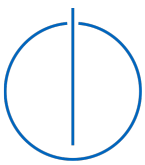
SCHOOL OF COMPUTATION,
INFORMATION AND TECHNOLOGY

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics

Single- and Multi-Fidelity Gaussian Process Regression Models with Uncertain Inputs

Julian Walker



SCHOOL OF COMPUTATION,
INFORMATION AND TECHNOLOGY

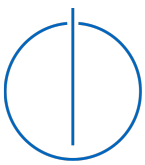
TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics

Single- and Multi-Fidelity Gaussian Process Regression Models with Uncertain Inputs

Einzel- und Multi-Fidelity-Gaußprozess-regressionsmodelle mit unsicheren Eingaben

Author:	Julian Walker
Supervisor:	Prof. Dr. Felix Dietrich
Advisor:	Vladyslav Fediukov, M.Sc.
Submission Date:	30.09.2024



I confirm that this bachelor's thesis in informatics is my own work and I have documented all sources and material used.

Feldkirchen, 30.09.2024 Julian Walker

Acknowledgements

I would like to express my deepest gratitude to Vladyslav for his encouraging and supporting guidance throughout the entire process of this thesis. I am particularly thankful to him for the fascinating topic he provided and for his continuous feedback. I also extend my sincere appreciation to Prof. Dietrich for his invaluable advice.

Additionally, I am deeply grateful to my mother and sister for always being there for me and supporting me. Lastly, I would like to thank my good friend Dominik for his spell-checking, which helped me enhance my writing.

Abstract

Gaussian process (GP) models are widely used for regression tasks. Recently, several authors have developed GP models that fuse information from sources of different fidelity levels, with successful application in building a surrogate model for the dynamics of a rover wheel traversing soft soil. In its current version, noise in the inputs, which are given as signals, is removed by a low-pass filter. In this thesis, we explore an alternative approach by treating the inputs as randomly distributed around the smoothed signals, which might recover valuable information in the high-frequency components of the input signals. Since standard GP models assume deterministic inputs, they are not suited for this task. Nonetheless, multiple authors have developed GP regression models that specifically address uncertain inputs. We review a set of these models and evaluate their predictive performance against standard GP regression models trained on smoothed input signals in a synthetic example. Furthermore, we discuss two multi-fidelity GP regression models and develop extensions to make them account for uncertain inputs. Using these models, we conduct another synthetic experiment to assess the predictive performance of our extensions in comparison to the baseline models trained on smoothed input signals. The results of both synthetic examples suggest that employing input-uncertainty-aware models is the preferable choice when the input data consists of noisy signals. Furthermore, we apply our best-performing multi-fidelity model to rover wheel-soil interaction data. Preliminary results suggest that our model may serve as a surrogate model with enhanced predictive performance. However, further experiments should be conducted to substantiate this claim. Moreover, the insights gained from our synthetic experiments may also assist researchers from other fields in making informed decisions about model selection for more general applications involving uncertain inputs.

Contents

1 Introduction

Machine learning algorithms are widely used in areas that require high-accuracy predictions, such as health care [1], disaster forecasting [2], and rover locomotion on planetary bodies [3]. In the latter case, a single mistake can lead to the failure of the entire mission, as extraterrestrial vehicles cannot be repaired or maintained during their missions [4]. For instance, after six successful years of Mars exploration, NASA lost contact to their Spirit rover. A few months prior, the rover had become embedded in the martian soil after encountering an unexpected hazard. Despite numerous efforts to free Spirit, the rover remained trapped in a position where it could only harness an insufficient amount of solar energy, which is assumed to be a major contributer to the shutdown of the rover [5]. Since then, considerable advancements have been made in gaining a deeper understanding of the locomotion of a rover on loose terrain, and a growing body of literature focuses on machine learning-based methods to model the underlying mechanics [3].

One particular data-driven approach to modeling rover wheel locomotion, developed by Fediukov et al. [4] and later by Ravi, Fediukov, et al. [6], employs Gaussian process (GP) regression models, which generate a distribution over predictions rather than point estimates. They also propose to leverage data from sources of different accuracy to build an adequate surrogate model for the complex mechanics of a rover wheel traversing soft soil. For this purpose, the authors utilize multi-fidelity GP models, which adapt traditional GP regression models to fuse data of different sources with varying fidelities.

While GP regression models typically account for noisy observations of the target values in a given dataset, they assume that the input data to the model is deterministic. However, when this assumption is violated, the predictive performance of the model might significantly suffer. Even with simpler linear regression models, it is well known that untreated measurement noise in the inputs can drastically reduce prediction accuracy [7].

To address this issue, several authors have developed GP regression models that account for uncertainty in the inputs. In this thesis, we will review a selection of these input-uncertainty-aware GP models for general regression tasks and compare their predictive performance in the presence of input uncertainty using a synthetic example. In addition, we develop extensions to two multi-fidelity GP models to incorporate input

uncertainty. We also assess their predictive performance on a noise-corrupted version of a synthetic example that is commonly examined in the literature. Our experiments aim to explore a possible extension to the experiments about rover wheel-soil interaction modeling carried out by Ravi, Fediukov, et al. [6]. In their study, the data consists of a set of signals generated from simulations or observed in real-world experiments, and the their model is designed to predict the traction force acting on a rover wheel. Since only constant application of force can affect the movement of a wheel, the authors smooth all signals, treating high-frequency components as noise. In our work, we specifically explore whether we can obtain better results by interpreting the inputs as randomly distributed around smoothed signals. We design our toy experiments to address this question, and as they yielded positive results, we apply the best-performing multi-fidelity model to similar data to that used in the experiments of Ravi, Fediukov, et al [6].

Our thesis is structured as follows. In the next chapter, we introduce essential preliminaries about input noise models, Bayesian smoothing, and both single- and multi-fidelity GP regression models. We also discuss input-uncertainty-aware GP models for tasks other than standard regression as well as related multi-fidelity GP models that are not the focus of our work. Chapter 3 details all single-fidelity input-uncertainty-aware GP models that we consider, along with our extensions to multi-fidelity models. This chapter also includes all experiments that we conduct. Finally, chapter 4 presents the conclusions of our work.

2 Background

2.1 Noise in Regression Models

In a standard parametric regression setting, we are given a training dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)|i = 1, ..., N\}$ of N observations of inputs $\mathbf{x}_i \in \mathbb{R}^D$ and corresponding target values $y_i \in \mathbb{R}$. In order to predict y_* for a new test input x_*, we assume that $\mathcal{D}$ has been generated by the functional relationship

$$y = f(\mathbf{x}, \mathbf{w}) + \varepsilon.$$

Here, f stems from a predefined function space whose elements are parametrized by the parameter vector $\mathbf{w}$. A typical example of such a function space is the space of linear functions $f(\mathbf{x}, \mathbf{w}) = \mathbf{x}^\top \mathbf{w}$, in which case the regression model is referred to as linear regression. The term ε represents additive noise, which is assumed to be independent across different inputs and identically normally distributed with zero mean and variance σ_n^2 [8]. The additive noise term introduces a random component to our model, which accounts for variability around our model. This variability may arise from various sources. For instance, the target values y_i in our dataset may include measurement uncertainty, or we may have omitted relevant features from our dataset [9]. Additionally, the choice of the underlying function space might be too restrictive to accurately capture the exact relationship between inputs and target values [8].
While the standard regression model addresses uncertainty in y_i, it typically assumes that the inputs $\mathbf{x}_i$ are deterministic quantities. Nonetheless, this assumption is often inappropriate in circumstances in which the inputs are subject to uncertainty, which might be caused by noisy measurements, for instance. One way to account for uncertainty in the inputs is the adoption of a probabilistic model. In the statistics literature, one usually distinguishes between making assumptions about the distribution of the true inputs $\mathbf{x}$ given fixed observations $\tilde{\mathbf{x}}$, referred to as Berkson-type errors-in-variables model, and the reverse approach, known as the classical errors-in-variables model [10, 11].
The Berkson-type model was originally introduced by Berkson [12] in the context of controlled observations $\tilde{\mathbf{x}}$. More precisely, $\tilde{\mathbf{x}}$ represents a desired value for a specified quantity in an experimental setup. However, this quantity is assumed to be measurable only with a noisy device, so the true inputs to the experiments will vary around $\mathbf{x}$. In

our scenario involving a fixed training dataset, we adopt the view of fixed observations $\tilde{\mathbf{x}}_i$ in the dataset around which the true latent inputs $\mathbf{x}_i$ vary. For the models discussed in chapter 3, we assume that $\mathbf{x}_i \sim \mathcal{N}(\tilde{\mathbf{x}}_i, \Sigma_{\mathbf{x}_i})$, meaning that $\mathbf{x}_i$ follows a multivariate normal distribution with mean vector $\tilde{\mathbf{x}}_i$ and covariance matrix $\Sigma_{\mathbf{x}_i}$.

2.2 Modeling Input Uncertainty with Bayesian Smoothing

In chapter 3, we will explore several GP models designed to handle noisy inputs. Most of these models require prior knowledge of the distribution of the true inputs, whereas others allow for the input distribution to be learned. However, the input distribution is rarely explicitly provided. Therefore, an appropriate model for the input distribution, specific to the context of the modeling task, must be chosen. For instance, Chau et al. [13] propose a method to account for uncertain data in K-means clustering, where they represent the uncertain position of moving objects with a uniform distribution over a line segment in the direction of movement. Another approach is adopted by Tsang et al. [14], who use a discrete distribution over samples to construct decision trees that handle uncertain inputs. In other fields, more complex noise models are often required. For example, in medical imaging, Gaussian, Poisson, or Rician noise models are commonly employed based on the specific imaging technique [15].
For our purposes, we consider the inputs provided as samples from a noisy signal, since we aim to extend the application of surrogate modeling the interaction between a rover wheel and soft soil by Ravi, Fediukov, et al. [6]. In their experiments, noisy signals are smoothed using a low-pass filter, which may discard relevant high-frequency information. It also corresponds to making potentially incorrect point estimates of the true inputs. However, estimating a distribution over the inputs reduces the chance of misrepresenting the true inputs and may therefore be more appropriate. To estimate these distributions, we apply a Bayesian smoother to the noisy signals, specifically the Rauch-Tung-Striebel smoother [16] (RTSS). For a comprehensive introduction to Bayesian filtering and smoothing, we refer to the textbook by Särkkä [17], which also serves as the basis for this short overview.
Bayesian smoothing builds upon Bayesian filtering, which adopts a Bayesian framework to estimate the true state of a time-varying system that is only indirectly observed through noisy measurements. While Bayesian filtering estimates the current state from a history of noisy observations, Bayesian smoothing also considers future measurements. In both cases, we are given a set of noisy measurements $\mathbf{z}_k$ at time points $k = 1, ..., T$ and we aim to compute the filtering distribution of the hidden states $\mathbf{x}_k$. A commonly used filter is the Kalman filter [18], which computes the posterior distribution of $\mathbf{x}_k$

given the observations $\mathbf{z}_1, ..., \mathbf{z}_k$ for linear Gaussian state space models

$$\begin{aligned} \mathbf{x}_k &= A_{k-1}\mathbf{x}_{k-1} + \boldsymbol{\varepsilon}_{p,k-1} \\ \mathbf{z}_k &= H_k\mathbf{x}_k + \boldsymbol{\varepsilon}_{n,k} \ . \end{aligned} \tag{2.1}$$

In this model, the time evolution of the hidden states is represented as a discrete linear dynamical system given by $\mathbf{x}_k = A_{k-1}\mathbf{x}_{k-1}$. The particular choice of A_{k-1} depends on the specific application and reflects our general assumptions about the dynamics of the system. However, since we do not have complete knowledge about the dynamics of the system, we allow for random perturbations by introducing normally distributed process noise $\boldsymbol{\varepsilon}_{p,k} \sim \mathcal{N}(\mathbf{0}, Q_{k-1})$. Furthermore, the matrix H_k defines the relationship between the hidden states and the measurements, as in some instances, direct measurements of the latent states are not availabe. Additionally, our measurements are assumed to be corrupted by Gaussian measurement noise $\boldsymbol{\varepsilon}_{n,k} \sim \mathcal{N}(\mathbf{0}, R_k)$. While we must make assumptions about the distribution of the noise, the assumption of normality is reasonable in many engineering applications. Noise often results from the superposition of many small, independent contributions, and by the central limit theorem, such a distribution approximates a Gaussian [19]. Moreover, we assume a prior distribution $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{m}_0, P_0)$ over the initial hidden state.
To estimate $\mathbf{x}_k$ for $k \geq 1$ given the measurements $\mathbf{z}_1, ..., \mathbf{z}_k$ and the posterior distribution $\mathbf{x}_{k-1}|\mathbf{z}_1, ..., \mathbf{z}_{k-1} \sim \mathcal{N}(\mathbf{m}_{k-1}, P_{k-1})$, we first predict $\mathbf{x}_k$ by applying the dynamic model in (2.1) on $\mathbf{x}_{k-1}|\mathbf{z}_1, ..., \mathbf{z}_{k-1}$. This prediction step yields the predictive distribution $\mathbf{x}_k|\mathbf{z}_1, ..., \mathbf{z}_{k-1} \sim \mathcal{N}(\mathbf{m}_k^-, P_k^-)$, where

$$\mathbf{m}_k^- = A_{k-1}\mathbf{m}_{k-1}, \quad P_k^- = A_{k-1}P_{k-1}A_{k-1}^\top + Q_{k-1} \ .$$

Next, we correct the predictive distribution by additionally conditioning on $\mathbf{z}_k$. The resulting posterior distribution is $\mathbf{x}_k|\mathbf{z}_1, ..., \mathbf{z}_k \sim \mathcal{N}(\mathbf{m}_k, P_k)$, where

$$\mathbf{m}_k = \mathbf{m}_k^- + K_k\mathbf{v}_k, \quad P_k = P_k^- - K_kS_kK_k^\top \ ,$$

with

$$\mathbf{v}_k = \mathbf{z}_k - H_k\mathbf{m}_k^-, \quad S_k = H_kP_k^-H_k^\top + R_k, \quad K_k = P_k^-H_k^\top S_k^{-1} \ .$$

Thus, starting with the prior distribution over $\mathbf{x}_0$, we can recursively determine the posterior distribution of $\mathbf{x}_k$.
While the Kalman filter computes the filtering distribution for each hidden state through forward recursion, the RTSS further refines these estimates through backward recursion. Specifically, it computes

$$\mathbf{m}_k^s = \mathbf{m}_k + G_k(\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^-), \quad P_k^s = P_k + G_k(P_{k+1}^s - P_{k+1}^-)G_k^\top$$

such that $\mathbf{x}_k|\mathbf{z}_1, ..., \mathbf{z}_T \sim \mathcal{N}(\mathbf{m}_k^s, P_k^s)$ starting from $k = T$. I.e., it conditions on all T measurements rather than only those up to time k. In the above equations, G_k is given by $P_k A_k^\top (P_{k+1}^-)^{-1}$, $\mathbf{m}_T^s$ equals $\mathbf{m}_T$, and P_T^s equals P_T. Furthermore, $\mathbf{m}_l, \mathbf{m}_l^-, P_l$ and P_l^- for all $l = 1, ..., T$ are obtained from a preceding application of the Kalman filter.
In many instances, general knowledge about the dynamics of the system is available and can be incorporated in the model through the choice of A_k. When such knowledge is not available, a more general model may be employed. For instance, in our experiments, we use a linear regression model with drift for one-dimensional signals. In this model, we assume that the signal is differentiable and approximate $x_k - x_{k-1}$ with $\Delta t \cdot \dot{x}_{k-1}$, where Δt is the constant time interval between two measurements. The residuals $(x_k - x_{k-1}) - \Delta t \cdot \dot{x}_{k-1}$ are treated as random and are modeled as normally distributed process noise. In addition, we model the derivatives as hidden states and we allow small changes in expectation at each time step by adding normally distributed process noise. Thus, given x_{k-1} and $\dot{x}_{k-1}$, our best estimate for x_k and $\dot{x}_k$ is $x_{k-1} + \Delta t \cdot \dot{x}_{k-1}$ and $\dot{x}_{k-1}$, respectively. In the form of a linear Gaussian state space model, the hidden states are $\mathbf{x}_k^h = [x_k, \dot{x}_k]^\top$ and we measure only x_k, therefore

$$A_k = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}, \quad H_k = \begin{bmatrix} 1 & 0 \end{bmatrix} .$$

In practice, the transition and measurement matrices in (2.1), along with the the covariance matrices of the process and measurement noise, are often parametrized. Specifically, in our real-world data experiments, we will parameterize the covariance matrices, as the process and measurement noise variance are unknown. The corresponding parameters $\boldsymbol{\theta}$ are usually estimated based on the posterior distribution $p(\boldsymbol{\theta}|\mathbf{z}_1, ..., \mathbf{z}_T)$. We do not explicit the specific estimation procedures in this thesis, but we refer interested readers to chapter 12 in the textbook by Särkkä [17] for a comprehensive discussion.

2.3 Gaussian Process Regression

In this thesis, we focus on GP regression, for which we now provide a short introduction based on the textbook by Rasmussen and Williams [8]. Here, we revisit the standard regression setting, where the inputs $\mathbf{x}_i$ are assumed to be deterministic. Parametric regression, as introduced in section 2.1, requires the modeler to select a suitable space of admissible functions. On the one hand, a too restrictive function space may not be able to appropriately model the input-output relationship in our training dataset, under which the predictive performance of the model may suffer. On the other hand, selecting an overly flexible function space increases the risk of overfitting, i.e., our model fits the

training data well, but performs poorly on new test data.
In GP regression, we follow an alternative, nonparametric approach. Loosely speaking, we assign probabilities to functions based on their ability to capture the functional relationship between inputs and targets in our dataset. We adopt a Bayesian perspective, where we initially specify a prior distribution over functions, for example, based on prespecified regularity assumptions, such as smoothness. Consequently, we use the training data to update this prior distribution, giving more weight to functions that align closely with the observed data. The updated distribution is the posterior distribution over functions. Typically, distributions over functions are represented by stochastic processes with index set $\mathcal{X}$. For our purposes, $\mathcal{X}$ is the space from which the training and test inputs originate and is often chosen to be $\mathbb{R}^D$ if the inputs are D-dimensional. Such a stochastic process with index set $\mathcal{X}$ can be defined as a collection of random variables $\{f(\mathbf{x})|\mathbf{x} \in \mathcal{X}\}$, which implies that realizations of these random variables define a function on $\mathcal{X}$. Particularly, a GP is a special case of a stochastic process $\{f(\mathbf{x})|\mathbf{x} \in \mathcal{X}\}$ which is characterized by the property that the joint distribution of any finite subset $\{f(\mathbf{x}^{(1)}), ..., f(\mathbf{x}^{(k)})|\mathbf{x}^{(1)}, ..., \mathbf{x}^{(k)} \in \mathcal{X}\}$ of random variables is multivariate normal. For notational convenience, we will often loosely denote GPs as $f(\mathbf{x})$ or simply as f. The distribution of a GP is completely determined by its mean function

$$m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]$$

and its covariance function

$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))],$$

which is also referred to as a kernel. Furthermore, we denote the distribution of a GP with given mean and covariance functions as $\mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$.
The specific choice of the prior mean and prior covariance function is a crucial modeling decision that defines the prior GP distribution over functions. In the absence of particular prior knowledge, the mean function is often chosen as the zero function. Moreover, the choice of the covariance function encodes our assumptions about the probable characteristics of the functions. A common choice for the covariance function is the squared exponential

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left(-\frac{|\mathbf{x} - \mathbf{x}'|^2}{\ell}\right), \tag{2.2}$$

where $|\cdot|$ denotes the Euclidean distance in $\mathbb{R}^D$, and σ_f^2 and ℓ are positive hyperparameters referred to as the signal variance and the lengthscale, respectively. Intuitively, the lengthscale can be interpreted as the distance one needs to move in the input space to expect a significant change in the function values. Furthermore, the signal variance

determines the variance of $f(\mathbf{x})$ for any fixed $\mathbf{x}$. It is noteworthy that this covariance function depends solely on the distance between two input points $\mathbf{x}$ and $\mathbf{x}'$, rather than on their specific locations. This property is also referred to as stationarity. Additionally, it favors smooth functions. In general, there are various other possible covariance functions in use. We note that not every function of two arguments $\mathbf{x}$ and $\mathbf{x}'$ qualifies as a covariance function, as some conditions need to be met. Nevertheless, given two admissible covariance functions k_1 and k_2, $k_1 + k_2$ as well as $k_1 \cdot k_2$ are also admissable covariance functions.
To simplify notation, we will gather the N training inputs $\mathbf{x}_i$, $i = 1, ..., N$, into a $D \times N$ matrix X, and the corresponding target values into a vector $\mathbf{y}$. Single test inputs will be denoted as $\mathbf{x}_*$, and similarly to the training inputs, sets of M test inputs will be aggregated into a $D \times M$ matrix X_*. Furthermore, we denote the random variable corresponding to the function evaluation at a training point $\mathbf{x}_i$ as f_i, and at a test point $\mathbf{x}_*$ as f_*. We then denote the random vector of function evaluations at our training inputs as $\mathbf{f}$ and at multiple test points as $\mathbf{f}_*$. Moreover, we denote the covariance matrix $\mathrm{cov}(\mathbf{f}, \mathbf{f}_*)$ as $K(X, X_*)$, and define $K(X, X)$ and $K(X_*, X_*)$ similarly. The specification of the GP prior distribution implies the prior distribution over the joint vectors of function values given by

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X, X) & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix}\right). \tag{2.3}$$

As discussed in section 2.1, we usually assume that the target value observations in the training dataset are corrupted by independent and identically distributed (i.i.d.) Gaussian noise, i.e.,

$$\mathbf{y} = \mathbf{f} + \boldsymbol{\varepsilon}, \quad \boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_n^2 I),$$

where I is the identity matrix of appropriate dimensionality. Since the additive noise is independent of the GP, it contributes a variance σ_n^2 to the diagonal of the covariance matrix $K(X, X)$. Thus, the prior distribution in (2.3) becomes

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X, X) + \sigma_n^2 I & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix}\right).$$

For GPs, conditioning the prior distribution over $\mathbf{f}_*$ on the observed data is analytically tractable and results in the predictive distribution

$$\mathbf{f}_* | X, \mathbf{y}, X_* \sim \mathcal{N}(\bar{\mathbf{f}}_*, \mathrm{cov}(\mathbf{f}_*, \mathbf{f}_* | X, \mathbf{y}, X_*)\ ,$$

where

$$\bar{\mathbf{f}}_* = \mathbb{E}[\mathbf{f}_* | X, \mathbf{y}, X_*] = K(X_*, X)(K(X, X) + \sigma_n^2 I)^{-1}\mathbf{y}\ , \tag{2.4}$$

$$\mathrm{cov}(\mathbf{f}_*, \mathbf{f}_* | X, \mathbf{y}, X_*) = K(X_*, X_*) - K(X_*, X)(K(X, X) + \sigma_n^2 I)^{-1}K(X, X_*). \tag{2.5}$$

This predictive distribution, which corresponds to the function evaluations at X_* of the posterior distribution over functions, allows us not only to predict at new test points, but also obtain error estimates for our prediction. It is important to note that, in the absence of noise variance, i.e., $\sigma_n^2 = 0$, (2.4) and (2.5) imply that samples from the posterior distribution will pass through the training data points.
From the preceding discussion, it is clear that a GP model does not involve explicit model parameters. However, specifying a GP model requires selecting both the covariance function and the model hyperparameters, which usually consist of parameters for the covariance function as well as the noise variance. This process is commonly referred to as the training of a GP. For instance, in a model that employs the squared exponential covariance function (2.2), the vector of hyperparameters is $\boldsymbol{\theta} = (\sigma_f^2, \ell, \sigma_n^2)$. The set of hyperparameters is generally dependend on the specific choice of the covariance function. For instance, a more general form of the squared exponential covariance function, which includes a larger set of hyperparameters, is given by

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{x}')^\top \operatorname{diag}(\boldsymbol{\ell})^{-2}(\mathbf{x} - \mathbf{x}')\right) + \sigma_n^2 \delta(\mathbf{x}, \mathbf{x}'), \tag{2.6}$$

where $\delta(\mathbf{x}, \mathbf{x}') = 1$ if and only if $\mathbf{x} = \mathbf{x}'$, and $\delta(\mathbf{x}, \mathbf{x}') = 0$ otherwise. This formulation absorbs the noise variance directly into the kernel, thereby also adjusting the variance of the predictive distribution (2.5) by the noise variance. Furthermore, $\operatorname{diag}(\boldsymbol{\ell})$ denotes the diagonal matrix constructed from the D-dimensional vector of lengthscales $\boldsymbol{\ell}$. By assigning a lengthscale hyperparameter to each input dimension, we allow for the potential deactivation of irrelevant input features through an appropriate hyperparameter optimizattestion procedure. Therefore, the covariance function is also said to implement automatic relevance determination (ARD).
We now address the optimization of hyperparameters. A widely used method maximizes the marginal likelihood

$$p(\mathbf{y}|X, \boldsymbol{\theta}) = \int p(\mathbf{y}|\mathbf{f}, X, \boldsymbol{\theta}) p(\mathbf{f}|X, \boldsymbol{\theta}) \, \mathrm{d}\mathbf{f}, \tag{2.7}$$

where p denotes the density of the respective random variable. Here, the term marginal refers to the process of integrating over the latent function values $\mathbf{f}$ with respect to their distribution, a procedure commonly known as marginalization over $\mathbf{f}$. Rather than maximizing the marginal likelihood directly, it is common practice to maximize the logarithm of the marginal likelihood. Whereas this approach leads to an equivalent optimization problem, taking the logarithm of probability densities often results in simpler expressions. One of the key advantages of GPs is that the integral in (2.7) is analytically tractable, allowing the log marginal likelihood to be exactly evaluated as

$$\log p(\mathbf{y}|X, \boldsymbol{\theta}) = -\frac{1}{2}\mathbf{y}^\top (K + \sigma_n^2 I)^{-1}\mathbf{y} - \frac{1}{2}\log|K + \sigma_n^2 I| - \frac{N}{2}\log 2\pi, \tag{2.8}$$

where we use $|\cdot|$ with matrices as argument to denote the determinant, and $K = K(X, X)$. When the noise variance is absorbed into the kernel, such as in (2.6), the term $K + \sigma_n^2 I$ must be replaced by K.
We can also incorporate prior information about the hyperparameters into the optimization procedure. As is common in Bayesian inference, we maximize the posterior distribution over the hyperparameters

$$p(\boldsymbol{\theta}|\mathbf{y}, X) = \frac{p(\mathbf{y}|X, \boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathbf{y}|X)} \ . \tag{2.9}$$

Here, $p(\boldsymbol{\theta})$ denotes the prior distribution over the hyperparameters, which reflects prior knowledge about them. For the maximization of (2.9) with respect to $\boldsymbol{\theta}$, the denominator can be ignored, as it is a constant independent of θ. A maximizer $\hat{\boldsymbol{\theta}}$ of the posterior (2.9) is also referred to as a maximum a posteriori (MAP) estimate. This estimate can equivalently be obtained by maximizing $\log(p(\mathbf{y}|X, \boldsymbol{\theta})p(\boldsymbol{\theta})) = \log p(\mathbf{y}|X, \boldsymbol{\theta}) + \log p(\boldsymbol{\theta})$, which comprises the log marginal likelihood and a penalty term $\log p(\boldsymbol{\theta})$. In the specific case of a flat prior $p(\boldsymbol{\theta}) \propto 1$, the hyperparameter posterior is proportional to the marginal likelihood (2.8), making the MAP estimate equivalent to the maximum marginal likelihood estimate [20].
The most expensive computation in GP regression is the inversion of the covariance matrix $K + \sigma_n^2$, which is necessary for both evaluating the marginal likelihood and calculating the predictive distribution. This inversion can be more efficiently performed using the Cholesky decomposition, as the covariance matrix is positive semidefinite. Nonetheless, the runtime complexity remains $\mathcal{O}(N^3)$. However, once the Cholesky decomposition is computed, the gradients of (2.8) can be obtained with a computational complexity $\mathcal{O}(N^2)$. Therefore, gradient descent-based optimization procedures are well-suited to obtain the maximum marginal likelihood or MAP estimate of the hyperparameters.
After training our model, it is essential to evaluate its performance on a set of test points. To this end, we will consider two metrics. The first metric is the standardized mean squared error (SMSE). The mean squared error (MSE) is defined as

$$MSE = \frac{1}{M} \sum_{i=1}^{M} (y_{i,*} - \overline{f}(\mathbf{x}_{i,*}))^2 \ .$$

That is, the MSE is the average squared error between the test target values $y_{i,*}$ and the predictive means at the test points $\mathbf{x}_{i,*}$. Since the MSE depends on the scale of the test target values, it is typically normalized by the sample variance of the test target values, which yields the SMSE. The SMSE solely considers predictive means at the test inputs. However, our model yields the whole predictive distribution, which is incorporated in

the second metric: the mean standardized log loss (MSLL). To compute the MSLL, we first evaluate the negative log predictive density

$$-\log p(y_{i,*}|\mathcal{D}, \mathbf{x}_{i,*}) = \frac{1}{2}\log(2\pi\sigma_{i,*}^2) + \frac{(y_{*,i} - \overline{f}(\mathbf{x}_{i,*}))}{2\sigma_{i,*}^2}$$

for all test data points $(\mathbf{x}_{i,*}, y_{i,*})$, $i = 1, ..., M$. Here, $\sigma_{i,*}^2 = \mathbb{V}(f_{i,*}) + \sigma_n^2$ with $\mathbb{V}(f_{i,*}) = \text{cov}(f_{i,*}, f_{i,*})$ as defined in (2.5). The noise variance is included, since the test targets are assumed to be noisy, similar to the training targets. These values are then standardized by subtracting $-\log \mathcal{N}(y_{i,*}|\overline{\mathbf{y}}, s^2)$, which is the negative log density of a normal distribution with the sample mean $\overline{\mathbf{y}}$ and sample variance s^2 of the training targets $\mathbf{y}$ evaluated at the training target $y_{i,*}$. Finally, the MSLL is obtained by averaging these standardized values. It is worth noting that we use the negative logarithm rather than the logarithm, so that the MSLL has the characteristics of a loss function, where lower values indicate better performance.

2.4 Multi-Fidelity Surrogate Modeling with Gaussian Processes

In this section, we introduce some multi-fidelity GP models along with the necessary preliminaries to understand them. The primary focus will be on the nonlinear autoregressive GP model [21] (NARGP) and the deep GP model for multi-fidelity modeling [22] (MFDGP), as we will extend these models to handle uncertain inputs in section 3.2 of this thesis.

2.4.1 Multi-Fidelity Surrogate Modeling

Many applications in engineering and science require high-accuracy predictions of specific quantities. These predictions may be obtained by experiments or by complex simulations, both of which can be costly and time-consuming. In such cases, surrogate models are often employed as a replacement. These are data-driven mathematical models that are built upon available expensive simulation or experimental results and commonly significantly accelerate the prediction of the quantitiy of interest. However, obtaining a sufficiently large high-accuracy dataset for an accurate surrogate model can be costly as well. One potential solution to this issue is to employ a multi-fidelity surrogate model. These models leverage not only data from high-fidelity sources, but also larger quantities of low-fidelity data. Low-fidelity data may be obtained from less expensive models that yield less accurate approximations to the high-accuracy data.
Assigning fidelity levels to datasets necessitates careful consideration. On the one hand, fidelity levels are inherently defined relative to each other, with the high-fidelity data

typically characterized by a desired level of accuracy. On the other hand, determining the fidelity of a particular data source can be ambiguous. For instance, one might have data from a one-dimensional numerical simulation with a fine grid and from a three-dimensional simulation with a coarser grid. Although finer grids are generally associated with higher accuracy, the dimensionality of the model may significantly influence the model's accuracy as well. In such a situation, the fidelity levels must be determined depending on the context.
After determining the fidelity levels of the available datasets, a multi-fidelity surrogate model is constructed. These models fuse data of different fidelity levels or combine surrogate-models for each fidelity level into a single surrogate model. Often, a hierarchy of multiple fidelity levels is considered, rather than just two. Ultimately, the objective of multi-fidelity surrogate modeling is to achieve high-accuracy results by enhancing low-fidelity models with data from high-fidelity sources [23].

2.4.2 The Linear Autoregressive Model

To illustrate multi-fidelity surrogate modeling with GPs and to provide the foundation for the model in subsection 2.4.3, we begin by introducing the linear autoregressive multi-fidelity GP regression model developed by Kennedy and O'Hagan [24]. In the multi-fidelity context, we consider a set of datasets $\{\mathcal{D}_s\}_{s=1}^{t}$, where the index s indicates the fidelity level of each respective dataset. We define higher indices to correspond to higher fidelity levels, with the highest fidelity level given by t. Furthermore, we denote the datasets as $\mathcal{D}_s = \{(\mathbf{x}_i^{(s)}, y_i^{(s)}) | i = 1, ..., N_s\}$ and use the notation introduced in section 2.3 for all other expressions. Where needed, we introduce additional indices to specify the corresponding fidelity level.
Kennedy and O'Hagan model the surrogate model for each fidelity s as a GP $f^{(s)}(\mathbf{x})$. They assume noise-free targets, which implies $\mathbf{y}^{(s)} = \mathbf{f}^{(s)}$. This is based on the assumption that the data is generated by simulations without any measurement error. For each fidelity $s \geq 2$, the corresponding surrogate model is linearly related a priori to $f^{(s-1)}$ through

$$f^{(s)}(\mathbf{x}) = \rho_{s-1} f^{(s-1)}(\mathbf{x}) + \delta^{(s)}(\mathbf{x}) \; . \tag{2.10}$$

At the lowest fidelity level, $f^{(1)}(\mathbf{x})$ is modeled as a GP with prior mean function $\mathbf{g}_1(\mathbf{x})^\top \boldsymbol{\beta}_1$ and squared exponential covariance function $k_1(\mathbf{x}, \mathbf{x}')$. Furthermore, $\delta^{(s)}$ is a GP independent of $f^{(1)}(\mathbf{x}), ..., f^{(s-1)}(\mathbf{x})$ with mean function $\mathbf{g}_s(\mathbf{x})^\top \boldsymbol{\beta}_s$ and covariance function $k_s(\mathbf{x}, \mathbf{x}')$, which is the squared exponential as well. All occuring prior mean functions are modeled using linear regressions, with $\mathbf{g}_s(\mathbf{x})$ as the vector of regression functions and $\boldsymbol{\beta}_s$ as parameter vector. These parameters are estimated alongside all kernel hyperparameters and the coefficients ρ_s.

This model is motivated by the theoretical insights provided by O'Hagan [25]. The author proves that a decomposition of the form (2.10), where ρ_{s-1} is a function of $\mathbf{x}$, exists if a kind of Markov property is satisfied. This property states that, once $f^{(s-1)}(\mathbf{x})$ is observed, no additional information about $f^{(s)}(\mathbf{x})$ can be gained by observing any other $f^{(s-1)}(\mathbf{x}')$ with $\mathbf{x} \neq \mathbf{x}'$. While the model (2.10) assumes ρ_{s-1} to be a constant, more general representations for ρ_{s-1} are also possible, such as a parameterizing ρ_{s-1} using a linear regression model, as applied in Le Gratiet and Garnier [26].
Computing the predictive distribution at fidelity s for a new test input $\mathbf{x}_*$ requires inverting the covariance matrix of $[\mathbf{f}^{(1)}, ..., \mathbf{f}^{(s)}]^\top$, which has a computational complexity of $\mathcal{O}((\sum_{i=1}^{s} N_i)^3)$. To accelerate this computation step, Le Gratiet and Garnier replace the prior GP $f_{(s-1)}$ in (2.10) with its posterior $f_{(s-1)}|\mathcal{D}_{s-1}$, thereby decoupling the surrogate models in the prior distribution. They show that this formulation results in the same posterior distribution at each fidelity level as the original model by Kennedy and O'Hagan. Thus, building an autoregressive GP model with t fidelities is equivalent to building t independent GP models. The consequence of this approach is that it reduces the computational complexity of determining the posterior distribution at fidelity level s to $\mathcal{O}(\sum_{i=1}^{s} N_i^3)$, as it only requires the inversion of covariances matrices of size $N_i \times N_i$ for $i = 1, ..., s$.
Furthermore, both models assume that the sets of inputs are nested with decreasing fidelity, such that

$$\{\mathbf{x}_i^{(s)} | i = 1, ..., N_s\} \subseteq \{\mathbf{x}_i^{(s-1)} | i = 1, ..., N_{s-1}\} \tag{2.11}$$

for all $s = 2, ..., t$. This assumption is reasonable, for instance, in scenarios where low-fidelity data is cheap to obtain and the low-fidelity model can be evaluated on the inputs of higher-fidelity levels with negligible additional cost. Since the data is assumed to be noise-free, the nestedness of the input sets implies that, for all $i = 1, ..., N_s$, $f^{(s-1)}(\mathbf{x}_i^{(s)})$ in (2.10) is a deterministic quantity, independent of the hyperparameters of the GP $f^{(s-1)}(\mathbf{x})$ and all surrogate models with lower fidelities than $s - 1$. As a result, the hyperparameters for the surrogate models at each fidelity level can be estimated separately.

2.4.3 The Nonlinear Autoregressive Model

The linear autoregressive models discussed in the preceding section are designed to capture a linear relationship between consecutive fidelity levels. If no such relationship can be detected during training, the autoregressive model often ignores the lower-fidelity data. Instead, it may rely on δ_s in (2.10) to fit a standard GP regression model to the higher-fidelity data. While this behaviour is often desirable, it disregards more complex, non-linear relations between datasets of successive fidelities, which might be

exploited to build a more accurate surrogate model for the high-fidelity data.
This potential limitation was noted by Perdikaris et al. [21], who address it through the development of the NARGP. Their approach modifies the linear relation in (2.10) to a nonlinear one given by

$$f^{(s)}(\mathbf{x}) = g_s(\mathbf{x}, f^{(s-1)}(\mathbf{x})) \,, \tag{2.12}$$

where $g_s(\mathbf{x}, y) \sim \mathcal{GP}(\mathbf{0}, k_s((\mathbf{x}, y), (\mathbf{x}', y')))$. However, placing a GP prior over $f^{(1)}(\mathbf{x})$ and another GP prior over $g_s(\mathbf{x}, y)$ for all $s = 2, ..., t$ results in a non-Gaussian $f^{(s)}(\mathbf{x})$ if $s \geq 2$. As a consequence, training and inference become intractable. To overcome this challenge, the authors replace the GP prior $f^{(s-1)}(\mathbf{x})$ in (2.12) with the GP posterior $f^{(s-1)}(\mathbf{x})|\mathcal{D}_{s-1}$, motivated by the approach of Le Gratiet and Garnier [26]. Given the assumption of nestedness as in (2.11) and noise-free data, the posterior distribution of $\mathbf{f}^{(s)}$ becomes analytically tractable for all $s = 1, ..., t$, as it depends on the deterministic data from lower fidelity levels and not on the surrogate models from lower fidelities. Consequently, the surrogate model for each fidelity can be trained independently.
In contrast to training, computing the predictive distribution for unseen test inputs remains computationally intractable. To address this, Perdikaris et al. estimate the predictive means and variances of a surrogate model with a specific fidelity by Monte Carlo integration. For this purpose, samples from the predictive distribution of the desired fidelity level are required. These can be obtained by initially drawing from the Gaussian posterior at fidelity level 1 and subsequently propagating the samples through the surrogate models until the desired fidelity level is reached. This involves iteratively drawing new samples from the Gaussian predictive distributions at each fidelity level.
Furthermore, all targets y in the dataset are assumed to be the results of experiments or simulations conducted at $\mathbf{x}$. It is therefore reasonable to treat $\mathbf{x}$ and y as originating from inherently different spaces. Consequently, the authors suggest using a more structured covariance function $k_s((\mathbf{x}, y), (\mathbf{x}', y'))$ in the GP prior $g_s(\mathbf{x}, y)$ for $s \geq 2$ that separates $\mathbf{x}$ and y. This covariance function is specified as

$$k_s((\mathbf{x}, y), (\mathbf{x}', y')) = k_\rho^{(s)}(\mathbf{x}, \mathbf{x}') \cdot k_f^{(s)}(y, y') + k_\delta^{(s)}(\mathbf{x}, \mathbf{x}') \,, \tag{2.13}$$

which reflects the autoregressive model (2.10). Each of the covariance functions $k_\rho^{(s)}$, $k_f^{(s)}$, and $k_\delta^{(s)}$ is defined as the ARD squared exponential as formulated in (2.6), excluding noise variance and including its own set of hyperparameters.
Moreover, the deep representation in (2.12) effectively projects the GP posterior $f^{(s-1)}(\mathbf{x})$ onto a $(D+1)$-dimensional smooth latent manifold within a $(D+2)$-dimensional space. From this latent manifold, the higher-fidelity surrogate $f^{(s)}(\mathbf{x})$ can be recovered. This illustrates the potential benefits of multi-fidelity modeling compared to directly

fitting the high-fidelity data. When the latent manifold assumes a relatively simple structure, a small amount of high-fidelity data may suffice to accurately capture it. More complicated features may be captured in the low-fidelity surrogate, for which larger datasets are typically be available [6, 21].

2.4.4 Multi-Fidelity Modeling with Deep Gaussian Processes

The NARGP discussed in the previous subsection addresses the intractability of the prior distribution defined by (2.12) by replacing the prior GP $f^{(s-1)}(\mathbf{x})$ with the posterior GP $f^{(s-1)}(\mathbf{x})|\mathcal{D}_{s-1}$. As an alternative, a so-called variational approximation can be applied, which results in a deep GP model [27] (DGP). In this section, we present this approach. Before doing so, we introduce some preliminaries that will be useful for other parts of this thesis as well.

Variational Inference

Variational inference is a method in Bayesian statistics for the approximation of posterior distributions when no analytical expressions are available. For a comprehensive yet concise introduction, we refer to Ganguly and Earp [28], which also serves as the basis for this brief overview. Consider an observable random variable X that depends on a latent random variable Z. In many problems in Bayesian statistics, the objective is to compute the posterior probability density

$$p(z|x) = \frac{p(x|z)p(z)}{p(x)} \ .$$

When an analytical expression for the posterior density is unavailable, variational inference approximates the posterior density $p(z|x)$ with a variational posterior $q(z)$ which originates from a predefined family of densities $\mathcal{Q}$.
To assess the quality of the approximation, a metric that measures the distance between probability densities is required. For this purpose, the Kullback-Leibler (KL) divergence is typically used. Given two arbitrary continuous probability densities p and q, the KL divergence between them is defined as

$$\mathrm{KL}[p(y)||q(y)] = \int p(y) \log \frac{p(y)}{q(y)} \mathrm{d}y \ = \mathbb{E}_{p(y)} \left[\log \frac{p(y)}{q(y)} \right] \ ,$$

where the subscript in $\mathbb{E}_{p(y)}$ denotes that the expectation is taken with respect to p. The KL divergence is non-negative and equals zero if and only if the probability densities p and q represent the same distribution. It is also non-symmetric, which means that, in general, $\mathrm{KL}[p(y)||p(y)] \neq \mathrm{KL}[q(y)||p(y)]$. As a consequence, minimizing the KL

divergence with respect to q typically yields different results depending on whether $p(y)$ or $q(y)$ is used as first argument. Specifically, a minimizer $q^*(y)$ of $\mathrm{KL}[q(y)||p(y)]$ usually underestimates $p(y)$, which means that $q(y)$ will be close to zero in regions where $p(y)$ is close to zero.
To approximate $p(z|x)$, we aim to minimize $\mathrm{KL}[q(z)||p(z|x)]$ with respect to the probability density $q(z)$ subject to $q(z) \in \mathcal{Q}$. In general, this KL divergence can not be directly evaluated, since the posterior density is unknown. However, reformulating the KL divergence yields

$$\begin{aligned}\mathrm{KL}[q(z)||p(z|x)] &= -\mathrm{E}_{q(z)}[\log p(x|z)] + \mathrm{KL}[q(z)||p(z)] + \mathbb{E}_{q(z)}[\log p(x)] \\ &= -\mathcal{L}(q(z)) + \log p(x) \ ,\end{aligned}$$

where $\mathcal{L}(q(z)) = \mathbb{E}_{q(z)}[\log p(x|z)] - \mathrm{KL}[q(z)||p(z)]$ is known as the evidence lower bound (ELBO), and $\mathbb{E}_{q(z)}[\log p(x)] = \log p(x)$, since $\log p(x)$ does not depend on the integration variable z. Consequently, it can be ignored for the purpose of optimization, and the ELBO, which is often analytically tractable, can be maximized instead.
Furthermore, the ELBO bounds the log marginal $\log p(x)$ from below, which is also referred to as log evidence. This can be derived by

$$\begin{aligned}\log p(x) &= \log \int p(x,z)\frac{q(z)}{q(z)}\mathrm{d}z = \log \mathbb{E}_{q(z)}\left[\frac{p(x,z)}{q(z)}\right] \geq \mathbb{E}_{q(z)}\left[\log \frac{p(x,z)}{q(z)}\right] \\ &= \mathbb{E}_{q(z)}[\log p(x|z)] - \mathbb{E}_{q(z)}\left[\log \frac{q(z)}{p(z)}\right] = \mathcal{L}(q(z)) \ ,\end{aligned} \tag{2.14}$$

where the inequality applied is also known as Jensen's inequality. Notably, the ELBO is a tight lower bound to the log marginal, and equality in (2.14) holds if and only if $q(z)$ represents the same distribution as the posterior denisty $p(z|x)$ [29]. Thus, variational inference can be interpreted in two ways: as a method for approximating the posterior density $p(z|x)$, or as a method to bound the log evidence $\log p(x)$ from below.

Sparse Approximations of Gaussian Processes Using Inducing Variables

As outlined in section 2.3, calculating the predictive distribution or the marginal likelihood of a GP requires inverting the covariance matrix $K(X,X)$, which has a computational complexity of $\mathcal{O}(N^3)$. For large datasets, this computation step may become infeasible. To overcome this limitation, several approximate methods have been proposed. For a review of early approaches, see [30, 8]. One possible strategy is to utilize inducing variables $\mathbf{u}$, which represent the function values of a GP at a set of $M \ll N$ pseudo-inputs, denoted in matrix form as $Z \in \mathbb{R}^{D\times M}$.
For instance, Snelson and Ghahramani [31] introduced an inducing variable framework

where the pseudo-inputs are treated as freely optimizable hyperparameters. If both the pseudo-inputs and the inducing variables are initially considered fixed, the likelihood of the data can be computed using the predictive distribution $p(\mathbf{y}|X, \mathbf{u}, Z)$ as shown in (2.4) and (2.5). Here, Z and $\mathbf{u}$ serve as a noise-free version of X and $\mathbf{y}$, respectively. A straightforward training procedure would maximize this likelihood with respect to Z, $\mathbf{u}$, and the model hyperparameters. However, Snelson and Ghahramani propose to treat $\mathbf{u}$ as latent variables. When placing a GP prior over $\mathbf{u}$, the posterior distribution $p(\mathbf{u}|\mathcal{D}, Z)$ becomes tractable. Consequently, $\mathbf{u}$ can be marginalized with respect to this posterior to obtain the predictive distribution $p(y_*|\mathbf{x}_*, Z, \mathcal{D})$. Moreover, the model can be trained by maximizing the analytically tractable marginal likelihood $p(\mathbf{y}|X, Z)$ with respect to both the model hyperparameters $\boldsymbol{\theta}$ and the pseudo-inputs Z. Using this inducing variable framework significantly reduces the computational cost of calculating the marginal likelihood to $\mathcal{O}(NM^2)$ and the predictive distribution for a single test input to $\mathcal{O}(M^2)$.

In effect, this approach replaces the covariance function of the original GP model with one that incorporates the pseudo-inputs as hyperparameters. Titsias [32] notes the potential risk of overfitting with this modification and addresses it using a variational approach. Specifically, the author approximates the predictive distribution $p(\mathbf{f}_*|X_*, \mathcal{D})$ for arbitrary X_*, which determines the posterior GP, with a variational distribution. To specify the form of the variational posterior, Titsias reexpresses the true predictive distribution by introducing inducing variables $\mathbf{u}$ and marginalizing over both $\mathbf{f}$ and $\mathbf{u}$, which involves the conditional densities $p(\mathbf{f}_*|\mathbf{u}, \mathbf{f})$ and $p(\mathbf{u}|\mathbf{y})$. The form of the variational posterior $q(\mathbf{f}_*)$ is then specified by replacing $p(\mathbf{f}_*|\mathbf{u}, \mathbf{f})$ with $p(\mathbf{f}_*|\mathbf{u})$ and $p(\mathbf{u}|\mathbf{y})$ with an arbitrary Gaussian distribution $\phi(\mathbf{u})$. Thus, $q(\mathbf{f}_*)$ is determined by the choice of ϕ and Z. To approximate the true posterior, $\mathrm{KL}[q(\mathbf{f}, \mathbf{u})||p(\mathbf{f}, \mathbf{u}|\mathcal{D})]$ is minimized with respect to $q(\mathbf{f}, \mathbf{u})$ while simultaneously optimizing the model hyperparameters. This is equivalent to maximizing the analytically tractable ELBO, which is evaluated in $\mathcal{O}(NM^2)$. When Z is fixed, the form of the variational posterior allows an analytical expression for an optimal Gaussian distribution $\phi(\mathbf{u})$. This reduces the optimization variables to only Z and the model hyperparameters. It is important to note that the pseudo-inputs Z are not model hyperparameters, but variational parameters; The original GP model is not changed. A notable benefit of this optimization procedure is that the ELBO includes an additional regularization term, which penalizes large deviations from the original GP model.

A notable application of this approach is the Bayesian GP latent variable model (GPLVM) proposed by Titsias and Lawrence [33], which we will discuss in more detail in section 3.1. Their work adopts the described variational procedure to a setting where the inputs X are latent, addressing the intractability of the log marginal likelihood by deriving an analytically tractable ELBO.

Deep Gaussian Processes

Having established the necessary preliminaries, we will now turn our attention to DGPs, which were introduced by Damianou and Lawrence [27] for conventional machine learning tasks. The development of this model was motivated by the empirical structural benefits of deep architectures for handling complex porblems [34]. A DGP consists of a hierarchy of $L+1$ layers, with the final L layers being vector-valued stochastic processes $\mathbf{y}^{(l)}$ of dimension D_l, where $l = 1, ..., L$. Each dimension $d = 1, ..., D_l$ of a process $\mathbf{y}^{(l)}$ is assigned a prior distribution through the functional relationship

$$y_d^{(l)} = f_d^{(l)}(\mathbf{y}^{(l-1)}) + \varepsilon_d^{(l)} \ . \tag{2.15}$$

Here, the processes $f_d^{(l)}(\mathbf{y})$ follow GP prior distributions $\mathcal{GP}(\mathbf{0}, k^{(l)}(\mathbf{y}, \mathbf{y}'))$, which are assumed to be independent across the dimensions $d = 1, ..., D_l$. Additionally, $\varepsilon_d^{(l)}$ is assumed to be i.i.d. Gaussian noise. The first layer $\mathbf{y}^{(0)}$ represents the inputs, which we assume to be deterministic, although the framework also permits the use of latent inputs. Unlike the frameworks discussed so far, DGPs generally consider vector-valued stochastic processes as well as training targets. Consequently, we represent the training targets as a matrix $Y \in \mathbb{R}^{D_L \times N}$ and denote the vectors of latent function evaluations, when the training input X are used in the input layer, as a matrix $F^{(l)}$.
In this model, computing posterior distributions and the marginal likelihood of the training targets is generally intractable. Therefore, a variational approach related to the Bayesian GP latent variable model is employed. Specifically, inducing variables $U^{(l)} \in \mathbb{R}^{D_l \times M}$ associated with pseudo-inputs $Z^{(l-1)}$ are introduced at each layer l. The joint posterior over all $F^{(l)}$ and $U^{(l)}$ for $l = 1, ...L$ as well as over all $\mathbf{y}^{(l)}$ for $l = 1, ..., L-1$ is approximated by a variational posterior. This variational posterior is assumed to factorize across layers. That is, the input $\mathbf{y}^{(l)}$ to layer $l+1$ is assumed to be independent of the output $f^{(l)}(\mathbf{y}^{(l-1)})$ of layer l conditioned on $\mathbf{y}^{(l-1)}$ and $U^{(l)}$. Additionally, the variational posterior approximates the posterior distribution of all $\mathbf{y}^{(l)}$ for $l = 1, ..., L-1$ by a Gaussian.
Although the specified form of the variational distribution makes the ELBO tractable, it discards the correlations between layers and may therefore fail to capture the full complexity of the model. To address this limitation, Salimbeni and Deisenroth [35] propose a variational distribution that preserves the correlations between layers. In addition, their variational distribution does not impose Gaussianity between layers. However, the ELBO can not be evaluated exactly anymore, and is instead approximated by Monte Carlo integration.
In the following, we outline their approach, which serves as a basis for the MFDGP.

Their model is trained by maximizing the ELBO

$$\mathcal{L}(q(\{F^{(l)}, U^{(l)}\}_{l=1}^{L})) = \mathbb{E}_{q(\{F^{(l)}, U^{(l)}\}_{l=1}^{L})}\left[\log \frac{p(Y, \{F^{(l)}, U^{(l)}\}_{l=1}^{L})}{q(\{F^{(l)}, U^{(l)}\}_{l=1}^{L})}\right] , \tag{2.16}$$

Here, the joint density $p(Y, \{F_l, U_l\}_{l=1}^{L})$ can be factorized as

$$p(Y, \{F_l, U_l\}_{l=1}^{L}) = \prod_{i=1}^{N} p\left(\mathbf{y}_i|\mathbf{f}_i^{(L)}\right) \prod_{l=1}^{L} p\left(F^{(l)}|U^{(l)}, F^{(l-1)}, Z^{(l-1)}\right) p\left(U^{(l)}|Z^{(l-1)}\right) ,$$

where $F^{(0)} = X$. Unlike the approach by Damianou and Lawrence [27], the noise between layers does not need to be separately mentioned and is consequently absorbed into the kernels. Thus, $f_d^{(l)}$ assumes the role of $y_d^{(l)}$ in (2.15) for all layers except the output layer. The posterior distribution $p(\{F_l, U_l\}_{l=1}^{L}|Y)$ is approximated by a variational posterior of the form

$$q(\{F^{(l)}, U^{(l)}\}_{l=1}^{L}) = \prod_{l=1}^{N} p(F^{(l)}|U^{(l)}, F^{(l-1)}, Z^{(l-1)}) q(U^{(l)}) , \tag{2.17}$$

where $q(U^{(l)}) = \prod_{d=1}^{D_l} q(\mathbf{u}_d^{(l)}) = \prod_{d=1}^{D_l} \mathcal{N}(\mathbf{u}_d^{(l)}|\mathbf{m}_d^{(l)}, S_d^{(l)})$ is factorized across dimensions within each layer. Here, $\mathcal{N}(\mathbf{u}_d^{(l)}|\mathbf{m}_d^{(l)}, S_d^{(l)})$ denotes the density of a multivariate Gaussian, with $\mathbf{m}_d^{(l)}$ and $S_d^{(l)}$ acting as variational parameters. To simplify the notation, we collect these parameters into sets $M^{(l)}$ and $S^{(l)}$.
Through reformulations and marginalization, the ELBO (2.16) can be expressed as

$$\mathcal{L}(q(\{F^{(l)}, U^{(l)}\}_{l=1}^{L})) = \sum_{i=1}^{N} \mathbb{E}_{q(\mathbf{f}_i^{(l)})}\left[\log p(\mathbf{y}_i|\mathbf{f}_i^{(l)})\right] + \sum_{l=1}^{L} \mathrm{KL}[q(U^{(l)})||p(U^{(l)}|Z^{(l-1)})] . \tag{2.18}$$

The KL divergence between the prior $p(U^{(l)})$ and the variational posterior $q(U^{(l)})$ is analytically tractable, since both terms are Gaussian. However, the expectations $\mathbb{E}_{q(\mathbf{f}_i^{(l)})}\left[\log p(\mathbf{y}_i|\mathbf{f}_i^{(l)})\right]$ cannot be evaluated exactly and are therefore approximated using Monte Carlo integration. This requires drawing samples from $q(\mathbf{f}_i^{(l)})$, for which all variables in (2.17) must be marginalized out except for $\mathbf{f}_i^{(l)}$. The marginalization over $\{U^{(l)}\}_{l=1}^{L}$ is analytically tractable and results in

$$q(\{F^{(l)}\}_{l=1}^{L}) = \prod_{l=1}^{L} q(F^{(l)}|M^{(l)}, S^{(l)}, F^{(l-1)}, Z^{(l-1)}) = \prod_{l=1}^{L} \mathcal{N}(F^{(l)}|\tilde{M}^{(l)}, \tilde{S}^{(l)}) ,$$

where the means $\tilde{M}^{(l)}$ and covariance matrices $\tilde{S}^{(l)}$ are dependent on $M^{(l)}$, $S^{(l)}$, $Z^{(l-1)}$, and $F^{(l-1)}$. It can be further shown that marginalizing out all latent variables $\{\mathbf{f}_j^{(l)}\}_{j\neq i}$

from $q(F^{(l)}M^{(l)}, S^{(l)}, F^{(l-1)}, Z^{(l-1)})$ for a given $i \leq N$ results in a Gaussian distribution $q(\mathbf{f}_i^{(l)}|M^{(l)}, S^{(l)}, \mathbf{f}_i^{(l-1)}, Z^{(l-1)})$, which depends on $F^{(l-1)}$ solely through $\mathbf{f}_i^{(l-1)}$. By iteratively applying this result, the marginal distribution $q(\mathbf{f}_i^{(L)})$ simplifies to

$$q(\mathbf{f}_i^{(L)}) = \int ... \int \prod_{l=1}^{L} q(\mathbf{f}_i^{(l)}|M^{(l)}, S^{(l)}, \mathbf{f}_i^{(l-1)}, Z^{(l-1)}) \mathrm{d}\mathbf{f}_i^{(1)} ... \mathrm{d}\mathbf{f}_i^{(l-1)} \ ,$$

which facilitates sampling from $q(\mathbf{f}_i^{(L)})$. Initially, samples from the Gaussian distribution $q(\mathbf{f}_i^{(1)}|M^{(l)}, S^{(l)}, \mathbf{x}_i, Z^{(l-1)})$ are drawn, resulting in a set of samples $\hat{\mathbf{f}}_i^{(1)}$. Subsequently, these samples are used to obtain samples $\hat{\mathbf{f}}_i^{(2)}$ from $q(\mathbf{f}_i^{(2)}|M^{(l)}, S^{(l)}, \hat{\mathbf{f}}_i^{(1)}, Z^{(l-1)})$. This procedure is iterated until a final set of samples $\hat{\mathbf{f}}_i^{(L)}$ is obtained, which are unbiased samples from $q(\mathbf{f}_i^{(L)})$. Using these samples, the intractable mean in the ELBO (2.18) can be approximated through Monte Carlo integration. This sampling scheme can be also applied to sample from the predictive distribution $q(\mathbf{f}_*^{(L)})$.

Deep Gaussian Processes for Multi-Fidelity Surrogate Modeling

We return to the context of multi-fidelity modeling, where we consider t datasets $\{(X^{(1)}, \mathbf{y}^{(1)}), ..., (X^{(t)}, \mathbf{y}^{(t)})\}$ with scalar training targets. The DGP method proposed by Salimbeni and Deisenroth [35] was extended to the MFDGP by Cutajar et al. [22] to suit this application. Unlike the NARGP, their approach does not require the training inputs to be nested. The core idea is to model the surrogate for the s-th fidelity as the s-th layer of a DGP. This achieved by modifying the prior in (2.15) to

$$y^{(s)} = f^{(s)}(\mathbf{x}, f^{(s-1)}(\mathbf{x}, ...)) + \varepsilon^{(s)}$$

for $s \geq 1$. Here, $f^{(s)}(\mathbf{x}, y) \sim \mathcal{GP}(\mathbf{0}, k^{(s)}((\mathbf{x}, y), (\mathbf{x}', y')))$ and $\varepsilon^{(s)}$ represents i.i.d. Gaussian noise. As before, the surrogate for the lowest fidelity level is assumed to follow a standard GP prior. Unlike standard DGPs, it is assumed that there is no noise between layers, but the observations $y^{(s)}$ are allowed to be corrupted by noise. Though, as in standard DGPs, inducing variables are introduced and the marginal likelihood is approximated by a lower bound, which is then maximized with respect to both model hyperparameters and variational parameters. Function evaluations at fidelity s for the inputs $X^{(s)}$ require evaluations of all lower-fidelity surrogates at these inputs. For this purpose, we denote the function evaluations of the surrogate for fidelity s with inputs from fidelity l recursively as $\mathbf{f}_l^{(s)} = \mathbf{f}^{(s)}(X^{(l)}, \mathbf{f}_l^{(s-1)})$. Here, the notation $\mathbf{f}(X)$ is used to represent the vector of function evaluations across all inputs aggregated in X. Additionally, $f_{l,i}^{(s)}$ denotes the function evaluation of the i-th input in X_l.

With this notation, the expanded marginal likelihood is expressed as

$$p(\{\mathbf{y}^{(s)}, \{\mathbf{f}_s^{(l)}\}_{l=1}^{s}, \mathbf{u}^{(s)}\}_{s=1}^{t}) = \left(\prod_{s=1}^{t}\prod_{i=1}^{N_s} p(y_i^{(s)}|f_{s,i}^{(s)}) \prod_{l=1}^{s} p(\mathbf{f}_s^{(l)}|\mathbf{u}^{(l)}, \{X_s, \mathbf{f}_s^{(l-1)}, Z_{l-1}\})\right) \cdot \prod_{l=1}^{t} p(\mathbf{u}^{(l)}|Z_{l-1}) \,. \tag{2.19}$$

The variational posterior maintains the model structure, but introduces a free Gaussian distribution over the inducing variables. It is given by

$$q(\{\{\mathbf{f}_s^{(l)}\}_{l=1}^{s}, \mathbf{u}^{(s)}\}_{s=1}^{t}) = \left(\prod_{s=1}^{t}\prod_{l=1}^{s} p(\mathbf{f}_s^{(l)}|\mathbf{u}^{(l)}, \{X_s, \mathbf{f}_s^{(l-1)}, Z_{l-1}\})\right) \cdot \prod_{s=1}^{t} q(\mathbf{u}^{(s)}) \,, \tag{2.20}$$

where $q(\mathbf{u}^s) \sim \mathcal{N}(\mathbf{m}^{(s)}, S^{(s)})$ with variational parameters $\mathbf{m}^{(s)}$ and $S^{(s)}$. In these formulas, $\mathbf{f}_s^{(0)}$ serves as an empty placeholder to simplify notations. The ELBO (2.16) for this model then becomes

$$\begin{aligned}\mathcal{L}(q(\{\{\mathbf{f}_s^{(l)}\}_{l=1}^{s}, \mathbf{u}^{(s)}\}_{s=1}^{t})) = &\sum_{s=1}^{t}\sum_{i=1}^{N_s} \mathbb{E}_{q(f_{s,i}^{(s)})}[\log p(y_i^{(s)}|f_{s,i}^{(s)})] \\ &+ \sum_{s=1}^{t} \mathrm{KL}[q(\mathbf{u}^s)||p(\mathbf{u}^{(s)}|Z^{(s-1)})]\end{aligned} \tag{2.21}$$

and samples from $q(f_{s,i}^{(s)})$ are obtained by using a sampling scheme similar to that of Salimbeni and Deisenroth [35].
Several additional factors need to be considered. For layer s, the pseudo-inputs $Z^{(s-1)}$ are $(D+1)$-dimensional, as the input to this layer comprises both the original D-dimensional inputs and the function evaluations from the previous layer in the last dimension. Consequently, the last dimension is linked to the first D dimensions, and freely optimizing the pseudo-input locations becomes rather inappropriate. To circumvent this issue, Cutajar et al. suggest using a subset of the lower-fidelity data as pseudo-inputs and fixing them during optimization. Furthermore, they employ a multi-step training procedure to enhance the stability of the optimization process. Initially, the covariances of the inducing variables and the noise variances are fixed to small values. After a certain number of optimization steps, these parameters are released and optimized alongside all other hyperparameters. Moreover, the authors recommend using the structured covariance function (2.13), and they also propose to extend it to more effectively capture linear relationships between fidelities.

2.5 Related Work

2.5.1 Gaussian Process Models with Uncertain Inputs for Applications beyond Standard Regression

We restrict this thesis to input-uncertainty-aware GP models specifically developed for general regression tasks. Nonetheless, several authors have developed methods tailored to either specialized regression tasks or GP applications other than regression, such as classification or optimization.
One such approach, developed by Daemi et al. [36], addresses situations in which the inputs to a GP regression model are assumed uncertain and, additionally, the training targets are prone to outliers. In order to make the GP model robust to outliers, the authors propose modeling the output noise as a mixture of two normal distributions, which allows for heavier tails in the noise distribution. For the purpose of training their model with both an adapted noise distribution and uncertain inputs, the authors developed an expectation-maximization (EM) algorithm. Another input-uncertainty-aware GP regression model was introduced by Tran et al. [37] for scenarios where the inputs are uncertain, but adhere to an ordering constraint. A typical example for this setting occurs when the inputs are noisy measurements of time, but the experimenter knows that the inputs were measured sequentially. The authors propose a variational approach to account for the input uncertainty by defining a variational posterior over the latent inputs after applying a variable transform to these. This variational posterior is modeled as a mixture of normal distributions, which allows for a flexible approximation of the true posterior distribution. The resulting lower bound on the marginal likelihood includes both analytically tractable terms and an intractable term, which the authors approximate using a second-order Taylor approximation. Jadaliha et al. [38] developed a further method for specific scenarios where multiple noisy measurements, taken by a sensor network at a set of sampling points, are available. The authors formulate both a Monte Carlo method and a Laplace approximation-based method to approximate the resulting posterior predictive distribution.
The challenge of accounting for input uncertainty in multi-class GP classification tasks, in which a GP is trained to predict discrete class labels, was addressed by Villacampa-Calvo et al. [39]. For this purpose, the authors introduced three distinct methods. The first method introduces a parametrized variational posterior over the inputs, whereby the number parameters grow with the size of the dataset. In order to mitigate the number of parameters for larger datasets, the authors also suggest modeling these parameters using a neural network. Their final method extends the Taylor approximation-based approach introduced by McHutchon and Rasmussen [40], which we will also discuss in section 3.1, to the classification task.

Another line of research deals with input uncertainty in GP surrogate models or metamodels for optimization tasks. For instance, Ryan et al. [41] model the objective function and the constraint function in an optimization problem as functions of a GP predictive distribution evaluated at the design variables to be optimized. These variables are assumed to be uncertain, and the authors employ methods for predicting at uncertain inputs to evaluate the moments of the predictive distribution analytically. Furthermore, in the context of Bayesian optimization, Bodin et al. [42] consider an unknown objective function, which is assumed to be ill-behaved. They propose utilizing a GP surrogate model to capture the well-behaved essential structure of the objective function while modeling its ill-behaved components as additional uncertain input dimensions to the GP surrogate model.

2.5.2 Related Multi-Fidelity Surrogate Modeling Approaches

We limit our work about multi-fidelity GP models to the NARGP and the MFDGP, both of which can be regarded as nonlinear extensions to the linear autoregressive model discussed in subsection 2.4.2. Beyond these methods, several authors have developed alternative nonlinear extensions of the linear autoregressive model. For instance, Qian and Wu [43] replace the scaling constant ρ in (2.10) with a nonlinear function modeled by a separate GP. They also allow for measurement noise in the high-fidelity target values, as they focus on applications in which the high-fidelity data is formed by observations of experiments. Their model further adopts a Bayesian approach for the majority of the parameters, integrating them out in the predictive distribution. As a result, evaluating the resulting predictive distribution requires Monte Carlo Markov-Chain (MCMC) sampling.
An alternative to correcting the low-fidelity surrogate through both a multiplicative correction term ρ and an additive correction term δ, as in (2.10), was considered by Fischer et al. [44]. Their work, which was developed for design optimization, introduces a weighting function that enables dynamic switching between multiplicative and additive corrections. To obtain both correction terms, the original low-fidelity and high-fidelity models are first approximated by GP surrogates. The correction terms were then obtained by dividing and subtracting the surrogates, respectively, and can be individually applied to the original low-fidelity model to approximate the high-fidelity model. The corresponding weighting function is then dynamically evaluated in a Bayesian manner to balance between the additive and multiplicative correction terms.

Another extension of the linear autoregressive model was proposed by Raissi and Karniadakis [45], which significantly improves its predictive performance when the underlying ground truth exhibits discontinuities. In their work, the authors initially map the inputs in both the low-fidelity and high-fidelity datasets through a deep

neural network. The resulting outputs are then used as inputs to the corresponding GP surrogates within the linear autoregressive model. Both the neural network and the GPs can then be jointly trained.

3 Single- and Multi-Fidelity Gaussian Process Regression with Uncertain Inputs

3.1 Single-Fidelity Gaussian Process Regression with Uncertain Inputs

In this section, we explore how input uncertainty can be incorporated in single-fidelity GP regression models. If the training inputs are uncertain, calculating the marginal likelihood and the predictive distribution for a GP model becomes more complex. Ideally, the uncertain training inputs would be marginalized out in the computation of these terms. However, the inputs typically appear in a highly nonlinear way in both the predictive distribution and the marginal likelihood, which makes marginalization generally intractable [46, 47]. To address this challenge, several approximate methods have been proposed. We introduce some of these methods and compare their performance on a synthetic experiment. While our primary focus is on uncertain training inputs, test inputs may also be subject to uncertainty. Although we consider deterministic test inputs in our experiment, we briefly discuss ideas for incorporating uncertain test inputs when they are mentioned by authors, as they will be relevant for section 3.2. For a collection of methods for predicting at uncertain test inputs in standard GP models, see [48].

3.1.1 Methods for Gaussian Process Regression with Uncertain Inputs

Employing the Expected Covariance Function

One approach to account for uncertain inputs is to absorb the uncertainty into the covariance function by calculating its expectation with respect to the latent inputs. For this approach, we initially consider the input to a GP as a stochastic process over the input space $\mathcal{X}$. Specifically, we define the inputs as the stochastic process $\mathbf{x} = \tilde{\mathbf{x}} + \boldsymbol{\varepsilon}(\tilde{\mathbf{x}})$ with $\tilde{\mathbf{x}}$ as argument. Here, $\boldsymbol{\varepsilon}(\tilde{\mathbf{x}}) \sim \mathcal{N}(\mathbf{0}, \Sigma_{\tilde{\mathbf{x}}})$ is assumed independent across observations. When substituting this process as input into an independent GP $f(\mathbf{z}) \sim \mathcal{GP}(\mathbf{0}, k(\mathbf{z}, \mathbf{z}'))$, the resulting process $f(\mathbf{x})$ is generally not Gaussian. Nevertheless, its mean function remains $\mathbf{0}$, and its covariance function can, in some instances, be computed analytically.

The core idea of the method we consider is to use the covariance function of the process $f(\mathbf{x})$ as the covariance function for a GP with certain inputs $\tilde{\mathbf{x}}$ [49]. Consequently, this method approximates the non-Gaussian prior distribution of $f(\mathbf{x})$ with a GP.
This concept has been adopted by Girard [48] and Dallaire et al. [47, 50] to derive uncertainty-aware covariance functions for the training of a GP. In general, these covariance functions can be computed as

$$k_n(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j) = \mathbb{E}_{p(\mathbf{x}_i)p(\mathbf{x}_j)}[\mathrm{cov}(f(\mathbf{x}_i), f(\mathbf{x}_j)|\mathbf{x}_i, \mathbf{x}_j)] = \int k(\mathbf{x}_i, \mathbf{x}_j)p(\mathbf{x}_i)p(\mathbf{x}_j)\mathrm{d}\mathbf{x}_i\ \mathrm{d}\mathbf{x}_j \tag{3.1}$$

where $i \neq j$ and $p(\mathbf{x}_l) = \mathcal{N}(\mathbf{x}_l|\tilde{\mathbf{x}}_l, \Sigma_{\tilde{\mathbf{x}}_l})$. For the case where $i = j$, the covariance equals the variance and

$$k_n(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_i) = \mathbb{E}_{p(\mathbf{x}_i)}[\mathbb{V}(f(\mathbf{x}_i)|\mathbf{x}_i)] = \int k(\mathbf{x}_i, \mathbf{x}_i)p(\mathbf{x}_i)\mathrm{d}\mathbf{x}_i\ . \tag{3.2}$$

As was shown by Seeger [49], and further extended by Girard [48] and Dallaire et al. [47], these integrals can be evaluated analytically when the covariance function is a squared exponential with ARD, as in (2.6), resulting in

$$k_n(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j) = \frac{\sigma_f^2 \exp\left(-\frac{1}{2}(\tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j)^\top(\mathrm{diag}(\boldsymbol{l})^2 + \Sigma_{\tilde{\mathbf{x}}_i} + \Sigma_{\tilde{\mathbf{x}}_j})^{-1}(\tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j)\right)}{|I + \mathrm{diag}(\boldsymbol{\ell})^{-2}(\Sigma_{\tilde{\mathbf{x}}_i} + \Sigma_{\tilde{\mathbf{x}}_j})(1 - \delta(i,j)|} + \sigma_n^2\delta(i,j)\ . \tag{3.3}$$

Additionally, Dallaire et al. [47] show that (3.1) and (3.2) are analytically tractable for linear and polynomial covariance functions, although these will not be further examined in this thesis.
In practice, using the covariance function (3.3) to account for uncertain inputs amounts to a standard GP regression, where the means of the inputs $\{\tilde{\mathbf{x}}_i|i = 1, ..., N\}$ are treated as the actual inputs, and their respective covariance matrices are absorbed into the prior covariance function. Furthermore, the model for the stochastic process $\mathbf{x}$ implies that all inputs should be assumed independent. The covariance function (3.3) also allows for prediction at uncertain test inputs when their means and covariance matrices are provided. In addition, it permits a combination of deterministic and uncertain inputs, by setting the covariance matrices of deterministic inputs to the zero matrix.
Dallaire et al. [47] discuss some further practical issues regarding this approach. On the one hand, taking the mean of the conditional covariance function $\mathrm{cov}(f(\mathbf{x}_i), f(\mathbf{x}_j)|\mathbf{x}_i, \mathbf{x}_j)$ over the latent inputs can be an inappropriate approximation when its distribution has high variance. However, the authors point out that this issue is rarely problematic for inferring the underlying function, particularly when the dataset is sufficiently large. On the other hand, the marginal likelihood can become riddled with local maxima, which complicates the optimization of hyperparameters. Often, standard conjugate gradient optimization methods overestimate both lengthscales and the noise variance,

which, in effect, causes the model to interprete the data as noise. To address this issue, a possible solution is to employ an MAP estimate of the hyperparameters by assigning them a prior distribution, which should be chosen to penalize large values.

Utilizing Taylor Approximations

An alternative approach to account for input uncertainty is to utilize Taylor approximations, which allow for the propagation of input distributions through nonlinear functions. The first Taylor approximation-based method that we examine was developed by Girard and Murray-Smith [51], and we will refer to it as the delta method approximation. In their method, they assume that the inputs are independent of each other and that $\mathbf{x}_i \sim \mathcal{N}(\tilde{\mathbf{x}}_i, \sigma_x^2 I)$. In contrast to the method discussed in the previous section, this approach makes the additional assumption that the dimensions of all inputs are independent and share the same variance σ_x^2, which functions as a learnable hyperparameter. Nonetheless, the authors mention that their method could potentially be extended to more complex noise models using similar arguments as those outlined below. We also note that it appears feasible to adapt this method to cases in which the variance is input-dependend and known a priori. However, we do not further explore possible extensions and focus on the model as originally specified by the authors.
Consider first the problem of propagating uncertain inputs through a deterministic, sufficiently smooth function f. We are interested in the expectation $\mathbb{E}[f(\tilde{\mathbf{x}} + \boldsymbol{\varepsilon})]$, where $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_x^2 I)$. Using a second order taylor expansion around $\tilde{\mathbf{x}}$, and denoting $\nabla f(\mathbf{z})$ as the gradient and $\nabla^2 f(\mathbf{z})$ as the Hessian matrix of f evaluated at $\mathbf{z}$, we obtain

$$f(\mathbf{x}) = f(\tilde{\mathbf{x}}) + (\mathbf{x} - \tilde{\mathbf{x}})^\top \nabla f(\tilde{\mathbf{x}}) + (\mathbf{x} - \tilde{\mathbf{x}})^\top \nabla^2 f(\tilde{\mathbf{x}})(\mathbf{x} - \tilde{\mathbf{x}}) + \mathcal{O}(|\mathbf{x} - \tilde{\mathbf{x}}|^3) \ .$$

Substituting this Tayler expansion into $\mathbb{E}[f(\tilde{\mathbf{x}} + \boldsymbol{\varepsilon})]$ yields the asymptotic approximation

$$\mathbb{E}[f(\tilde{\mathbf{x}} + \boldsymbol{\varepsilon})] \approx f(\tilde{\mathbf{x}}) + \frac{\sigma_x^2}{2} \mathrm{Tr}[\nabla^2 f(\tilde{\mathbf{x}})] \tag{3.4}$$

for $\sigma_x^2 \to 0$, which is also know as the delta method.
When f is considered random with a GP prior distribution $f(\tilde{\mathbf{x}}) \sim \mathcal{GP}(\mathbf{0}, k(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}'))$, the approximation (3.4) suggests to adjust $f(\tilde{\mathbf{x}})$ by $\frac{\sigma_x^2}{2}\mathrm{Tr}[\nabla^2 f(\tilde{\mathbf{x}})]$ to approximately propagate the uncertainty about the input through f. Here, the covariance function $k(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}')$ is assumed to be a squared exponential with ARD which does not integrate the output noise variance. The resulting stochastic process $g(\tilde{\mathbf{x}}) = f(\tilde{\mathbf{x}}) + \frac{\sigma_x^2}{2}\mathrm{Tr}[\nabla^2 f(\tilde{\mathbf{x}})]$ is again a Gaussian [52] with mean function $\mathbf{0}$ and covariance function

$$\mathrm{cov}(g(\tilde{\mathbf{x}}), g(\tilde{\mathbf{x}}')) = k(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}') + \sigma_x^2 \sum_{i=1}^{D} \frac{\partial^2 k(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}')}{\partial \tilde{x}_i^2} + \frac{\sigma_x^4}{4} \sum_{i,j=1}^{D} \frac{\partial^4 k(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}')}{\partial \tilde{x}_i^2 \partial \tilde{x}_j'^2} \ . \tag{3.5}$$

Having defined the mean function and covariance function of the prior GP $g(\tilde{\boldsymbol{x}})$, we can now proceed with standard GP regression methods for training and inference. Similar to the method discussed in the previous section, we use the means of the training input distribution as new deterministic inputs. However, computing the predictive distribution at a test input $\mathbf{x}_*$ as in (2.4) and (2.5) using the covariance function (3.5) assumes that the test input is uncertain. If the test input $\mathbf{x}_*$ is deterministic, though, we are interested in the posterior distribution of $f(\mathbf{x}_*)$ rather than $g(\mathbf{x}_*)$. This distribution is still obtained by using (2.4) and (2.5), but the prior covariance between the function values at certain test inputs and uncertain training inputs is replaced by

$$\operatorname{cov}(f(\mathbf{x}_*), g(\tilde{\mathbf{x}})) = k(\mathbf{x}_*, \tilde{\mathbf{x}}) + \frac{\sigma_x^2}{2} \sum_{i=1}^{D} \frac{\partial^2 k(\mathbf{x}_*, \tilde{\mathbf{x}}')}{\partial \tilde{x}_i^2} \ .$$

An alternative approach to propagate input uncertainty through a GP based on a Taylor approximation was developed by McHutchon and Rasmussen [40], which we will refer to as the mean function derivative method. In their work, the authors assume that only noisy measurements of the inputs are available, which are distributed around latent true inputs. Thus, their assumptions deviate slightly from the noise model discussed in section 2.1. To maintain consistency with the other methods that we examine, we make slight adaptations to their model. As before, we assume that the true inputs $\mathbf{x}$ are centered around their means $\tilde{\mathbf{x}}$, which we consider given. We note that, if these means are regarded as noisy observations, our slight reformulations result in the same model as the original.
When the targets y are generated by a GP model with noisy inputs, i.e.,

$$y = f(\tilde{\mathbf{x}} + \boldsymbol{\varepsilon}_x) + \varepsilon_y \ , \tag{3.6}$$

the distribution of y will no longer be Gaussian, as previously discussed. Here, $f(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, k(\mathbf{x}, \mathbf{x}'))$, and $\boldsymbol{\varepsilon}_x \sim \mathcal{N}(\mathbf{0}, \Sigma_x)$ is i.i.d. across different inputs, with Σ_x being a learnable diagonal matrix of variances. A direct Taylor approximation of f in (3.6) is not useful, as it involves products of derivatives of f with $\boldsymbol{\varepsilon}_x$, which results in non-Gaussian distributions. Instead, the authors propose using the derivative of the mean function of $f(\mathbf{x})$, denoted by $\bar{f}(\mathbf{x})$, which yields the approximate model

$$y = f(\tilde{\mathbf{x}}) + \boldsymbol{\varepsilon}_x^\top \nabla \bar{f}(\tilde{\mathbf{x}}) + \varepsilon_y \ . \tag{3.7}$$

The primary benefit of this model is that it replaces the random derivatives of $f(\mathbf{x})$ with the deterministic gradients of the mean function $\nabla \bar{f}(\mathbf{x})$, which makes (3.7) Gaussian. The authors note that, despite its simplicity, this model does not empirically exhibit any notable decline in performance compared to methods that

incorporate random derivatives or higher-order Taylor approximations. Furthermore, the model (3.7) together with the independence of f, ε_x, and ε_y implies that $p(y|f(\tilde{\mathbf{x}}), \nabla\bar{f}(\tilde{\mathbf{x}})) = \mathcal{N}(y|f(\tilde{\mathbf{x}}), \sigma_n^2 + \nabla\bar{f}(\tilde{\mathbf{x}})^\top \Sigma_x \nabla\bar{f}(\tilde{\mathbf{x}}))$. In other words, the uncertainty in the inputs translates into additional uncertainty in the outputs, with its variance given by the quadratic form $\nabla\bar{f}(\tilde{\mathbf{x}})^\top \Sigma_x \nabla\bar{f}(\tilde{\mathbf{x}})$, which depends on the slope of the mean function. Consequently, the predictions become more uncertain, whenever the gradient is relatively large, which reflects that small changes in the input are expected to have a large influence on the output. Conversely, when the gradient of the mean function vanishes, the effect of the input noise also vanishes.

The predictive distribution for this model can be calculated by replacing $K(X,X) + \sigma_n^2 I$ in (2.4) and (2.5) with $K(X,X) + \sigma_n^2 I + \operatorname{diag}(\Delta_{\bar{f}_*}^\top \Sigma_x \Delta_{\bar{f}_*})$, which adds additional variance to the function evaluations at the uncertain training inputs. Here, $\bar{f}_*$ denotes the mean function of the posterior GP, and $\Delta_{\bar{f}_*}$ is the $N \times D$ matrix in which the i-th row is given by $\nabla\bar{f}_*(\tilde{\mathbf{x}}_i)$. Furthermore, applying diag to a square matrix yields a diagonal matrix of the same dimensionality with the same diagonal elements as the original matrix.

An immediate issue with this predictive distribution is that the mean function depends on its own gradient, which leads to a differential equation which cannot be solved analytically. This problem affects not only predicting but also training: For the calculation of the marginal likelihood (2.8), the output variance must be corrected by $\operatorname{diag}(\Delta_{\bar{f}_*}^\top \Sigma_x \Delta_{\bar{f}_*})$, which requires knowledge about the posterior mean. To circumvent this challenge, the authors propose a multi-step approximate training procedure. First, the posterior distribution of a standard GP with an initial set of hyperparameters is computed. Next, $\operatorname{diag}(\Delta_{\bar{f}_*}^\top \Sigma_x \Delta_{\bar{f}_*})$ is analytically calculated with the mean function of the posterior GP, and is then used to correct the marginal likelihood (2.8). Finally, the marginal likelihood is optimized with respect to the model hyperparameters, including the parameters of the input noise Σ_x, using a gradient descent algorithm. The approximation in this training procedure is introduced when the posterior mean of a standard GP is used to calculate the corrective term $\operatorname{diag}(\Delta_{\bar{f}_*}^\top \Sigma_x \Delta_{\bar{f}_*})$, thereby avoiding the need to solve a differential equation. This process can also be iterated by recomputing the posterior mean function, including the term $\operatorname{diag}(\Delta_{\bar{f}_*}^\top \Sigma_x \Delta_{\bar{f}_*})$ from the current model, and retraining.

Prediction at an uncertain test input with mean $\tilde{\mathbf{x}}_*$ can be performed in a similar way. First, the posterior mean is evaluated by treating the mean $\tilde{\mathbf{x}}_*$ as deterministic. Then, its gradient is computed and used to update the posterior variance by $\nabla\bar{f}(\tilde{\mathbf{x}}_*)^\top \Sigma_x \nabla\bar{f}(\tilde{\mathbf{x}}_*)$. The authors also suggest a prediction method based on [53], which is related to the expected covariance function method.

Formulating Variational Lower Bounds on the Marginal Likelihood

As discussed in the beginning of this chapter, when we have access to the distribution of the uncertain inputs, the marginal likelihood of the data would ideally be computed by marginalizing out these inputs. However, this is in general not analytically tractable. To address this issue, some authors approximate the marginal likelihood with a variational lower bound. In this section, we explore several possible approaches.
We first consider two methods developed by Quiñonero-Candela. The first method is described in both a paper with Roweis [46] and his PhD thesis [54], while the second method is outlined exclusively in his thesis. Both approaches formulate the ELBO

$$\log p(\mathbf{y}|\boldsymbol{\theta}) = \log \int p(\mathbf{y}|\boldsymbol{\theta}, X)p(X)\mathrm{d}X \geq \int \log q(X) \frac{p(\mathbf{y}|\boldsymbol{\theta}, X)p(X)}{q(X)} \mathrm{d}X = \mathcal{L}(q, \boldsymbol{\theta}) \ , \quad (3.8)$$

on the log marginal likelihood, where $q(X)$ is a variational posterior over the inputs. In these methods, it is also assumed that all training inputs are independently normally distributed, $\mathbf{x}_i \sim \mathcal{N}(\tilde{\mathbf{x}}_i, \Sigma_{x_i})$, with the parameters $\tilde{\mathbf{x}}_i$ and Σ_{x_i} known. Together, these distributions define the prior distribution $p(X)$ over X.
The first of the two methods defines the variational posterior distribution as a product of N delta distributions, each of which centered around learnable estimates of the true inputs $\hat{\mathbf{x}}_i$, $i = 1, ..., N$. To simplify notations, we aggregate these into a matrix $\hat{X} \in \mathbb{R}^{D \times N}$. This choice of variational distribution means that

$$\mathcal{L}(\hat{X}, \boldsymbol{\theta}) = \log p(\mathbf{y}|\hat{X}, \boldsymbol{\theta})p(\hat{X}) = \log p(\mathbf{y}|\hat{X}, \boldsymbol{\theta}) + \sum_{i=1}^{N} \log p(\hat{\mathbf{x}}_i|\tilde{\mathbf{x}}_i, \Sigma_{x_i}) \ , \quad (3.9)$$

which corresponds to the optimization objective of an MAP estimate for $\hat{X}$. In this expression, we have reparametrized the ELBO $\mathcal{L}$ using the MAP estimates $\hat{X}$ instead of q. The first term in (3.9) is the GP log marginal likelihood (2.8) with $\hat{X}$ as deterministic inputs. The second term is a sum of log densities of normal distributions, which penalizes large deviations of $\hat{\mathbf{x}}_i$ from their respective expected values. For instance, if $\Sigma_{x_i} = \sigma_i^2 I$ for all $i = 1, ..., N$, then

$$\sum_{i=1}^{N} p(\hat{\mathbf{x}}_i|\tilde{\mathbf{x}}_i, \Sigma_{x_i}) = -\frac{1}{2}\sum_{i=1}^{N} \sigma_i^2 - \frac{1}{2}\sum_{i=1}^{N} \frac{|\hat{\mathbf{x}}_i - \tilde{\mathbf{x}}_i|^2}{\sigma_i^2} \ ,$$

which means that each $\hat{\mathbf{x}}_i$ inflicts a penalty that is proportional to the squared eucliden distance from the mean $\tilde{\mathbf{x}}_i$.
A major drawback of this method, as the authors note, is that it is highly susceptible to overfitting. This problem occurs because the penalization term is not strong enough to prevent the optimization procedure from driving all estimates $\hat{\mathbf{x}}_i$ to locations where the

model can perfectly fit the data with no variance. If the noise in the training targets is known a priori, an immediate solution is to fix the noise variance to its known value, though such information is typically not available. However, when alternating between optimizing $\hat{X}$ and $\boldsymbol{\theta}$, the authors observed that overfitting occurs more slowly. This suggests that the optimization procedure may initially approach a good optimum.
Consequently, the authors propose an annealing-type training procedure. In this procedure, the noise variance is initially fixed at a high value, and the ELBO (3.9) is maximized with respect to all other hyperparameters. Subsequently, the noise variance is gradually decreased and the ELBO re-optimized at each step with noise variance held fixed. However, the authors note that determining an appropriate stopping point for this procedure is challenging. In our experiments, we let the noise variance run over a prespecified range of values. For each value, we evaluate the log marginal likelihood on validation data after trainig to select a final choice of hyperparameters.
The second method developed by Quiñonero-Candela employs a stochastic EM training procedure to maximize the ELBO in (3.8). This procedure iteratively alternates between the Expectation step (E-step), which maximizes $\mathcal{L}(q, \boldsymbol{\theta})$ with respect to q while fixing $\boldsymbol{\theta}$, and the Maximization step (M-step), which optimizes $\boldsymbol{\theta}$ with q held constant.
For the E-step, the author decided to sample from the posterior distribution over X, which maximizes $\mathcal{L}(q, \boldsymbol{\theta})$ given $\boldsymbol{\theta}$, rather than parameterizing the variational posterior q. This approach can be interpreted in two ways. First, it can be understood as defining the variational posterior q as a mixture of delta distributions centered at the samples from the actual posterior. Alternatively, it can be viewed as a Monte Carlo approximation of the log marginal likelihood $p(\mathbf{y}|\boldsymbol{\theta})$. This is due to the fact that $\mathcal{L}(q, \boldsymbol{\theta})$ equals the log marginal likelihood when q is the posterior distribution over X. In order to obtain a set of samples from the posterior distribution, the author employs the Hamiltonian MCMC [55]. We do not explicit the algorithm here. Instead, we refer the reader to [56] for a concise explanation. In general, MCMC algorithms successively generate samples from a sequence of distributions that gradually approximate a desired target distribution. Typically, samples from a certain number of initial steps, also referred to as burn-in steps, are discarded to ensure that samples from the remaining steps follow distributions that are sufficiently close to the target distribution. The Hamiltonian MCMC algorithm specifically requires knowledge of the logarithm of the unnormalized posterior $p(\mathbf{y}|X, \boldsymbol{\theta})p(X)$, which is equal to the posterior $p(X|\mathbf{y}, \boldsymbol{\theta})$ up to a scaling constant independent of X, as well as the gradients of this log-posterior. The log unnormalized posterior is given by (3.9), and its gradients are available.
The E-step yields a set of S samples $\{\hat{X}_1, ..., \hat{X}_S\}$ drawn from close approximations of the posterior distribution over X. These samples are then used in the M-step to

optimize

$$\frac{1}{S}\sum_{s=1}^{S}\log p(\mathbf{y}|\hat{X}_s,\boldsymbol{\theta})+\log p(\hat{X}_s)$$

with respect to $\boldsymbol{\theta}$. This quantity is obtained by using a mixture of delta distributions centered at the samples from the E-step as variational posterior in (3.8). It also serves as a Monte Carlo approximation of $p(\mathbf{y}|\boldsymbol{\theta})$. For the purpose of optimization, the terms $\log p(\hat{X}_s)$ can be ignored as they are independent of $\boldsymbol{\theta}$.
Similarly, the obtained samples $\{\hat{X}_1,...,\hat{X}_S\}$ can be utilized to compute Monte Carlo estimates of the posterior mean and posterior variance at certain test inputs. To obtain these estimates, the posterior mean or posterior variance, respectively, is calculated for each sample, treating each sample as a set of deterministic training inputs. Subsequently, the Monte Carlo estimates are computed by averaging over all sample.
This stochastic EM approach does not introduce additional parameters and does not rely on assumptions about the form of the variational posterior to approximate the posterior distribution over X. Nevertheless, it carries a high computational load. In each E-step, S samples are drawn using the Hamiltonian MCMC, and additional samples are computed and then discarded in the burn-in phase. Each sample requires the computation of a specific number of so-called leapfrog steps, which involve evaluating the gradient of (3.9). Therefore, each leapfrog step necessitates the inversion of a covariance matrix of size $N \times N$, which scales cubically with N. In each M-step, several optimization steps are performed. During each iteration, the covariance matrices $K(\hat{X}_s,\hat{X}_s)$ for $s=1,...,S$ have to be inverted, which incurs a computational complexity of $\mathcal{O}(SN^3)$. In addition, for larger datasets or higher input dimensions, the number of samples must generally be increased to achieve a certain accuracy. Therefore, this method is practically limited to small datasets with relatively few input dimensions.
As a final method, we consider the application of the Bayesian GPLVM for uncertain inputs. The original method, developed by Titsias and Lawrence [33] and briefly discussed in subsection 2.4.4, was not specifically designed for GP regression. Instead, its primary application is nonlinear dimensionality reduction, where a vector-valued training set of targets Y is provided and the corresponding inputs X are latent variables of lower dimensionality.
Nonetheless, a subsequent paper by Damianou, Titsias, et al. [57] extended this method to several other applications, including GP regression with uncertain inputs. This uncertainty is modeled as a prior distribution $p(X)$ over the inputs. For more flexible prior distributions than products of delta distributions, an ELBO on the log marginal likelihood, obtained by marginalizing out uncertain inputs X, is generally intractable. However, the authors demonstrate that an analytically tractable ELBO can be derived by introducing inducing variables $\mathbf{u}$ corresponding to M pseudo-inputs Z. These inducing

variables are assumed to follow the GP prior $\mathbf{u} \sim \mathcal{N}(\mathbf{0}, K(Z,Z))$. Together with the training targets, latent function evaluations, and latent inputs, their joint density is given by

$$p(\mathbf{y}, \mathbf{f}, \mathbf{u}, X) = p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{u}, X, Z)p(\mathbf{u}|Z)p(X) \ .$$

An immediate benefit of using inducing variables is that the covariance matrix $K(X,X)$ does not need to be inverted to compute $p(\mathbf{f}|\mathbf{u}, X, Z)$, which simplifies marginalizing out X. This is because $p(\mathbf{f}|\mathbf{u}, X, Z)$ is the density of the noise-free version of the predictive distribution given by (2.4) and (2.5), with $\mathbf{u}$ and the deterministics Z assuming the role of $\mathbf{y}$ and X, respectively.
The variational posterior distribution over $\mathbf{f}$, $\mathbf{u}$, and X is chosen to be of the form

$$q(\mathbf{f}, \mathbf{u}, X) = p(\mathbf{f}|\mathbf{u}, X, Z)q(\mathbf{u})q(X)$$

where $q(X)$ is a Gaussian with mean and covariance parameterized by variational parameters $\mathcal{M}, \mathcal{S}$, respectively, and $q(\mathbf{u})$ is left arbitrary. With this variational posterior, an ELBO on the log marginal likelihood can be formulated as

$$\begin{aligned}
\log p(\mathbf{y}) &= \log \int q(\mathbf{f}, \mathbf{u}, X) \frac{p(\mathbf{y}, \mathbf{f}, \mathbf{u}, X)}{q(\mathbf{f}, \mathbf{u}, X)} \mathrm{d}\mathbf{f} \, \mathrm{d}\mathbf{u} \, \mathrm{d}X \\
&\geq \int p(\mathbf{f}|\mathbf{u}, X, Z)q(\mathbf{u})q(X) \log \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{u}|Z)p(X)}{q(\mathbf{u})q(X)} \mathrm{d}\mathbf{f} \, \mathrm{d}\mathbf{u} \, \mathrm{d}X \\
&= \int p(\mathbf{f}|\mathbf{u}, X, Z)q(\mathbf{u})q(X) \log \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{u}|Z)}{q(\mathbf{u})} \mathrm{d}\mathbf{f} \, \mathrm{d}\mathbf{u} \, \mathrm{d}X - \int q(X) \log \frac{q(X)}{p(X)} \mathrm{d}X \\
&= \mathcal{F}(q(X), q(\mathbf{u})) - \mathrm{KL}[q(X)||p(X)] \ .
\end{aligned} \tag{3.10}$$

The particular form of the variational posterior enables the simplification of the fraction in the logarithm by cancelling the complicated term $p(\mathbf{f}|\mathbf{u}, X, Z)$. Furthermore, for fixed $q(X)$, an analytically tractable Gaussian $q(\mathbf{u})$ which optimizes $\mathcal{F}(q(X), q(\mathbf{u}))$ among all possible distributions can be computed. Consequently, $\mathcal{F}(q(X), q(\mathbf{u}))$ can be reparameterized as $\mathcal{F}(q(X))$, and $q(\mathbf{u})$ can always be selected as the optimal distribution. Additionally, $\mathcal{F}(q(X))$ is analytically tractable for some specific covariance functions, among which is the squared exponential with ARD. The KL divergence between $q(X)$ and $p(X)$ is also analytically tractable when $p(X)$ is Gaussian. In particular, when we assume that all inputs are independent under both the prior and the variational posterior distribution, with $p(\mathbf{x}_i) = \mathcal{N}(\mathbf{x}_i|\tilde{\mathbf{x}}_i, \Sigma_{x_i})$ and $q(\mathbf{x}_i) = \mathcal{N}(\mathbf{x}_i|\boldsymbol{\mu}_i, S_i)$, the KL divergence is given by [58]

$$\mathrm{KL}[q(X)||p(X)] = \frac{1}{2} \sum_{i=1}^{N} \left((\boldsymbol{\mu}_i - \tilde{\mathbf{x}}_i)^\top \Sigma_{x_i}^{-1} (\boldsymbol{\mu}_i - \tilde{\mathbf{x}}_i) + \mathrm{Tr}[\Sigma_{x_i}^{-1} S_i] - \log \frac{|S_i|}{|\Sigma_{x_i}|} - D \right) \ . \tag{3.11}$$

Therefore, the model can be trained by maximizing the analytically tractable ELBO (3.10) with respect to the model hyperparameters $\boldsymbol{\theta}$ and the variational parameters $\{Z, \mathcal{M}, \mathcal{S}\}$ using a gradient descent algorithm. As discussed in subsection 2.4.4, the computational complexity for each training step is in $\mathcal{O}(M^2N)$, which is a significant reduction compared to $\mathcal{O}(N^3)$ when $M \ll N$. However, it should be noted that the number of parameters increases linearly with N, as a parametrized variational posterior is introduced for each training input. Though, the authors note that their model is robust to overfitting, particularly due to the regularization provided by the KL divergence.
To make predictions at new deterministic inputs $\mathbf{X}_*$, we use the variational posterior predictive distribution $q(\mathbf{f}_*|X_*)$. This distribution is Gaussian, and its mean and covariance matrix can be analytically computed. The exact formulas are relatively complex. Therefore, we refer the reader to section 4.1 of the paper by Damianou, Titsias, et al. [57]. Additionally, the authors mention how to predict at uncertain test inputs based on the method by Girard et al. [53].

3.1.2 Experiments on Synthetic Data

After detailing the models in the last section, we now proceed with comparing their performance on a synthetic example. To this end, we have implemented all methods as described in the previous subsection with gpflow 2.9.2 [59], except for the Bayesian GPLVM, which is already available in this library. Additionally, for the delta method approximation and the mean function derivative method, we computed all occuring derivatives analytically.
Preliminary experiments with simpler functions showed that all methods performed well. Consequently, we devised a more challenging scenario. In this scenario, we assumed that the inputs are components of a noisy signal, and we estimated their true states by applying the RTSS described in section 2.2 provided by the pykalman library [60]. For the dynamics of the hidden states in the RTSS, we used the linear regresssion model with drift. As a consequence of employing the RTSS, we lack complete knowledge about the actual distribution of the true inputs, which were instead derived from additional model assumptions on the inputs, a scenario that we consider realistic. Our primary goal is therefore to explore whether, and to what extent, the described input-uncertainty-aware methods can outperform a standard GP regression model trained on smoothed inputs, simulating a realistic scenario.
We generated two input signals by sampling 10 points from a uniform distribution over the interval $[-6, 6]$ and interpolating them using cubic splines, yielding 100 samples per signal. We then added Gaussian white noise with variance 0.1 to these signals which ensures that smoothing them will result in different signals. These two signals constitute

our clean inputs. Subsequently, we corrupted the signals with additional Gaussian white noise with variances of 0.2 and 0.4, respectively, representing noisy measurements. By doing so, we incorporated varying levels of noise into our experiments. To estimate the distribution of the true signal, we applied the RTSS seperately to these two signals. The first of the two signals along with the results of the application of the RTSS are depicted in Figure 3.1.

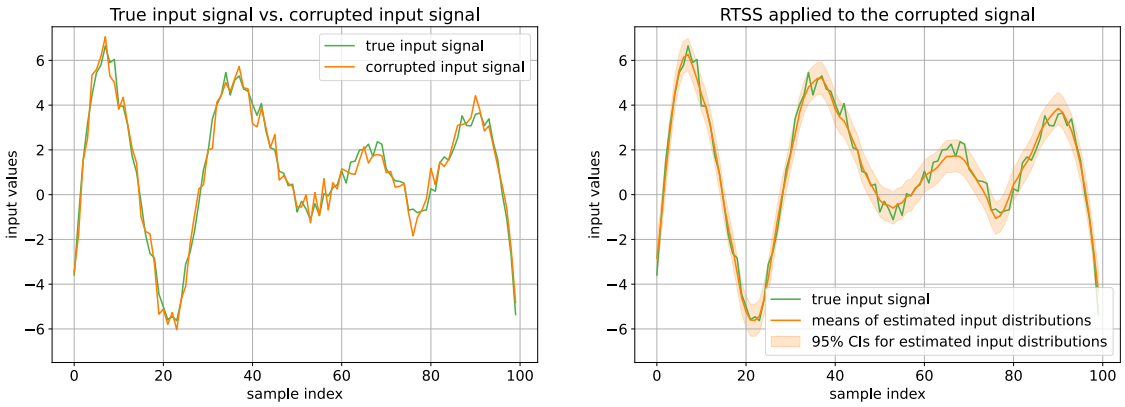


Figure 3.1: The first of the two signals that constitute the inputs. The true signal was corrupted with Gaussian noise with variance 0.2. Subsequently, we applied the RTSS to estimate the true signal from the corrupted signal.

For the underlying function that we aim to infer, we chose

$$f(x) = 3\sin(x\cos(x))\exp\left(-\frac{x^2}{32}\right) ,$$

as we considered this function to be particularly challenging. It exhibits irregular oscillations with a high amplitude near the center at 0 and increasing frequencies towards the edges of the interval $[-6, 6]$. To illustrate the effects of corrupting inputs with noise, we trained standard GP regression models on both corrupted inputs and clean inputs. For this purpose, we utilized half of the inputs from each signal together with their function values, which we corrupted with Gaussian white noise of variance 0.01, as training data. To train the models, we employed the L-BFGS-B [61] optimizer provided by Scipy 1.11.4 [62], which we also used for all other optimization tasks in this set of experiments. The initial values of all hyperparameters were set to 1, and we maintained this choice throughout all experiments in this section unless stated otherwise. The resulting predictive distributions of the standard GP models are shown in Figure 3.2. While a standard GP performs well on a dataset without input noise,

it cannot adequately infer the underlying function when the inputs are corrupted by noise, attributing a high output variance to the training data.

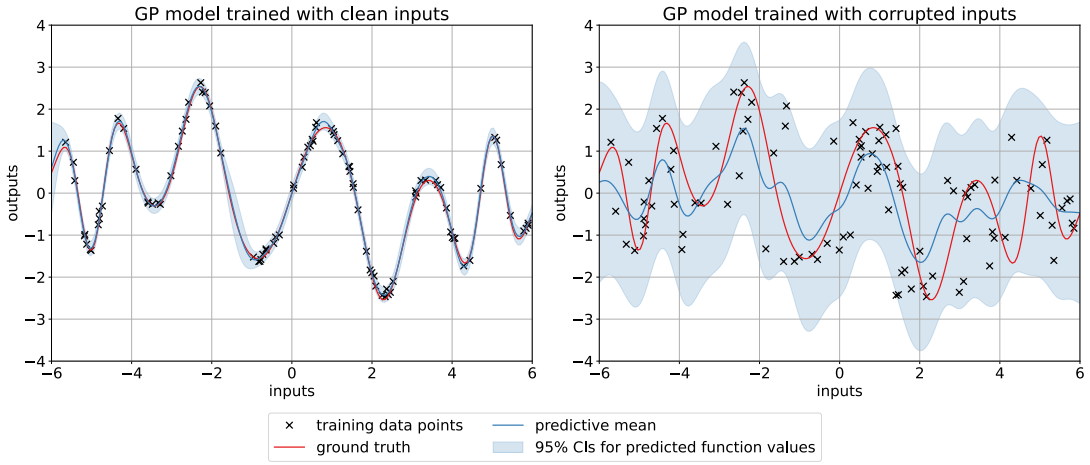


Figure 3.2: A standard GP trained with clean inputs and noisy inputs, respectively.

We obtained fairly better results when we used the means of the estimated input distributions as inputs, which corresponds to smoothing the input signals. The resulting predictive distribution is shown in Figure 3.3, along with the predictive distributions of all methods described in the last section. To train these models, we used the estimated input distributions, or their respective means, in correspondance to the inputs used to train the standard GP models in Figure 3.2. For the mean function derivative method, we iterated the nested training procedure twice, since we could not observe a notable improve in performance with more iterations. In both the mean function derivative method and the delta method approximation, we initialized the input variance to 0.3 to create more comparable conditions to the other methods which had access to more information about the inputs. Regarding the stochastic EM method, we alternated between the E- and M-step 30 times and resampled the training inputs 100 times after 100 burn-in steps. The Bayesian GPLVM used 30 pseudo-inputs, with their initial locations uniformly sampled from the range $[-6, 6]$. Moreover, we trained the model that infers the latent inputs through an MAP estimate in two different ways. We did this to evaluate the effectiveness of the annealing-type procedure, as described in subsection 3.1.1, in preventing overfitting. First, we trained the model by switching between fixing the estimated input locations and fixing all other hyperparameters, using gradient descent to optimize both. Second, we trained it using the annealing-type training procedure, with 80% of the data points used for training and 20% for validation. For this training procedure, we initialized the noise variance to 1 first, and we gradually

lowered it to 0.001 in 1000 iterations. We selected the final hyperparameters based on the setting that obtained the lowest negative log likelihood. The resulting predictive distributions of each trained model are depicted in Figure 3.3. In addition, the SMSEs and MSLLs for 100 test data points, which consist of the unused portions of the input signals along with their function values, are collected in Table 3.1.

Model	Metric	
	SMSE	MSLL
GP trained on clean inputs	0.0102	−2.2304
GP trained on corrupted inputs	0.3911	−0.3584
GP trained on means of the inputs	0.2801	−0.4895
GP with expected covariance function	0.2628	−0.6739
Delta method approximation	0.2656	−0.2851
Mean function derivative method	0.2541	−0.5264
Stochastic EM method	0.2693	−0.5040
Bayesian GPLVM	0.2733	−0.5597
GP with MAP input estimate	2.1326	216 923.6259
GP with MAP input estimate, annealing	0.2351	−0.7424

Table 3.1: SMSE and MSLL on the test dataset for all models considered in our experiments. Lower values correspond to better performance.

Most of the models we analyzed outperformed a standard GP trained on the means of the inputs both with respect to the MSLL and SMSE. We obtained notably strong results using an MAP estimate of the inputs using the annealing-type training procedure. This model adequately captured the higher-frequency oscillations towards the edges of the interval $[-6, 6]$ as well as the region with higher amplitude near the center. However, our implementation of the annealing-type training procedure requires vague prior knowledge of an interval that contains the true noise variance and an increased computation time, as the GP model is retrained for each considered value of the noise variance. The effect of the annealing-type training process on the MAP estimate of the inputs can be assessed by comparing the resulting predictive distribution to that obtained when the noise variance is learned via gradient descent: The noise variance is effectively prevented from approaching 0 by moving the inputs to locations where the GP can fit them without noise variance.
We also achieved good results with the GP model that employs the expected covariance function. However, this model performed worse in higher-frequency regions, which

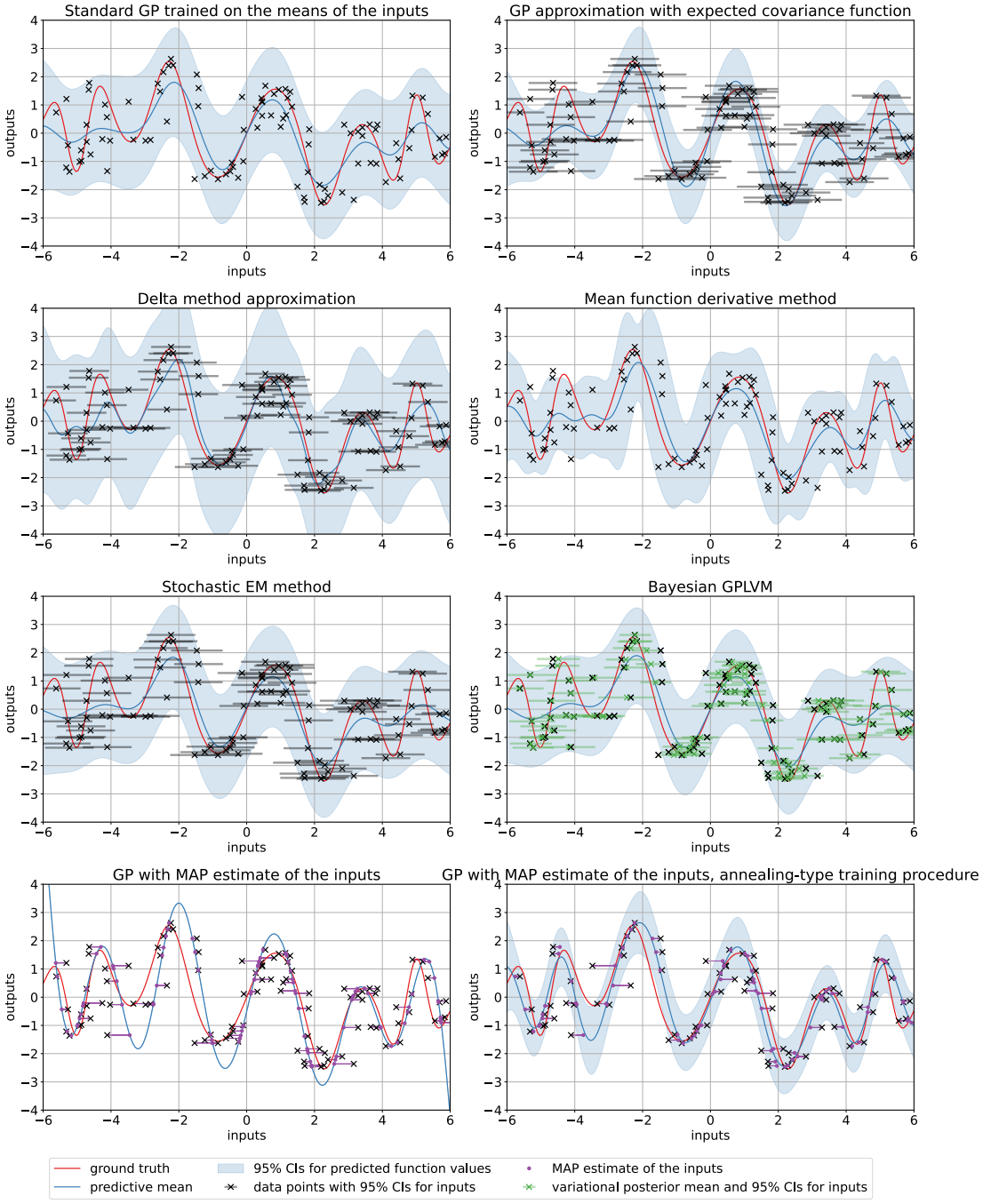


Figure 3.3: Predictive distributions of all described input uncertainty-aware models and of a GP trained with smoothed inputs. We omitted the visualization of the input distribution when it is not necessary or the plots become cluttered.

may be attributed to the fact that incorporating the covariance matrices of the inputs into the covariance function imposes lower bounds to the lengthscales. This issue was also mentioned by Girard [48] and can result in worse performance on higher-resolution functions. One possible solution is to introduce a suitable prior on the lengthscales, but we did not consider this information as given to ensure a fairer comparison.
While the mean function derivative method also yielded a good fit to the data-generating function, it significantly overestimated the input variance, even with reasonable initial values, resulting in broader confidence intervals. The delta method approximation model likewise yielded large confidence intervals and a higher MSLL than the baseline GP which used the input means as deterministic inputs. Nonetheless, the model only slightly underestimated the input noise and provided a reasonably good fit to the latent function.
Furthermore, the improvements of the stochastic EM method were rather modest relative to its training time. Though, the results may be further improved by using more samples or more iterations of the EM algorithm. Lastly, the Bayesian GPLVM only slightly improved the SMSE, but yielded a fairly lower MSLL, while also reducing the computation time.

3.2 Multi-Fidelity Gaussian Process Regression with Uncertain Inputs

In this section, we propose adaptations to the NARGP and MFDGP introduced in section 2.4 to account for uncertainty in the inputs. We validate our methods using a synthetic two-fidelity example. The results show that the proposed methods can sufficiently recover the underlying data-generating functions from data that is corrupted by both input and target noise when the NARGP and the MFDGP fail. Furthermore, we apply our extended MFDGP to rover wheel-soil interaction data and compare its predictive performance to the standard MFDGP.

3.2.1 The Input-Uncertainty-Aware Nonlinear Autoregressive Gaussian Process Model

In the original NARGP model by Perdikaris et al. [21], the authors assumed that the training inputs of adjacent fidelity levels are nested. This assumption is particularly appropriate when we know that the inputs are sufficiently precise and when obtaining lower-fidelity data at the inputs of higer-fidelity levels is inexpensive. The primary benefit of this assumption is that it allows for an analytically tractable training scheme. As the latent function evaluations from the posterior of the GP surrogate for fidelity

level $s-1$ are deterministic quantities at its training data, the inputs for the surrogate for fidelity level s are also deterministic. Therefore, the training procedure for the surrogate $f^{(s)}$ does not depend on the surrogate $f^{(s-1)}$, and all surrogates can be trained independently.
However, if the inputs are uncertain at all fidelity levels, the assumption of nestedness is not reasonable, since we do not know the true inputs that were used to generate the data for each fidelity level. Thus, if the data for each fidelity level was generated by a different technique, we cannot assume that some inputs were coincidentally the same. It is also not a viable solution to assume that the distributions of the training inputs are nested: Data from different fidelity levels typically comes from different sources likely associated with a different level of uncertainty.
As a consequence, we abandon the assumption of nestedness in our data. Instead, we consider a collection of datasets $\mathcal{D}_{s=1}^{t}$ where the inputs in each dataset are independent normal distributions. More specifically, $\mathcal{D}_s = \{(\tilde{\mathbf{x}}_i^{(s)}, \Sigma_{x_i}^{(s)}, \mathbf{y}) | i = 1, ..., N^{(s)}\}$, such that the true input $\mathbf{x}_i^{(s)} \sim \mathcal{N}(\tilde{\mathbf{x}}_i^{(s)}, \Sigma_{x_i}^{(s)})$. While we can no longer rely on the nestedness of our data, we can leverage the GP regression models that account for input uncertainty discussed in section 3.1. Accordingly, we propose stacking input-uncertainty-aware GP regression models to construct an input-uncertainty-aware multi-fidelity GP model. As most of the methods approximate the posterior process with a GP, our approach accordingly approximates the posterior distribution of each surrogate by a GP. A similar idea was already proposed by Perdikaris et al. [21] to simplify computing predictive distributions.
More specifically, we propose the following training scheme. For the lowest fidelity level, we train a GP regression model that accounts for input uncertainty with $\mathcal{D}_1$. To train the surrogate for any fidelity level $s \geq 2$, we first compute the predictive means $\bar{\mathbf{f}}_{(s)}^{(s-1)} | \mathcal{D}_1$ and variances $\mathbb{V}(\mathbf{f}_{(s)}^{(s-1)}) | \mathcal{D}_1$ from the posterior of the surrogate for fidelity $s-1$ at the input distributions in $\mathcal{D}_s$. For that purpose, we propagate the input distributions in $\mathcal{D}_s$ through all surrogates of lower fidelity than s. This involves predicting at uncertain inputs. Subsequently, we add an independent normally distributed dimension to each of the random inputs in $\mathcal{D}_s$, where the distribution of each additional dimension is defined by the corresponding predictive mean and variance obtained from the lower fidelity surrogate. We then train an input uncertainty-aware surrogate for fidelity s on this set of distributions. After iteratively training the surrogates $f^{(s)}$ for $s = 1, ..., t$, we obtain a set of stacked surrogate GPs, which can be used to predict at any fidelity level. We do this for both certain and uncertain test inputs by propagating them through all surrogate models until a desired fidelity level is reached. For deterministic inputs, all surrogates for fidelities greater than 1 must make predictions at inputs where only the last dimension is uncertain.

In addition to assuming Gaussianity of the posterior at each fidelity level and independence of the training inputs, we have made two further approximations to agree with most of the models from section 3.1. First, we assumed independence the latent function evaluations from the posterior GP at a fixed fidelity. This assumption is generally not satisfied, as the latent function evaluations from the posterior are typically correlated, quantified by (2.5). Second, we assumed that the latent function evaluations are independent of their corresponding inputs, which is also generally wrong. Though, this aligns with the assumption that the random inputs have diagonal covariance matrices, and it also facilitates the application of the expected covariance function model with a composite kernel (2.13).
To employ an input-uncertainty-aware GP regression model for this adaptation of the NARGP, it must be capable of handling known input distributions where covariance matrices vary across different inputs. This is required, since the predictive distributions of the surrogates generally do not exhibit constant variances across the latent function evaluations. Consequently, both the delta method approximation and the mean derivative method, as they are described in section 3.1, are not suitable for this task. However, future work might adapt these models to handle known input distributions. Moreover, our adaptation to the NARGP requires a method for making predictions at uncertain inputs. Although Girard [48] provides a flexible framework to predict at uncertain inputs with GP regression models, we do not further consider the MAP estimate method or the stochastic EM method, as these do not directly address uncertain test inputs. Additionally, the stochastic EM method requires a high computation time, and our implementation of the annealing-type training procedure lacks sufficient flexibility.
A further issue with both the expected covariance function method and the Bayesian GPLVM is that they require specific covariance functions. Since the expected covariance function approach provided the most stable results in our experiments, we exlusively focus on this approach in our adaptation to the NARGP. While this methods supports the squared exponential covariance functions with ARD, we intend to use the composite covariance function (2.13) for all fidelity levels higher than 1. Nonetheless, under the independence assumptions previously discussed, we can extend the principles of the expected covariance function method to models that use the composite covariance function. We show this by calculating the expectation of the composite covariance function $k((\mathbf{x}_i, y_i), (\mathbf{x}_j, y_j))$ defined through (2.13) analytically. For this purpose, we assume that $(\mathbf{x}_i, y_i)$ follows a normal distribution with mean $(\tilde{\mathbf{x}}_i, \tilde{y}_i)$ and that y_i is independent of $\mathbf{x}_i$. Furthermore, we define $(\mathbf{x}_j, y_j)$ with mean $(\tilde{\mathbf{x}}_j, \tilde{y}_j)$ for $j \neq i$ in the

same way, and assume that $(\mathbf{x}_i, y_i)$ is independent of $(\mathbf{x}_j, y_j)$. Then it holds that

$$\begin{aligned}
&\int k((\mathbf{x}_i, y_i), (\mathbf{x}_j, y_j)) p(\mathbf{x}_i, y_i) p(\mathbf{x}_j, y_j) \mathrm{d}\mathbf{x}_i \, \mathrm{d}y_i \, \mathrm{d}\mathbf{x}_j \, \mathrm{d}y_j \\
&= \int k_\rho(\mathbf{x}_i, \mathbf{x}_j) \cdot k_f(y_i, y_j) p(\mathbf{x}_i) p(y_i) p(\mathbf{x}_j) p(y_j) \mathrm{d}\mathbf{x}_i \, \mathrm{d}y_i \, \mathrm{d}\mathbf{x}_j \, \mathrm{d}y_j \\
&+ \int k_\delta(\mathbf{x}_i, \mathbf{x}_j) p(\mathbf{x}_i, y_i) p(\mathbf{x}_j, y_j) \mathrm{d}\mathbf{x}_i \, \mathrm{d}y_i \, \mathrm{d}\mathbf{x}_j \, \mathrm{d}y_j \\
&= \int k_\rho(\mathbf{x}_i, \mathbf{x}_j) p(\mathbf{x}_i) p(\mathbf{x}_j) \mathrm{d}\mathbf{x}_i \, \mathrm{d}\mathbf{x}_j \cdot \int k_f(y_i, y_j) p(y_i) p(y_j) \mathrm{d}y_i \, \mathrm{d}y_j \\
&+ \int k_\delta(\mathbf{x}_i, \mathbf{x}_j) p(\mathbf{x}_i) p(\mathbf{x}_j) \mathrm{d}\mathbf{x}_i \, \mathrm{d}\mathbf{x}_j = k_{\rho,n}(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j) \cdot k_{f,n}(\tilde{y}_i, \tilde{y}_j) + k_{\delta,n}(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j) \,,
\end{aligned}$$

where each kernel on the right-hand side has the form (3.3) without the noise variance term. A similar result applies to the expectation of $k((\mathbf{x}_i, y_i), (\mathbf{x}_i, y_i))$ with respect to $p(\mathbf{x}_i, y_i)$. Consequently, we use the uncertainty-aware covariance function

$$k_n((\tilde{\mathbf{x}}_i, \tilde{y}_i), (\tilde{\mathbf{x}}_j, \tilde{y}_j)) = k_{\rho,n}(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j) \cdot k_{f,n}(\tilde{y}_i, \tilde{y}_j) + k_{\delta,n}(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j)$$

for all fidelity levels except the first one.
Lastly, we note that training each surrogate independently neglects that the posterior for the surrogate at fidelity s depends on the hyperparameters from all lower fidelity levels. This may be achieved by propagating the gradients for the hyperparameters through all higher fidelity levels and training all surrogates collectively. However, we currently retain the independent training procedure for simplification. Thus, this approach, with all its assumptions and simplifications, may be interpreted as the path of least resistance and also serves as a baseline for the more complex approach we consider in the next subsection.

3.2.2 The Input-Uncertainty-Aware Multi-Fidelity Deep Gaussian Process Model

The MFDGP discussed in subsection 2.4.4 employs a variational framework to approximate the intractable log marginal likelihood of the training targets. This intractability follows from the fact that the function evaluations of each surrogate, which are inputs to the subsequent fidelity level, are latent. Thus, a natural way to incorporate input uncertainty is to also treat the inputs as latent variables, as we will explain in this section. Our approach is therefore closely aligned with the application of the Bayesian GPLVM for uncertain inputs. However, we follow the approach by Cutajar et al. [22], who consider a different variational approach that results in an ELBO that is not analytically tractable.
As in the previous subsection, we assume that the distribution of the inputs is provided

for each fidelity level, and that the inputs are independent across fidelities. These distributions serve as the prior distribution over the training inputs, which consequently factorizes across fidelities. To formulate an ELBO on the marginal log likelihood of the training targets, we augment the joint density (2.19) with the uncertain inputs, which factorizes as

$$p(\{\mathbf{y}^{(s)}, \{\mathbf{f}_s^{(l)}\}_{l=1}^{s}, \mathbf{u}^{(s)}, X_s\}_{s=1}^{t}) = \left(\prod_{s=1}^{t}\prod_{i=1}^{N_s} p(y_i^{(s)}|f_{s,i}^{(s)}) \prod_{l=1}^{s} p(\mathbf{f}_s^{(l)}|\mathbf{u}_l, \{X_s, \mathbf{f}_s^{(l-1)}, Z_{l-1}\})\right) \cdot \prod_{l=1}^{t} p(\mathbf{u}^{(l)}|Z_{l-1})p(X_l) \ . \tag{3.12}$$

Accordingly, we incorporate latent inputs that are independent across fidelities in the variational posterior (2.20), i.e.,

$$q(\{\{\mathbf{f}_s^{(l)}\}_{l=1}^{s}, \mathbf{u}^{(s)}, X_s\}_{s=1}^{t}) = \left(\prod_{s=1}^{t}\prod_{l=1}^{s} p(\mathbf{f}_s^{(l)}|\mathbf{u}_l, \{X_s, \mathbf{f}_s^{(l-1)}, Z_{l-1}\})\right) \cdot \prod_{s=1}^{t} q(\mathbf{u}^{(s)})q(X_l) \ , \tag{3.13}$$

where $q(X_s)$ is modeled as a Gaussian distribution, with its mean and covariance matrix parameterized by the variational parameters $\mathcal{M}^{(s)}$ and $\mathcal{S}^{(s)}$, respectively. Consequently, the ELBO (2.21) becomes

$$\begin{aligned}
&\mathcal{L}(q(\{\{\mathbf{f}_s^{(l)}\}_{l=1}^{s}, \mathbf{u}^{(s)}, X_s\}_{s=1}^{t})) \\
&= \mathbb{E}_{q(\{\{\mathbf{f}_s^{(l)}\}_{l=1}^{s}, \mathbf{u}^{(s)}, X_s\}_{s=1}^{t})}\left[\log \frac{p(\{\mathbf{y}^{(s)}, \{\mathbf{f}_s^{(l)}\}_{l=1}^{s}, \mathbf{u}^{(s)}, X_s\}_{s=1}^{t})}{q(\{\{\mathbf{f}_s^{(l)}\}_{l=1}^{s}, \mathbf{u}^{(s)}, X_s\}_{s=1}^{t})}\right] \\
&= \sum_{s=1}^{t}\sum_{i=1}^{N_s} \mathbb{E}_{q(\{\{\mathbf{f}_k^{(l)}\}_{l=1}^{k}, \mathbf{u}^{(k)}, X_k\}_{k=1}^{t})}\left[\log p(y_i^{(s)}|f_{s,i}^{(s)})\right] \\
&+ \sum_{l=1}^{t} \mathbb{E}_{q(\{\{\mathbf{f}_s^{(k)}\}_{k=1}^{s}, \mathbf{u}^{(s)}, X_s\}_{s=1}^{t})}\left[\log \frac{p(\mathbf{u}^{(l)}|Z_{l-1})}{q(\mathbf{u}^{(l)})}\right] + \sum_{l=1}^{t} \mathbb{E}_{q(\{\{\mathbf{f}_s^{(k)}\}_{k=1}^{s}, \mathbf{u}^{(s)}, X_s\}_{s=1}^{t})}\left[\log \frac{p(X_l)}{q(X_l)}\right] \ ,
\end{aligned} \tag{3.14}$$

where we substituted the factorization from (3.12) and (3.13) and cancelled out

$$\prod_{s=1}^{t}\prod_{l=1}^{s} p(\mathbf{f}_s^{(l)}|\mathbf{u}_l, \{X_s, \mathbf{f}_s^{(l-1)}, Z_{l-1}\})$$

in the corresponding fraction. Additionally, we used the product rule for logarithms and the linearity of the expectation to decompose the ELBO into sums. Following

Cutajar et al. [22], we have

$$\mathbb{E}_{q(\{\{\mathbf{f}_s^{(k)}\}_{k=1}^s, \mathbf{u}^{(s)}, X_s\}_{s=1}^t)}\left[\log \frac{p(\mathbf{u}^{(l)}|Z_{l-1})}{q(\mathbf{u}^{(l)})}\right] = -\mathbb{E}_{q(\mathbf{u}^{(l)})}\left[\log \frac{q(\mathbf{u}^{(l)})}{p(\mathbf{u}^{(l)}|Z_{l-1})}\right] = -\mathrm{KL}[q(\mathbf{u}^{(l)})||p(\mathbf{u}^{(l)}|Z_{l-1})]$$

for $l = 1, ..., t$, since $\log \frac{p(\mathbf{u}^{(l)}|Z_{l-1})}{q(\mathbf{u}^{(l)})}$ only depends on $\mathbf{u}^{(l)}$. To simplify the expectations that involve the uncertain inputs, we assume that the inputs for a fidelity level l are independent under both the prior distribution $p(X_l)$ as well as the variational posterior $q(X_l)$. Then, it follows that

$$\mathbb{E}_{q(\{\{\mathbf{f}_s^{(k)}\}_{k=1}^s, \mathbf{u}^{(s)}, X_s\}_{s=1}^t)}\left[\log \frac{p(X_l)}{q(X_l)}\right] = -\sum_{i=1}^{N_l} \mathbb{E}_{q(\mathbf{x}_i^{(l)})}\left[\log \frac{q(\mathbf{x}_i^{(l)})}{p(\mathbf{x}_i^{(l)})}\right] = -\sum_{i=1}^{N_l} \mathrm{KL}[q(\mathbf{x}_i^{(l)})||p(\mathbf{x}_i^{(l)})]$$

for $l = 1, ..., t$. Here, the term $\log \frac{p(\mathbf{x}_i^{(s)})}{q(\mathbf{x}_i^{(s)})}$ depends only on $\mathbf{x}_i^{(s)}$, analogous to the case of the inducing variables. Notably, these negative KL divergences between the variational posterior and the prior over the uncertain inputs act as a regularizing term, which penalizes large deviations from the prior.
Moreover, for $s = 1, ..., t$ and $i = 1, ..., N_s$, it holds that

$$\mathbb{E}_{q(\{\{\mathbf{f}_k^{(l)}\}_{l=1}^k, \mathbf{u}^{(k)}, X_k\}_{k=1}^t)}\left[\log p(y_i^{(s)}|f_{s,i}^{(s)})\right] = \mathbb{E}_{q(f_{s,i}^{(s)})}\left[\log p(y_i^{(s)}|f_{s,i}^{(s)})\right] \ .$$

Similar to the original model, this term is generally intractable and requires an approximation via Monte Carlo integration. For this purpose, we need samples from $q(f_{s,i}^{(s)})$. Since

$$q(f_{s,i}^{(s)}) = \int q(f_{s,i}^{(s)}|\mathbf{x}_i^{(s)}) q(\mathbf{x}_i^{(s)}) \mathrm{d}\mathbf{x}_i^{(s)} \ ,$$

these samples can be obtained by first generating a set of S samples from the multivariate normal $q(\mathbf{x}_i^{(s)})$, and then drawing samples from the conditional $q(f^{(s)}|\mathbf{x}_i^{(s)})$ [63] using the sampling scheme from Salimbeni and Deisenroth [35] discussed in subsection 2.4.4.
Combining all results, the ELBO (3.14) becomes

$$\mathcal{L}(q(\{\{\mathbf{f}_s^{(l)}\}_{l=1}^s, \mathbf{u}^{(s)}, X_s\}_{s=1}^t)) = \sum_{s=1}^t \sum_{i=1}^{N_s} \mathbb{E}_{q(f_{s,i}^{(s)})}\left[\log p(y_i^{(s)}|f_{s,i}^{(s)})\right] - \sum_{l=1}^t \mathrm{KL}[q(\mathbf{u}^{(l)})||p(\mathbf{u}^{(l)}|Z_{l-1})] - \sum_{l=1}^t \sum_{i=1}^{N_l} \mathrm{KL}[q(\mathbf{x}_i^{(s)})||p(\mathbf{x}_i^{(s)})] \ .$$

Finally, the predictive distribution can be computed in the same manner as described in subsection 2.4.4.
In comparison to the method of Cutajar et al. [22], our approach requires the additional computation of the KL divergences between the prior distributions over the uncertain inputs and their corresponding variational posteriors. Furthermore, we also need to sample from the variational posteriors over the inputs to compute the expectations with respect to $q(f_{s,i}^{(s)})$. For a fixed number of input dimensions, the KL divergence between two multivariate normal distributions can be evaluated in constant time using (3.11). Therefore, the computation of all KL divergences involving inputs scales linearly with the total size of the training data. Moreover, for each sample drawn from $q(f_{s,i}^{(s)})$, one additional sample from $q(\mathbf{x}_i^{(s)})$ is required. When the number of input dimensions is fixed, sampling from $q(\mathbf{x}_i^{(s)})$ can be performed in constant time per sample, resulting in an additional computational cost that scales linearly with both the size of the training dataset as well as the number of samples S used to approximate the expectations. Furthermore, our approach introduces parameters for the variational posterior over each input. In our experiments, we parametrized each of these with its mean vector and a diagonal covariance matrix to reduce the number of parameters. With variational posteriors over the inputs defined in this way, an additional $2D(N_1 + ... + N_s)$ variational parameters are needed for the inputs. While this increase in parameters is not ideal, we presume that the KL divergence between the variational posterior and the prior over the inputs will provide adequate regularization to prevent overfitting.
A further issue is the selection of the pseudo-input locations. While Cutajar et al. [22] use subsets from the lower-fidelity data for all fidelities except the first, extending this approach to scenarios with uncertain inputs is not straightforward. We refrain from introducing additional uncertainty over the pseudo-inputs, since it further complicates the calculation of the ELBO. As a short-term solution, we use the input means from the lower fidelity model along with the respective training targets and assume that these are sufficient to represent inputs to our model.
For model training, we achieved the best results by first employing the training procedure proposed Cutajar et al. [22]. During this initial training stage, we fix the variational posteriors over the inputs to their mean vectors and set their variances to a value close to zero. Subsequently, we free the parameters of the variational posterior and train all parameters jointly. Thus, our training procedure corresponds to first training a multi-fidelity DGP on the means of the input distributions, and then correcting it to account for input uncertainty.

3.2.3 Experiments on Synthetic Data

To assess the performance of the methods developed in the previous two subsections, we consider a synthetic example explored by Perdikaris et al. [21] and Cutajar et al. [22]. We generate the low-fidelity data using the function

$$f_l(x) = \sin(8\pi x)$$

and the high-fidelity data through

$$f_h(x) = (x - \sqrt{2}) f_l(x)^2 \, .$$

We follow Cutajar et al. and restrict the inputs in the high-fidelity dataset to roughly the interval $[0, 0.7]$, whereas the inputs for the low-fidelity surrogate cover the broader interval $[0, 1]$. This allows us to compare the performance of the methods in regions where high-fidelity observations are scarce or not available. The inputs are again measurements of a signal which we created in a similar fashion to subsection 3.1.2. Though, we added Gaussian white noise with smaller variances to the input signals. Additionally, we used a smaller noise variance for the inputs to high-fidelity function compared to the low-fidelity inputs, as we considered this choice more realistic. To estimate the true inputs, we modeled the input signals by the linear regression model with drift and applied a RTSS to them, tuning the parameters such that the true signal was approximately contained in 95% of the 1.96 standard deviation confidence intervals. As training data, we sampled 55 and 28 clean inputs, along with their corresponding estimated distributions, for the low-fidelity and high-fidelity datasets, respectively. Subsequently, we computed the corresponding function values and added small Gaussian white noise to them when the inputs were uncertain. The resulting data is shown in Figure 3.4.

For comparison with the standard NARGP, we additionally required nested training data. To this end, we replaced 28 points from the low-fidelity dataset, which had inputs within $[0, 0.7]$, with 28 new training points generated by evaluating the low-fidelity function at the high-fidelity inputs. In the experiments with noisy inputs, we substituted these inputs with the means of their respective input distribution and generated the nested training data in a similar manner.

To implement the baseline models and our proposed extensions, we modified the implementations of the NARGP [64] and MFDGP [65] to be compatible with GPflow 2.9.2 and incorporated our additions proposed in subsection 3.2.1 and subsection 3.2.2. In all experiments with versions of the MFDGP, we utilized all low-fidelity training inputs or their estimated means as pesudo-inputs for the low-fidelity surrogate. For the high-fidelity surrogate, we used half of these inputs, concatenated with their corresponding target values, as pseudo-inputs.

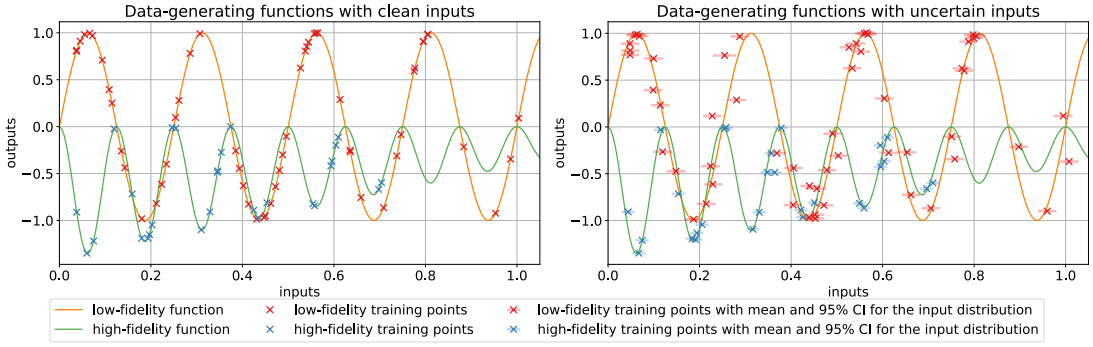


Figure 3.4: The underlying low-fidelity and high-fidelity functions together with the data points with and without uncertainty in the inputs.

To train the standard MFDGP with clean inputs, we employed the Adam optimizer [66] for 3,000 optimization steps while fixing the noise variances and the variances of the inducing variables at a value close to zero. Afterward, we ran the optimizer for an additional 15,000 steps, during which all hyperparameters were optimized simultaneously. We assigned a learning rate of 0.003 to the optimizer for the initial set of iterations and reduced it to 0.001 for the second set of iterations. The numbers of training steps and the learning rates were chosen to achieve a comparable result to that presented by Cutajar et al. for the MFDGP trained on clean inputs. The resulting predictive distributions are shown in Figure 3.5. In general, fewer optimization steps were needed for the models to converge. However, in some instances, we observed that the expected log likelihoods in (2.21) converged first, while the penalty term from the KL divergences continued to improve. After a few more iterations, the increase of the penalty term came at the expense of a significant reduction of the expected log likelihoods. As a result, the hyperparameters converged to an optimum where the KL divergences inflicted only a small penalty, but the target variance was highly overestimated. We experimented with different numbers of optimization steps, learning rates, and applying exponential decay to the learning rates, but were unable to find a general way to prevent this issue.
We obtained comparably strong results with a standard NARGP trained on clean inputs, employing the L-BFGS-B optimizer from Scipy 1.11.4. However, the performance of the of the high-fidelity surrogate significantly declined within the interval $[0.8, 1]$, which we attribute to the scarcity of low-fidelity training points in this region. When the training inputs were uniformly sampled from $[0, 1]$ and $[0, 0.7]$, respectively, we obtained an appropriate fit to the high-fidelity function within the entire interval $[0, 1]$.
When both the MFDGP and the NARGP were trained using only the means of the

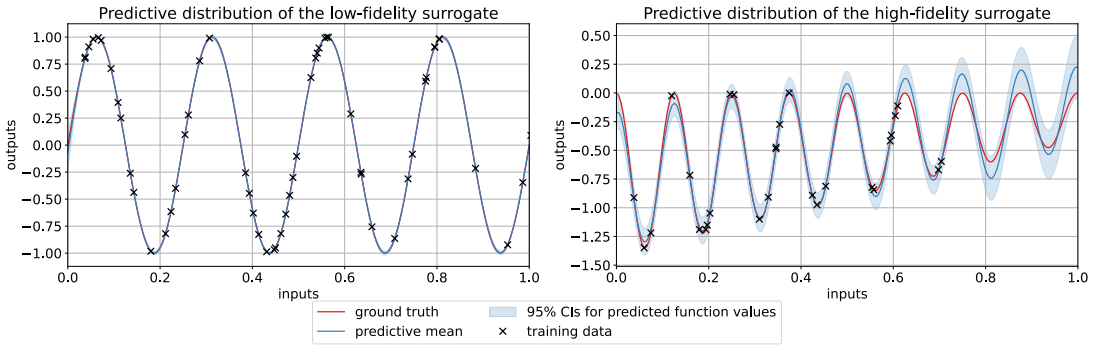


Figure 3.5: Predictive distribution for each fidelity level of the MFDGP trained with clean inputs. For the low-fidelity surrogate, the predictive mean closely aligns with the underlying function and the variance

estimated input distributions in the presence of input uncertainty, their performance significantly deteriorated on this synthetic example. This approach effectively trains the models on smoothed input signals, disregarding the distribution of the inputs. The resulting predictive distributions are shown in Figure 3.6 and Figure 3.7.
For the MFDGP, we used the same number of optimization steps and learning rates as specified for training the MFDGP on clean inputs. During training, we observed the previously discussed issue that the expected log likelihoods were significantly decreased in order to improve the penalty term. Under different training configurations, we occasionally obtained a slightly better fit, with an appropriate predictive mean of the high-fidelity surrogate for inputs in the interval $[0.1, 0.3]$.
Moreover, the NARGP attempts to fit all noisy data points exactly, resulting in small lengthscales and strong oscillations. These results can be expected, as the NARGP assumes noise-free data. Therefore, it is ill-suited for application to noisy data. Nevertheless, it is noteworthy that the predictive means mostly remain close to both the low-fidelity and high-fidelity functions.
We trained our input-uncertainty-aware models using the input distributions estimated by the RTSS, which once again resembles the realistic scenario that we do not have exact knowledge about the distribution of the inputs. For our extension to the MFDGP, we initially trained our model as a standard MFDGP by fixing the variational posterior over the inputs fixed to the means of the prior, with a variance close to zero. In this initial training stage, we ran 3,000 optimization steps in its first phase and 7500 optimization steps for its second phase, employing a learning rate of 0.003 and 0.001, respectively. Subsequently, we released the variational parameters for the variational posterior over the inputs and jointly trained all hyperparameters for another 7,500 training steps with

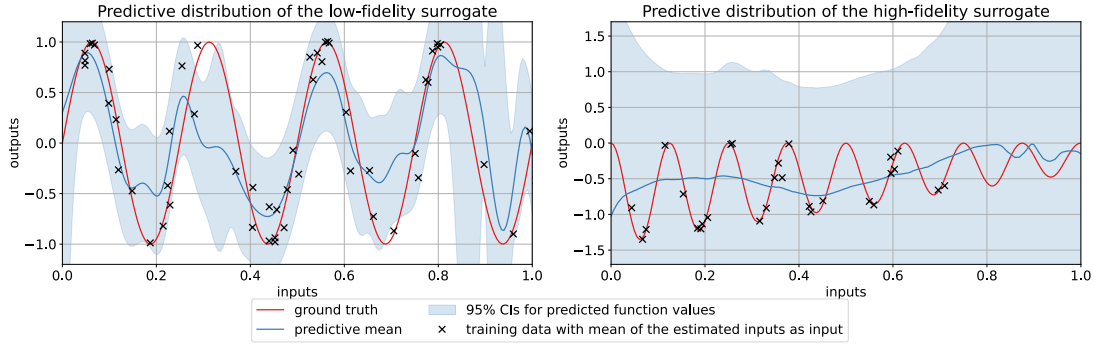


Figure 3.6: The predictive distribution of both MFDGP layers, trained on the means of the estimated inputs.

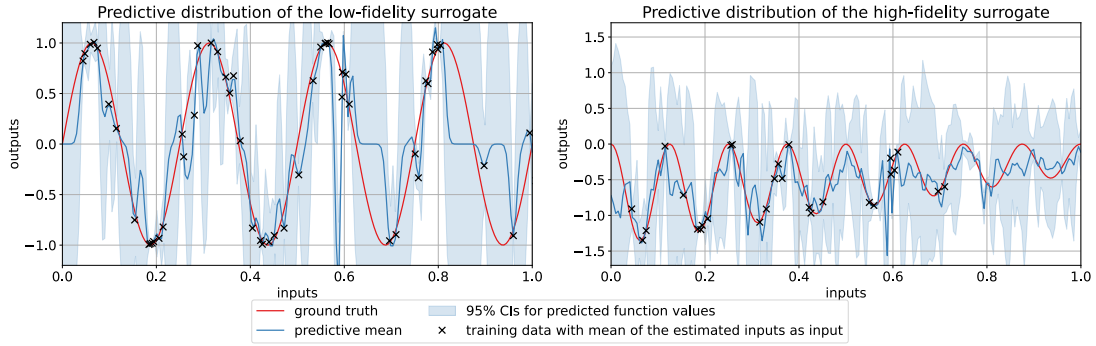


Figure 3.7: The predictive distribution of both NARGP layers, trained on the means of the estimated inputs.

a learning rate of 0.001. The resulting predictive distributions of our extensions are illustrated in Figure 3.8 and Figure 3.9.

To quantify the performance of all models, we calculated the SMSE and MSLL for each model trained. The test data for the low-fidelity surrogates were created by evaluating the low-fidelity function on 100 equally spaced inputs spanning the interval $[0, 1]$. For the high-fidelity test data, we distinguished between the input interval $[0, 0.7]$, which is covered by both low- and high-fidelity training points, and the input interval $[0.7, 1]$, which is exclusively covered by low-fidelity training data. In both cases, we evaluated the high-fidelity function for equally spaced input points, using 100 data points for the interval $[0, 0.7]$ and 50 data points for $[0.7, 1]$. Furthermore, we used a Gaussian approximation of the predictive distribution to compute the MSLL for all versions of the MFDGP. The results are presented in Table 3.2.

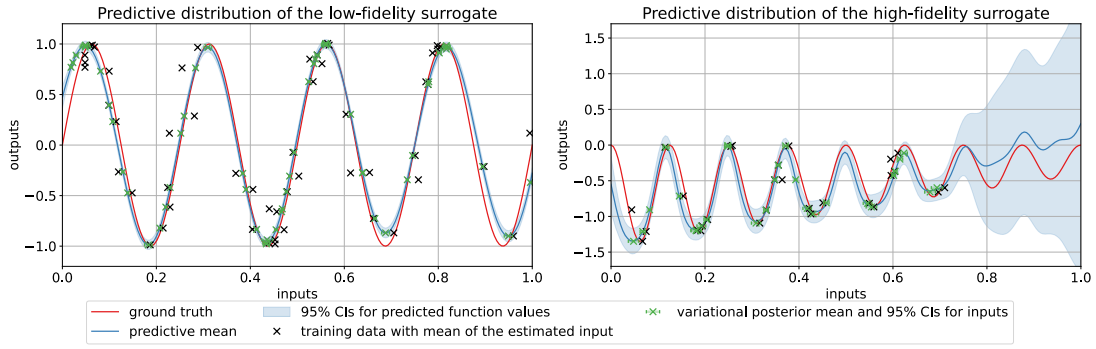


Figure 3.8: Predictive distribution of both layers of our input-uncertainty-aware adaptation to the MFDGP.

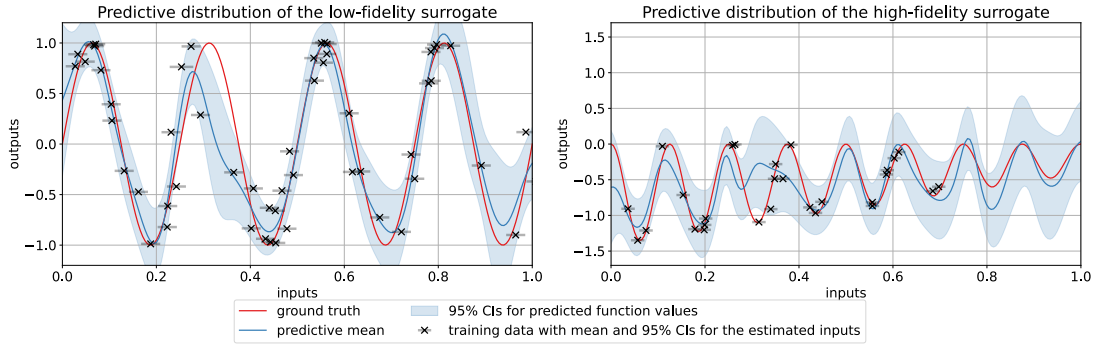


Figure 3.9: Predictive distribution of both layers of our input-uncertainty-aware adaptation to the NARGP.

As shown in Figure 3.8, our extension of the MFDGP adequately recovers the low-fidelity function and the high-fidelity function within the area covered by high-fidelity data points. Additionally, it roughly generalizes the shape of the high-fidelity function to the region solely covered by low-fidelity training data. Nonetheless, the SMSE in this area is rather high and the generalization is quite imprecise, accompanied by a large predictive variance. It is also noteworthy that our model demonstrated great performance in recovering the low-fidelity function, reducing the SMSE of a standard MFDGP by a factor of 10. In particular, it correctly recovers the latent inputs of the low-fidelity training data within the interval $[0.2, 0.6]$ with minimal variance. Nevertheless, our model, trained with the specified training configuration, predicts overly narrow confidence intervals, resulting in a high MSLL for the low-fidelity test data and a relatively high MSLL for the high-fidelity test data with inputs in $[0, 0.7]$. These narrow

Model	Metric for low-fidelity test data test data		Metric for high-fidelity test data within $[0, 0.7]$		Metric for high-fidelity test data within $[0, 1]$	
	SMSE	MSLL	SMSE	MSLL	SMSE	MSLL
MFDGP trained with clean inputs	0.000	−4.891	0.019	−2.074	0.364	−1.250
MFDGP trained with estimated input means	0.336	−0.472	1.102	0.307	1.614	0.751
MFDGP with variational posterior over inputs	0.033	6.637	0.321	0.371	1.500	−0.420
NARGP trained with clean inputs	0.000	−5.652	0.0159	−2.802	4.335	−0.037
NARGP trained with estimated input means	0.419	38.660	0.645	0.675	0.799	−0.863
NARGP with expected covariance function	0.125	−1.312	0.479	0.552	0.805	−0.703

Table 3.2: Results for the toy data experiments for different sets of test data. All values were rounded to the third decimal place. Lower values indicate better performance.

confidence intervals can be attributed to slightly inaccurate variational posteriors over the inputs with high confidence.
However, other training configurations resulted in broader confidence intervals, while adequately preserving the shape of the low-fidelity function. Although the high-fidelity function could generally be recovered, our model interpreted small peaks in the valleys of the high-fidelity function which increased with larger input values. Consequently, the MSLL could be reduced at the expense of a higher SMSE.
In contrast to the predictive distribution generated by our input-uncertainty-aware MFDGP, our modification of the NARGP predicted broader confidence intervals, but generally performed worse in recovering both the underlying low-fidelity and high-fidelity function. Nonetheless, it provided better approximations to the data-generating functions than the baseline models. Furthermore, while achieving merely a relatively high MSLL for the high-fidelity test data with inputs in $[0, 0.7]$, it predicted relatively narrow confidence intervals that still managed to cover the underlying functions in most regions. Thus, this method can be regarded as a simpler yet lower-quality alternative to our MFDGP extension, which required more iterations to converge and parametrized

each input with a variational posterior.

3.2.4 Application to Rover Wheel-Soil Interaction Modeling

We further evaluate the performance of our extension to the MFDGP by applying it to model the movement of a rover wheel on soft soil. Maneuvering rovers on extraterrestrial surfaces requires highly accurate predictions of the locomotion of the vehicle, as rovers typically cannot be maintained or repaired on extraterrestrial missions, and any mistake can jeopardize the entire mission [4]. When operating on planetary bodies, rovers often traverse soft deformable soils such as sand or dust. Under these conditions, the rover encounters significant challenges, such as slip-sinkage, which refers to increased vehicle sinkage and motion resistance due to vehicle slip [67]. Such situations can result in the rover becoming trapped in the soft soil, and therefore necessitate explicit modeling of the wheel-soil interaction, as kinematic theory is insufficient for such complex dynamics [68]. However, no exact physical models exist to describe these interactions [69]. While several numerical simulation tools are available, they either demand considerable computational resources or yield relatively inaccurate predictions, which renders them ill-suited for application in extraterrestrial missions. A promising alternative is to employ machine learning-based surrogate models using data from simulations or experiments to build a prediction tool which is both fast and accurate [4].

Several machine learning-based approaches for modeling rover wheel-soil interaction have been successfully developed [3]. In our work, we build upon the multi-fidelity GP approaches presented by Ravi, Fediukov, et al. [6]. Their experiments aim to predict the traction force acting on a wheel, which is a critical factor in modeling rover wheel locomotion [4]. The inputs to their models comprise four three-dimensional vectors: the position of the center of the wheel relative to the ground, the velocity of the wheel, its angular velocity, and the direction of gravity. The authors considered multiple data sources, among which were the Soil Contact Model [69] (SCM), a semi-empirical simulation model with intermediate approximation capability [4], and the Terramechanics Robotics Locomotion Lab (TROLL) testbed at the German Aerospace Center, a physical test rig for automated analysis of ground interactions [70]. Their findings demonstrate that two fidelity levels suffice to model the traction force using multi-fidelity GPs, with the SCM serving as the low-fidelity data source and the TROLL testbed as the high-fidelity data source. Additionally, they achieved superior results with the MFDGP. Therefore, we do not consider the application of the NARGP and its extension, as these models also produced inferior results in the experiments from the previous subsection. The data for both the low-fidelity and high-fidelity surrogates is generated as a series of runs, each producing inputs and target values represented as signals. Ravi, Fediukov,

et al. [6] note that only a constant application of traction force has an impact on the movement of a wheel. Consequently, they treat spikes in the traction force as noise and smooth all signals. Our experiments aim to investigate whether cutting off higher-frequency components in the signals can negatively impact the performance of the models, as this approach corresponds to discarding information in the data. While we still work with smoothed signals in our experiments, we allow for the true inputs to be unknown and distributed around these smoothed signals.
For the low-fidelity training dataset, we utilized 200 runs generated by the SCM. Each of these runs was 20 seconds long and we sampled data points at a rate of 10 points per second. The high-fidelity training dataset was derived from 38 runs in the TROLL testbed, all with the same sampling rate but lengths ranging from 8 seconds to 77 seconds, resulting in 8,610 data points. For testing, we reserved an additional 12 runs from the TROLL testbed, with lengths varying between 9 and 77 seconds, resulting in 3,140 test data points.
After normalizing the data, we applied the RTSS to each run and each input dimension separately. We also used it to smooth the target values. This approach does not account for correlations between individual input dimensions, as well as between inputs and targets, and it assumes that the inputs are normally distributed. We made these assumptions to comply with our model formulation. For simplicity, we also employed the linear regression model with drift in the RTSS. To apply the RTSS, we were required to specify the covariance matrices for both the process and measurement noise, as well as the distribution of the initial state, individually for each signal. For the mean of the initial states, we used the first state in the respective signal and the difference between the first and the second state. For each signal, we further calculated its sample variance, as well as the sample variance of the differences between neighboring samples. We scaled these variances by constants and then used them as initial parameters in the covariance matrices for the RTSS, which we assumed to be diagonal. Subsequently, we executed three iterations of the EM algorithm provided by pykalman [60] to further adapt the parameters of the RTSS to each respective signal. We obtained mostly relatively under-confident confidence intervals, which was intentional, since all signals were provided to us pre-smoothed. Exemplary cases from the TROLL dataset are shown in Figure 3.10.
We further randomly subsampled the training datasets, resulting in 3,000 low-fidelity training points and 600 high-fidelity training points. For the standard MFDGP, we used the means of the input distributions as training inputs, which corresponds to using smoothed signals as inputs. Conversely, we used the estimated input distributions to train our extension to the MFDGP. Both training datasets included the smoothed version of the traction force as target values, discarding their variance estimates from the RTSS. For both models, we used 600 pseudo-inputs for the low-fidelity surrogate

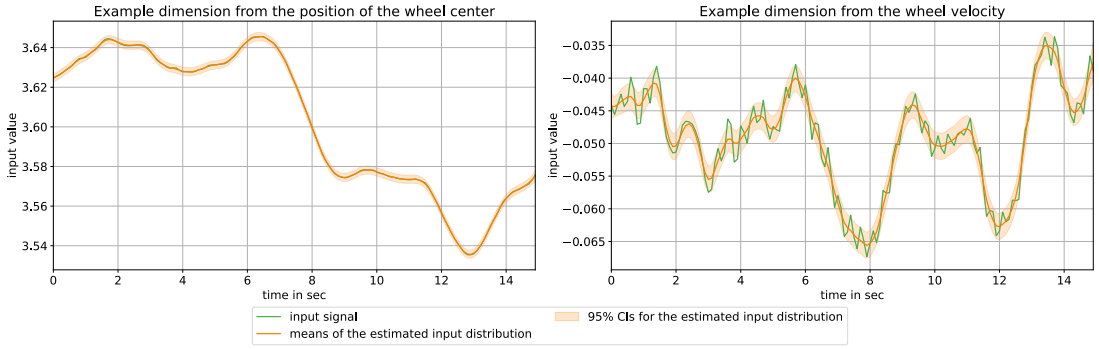


Figure 3.10: Examples for the application of the RTSS to an input signal for the high-fidelity data. In the graphic on the left, the signal is relatively smooth, resulting in under-confident confidence intervals. Such cases were common.

and 200 pseudo-inputs for the high-fidelity surrogate. The low-fidelity pseudo-inputs consisted of randomly sampled means from the distributions of the training inputs. Since SCM evaluations were available for all high-fidelity training inputs, we randomly sampled 200 input means from the high-fidelity training dataset and concatenated them with the respective traction force predictions from the SCM to create the set of high-fidelity pseudo-inputs. We hypothesized that this choice of pseudo-inputs might yield better performance, as the pseudo-inputs would align with the inputs for the high-fidelity surrogate.

We devided the training procedure of the standard MFDGP into 3,000 optimization steps for the initial training phase, followed by 20,000 additional iterations for the second training phase, employing learning rates of 0.003 and 0.001, respectively. For our extension to the MFDGP, we split the training process into three stages consisting of 3,000, 10,000 and another 10,000 optimization steps, employing learning rates of 0.003, 0.001, and 0.001, respectively. For both models, the differences between the respective training phases is analogous to those discussed in the previous subsection. The resulting predictive means of the high-fidelity surrogates for the test dataset are illustrated in Figure 3.11, and the corresponding loss metrics, along with some statistics for the resulting standard deviations, are presented in Table 3.3.

The results show that our input-uncertainty-aware MFDGP variant outperformed the standard MFDGP in terms of both the test MSLL and the test SMSE, which can also be seen by comparing the predictive means in Figure 3.11. Yet, there are some regions where the standard MFDGP provides a better fit to the test data than our version. Both models significantly overestimated the output variance, resulting in confidence

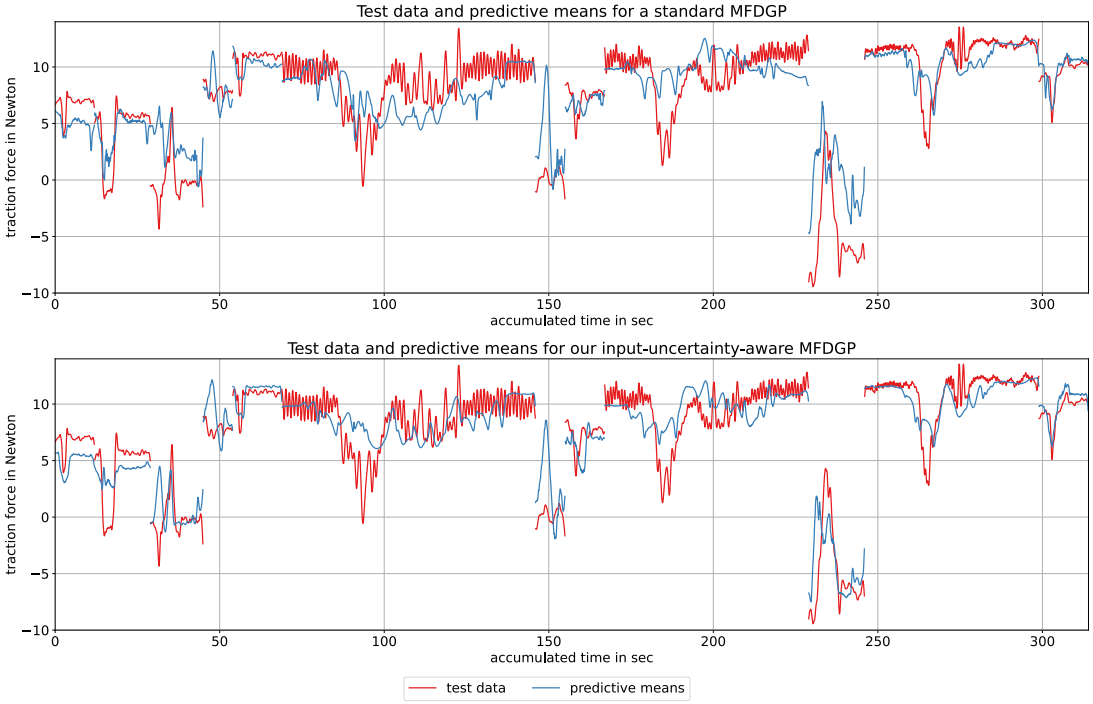


Figure 3.11: Predictive means of the considered model along with the test data. All test runs were accumulated into single plots and discontinuities indicate the transition between separate runs. We omitted the confidence intervals for clarity. Average and bounds of the standard variations can be found in Table 3.3.

intervals that were so broad that they lack practical meaning. This issue is also reflected in the relatively high MSLLs. As previously mentioned, we observed a similar issue with the standard MFDGP in our toy data experiment. However, in this real-world data experiment, we could not observe a decrease of the expected log likelihoods during training. Additionally, we did not obtain notably smaller standard deviations with other configurations of the training procedure. Nevertheless, our variant of the MFDGP still produced slightly smaller standard deviations compared to the baseline model.
We also note that Ravi, Fediukov, et al. [6] achieved lower SMSEs with their implementation of the MFDGP than both the models we considered. Moreover, their experiments did not exhibit the issue of significantly overestimated confidence intervals. However, our experiments differed in several ways. For training, we used different subsets from the original datasets, which also differed slightly from the ones they used. In addition, the numbers of training points and pseudo inputs in our experiment deviated from

Model	Metric		Standard deviations		
	SMSE	MSLL	Minimum	Average	Maximum
MFDGP baseline	0.3157	0.1634	7.2153	9.4333	26.0149
input-uncertainty-aware MFDGP	0.1977	-0.0018	6.1818	8.1873	22.9103

Table 3.3: Test metrics as well as the standard deviations for the application of both variants of the MFDGP to the rover wheel locomotion dataset. Lower values correspond to better performance.

theirs. We also smoothed all signals differently and adapted the parameters of the smoother to each signal individually. Apart from that, we implemented the MFDGP in a different way, along with another training procedure and optimizer. These factors likely contributed substantially to the reduced predictive performance of our models. Although our variant of the MFDGP performed notably better than our baseline model, our results do not permit us to make a definite statement whether our proposed framework provides an enhanced surrogate model for rover wheel locomotion. There remains room for improvement, and both methods should be assessed under more optimal conditions.

4 Conclusions

4.1 Summary

In this thesis, we explored various methods that address uncertainty in the inputs of GP regression models, using some as the basis for developing input-uncertainty-aware extensions to two prominent multi-fidelity GP models. We focused particularly on scenarios where the inputs are noisy signals, which are smoothed during preprocessing to reduce the noise. In these cases, we investigated whether input-uncertainty-aware GP models can outperform standard single- and multi-fidelity GP models in terms of predictive performance. We approached this by treating the true inputs as randomly distributed around the smoothed signal. The results from our synthetic examples in both the single- and the multi-fidelity frameworks suggest that treating the inputs as uncertain generally leads to better predictive performance. We further tested our best-performing multi-fidelity model on the task of modeling rover wheel locomotion. Although our setup was likely not optimal, our results indicate that our model may offer valuable improvements in practical applications.

We considered three different approaches to incorporating uncertain inputs into single-fidelity GP regression models. The first approach involved taking the expectation of a squared exponential covariance function to obtain a GP that approximately incorporates the input distributions. The second approach comprised two methods that utilize Taylor approximations to derive input-uncertainty-aware GPs. Lastly, we examined three further methods that utilize different variational techniques. These encompass the well-known Bayesian GPLVM, a method that infers inputs using MAP estimation, and a stochastic EM method that samples from the posterior distribution over the inputs using an MCMC sampler. The multi-fidelity segment of our work focused on the NARGP and the MFDGP. These multi-fidelity GP models served as baselines, and we proposed extensions to them to account for uncertain inputs. Specifically, we extended the NARGP by stacking GP regression models using an adapted expected covariance function, and we modified the MFDGP by introducing a variational posterior over the inputs.

In our experiments, we employed a Bayesian smoother to jointly smooth noisy input signals and estimate the distributions of the true latent inputs. Consequently, we maintained a Gaussian assumption of the input distributions throughout this entire

study. While most of the methods we considered assumed the input distribution was known, both Taylor approximation-based methods learned the distribution without relying on this prior knowledge. Our single-fidelity experiments indicated superior predictive performance when an MAP estimate of the inputs was computed using a specific training procedure that prevents overfitting. Additionally, we achieved notable results by employing an expected covariance function. In the multi-fidelity setting, our MFDGP variant achieved the best performance, while our NARGP variant could also produce more stable predictions than the baseline models in the presence of uncertain inputs.

4.2 Discussion

In addition to quantitatively showing superior predictive performance of input-uncertainty-aware GP regression models in the presence of noise in input signals, our work also involved a comparative study of these methods using a synthetic example. While most single-fidelity models were validated in their respective original papers, the authors generally considered different examples, complicating the specific choice of a model for applications with uncertain inputs. The models varied in whether they assume the input distribution as known, yet, all are applicable for general regression tasks. We aimed to conduct an experiment that highlights the strengths and shortcomings of individual models, allowing researches to make informed choices among them. For instance, in scenarios in which only some inputs are uncertain or in particularly safety-critical applications, it may be desirable to employ a model that produces wider confidence intervals around uncertain data, rather than one with narrower confidence intervals yet a better fit to the ground truth.
Moreover, our work has explored how to incorporate input uncertainty in multi-fidelity GP regression models, providing two novel input-uncertainty-aware models. To the best of our knowledge, no such extensions of the NARGP or the MFDGP currently exist. Our models may be readily applied in practice when researchers know that the inputs have uncertain values with a known Gaussian distribution.
Nonetheless, we mainly compared the models with respect to their ability to recover the underlying ground truth. Other important factors, such as runtime complexity and memory usage, should also be considered, though. For instance, while the Bayesian GPLVM was outperformed by most of the input-uncertainty-aware models in our synthetic examples, its inducing variable framework makes it suitable for larger scale applications. Conversely, it requires the parametrization of the variational posterior over each individual inputs. Similarly, our extension of the MFDGP parametrizes all inputs and needed demanded more optimization steps to converge than our simpler

extension of the NARGP.
Furthermore, we upheld the assumption that the inputs are normally distributed. In applications where this is not the case, the predictive performance of individual models might be significantly compromised. We also did not directly assess the performance of the methods for predicting at uncertain inputs. Although this was not necessary to compare the predictive performance of models trained with uncertain inputs, more realistic experiments should include testing on uncertain inputs, as the test and training datasets are typically derived from the same source.

4.3 Future Work

A primary objective for future work regarding our methods should be to compare our input-uncertainty-aware MFDGP with a standard MFDGP in the context of rover wheel locomotion, ensuring both models achieve their optimal performance. The findings of our real-world data experiments do not provide a clear indication of whether our model is appropriate for this application, given that the baseline method is known achieve better results. Apart from that, future work on our input-uncertainty-aware MFDGP might feature reducing the number of parameters used for the variational posteriors over the inputs. Possible approaches include those suggested by Damianou, Titsias, et al. [57], who propose fixing the means of the variational posteriors to the means of the prior, or by Villacampa-Calvo et al. [39], who use a deep neural network to parametrize the variational posteriors.
Furthermore, our adaptation of the NARGP only considered the use of the expected covariance function. Other approaches are possible, as long as the input noise is allowed to vary across inputs. While this applies to all models except the Taylor approximation-based ones, they might also be adapted for known, varying input distributions. Furthermore, stacking other models requires a method for predicting at uncertain inputs, which is offered for general GP regression tasks by Girard et al. [48]. Additionally, it is essential to further investigate whether the newly considered single-fidelity model remains applicable when the structured covariance function from the NARGP is used.
Another line of future work could build upon the suggestions by Lee et al. [71], which were also adopted by Ravi, Fediukov, et al. [6]. Their approach incorporates gradient information or evaluations at slightly shifted locations from low-fidelity model in the high-fidelity surrogate. Although we did not address these approaches, combining them with our input-uncertainty-aware GP models could yield further interesting results.

Bibliography

[1] H. Habehh and S. Gohel. "Machine learning in healthcare." In: *Current genomics* 22.4 (2021), p. 291.

[2] V. Linardos, M. Drakaki, P. Tzionas, and Y. L. Karnavas. "Machine learning in disaster management: recent developments in methods and applications." In: *Machine Learning and Knowledge Extraction* 4.2 (2022).

[3] A. J. R. Lopez-Arreguin and S. Montenegro. "Machine learning in planetary rovers: A survey of learning versus classical estimation methods in terramechanics for in situ exploration." In: *Journal of Terramechanics* 97 (2021), pp. 1–17.

[4] V. Fediukov, F. Dietrich, and F. Buse. "Multi-fidelity machine learning modeling for wheel locomotion." In: *11th Asia-Pacific Regional Conference of the International society for terrain-vehicle systems, ISTVS 2022*. ISTVS. 2022.

[5] J. L. Callas. *Mars Exploration Rover Spirit end of mission report*. Tech. rep. 2015.

[6] K. Ravi, V. Fediukov, F. Dietrich, T. Neckel, F. Buse, M. G. Bergmann, and H.-J. Bungartz. "Multi-fidelity Gaussian process surrogate modeling for regression problems in physics." In: *Machine Learning: Science and Technology* (2024).

[7] Y. Guo and R. J. Little. "Regression analysis with covariates that have heteroscedastic measurement error." In: *Statistics in medicine* 30.18 (2011), pp. 2278–2294.

[8] C. E. Rasmussen and C. K. Williams. *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006.

[9] B. Abraham and J. Ledolter. *Introduction to regression modeling*. Thomson Brooks/-Cole, 2006, pp. 17–21.

[10] J. P. Buonaccorsi. *Measurement Error: Models, Methods, and Applications*. Chapman and Hall/CRC, Mar. 2010, p. 6. ISBN: 9780429150357. DOI: 10.1201/9781420066586.

[11] P. Dellaportas and D. A. Stephens. "Bayesian analysis of errors-in-variables regression models." In: *Biometrics* (1995), pp. 1085–1095.

[12] J. Berkson. "Are there two regressions?" In: *Journal of the american statistical association* 45.250 (1950), pp. 164–180.

[13] M. Chau, R. Cheng, B. Kao, et al. "Uncertain data mining: A new research direction." In: *Proceedings of the Workshop on the Sciences of the Artificial, Hualien, Taiwan*. Citeseer. 2005, pp. 199–204.

[14] S. Tsang, B. Kao, K. Y. Yip, W.-S. Ho, and S. D. Lee. "Decision trees for uncertain data." In: *IEEE transactions on knowledge and data engineering* 23.1 (2009), pp. 64–78.

[15] P. Gravel, G. Beaudoin, and J. A. De Guise. "A method for modeling noise in medical images." In: *IEEE Transactions on medical imaging* 23.10 (2004), pp. 1221–1232.

[16] H. E. Rauch, F. Tung, and C. T. Striebel. "Maximum likelihood estimates of linear dynamic systems." In: *AIAA journal* 3.8 (1965), pp. 1445–1450.

[17] S. Särkkä. *Bayesian Filtering and Smoothing*. Vol. 3. Cambridge University Press, Sept. 2013. ISBN: 9781107619289. DOI: 10.1017/cbo9781139344203.

[18] R. E. Kalman. "A new approach to linear filtering and prediction problems." In: (1960).

[19] R. G. Brown and P. Y. Hwang. *Introduction to random signals and applied Kalman filtering: with MATLAB exercises and solutions*. Third. 1997, p. 41.

[20] M. P. Deisenroth. *Efficient reinforcement learning using Gaussian processes*. Vol. 9. KIT Scientific Publishing, 2010.

[21] P. Perdikaris, M. Raissi, A. Damianou, N. D. Lawrence, and G. E. Karniadakis. "Nonlinear information fusion algorithms for data-efficient multi-fidelity modelling." In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 473.2198 (2017), p. 20160751.

[22] K. Cutajar, M. Pullin, A. Damianou, N. Lawrence, and J. González. "Deep gaussian processes for multi-fidelity modeling." In: *arXiv preprint arXiv:1903.07320* (2019).

[23] M. Giselle Fernández-Godino. "Review of multi-fidelity models." In: *Advances in Computational Science and Engineering* 1.4 (2023), pp. 351–400. ISSN: 2837-1739. DOI: 10.3934/acse.2023015.

[24] M. C. Kennedy and A. O'Hagan. "Predicting the output from a complex computer code when fast approximations are available." In: *Biometrika* 87.1 (2000), pp. 1–13.

[25] A. O'Hagan. "A Markov property for covariance structures." In: *Statistics Research Report* 98.13 (1998), p. 510.

[26] L. Le Gratiet and J. Garnier. "Recursive co-kriging model for design of computer experiments with multiple levels of fidelity." In: *International Journal for Uncertainty Quantification* 4.5 (2014).

[27] A. Damianou and N. D. Lawrence. "Deep gaussian processes." In: *Artificial intelligence and statistics*. PMLR. 2013, pp. 207–215.

[28] A. Ganguly and S. W. Earp. "An introduction to variational inference." In: *arXiv preprint arXiv:2108.13083* (2021).

[29] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. "Variational inference: A review for statisticians." In: *Journal of the American statistical Association* 112.518 (2017), pp. 859–877.

[30] J. Quiñonero-Candela and C. E. Rasmussen. "A unifying view of sparse approximate Gaussian process regression." In: *The Journal of Machine Learning Research* 6 (2005), pp. 1939–1959.

[31] E. Snelson and Z. Ghahramani. "Sparse Gaussian processes using pseudo-inputs." In: *Advances in neural information processing systems* 18 (2005).

[32] M. Titsias. "Variational learning of inducing variables in sparse Gaussian processes." In: *Artificial intelligence and statistics*. PMLR. 2009, pp. 567–574.

[33] M. Titsias and N. D. Lawrence. "Bayesian Gaussian process latent variable model." In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. 2010, pp. 844–851.

[34] Y. Bengio. *Learning Deep Architectures for AI*. 2009.

[35] H. Salimbeni and M. Deisenroth. "Doubly stochastic variational inference for deep Gaussian processes." In: *Advances in neural information processing systems* 30 (2017).

[36] A. Daemi, Y. Alipouri, and B. Huang. "Identification of robust Gaussian Process Regression with noisy input using EM algorithm." In: *Chemometrics and Intelligent Laboratory Systems* 191 (2019), pp. 1–11.

[37] C. Tran, V. Pavlovic, and R. Kopp. *Gaussian Process for Noisy Inputs with Ordering Constraints*. 2015. eprint: `arXiv:1507.00052`.

[38] M. Jadaliha, Y. Xu, J. Choi, N. S. Johnson, and W. Li. "Gaussian process regression for sensor networks under localization uncertainty." In: *IEEE Transactions on Signal Processing* 61.2 (2012), pp. 223–237.

[39] C. Villacampa-Calvo, B. Zaldıévar, E. C. Garrido-Merchán, and D. Hernández-Lobato. "Multi-class gaussian process classification with noisy inputs." In: *Journal of Machine Learning Research* 22.36 (2021), pp. 1–52.

[40] A. Mchutchon and C. Rasmussen. "Gaussian Process Training with Input Noise." In: *Advances in Neural Information Processing Systems*. Ed. by J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger. Vol. 24. Curran Associates, Inc., 2011.

[41] K. M. Ryan, J. Kristensen, Y. Ling, S. Ghosh, I. Asher, and L. Wang. "A Gaussian Process Modeling Approach for Fast Robust Design With Uncertain Inputs." In: *Volume 7A: Structures and Dynamics*. GT2018. American Society of Mechanical Engineers, June 2018. DOI: 10.1115/gt2018-77007.

[42] E. Bodin, M. Kaiser, I. Kazlauskaite, Z. Dai, N. Campbell, and C. H. Ek. "Modulating Surrogates for Bayesian Optimization." In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by H. D. III and A. Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, 13–18 Jul 2020, pp. 970–979.

[43] P. Z. G. Qian and C. F. J. Wu. "Bayesian Hierarchical Modeling for Integrating Low-Accuracy and High-Accuracy Experiments." In: *Technometrics* 50.2 (May 2008), pp. 192–204. ISSN: 1537-2723. DOI: 10.1198/004017008000000082.

[44] C. C. Fischer, R. V. Grandhi, and P. S. Beran. "Bayesian-Enhanced Low-Fidelity Correction Approach to Multifidelity Aerospace Design." In: *AIAA Journal* 56.8 (Aug. 2018), pp. 3295–3306. ISSN: 1533-385X. DOI: 10.2514/1.j056529.

[45] M. Raissi and G. Karniadakis. *Deep Multi-fidelity Gaussian Processes*. 2016. eprint: arXiv:1604.07484.

[46] J. Quiñonero-Candela and S. T. Roweis. *Data Imputation and Robust Training with Gaussian Processes*. http://cogsys.imm.dtu.dk/staff/jqc/pub/nips2003revised.pdf. 2003.

[47] P. Dallaire, C. Besse, and B. Chaib-draa. "An approximate inference with Gaussian process to latent functions from uncertain data." In: *Neurocomputing* 74 (2011), pp. 1945–1955.

[48] A. Girard. "Approximate Methods for Propagation of Uncertainty with Gaussian Process." PhD thesis. University of Glasgow, 2004.

[49] M. W. Seeger. "Bayesian Gaussian process models : PAC-Bayesian generalisation error bounds and sparse approximations." PhD thesis. University of Edinburgh, 2003.

[50] P. Dallaire, C. Besse, and B. Chaib-draa. "Learning Gaussian Process Models from Uncertain Data." In: *International Conference on Neural Information Processing*. 2009.

[51] A. Girard and R. Murray-Smith. *Learning a Gaussian Process Model with Uncertain Inputs*. https://www.dcs.gla.ac.uk/~rod/publications/GirMur03-tr-144.pdf. July 2003.

[52] E. Solak, R. Murray-Smith, W. Leithead, D. Leith, and C. Rasmussen. "Derivative observations in Gaussian process models of dynamic systems." In: *Advances in neural information processing systems* 15 (2002).

[53] A. Girard, C. E. Rasmussen, J. Q. Candela, and R. Murray-Smith. "Gaussian Process Priors with Uncertain Inputs - Application to Multiple-Step Ahead Time Series Forecasting." In: *Neural Information Processing Systems*. 2002.

[54] J. Quiñonero-Candela. "Learning with Uncertainty: Gaussian Processes and Relevance Vector Machines." PhD thesis. Technical University of Denmark, 2004.

[55] S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. "Hybrid monte carlo." In: *Physics letters B* 195.2 (1987), pp. 216–222.

[56] K. M. Hanson. "Markov Chain Monte Carlo posterior sampling with the Hamiltonian method." In: *Medical Imaging 2001: Image Processing*. Vol. 4322. SPIE. 2001, pp. 456–467.

[57] A. C. Damianou, M. K. Titsias, and N. D. Lawrence. *Variational inference for uncertainty on the inputs of gaussian process models*. 2014. eprint: `arXivpreprintarXiv:1409.2287`.

[58] J. Duchi. *Derivations for linear algebra and optimization*. `http://ai.stanford.edu/~jduchi/projects/general_notes.pdf`.

[59] A. G. d. G. Matthews, M. van der Wilk, T. Nickson, K. Fujii, A. Boukouvalas, P. León-Villagrá, Z. Ghahramani, and J. Hensman. "GPflow: A Gaussian process library using TensorFlow." In: *Journal of Machine Learning Research* 18.40 (Apr. 2017), pp. 1–6.

[60] D. Cuckworth et al. *pykalman*. `https://github.com/pykalman/pykalman`. 2012.

[61] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu. "A limited memory algorithm for bound constrained optimization." In: *SIAM Journal on scientific computing* 16.5 (1995), pp. 1190–1208.

[62] P. Virtanen, R. Gommers, T. E. Oliphant, et al. "SciPy 1.0: fundamental algorithms for scientific computing in Python." In: *Nature Methods* 17.3 (Feb. 2020), pp. 261–272. ISSN: 1548-7105. DOI: `10.1038/s41592-019-0686-2`.

[63] M. L. Rizzo. *Statistical computing with R*. Chapman and Hall/CRC, 2019, pp. 75–83.

[64] P. Perdikaris. *Multi-fidelity modeling using Gaussian processes and nonlinear autoregressive schemes*. `https://github.com/paraklas/NARGP`. 2016.

[65] M. Pullin. *Deep Gaussian Processes for Multi-fidelity Modelling*. `https://github.com/EmuKit/emukit/tree/main/emukit/examples/multi_fidelity_dgp`. 2019.

[66] D. P. Kingma and J. Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[67] M. Lyasko. "Slip sinkage effect in soil–vehicle mechanics." In: *Journal of Terramechanics* 47.1 (2010), pp. 21–31.

[68] L. Ding, Z. Deng, H. Gao, K. Nagatani, and K. Yoshida. "Planetary rovers' wheel–soil interaction mechanics: new challenges and applications for wheeled mobile robots." In: *Intelligent Service Robotics* 4 (2011), pp. 17–38.

[69] F. Buse. "Development and Validation of a Deformable Soft Soil Contact Model for Dynamic Rover Simulations." PhD thesis. Tohoku University, 2022.

[70] F. Buse, T. Bellmann, R. Lichtenheldt, and R. Krenn. "The DLR Terramechanics Robotics Locomotion Lab." In: *International Symposium on Artificial Intelligence, Robotics and Automation in Space*. June 2018.

[71] S. Lee, F. Dietrich, G. E. Karniadakis, and I. G. Kevrekidis. "Linking Gaussian process regression with data-driven manifold embeddings for nonlinear data fusion." In: *Interface Focus* 9.3 (Apr. 2019), p. 20180083. ISSN: 2042-8901. DOI: 10.1098/rsfs.2018.0083.