Joint and Condition Modeling for High-Dimensional Aleatoric and Epistemic Uncertainty Quantification

Chunlin Ji, CHUNLIN.JI@KUANG-CHI.ORG
Kuang-Chi Institute of Advanced Technology
Shenzhen 518000, China

Dingwei Gong, GONGDW@MAIL2.SYSU.EDU.CN
School of Electronics and Information Technology,
Sun Yat-Sen University
Kuang-Chi Institute of Advanced Technology
Shenzhen 518000, China

Yongxiang Li, YONGXIANGLI@SJTU.EDU.CN
Department of Industrial Engineering and Management
Shanghai Jiao Tong University
Shanghai 200240, China

Meiying Zhang, ZHANGMY@SUSTECH.EDU.CN
Research Institute of Trustworthy Autonomous Systems
Southern University of Science and Technology
Shenzhen 518055, China

Terry Ma, TERRYMA@CS.CMU.EDU
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, USA

Jun S. Liu, JLIU@STAT.HARVARD.EDU
Department of Statistics
Harvard University
Cambridge, MA 02138, USA

Editor:

Abstract

By providing insights into the complex landscape of systems, uncertainty quantification (UQ) is a critical aspect of modeling and prediction, making it valuable for trustworthy predictions and informed decision making, such as Bayesian optimization. In the context of nonparametric regression, we propose a unified approach to jointly quantify aleatoric and epistemic uncertainties by modeling input and output (response) variables using a Gaussian mixture model and inferring conditional distributions for given inputs. Through a linear dimension reduction model with a projection matrix on the Stiefel manifold, input variables are transformed into a meaningful low-dimensional representation, addressing the challenge of modeling high-dimensional uncertainties. We employ a Riemannian Stochastic Gradient Descent (SGD) algorithm to optimize the projection matrix. Our approach jointly quantifies aleatoric and epistemic uncertainty in a practical, scalable, and computationally efficient manner. The proposed method stands out for its excellent expressiveness to efficiently model both aleatoric and epistemic uncertainty in high-dimensional space without requiring specified model structures. It offers an implicit regression model with limited hyperparameters, enhancing usability and reducing complexity. Numerical examples demonstrate its effectiveness in achieving state-of-the-art results in regression, quantile interval estimation, and Bayesian optimization.

Keywords: Uncertainty quantification, Gaussian mixture model, Stiefel manifold, Quantile estimation, Bayesian optimization

1. Introduction

Uncertainty quantification (UQ) is a discipline that deals with the evaluation and characterization of uncertainty in various types of models, simulations, and systems (Ghanem et al., 2017; Tagasovska and Lopez-Paz, 2019). UQ can provide essential insights into the intricate landscape of a model or system from small amounts of data, fostering deeper understanding, providing convincing predictions, and enabling improved decision making (Yan et al., 2023). By enabling informed decisions, UQ has impacted a wide range of areas in science and engineering, such as active learning (Garcia et al., 2023), Bayesian optimization (Wang et al., 2023), and reinforcement learning (Shakya et al., 2023). Thus, UQ has attracted increasing interest from both the machine learning and statistics communities.

In practical applications, there are two types of uncertainty (Hüllermeier and Waegeman, 2021; Valdenegro-Toro and Mori, 2022): aleatoric and epistemic uncertainty. Aleatoric uncertainty captures the noise inherent in the observations, such as sensor noise or motion noise. Aleatoric uncertainty can be further categorized into homoscedastic and heteroscedastic uncertainty (Tao Yu, 2020). Heteroscedastic uncertainty depends on the inputs, with some inputs potentially having more noisy outputs than others (Alex Kendall, 2017). Conversely, epistemic uncertainty accounts for the uncertainty in our knowledge of the underlying model, which can be explained away with sufficient data, and is often referred to as model uncertainty (Detlefsen et al., 2019). Epistemic uncertainty is the reducible part of the total uncertainty, while aleatoric uncertainty is the irreducible part (Hüllermeier and Waegeman, 2021).

Significant effort has been directed towards estimating uncertainty in regression and classification tasks. Gaussian process (GP) models (Rasmussen and Williams, 2006; Wang et al., 2024) have emerged as powerful tools for quantifying uncertainty, particularly epistemic uncertainty. However, standard GPs often struggle to effectively model heteroscedastic uncertainty, which is commonly encountered when estimating aleatoric uncertainty. This limitation has prompted the development of more advanced GP variants, such as dual GP (Goldberg et al., 1997; Lázaro-Gredilla and Titsias, 2011) and deep GPs (DGPs, Damianou and Lawrence 2013). The dual GP approach uses another GP to model the logarithm of the observation noise variance, whereas DGPs utilize hierarchical kernel functions to accommodate heteroscedasticity. However, fitting these models presents considerable computational challenges (Sauer et al., 2023).

Alternatively, various neural network (NN) architectures supporting uncertainty quantification have been proposed. Two primary strategies exist for incorporating uncertainty into NN models. The first involves explicitly modeling the output variance, often by adding a dedicated network branch, demonstrating competence in handling heteroscedastic uncertainty. Examples include the mixture density networks (MDNs, Bishop 1994), the mean-variance neural networks (MVNN, Nix and Weigend 1994), and methods employing combined variance estimation techniques (Detlefsen et al., 2019). The second strategy adopts a Bayesian framework. Bayesian neural networks (BNNs, Mackay 1992; Yacoby et al. 2022) provide uncertainty estimates by characterizing the posterior distribution over network parameters. This is typically achieved using Monte Carlo methods like Hamiltonian Monte Carlo (Izmailov et al. 2021), or approximate inference techniques, such as variational inference (VI, Hoffman et al. 2013; Rezende et al. 2014) and expectation propagation (EP, Hernández-Lobato and Adams 2015). From a frequentist perspective, deep ensembles (Lakshminarayanan et al., 2017) offer another approach, enabling uncertainty quantification by aggregating predictions from an ensemble of NNs trained with different initializations. While NN-based methods provide substantial model flexibility, obtaining reliable uncertainty estimates remains a challenge. It has been reported that uncertainty quantification using current BNNs, particularly those relying on approximate inference, often suffers from underestimation (Murphy, 2012; Blei et al., 2017; Giordano et al., 2018). This issue can be especially pronounced when dealing with heteroscedastic uncertainty.

Despite recent advances, the simultaneous quantification of high-dimensional aleatoric and epistemic uncertainty remains a significant challenge. This difficulty stems primarily from two key issues: 1) Due to the coexistence of aleatoric and epistemic uncertainty, the joint distribution between input and output variables may not be effectively modeled by the simple-distribution assumptions (e.g., unimodal Gaussian distribution) widely used in GP and BNN. 2) Effective quantification of high-dimensional uncertainty is inherently difficult, particularly in applications constrained by limited data availability.

To address these challenges, this work introduces a joint and condition modeling (JCM) approach designed for computationally scalable and statistically efficient quantification of both types of uncertainty. We utilize a Gaussian mixture model (GMM) to capture the joint distribution of the input and response variables. The inherent flexibility of GMMs, capable of approximating arbitrary densities given sufficient components, allows for a more expressive representation of uncertainty compared to models relying on simpler distributions. By modeling the joint density of inputs (probably reduced-dimensional features) and outputs, we can subsequently derive the conditional distribution of the response given any input in closed form, providing a direct pathway to uncertainty quantification.

A core component of the proposed JCM approach is the integration of dimensionality reduction. We propose a linear dimension reduction model to identify a low-dimensional representation of the input variables that is most relevant for predicting the output. This reduction empowers JCM to effectively quantify high-dimensional uncertainty even in data-limited scenarios. To ensure this low-dimensional projection is meaningful and avoids trivial solutions, we constrain the projection matrix to lie on a Stiefel manifold (Absil et al., 2009) and optimize it specifically to capture relevant predictive information.

The proposed algorithm proceeds iteratively through two steps. First, given the current GMM parameters, a Riemannian stochastic gradient descent (SGD) algorithm (Bonnabel, 2013) optimizes the projection matrix on the Stiefel manifold. Second, using the features obtained from this projection, the GMM parameters are updated using a standard expectation-maximization (EM) algorithm (Dempster, 1977). The resulting GMM, representing the joint distribution, allows for the analytical derivation of the conditional (posterior) distribution of the response. Within a single model, the proposed method can handle both typical epistemic uncertainty estimation tasks, such as quantile interval estimation, and aleatoric uncertainty estimation tasks, such as active learning and Bayesian optimization. Numerical experiments demonstrate that the proposed method achieves competitive, state-of-the-art performance in regression, quantile interval estimation, active learning, and Bayesian optimization tasks.

This paper introduces a JCM approach for high-dimensional aleatoric and epistemic uncertainty quantification with three distinctive features. First, the proposed JCM approach employs a mixture of normal distributions, which has a more expressive model structure, to quantify the uncertainty, rather than using a single normal distribution in conventional methods such as GP and BNN. Second, the proposed method is computationally scalable compared to GP regression methods, which often have a prohibitive time complexity. Third, unlike neural network methods that often require numerical methods for uncertainty quantification, the proposed method provides a closed-form expression for the posterior distribution, resulting in an accurate uncertainty quantification.

The remainder of this paper is structured as follows. Section 2 reviews the relevant literature on high-dimensional uncertainty quantification, discussing existing methodologies and their limitations. Section 3 introduces the proposed JCM, detailing its framework, methodology, and learning algorithm. Section 4 develops JCM-based uncertainty quantification techniques, including predictive mean, covariance, quantile estimation, and entropy bounds, and outlines their application to active learning and Bayesian optimization. Section 5 presents a comprehensive evaluation of JCM's performance through several numerical examples. Finally, Section 6 concludes the paper and discusses potential directions for future research.

2. Brief Literature Review

Uncertainty quantification in regression is critical because it enables a nuanced understanding of the uncertainties inherent in model predictions. This section initiates a thorough exploration of uncertainty quantification in the context of regression problems. Readers with a keen interest in uncertainty quantification are warmly encouraged to delve further into these explorations (Ghanem et al., 2017; Hüllermeier and Waegeman, 2021; Valdenegro-Toro and Mori, 2022).

Consider an open domain $\Omega \subset \mathbb{R}^d$. The input is represented by a vector of d variables (or covariates), denoted as $\boldsymbol{x} = [x_1, \ldots, x_d]^T \in \Omega$. Let y_i be the observed response for the input $\boldsymbol{x}_i$, for $i = 1, ..., N$. The collection of all inputs forms the matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N]^T \in \mathbb{R}^{N \times d}$, and the associated responses form the vector $\boldsymbol{y} = [y_1, \cdots, y_N]^T$. The complete dataset is then denoted as $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{y}\}$. To characterize the uncertainty in predicting y given $\boldsymbol{x}$, we are interested in obtaining quantities such as the conditional probability distribution $\mathcal{P}(y|\boldsymbol{x})$, the conditional variance $\mathrm{Var}(y|\boldsymbol{x}) = \mathbb{E}\left[(y - \mathbb{E}[y|\boldsymbol{x}])^2 \middle| \boldsymbol{x}\right]$, and the conditional entropy $H(y|\boldsymbol{x}) = -\mathbb{E}\left[\log \mathcal{P}(y|\boldsymbol{x}) \middle| \boldsymbol{x}\right]$.

A general regression model can be defined by the relationship $y \sim \eta_{\boldsymbol{\phi}}(f_{\boldsymbol{\theta}}(\boldsymbol{x}))$. Here, $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ is a feature extraction function parameterized by $\boldsymbol{\theta}$, which maps the input $\boldsymbol{x}$ into a (potentially lower-dimensional) representation. The function $\eta_{\boldsymbol{\phi}}$, defined as the observation model, characterizes the probability distribution of the response y given the extracted features $f_{\boldsymbol{\theta}}(\boldsymbol{x})$.

The observation model $\eta_{\boldsymbol{\phi}}$ is often characterized by a link function that maps the feature $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ to the parameters (e.g., the mean and variance) of a distribution of y. For instance, if the observation model $\eta_{\boldsymbol{\phi}}$ corresponds to a normal (or Gaussian) distribution and employs an identity link function, the regression model is commonly written as $y_i = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) + \epsilon_i$. In this specific case, $f_{\boldsymbol{\theta}}$ still represents the feature extraction function parameterized by $\boldsymbol{\theta}$, and ϵ_i denotes Gaussian noises (i.e., $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$, where σ^2 would be part of $\boldsymbol{\phi}$).

The function $f_{\boldsymbol{\theta}}$ can take various forms, such as a linear model employing basis functions, a stochastic process (like a GP), or a neural network. For example, a single-hidden-layer neural network uses the following function:

$$y_i = u\left(\omega_{i0}^{(2)} + \sum_{j=1} \omega_{ij}^{(2)} v\left(\sum_{k=1}^{d} \omega_{jk}^{(1)} x_{i,k} + \omega_{j0}^{(1)}\right)\right),$$

where $u(\cdot)$ and $v(\cdot)$ are activation functions, and $\boldsymbol{\theta} \equiv \boldsymbol{\omega}$ is a vector of the network weights $\omega_{ij}^{(1)}$ and $\omega_{ij}^{(2)}$. Modern neural networks, such as deep neural networks, use many such hidden layers.

To enable uncertainty quantification, BNNs (Mackay, 1992) treat the weights $\boldsymbol{\omega}$ as random variables and inferred according to the Bayes theorem

$$\mathcal{P}(\boldsymbol{\omega}|\boldsymbol{y}, \boldsymbol{X}) \propto \prod_{i=1}^{N} \mathcal{P}(y_i|\boldsymbol{x}_i, \boldsymbol{\omega}) \mathcal{P}(\boldsymbol{\omega}),$$

where the prior $\mathcal{P}(\boldsymbol{\omega})$ and likelihood $\mathcal{P}(y_i|\boldsymbol{x}_i, \boldsymbol{\omega})$ are typically assumed to be normal. The predictive/conditional distribution of y corresponding to any input $\boldsymbol{x}$ is

$$\mathcal{P}(y|\boldsymbol{x}) = \int \mathcal{P}(y|\boldsymbol{x}, \boldsymbol{\omega}) \mathcal{P}(\boldsymbol{\omega}|\boldsymbol{y}, \boldsymbol{X}) \, d\boldsymbol{\omega}.$$

Modern BNNs (Gal and Ghahramani, 2016; Ghosh et al., 2019; Goan and Fookes, 2020) primarily utilize approximate inference methods, notably Variational Inference (VI) (Graves, 2011; Hoffman et al., 2013; Rezende et al., 2014), to determine a variational distribution $q_{\boldsymbol{\psi}}(\boldsymbol{\omega})$. This distribution provides a tractable form parameterized by $\boldsymbol{\psi}$ to approximate the true posterior $\mathcal{P}(\boldsymbol{\omega}|\boldsymbol{y}, \boldsymbol{X})$. The optimal $\boldsymbol{\psi}$ is found by maximizing the VI objective function:

$$\mathcal{L}(\boldsymbol{\psi}) = \mathbb{E}_{\boldsymbol{\omega} \sim q_{\boldsymbol{\psi}}(\boldsymbol{\omega})}\left[\log \mathcal{P}(y|\boldsymbol{X}, \boldsymbol{\omega})\right] - \mathbb{E}_{\boldsymbol{\omega} \sim q_{\boldsymbol{\psi}}(\boldsymbol{\omega})}\left[\log q_{\boldsymbol{\psi}}(\boldsymbol{\omega}) - \log \mathcal{P}(\boldsymbol{\omega})\right],$$

known as the evidence lower bound (ELBO, Barber and Bishop 1998). However, VI commonly employs simple distributional families (e.g., a multivariate Gaussian family) for the variational approximation. This choice often leads to posterior approximations or variance estimates that inadequately capture the true uncertainty. This limitation significantly impacts the application of BNNs in scenarios that require accurate uncertainty estimation, such as BO.

Hamiltonian Monte Carlo (HMC, Neal 2012; Izmailov et al. 2021 offers an alternative to VI for approximating the posterior distribution over the weights $\boldsymbol{\omega}$ in BNN. In HMC, the posterior is explored by simulating a Hamiltonian system via a Hamiltonian function, leveraging gradients to simulate trajectories in weight space for efficient posterior sampling. HMC uses gradients to avoid the slow random walk behavior of traditional Markov chain Monte Carlo (MCMC) methods, thereby providing high-fidelity estimates of uncertainty. However, HMC requires multiple forward and backward passes through the network to compute gradients, making it computationally expensive for large-scale datasets or deep models.

However, a simple and explicit distributional assumption (typically i.i.d. normal distributions) is commonly imposed on $\mathcal{P}\left(y|\boldsymbol{x},\boldsymbol{\omega}\right)$ to reduce computational complexity. Even with HMC, such an assumption limits the uncertainty expressiveness of BNNs, potentially rendering them insufficiently flexible to quantify aleatoric uncertainty, which often includes heteroscedastic uncertainty.

GP models offer an alternative modeling framework. Within this framework, a standard GP is typically formulated as:

$$y_i = f_{\boldsymbol{\theta}}\left(\boldsymbol{x}_i\right) + \epsilon_i,$$

where $f_{\boldsymbol{\theta}}\left(\cdot\right)$ is a function drawn from a GP defined by a mean function $\mu\left(\cdot\right)$ and a kernel $k_{\boldsymbol{\theta}}(\cdot,\cdot)$. Conventional GPs often employ stationary kernels and assume homoscedastic observation noise ϵ_i. These assumptions restrict their ability to model heteroscedastic uncertainty. Thus, DGPs were introduced to enhance uncertainty expressiveness (Damianou and Lawrence, 2013), extending GPs into hierarchical, multi-layered architectures inspired by deep learning. A common DGP formulation is:

$$y_i = f_{\boldsymbol{\theta}}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}_i\right)\right) + \epsilon_i,$$

where $h_{\boldsymbol{\Theta}}\left(\cdot\right)$ represents a latent GP layer with its own kernel $k_{\boldsymbol{\Theta}}(\cdot,\cdot)$. Despite their expressive power, the layered structure of DGPs brings significant training challenges, often necessitating sophisticated variational inference techniques (e.g., Salimbeni and Deisenroth 2017).

In summary, contemporary uncertainty quantification frameworks, including BNNs and GPs, exhibit constrained expressiveness to characterize aleatoric and epistemic uncertainty. This is primarily due to their reliance on Gaussian distributional assumptions, imposed for analytical tractability and computational efficiency. Conversely, more sophisticated approaches, such as DGPs, present excessive computational complexity that impedes effective uncertainty quantification in high-dimensional parameter spaces, particularly in scenarios with limited observational data.

3. Joint and Condition Modeling for Uncertainty Quantification

As discussed in Section 2, a typical regression model generally consists of two components. The first component is feature extraction, which processes the input through linear or nonlinear models, such as basis functions, neural networks, or kernel methods, to extract features relevant to the response variable. The second component is the link function, such as identity, logistic, or log, which employs linear or nonlinear methods to map the extracted features to the response variable.

3.1 Joint and Condition Modeling

Unlike conventional regression models, such as GPs and BNNs, which utilize explicit link functions, this study proposes an implicit link function. Specifically, we assume that the joint distribution of the vector $\boldsymbol{\xi} = [y, \boldsymbol{z}^T]^T$ is modeled as $\eta_{\boldsymbol{\phi}}(\cdot)$ for some parameterization $\boldsymbol{\phi}$, i.e., $\boldsymbol{\xi} \sim \eta_{\boldsymbol{\phi}}(\cdot)$, where $\boldsymbol{z} \in \mathbb{R}^p$ denotes the features extracted from the input $\boldsymbol{x}$. The proposed model differs from an explicit observation model, where $y \sim \eta_{\boldsymbol{\phi}}(\boldsymbol{z})$ is commonly assumed. By appropriately choosing the joint distribution $\eta_{\boldsymbol{\phi}}(\cdot)$, the conditional distribution $\mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{z})$ can be inferred. This conditional distribution is then interpreted as the link function. This framework is called the Joint and Condition Modeling (JCM) approach.

We begin by assuming a Gaussian form for the joint distribution. Specifically, we assume that the joint distribution of $\boldsymbol{\xi}$ follows a multivariate normal distribution: $\eta_{\boldsymbol{\phi}}(\cdot) \equiv \mathcal{N}(\cdot|\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where the mean vector $\boldsymbol{\mu}$ and the covariance matrix $\boldsymbol{\Sigma}$ can be partitioned as $\boldsymbol{\mu} = [\mu_y, \boldsymbol{\mu}_{\boldsymbol{z}}^T]^T$ and

$$\boldsymbol{\Sigma} = \begin{bmatrix} \Sigma_{yy} & \boldsymbol{\Sigma}_{y\boldsymbol{z}} \\ \boldsymbol{\Sigma}_{\boldsymbol{z}y} & \boldsymbol{\Sigma}_{\boldsymbol{z}\boldsymbol{z}} \end{bmatrix}.$$

The conditional distribution can be calculated in a closed form: $\mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{z}) = \mathcal{N}(y|\mu_{y|\boldsymbol{z}}, \boldsymbol{\Sigma}_{y|\boldsymbol{z}})$, where $\mu_{y|\boldsymbol{z}} = \mu_y + \boldsymbol{\Sigma}_{y\boldsymbol{z}}\boldsymbol{\Sigma}_{\boldsymbol{z}\boldsymbol{z}}^{-1}(\boldsymbol{z} - \boldsymbol{\mu}_{\boldsymbol{z}})$ and $\boldsymbol{\Sigma}_{y|\boldsymbol{z}} = \Sigma_{yy} - \boldsymbol{\Sigma}_{y\boldsymbol{z}}\boldsymbol{\Sigma}_{\boldsymbol{z}\boldsymbol{z}}^{-1}\boldsymbol{\Sigma}_{\boldsymbol{z}y}$.

To enhance the expressiveness of the proposed model, we employ a GMM with K components to characterize the joint distribution $\eta_{\boldsymbol{\phi}}(\cdot)$, i.e., $\eta_{\boldsymbol{\phi}}(\cdot) \equiv \sum_{k=1}^{K} \pi_k \mathcal{N}(\cdot|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, where $\sum_{k=1}^{K} \pi_k = 1$. The parameter set $\boldsymbol{\phi} = \{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$ fully specifies $\eta_{\boldsymbol{\phi}}(\cdot)$ and hence the resulting link function. Given this GMM-based joint distribution of $\boldsymbol{\xi}$, we can derive the conditional distribution using the JCM approach. Specifically, given the joint distribution

$$\boldsymbol{\xi} = \begin{bmatrix} y \\ \boldsymbol{z} \end{bmatrix} \sim \sum_{k=1}^{K} \pi_k \mathcal{N}\left(\boldsymbol{\xi}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right), \tag{1}$$

where

$$\boldsymbol{\mu}_k = \begin{bmatrix} \mu_{k,y} \\ \boldsymbol{\mu}_{k,\boldsymbol{z}} \end{bmatrix} \text{ and } \boldsymbol{\Sigma}_k = \begin{bmatrix} \Sigma_{k,y} & \boldsymbol{r}_k^T \\ \boldsymbol{r}_k & \boldsymbol{\Sigma}_{k,\boldsymbol{z}} \end{bmatrix},$$

the conditional distribution can be expressed as

$$\mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{z}) = \sum_{k=1}^{K} \pi_k^*(\boldsymbol{z}) \mathcal{N}(y|\mu_{k,y|\boldsymbol{z}}, \Sigma_{k,y|\boldsymbol{z}}), \tag{2}$$

where for each component k, the weight, mean, and covariance are

$$\pi_k^*(\boldsymbol{z}) = \frac{\pi_k \mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu}_{k,\boldsymbol{z}}, \boldsymbol{\Sigma}_{k,\boldsymbol{z}})}{\sum_{l=1}^{K} \pi_l \mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu}_{l,\boldsymbol{z}}, \boldsymbol{\Sigma}_{l,\boldsymbol{z}})}, \tag{3}$$

$$\mu_{k,y|\boldsymbol{z}} = \mu_{k,y} + \boldsymbol{r}_k^T \Sigma_{k,\boldsymbol{z}}^{-1}(\boldsymbol{z} - \boldsymbol{\mu}_{k,\boldsymbol{z}}), \tag{4}$$

$$\Sigma_{k,y|\boldsymbol{z}} = \Sigma_{k,y} - \boldsymbol{r}_k^T \boldsymbol{\Sigma}_{k,\boldsymbol{z}}^{-1} \boldsymbol{r}_k. \tag{5}$$

Note that this weighted normal formula in Eq. (2) offers a favorable advantage in that it can serve as a posterior distribution because it has been normalized (i.e., $\sum_{k=1}^{K} \pi_k^*(\boldsymbol{z}) = 1$).

In contrast to the unimodal Gaussian distribution typically employed in BNNs and GPs, the mixture of Gaussian distributions $\eta_{\boldsymbol{\phi}}(\cdot)$ provides a conditional distribution for $\mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{z})$ in Eq. (2), which is itself a GMM. This inherent GMM structure for the conditional distribution is a key advantage of JCM, providing significantly enhanced expressiveness for modeling complex uncertainty.

Furthermore, this formulation offers notable computational scalability in deriving the conditional distribution $\mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{z})$. The dimensionality of the covariance matrix $\boldsymbol{\Sigma}$ for each mixture component is $p+1$, where p is the dimensionality of $\boldsymbol{z}$. Crucially, p remains constant and independent of the sample size n in this study. Such scalability presents a significant improvement over conventional methods like GPs, whose computational complexity often grows cubically with n.

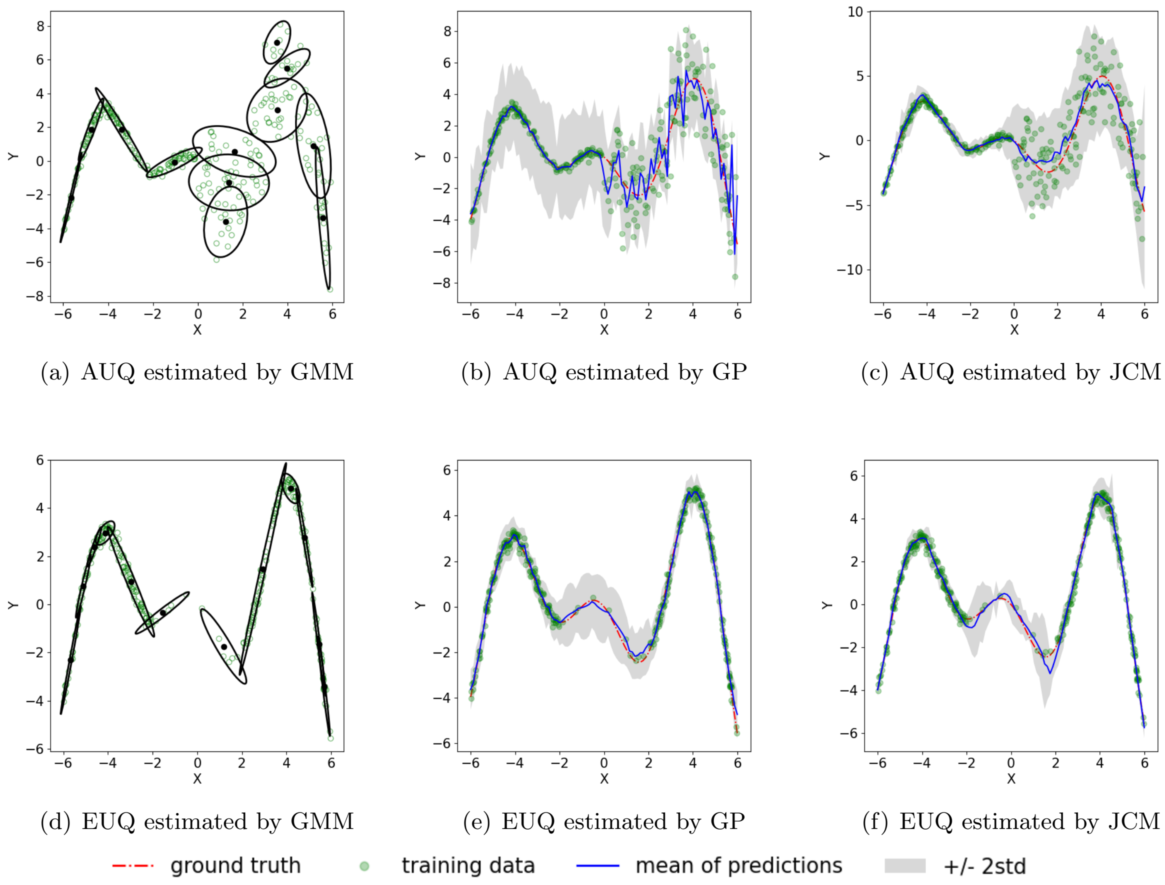
To illustrate the core concept of the JCM approach, we present a simple example in one-dimensional space (see Example 1). In this case, the input $\boldsymbol{x}$ is scalar (i.e., $d = 1$), and the feature $\boldsymbol{z}$ is directly set equal to the input, $\boldsymbol{z} = \boldsymbol{x}$, thereby eliminating the need for an explicit feature extraction model.

Example 1. *The data are generated from a simple regression function:* $y = -(1+x)\sin(1.2x) +\epsilon$, *where* $x \in [-6, 6]$, *and* ϵ *denotes the noise term. To simulate heteroscedastic noise for aleatoric uncertainty quantification (AUQ), we deliberately introduce noise levels in the region* $x \in [0, 6]$ *that are significantly higher than those in* $x \in [-6, 0]$, *as depicted in Fig. 1. Conversely, for epistemic uncertainty quantification (EUQ), we ensure that the number of given samples in* $x \in [-2, 2]$ *is smaller than in other regions. We employ a 12-component GMM model to fit the datasets, and the means (dots) and covariances (ellipses) of the fitted GMM are shown in Fig. 1 (a) and Fig. 1 (d). Given the GMM and the test data, we estimate the conditional distribution according to Eq.* (2)*, as shown in Fig. 1 (e) and (f). For comparison, we apply GP regression with the Matern kernel to the same task.*

Due to its limited capacity to effectively model heteroscedastic noise, the GP exhibits high variance in the $x \in [-6, 0]$ region and an insufficiently smooth mean estimation in $x \in [0, 6]$ (see Fig. 1 (b)). This means that GP fails to characterize AUQ accurately. In contrast, JCM provides more reliable results for heteroscedastic noise by accurately estimating the variance across the entire domain, as illustrated in Fig. 1 (c). For EUQ, the standard GP model is recognized for its effectiveness, performing well in quantifying EUQ within the sparse-data region $x \in [-2, 2]$ (Fig. 1 (e)). JCM also demonstrates satisfactory performance for EUQ, as depicted in Fig. 1 (f). This example effectively demonstrates JCM's excellent expressiveness, which allows the quantification of both aleatoric and epistemic uncertainty.

3.2 Linear Dimensionality Reduction Model

To manage high-dimensional input data $\boldsymbol{x} \in \mathbb{R}^d$, we employ a linear dimensionality reduction model. This model seeks a low-dimensional representation $\boldsymbol{z} \in \mathbb{R}^p$ by projecting the d-dimensional input $\boldsymbol{x}$ onto a p-dimensional feature space using an unknown projection matrix



(a) AUQ estimated by GMM

(b) AUQ estimated by GP

(c) AUQ estimated by JCM

(d) EUQ estimated by GMM

(e) EUQ estimated by GP

(f) EUQ estimated by JCM

Figure 1: UQ estimated by the GMM, GPR, and JCM methods.

$\boldsymbol{W} \in \mathbb{R}^{p \times d}$. We opt for a linear feature extraction function, $\boldsymbol{z}_i = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) = \boldsymbol{W}\boldsymbol{x}_i$, to strike a balance between model simplicity and uncertainty expressiveness.

To ensure identifiability in the estimation of $\boldsymbol{W}$, we adopt a constraint inspired by Suzuki and Sugiyama (2010):

$$\mathcal{M} = \left\{ \boldsymbol{W} \in \mathbb{R}^{p \times d} | \boldsymbol{W}\boldsymbol{W}^T = \boldsymbol{I}_p \right\}, \tag{6}$$

where $\boldsymbol{I}_p$ is the p-dimensional identity matrix. This constraint restricts $\boldsymbol{W}$ to the Stiefel manifold. The target dimensionality p is assumed to be known, which in practice can be selected using cross-validation (Efron and Tibshirani, 1995).

3.3 Loss Function of Dimensionality Reduction Model

This section addresses the optimization of the dimensionality reduction model. We assume that the GMM for the joint variable $\boldsymbol{\xi} = [y, \boldsymbol{z}^T]^T$ has already been fitted. Consequently, the parameters $\boldsymbol{\phi}$ defining the link function $\mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{z})$, as specified in Eq. (2), are considered known. Our objective is to determine the optimal projection matrix $\boldsymbol{W}$ by minimizing the loss functions detailed subsequently.

Regression Loss: Given the conditional distribution $\mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{z})$ derived from Eq. (2), we define a regression loss based on the negative log likelihood of $y_1, y_2, \cdots, y_N$. This loss function for optimizing $\boldsymbol{W}$ is given by:

$$\mathcal{L}_{Reg}(\boldsymbol{W}; \boldsymbol{\phi}) = \sum_{i=1}^{N} -\log \mathcal{P}_{\boldsymbol{\phi}}(y_i|\boldsymbol{W}\boldsymbol{x}_i), \tag{7}$$

where $\mathcal{P}_{\boldsymbol{\phi}}(y_i|\boldsymbol{W}\boldsymbol{x}_i)$ represents the conditional probability density $\mathcal{P}(y_i|\boldsymbol{z}_i)$, evaluated using Eq. (2) with $\boldsymbol{z}_i = \boldsymbol{W}\boldsymbol{x}_i$. Notably, unlike standard regression models, which often assume a single Gaussian distribution for $y|\boldsymbol{z}$, our approach leverages a mixture of Gaussian distributions. This provides superior uncertainty expressiveness due to the GMM's inherent flexibility in fitting complex distributions.

Reconstruction loss: The reconstruction error measures how effectively the low-dimensional embedding $\boldsymbol{z}$ preserves information from the original data $\boldsymbol{x}$. We incorporate this loss to ensure that the dimension reduction model avoids trivial solutions. For mathematical tractability, we express our dimension reduction model in a matrix form: $\boldsymbol{Z} = \boldsymbol{X}\boldsymbol{W}^T$, where $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N]^T$ and $\boldsymbol{Z} = [\boldsymbol{z}_1, \cdots, \boldsymbol{z}_N]^T$.

The reconstruction error quantifies the discrepancy between the original data and its reconstruction: $\|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{W}^T\boldsymbol{W}\|_F^2$, where $\|\cdot\|_F^2$ denotes the Frobenius norm. Using the identity $\|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{W}^T\boldsymbol{W}\|_F^2 = \text{trace}\left((\boldsymbol{X} - \boldsymbol{X}\boldsymbol{W}^T\boldsymbol{W})^T(\boldsymbol{X} - \boldsymbol{X}\boldsymbol{W}^T\boldsymbol{W})\right)$, we obtain

$$\|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{W}^T\boldsymbol{W}\|_F^2 = \text{trace}(\boldsymbol{X}\boldsymbol{X}^T) - \text{trace}(\boldsymbol{X}\boldsymbol{W}^T\boldsymbol{W}\boldsymbol{X}^T).$$

Since $\text{trace}(\boldsymbol{X}\boldsymbol{X}^T)$ is constant with respect to $\boldsymbol{W}$, minimizing the reconstruction error is equivalent to maximizing $\text{trace}(\boldsymbol{X}\boldsymbol{W}^T\boldsymbol{W}\boldsymbol{X}^T)$. Therefore, we define the reconstruction loss as:

$$\mathcal{L}_{\text{Rec}}(\boldsymbol{W}) = -\frac{1}{N}\text{trace}(\boldsymbol{X}\boldsymbol{W}^T\boldsymbol{W}\boldsymbol{X}^T). \tag{8}$$

Regularization loss: Additionally, to enhance the interpretability of the dimension reduction model, particularly when $d \gg p$, a sparsity penalty, inspired by Lasso (Tibshirani, 1996), is introduced on the projection matrix $\boldsymbol{W}$. This penalty is expressed as:

$$\mathcal{L}_{\|\cdot\|_1}(\boldsymbol{W}) = \|\boldsymbol{W}\|_1, \tag{9}$$

where $\|\cdot\|_1$ represents the L_1 norm. This L_1 regularization helps prevent the proposed method from overfitting.

Finally, the total loss function can be obtained as follows:

$$\mathcal{L}(\boldsymbol{W}) = \mathcal{L}_{\text{Reg}}(\boldsymbol{W}; \boldsymbol{\phi}) + \lambda_1\mathcal{L}_{\text{Rec}}(\boldsymbol{W}) + \lambda_2\mathcal{L}_{\|\cdot\|_1}(\boldsymbol{W}), \tag{10}$$

subject to $\boldsymbol{W}\boldsymbol{W}^T = \boldsymbol{I}_p$. Here, λ_1 and λ_2 are hyperparameters that can be prescribed or determined by cross-validation (Efron and Tibshirani, 1995). Optimizing the projection matrix $\boldsymbol{W}$ using this loss function, for instance via an SGD-based algorithm, can be challenging due to the orthonormality constraint $\boldsymbol{W}\boldsymbol{W}^T = \boldsymbol{I}_p$, which requires specialized optimization methods.

3.4 Manifold Optimization via Riemannian Stochastic Gradient Descent

To address the optimization difficulty caused by the orthogonality constraint in Eq. (10), we place the constraint on a Stiefel manifold and employ a manifold optimization technique (Absil et al., 2009). This manifold optimization can effectively address nonlinear optimization problems of the form $\min_{\boldsymbol{W}\in\mathcal{M}} \mathcal{L}(\boldsymbol{W})$, where $\mathcal{L}$ denotes the loss function and the search space $\mathcal{M}$ is a smooth Riemannian manifold (Absil et al., 2009). In this paper, the projection matrix $\boldsymbol{W}$ lies within the Stiefel manifold defined as $\mathcal{M} := \left\{\boldsymbol{W} \in \mathbb{R}^{d\times p} : \boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}\right\}$, which enforces the orthogonality of its columns. Essentially, manifold optimization transforms constrained optimization problems into unconstrained ones, enabling operations directly on the manifold $\mathcal{M}$ and thus inherently fulfilling the desired constraints (Absil et al., 2009; Bonnabel, 2013).

Here, we provide a brief introduction to the fundamental concepts of manifold optimization. Several key Riemannian operators used in the proposed manifold optimization are summarized in Tab. 1. For a comprehensive understanding of the underlying geometric and differential geometric principles, readers are referred to (Lee, 2006; Absil et al., 2009).

Table 1: Operators on the Stiefel manifold

Definition	Expression for the Stiefel manifold
—rule Metric between ζ and η at $\boldsymbol{W}$	$\rho_w(\zeta, \eta) = \frac{1}{2}tr(\zeta^T\eta)$
Gradient at $\boldsymbol{W}$ if Euclidean gradient is $\nabla_E f$	$\nabla f(\boldsymbol{W}) = (I - \boldsymbol{W}\boldsymbol{W}^T)\nabla_E f + \boldsymbol{W}\text{skew}(\boldsymbol{W}^T\nabla_E f)$
Exponential map at $\boldsymbol{W}$ in direction ζ	$Exp_{\boldsymbol{W}}(\zeta) = \boldsymbol{W}\exp(\boldsymbol{W}^T\zeta - \zeta^T\boldsymbol{W})$
Project ζ to the tangent space of $\boldsymbol{W}$	$\mathcal{P}_{\boldsymbol{W}}(\zeta) = \zeta - \frac{1}{2}\boldsymbol{W}(\boldsymbol{W}^T\zeta + (\boldsymbol{W}^T\zeta)^T)$
Retraction at $\boldsymbol{W}$ in direction ζ	$Ret_{\boldsymbol{W}}(\zeta) = \text{qf}(\boldsymbol{W} + \zeta)$

Riemannian Manifold: A Riemannian manifold $(\mathcal{M}, g)$ is defined as a smooth manifold $\mathcal{M}$ equipped with a Riemannian metric g. At each point $\boldsymbol{W} \in \mathcal{M}$, this metric g induces an inner product $g_{\boldsymbol{W}}(\cdot,\cdot)$ on the tangent space $T_{\boldsymbol{W}}\mathcal{M}$, specifically $g_W : T_{\boldsymbol{W}}\mathcal{M} \times T_{\boldsymbol{W}}\mathcal{M} \to \mathbb{R}$. This inner product allows for the measurement of lengths and angles within the tangent spaces, thereby establishing a well-defined notion of distance and geometry on the manifold. To extend the concept of the Euclidean gradient, denoted by ζ, to the tangent space $T_W\mathcal{M}$ at the point $\boldsymbol{W}$, we need to employ a projection operator. In particular, for the Stiefel manifold, as shown in Tab. 1, we can utilize the closed-form projection given by $\mathcal{P}_{\boldsymbol{W}}(\zeta) = \zeta - \frac{1}{2}\boldsymbol{W}(\boldsymbol{W}^T\zeta + (\boldsymbol{W}^T\zeta)^T)$. Thus, in order to perform gradient-based Manifold optimization, we first obtain the Riemannian gradient on the tangent of $\mathcal{M}$ as follows:

$$\widehat{\nabla}_{\boldsymbol{W}}\mathcal{L} = \nabla_{\boldsymbol{W}}\mathcal{L} - \frac{1}{2}\boldsymbol{W}\left(\boldsymbol{W}^T\nabla_{\boldsymbol{W}}\mathcal{L} + \left(\boldsymbol{W}^T\nabla_{\boldsymbol{W}}\mathcal{L}\right)^T\right) \tag{11}$$

Exponential Map: The exponential map $\text{Exp}_{\boldsymbol{W}}(\cdot)$ extends the idea of movement along a vector in Euclidean space to the curved geometry on the manifold. Specifically, starting from a point $\boldsymbol{W}$ and following the geodesic shortest path on the manifold defined by a tangent vector $\zeta_{\boldsymbol{W}} \in T_{\boldsymbol{W}}\mathcal{M}$ for a unit distance yields the point $Exp_{\boldsymbol{W}}(\zeta_{\boldsymbol{W}})$. For the tangent vector $\widehat{\nabla}_{\boldsymbol{W}}\mathcal{L}$ defined in Eq. (11), the exponential map $Exp_W(\alpha\widehat{\nabla}_{\boldsymbol{W}}\mathcal{L})$ effectively moves from $\boldsymbol{W}$ in the direction of $\widehat{\nabla}_{\boldsymbol{W}}$ over a distance scaled by α, where the parameter α serves as a step size to control the movement along the geodesic. This operator provides a generalization of the Euclidean operation $\boldsymbol{W} + \alpha\widehat{\nabla}_{\boldsymbol{W}}$ constrained to the manifold $\mathcal{M}$. The exponential map is essential for optimization on Riemannian manifolds because updates must respect the geometric structure of the manifold (Absil et al., 2009).

Retraction: As computing an exponential map is usually expensive, a retraction map is often used as an efficient alternative. For the Stiefel manifold, we utilize a simple closed-form for the retraction:

$$R_{\boldsymbol{W}}(\widehat{\nabla}_{\boldsymbol{W}}\mathcal{L}) = \mathrm{qf}(\boldsymbol{W} + \alpha\widehat{\nabla}_{\boldsymbol{W}}\mathcal{L}), \tag{12}$$

where $\mathrm{qf}(\cdot)$ denotes the QR decomposition. This retraction operation ensures that the updated matrix $\boldsymbol{W}$ always fulfills the constraint $\boldsymbol{W}\boldsymbol{W}^T = \boldsymbol{I}_p$.

To optimize $\boldsymbol{W}$ iteratively with minibatch data, the Riemannian SGD algorithm (Bonnabel, 2013) is applied. It extends the traditional SGD's gradient update in Euclidean space to the Riemannian space using the following updating rule:

$$\boldsymbol{W}_{t+1} = R_{\boldsymbol{W}_t}(-\alpha_t\widehat{\nabla}_{\boldsymbol{W}_t}\mathcal{L}), \tag{13}$$

where $\alpha_t > 0$ is a (decaying) step size. Here, the term $\widehat{\nabla}_{\boldsymbol{W}_t}\mathcal{L}$ represents a Riemannian stochastic gradient defined in Eq. (11). The retraction $R_{\boldsymbol{W}_t}$, which is realized by Eq. (12), maps the tangent space $\mathcal{T}_{\boldsymbol{W}}$ onto $\mathcal{M}$, with a local rigidity condition that preserves the gradients at $\boldsymbol{W}$.

3.5 The Complete Algorithm

The core of our proposed method is an iterative algorithm that alternates between two distinct optimization stages. In the first stage, the projection matrix $\boldsymbol{W}$ is considered fixed, and the GMM parameters $\boldsymbol{\phi}$ are updated via the EM algorithm. In the second stage, the GMM parameters $\boldsymbol{\phi}$ are held constant while the projection matrix $\boldsymbol{W}$ is optimized. This optimization is performed using a Riemannian SGD algorithm, which is suitable for the orthogonal constraints on $\boldsymbol{W}$. The comprehensive procedure is outlined in Algorithm 1. The

4. Uncertainty Quantification Based on Joint and Condition Modeling

After learning the dimensionality reduction model and the joint distribution by Algorithm 1, the conditional/predictive distribution $\mathcal{P}(y|\boldsymbol{x}) = \mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{W}\boldsymbol{x})$ can be evaluated by Eq. (2). Here, we derive the predictive mean, covariance, quantile estimation, and conditional entropy, which can be obtained based on the conditional distribution.

4.1 Predictive Mean and Variance

For any new input $\boldsymbol{x}$, we have $\boldsymbol{z} = \boldsymbol{W}\boldsymbol{x}$ according to the dimension reduction model. Then, the mean function $E\left[y|\boldsymbol{x}\right]$ of the input $\boldsymbol{x}$ can be expressed in closed form as follows:

$$\mathbb{E}\left[y|\boldsymbol{x}\right] = \mathbb{E}_{\mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{W}\boldsymbol{x})}\left[y|\boldsymbol{W}\boldsymbol{x}\right] = \sum_{k=1}^{K}\pi_k^*(\boldsymbol{z})\left(\mu_{k,y} + \boldsymbol{r}_k^T\Sigma_{k,\boldsymbol{z}}^{-1}(\boldsymbol{z} - \boldsymbol{\mu}_{k,\boldsymbol{z}})\right), \tag{14}$$

Algorithm 1 Two Stage Optimization for the Joint and Condition Modeling

```
• Initialization: initialize $\boldsymbol{W}$ on the Stiefel manifold and $\{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$ for $k = 1, ..., K$, and set the learning rate $\alpha$ for the Riemannian SGD.
• For $t = 1, 2, \cdots, T$,
    – Given $\boldsymbol{\xi}_i = [y_i, (\boldsymbol{W}\boldsymbol{x}_i)^T]^T$ (for $i = 1, ..., N$), update $\boldsymbol{\phi} = \{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$ by EM as follows:
        $r(c|\boldsymbol{\xi}_i) \leftarrow \frac{\pi_c N(\boldsymbol{\xi}_i|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)}{\sum_{c'} \pi_{c'} N(\boldsymbol{\xi}_i|\boldsymbol{\mu}_{c'}, \boldsymbol{\Sigma}_{c'})}$    ▷ E-step
        $\pi_k \leftarrow \frac{1}{\sum_{i=1}^N \sum_{c=1}^K r(c=k|\boldsymbol{\xi}_i)} \sum_{i=1}^N r(c = k|\boldsymbol{\xi}_i)$
        $\boldsymbol{\mu}_k \leftarrow \frac{1}{\sum_{i=1}^N r(c=k|\boldsymbol{\xi}_i)} \sum_{i=1}^N r(c = k|\boldsymbol{\xi}_i)\boldsymbol{\xi}_i$    ▷ M-Step
        $\boldsymbol{\Sigma}_k \leftarrow \frac{1}{\sum_{i=1}^N r(c=k|\boldsymbol{\xi}_i)} \sum_{i=1}^N r(c = k|\boldsymbol{\xi}_i)(\boldsymbol{\xi}_i - \boldsymbol{\mu}_k)(\boldsymbol{\xi}_i - \boldsymbol{\mu}_k)^T$
    – Given $\boldsymbol{\phi}$, update $\boldsymbol{W}$ by the Riemannian SGD algorithm:
        $\widehat{\nabla}_{\boldsymbol{W}}\mathcal{L} \leftarrow \frac{1}{2}\left(\nabla_{\boldsymbol{W}}\mathcal{L} - \boldsymbol{W}\left(\nabla_{\boldsymbol{W}}\mathcal{L}\right)^T \boldsymbol{W}\right)$    ▷ project to the tangent space
        $\boldsymbol{W}_{t+1} \leftarrow \mathrm{qf}(\boldsymbol{W}_t - \alpha\widehat{\nabla}_{\boldsymbol{W}}\mathcal{L})$    ▷ retraction
• Output: The dimension reduction matrix $\boldsymbol{W}$ and the GMM with parameters $\boldsymbol{\phi} = \{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$.
```

where $\pi_k^*(\boldsymbol{z})$ and $\mu_{k,y|\boldsymbol{z}}$ are given by Eqs. (3) and (4). The variance $\mathrm{Var}(y|\boldsymbol{x})$ can be expressed as follows:

$$\begin{aligned}\mathrm{Var}(y|\boldsymbol{x}) =& \mathbb{E}_{\mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{W}\boldsymbol{x})}\left[(y - \mathbb{E}[y|\boldsymbol{W}\boldsymbol{x}])^2\middle|\boldsymbol{W}\boldsymbol{x}\right] \\ =& \sum_{k=1}^K \pi_k^*(\boldsymbol{z})\Sigma_{k,y|\boldsymbol{z}} + \sum_{k=1}^K \pi_k^*(\boldsymbol{z})\mu_{k,y|\boldsymbol{z}}^2 - \left(\sum_{k=1}^K \pi_k^*(\boldsymbol{z})\mu_{k,y|\boldsymbol{z}}\right)^2\end{aligned} \tag{15}$$

where the expectation is taken for distribution $\mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{W}\boldsymbol{x})$ and $\Sigma_{k,y|\boldsymbol{z}}$ is given by Eq. (5). The last term in the above equation is the variance of $E\left[y|\boldsymbol{W}\boldsymbol{x}\right]$.

4.2 Quantile Estimation

Quantile regression allows us to estimate the conditional quantiles of the target distribution, providing a comprehensive view of the uncertainty beyond the mean prediction. Accurate prediction intervals are crucial for reliable uncertainty quantification and improved decision-making in real-world applications (Shen et al., 2024).

Let $F(y_q) = \mathcal{P}(y|\boldsymbol{z} \leq y_q)$ be the strictly monotone cumulative distribution function (CDF) of $y|\boldsymbol{z}$ taking real values y_q. Consequently, let $F^{-1}(q) = \inf\{y_q : F(y_q) \geq q\}$ denote the quantile distribution function of the variable $y|\boldsymbol{z}$, for any quantile level $0 \leq q \leq 1$.

Given the conditional distribution expressed in Eq. (2), we can simulate a batch of samples from the conditional distribution by the Monte Carlo method and then estimate the quantile based on these samples. However, as the simulation study shows, this Monte Carlo-based approach suffers from low accuracy and high computational cost.

Alternatively, we propose a more effective way for quantile estimation. Given the conditional distribution defined in Eq. (2), we have $\mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{z}) = \sum_{k=1}^{K} \pi_k^* (\boldsymbol{z}) \mathcal{P}_k(y|\boldsymbol{z})$, where $\mathcal{P}_k(y|\boldsymbol{z}) = \mathcal{N}(y|\mu_{k,y|\boldsymbol{z}}, \Sigma_{k,y|\boldsymbol{z}})$. It follows that the CDF of the observed data is

$$F(y_q) = \int_{-\infty}^{y_q} \sum_{k=1}^{K} \pi_k^* (\boldsymbol{z}) \mathcal{P}_k(y|\boldsymbol{z}) dy = \sum_{k=1}^{K} \pi_k^* (\boldsymbol{z}) \int_{-\infty}^{y_q} \mathcal{P}_k(y|\boldsymbol{z}) dy = \sum_{k=1}^{K} \pi_k^* (\boldsymbol{z}) F_k(y_q), \quad (16)$$

where

$$F_k(y_q) = \varPhi \left(\frac{y_q - \mu_{k,y|\boldsymbol{z}}}{\sqrt{\Sigma_{k,y|\boldsymbol{z}}}} \right)$$

is the CDF of the standard normal distribution. This CDF is continuous and easy enough to calculate, so the inverse CDF $F^{-1}(q)$, also known as the quantile function, can be efficiently calculated by performing a line search (LS).

4.3 Entropy and Its Lower Bound

Entropy is a fundamental concept in information theory that quantifies the uncertainty of a random variable. In our setting, we have

$$H(y|\boldsymbol{x}) = -\mathbb{E}[\log \mathcal{P}(y|\boldsymbol{x})] = -\mathbb{E}_{\mathcal{P}_{\boldsymbol{\phi}}}[\log \mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{W}\boldsymbol{x})]. \quad (17)$$

where the distribution $\mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{W}\boldsymbol{x})$ is a Gaussian mixture, as shown in Eq. (2). However, the entropy of Gaussian mixtures cannot generally be computed in closed form because it involves the logarithm of a sum of exponential terms. To overcome this challenge, we introduce a lower and upper bounds for the true entropy values, which are computationally cheap. These bounds can be computed in closed form and provide a practical approximation for the entropy. The detailed expressions for these bounds are presented in Theorem 1, with a full proof provided in the Appendix.

Theorem 1. ***(Lower and Upper Bounds)*** *Given the dimension reduction model $\boldsymbol{z} = \boldsymbol{W}\boldsymbol{x}$, and the predicted conditional distribution $\mathcal{P}(y|\boldsymbol{z})$ defined in Eq.* (2)*, the following inequality holds,*

$$H_l(y|\boldsymbol{x}) \leq H(y|\boldsymbol{x}) \leq H_u(y|\boldsymbol{x}) \quad (18)$$

where the lower bound is

$$H_l(y|\boldsymbol{x}) = -\sum_{i=1}^{K} \pi_i^*(\boldsymbol{z}) \log \left(\sum_{j=1}^{K} \pi_j^*(\boldsymbol{z}) \mathcal{N}(\mu_{i,y|\boldsymbol{z}} | \mu_{j,y|\boldsymbol{z}}, \Sigma_{i,y|\boldsymbol{z}} + \Sigma_{j,y|\boldsymbol{z}}) \right), \quad (19)$$

and the upper bound is

$$H_u(y|\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k^*(\boldsymbol{z}) \frac{\log(2\pi) + \log\left|\Sigma_{k,y|\boldsymbol{z}}\right| + 1}{2}. \quad (20)$$

Here, $\pi_i^*(\boldsymbol{z})$ given in Eq. (3) represents the optimal weight of the i-th mixture component for input $\boldsymbol{z}$, and $\mathcal{N}(\mu_{i,y|\boldsymbol{z}} | \mu_{j,y|\boldsymbol{z}}, \Sigma_{i,y|\boldsymbol{z}} + \Sigma_{j,y|\boldsymbol{z}})$ evaluates the Gaussian probability density function of the i-th component's mean, centered at the j-th component's mean with a variance reflecting the sum of the i-th and j-th component variances.

4.4 Application to Active Learning

Active learning (AL) is a machine learning paradigm that aims to reduce the cost of data acquisition by enabling the algorithm to selectively query the most informative data points (Settles, 2009). The core idea is that, by strategically selecting which samples to evaluate or include, the model can achieve improved performance with fewer data points compared to random sampling. A widely used class of AL strategies is uncertainty sampling, in which the algorithm prioritizes querying regions of the input space where its predictions are most uncertain. The effectiveness of such strategies critically depends on the accurate quantification of model uncertainty, especially epistemic uncertainty, which reflects the model's uncertainty due to limited knowledge from the training data (Gal et al., 2017; Houlsby et al., 2011). Actively querying points in high-uncertainty regions helps reduce epistemic uncertainty and improves model accuracy.

In many conventional AL approaches, particularly those based on Gaussian processes or Bayesian neural networks assuming Gaussian posteriors, predictive variance has served as a common proxy for epistemic uncertainty (Mackay, 1992). This choice often stems from the analytical tractability of variance under Gaussian assumptions. However, representing uncertainty using a single Gaussian distribution imposes a unimodal structure, which may be insufficient to capture more complex uncertainty structures.

Consequently, we address this limitation by modeling the predictive conditional distribution as a GMM, which offers a substantially more flexible and expressive representation. In particular, it enables the model to capture complex predictive distributions that cannot be represented by a single Gaussian, such as multimodal or non-Gaussian distributions. In this study, we propose using the differential entropy of the predictive conditional distribution as a more comprehensive measure of epistemic uncertainty. The corresponding acquisition function $\alpha_n(\boldsymbol{x})$ aims to select the input $\boldsymbol{x}$ that maximizes this entropy:

$$\text{argmax}_{\boldsymbol{x}}\alpha_n(\boldsymbol{x}) = \text{argmax}_{\boldsymbol{x}} H(y|\boldsymbol{x}). \tag{21}$$

As discussed in the previous section, computing the exact differential entropy for a GMM is often intractable. However, we derived bounds for this quantity. For the purpose of active learning, maximizing the entropy aligns with maximizing its computationally tractable lower bound. Therefore, we utilize the following approximation for the acquisition function, based on the derived lower bound $H_l(y|\boldsymbol{x})$ in Eq. (19):

$$\text{argmax}_{\boldsymbol{x}}\alpha_n(\boldsymbol{x}) \approx \text{argmin}_{\boldsymbol{x}} \sum_{i=1}^{K} \pi_i^*(\boldsymbol{z}) \log\left(\sum_{j=1}^{K} \pi_j^*(\boldsymbol{z}) \mathcal{N}(\mu_{i,y|\boldsymbol{z}}|\mu_{j,y|\boldsymbol{z}}, \Sigma_{i,y|\boldsymbol{z}} + \Sigma_{j,y|\boldsymbol{z}})\right), \tag{22}$$

which is called the lower bound entropy criterion in this study. This metric, derived from the GMM posterior's structure, guides the selection of new data points in our active learning framework.

4.5 Application to Bayesian Optimization

BO is an efficient global optimization method that is widely used in optimizing expensive black-box functions and has been applied in a broad research area (Bergstra et al., 2011; Snoek et al., 2012, 2015; B.Shahriari et al., 2016), such as recommendation systems, robotics, reinforcement learning, and gene synthesis. BO is a sequential and model-based framework that operates through a two-stage procedure (B.Shahriari et al., 2016). In the first stage, a surrogate model (typically a GP model) is built based on some observations to approximate the objective function $y = f(\boldsymbol{x})$. In the second stage, new inputs are chosen by optimizing an acquisition function inheriting from the first stage, which balances the exploration and exploitation of the input space (Jones et al., 1998; Brochu et al., 2010). A variety of acquisition functions are proposed, including probability improvement (PI, Törn and Zilinskas 1989), expected improvement (EI, Jones et al. 1998), upper confidence bound (UCB, Srinivas et al. 2010), UCB with batch query selection (qUCB, Wilson et al. 2017), entropy search (ES, Henrández-Lobato et al. 2014), and knowledge gradient (KG, Scott et al. 2011). Numerous variants and extensions of these methods have since emerged (B.Shahriari et al., 2016; Wang et al., 2023).

Two broadly used acquisition functions are EI and UCB because they have analytical forms and are easy to implement. The EI acquisition function (Jones et al., 1998) computes the expected improvement with respect to the current maximum $f(\boldsymbol{x}^+)$. The improvement function is written as $\mathrm{EI}(\boldsymbol{x}) = \mathbb{E}_{f(\boldsymbol{x})}\left[\max\left\{0, f(\boldsymbol{x}) - f(\boldsymbol{x}^+)\right\}\right]$. The UCB acquisition function (Srinivas et al., 2010) is defined as

$$\mathrm{UCB}(\boldsymbol{x}) = \mu_f(\boldsymbol{x}) + \beta_t \sigma_f(\boldsymbol{x}) \tag{23}$$

where $\mu_f(\boldsymbol{x})$ is the predicted mean value of the objection function f at input $\boldsymbol{x}$, $\sigma_f^2(\boldsymbol{x})$ is the predicted variance of f at input $\boldsymbol{x}$, and β_t is a positive number to control the balance between exploitation and exploration.

In this study, we employ the proposed JCM method as the surrogate model for BO. The standard UCB acquisition function in Eq. (23) uses the variance term, which implicitly assumes a Gaussian posterior. However, when the uncertainty does not follow a Gaussian distribution, the conditional entropy becomes a more appropriate measure of uncertainty. Therefore, we propose replacing the variance term with the conditional entropy, yielding a new acquisition function:

$$\mathrm{UCB}(\boldsymbol{x}) = \mu_f(\boldsymbol{x}) + \beta_t H_f(\boldsymbol{x}), \tag{24}$$

where $\mu_f(\boldsymbol{x}) \equiv \mathbb{E}\left[y|\boldsymbol{x}\right]$ is the predicted mean given by Eq. (14), $H_f(\boldsymbol{x}) \equiv H(y|\boldsymbol{x}) = -\mathbb{E}_{\mathcal{P}_{\boldsymbol{\phi}}}[\log \mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{W}\boldsymbol{x})]$ is the conditional entropy. Since the distribution $P_{\boldsymbol{\phi}}(y|\boldsymbol{W}\boldsymbol{x})$ is a Gaussian mixture, the entropy $H(y|\boldsymbol{x})$ has no closed-form expression. To address this, we propose to use the lower bound, i.e., $H_l(y|\boldsymbol{x})$ given in Eq.(19), to calibrate the uncertainty in the UCB acquisition. This leads to the modified acquisition function:

$$\mathrm{UCB}(\boldsymbol{x}) = \mathbb{E}\left[y|\boldsymbol{x}\right] + \beta_t H_l(y|\boldsymbol{x}), \tag{25}$$

which is called the entropy-based UCB in this study. As shown in Theorem 1, the lower bound $H_l(y|\boldsymbol{x})$ is a straight bound for the entropy $H(y|\boldsymbol{x})$, so maximizing this lower bound is equivalent to maximizing the entropy.

5. Numerical Examples

In this section, we comprehensively demonstrate the effectiveness of our proposed method for uncertainty quantification by investigating both aleatoric and epistemic uncertainties across several benchmark datasets. For aleatoric uncertainty, we perform regression and quantile estimation to assess the proposed method. For epistemic uncertainty, we evaluate our method through active learning and Bayesian Optimization experiments, showcasing its capability to optimize complex objective functions effectively.

Implementation. We use the Python package ***geomstats*** to implement the basic operators on Stiefel manifolds, such as initializing a matrix on a Stiefel manifold. For all experiments, we set the hyperparameters in the loss function Eq. (10) to $\lambda_1 = 0.5$ and $\lambda_2 = 0.05$. The learning rate for the Riemannian SGD takes a fixed value $\alpha = 0.02$. The batch size for the Riemannian SGD is 512. We terminate the optimization process after 50 epochs. The EM algorithm begins with k-means initialization.

5.1 Ablation Study

The proposed JCM features a small number of hyperparameters, requiring only the selection of the embedding dimension p and the number of mixture components K. To investigate the impact of these hyperparameters on performance, we conduct ablation studies on three UCI benchmark datasets (Asuncion, 2007): Boston, Kin8nm, and Protein. We evaluate performance using log likelihood and Root Mean Square Error (RMSE).

In the first study, we fix the number of mixture components at $K = 8$ and vary the embedding dimension p from 2 to the original data dimension d. The resulting log likelihood and RMSE are shown in the top row of Figs. 2 and 3, respectively. In the second study, we fix the embedding dimension at $p = 5$ and vary K from 4 to 20 in steps of 2. These results are presented in the bottom row of the same figures.

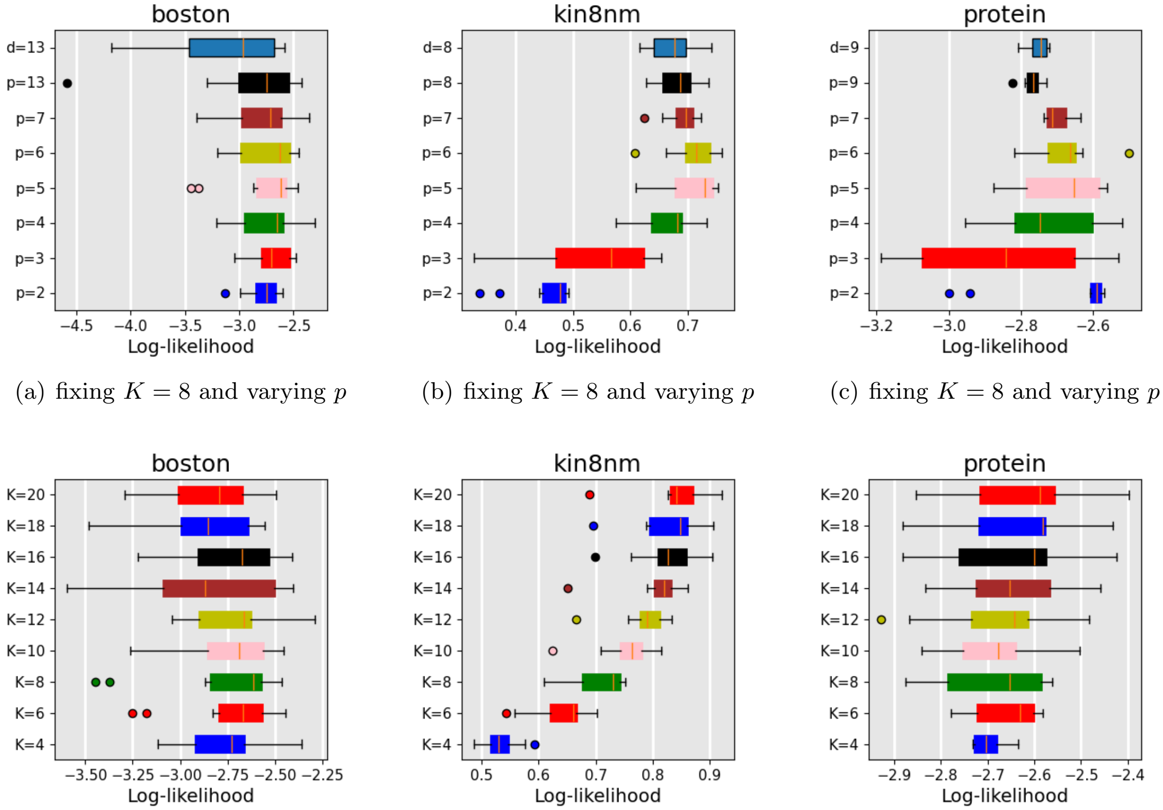
Based on the results in Figs. 2 and 3, we recommend setting the embedding dimension $p = 5$ for all datasets (or $p = d$ if the original dimension d is less than 5). For the number of mixture components, we recommend $K = 8$ for small datasets ($N < 5000$) and $K = 18$ for medium or large datasets.

Additionally, we compare using the original input $\boldsymbol{x}$ directly in JCM (conceptually $p = d$ without projection) against explicitly projecting $\boldsymbol{x}$ to an embedding $\boldsymbol{z}$ even when the target dimension equals the input dimension ($p = d$). The performance shown in the first two boxplots of Fig. 2(a-c) and Fig. 3(a-c) indicates that applying the embedding projection is beneficial, even in this $p = d$ scenario.

5.2 Aleatoric Uncertainty Quantification: Regression and Log-likelihood Estimation

To assess the aleatoric uncertainty captured by our proposed JCM approach, we perform a regression analysis on several UCI benchmark datasets. We focus on log-likelihood estimation in this subsection because it provides a better assessment of uncertainty quantification. Log likelihood considers not only the difference between the predicted and target values, but also the probability distribution of the predictions, offering comprehensive information about the model's uncertainty.

We compare JCM against several baseline models, including BNN-VI (Hernández-Lobato and Adams, 2015), sparse GP with 500 inducing points (SGP, Hensman et al. 2013), DGP with 5 layers, Monte Carlo Dropout (MC-Dropout, Gal and Ghahramani 2016), ensembles of neural networks (NN-Ens, Lakshminarayanan et al. 2017), and the combined variance



(a) fixing $K = 8$ and varying p

(b) fixing $K = 8$ and varying p

(c) fixing $K = 8$ and varying p

(d) fixing $p = 5$ and varying K

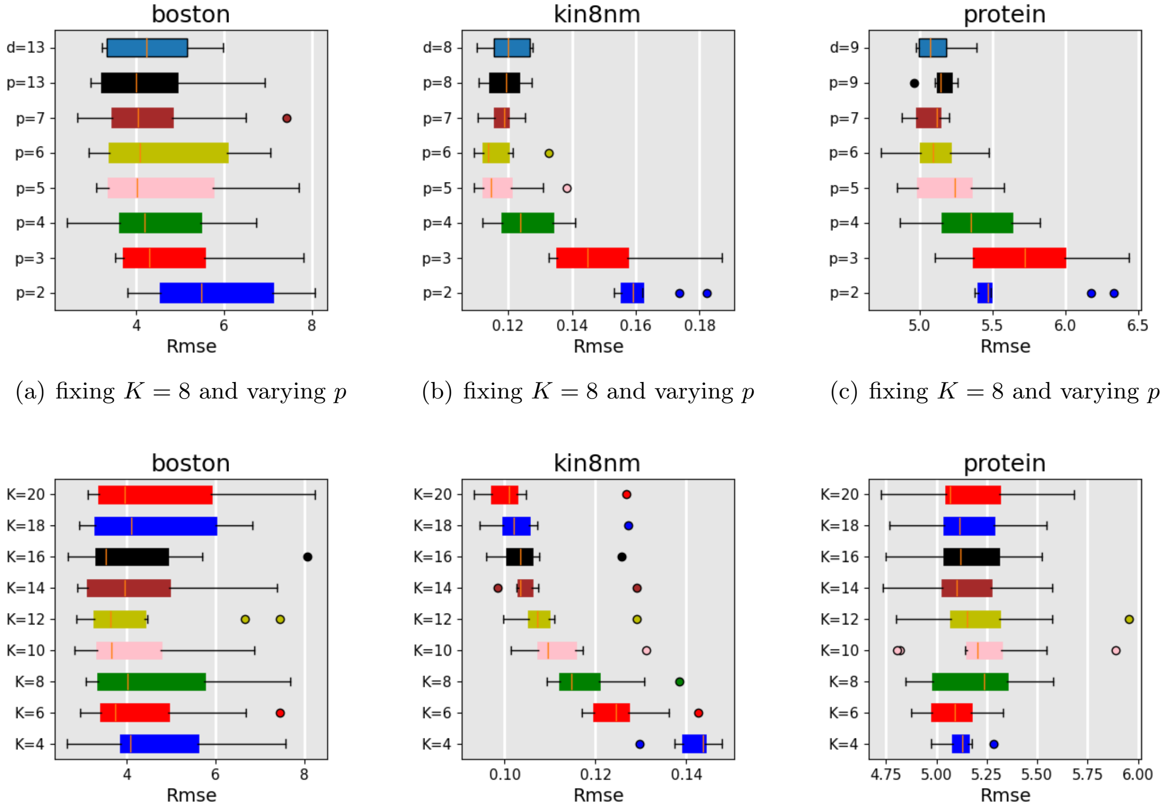
(e) fixing $p = 5$ and varying K

(f) fixing $p = 5$ and varying K

Figure 2: Log-likelihood values for different embedding dimensions and mixture component numbers across the Boston, Kin8nm, and Protein datasets. The upper row shows the impact of varying p while fixing $K = 8$, and the lower row illustrates the impact of varying K while fixing $p = 5$. The legend provides information about the GMM components.

estimation methods (Combined, Detlefsen et al. 2019). The test datasets utilized include Boston (n=506 and d=13), Energy (n=768 and d=8), Concrete (n=1030 and d=8), Kin8nm (n=8192 and d=8), Power (n=9568 and d=4), Protein (n=45730 and d=9), and Red wine (n=1599 and d=11).

As shown in Tab. 2, SGP performs best on low-dimensional data, such as Boston and Energy. In contrast, DGP exhibits superior performance on more complex datasets such as Energy and Concrete. However, JCM consistently outperforms most of the baseline models on higher-dimensional datasets such as Protein, Kin8nm, and Red Wine. The log-likelihood estimation results validate the superior expressiveness of JCM in capturing high-dimensional aleatoric uncertainty. This superior performance is mainly attributed to the use of GMM, which enables JCM to capture complex uncertainty structures that the typical distribution (i.e., the normal distribution) widely used in GP and BNN may miss. The RMSE results,



(a) fixing $K = 8$ and varying p (b) fixing $K = 8$ and varying p (c) fixing $K = 8$ and varying p

(d) fixing $p = 5$ and varying K (e) fixing $p = 5$ and varying K (f) fixing $p = 5$ and varying K

Figure 3: RMSE values for different embedding dimensions and mixture component numbers across the Boston, Kin8nm, and Protein datasets. The upper row shows the impact of varying p while fixing $K = 8$, and the lower row illustrates the impact of varying K while fixing $p = 5$. The legend provides information about the GMM components.

as illustrated in Tab. 3, further confirm the conclusion of using JCM for higher-dimensional datasets.

5.3 Aleatoric Uncertainty Quantification: Quantile Estimation

To further evaluate the aleatoric uncertainty captured by our proposed JCM approach, we perform quantile regression and construct prediction intervals on several UCI benchmark datasets.

We construct $(1 - \alpha)$ prediction intervals using JCM and compare its performance against four popular baseline methods: conditional Gaussian (Lakshminarayanan et al., 2017), MC-Dropout, quantile forest (Meinshausen and Ridgeway, 2006), and simultaneous quantile regression (SQR, Tagasovska and Lopez-Paz 2019). For each method, we evaluate the prediction interval coverage probability (PICP) and the mean prediction interval

Table 2: Mean and standard error of log likelihoods over 10 splits for various models on UCI datasets. Higher log likelihoods indicate better uncertainty quantification. The best-performing models are highlighted in **bold**.

Dataset	BNN-VI	SGP	NN-Ens	MC-Dropout	DGP	Combined	JCM
Boston	-2.59(0.11)	**-2.40(0.07)**	***-2.45(0.25)***	-2.51(0.31)	-2.49(0.05)	-2.59(0.10)	-2.54(0.20)
Energy	-2.07(0.08)	**-0.63(0.03)**	-1.48(0.31)	-2.01(0.11)	***-0.74(0.02)***	-2.14(0.03)	-1.08(0.22)
Concrete	-3.31(0.05)	***-3.09(0.02)***	**-3.06(0.32)**	-3.11(0.12)	-3.13(0.01)	-3.27(0.04)	-3.23(0.08)
Kin8nm	0.95(0.08)	1.15(0.00)	***1.18(0.03)***	0.95(0.15)	**1.38(0.01)**	1.09(0.04)	0.96(0.03)
Power	-2.89(0.01)	-2.75(0.01)	-2.77(0.04)	-2.89(0.14)	***-2.73(0.01)***	-3.24(0.01)	**-2.71(0.03)**
Protein	-2.91(0.00)	-2.83(0.00)	-2.80(0.02)	-2.93(0.14)	***-2.71(0.00)***	-2.91(0.01)	**-2.39(0.06)**
Red wine	-0.98(0.01)	-0.93(0.01)	-0.93(0.09)	-0.94(0.01)	-0.95(0.01)	***-0.91(0.08)***	**1.71(0.95)**

Table 3: Mean and standard error of rmse over 10 splits.

Dataset	BNN-VI	SGP	NN-Ens	MC-Dropout	DGP	Combined	JCM
Boston	3.45(0.87)	**2.73(0.12)**	3.33(1.33)	3.01(0.99)	***2.92(0.17)***	3.35(0.88)	3.46(1.25)
Energy	1.89(0.02)	***0.47(0.02)***	2.13(0.46)	1.69(0.19)	**0.47(0.01)**	0.61(0.35)	1.07(0.27)
Concrete	5.78(0.21)	5.53(0.12)	5.65(0.55)	***5.33(0.65)***	5.61(0.10)	**4.87(0.75)**	6.67(0.46)
Kin8nm	0.18(0.02)	0.08(0.00)	**0.01(0.01)**	0.12(0.01)	***0.06(0.00)***	0.08(0.00)	0.09(0.00)
Power	4.12(0.45)	**3.79(0.03)**	4.11(0.21)	4.13(0.13)	**3.79(0.03)**	4.16(0.20)	***3.94(0.14)***
Protein	4.67(0.94)	***4.10(0.03)***	4.36(0.07)	4.19(0.08)	**4.00(0.03)**	4.64(0.06)	4.54(0.10)
Red wine	0.69(0.41)	0.62(0.01)	0.67(0.06)	0.64(0.06)	0.63(0.01)	**0.60(0.03)**	***0.63(0.03)***

Table 4: Mean and standard deviation of PICP for 95% prediction intervals, as well as the mean and standard deviation of MPIW. Models unable to achieve the desired PICP bounds are labeled as "none".

	Conditional Gaussian		MC-Dropout		Quantile Forest	
	PICP	MPIW	PICP	MPIW	PICP	MPIW
Concrete	0.95(0.01)	**0.33(0.02)**	none	none	0.97(0.02)	0.77(0.01)
Power	0.96(0.01)	0.21(0.00)	0.86(0.01)	0.22(0.00)	0.98(0.00)	0.32(0.00)
Red wine	0.95(0.01)	0.46(0.01)	none	none	none	none
Energy	0.93(0.03)	0.20(0.01)	0.90(0.03)	0.34(0.01)	0.97(0.02)	0.71(0.02)
Boston	0.94(0.04)	0.30(0.02)	none	none	0.94(0.02)	0.93(0.00)
Kin8nm	0.94(0.01)	0.27(0.01)	none	none	none	none
Yacht	0.94(0.03)	0.49(0.06)	0.93(0.05)	0.25(0.02)	0.93(0.05)	0.85(0.01)

	SQR		JCM-MC		JCM-LS	
	PICP	MPIW	PICP	MPIW	PICP	MPIW
Concrete	0.95(0.02)	0.48(0.02)	0.94(0.02)	0.34(0.02)	0.94(0.02)	0.34(0.02)
Power	0.95(0.01)	0.20(0.00)	0.96(0.01)	**0.03(0.00)**	0.95(0.01)	**0.03(0.00)**
Red wine	0.96(0.01)	0.48(0.02)	0.93(0.02)	0.23(0.02)	0.94(0.03)	**0.21(0.02)**
Energy	0.95(0.02)	0.39(0.06)	0.93(0.04)	0.14(0.02)	0.93(0.05)	**0.13(0.02)**
Boston	0.94(0.02)	0.44(0.03)	0.94(0.04)	**0.28(0.03)**	0.94(0.04)	**0.28(0.03)**
Kin8nm	0.95(0.01)	0.39(0.02)	0.97(0.01)	0.10(0.00)	0.95(0.01)	**0.09(0.00)**
Yacht	0.93(0.06)	0.64(0.09)	0.98(0.02)	0.31(0.08)	0.95(0.04)	**0.29(0.08)**

width (MPIW) across seven UCI datasets: Concrete(n=1030 and d=8), Power(n=9568 and d=4), Red wine (n=1599 and d=11), Energy(n=768 and d=8), Boston(n=506 and d=13), Kin8nm(n=8192 and d=8), and Yacht(n=308 and d=6).

Table 5: Mean and standard deviation of PICP for 80% prediction intervals, as well as the mean and standard deviation of MPIW. Models unable to achieve the desired PICP bounds are labeled as "none".

	Conditional Gaussian		MC-Dropout		Quantile Forest	
	PICP	MPIW	PICP	MPIW	PICP	MPIW
Concrete	0.80(0.02)	0.22(0.02)	none	none	0.87(0.03)	0.54(0.00)
Power	0.82(0.01)	0.13(0.00)	0.72(0.01)	0.14(0.00)	0.90(0.01)	0.20(0.00)
Red wine	0.81(0.03)	0.30(0.01)	none	none	none	none
Energy	0.70(0.03)	0.13(0.01)	0.84(0.03)	0.22(0.00)	0.84(0.04)	0.32(0.01)
Boston	0.84(0.05)	0.20(0.01)	none	none	0.79(0.04)	0.49(0.01)
Kin8nm	0.78(0.01)	0.17(0.00)	none	none	none	none
Yacht	0.91(0.03)	0.32(0.04)	0.72(0.08)	0.17(0.01)	0.82(0.06)	0.56(0.01)

	SQR		JCM-MC		JCM-LS	
	PICP	MPIW	PICP	MPIW	PICP	MPIW
Concrete	0.80(0.04)	0.28(0.02)	0.80(0.03)	**0.22(0.01)**	0.78(0.03)	**0.22(0.01)**
Power	0.80(0.01)	0.13(0.00)	0.82(0.02)	**0.02(0.00)**	0.80(0.02)	**0.02(0.00)**
Red wine	0.81(0.02)	0.29(0.01)	0.82(0.04)	0.15(0.01)	0.82(0.06)	**0.13(0.01)**
Energy	0.76(0.04)	0.20(0.02)	0.79(0.06)	0.09(0.01)	0.76(0.06)	**0.08(0.02)**
Boston	0.78(0.05)	0.20(0.01)	0.82(0.06)	0.18(0.02)	0.80(0.07)	**0.17(0.02)**
Kin8nm	0.79(0.01)	0.20(0.01)	0.85(0.01)	**0.06(0.00)**	0.83(0.01)	**0.06(0.00)**
Yacht	0.75(0.09)	0.33(0.05)	0.88(0.08)	0.20(0.05)	0.83(0.07)	**0.19(0.06)**

We employ two variants of JCM for quantile estimation: JCM with Monte Carlo sampling (JCM-MC) and JCM with line search (JCM-LS). JCM-LS utilizes an approximate closed-form solver for quantiles, enhancing computational efficiency and estimation accuracy, while JCM-MC relies on empirical estimation through Monte Carlo sampling. The baseline methods are implemented following their respective original methodologies. Conditional Gaussian assumes a Gaussian posterior distribution for the target variable, MC-Dropout uses dropout during inference to estimate uncertainty, quantile forest is an ensemble tree-based method for quantile regression, and SQR estimates multiple quantiles simultaneously using a single model. Notably, for the tree-based methods, we cross-validate the number of trees and the minimum number of examples to make a split (Tagasovska and Lopez-Paz, 2019). All models are trained and evaluated using the same train-test splits to ensure a fair comparison. Each experiment is repeated 10 times with different random splits to obtain reliable statistics.

As presented in Tabs. 4 and 5, our JCM approach demonstrates superior performance in constructing prediction intervals compared to baseline methods. Specifically, for the 95% prediction intervals, JCM-LS (line search variant) achieves PICP values closer to the desired 0.95 across all datasets while maintaining narrower MPIW compared to other methods. Similarly, for the 80% prediction intervals, JCM consistently provides well-calibrated coverage probabilities with competitive interval widths.

These results validate the contribution of our JCM approach, which allows us to capture complex uncertainty structures more effectively. The ability of JCM to produce accurate and precise prediction intervals across diverse datasets highlights its super uncertainty expressiveness.

5.4 Epistemic Uncertainty Quantification: Active Learning

The effectiveness of active learning critically depends on the accurate estimation of predictive epistemic uncertainty (Settles, 2009). In this subsection, we use the lower bound entropy criterion proposed in Eq. (22) for selecting new data points. We utilize the same network architectures and datasets as the UCI benchmark (Asuncion, 2007).

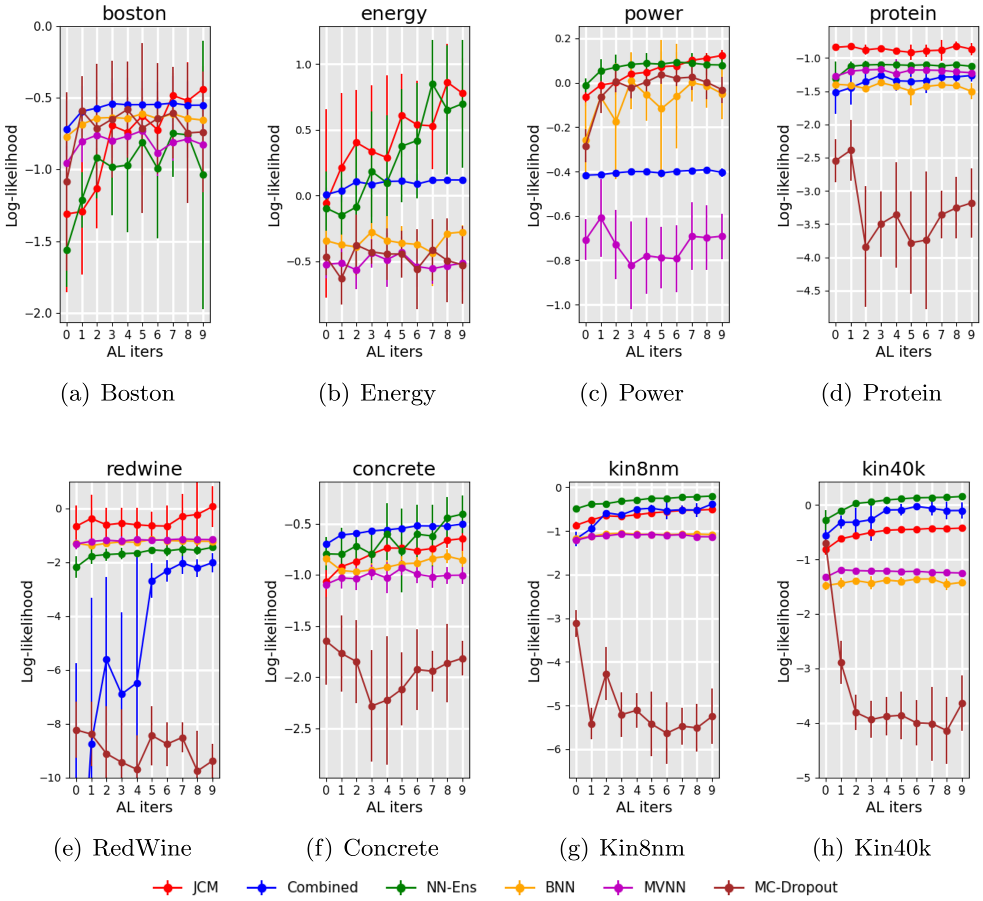
To accommodate datasets of varying sizes, we adopt a size-dependent data splitting strategy. Specifically, small datasets ($N < 5000$) are divided into 20% for initial training, 60% for the acquisition pool, and 20% for testing. For medium-sized datasets ($10000 \approx N < 20000$), we use a 10%:70%:20% split strategy, while large datasets ($N > 20000$) are split into 5% initial training, 75% pool, and 20% testing sets. This adaptive splitting strategy differs from the common practice of using a fixed 20% initial training set across all datasets, which has been shown to limit the effectiveness of active sampling on larger datasets (Detlefsen et al., 2019). Reducing the size of the initial training set for medium and large datasets allows for the acquisition of more informative samples during the active learning process, thereby better leveraging the potential of active learning.

In each active learning iteration, we first train a model using the current training set and then evaluate its performance on the test set. Subsequently, we estimate the uncertainty of all the remaining data points in the pool set. We select the n points with the highest uncertainty (based on the lower bound entropy criterion) and add them to the training set. Here, n is set to 5% of the initial pool size, and this process is repeated 10 times. The entire procedure is conducted across 10 random training-test splits to compute standard errors. We compare our proposed JCM method against four popular alternatives: BNN, MVNN, MC-Dropout, Ens-NN, and Combined.

Fig. 4 displays the average log likelihood of each method across all 8 datasets: Boston (n=506 and d=13), Energy (n=768 and d=8), Power (n=9568 and d=4), Protein (n=45730 and d=9), Red wine (n=1599 and d=11), Concrete (n=1030 and d=8), Kin8nm (n=8192 and d=8), and Kin40k(n=40000 and d=8). While no single baseline method dominates across all examples, Ens-NN and the Combined method generally perform well but yield inferior results on certain datasets. In contrast, our JCM method consistently ranks among the top tier in all examples and outperforms all baseline models for five out of eight datasets, demonstrating significantly faster learning of the log likelihood. This suggests that our model accurately estimates the epistemic uncertainty of data points in the pool set, compared to baseline methods.

5.5 Epistemic Uncertainty Quantification: Bayesian Optimization

To evaluate the effectiveness of our proposed JCM approach in capturing epistemic uncertainty, we conduct BO experiments on a variety of benchmark functions. BO leverages uncertainty estimates to efficiently explore and exploit the search space, making it an ideal framework to assess the quality of epistemic uncertainty quantification provided by different surrogate models. We compare JCM against two prominent surrogate models: GP and BNN using HMC (BNN-HMC).



(a) Boston (b) Energy (c) Power (d) Protein

(e) RedWine (f) Concrete (g) Kin8nm (h) Kin40k

Figure 4: Average test log likelihood and standard errors.

The benchmark functions used in our experiments are categorized into synthetic and real-world datasets. The synthetic benchmarks include the Ackley function (Surjanovic and Bingham, 2024), a multidimensional function with numerous local minima, and the Styblinski-Tang (or Stybtang for short) function (Surjanovic and Bingham, 2024), a multimodal function with multiple local minima. Both are commonly used benchmark functions to test optimization algorithms. We set the input ranging of the Ackley function as [-5 5], following the settings given in Stybtang (Surjanovic and Bingham, 2024). The real-world benchmarks encompass the Mopta vehicle design problem (Jones, 2008), which aims to minimize the mass of a vehicle with respect to 124 design variables and subject to 68 performance constraints, the hyper-parameter tuning problem of support vector machine (SVM) (Eriksson, 2021), involving 3 regularization parameters and 385 length-scale parameters, and the rover trajectory planning problem (Wang et al., 2017), where the objective is to find an optimal trajectory determined by the locations of 30 waypoints in a 2D environment. All benchmark functions are converted to maximization problems to align with the objective of BO, which seeks to identify the maximum value of the objective function. The dimensionality of these functions is varied to assess the scalability and performance of the surrogate models in high-dimensional spaces.

In the Bayesian optimization experiments, we use JCM with various acquisition functions, including PI, EI, UCB, and the entropy-based UCB proposed in Eq. (25). The initial dataset size for the BO experiment is 30 and the optimization process is terminated after 50 epochs. The experiment result is illustrated in Fig. 5. Among the acquisition functions used for BO based on JCM, UCB (especially the entropy-based one) achieves better performance. This suggests that incorporating entropy into the UCB acquisition function enhances the ability of JCM to balance exploration and exploitation more effectively.

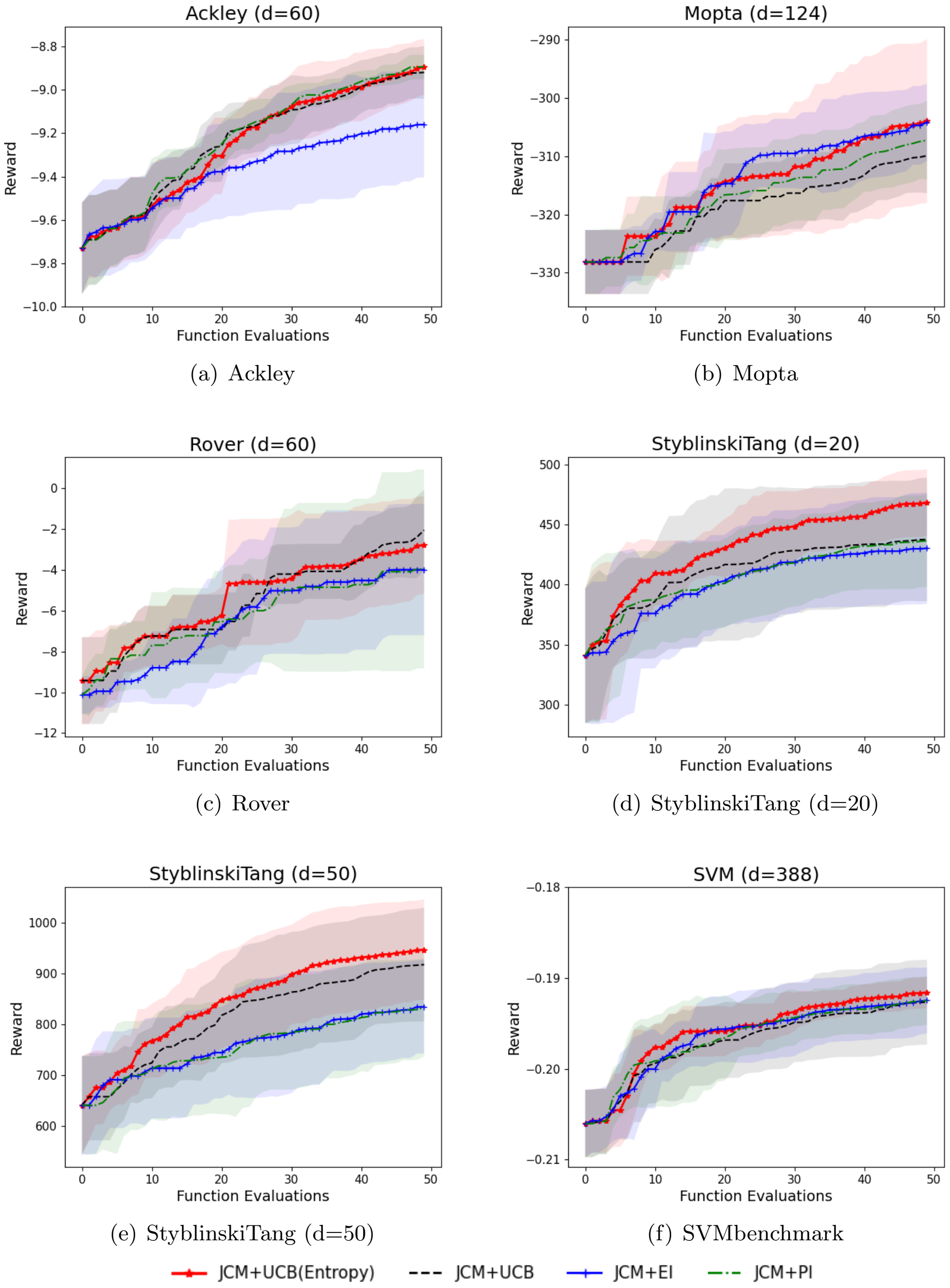
Building on the insights from the initial experiment, we focused solely on the UCB acquisition function in the next phase to compare JCM with the two other surrogate models (GP and BNN-HMC) across the aforementioned benchmark functions. The optimization performance, as depicted in Fig. 6, shows that BO based on JCM consistently outperforms BO based on GP and BNN-HMC in high-dimensional and complex scenarios such as Mopta and SVMbenchmark. This superior performance of JCM is primarily attributed to its enhanced capability in modeling and expressing high-dimensional uncertainty.

To further verify this advantage, we compared the uncertainty quantification accuracy of these surrogate models on the benchmark functions, as shown in Fig. 7. The proposed JCM approach maintains robust performance in most cases, especially in high-dimensional settings. This consistent superiority highlights JCM's enhanced ability to model and utilize high-dimensional epistemic uncertainty effectively, leading to more efficient exploration and exploitation within the BO framework.

6. Conclusion

In this paper, we introduced the JCM approach for uncertainty quantification in high-dimensional regression tasks, addressing the limitations of traditional models such as GP and BNN. By formatting a GMM for posterior estimation, JCM effectively captures complex uncertainty structures, demonstrating superior performance across various high-dimensional and complex benchmark functions. Our extensive experiments on regression, quantile estimation, active learning, and BO highlighted JCM's enhanced ability to model high-dimensional aleatoric and epistemic uncertainties, leading to more accurate predictive distributions and efficient optimization processes.

Further research will explore potential ways to extend and improve the proposed model. First, nonlinear dimension reduction models, such as the variational autoencoder (Vaswani et al., 2017) and the transformer (Kingma et al., 2019), will be explored as potential replacements for the current linear dimension reduction model, which could enhance the model's capacity to capture complex functional relationships and uncertainty structure. Second, we will explore some advanced probabilistic models like the Dirichlet process mixture model (Catalano et al., 2022) as alternatives to GMM. These models have the potential to provide more finely tuned representations of uncertainty structures. Third, expanding the proposed approach to handle multiple output scenarios (Li and Zhou, 2016; Li et al., 2018) would broaden its applicability. Lastly, our future research will examine the discrimination of aleatoric and epistemic uncertainty (Valdenegro-Toro and Mori, 2022) to provide more in-depth insights into the different sources of uncertainty within the regression model.



(a) Ackley

(b) Mopta

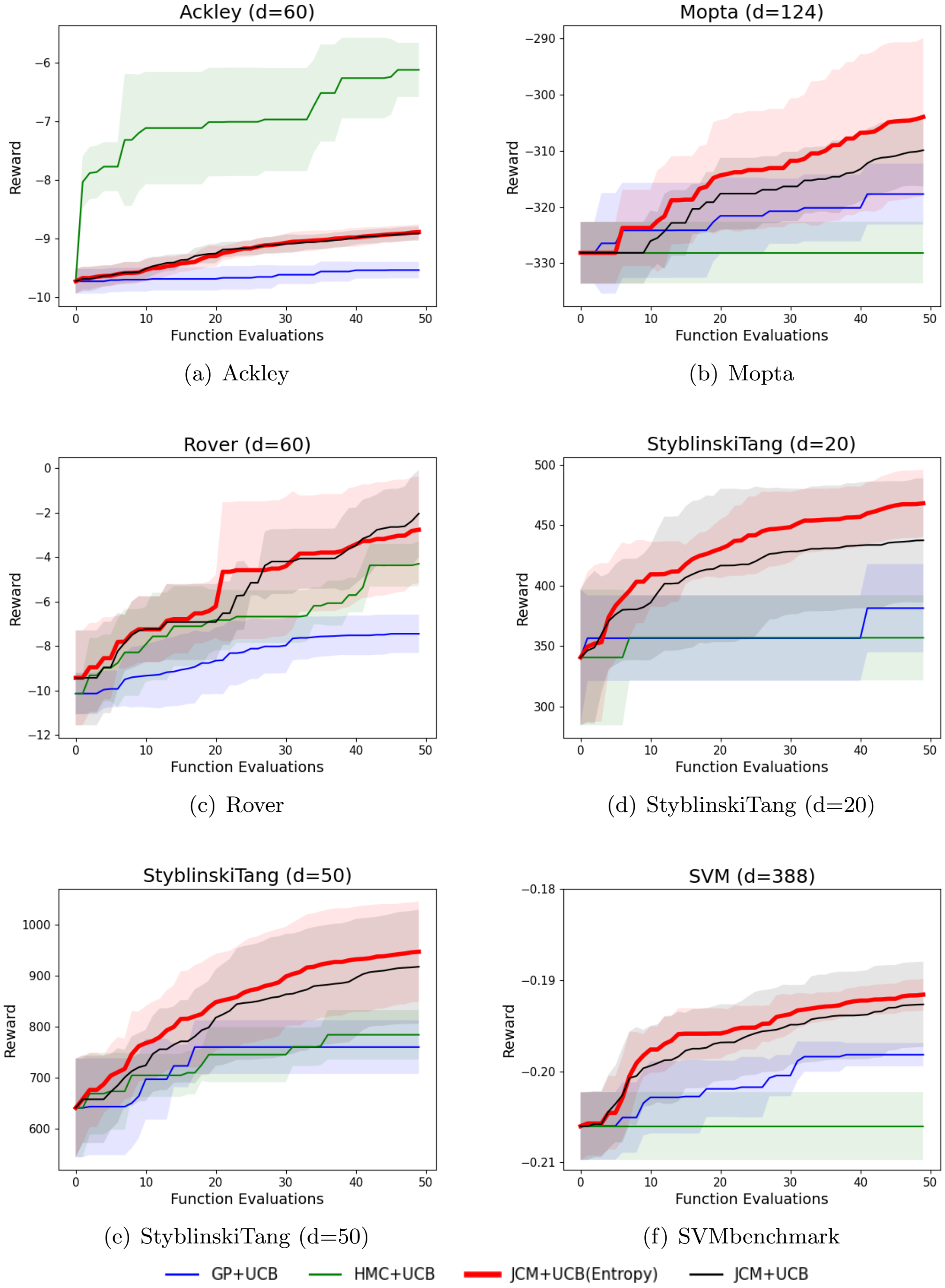
(c) Rover

(d) StyblinskiTang (d=20)

(e) StyblinskiTang (d=50)

(f) SVMbenchmark

Figure 5: Performance of BO with different acquisition functions. For each benchmark function, we use d to denote the number of input dimensions. The figure plots the mean and one standard error of the mean over 5 trials.



(a) Ackley

(b) Mopta

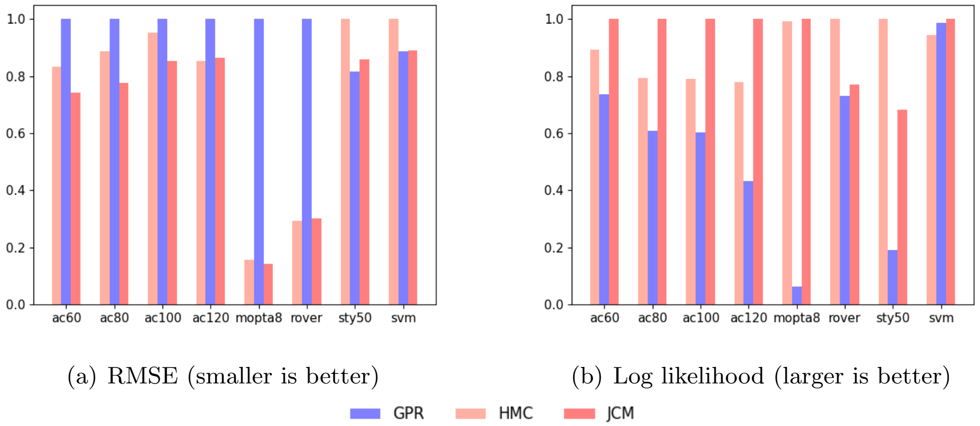
(c) Rover

(d) StyblinskiTang (d=20)

(e) StyblinskiTang (d=50)

(f) SVMbenchmark

Figure 6: BO on high-dimensional synthetic benchmarks for the different surrogate models. For each benchmark function, we include d for the number of input dimensions. We plot the mean and one standard error of the mean over 5 trials.



(a) RMSE (smaller is better)

(b) Log likelihood (larger is better)

Figure 7: The normalized regression test rmse and log likelihood for high-dimensional input spaces. Reported is the mean over 5 splits. The x-axis represents different datasets and dimensions, for example ac60 represents the dataset Ackley with input dim 60 and sty50 represents the dataset StyblinskiTang with input dim 50.

Acknowlegments

This work was sponsored by National Key R&D Program of China No. 2021YFB3802103.

Appendix

A. Proof of Theorem 1

Recall that, for continuous random variables, the differential entropy is defined as:

$$H(\boldsymbol{x}) = \int -\mathcal{P}(\boldsymbol{x}) \log \mathcal{P}(\boldsymbol{x}) d\boldsymbol{x}.$$

For Gaussian distributions, the entropy has a well-known closed-form solution. For example, the differential entropy of a multi-dimensional Gaussian variable $\boldsymbol{x} \sim \mathcal{N}(\mu, \Sigma_{\boldsymbol{x}})$ can be represented by the logarithmic determinant of its covariance matrix:

$$H(\boldsymbol{x}) = \frac{1}{2} \log |\Sigma_{\boldsymbol{x}}| + \frac{d}{2}(\log 2\pi + 1)$$

where $|\cdot|$ denotes the determinant function, and d is the dimension of variables.

The proof mainly uses Jensen's inequality on the convex function $-\log(\cdot)$. We first prove the upper bound:

$$\begin{aligned}
H(y|\boldsymbol{x}) &= \mathbb{E}\left[-\log \mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{z})\right] = \mathbb{E}\left[-\log \sum_{k=1}^{K} \pi_k^*(\boldsymbol{z}) \mathcal{N}(y|\mu_{k,y|\boldsymbol{z}}, \Sigma_{k,y|\boldsymbol{z}})\right] \\
&\leq \mathbb{E}\left[-\sum_{k=1}^{K} \pi_k^*(\boldsymbol{z}) \log \mathcal{N}(y|\mu_{k,y|\boldsymbol{z}}, \Sigma_{k,y|\boldsymbol{z}})\right] \\
&= \sum_{k=1}^{K} \pi_k^*(\boldsymbol{z}) \mathbb{E}\left[-\log \mathcal{N}(y|\mu_{k,y|\boldsymbol{z}}, \Sigma_{k,y|\boldsymbol{z}})\right] \\
&= \sum_{k=1}^{K} \pi_k^*(\boldsymbol{z}) \frac{\log(2\pi) + \log\left|\Sigma_{k,y|\boldsymbol{z}}\right| + 1}{2} = H_u(y|\boldsymbol{x})
\end{aligned}$$

because

$$-\log\left(\sum_{k=1}^{K} \pi_k^*(\boldsymbol{z}) \mathcal{N}(y|\mu_{k,y|\boldsymbol{z}}, \Sigma_{k,y|\boldsymbol{z}})\right) \leq -\sum_{k=1}^{K} \pi_k^*(\boldsymbol{z}) \log \mathcal{N}(y|\mu_{k,y|\boldsymbol{z}}, \Sigma_{k,y|\boldsymbol{z}}).$$

The lower bound is

$$\begin{aligned}
H(y|\boldsymbol{x}) &= \mathbb{E}\left[-\log \mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{z})\right] = -\int \sum_{k=1}^{K} \pi_k^*(\boldsymbol{z}) \mathcal{N}(y|\mu_{k,y|\boldsymbol{z}}, \Sigma_{k,y|\boldsymbol{z}}) \log \mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{z}) dy \\
&= -\sum_{k=1}^{K} \pi_k^*(\boldsymbol{z}) \int \mathcal{N}(y|\mu_{k,y|\boldsymbol{z}}, \Sigma_{k,y|\boldsymbol{z}}) \log \mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{z}) dy \\
&\geq -\sum_{k=1}^{K} \pi_k^*(\boldsymbol{z}) \log\left(\int \mathcal{N}(y|\mu_{k,y|\boldsymbol{z}}, \Sigma_{k,y|\boldsymbol{z}}) \mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{z}) dy\right) \\
&= -\sum_{i=1}^{K} \pi_i^*(\boldsymbol{z}) \log\left(\int \mathcal{N}(y|\mu_{i,y|\boldsymbol{z}}, \Sigma_{i,y|\boldsymbol{z}}) \sum_{j=1}^{K} \pi_j^*(\boldsymbol{z}) \mathcal{N}(y|\mu_{j,y|\boldsymbol{z}}, \Sigma_{j,y|\boldsymbol{z}}) dy\right) \\
&= -\sum_{i=1}^{K} \pi_i^*(\boldsymbol{z}) \log\left(\int \sum_{j=1}^{K} \pi_j^*(\boldsymbol{z}) \mathcal{N}(y|\mu_{i,y|\boldsymbol{z}}, \Sigma_{i,y|\boldsymbol{z}}) \mathcal{N}(y|\mu_{j,y|\boldsymbol{z}}, \Sigma_{j,y|\boldsymbol{z}}) dy\right) \\
&= -\sum_{i=1}^{K} \pi_i^*(\boldsymbol{z}) \log\left(\sum_{j=1}^{K} \pi_j^*(\boldsymbol{z}) \int \mathcal{N}(y|\mu_{i,y|\boldsymbol{z}}, \Sigma_{i,y|\boldsymbol{z}}) \mathcal{N}(y|\mu_{j,y|\boldsymbol{z}}, \Sigma_{j,y|\boldsymbol{z}}) dy\right) \\
&= -\sum_{i=1}^{K} \pi_i^*(\boldsymbol{z}) \log\left(\sum_{j=1}^{K} \pi_j^*(\boldsymbol{z}) \mathcal{N}(\mu_{i,y|\boldsymbol{z}}|\mu_{j,y|\boldsymbol{z}}, \Sigma_{i,y|\boldsymbol{z}} + \Sigma_{j,y|\boldsymbol{z}})\right) = H_l(y|\boldsymbol{x}),
\end{aligned}$$

because

$$-\int \mathcal{N}(y|\mu_{k,y|\boldsymbol{z}}, \Sigma_{k,y|\boldsymbol{z}}) \log \mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{z}) dy \geq -\log\left(\int \mathcal{N}(y|\mu_{k,y|\boldsymbol{z}}, \Sigma_{k,y|\boldsymbol{z}}) \mathcal{P}_{\boldsymbol{\phi}}(y|\boldsymbol{z}) dy\right).$$

This completes the proof.

References

P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

Yarin Gal Alex Kendall. What uncertainties do we need in bayesian deep learning for computer vision? 2017.

A. Asuncion. Uci machine learning repository, university of california, irvine, school of information and computer sciences. *http://www.ics.uci.edu/ mlearn/MLRepository.html*, 2007.

David Barber and Christopher M Bishop. Ensemble learning in bayesian neural networks. *Nato ASI Series F Computer and Systems Sciences*, 168:215–238, 1998.

J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. Algorithms for hyper-parameter optimization. In *International Conference on Neural Information Processing Systems*, 2011.

Christopher M Bishop. Mixture density networks. 1994.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

Silvere Bonnabel. Stochastic gradient descent on riemannian manifolds. *IEEE Transactions on Automatic Control*, 58(9):2217–2229, 2013.

Eric Brochu, Vlad M. Cora, and Nando De Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *Computer Science*, 2010.

B.Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016.

Marta Catalano, Pierpaolo De Blasi, Antonio Lijoi, and Igor Prünster. Posterior asymptotics for boosted hierarchical dirichlet process mixtures. *The Journal of Machine Learning Research*, 23(1):3471–3493, 2022.

Andreas Damianou and Neil D. Lawrence. Deep Gaussian processes. In Carlos M. Carvalho and Pradeep Ravikumar, editors, *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, volume 31 of *Proceedings of Machine Learning Research*, pages 207–215, Scottsdale, Arizona, USA, 29 Apr–01 May 2013. PMLR. URL `https://proceedings.mlr.press/v31/damianou13a.html`.

A. P. Dempster. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 39, 1977.

Nicki Skafte Detlefsen, Martin Jørgensen, and Søren Hauberg. Reliable training and estimation of variance networks. In *Advances in Neural Information Processing Systems*, 2019.

Bradley Efron and Robert Tibshirani. Cross-validation and the bootstrap: Estimating the error rate of a prediction rule. *Handbook of Systemic Autoimmune Diseases*, 176, 1995.

David Eriksson. High-dimensional bayesian optimization with sparse axis-aligned subspaces (supplementary material). 2021.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. *ArXiv*, abs/1703.02910, 2017.

Dibet Garcia, Joao Carias, Telmo Adao, Rui Jesus, Antonio Cunha, and Luis G. Magalhaes. Ten years of active learning techniques and object detection: A systematic review. *Applied Sciences-Basel*, 13(19), OCT 2023. doi: 10.3390/app131910667.

Roger Ghanem, David Higdon, and Houman Owhadi. *Handbook of uncertainty quantification*. Springer, 2017.

Soumya Ghosh, Jiayu Yao, and Finale Doshi-Velez. Model selection in bayesian neural networks via horseshoe priors. *J. Mach. Learn. Res.*, 20(182):1–46, 2019.

Ryan Giordano, Tamara Broderick, and Michael I Jordan. Covariances, robustness and variational bayes. *Journal of machine learning research*, 19(51), 2018.

Ethan Goan and Clinton Fookes. Bayesian neural networks: An introduction and survey. *Case Studies in Applied Bayesian Data Science: CIRM Jean-Morlet Chair, Fall 2018*, pages 45–87, 2020.

Paul Goldberg, Christopher Williams, and Christopher Bishop. Regression with input-dependent noise: A gaussian process treatment. *Advances in neural information processing systems*, 10, 1997.

Alex Graves. Practical variational inference for neural networks. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011. URL https://proceedings.neurips.cc/paper_files/paper/2011/file/7eb3c8be3d411e8ebfab08eba5f49632-Paper.pdf.

José Miguel Henrández-Lobato, Matthew W. Hoffman, and Zoubin Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 1*, NeurIPS'14, pages 918–926, Cambridge, MA, USA, 2014.

James Hensman, Nicolo Fusi, and Neil D Lawrence. Gaussian processes for big data. In *Uncertainty in Artificial Intelligence*, page 282. Citeseer, 2013.

José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International conference on machine learning*, pages 1861–1869. PMLR, 2015.

Matthew D. Hoffman, David M. Blei, Chong Wang, and John William Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14:1303–1347, 2013.

Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *ArXiv*, abs/1112.5745, 2011.

Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning*, 110:457–506, 2021.

Pavel Izmailov, Sharad Vikram, Matthew D. Hoffman, and Andrew Gordon Wilson. What are bayesian neural network posteriors really like? 2021.

D. R. Jones. Large-scale multi-disciplinary mass optimization in the auto industry. 2008.

Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13:455–492, 1998.

Diederik P Kingma, Max Welling, et al. An introduction to variational autoencoders. *Foundations and Trends in Machine Learning*, 12(4):307–392, 2019.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NeurIPS*, 2017.

Miguel Lázaro-Gredilla and Michalis K Titsias. Variational heteroscedastic gaussian process regression. In *ICML*, pages 841–848, 2011.

John M Lee. *Riemannian manifolds: an introduction to curvature*, volume 176. Springer Science & Business Media, 2006.

Yongxiang Li and Qiang Zhou. Pairwise meta-modeling of multivariate output computer models using nonseparable covariance function. *Technometrics*, 58(4):483–494, 2016.

Yongxiang Li, Qiang Zhou, Xiaohu Huang, and Li Zeng. Pairwise estimation of multivariate gaussian process models with replicated observations: Application to multivariate profile monitoring. *Technometrics*, 60(1):70–78, 2018.

David J. C. Mackay. A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472, 1992.

Nicolai Meinshausen and Greg Ridgeway. Quantile regression forests. *Journal of machine learning research*, 7(6), 2006.

Kevin P Murphy. *Machine learning: a probabilistic perspective.* MIT press, 2012.

Radford M Neal. Mcmc using hamiltonian dynamics. *Eprint Arxiv*, 2012.

David A. Nix and Andreas S. Weigend. Estimating the mean and variance of the target probability distribution. *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, 1:55–60 vol.1, 1994.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006. ISBN 978-0-262-18253-9.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, 2014.

Hugh Salimbeni and Marc Deisenroth. Doubly stochastic variational inference for deep gaussian processes. *Advances in Neural Information Processing Systems*, 30, 2017.

Annie Sauer, Robert B Gramacy, and David Higdon. Active learning for deep gaussian process surrogates. *Technometrics*, 65(1):4–18, 2023.

Warren Scott, Peter Frazier, and Warren Powell. The correlated knowledge gradient for simulation optimization of continuous parameters using gaussian process regression. *SIAM Journal on Optimization*, 21(3):996–1026, 2011.

Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

Ashish Kumar Shakya, Gopinatha Pillai, and Sohom Chakrabarty. Reinforcement learning algorithms: A brief survey. *Expert Systems with Applications*, 231, NOV 30 2023. ISSN 0957-4174. doi: 10.1016/j.eswa.2023.120495.

Guohao Shen, Yuling Jiao, Yuanyuan Lin, Joel L Horowitz, and Jian Huang. Nonparametric estimation of non-crossing quantile regression process with deep requ neural networks. *Journal of Machine Learning Research*, 25(88):1–75, 2024.

Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical bayesian optimization of machine learning algorithms. In *International Conference on Neural Information Processing Systems*, 2012.

Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Md. Mostofa Ali Patwary, Prabhat Prabhat, and Ryan P. Adams. Scalable bayesian optimization using deep neural networks. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 2171–2180. JMLR.org, 2015.

Niranjan Srinivas, Andreas Krause, Sham Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML, pages 1015–1022, USA, 2010.

S. Surjanovic and D. Bingham. Virtual library of simulation experiments: Test functions and datasets. 2024.

Taiji Suzuki and Masashi Sugiyama. Sufficient dimension reduction via squared-loss mutual information estimation. *Neural Computation*, 25:725–758, 2010.

Natasa Tagasovska and David Lopez-Paz. Single-model uncertainties for deep learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Hao Xiong Tao Yu, Zhaonian Zou. Can uncertainty quantification enable better learning-based index tuning? 2020.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

A. Törn and A. Zilinskas. Global optimization. *lecture notes in computer science*, 350, 1989.

Matias Valdenegro-Toro and Daniel Saromo Mori. A deeper look into aleatoric and epistemic uncertainty disentanglement. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1508–1516. IEEE, 2022.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Xilu Wang, Yaochu Jin, Sebastian Schmitt, and Markus Olhofer. Recent advances in bayesian optimization. *ACM Computing Surveys*, 55(13S), DEC 2023. ISSN 0360-0300. doi: 10.1145/3582078.

Zi Wang, Clement Gehring, Pushmeet Kohli, and Stefanie Jegelka. Batched large-scale bayesian optimization in high-dimensional spaces. 2017.

Zi Wang, George E Dahl, Kevin Swersky, Chansoo Lee, Zachary Nado, Justin Gilmer, Jasper Snoek, and Zoubin Ghahramani. Pre-trained gaussian processes for bayesian optimization. *Journal of Machine Learning Research*, 25(212):1–83, 2024.

James T. Wilson, Riccardo Moriconi, Frank Hutter, and Marc Peter Deisenroth. The reparameterization trick for acquisition functions. 2017.

Yaniv Yacoby, Weiwei Pan, and Finale Doshi-Velez. Mitigating the effects of non-identifiability on inference for bayesian neural networks with latent variables. *The Journal of Machine Learning Research*, 23(1):11114–11167, 2022.

Xing Yan, Yonghua Su, and Wenxuan Ma. Ensemble multi-quantiles: Adaptively flexible distribution prediction for uncertainty quantification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(11):13068–13082, 2023.