanced accuracy, as they do not impose any parametric assumptions on a feasible family of distributions in the optimization to solve the variational problem.

3.2.1 Energetic Variational Inference

We first introduce the EVI using the continuous formulation. Let $\boldsymbol{\phi}_t$ be the dynamic flow map $\boldsymbol{\phi}_t : \mathbb{R}^d \to \mathbb{R}^d$ at time t [46] that continuously transforms the d-dimensional distribution from an initial distribution toward the target one and we require the map $\boldsymbol{\phi}_t$ to be smooth and one-to-one. The functional $\mathcal{F}(\boldsymbol{\phi}_t)$ is a user-specified divergence or other machine learning objective functional. Taking the analogy of a thermodynamics system, $\mathcal{F}(\boldsymbol{\phi}_t)$ is the Helmholtz free energy. Following the First and Second Law of thermodynamics [47] and set the kinetic energy to zero, we have

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathcal{F}(\boldsymbol{\phi}_t) = -\triangle(\boldsymbol{\phi}_t, \dot{\boldsymbol{\phi}}_t), \tag{3.2.1}$$

where $\triangle(\boldsymbol{\phi}_t, \dot{\boldsymbol{\phi}}_t)$ is a user-specified functional representing the rate of energy dissipation, and $\dot{\boldsymbol{\phi}}_t$ is the derivative of $\boldsymbol{\phi}_t$ with time t. So $\dot{\boldsymbol{\phi}}_t$ can be interpreted as the "velocity" of the transformation. Each variational formulation gives a natural path of decreasing the objective functional $\mathcal{F}(\boldsymbol{\phi}_t)$ toward an equilibrium. The dissipation functional should satisfy $\triangle(\boldsymbol{\phi}_t, \dot{\boldsymbol{\phi}}_t) \geq 0$ so that $\mathcal{F}(\boldsymbol{\phi}_t)$ decreases with time. A simple yet effective specification of $\triangle(\boldsymbol{\phi}_t, \dot{\boldsymbol{\phi}}_t)$ is a quadratic functional [48] in terms of $\dot{\boldsymbol{\phi}}_t$,

$$\triangle(\boldsymbol{\phi}_t, \dot{\boldsymbol{\phi}}_t) = \int_{\Omega_t} \rho_{[\boldsymbol{\phi}_t]} \|\dot{\boldsymbol{\phi}}_t\|_2^2 \, \mathrm{d}\boldsymbol{x},$$

and it has the variation (functional derivative [49])

$$\frac{\delta \triangle(\boldsymbol{\phi}_t, \dot{\boldsymbol{\phi}}_t)}{\delta \dot{\boldsymbol{\phi}}_t} = 2\rho_{[\boldsymbol{\phi}_t]}\dot{\boldsymbol{\phi}}_t,$$

where $\rho_{[\boldsymbol{\phi}_t]}$ denotes the pdf of the current distribution which is the initial distribution transformed by $\boldsymbol{\phi}_t$, Ω_t is the current support, $\|\boldsymbol{a}\|_2 = \boldsymbol{a}^\top \boldsymbol{a}$ for $\forall \boldsymbol{a} \in \mathbb{R}^d$, and δ is the variation operator.

With the specified energy-dissipation law (3.2.1), the energy variational approach derives the dynamics of the systems through two variational procedures, the Least Action Principle (LAP) [50] and the Maximum Dissipation Principle (MDP) [51], which leads to

$$\frac{\delta \frac{1}{2}\triangle}{\delta \dot{\boldsymbol{\phi}}_t} = -\frac{\delta \mathcal{F}}{\delta \boldsymbol{\phi}_t}, \quad \text{and} \quad \rho_{[\boldsymbol{\phi}_t]}\dot{\boldsymbol{\phi}}_t = -\frac{\delta \mathcal{F}}{\delta \boldsymbol{\phi}_t},$$

using the quadratic $\Delta(\boldsymbol{\phi}_t, \dot{\boldsymbol{\phi}}_t)$.

In general, this continuous formulation (3.2.1) is difficult to solve since the manifold of $\boldsymbol{\phi}_t$ is of infinite dimension. Naturally, there are different approaches to approximate an infinite-dimensional manifold by a finite-dimensional manifold. A classical example is the JKO scheme [52], where infinite-dimensional diffusion dynamics are approximated by discrete-time minimization over finite-dimensional Wasserstein space[53]. Another approach, as used in [54], is to use particles (or samples) to approximate the $\rho_{[\boldsymbol{\phi}_t]}$ in (3.2.1) with kernel regularization, before any variational steps. It leads to a discrete version of the energy-dissipation law, i.e.,

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathcal{F}_h(\{\boldsymbol{p}_i(t)\}_{i=1}^N) = -\triangle_h(\{\boldsymbol{p}_i(t)\}_{i=1}^N, \{\dot{\boldsymbol{p}}_i(t)\}_{i=1}^N). \tag{3.2.2}$$

Here $\{\boldsymbol{p}(t)\}_{i=1}^N$ is the locations of N particles at time t and $\dot{\boldsymbol{p}}_i(t)$ is the derivative of

$\boldsymbol{p}_i$ with t, and thus is the velocity of the ith particle as it moves toward the target distribution. The subscript h of $\mathcal{F}$ and $\triangle$ denotes the bandwidth parameter of the kernel function used in the kernelization operation. Applying the variational steps to (3.2.2), we obtain the dynamics of decreasing $\mathcal{F}$ at the particle level,

$$\frac{\delta \frac{1}{2} \triangle_h}{\delta \dot{\boldsymbol{p}}_i(t)} = -\frac{\delta \mathcal{F}_h}{\delta \boldsymbol{p}_i}, \quad \text{for } i = 1, \ldots, N. \tag{3.2.3}$$

This leads to an ODE system of $\{\boldsymbol{p}_i(t)\}_{i=1}^N$ and can be solved by different numerical schemes, such as first order explicit and implicit Euler approaches shown in [54]. The solution is the particles approximating the target distribution.

For the energy functional, we use the KL-divergence as demonstrated in [54],

$$D_{\mathrm{KL}}(\rho||\rho^*) = \int_\Omega \rho(\boldsymbol{x}) \log\left(\frac{\rho(\boldsymbol{x})}{\rho^*(\boldsymbol{x})}\right) \mathrm{d}\boldsymbol{x},$$

where $\rho^*(\boldsymbol{x})$ is the density function of the target distribution with support region Ω and $\rho(\boldsymbol{x})$ is to approximate $\rho^*(\boldsymbol{x})$. For EVI-GP, ρ^* is the posterior distribution of $p(\boldsymbol{\omega}, \tau^2, \eta|\boldsymbol{y}_n)$ or $p(\boldsymbol{\omega}, \eta|\boldsymbol{y}_n)$.

Using the KL-divergence, the divergence functional at time t is

$$\mathcal{F}(\boldsymbol{\phi}_t) = \int \left(\rho_{[\boldsymbol{\phi}_t]}(\boldsymbol{x}) \log \rho_{[\boldsymbol{\phi}_t]}(\boldsymbol{x}) + \rho_{[\boldsymbol{\phi}_t]}(\boldsymbol{x}) V(\boldsymbol{x})\right) \mathrm{d}\boldsymbol{x},$$

where $V(\boldsymbol{x}) = \log \rho^*(\boldsymbol{x})$, which is known up to a scaling constant. The discrete version of the energy becomes

$$\mathcal{F}_h\left(\{\boldsymbol{p}_i\}_{i=1}^N\right) = \frac{1}{N} \sum_{i=1}^N \left(\ln\left[\frac{1}{N} \sum_{j=1}^N K_h(\boldsymbol{p}_i, \boldsymbol{p}_j)\right] + V(\boldsymbol{p}_i)\right),$$

where $K_h(\boldsymbol{x}_i, \boldsymbol{x}_j) \propto \exp(-\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2 / h^2)$, h is kernel bandwidth and the discrete

dissipation is

$$-2\Delta_h\left(\{\boldsymbol{p}_i\}_{i=1}^N\right) = -\frac{1}{N}\sum_{i=1}^N |\dot{\boldsymbol{p}}_i(t))|^2.$$

Applying variational step to (3.2.2), we obtain (3.2.3) which is equivalent to the following nonlinear ODE system:

$$\dot{\boldsymbol{p}}_i(t) = -\left(\frac{\sum_{j=1}^N \nabla_{\boldsymbol{p}_i} K_h(\boldsymbol{p}_i,\boldsymbol{p}_j)}{\sum_{j=1}^N K_h(\boldsymbol{p}_i,\boldsymbol{p}_j)} + \sum_{k=1}^N \frac{\nabla_{\boldsymbol{p}_i} K_h(\boldsymbol{p}_k,\boldsymbol{p}_i)}{\sum_{j=1}^N K_h(\boldsymbol{p}_k,\boldsymbol{p}_j)} + \nabla_{\boldsymbol{p}_i} V(\boldsymbol{p}_i)\right), \tag{3.2.4}$$

for $i = 1,\ldots,N$. The iterative update of N particles collection $\{\boldsymbol{p}_i\}_{i=1}^N$ involves solving the nonlinear ordinary differential equations (ODEs), specified in (3.2.4), using implicit Euler method [55] for enhanced numerical stability. To begin, we discretize the temporal domain via the implicit Euler scheme

$$\frac{1}{N}\frac{\boldsymbol{p}_i^{n+1}-\boldsymbol{p}_i^n}{T} = -\frac{\delta \mathcal{F}_h}{\delta \boldsymbol{p}_i}\left(\{\boldsymbol{p}_i^{n+1}\}_{i=1}^N\right), \tag{3.2.5}$$

where $T>0$ is the time step size. The discrete equation (3.2.5) constitutes a system of coupled nonlinear equations. Rather than solving it directly, one may equivalently recast this as a finite-dimensional optimization problem by defining the following objective functional

$$J_m(\{\boldsymbol{p}_i\}_{i=1}^N) := \frac{1}{2T}\sum_{i=1}^N ||\boldsymbol{p}_i - \boldsymbol{p}_i^m||^2/N + \mathcal{F}_h(\{\boldsymbol{p}_i\}_{i=1}^N),$$

where $\boldsymbol{p}_i^m$ denotes the current particle state at iteration m. The gradient of J_m with respect to $\{\boldsymbol{p}_i\}_{i=1}^N$ recovers the right-hand side of the implicit Euler equation (3.2.5). Hence, the particle configuration at the next iteration is obtained by solving the following proximal minimization problem:

$$\{\boldsymbol{p}_i^{m+1}\}_{i=1}^N = \mathrm{argmin}_{\{\boldsymbol{p}_i\}_{i=1}^N} J_m(\{\boldsymbol{p}_i\}_{i=1}^N), \tag{3.2.6}$$

This optimization-based interpretation corresponds to the well-known Proximal Point Algorithm (PPA) [56], which is particularly advantageous in dealing with stiff or non-smooth objective landscapes. In our implementation, we employ the L-BFGS [57] algorithm to solve (3.2.6) efficiently within each iteration.

[54] emphasized the advantages of using the Implicit-Euler solver for enhanced numerical stability in this process with a detailed proof. We summarize the algorithm of using the implicit Euler scheme to solve the ODE system (3.2.4) into Algorithm 1.

Algorithm 1 EVI with Implicit Euler Scheme (EVI-Im).

Input: The target distribution $\rho^*(\boldsymbol{x})$ and a set of initial particles $\{\boldsymbol{p}_i^0\}_{i=1}^N$ drawn from a prior $\rho_0(\boldsymbol{x})$.
Output: A set of particles $\{\boldsymbol{p}_i^*\}_{i=1}^N$ approximating ρ^*.
for $m = 0$ **to** MaxIter **do**
  Solve $\{\boldsymbol{p}_i^{m+1}\}_{i=1}^N = \text{argmin}_{\{\boldsymbol{p}_i\}_{i=1}^N} J_m(\{\boldsymbol{p}_i\}_{i=1}^N)$.
  Update $\{\boldsymbol{p}_i^m\}_{i=1}^N$ by $\{\boldsymbol{p}_i^{m+1}\}_{i=1}^N$.
end for

3.2.2 EVI-GP

We propose two adaptations of the EVI-Im algorithm for GP model estimation and prediction. The first approach, named EVI-post for short, involves generating N particles using Algorithm 1 to approximate the posterior $p(\boldsymbol{\omega}, \tau^2, \eta|\boldsymbol{y}_n)$ when an informative normal prior distribution is adopted for $\boldsymbol{\beta}$, or $p(\boldsymbol{\omega}, \eta|\boldsymbol{y}_n)$ when a non-informative prior distribution is used for $\boldsymbol{\beta}$. These N particles serve as posterior samples. Conditional on their values, we can generate samples for $\boldsymbol{\beta}$ based on its conditional posterior distribution (Proposition 3.1.1) or generate samples for both $\boldsymbol{\beta}$ and τ^2 according to their conditional posterior distribution (Proposition 3.1.2). Following Proposition 3.1.3, we can generally predict $y(\boldsymbol{x})$ and confidence intervals conditional on the posterior samples.

In the second approach, we employ EVI-Im solely as an optimization tool for Maximum A Posteriori (MAP), entailing the minimization of $V(\boldsymbol{p}) = -\log \rho^*(\boldsymbol{p})$. So we call it EVI-MAP for short. This can be done by simply setting the free energy as $\mathcal{F}(\boldsymbol{p}) = V(\boldsymbol{p})$ and $N = 1$. As a result, the optimization problem (3.2.6) at the mth iteration becomes

$$\boldsymbol{p}^{m+1} = \arg\min_{\boldsymbol{p}} \frac{1}{2T}||\boldsymbol{p} - \boldsymbol{p}^m||^2 - \log V(\boldsymbol{p}),$$

Therefore, EVI is a method that connects the posterior sampling and MAP under the same general framework. Based on the MAP, we can obtain the posterior mode for $\boldsymbol{\beta}$ (or $\boldsymbol{\beta}$ and τ^2) and the mode of the prediction and the corresponding inference based on the posterior mode.

3.3 Numerical Examples

In this section, we demonstrate the performances of EVI-GP and compare it with three commonly used GP packages in `R`, which are `gpfit` [58], `mlegp` [59], `laGP` [60, 61].

The first one is a 1-dim toy example and the second one is the Borehole example chosen from the online library built by [62]. The codes for EVI-GP and all the examples are available on GitHub with the link https://github.com/XavierOwen/EVIGP. The EVI-GP is implemented in `Python`. The proximal point optimization in Algorithm 1 is solved by the L-BFGS function of `Pytorch` library [63]. Some arguments of the EVI-GP codes are set the same for all examples, which are explained in [1].

These parameter settings are done through many trials and they lead to satisfactory performance in most examples we have studied. In each example, we use the same pair of training and testing datasets for all the methods under comparison. The designs for both datasets are generated via `maximinLHS` procedure from the `lhs`

package in R [64]. We run 100 simulations. In each simulation, we compute the standardized Root Mean Square Prediction Error (RMSPE) on the test data set, which is defined as follows

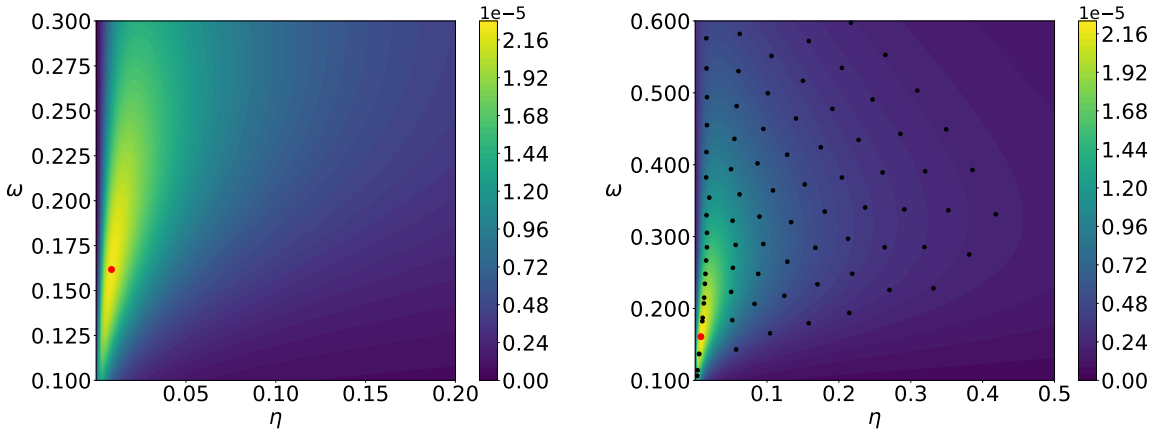
$$\text{standardized RMSPE} = \left(\sqrt{\frac{1}{n_{\text{test}}}\sum_i (\hat{y}_i - y_i)^2}\right) \Big/ \text{standard deviation of test}(\boldsymbol{y}),$$

where $\hat{y}_i$ is the predicted value at the test point $\boldsymbol{x}_i$ and y_i is the corresponding true value. Box plots of the 100 standardized RMSPEs of all methods are shown for comparison.

3.3.1 One-Dim Toy Example

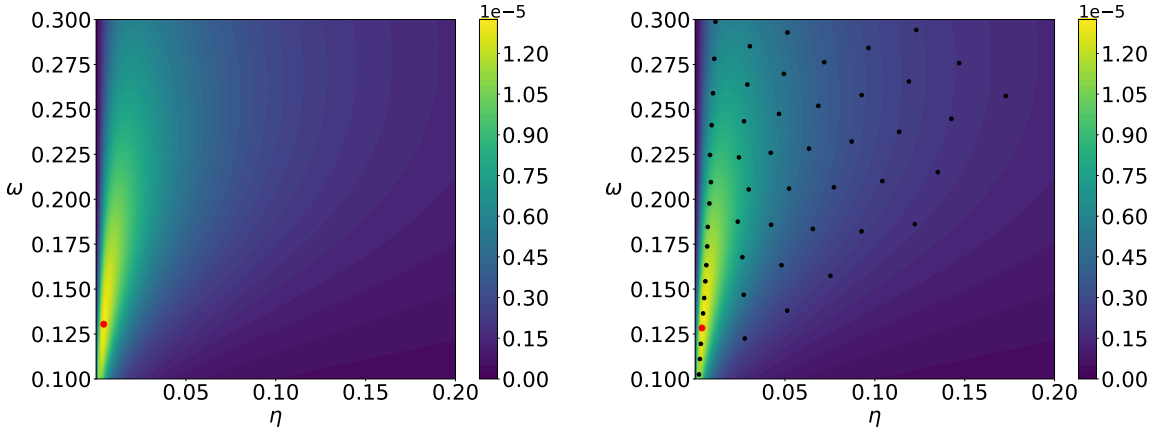
In the one-dim example, the data are generated from the test function $y(x) = x\sin(x) + \epsilon$ for $x \in [0, 10]$ and $\sigma^2 = 0.5^2$. The size of the training and test data sets are $n = 11$ and $m = 100$, respectively.

The parameters of the prior distributions of the EVI-GP are $a_\omega = a_\eta = 1$, $b_\omega = b_\eta = 0.5$, and $df_{\tau^2} = 0$ which is equivalent to $p(\tau^2) \propto 1/\tau^2$. We consider two possible mean functions for the GP model, the constant mean $\mu(x) = \beta_0$ and the linear mean $\mu(x) = \beta_0 + \beta_1 x$. There is no need for parameter regularization for these simple mean functions, and thus we use non-informative prior for $\boldsymbol{\beta}$. We use the EVI-post to approximate $p(\omega, \eta|\boldsymbol{y}_n)$ and set the number of particles $N = 100$, kernel bandwidth $h = 0.02$ and stepsize $\tau = 1$ in the EVI procedure. The initial particles are sampled from the uniform distribution in $[0, 0.1] \times [0.1, 0.4]$. Figure 3.1 and 3.2 show the posterior modes and particles returned by EVI-MAP and EVI-post, as well as the prediction and predictive confidence interval returned by both methods. From Figure 3.1, we can see that the particles generated from EVI-post well approximate the target distribution $p(\omega, \eta|\boldsymbol{y}_n)$ represented by the contours. The posterior modes are accurately identified by EVI-MAP.



(a) Constant mean, EVI-MAP

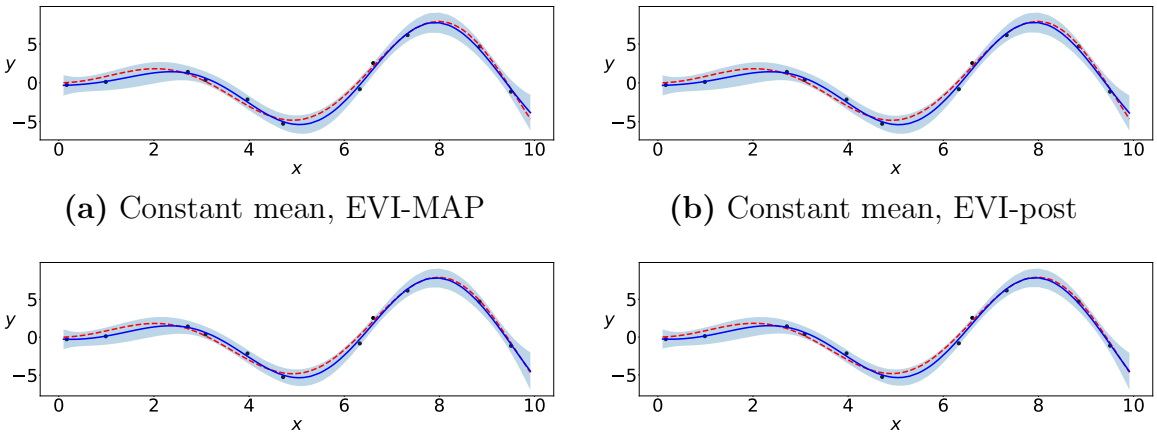
(b) Constant mean, EVI-post



(c) Linear mean, EVI-MAP

(d) Linear mean, EVI-post

Figure 3.1. One-Dim Toy Example: in Figure (a)–(d) the contours are plotted according to $p(\omega, \eta|\boldsymbol{y}_n)$ evaluated on mesh points without the normalizing constant, the largest red points are the modes of the posterior distribution of (ω, η) returned by EVI-MAP approach, and the black dots in Figure (b) and (d) are the particles returned by EVI-post approach. © 2024 IEEE. Reprinted, with permission.



(a) Constant mean, EVI-MAP

(b) Constant mean, EVI-post

(c) Linear mean, EVI-MAP

(d) Linear mean, EVI-post

Figure 3.2. The Gaussian process prediction by four different approaches. Black dots are training data, the red dashed curve is the true test function without noise $y(x) = x\sin(x)$, the blue curve is the predicted curve, and the light blue area shows the 95% predictive confidence interval.

We compare the EVI-GP with three `R` packages. For the `gpfit` package, we set the nugget threshold to be $[0, 25]$, corresponding to η in our method, and the correlation is the Gaussian kernel function. For the `mlegp` package, we set the argument `constMean` to be 0 for the constant mean model and 1 for the linear mean model. The optimization-related arguments in `mlegp` are set as follows. The number of simplexes tries is 5. Each simplex maximum iteration is 5000. The relative tolerance is $1e^{-8}$, BFGS maximum iteration is 500, BFGS tolerance is 0.01, and BFGS bandwidth is $1e^{-10}$. Other parameters in these two packages are set to default. For the `laGP` package, we set the argument `start` at 6, `end` at 10, `d` at null, `g` at $1e^{-4}$, `method` to a list of "alc", "alcray", "mspe", "nn", "fish", and `Xi.ret` at True. The comparison in terms of prediction accuracy is shown in Table 3.1 and Figure 3.3. Table 3.1 shows the mean of RMSPEs from 100 simulations and Figure 3.3 shows the box plots of the RMSPEs. In the third column of Table 3.1, we only list the best result from the three `R` packages. The prediction accuracy of both versions of the EVI-GP performs almost equally well and both significantly outperform the three `R` packages.

Table 3.1. Standardized RMSPE of the One-Dim Toy Example

Mean Model	EVI-post	EVI-MAP	gpfit/mlegp/lagp
Constant	0.1310	0.1311	0.1500
Linear	0.1195	0.1194	0.1584

Figure 3.3 compares the box plots of the RMSPEs from 100 simulations. The prediction accuracy of both versions of the EVI-GP performs almost equally well and significantly outperforms the three `R` packages.

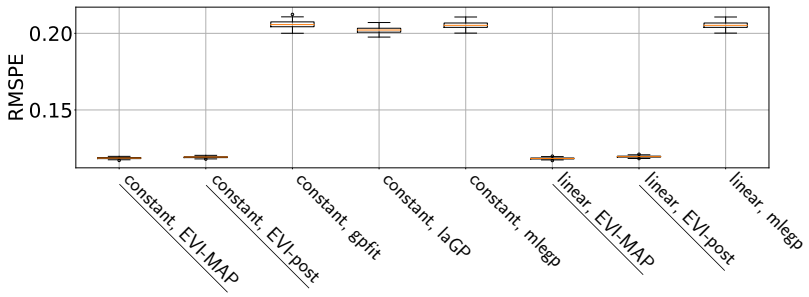


Figure 3.3. Box plots of the standardized RMSPEs of the toy example with different mean models (constant or linear) and different methods (EVI-MAP, EVI-post, gpfit, laGP, mlegp). The labels of the two EVI-GP methods are underlined.

3.3.2 OTL-Circuit function

We test the proposed method using the OTL-circuit function of input dimension $d = 6$. The OTL-circuit function models an output transformerless push-pull circuit. The function is defined as follows,

$$V_m(\boldsymbol{x}) = \frac{(V_{b1} + 0.74)I(R_{c2} + 9)}{I(R_{c2} + 9) + R_f} + \frac{11.35R_f}{I(R_{c2} + 9) + R_f} + \frac{0.74R_f I(R_{c2} + 9)}{(I(R_{c2} + 9) + R_f)R_{c1}}.$$

where $V_{b1} = 12R_{b2}/(R_{b1} + R_{b2})$, $R_{b1} \in [50, 150]$ is the resistance of b1 (K-Ohms), $R_{b2} \in [25, 70]$ is the resistance of b2 (K-Ohms), $R_f \in [0.5, 3]$ is the resistance of f (K-Ohms), $R_{c1} \in [1.2, 2.5]$ is the resistance of c1 (K-Ohms), $R_{c2} \in [0.25, 1.2]$ is the

resistance of c2 (K-Ohms) and $I \in [50, 300]$ is the current gain (Amperes).

The size of training and testing datasets is 200 and 1000, respectively. We set the variance of the noise $\sigma^2 = 0.02^2$. For the prior distributions of $\boldsymbol{\omega}$, we let $\omega_i \overset{\text{i.i.d.}}{\sim} \text{Gamma}(a_\omega = 1, b_\omega = 2)$ for $i = 2, \ldots, 8$, but $\omega_1 \sim \text{Gamma}(a_\omega = 4, b_\omega = 2)$. The prior distribution of $\eta \sim \text{Gamma}(a_\eta = 1, b_\eta = 2)$. Both non-informative and informative prior distributions are considered. When using the informative prior, we set $r = 3$ and $df_{\tau^2} = 7$ for the prior distribution of $\boldsymbol{\beta}$ and τ^2. For the variance of the prior distribution of $\boldsymbol{\beta}$, we choose $\nu = 4.35$, which was the result of 5-fold cross-validation when the biggest mean model is used. After variable selection, 5-fold cross-validation is conducted again to find the optimal ν to fit the finalized GP model, which leads to $\nu = 4.05$. In both cross-validation procedures, ν is searched in an evenly spaced grid from 0 to 5 with 0.05 grid size.

For both EVI-post and EVI-MAP, we set $h = 0.001$ and $\tau = 0.1$. For the EVI-post approach, we use $N = 100$ particles, and the initial particles of $(\boldsymbol{\omega}, \eta)$ are sampled uniformly in $[0, 0.1]^7$. The two versions of EVI-GP perform almost identically, so we only return the result of EVI-MAP. The initial mean model of the GP is assumed as a quadratic function of the input variables, including the 2-way interactions. Based on the 95% posterior confidence interval in Figure 3.4 of the $\boldsymbol{\beta}$, we select x_2 and x_2^2 as the significant terms to be kept for the final model.

For comparison, we choose the `R` package `mlegp` because only this package allows the specification of the mean model. We set the argument `constMean` to be 0 for the constant mean and 1 for the linear mean model. For the full quadratic mean model, we manually add all the second-order effect terms as the input and set `constMean` to 1. Unfortunately, the input of the mean and correlation cannot be separately specified in `mlegp`. So once we specify the mean part to have all the quadratic terms, the input of the correlation function also contains the quadratic terms. But

we believe this might give more advantage for `mlegp` because the correlation involves more input variables. If some of the variables are not needed, their corresponding correlation parameter estimated by `mlegp` can be close to zero. The other arguments in `mlegp` are set in the same way as the previous example. Table 3.2 compares the mean of 100 standardized RMSPEs of the 100 simulations of EVI-GP and `mlegp` and Figure 3.5 shows the box plots of the RMSPEs of different models returned by the two methods. The EVI-GP is much more accurate than `mlegp` package in terms of prediction. However, for this example, variable selection does not improve the model in terms of prediction. The most accurate model is the GP with a linear function of the input variables as the mean model.

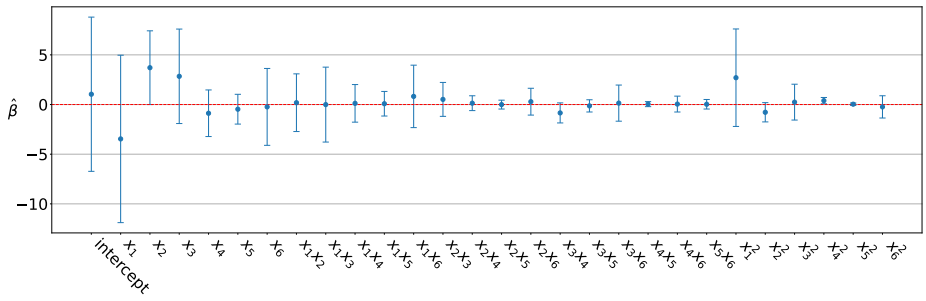


Figure 3.4. The 95% posterior confidence interval of $\boldsymbol{\beta}$ for the full quadratic mean model. © 2024 IEEE. Reprinted, with permission.

Table 3.2. Standardized RMSPE of the OTL-Circuit Example.
© 2024 IEEE. Reprinted, with permission.

Type of mean	EVIGP	mlegp
Constant	0.01608	0.04882
Linear	0.01399	0.03584
Quadratic	0.01792	0.03006
Quadratic, after selection	0.01625	N/A

3.3.3 Borehole function

In this example, we test the EVI-GP with the famous Borehole function, which

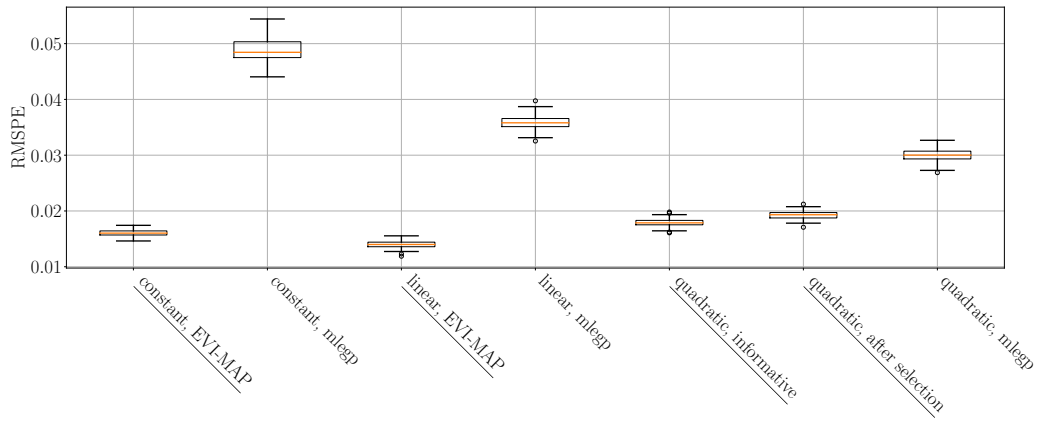


Figure 3.5. Box plots of standardized RMSPE's of the OTL-Circuit Example. © 2024 IEEE. Reprinted, with permission.

is defined as follows:

$$f(\boldsymbol{x}) = \frac{2\pi T_u (H_u - H_l)}{\ln(r/r_\omega)\left(1 + \frac{2LT_u}{\ln(r/r_\omega) r_\omega^2 K_\omega} + \frac{T_u}{T_l}\right)},$$

where $r_\omega \in [0.05, 0.15]$ is the radius of the borehole (m), $r \in [100, 50000]$ is the radius of influence (m), $T_u \in [63070, 115600]$ is the transmissivity of the upper aquifer (m2/yr), $H_u \in [990, 1110]$ is the potentiometric head of upper aquifer (m),$T_l \in [63.1, 116]$ is the transmissivity of the lower aquifer (m2/yr), $H_l \in [700, 820]$ is the potentiometric head of lower aquifer (m), $L \in [1120, 1680]$ is the length of the borehole (m), $K_\omega \in [9855, 12045]$ is the hydraulic conductivity of borehole (m/yr).

We use a training dataset of size 200 and 100 testing datasets of size 100. The noise variance is $\sigma^2 = 0.02^2$. We use the same parameter settings for the prior distributions and EVI-GP method as in the OTL-Circuit example. Regarding the variance of the prior distribution of $\boldsymbol{\beta}$, $\nu = 4.55$ was the result of 5-fold cross-validation when the biggest mean model was used. After variable selection, the 5-fold cross-validation is conducted again to find the optimal ν to fit the finalized GP model. Coincidently, the optimal ν is also 4.55. The initial mean model of the GP is assumed

as a quadratic function of the input variables, including the 2-way interactions. Based on the 95% posterior confidence interval in Figure 3.6 of the $\boldsymbol{\beta}$, we select intercept, x_1, x_4, and x_1x_4 as the significant terms to be kept for the final model. Similarly, we compare the EVI-GP with the `mlegp` package, and the RMSPEs of the 100 simulations are shown in Table 3.3 and Figure 3.7. Again, EVI-GP is much better than the `R` package. More importantly, variable selection proves to be essential for the Borehole example, as the final GP model with the reduced mean model is more accurate than both the quadratic and linear mean models.

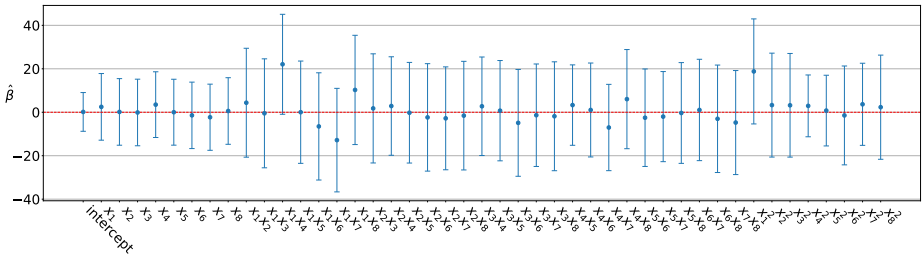


Figure 3.6. The 95% posterior confidence interval of $\boldsymbol{\beta}$ for the full quadratic mean model. © 2024 IEEE. Reprinted, with permission.

Table 3.3. Standardized RMSPE of the Borehole Example.
© 2024 IEEE. Reprinted, with permission.

Type of mean	EVIGP	mlegp
Constant	0.01151	0.3354
Linear	0.04191	0.1810
Quadratic	0.01212	0.2019
Quadratic, after selection	0.01019	N/A

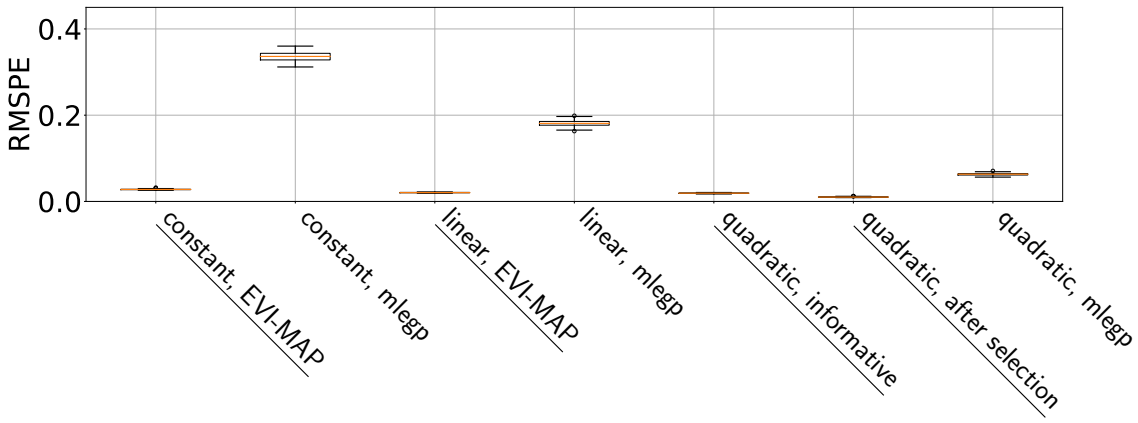


Figure 3.7. Box plots of standardized RMSPE's of the Borehole Example. © 2024 IEEE. Reprinted, with permission.

CHAPTER 4

ACTIVE LEARNING FOR MANIFOLD GAUSSIAN PROCESS REGRESSION

4.1 Manifold Gaussian Process Regression

[3]The Manifold Gaussian Process (mGP) [23] addresses key limitations of standard Gaussian Process Regression (GPR) by jointly learning a data transformation and a GPR model. Standard GPs rely on covariance functions (e.g., squared exponential) that encode smoothness assumptions, which may be inadequate for modeling discontinuous or complex functions (e.g., robotics tasks with contact dynamics), as pointed out in [65]. To counter this, the mGP framework decomposes the regression task into:

$$F = G \circ M,$$

where $M : \mathcal{X} \to \mathcal{H}$ is a deterministic mapping from the input space $\mathcal{X}$ to a latent feature space $\mathcal{H} \subset \mathbb{R}^Q$, and $G : \mathcal{H} \to \mathcal{Y}$ is a GPR model on $\mathcal{H}$. Here $\mathcal{Y} \subset \mathbb{R}$ is the space of the response, and G now assume the mean function is $\mathbf{0}$.

For the map M, we employ multilayer neural networks. In each layer, the transformation is defined as $\boldsymbol{Z}_i = T_i(\boldsymbol{x}) = \sigma_M(\boldsymbol{W}_i \boldsymbol{Z}_{i-1} + \boldsymbol{B}_i)$, where σ_M is the activation function and $\boldsymbol{W}_i$ and $\boldsymbol{B}_i$ are the weight matrix and bias vector for the i-th layer. Consequently, $M(\boldsymbol{x}) = T_l \circ T_{l-1} \circ \cdots \circ T_1(\boldsymbol{Z}_0)$ with $\boldsymbol{Z}_0 = \boldsymbol{x}$, and the collection of these hyperparameters is denoted by $\boldsymbol{\theta}_M$.

The mGP defines a valid GP over $\mathcal{X}$ with the covariance function:

$$\tau^2 \tilde{k}(\boldsymbol{x}_1, \boldsymbol{x}_2) = \tau^2 k(M(\boldsymbol{x}_1), M(\boldsymbol{x}_2)),$$

[3]This chapter is adapted from [2]. See authorship statement for more details.

where k is the standard kernel (e.g., SE-ARD or neural network kernel) with hyperparameters $\boldsymbol{\theta}_G$. This formulation allows the model to adaptively learn feature representations that align with the regression objective, overcoming the limitations of unsupervised transformations (e.g., PCA or random embeddings) that may not preserve the task-relevant structure.

The predictive distribution of y at any new query point $\boldsymbol{x}$ is also normally distributed with

$$\mathrm{E}(y(\boldsymbol{x})|\boldsymbol{y}, \tau^2, \eta, \boldsymbol{\theta}) = \tilde{\boldsymbol{k}}_n(\boldsymbol{x})'(\tilde{\boldsymbol{K}}_n + \hat{\eta}\boldsymbol{I}_n)^{-1}\boldsymbol{y}, \tag{4.1.1}$$

$$s_n^2(\boldsymbol{x}) \triangleq \mathrm{var}(y(\boldsymbol{x})|\boldsymbol{y}, \tau^2, \eta, \boldsymbol{\theta}) = \hat{\tau}^2 \left[1 - \tilde{\boldsymbol{k}}(\boldsymbol{x})'(\tilde{\boldsymbol{K}}_n + \hat{\eta}\boldsymbol{I}_n)^{-1}\tilde{\boldsymbol{k}}(\boldsymbol{x})\right] + \hat{\sigma}^2, \tag{4.1.2}$$

where $\tilde{\boldsymbol{k}}_n(\boldsymbol{x})$ is the correlation between $M(\boldsymbol{x})$ and $M(\boldsymbol{x}_i)$, i.e., $\tilde{\boldsymbol{k}}_n(\boldsymbol{x}) = [k(M(\boldsymbol{x}), M(\boldsymbol{x}_1)), \ldots, k(M(\boldsymbol{x}), M(\boldsymbol{x}_n))]^\top$. The GPR predictor is the conditional mean $\hat{y}(\boldsymbol{x}) = \tilde{\boldsymbol{k}}_n(\boldsymbol{x})'(\tilde{\boldsymbol{K}}_n + \hat{\eta}\boldsymbol{I}_n)^{-1}\boldsymbol{y}$. It is also the Best Linear Unbiased Predictor (BLUP) [66] for $y(\boldsymbol{x})$ under the GP assumption. We can construct the predictive confidence interval for $y(\boldsymbol{x})$ from (4.1.2).

The mGP optimizes the parameters $\boldsymbol{\theta}_{\mathrm{mGP}} = [\boldsymbol{\theta}_M, \boldsymbol{\theta}_G]$ jointly by minimizing the Negative Log Marginal Likelihood (NLML):

$$\mathrm{NLML}(\boldsymbol{\theta}_{\mathrm{mGP}}) = n \log \tau^2 + \log\det(\tilde{\boldsymbol{K}}_n + \eta\boldsymbol{I}_n) + \frac{1}{\tau^2}\boldsymbol{y}^\top(\tilde{\boldsymbol{K}}_n + \eta\boldsymbol{I}_n)^{-1}\boldsymbol{y},$$

where $\tilde{\boldsymbol{K}}_n$ is the kernel matrix evaluated on the transformed inputs $M(\boldsymbol{x})$. Gradients are computed via backpropagation through M (if M is a neural network) and the kernel hyperparameters $\boldsymbol{\theta}_G$. The predictive distribution at a test point $\boldsymbol{x}_{n+1}$ is a normal distribution with mean and variance specified by (4.1.1) and (4.1.2) except $\boldsymbol{k}_n$ is replaced by $\tilde{\boldsymbol{k}}_n$ and $\boldsymbol{K}_n$ by $\tilde{\boldsymbol{K}}_n$.

One can see that the Deep Gaussian Process (or DGP) [67] is closely connected with mGP since both involve latent space transformations before applying GPR. However, they are distinct extensions of standard GPR because DGPs learn this transformation hierarchically through layers of GPs, while mGPs often assume a geometric prior and embed the data accordingly. The mGP framework offers several key advantages over traditional GPs and unsupervised feature learning methods. By jointly optimizing the feature mapping M and GPR G under a supervised objective, it learns task-aware representations that can handle discontinuities (e.g., step functions) and multi-scale patterns more effectively than standard kernels like SE-ARD or neural network covariances. The flexibility of parameterizing M as a neural network (e.g., $[1-6-2]$ layer architecture) allows it to unwrap complex input geometries, while retaining the GP's probabilistic uncertainty quantification—critical for robotics and control applications. However, this approach also introduces challenges: the joint optimization $\boldsymbol{\theta}_{\text{mGP}} = [\boldsymbol{\theta}_M, \boldsymbol{\theta}_G]$ is non-convex and may converge to suboptimal local minima, particularly with high-dimensional $\boldsymbol{\theta}_M$. Additionally, sparse data around discontinuities can lead to overfitting in the deterministic mapping M, causing misaligned feature representations. While probabilistic extensions (e.g., Bayesian neural networks for M) could mitigate this, they often sacrifice tractability. Despite these limitations, the mGP's empirical performance on benchmarks in [23] demonstrates its superiority in scenarios where input-space geometry violates standard GP assumptions.

4.2 Neural Network Architecture and Training Considerations

Note: This section is exploratory and implementation-oriented. It discusses architecture and training heuristics that were helpful in our empirical study but are not part of the core theoretical framework. Our goal is to document the process and provide transparency for potential reproduction and extension.

In our proposed framework, the input data is first mapped to a lower dimensional latent space using a neural network-based encoder. The resulting latent representation is then used as the input for Gaussian Process (GP) regression. While the conceptual architecture follows the structure of a typical autoencoder [68, 69], several design decisions were made to enhance representation learning, numerical stability, and GP compatibility.

4.2.1 Choice of Activation Function

The encoder and decoder networks employ the log-sigmoid activation function, defined as:

$$\text{log-}\sigma(x) := \log\left(\frac{1}{1+e^{-x}}\right) = -\log(1+e^{-x}).$$

This function is a smoothed, bounded transformation that compresses the real line into the interval $(-\infty, 0)$. Its derivative is given by:

$$\frac{d}{dx}\,\text{log-}\sigma(x) = \frac{1}{1+e^{x}} = \sigma(-x),$$

which remains bounded and non-zero across a wide range of input values, avoiding the vanishing gradient problem encountered in saturated regions of the sigmoid or tanh activations.

Compared to ReLU, which yields sparse and non-negative outputs, the log-sigmoid activation introduces smoother transitions and ensures that the latent representation does not explode in magnitude [70]. Its outputs naturally decay for large negative x and asymptotically approach 0 as $x \to \infty$:

$$\lim_{x\to-\infty} \text{log-}\sigma(x) = -\infty, \quad \lim_{x\to\infty} \text{log-}\sigma(x) = 0.$$

This asymmetry is particularly useful when encoding curved manifolds with direc-

tional variance. Furthermore, its smoothness ensures stable gradient propagation through the autoencoder, especially under GP loss backpropagation.

4.2.2 Normalization for Latent Output Stability

Let $\boldsymbol{x}_i \in \mathbb{R}^D$ denote the original input, and $\boldsymbol{z}_i = M(\boldsymbol{x}_i) \in \mathbb{R}^d$ denote the latent representation obtained from the encoder M. When training a Gaussian Process (GP) model on latent inputs $\{\boldsymbol{z}_i\}_{i=1}^N$, it is crucial to ensure that the distribution of $\boldsymbol{z}_i$ remains well-scaled and numerically stable.

Without normalization, the latent outputs may drift during training. This affects the evaluation of the GP kernel $k(\boldsymbol{z}_i, \boldsymbol{z}_j)$, especially for distance-based kernels like the RBF. If $\|\boldsymbol{z}_i - \boldsymbol{z}_j\|$ becomes large because of unbounded outputs of M, the kernel values rapidly decay to zero, resulting in a near-singular covariance matrix K, leading to numerical instability in computing $\log|K|$ and $K^{-1}\boldsymbol{y}$.

To mitigate this, we insert a Batch Normalization (BN) layer at the encoder's output [71], just before the GP layer. Denote the batch of latent vectors as $\boldsymbol{Z} = [\boldsymbol{z}_1, \ldots, \boldsymbol{z}_N]^\top \in \mathbb{R}^{N\times d}$. The BN-transformed output is:

$$\tilde{z}_{i,j} = \gamma_j \cdot \frac{z_{i,j} - \mu_j}{\sqrt{\sigma_j^2 + \epsilon}} + \beta_j, \quad j = 1, \ldots, d,$$

where $\mu_j = \frac{1}{N}\sum_{i=1}^N z_{i,j}$ is the batch mean of dimension j, $\sigma_j^2 = \frac{1}{N}\sum_{i=1}^N (z_{i,j} - \mu_j)^2$ is the batch variance, γ_j, β_j are learnable scale and shift parameters, and ϵ is a small constant for numerical stability.

This transformation ensures that each latent dimension is approximately zero-mean and unit-variance over each batch:

$$\mathbb{E}_{\text{batch}}[\tilde{z}_j] \approx 0, \quad \text{Var}_{\text{batch}}[\tilde{z}_j] \approx \gamma_j^2.$$

This stabilizes the scale of pairwise distances:

$$\|\tilde{\boldsymbol{z}}_i - \tilde{\boldsymbol{z}}_j\|^2 = \sum_{k=1}^{d} (\tilde{z}_{i,k} - \tilde{z}_{j,k})^2,$$

ensuring that kernel values remain in a numerically useful range, and that gradients propagated through the GP layer remain well-behaved.

As an added benefit, the normalization also helps mitigate internal covariate shift during training, which otherwise leads to unstable updates and degraded GP.

4.2.3 A combined loss

A key design motivation is to emphasize learning a meaningful manifold structure rather than treating the encoder as a black-box transformation, like elastic net regression using both $l1$ loss and $l2$ loss. To this end, we formulate the encoder-decoder pair as a full autoencoder and include a reconstruction loss:

$$\mathcal{L}_{\text{recon}} = \frac{1}{N} \sum_{i=1}^{N} \|M_{\text{decoder}}(M_{\text{encoder}}(\boldsymbol{x}_i)) - \boldsymbol{x}_i\|^2,$$

This term encourages the latent embedding to preserve the essential geometry and topology of the input space. Without this term, the learned latent manifold may collapse or become degenerate during optimization purely driven by the GP marginal likelihood.

In practice, the joint training of the autoencoder and GP leads to unstable optimization trajectories, particularly when the GP's loss gradient flows back through the encoder. To mitigate this, we apply L_2 regularization to the encoder and decoder weights:

$$\mathcal{L}_{\text{reg}} = \lambda \sum_{l} \|W_l\|_F^2.$$

This regularization term penalizes overly large weights and encourages smoother mappings in the latent space, improving robustness across different initialization seeds and batch sizes.

The final training objective is a composite loss that balances GP regression, reconstruction fidelity, and model stability:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{GP}} + \alpha \mathcal{L}_{\text{recon}} + \beta \mathcal{L}_{\text{reg}},$$

where α and β are tuning constants chosen empirically to balance learning quality and numerical stability. In the following experiments, we set $\alpha = 1, \beta = 0$. These constants allow us to adjust the influence of reconstruction and regularization relative to the GP's negative log marginal likelihood loss $\mathcal{L}_{\text{GP}}$.

4.2.4 Remark on Latent Space Geometry

The effectiveness of Gaussian Process regression in the latent space $\mathcal{H}$ implicitly depends on the geometric properties of the encoder mapping $M : \mathcal{X} \to \mathcal{H}$. In particular, for the GP kernel to remain meaningful (e.g., stationary and distance-based), we assume that the encoder preserves local neighborhoods and maintains smooth transitions in $\mathcal{H}$.

While autoencoders are trained using reconstruction loss, which encourages local continuity, they do not inherently enforce global distance preservation or isometry. In this work, we adopt standard fully connected encoders without additional geometric regularization, but we acknowledge that incorporating loss terms such as contrastive penalties, neighborhood preservation (e.g., Laplacian Eigenmaps), or isometric mappings may improve structure alignment and are potential future directions.

This design enables the encoder to not only serve the GP's inference quality,

but also to learn a semantically meaningful and stable representation of the input space.

4.3 Active Learning for mGP

The Active Learning Cohn (ALC) criterion [72, 27] is a foundational strategy for sequential experimental design in Gaussian Process Regression (GPR). It selects new data points $\boldsymbol{x}_{n+1}$ that maximize the model's average predictive uncertainty over the input space. Formally, this is achieved by minimizing the *integrated mean squared error (IMSE)* across the input space [26, 73], which corresponds to reducing the average posterior variance. Neglecting the constant noise variance, the IMSE is defined as:

$$\breve{s}^2_{n+1}(\boldsymbol{x}|\boldsymbol{x}_{n+1}) \triangleq \hat{\tau}^2 \left[1 - \tilde{\boldsymbol{k}}_{n+1}(\boldsymbol{x})^\top (\tilde{\boldsymbol{K}}_{n+1} + \eta \boldsymbol{I}_{n+1})^{-1} \tilde{\boldsymbol{k}}_{n+1}(\boldsymbol{x})\right],$$
$$\mathrm{IMSE}(\boldsymbol{x}_{n+1}) \triangleq \int_{\mathcal{X}} \breve{s}^2_{n+1}(\boldsymbol{x}|\boldsymbol{x}_{n+1})\, \mathrm{d}\boldsymbol{x},$$

where $\boldsymbol{x}_{n+1}$ is selected from a finite candidate pool $\mathcal{X}_{\text{cand}} \subset \mathcal{X}$, typically constructed using a space-filling design. The term $\tilde{\boldsymbol{k}}_{n+1}(\boldsymbol{x}) = [\tilde{k}(\boldsymbol{x}, \boldsymbol{x}_1), \ldots, \tilde{k}(\boldsymbol{x}, \boldsymbol{x}_{n+1})]^\top$ and $\tilde{\boldsymbol{K}}_{n+1}$ denote the kernel vector and $(n+1) \times (n+1)$ correlation matrix evaluated in the learned latent space $\mathcal{H} = M(\mathcal{X})$ using the current mGP parameters.

Minimizing the IMSE is equivalent to maximizing the expected reduction in posterior variance, leading to the ALC acquisition function:

$$\begin{aligned}\mathrm{ALC}(\boldsymbol{x}_{n+1}) &= \int_{\mathcal{X}} s^2_n(\boldsymbol{x}) - \breve{s}^2_{n+1}(\boldsymbol{x}|\boldsymbol{x}_{n+1})\, \mathrm{d}\boldsymbol{x} \\ &\propto \int_{\mathcal{X}} \hat{\tau}^2 \tilde{\boldsymbol{k}}_{n+1}(\boldsymbol{x})^\top (\tilde{\boldsymbol{K}}_{n+1} + \eta \boldsymbol{I}_{n+1})^{-1} \tilde{\boldsymbol{k}}_{n+1}(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x},\end{aligned}$$

where $s^2_n(\boldsymbol{x})$ is computed via (4.1.2) except that k function is replaced by $\tilde{k}$, and thus $\boldsymbol{k}_n(\boldsymbol{x})$ by $\tilde{\boldsymbol{k}}_n(\boldsymbol{x})$ and $\boldsymbol{K}_n$ by $\tilde{\boldsymbol{K}}_n$. To approximate the integration numerically, we

introduce a fixed reference set $\mathcal{X}_{\text{ref}}$ of size m sampled from $\mathcal{X}$ using Latin Hypercube Design. The ALC objective becomes:

$$\text{ALC}(\boldsymbol{x}_{n+1}) \propto \sum_{\boldsymbol{x} \in \mathcal{X}_{\text{ref}}} \hat{\tau}^2 \tilde{\boldsymbol{k}}_{n+1}(\boldsymbol{x})^\top (\tilde{\boldsymbol{K}}_{n+1} + \eta \boldsymbol{I}_{n+1})^{-1} \tilde{\boldsymbol{k}}_{n+1}(\boldsymbol{x}).$$

In contrast to [74], which employs DGPs, our method integrates ALC with mGPs. At each active learning iteration, the most informative training point is selected via:

$$\boldsymbol{x}^*_{n+1} = \arg\max_{\boldsymbol{x} \in \mathcal{X}_{\text{cand}}} \text{ALC}(\boldsymbol{x}),$$

and subsequently removed from the candidate pool $\mathcal{X}_{\text{cand}}$. This selection procedure iteratively reduces overall model uncertainty, leveraging the mGP's ability to capture complex geometric structures in latent space. To enhance computational efficiency, we adopt two strategies. First, we support batch selection by choosing B design points per iteration, indexed by round r. Second, instead of evaluating ALC on the full candidate set, we incorporate a pre-screening step: among the candidate set $\mathcal{X}_{\text{cand}}$, we identify the top $K > B$ points with the highest predictive variance, denoted $\mathcal{X}^*_r$:

$$\mathcal{X}^{(r)}_{\text{cand}} = \left\{ \boldsymbol{x}^{(r)}_1, \boldsymbol{x}^{(r)}_2, \ldots, \boldsymbol{x}^{(r)}_K \right\},$$

and compute ALC only on $\mathcal{X}^{(r)}_{\text{cand}}$. We then select the B points with the highest ALC scores. The complete procedure is outlined in Algorithm 2.

4.4 Examples

We evaluate the Active Learning Manifold Gaussian Process (ALmGP) framework on four benchmark experiments. These experiments are designed to assess the effectiveness of ALmGP under varying geometric structures and data complexities. The acquisition strategy is based on the ALC criterion and model optimiza-

Algorithm 2 Active Learning for Manifold Gaussian Process Regression (ALmGP)

Input: Initial training dataset $\{\boldsymbol{x}_i, y_i\}_{i=1}^{n_0}$ with small sample size n_0; reference set $\mathcal{X}_{\text{ref}} \subset \mathcal{X}$ and candidate set $\mathcal{X}_{\text{cand}} \subset \mathcal{X}$; tuning parameters including screening size K, batch size B, convergence threshold `Tol`, maximum total sample size $N_{\max}$, and initial values and optimization settings for fitting the mGP.

Output: Trained mGP model and estimated parameters after active data acquisition.

Step 0: Fit the mGP model using the training set $\{\boldsymbol{x}_i, y_i\}_{i=1}^{n_0}$ by minimizing the negative log marginal likelihood (NLML) with respect to $\boldsymbol{\theta}_{\text{mGP}}$ using the L-BFGS algorithm [75].

for $r = 1, 2, \ldots, \lfloor N_{\max}/B \rfloor$ **do**

Step 1: Compute posterior predictive variance for all points in $\mathcal{X}_{\text{cand}}$, and select the top K points with highest variance. Denote this screened set as $\mathcal{X}_{\text{cand}}^{(r)}$.

Step 2: For each $\boldsymbol{x} \in \mathcal{X}_{\text{cand}}^{(r)}$, compute the ALC acquisition score $\text{ALC}(\boldsymbol{x})$ using the current mGP model.

Step 3: Select the top B points in $\mathcal{X}_{\text{cand}}^{(r)}$ with the highest ALC values. Add them to the training set and remove them from $\mathcal{X}_{\text{cand}}$.

Step 4: Re-train or update the mGP model using the expanded training dataset.

Step 5: Evaluate the cross-validation error or training mean squared error (MSE). If the error falls below the threshold `Tol`, terminate the loop and return the final fitted mGP model.

end for

tion is carried out using the L-BFGS algorithm implemented in the PyTorch library [13]. The optimizer employs strong Wolfe line search conditions (`line_search_fn = strong_wolfe`), with early stopping triggered when the relative RMSE change

$$\text{Relative RMSE Change} = \left| \frac{\mathcal{L}_{\text{current}} - \mathcal{L}_{\text{previous}}}{\mathcal{L}_{\text{current}} - \mathcal{L}_{\text{initial}}} \right|$$

falls below a predefined tolerance threshold. To ensure positivity and improve stability during optimization, all hyperparameters of the mGP model are squared before being passed to the optimizer. All neural network weights are initialized using PyTorch's default scheme, while GP hyperparameters—output variance and length scale—are initialized to 1.

Model performance is evaluated using the Root Mean Square Error (RMSE)

computed on an independently generated test set, which has n_{test} number of points sampled from $\mathcal{X}$ using Latin Hypercube Design:

$$\text{RMSE} = \sqrt{\frac{1}{n_{\text{test}}} \sum_i (\hat{y}_i - y_i)^2},$$

where $\hat{y}_i$ and y_i denote the predicted and ground truth values at test location $\boldsymbol{x}_i$, respectively. Each experiment is repeated 10 times to ensure statistical robustness. The mean and range of RMSE values are reported and compared against a baseline strategy using completely random sampling B design points from $\mathcal{X}_{\text{cand}}$.

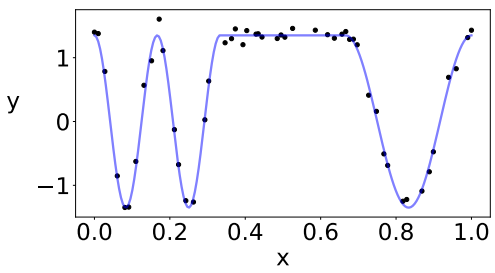
4.4.1 Piecewise Trigonometric Function

This example features a piecewise trigonometric function adapted from [74], with noise $\epsilon \sim \mathcal{N}(0, 0.1^2)$:

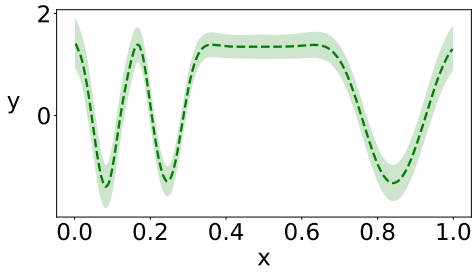
$$F(x) = \begin{cases} 1.35\cos(12\pi x), & x \in [0, 0.33], \\ 1.35, & x \in [0.33, 0.66], \\ 1.35\cos(6\pi x), & x \in [0.66, 1]. \end{cases}$$

The initial dataset consists of $n_0 = 10$ points generated using Latin Hypercube Design over $[0, 1]$. The test set contains 500 uniformly spaced points, and both $\mathcal{X}_{\text{cand}}$ and $\mathcal{X}_{\text{ref}}$ consist of 100 evenly spaced points in the same interval. The neural network uses [1-6-2] architecture and LogSigmoid activation. The batch size is $B = 1$, and the total budget is $N_{\max} = 50$. L-BFGS uses `history_size` $= 20$, a learning rate of 0.001, a maximal number of iterations per optimization step 20 and a maximum of 5,000 training iterations with early stopping tolerance `Tol` $= 10^{-5}$. The active learning process ends with 15 new samples and achieves an average completion RMSE of 0.156. Figure 4.1 shows the details of the training data, the fitted function, the latent space, and the comparison of RMSE of active learning for mGP using the proposed

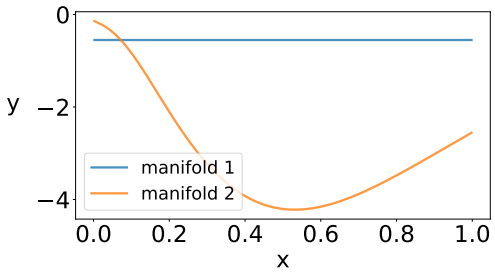
ALC strategy and random sampling.



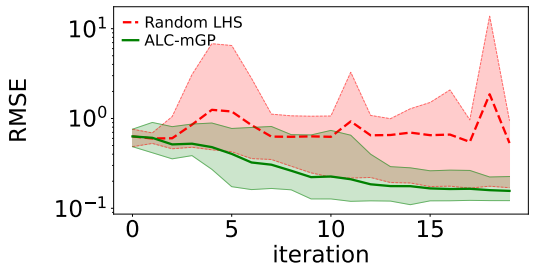
(a) Final selected training data (black dots) and true function (blue curve)



(b) Final predicted value (green dashed line) and confidence interval (green shaded area)



(c) Learned two-dimensional latent space



(d) test RMSE over iterations

Figure 4.1. (a) The black dots indicate the final selected training data and the blue solid line is the true piecewise trigonometric function. (b) The green dashed line represents the final predicted value and the green shaded area denotes the prediction confidence interval. (c) Learned two-dimensional latent space. (d) Comparison of test RMSE over iterations: our method (green, solid) vs. random acquisition (red, dashed). Lines show the mean of 10 runs; shaded areas indicate the minimum to maximum range. © 2025 IEEE. Reprinted, with permission.

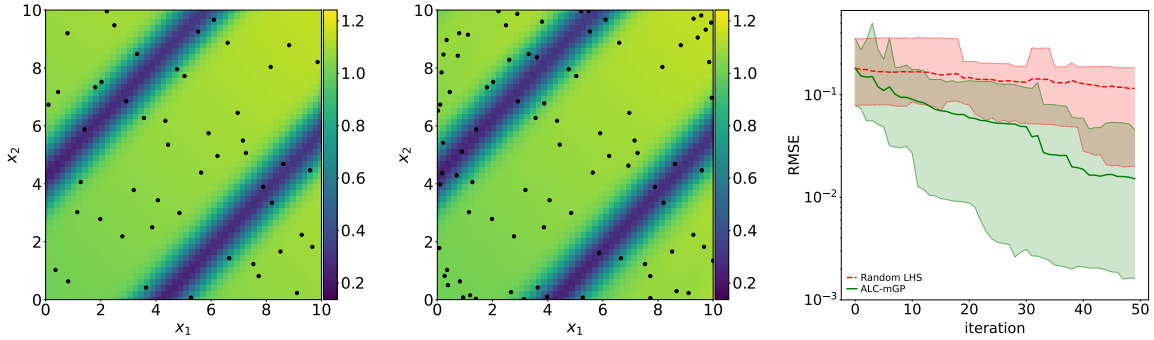
4.4.2 Two-Dimensional Deterministic Function

We consider a two-dimensional function also from [23], defined as:

$$f(x_1, x_2) = 1 - \phi(x_2; 3, 0.5^2) - \phi(x_2; -3, 0.5^2) + \frac{x_1}{100},$$

which is then rotated by 45°. The function $\phi(\cdot; \mu, \sigma^2)$ is the PDF of $\mathcal{N}(\mu, \sigma^2)$. Then initial training data ($n_0 = 50$), test data, and the reference set $\mathcal{X}_{\text{ref}}$ are all of size 500 and sampled from $[0, 10]^2$ using LHD. Predictions are also evaluated on a grid over $\mathcal{X}$ with mesh size 0.2. This neural network maps the 2D input to a latent space, and

the architecture is $[2 - 10 - 3]$, with the same LogSigmoid activation. The active learning proceeds with batch size $B = 1$ and budget $N_{\max} = 50$. The L-BFGS is run with `history_size` $= 50$, learning rate 0.01, a maximal number of iterations per optimization step 50, and a max of 5000 iterations. The active learning procedure concludes with 100 samples after 50 iterations, reaching an average RMSE of 0.0152 for the proposed method. Figure 4.2 and 4.3 show the detailed results.

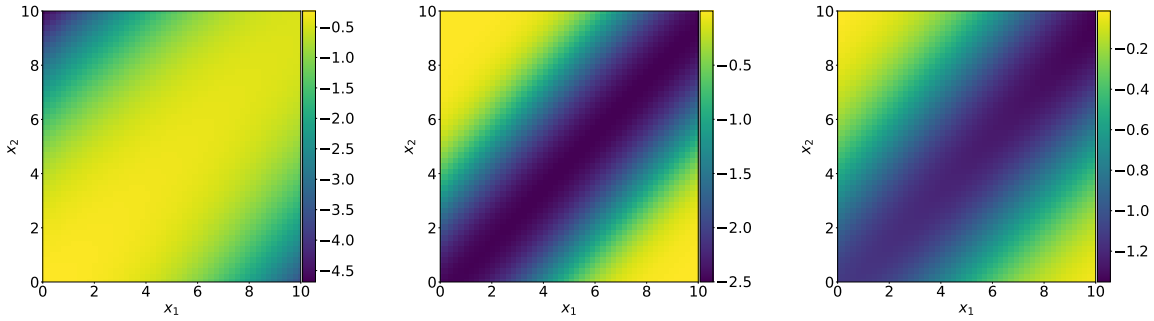


(a) True function heatmap with initial training data (black dots)

(b) predicted heatmap with full training data using all the budgets

(c) Test RMSE: our method (green, solid) vs. random (red, dashed)

Figure 4.2. (a) The heat map is the true function value with the black dots representing the initial training data. (b) The heat map is the final predicted value, and the black dots are the overall training data. (c) Comparison of test RMSE over iterations: our method (green, solid) vs. random acquisition (red, dashed). Lines show the mean of 10 runs; shaded areas indicate the minimum to maximum range.



(a) Latent dimension 1

(b) Latent dimension 2

(c) Latent dimension 3

Figure 4.3. The three heat maps represent three latent dimensions with respect to the original input (x_1, x_2).

4.4.3 Three-Dimensional Function on the Unit Sphere

This example uses a test function defined on the 3D unit sphere:

$$f(x, y, z) = \cos(x) + y^2 + e^z, \quad \text{where } x^2 + y^2 + z^2 = 1.$$

It can be reformulated on the disk as $g(x, y) = \cos(x) + y^2 + e^{1-x^2-y^2}$ for $x^2 + y^2 \leq 1$. We construct training, test, and reference sets (each of size 500) by sampling $(v, \alpha) \in [-1, 1] \times [0, 2\pi]$ using Latin Hypercube Design and mapping them to (x, y, z) via:

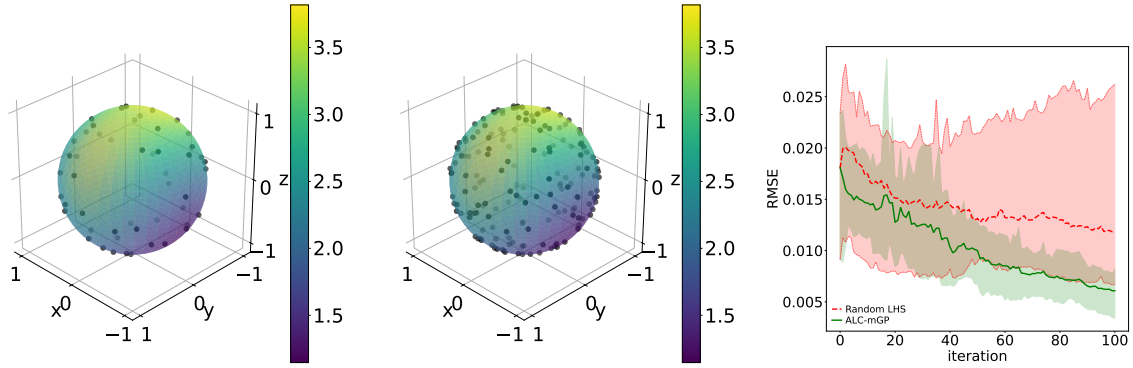
$$z = v, \quad x = \sqrt{1 - v^2}\cos(\alpha), \quad y = \sqrt{1 - v^2}\sin(\alpha).$$

The neural network has architecture [3-10-2], and training starts from $n_0 = 50$ samples. Active learning is run with $N_{\max} = 100$ and $B = 1$. The L-BFGS is configured with `history_size` $= 50$, a learning rate of 0.01, a maximal number of iterations per optimization step 20 and 5,000 training iterations. The active learning stops when it reaches the full budget of 100 samples, yielding the averaged RMSE as low as 6×10^{-3} for the proposed method. See the results in Figure 4.4 and 4.5.

4.4.4 Borehole

This example models the groundwater flow function of the borehole, a well-known benchmark for nonlinear regression. The function depends on eight physical parameters, such as radius and aquifer transitivity. The detailed definition of the variable and the math expression of the function are included in [76]. Inputs are generated via LHS and mapped to physical ranges.

The neural network has architecture [8-30-4], and training starts from $n_0 = 50$ samples. Active learning is run with $N_{\max} = 150$ and $B = 1$. The L-BFGS uses `history_size` $= 50$, learning rate 0.001, a maximal number of iterations per

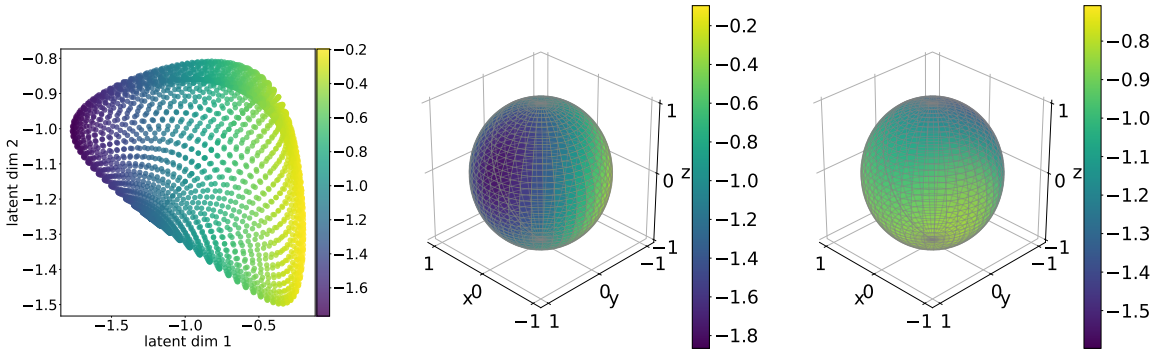


(a) True on 3D grid and initial training data

(b) Prediction on 3D grid and training data using all the budgets

(c) test RMSE vs iteration: our method (green, solid) vs. random (red, dashed)

Figure 4.4. (a) The heat map represents the true function value on the 3-dim unit sphere and the dots are the initial training data. (b) The heat map is the final predicted value and the black dots are the overall training data. (c) Comparison of test RMSE over iterations: our method (green, solid) vs. random acquisition (red, dashed). Lines show the mean of 10 runs; shaded areas indicate the minimum to maximum range. © 2025 IEEE. Reprinted, with permission.



(a) Learnt latent space

(b) Mapping from original 3D space to latent dimension 1

(c) Mapping from original 3D space to latent dimension 2

Figure 4.5. (a) The heat map represents the true function value f with respect to the two latent dimensions. (b) The heat map represents the 1st latent dimension with respect to (x, y, z). (c) The heat map represents the 2nd latent dimension with respect to (x, y, z). © 2025 IEEE. Reprinted, with permission.

optimization step 100, and 10,000 training iterations. Early stopping is triggered when the relative test RMSE change is below 10^{-8}. The proposed active learning stops when it reaches the full budget of 150 samples, yielding an average RMSE as

low as 0.605. See the results in Figure 4.6.

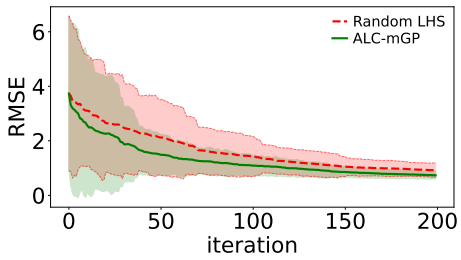


Figure 4.6. Comparison of test RMSE over iterations: our method (green, solid) vs. random acquisition (red, dashed). Lines show the mean of 10 runs; shaded areas indicate the minimum to maximum range.

CHAPTER 5
DISCUSSION AND CONCLUSION

5.1 Motivation Revisited

This dissertation began with a fundamental question: how can Gaussian Process (GP) regression be extended to complex scientific systems where data are scarce, dimensions are high, and computational resources are constrained?

To address this, we developed two complementary methodological directions. First, we introduced an energetic variational inference (EVI) framework for GP regression, which formulates Bayesian inference as a dynamic energy minimization process. This approach bypasses fixed-form variational assumptions and enables stable posterior approximation through particle-based dynamics, with built-in support for variable selection.

Second, we proposed a manifold-based active learning strategy that performs GP regression in a low-dimensional latent space constructed by an autoencoder. Samples are sequentially selected using the Active Learning Cohn (ALC) criterion to reduce global uncertainty efficiently, especially in high-dimensional, simulation-based contexts.

Together, these contributions aim to make GP modeling more adaptive to structural constraints and more scalable for scientific surrogate modeling. They provide practical tools and conceptual insights for improving both inference and data acquisition in probabilistic learning systems.

5.2 Limitations

Despite their advantages, the proposed methods come with limitations.

The EVI framework is sensitive to initialization, particularly in the choice of kernel and the placement of initial particles. While heuristics were sufficient in our experiments, a more principled approach to initialization remains an open question.

In the manifold method, the quality of the learned latent space heavily depends on the expressiveness and training stability of the autoencoder. For data with complex topologies or weak manifold structure, dimensionality reduction may distort important features, reducing GP performance.

Both methods also introduce computational overhead. Particle evolution in EVI and encoder-decoder training in active learning require additional resources, which may limit scalability in real-time or resource-constrained environments.

5.3 Future Work

Several directions may extend and deepen the current work.

For EVI, incorporating adaptive kernels or hierarchical priors could improve flexibility and posterior accuracy. Time-dependent particle evolution and connections to Wasserstein gradient flows also present interesting avenues.

For active learning, one direction is to integrate manifold regularization or topological priors directly into the autoencoder training. Another is to couple acquisition and representation learning in an end-to-end framework that co-optimizes uncertainty and geometry.

More broadly, combining the two contributions into a unified, dynamically trained surrogate system—one that evolves both its inference scheme and acquisition strategy based on feedback—could unlock powerful new applications in scientific computing, PDE modeling, and autonomous experimentation.

5.4 Closing Remarks

This dissertation contributes to the broader effort of making probabilistic modeling methods more adaptive, scalable, and structure-aware. By revisiting GP regression through the lenses of energy and geometry, we provide new formulations that move beyond fixed approximations and point toward a more dynamic, data-sensitive future for surrogate modeling.

APPENDIX A

DERIVATIONS OF PROPOSITIONS

A.1 Derivation of Proposition 2.1.1

Proof. In this proof we only consider the zero mean case. The joint distribution of $\boldsymbol{Y}_n$ and $y(\boldsymbol{x})$, where $\boldsymbol{x}$ is the query point. Note that by definition, we have $\text{Cov}\left[y(\boldsymbol{x}), y_i\right] = \tau^2 k(\boldsymbol{x}, \boldsymbol{x}_i)$ and $\text{Var}\left[\boldsymbol{x}\right] = \tau^2(k(\boldsymbol{x}, \boldsymbol{x}) + \eta)$, then we have

$$\begin{bmatrix} \boldsymbol{Y}_n \\ y(\boldsymbol{x}) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \tau^2 \begin{bmatrix} \boldsymbol{K}_n + \eta \boldsymbol{I}_n & \boldsymbol{k}(\boldsymbol{x}) \\ \boldsymbol{k}(\boldsymbol{x})^\top & k(\boldsymbol{x}, \boldsymbol{x}) + \eta \end{bmatrix} \right)$$

where $\boldsymbol{k}(\boldsymbol{x}) := k(\boldsymbol{x}, \boldsymbol{X}_n) \in \mathbb{R}^n$, $\boldsymbol{K}_n := [k(\boldsymbol{x}_i, \boldsymbol{x}_j)]_{i,j=1}^n$.

Based on the conditional formula, we have

$$y(\boldsymbol{x}) \,|\, \boldsymbol{Y}_n \sim \mathcal{N}(\hat{\mu}(\boldsymbol{x}), \sigma^2(\boldsymbol{x}))$$

where

$$\begin{aligned} \hat{\mu}(\boldsymbol{x}) &= \tau^2 \boldsymbol{k}(\boldsymbol{x})^\top \left(\tau^2(\boldsymbol{K}_n + \eta \boldsymbol{I}_n)\right)^{-1} \boldsymbol{Y}_n \\ &= \boldsymbol{k}(\boldsymbol{x})^\top (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1} \boldsymbol{Y}_n \\ \sigma^2(\boldsymbol{x}) &= \tau^2(1 + \eta) - \tau^4 \boldsymbol{k}(\boldsymbol{x})^\top (\tau^2(\boldsymbol{K}_n + \eta \boldsymbol{I}_n))^{-1} \boldsymbol{k}(\boldsymbol{x}) \\ &= \tau^2 \left(1 + \eta - \boldsymbol{k}(\boldsymbol{x})^\top (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1} \boldsymbol{k}(\boldsymbol{x})\right) \end{aligned}$$

□

A.2 Derivation of Proposition 3.1.1

Proof. The marginal posterior distribution of $(\boldsymbol{\omega}, \tau^2, \eta)$ can be obtained by integration as follows.

$$p(\boldsymbol{\omega}, \tau^2, \eta | \boldsymbol{y}_n) \propto \int p(\boldsymbol{\beta}) p(\boldsymbol{y}_n | \boldsymbol{\theta}) p(\boldsymbol{\omega}) p(\tau^2) p(\eta) \, \mathrm{d}\boldsymbol{\beta}$$

$$
\begin{aligned}
&\propto \int \Bigg\{ \exp\left[-\frac{1}{2} (\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}_n)^\top \boldsymbol{\Sigma}_{\boldsymbol{\beta}|n}^{-1} (\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}_n) \right] \det(\boldsymbol{\Sigma}_{\boldsymbol{\beta}|n})^{-1/2} \\
&\quad \det(\boldsymbol{\Sigma}_{\boldsymbol{\beta}|n})^{1/2} \exp\left[-\frac{1}{2} \hat{\boldsymbol{\beta}}_n^\top \boldsymbol{\Sigma}_{\boldsymbol{\beta}|n}^{-1} \hat{\boldsymbol{\beta}}_n - \frac{1}{2\tau^2} \boldsymbol{y}_n^\top (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1} \boldsymbol{y}_n \right] \\
&\quad (\tau^2)^{-n/2} \det(\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1/2} p(\tau^2) p(\boldsymbol{\omega}) p(\eta) \Big\} \, \mathrm{d}\boldsymbol{\beta} \\
&\propto \det(\boldsymbol{\Sigma}_{\boldsymbol{\beta}|n})^{1/2} \exp\left[-\frac{1}{2} \hat{\boldsymbol{\beta}}_n^\top \boldsymbol{\Sigma}_{\boldsymbol{\beta}|n}^{-1} \hat{\boldsymbol{\beta}}_n - \frac{1}{2\tau^2} \boldsymbol{y}_n^\top (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1} \boldsymbol{y}_n \right] \\
&\quad \times (\tau^2)^{-n/2} \det(\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1/2} p(\tau^2) p(\boldsymbol{\omega}) p(\eta).
\end{aligned}
$$

□

A.3 Derivation of Proposition 3.1.2

Based on the non-informative prior,

$$
\begin{aligned}
\boldsymbol{\Sigma}_{\boldsymbol{\beta}|n} &= \tau^2 \left[\boldsymbol{G}^\top (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1} \boldsymbol{G} \right]^{-1} \\
\hat{\boldsymbol{\beta}}_n &= \left[\boldsymbol{G}^\top (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1} \boldsymbol{G} \right]^{-1} \left[\boldsymbol{G}^\top (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1} \right] \boldsymbol{y}_n.
\end{aligned}
$$

Define

$$
\begin{aligned}
s_n^2 &= \tau^{-2} \hat{\boldsymbol{\beta}}_n^\top \boldsymbol{\Sigma}_{\boldsymbol{\beta}|n}^{-1} \hat{\boldsymbol{\beta}}_n + \boldsymbol{y}_n^\top (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1} \boldsymbol{y}_n \\
&= \boldsymbol{y}_n^\top (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1} \boldsymbol{G} \left(\boldsymbol{G}^\top (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1} \boldsymbol{G} \right)^{-1} \boldsymbol{G}^\top (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1} \boldsymbol{y}_n \\
&\quad + \boldsymbol{y}_n^\top (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1} \boldsymbol{y}_n \\
&= \boldsymbol{y}_n^\top (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1} \\
&\quad \Big[\boldsymbol{G} \left(\boldsymbol{G}^\top (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1} \boldsymbol{G} \right)^{-1} \boldsymbol{G}^\top + (\boldsymbol{K}_n + \eta \boldsymbol{I}_n) \Big] (\boldsymbol{K}_n + \eta \boldsymbol{I}_n)^{-1} \boldsymbol{y}_n.
\end{aligned}
$$

Then $p(\boldsymbol{\omega}, \tau^2, \eta | \boldsymbol{y}_n)$ is

$$
p(\boldsymbol{\omega}, \tau^2, \eta | \boldsymbol{y}_n) \propto (\tau^2)^{p/2} \exp\left(-\frac{s_n^2}{2\tau^2} \right) (\tau^2)^{-n/2} p(\tau^2)
$$

$$
\begin{aligned}
&\det(\boldsymbol{G}^\top(\boldsymbol{K}_n+\eta\boldsymbol{I}_n)^{-1}\boldsymbol{G})^{-1/2}\det(\boldsymbol{K}_n+\eta\boldsymbol{I}_n)^{-1/2}p(\boldsymbol{\omega})p(\eta)\\
\propto\ &(\tau^2)^{-\left[\frac{1}{2}(df_{\tau^2}+n-p)+1\right]}\exp\left(-\frac{1+s_n^2}{2\tau^2}\right)\\
&\det(\boldsymbol{G}^\top(\boldsymbol{K}_n+\eta\boldsymbol{I}_n)^{-1}\boldsymbol{G})^{-1/2}\det(\boldsymbol{K}_n+\eta\boldsymbol{I}_n)^{-1/2}p(\boldsymbol{\omega})p(\eta).
\end{aligned}
$$

because of the conditional conjugacy, we can see the conditional posterior distribution of $\tau^2|\boldsymbol{\omega},\eta,\boldsymbol{y}_n$ is

$$
\tau^2|\boldsymbol{\omega},\eta,\boldsymbol{y}_n \sim \text{Scaled Inverse-}\chi^2(df_{\tau^2}+n-p,\hat{\tau}^2), \tag{A.3.1}
$$

where

$$
\hat{\tau}^2=\frac{1+s_n^2}{df_{\tau^2}+n-p}.
$$

Based on it, we can obtain the integration

$$
p(\boldsymbol{\omega},\eta|\boldsymbol{y}_n)\propto(\hat{\tau}^2)^{-\frac{1}{2}(df_{\tau^2}+n-p)}\det(\boldsymbol{G}^\top(\boldsymbol{K}_n+\eta\boldsymbol{I}_n)^{-1}\boldsymbol{G})^{-1/2}\det(\boldsymbol{K}_n+\eta\boldsymbol{I}_n)^{-1/2}p(\boldsymbol{\omega})p(\eta).
$$

BIBLIOGRAPHY

[1] L. Kang, Y. Cheng, Y. Wang, and C. Liu, "Energetic variational gaussian process regression," in *Proceedings of the 2024 Winter Simulation Conference (WSC)*, pp. 3542–3553, Dec. 2024. © 2024 IEEE. Reprinted, with permission, from Lulu Kang et al., "Energetic Variational Gaussian Process Regression," 2024 Winter Simulation Conference (WSC), IEEE.

[2] Y. Cheng, L. Kang, Y. Wang, and C. Liu, "Active learning for manifold gaussian process regression," in *Proceedings of the 2025 Winter Simulation Conference (WSC)*, 2025. © 2025 IEEE. Reprinted, with permission, from Yuanxing Cheng et al., "Active Learning for Manifold Gaussian Process Regression," Proc. 2025 Winter Simulation Conference (WSC), IEEE.

[3] B. Sousedík *et al.*, "On surrogate learning for linear stability assessment of navier-stokes equations with stochastic viscosity," *Applications of Mathematics*, 2022. online first.

[4] H. Choi and P. Moin, "Large eddy simulation of turbulent flows: Achievements and challenges," *Journal of Computational Physics*, vol. 230, no. 10, pp. 4121–4140, 2011.

[5] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning, MIT Press, 2006.

[6] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational Inference: A Review for Statisticians," *Journal of the American Statistical Association*, vol. 112, pp. 859–877, Apr. 2017.

[7] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient Global Optimization of Expensive Black-Box Functions," *Journal of Global Optimization*, vol. 13, pp. 455–492, Dec. 1998.

[8] B. Fröhlich, E. Rodner, M. Kemmler, and J. Denzler, "Efficient Gaussian process classification using random decision forests," *Pattern Recognition and Image Analysis*, vol. 21, pp. 184–187, June 2011.

[9] T. J. Santner, B. J. Williams, and W. I. Notz, *The Design and Analysis of Computer Experiments*. Springer Series in Statistics, Springer New York, 2018.

[10] R. B. Gramacy, *Surrogates: Gaussian Process Modeling, Design, and Optimization for the Applied Sciences*. Chapman and Hall/CRC, 2020.

[11] M. L. Stein, *Interpolation of Spatial Data: Some Theory for Kriging*. Springer, 1999.

[12] C.-Y. Peng and C. F. J. Wu, "On the Choice of Nugget in Kriging Modeling for Deterministic Computer Experiments," *Journal of Computational and Graphical Statistics*, Jan. 2014.

[13] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," Dec. 2019.

[14] R. Bellman and R. Kalaba, "A Mathematical Theory of Adaptive Control Processes," *Proceedings of the National Academy of Sciences*, vol. 45, pp. 1288–1290, Aug. 1959.

[15] M. Titsias, "Variational Learning of Inducing Variables in Sparse Gaussian Processes," in *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, pp. 567–574, PMLR, Apr. 2009.

[16] D. Tran, R. Ranganath, and D. M. Blei, "The Variational Gaussian Process," *arXiv preprint arXiv:1511.06499*, Apr. 2016.

[17] H. Liu, Y.-S. Ong, X. Shen, and J. Cai, "When Gaussian Process Meets Big Data: A Review of Scalable GPs," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, pp. 4405–4423, Nov. 2020.

[18] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, "When Is "Nearest Neighbor" Meaningful?," in *Database Theory — ICDT'99* (C. Beeri and P. Buneman, eds.), (Berlin, Heidelberg), pp. 217–235, Springer, 1999.

[19] M. Meilă and H. Zhang, "Manifold Learning: What, How, and Why," *Annual Review of Statistics and Its Application*, vol. 11, pp. 393–417, Apr. 2024.

[20] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A Global Geometric Framework for Nonlinear Dimensionality Reduction," *Science*, vol. 290, pp. 2319–2323, Dec. 2000.

[21] S. T. Roweis and L. K. Saul, "Nonlinear Dimensionality Reduction by Locally Linear Embedding," *Science*, vol. 290, pp. 2323–2326, Dec. 2000.

[22] M. Belkin and P. Niyogi, "Laplacian Eigenmaps for Dimensionality Reduction and Data Representation," *Neural Computation*, vol. 15, pp. 1373–1396, June 2003.

[23] R. Calandra, J. Peters, C. E. Rasmussen, and M. P. Deisenroth, "Manifold Gaussian Processes for Regression," *arXiv preprint arXiv:1402.5876*, Apr. 2016.

[24] A. Mallasto and A. Feragen, "Wrapped Gaussian Process Regression on Riemannian Manifolds," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (Salt Lake City, UT), pp. 5580–5588, IEEE, June 2018.

[25] B. Fichera, V. Borovitskiy, A. Krause, and A. Billard, "Implicit manifold Gaussian process regression," in *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, (Red Hook, NY, USA), pp. 67701–67720, Curran Associates Inc., Dec. 2023.

[26] D. A. Cohn, "Neural Network Exploration Using Optimal Experiment Design," *Neural Networks*, vol. 9, pp. 1071–1083, Aug. 1996.

[27] S. Seo, M. Wallat, T. Graepel, and K. Obermayer, "Gaussian process regression: Active data selection and test point rejection," in *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*, vol. 3, pp. 241–246 vol.3, July 2000.

[28] P. I. Frazier, "A Tutorial on Bayesian Optimization," *arXiv preprint arXiv:1807.02811*, July 2018.

[29] R. B. Gramacy and H. Lian, "Gaussian Process Single-Index Models as Emulators for Computer Experiments," *Technometrics*, vol. 54, pp. 30–41, Feb. 2012.

[30] Y. Hu, R. B. Gramacy, and H. Lian, "Bayesian quantile regression for single-index models," *Statistics and Computing*, vol. 23, pp. 437–454, July 2013.

[31] M. Hamada and C. F. J. Wu, "Analysis of Designed Experiments with Complex Aliasing," *Journal of Quality Technology*, vol. 24, pp. 130–137, July 1992.

[32] C.-F. Wu and M. Hamada, *Experiments: Planning, Analysis, and Optimization*. Wiley Series in Probability and Statistics, Wiley, 3rd ed ed., 2021.

[33] V. R. Joseph, "A Bayesian Approach to the Design and Analysis of Fractionated Experiments," *Technometrics*, vol. 48, pp. 219–229, May 2006.

[34] L. Kang and V. R. Joseph, "Bayesian Optimal Single Arrays for Robust Parameter Design," *Technometrics*, vol. 51, pp. 250–261, Aug. 2009.

[35] M. Ai, L. Kang, and V. R. Joseph, "Bayesian optimal blocking of factorial designs," *Journal of Statistical Planning and Inference*, vol. 139, pp. 3319–3328, Sept. 2009.

[36] L. Kang, X. Kang, X. Deng, and R. Jin, "A Bayesian hierarchical model for quantitative and qualitative responses," *Journal of Quality Technology*, vol. 50, pp. 290–308, July 2018.

[37] L. Kang and X. Huang, "Bayesian A-Optimal Design of Experiment with Quantitative and Qualitative Responses," *Journal of Statistical Theory and Practice*, vol. 13, p. 64, Oct. 2019.

[38] X. Kang, S. Ranganathan, L. Kang, J. Gohlke, and X. Deng, "Bayesian auxiliary variable model for birth records data with qualitative and quantitative responses," *Journal of Statistical Computation and Simulation*, vol. 91, no. 16, pp. 3283–3303, 2021.

[39] L. Kang, X. Deng, and R. Jin, "Bayesian D-Optimal Design of Experiments with Quantitative and Qualitative Responses," *The New England Journal of Statistics in Data Science*, vol. 1, pp. 371–385, Apr. 2023.

[40] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods*. Springer Texts in Statistics, Springer, 2nd ed ed., 2004.

[41] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin, "Bayesian Data Analysis Third edition (with errors fixed as of 20 February 2025)," 2025.

[42] P. Bickel, *Bayesian Learning for Neural Networks*. No. v.118 in Lecture Notes in Statistics Ser, Springer New York, 1st ed ed., 1996.

[43] G. O. Roberts and J. S. Rosenthal, "Optimal Scaling of Discrete Approximations to Langevin Diffusions," *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, vol. 60, no. 1, pp. 255–268, 1998.

[44] C.-A. Cheng and B. Boots, "Variational Inference for Gaussian Process Models with Linear Complexity," in *Advances in Neural Information Processing Systems*, vol. 30, Curran Associates, Inc., 2017.

[45] G. Wynne and V. Wild, "Variational Gaussian Processes: A Functional Analysis View," in *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, pp. 4955–4971, PMLR, May 2022.

[46] Y.-c. Fung, *A First Course in Continuum Mechanics*. San Diego: Elsevier Academic Press, 4th ed., 2001.

[47] H. B. Callen, *Thermodynamics and an Introduction to Thermostatistics*. John Wiley & Sons, 2nd ed., 1985.

[48] C. Liu, "Variational approaches for complex fluids," *Entropy*, vol. 24, no. 5, p. 721, 2022.

[49] I. M. Gelfand and S. V. Fomin, *Calculus of Variations*. Mineola, NY: Dover Publications, 2000.

[50] L. D. Landau and E. M. Lifshitz, *Mechanics*, vol. 1 of *Course of Theoretical Physics*. Oxford: Pergamon Press, 3rd ed., 1976.

[51] L. Onsager, "Reciprocal Relations in Irreversible Processes. I.," *Physical Review*, vol. 37, pp. 405–426, Feb. 1931.

[52] R. Jordan, D. Kinderlehrer, and F. Otto, "The Variational Formulation of the Fokker–Planck Equation," *SIAM Journal on Mathematical Analysis*, vol. 29, pp. 1–17, Jan. 1998.

[53] C. Villani, *Optimal Transport: Old and New*, vol. 338 of *Grundlehren der mathematischen Wissenschaften*. Springer, 2009.

[54] Y. Wang, J. Chen, C. Liu, and L. Kang, "Particle-based Energetic Variational Inference," *Statistics and Computing*, vol. 31, p. 34, May 2021.

[55] E. Hairer, S. P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations I: Nonstiff Problems*, vol. 8 of *Springer Series in Computational Mathematics*. Springer, 2nd ed., 1991.

[56] R. T. Rockafellar, "Monotone Operators and the Proximal Point Algorithm," *SIAM Journal on Control and Optimization*, vol. 14, pp. 877–898, Aug. 1976.

[57] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical Programming*, vol. 45, no. 1–3, pp. 503–528, 1989.

[58] B. MacDonald, P. Ranjan, and H. Chipman, "GPfit: An R Package for Fitting a Gaussian Process Model to Deterministic Simulator Outputs," *Journal of Statistical Software*, vol. 64, pp. 1–23, Apr. 2015.

[59] R. B. Gramacy and H. K. Lee, "Bayesian treed gaussian process models with an application to computer modeling," *Journal of the American Statistical Association*, vol. 103, no. 483, pp. 1119–1130, 2008.

[60] R. B. Gramacy and D. W. Apley, "Local Gaussian Process Approximation for Large Computer Experiments," *Journal of Computational and Graphical Statistics*, vol. 24, pp. 561–578, Apr. 2015.

[61] R. B. Gramacy, "**laGP** : Large-Scale Spatial Modeling via Local Approximate Gaussian Processes in *R*," *Journal of Statistical Software*, vol. 72, no. 1, 2016.

[62] S. Surjanovic and D. Bingham, "Virtual Library of Simulation Experiments: Test Functions and Datasets." http://www.sfu.ca/~ssurjano/, 2023.

[63] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, pp. 8024–8035, Red Hook, NY, USA: Curran Associates, Inc., 2019.

[64] R. Carnell, *lhs: Latin Hypercube Samples*, 2022. R package version 1.1.5.

[65] H. Mohammadi, P. Challenor, M. Goodfellow, and D. Williamson, "Emulating computer models with step-discontinuous outputs using gaussian processes," *arXiv preprint arXiv:1903.02071*, 2020.

[66] T. J. Santner, B. J. Williams, W. I. Notz, and B. J. Williams, *The Design and Analysis of Computer Experiments*, vol. 1. New York, NY, USA: Springer, 2nd ed., 2003.

[67] A. Damianou and N. D. Lawrence, "Deep Gaussian processes," in *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics* (C. M. Carvalho and P. Ravikumar, eds.), vol. 31 of *Proceedings of Machine Learning Research*, (Scottsdale, Arizona, USA), pp. 207–215, PMLR, 29 Apr–01 May 2013.

[68] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[69] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[70] S. R. Dubey, S. K. Singh, and B. B. Chaudhuri, "Activation functions in deep learning: A comprehensive survey and benchmark," *Neurocomputing*, vol. 503, pp. 92–108, Sept. 2022.

[71] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," Mar. 2015.

[72] D. Cohn, Z. Ghahramani, and M. Jordan, "Active Learning with Statistical Models," in *Advances in Neural Information Processing Systems*, vol. 7, MIT Press, 1994.

[73] A. Sauer, R. B. Gramacy, and D. Higdon, "Active Learning for Deep Gaussian Process Surrogates," *Technometrics*, vol. 65, pp. 4–18, Jan. 2023.

[74] A. Sauer, R. B. Gramacy, and D. H. and, "Active learning for deep gaussian process surrogates," *Technometrics*, vol. 65, no. 1, pp. 4–18, 2023.

[75] J. Nocedal and S. J. Wright, *Numerical Optimization*. Springer Series in Operations Research, Springer, 2nd ed ed., 2006.

[76] L. Kang, Y. Cheng, Y. Wang, and C. Liu, "Energetic variational gaussian process regression for computer experiments," *arXiv preprint arXiv:2401.00395*, 2023.