THE UNIVERSITY *of* EDINBURGH

Gaussian surrogate models for Bayesian inverse problems

Tianming Bai

Doctor of Philosophy
University of Edinburgh
2024

Declaration

I declare that this thesis was composed by myself and that the work contained therein is my own, except where explicitly stated otherwise in the text.

(*Tianming Bai*)

To my grandmother

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my supervisors, Aretha Teckentrup and Kostas Zygalakis. Your patience, unwavering support, and invaluable guidance have been essential in the completion of this thesis. Thank you for sharing countless opportunities and always being there to help me. Your expertise and encouragement have been truly inspiring.

I am also deeply grateful to my family. To my grandparents, my parents and my aunt, your love and belief in me have been a constant source of motivation throughout this journey. I am incredibly fortunate to have such a loving and supportive family.

I would like to thank my PhD mates and friends from Teckentroop and MI2G Group. The moments we have shared have been some of the most memorable experiences of my PhD journey.

Lastly, I want to express my heartfelt thanks to my partner Jiaao. Your love and encouragement have been my greatest strength.

Lay Summary

Inverse problems are ubiquitous across a number of scientific fields, including geophysics, image processing and machine learning to name a few. They arise in situations where certain quantities of interest cannot be directly measured and must be inferred from observable effects. Solving inverse problems using Bayesian methods requires numerous simulations, which are often computationally expensive. To address this, one can use computationally cheap surrogate models, such as Gaussian process emulators, to approximate the behaviour of complex mathematical models.

A key challenge in using Gaussian process emulators is to maintain accuracy when only a limited number of simulations are available due to computational constraints. We introduce a method that integrates the physics model into the Gaussian emulator, thus enhancing approximation accuracy. This method is extended to solve Bayesian inverse problems by approximating the probability distribution of unknown parameters, greatly reducing computational cost.

Abstract

Solving Bayesian inverse problems using Markov Chain Monte Carlo (MCMC) methods poses significant computational challenges due to the extensive numerical simulations required for each sample. To address this issue, surrogate models are often employed to approximate the complex models, thereby reducing computational costs.

This thesis focuses on the use of Gaussian surrogate models for Bayesian inverse problems associated with linear partial differential equations, particularly in scenarios when only limited training data are available. To enhance the accuracy and robustness of prediction without requiring additional observational data, we investigate the physics-informed Gaussian process regression (PI-GPR) method which provides a flexible framework for integrating physical information into the Gaussian process, and extend the method to construct different approximate posteriors for solving the Bayesian inverse problems. Benefiting from the nature of Gaussian process regression as a statistical model, the error of approximation can be quantified and integrated into the approximation of posteriors. Meanwhile, the gradient of the approximate posteriors based on Gaussian surrogate models can be analytically computed, enabling the use of gradient-based MCMC samplers like the Metropolis-adjusted Langevin algorithm (MALA) for efficient sampling. Finally, the approximate posterior can be used in the delayed-acceptance Metropolis-Hastings sampling algorithm, which helps reject unlikely candidates with a much lower cost and hence significantly reduces the overall computational cost.

Contents

Chapter 1

Introduction

1.1 Inverse problems and Bayesian perspective

In scientific research, not all quantities of interest can be directly observed or measured. For example, while we can directly measure the size of most objects in our lives, certain intrinsic properties such as their centre of mass, must be inferred indirectly through their effects like gravitational forces. This necessity gives rise to a class of problems known as "inverse problems", which aims to deduce unknown input parameters of a model from data of observable outcomes. Such problems are ubiquitous across various fields, including geophysics [63, 94, 95], image processing [9, 73] and machine learning [4, 77], to name a few.

Typical inverse problems of interest are ill-posed (in the sense of Hadamard [39]). In particular, their solution might not exist, not be unique, or may exhibit sensitive dependence on inputs. There are several factors that contribute to their ill-posedness. First, the forward model, which maps input parameters to observable outcomes, is often non-linear. This non-linearity means the inverse problems may have multiple solutions or that the solution may not change proportionally to changes in input parameters. Meanwhile, models used to describe the processes underlying inverse problems might not perfectly capture the actual phenomena, leading to discrepancies between model outputs and actual observations. In addition, the observational data are often partially observed and corrupted by noise. For example, the acquirement of data might be computationally intensive and

time-consuming, with noise coming from measurement processes. Consequently, the incomplete nature and noise of the observational data might introduce large uncertainty into the problem and lead to significant differences in inference.

In this thesis, we restrict our attention to a finite-dimensional setting for inverse problems, which can be formulated mathematically as

$$\mathbf{y} = \mathcal{G}(\boldsymbol{\theta}) + \eta, \tag{1.1}$$

where

- $\mathbf{y} \in \mathbb{R}^{d_{\mathbf{y}}}$ is the observation.
- $\boldsymbol{\theta} \in \mathcal{T} \subseteq \mathbb{R}^{d_{\boldsymbol{\theta}}}$ is the unknown parameter to be inferred.
- $\mathcal{G} : \mathcal{T} \to \mathbb{R}^{d_{\mathbf{y}}}$ is the *parameter-to-observation map*
- $\eta \in \mathbb{R}^{d_{\mathbf{y}}}$ is an additive noise term.

The modelling assumptions for the noise term can vary across different applications. Here, we assume that the noise affects the output additively.

We now present an example of an inverse problem in geophysics in the context of subsurface flow. This application is crucial for scenarios such as risk analysis for radioactive waste disposal and oil reservoir management, where understanding the fluid flow through porous media is essential. The governing equation for this scenario, derived from Darcy's law [26] coupled with an incompressibility condition, is represented as:

$$-\nabla \cdot (k(\mathbf{x}, \boldsymbol{\theta}) \nabla u(\mathbf{x}, \boldsymbol{\theta})) = f(\mathbf{x}), \tag{1.2}$$

where $k(\mathbf{x}, \boldsymbol{\theta})$ is a positive scalar representing the hydraulic conductivity, $u(\mathbf{x}, \boldsymbol{\theta})$ denotes the water pressure field and $f(\mathbf{x})$ is a source function. In addition, $\boldsymbol{\theta}$ denotes the parameter used to describe $k(\mathbf{x}, \boldsymbol{\theta})$. Typically, $k(\mathbf{x}, \boldsymbol{\theta})$ is described by a finite series expansion and hence $\boldsymbol{\theta}$ is finite-dimensional.

Equation (1.2) models the fluid movement through a medium where the hydraulic properties vary spatially. Properties such as hydraulic conductivity cannot

be measured directly and must be inferred from observed data (pressure measurements). In this case, the forward map $\mathcal{G}$ mapping from the parameter $\boldsymbol{\theta}$ to the solution u is nonlinear. An additional complication when trying to estimate $\boldsymbol{\theta}$ from data comes from the data acquisition process. Obtaining data related to the value of pressure in various spatial locations requires the construction of wells. This example illustrates the inherent complexities and uncertainties of the inverse problems discussed earlier, emphasizing the importance of uncertainty quantification in addressing real-world challenges. In the following chapters, it will serve as the primary example throughout our numerical experiments.

The two predominant approaches for solving inverse problems are the variational and the statistical approach [46]. In the variational approach, one writes down an optimisation problem containing a data-misfit function and possibly a regularisation term. In particular, one example of such an optimisation problem is

$$\arg\min_{\boldsymbol{\theta}\in\mathcal{T}} \left(\frac{1}{2}\|\mathbf{y} - \mathcal{G}(\boldsymbol{\theta})\|_a^2 + \frac{1}{2}\|\boldsymbol{\theta}\|_b^2 \right), \tag{1.3}$$

where $\|\cdot\|_a$ and $\|\cdot\|_b$ are norms defined in their corresponding vector spaces. The first term is the misfit function and the second is a regularisation function added to overcome the ill-posedness and prevent data over-fitting. The choices of the norms could be flexible, problem-dependent and possibly data dependent. For example, the norm induced by the quadratic Wasserstein metric is used in the misfit function to measure the difference between observed seismic data and simulated data in the study of full-waveform inversion in geophysics regarding its convexity and insensitivity to noise [93]. Furthermore, the total variational norm is used as a regularizer in image reconstruction to preserve edges while reducing noise [16].

On the other hand, in the statistical approach, one adopts a Bayesian viewpoint, treating all quantities of interest as random variables. For the typical inverse problem of interest (see (1.1)), Bayes' theorem can be written as

$$\pi(\boldsymbol{\theta}|\mathbf{y}) = \frac{1}{Z} L(\mathbf{y}|\boldsymbol{\theta})\pi_0(\boldsymbol{\theta}), \tag{1.4}$$

where Z is the normalization constant that ensures the posterior density integrates into one. In addition, the interpretation of the rest of the terms in (1.4) is

- *Prior* $\pi_0(\boldsymbol{\theta})$: A probability distribution for our unknown parameter $\boldsymbol{\theta}$ that reflects our belief before taking any observations into account.

- *Likelihood* $L(\mathbf{y}|\boldsymbol{\theta})$: The *likelihood* function describes how likely the observation is obtained by a certain choice of $\boldsymbol{\theta}$. The precise form of the likelihood depends on the exact statistics of the noise term η.

- *Posterior* $\pi(\boldsymbol{\theta}|\mathbf{y})$: The *posterior* is an updated belief on the unknown parameter $\boldsymbol{\theta}$ given the observation $\mathbf{y}$.

A common assumption often made on the noise term is that it follows the Gaussian distribution. In this thesis, we assume that the noise term follows the Gaussian distribution with zero mean and diagonal covariance matrix, that is, $\eta \sim \mathcal{N}(0, \Gamma_\eta)$. Hence, from (1.1) we have $\mathbf{y}|\boldsymbol{\theta} \sim \mathcal{N}(\mathcal{G}(\boldsymbol{\theta}), \Gamma_\eta)$, so the *likelihood* is

$$L(\mathbf{y}|\boldsymbol{\theta}) \propto \exp\left(-\frac{1}{2}\|\mathcal{G}(\boldsymbol{\theta}) - \mathbf{y}\|^2_{\Gamma_\eta}\right) := \exp\left(-\Phi(\boldsymbol{\theta}, \mathbf{y})\right), \tag{1.5}$$

where the function $\Phi : \mathcal{T} \times \mathbb{R}^{d_\mathbf{y}} \to \mathbb{R}$ is called the *negative log-likelihood* or *potential* and $\|\mathbf{z}\|_{\Gamma_\eta} = \mathbf{z}^{\mathrm{T}}\Gamma_\eta^{-1}\mathbf{z}$ denotes the norm weighted by Γ_η^{-1}. In real applications, the assumption made on the noise term can be relaxed and adjusted to suit different situations. Meanwhile, the noise variance might be unknown and need to be estimated from data in real-world scenarios. In this thesis, in order to focus on the core aspects of the model without introducing additional noise estimation techniques, we make this assumption with a known variance.

The Bayesian approach provides a structured methodology to systematically incorporate prior information and quantifies uncertainty through the posterior distribution, offering the probabilities for all possible solutions. However, extracting information from a high-dimensional probability distribution poses significant challenges, as it generally lacks a closed-form expression and involves high-dimensional integration for the normalization constant. To address this issue, one

commonly used method is the maximum a-posteriori (MAP) estimation, that is to compute $\arg\max_{\boldsymbol{\theta}\in\mathcal{T}} \pi(\boldsymbol{\theta}|\mathbf{y})$. Since $\arg\max_{\boldsymbol{\theta}\in\mathcal{T}} \pi(\boldsymbol{\theta}|\mathbf{y}) = \arg\min_{\boldsymbol{\theta}\in\mathcal{T}} -\log \pi(\boldsymbol{\theta}|\mathbf{y})$, it aligns with equation (1.3) given that appropriate norms, priors and likelihoods are selected. This relation establishes a link between Bayesian and variational methods. A comprehensive exploration of the connection between optimisation methods and the Bayesian approach is discussed in [85]. When estimation of distribution properties or expectation of quantities of interest is required, sampling methods such as Markov Chain Monte Carlo are often employed, which will be introduced in Section 1.3.

1.2 Gaussian process for regression

A *stochastic process* is a collection of random variables $\{g(\boldsymbol{\theta}) : \boldsymbol{\theta} \in \mathcal{T}\}$, where $\mathcal{T}$ represents the index set, often corresponding to time, space, or any other parameter domain over which the process is defined. If every finite collection of the random variables follows a multivariate Gaussian distribution, then the process is called a Gaussian process. A Gaussian process can be written as

$$g(\boldsymbol{\theta}) \sim \mathrm{GP}(m(\boldsymbol{\theta}), k(\boldsymbol{\theta}, \boldsymbol{\theta}')),$$

where $m : \mathcal{T} \to \mathbb{R}$ is the mean function and $k(\boldsymbol{\theta}, \boldsymbol{\theta}') : \mathcal{T} \times \mathcal{T} \to \mathbb{R}$ is the covariance function. A Gaussian process is fully specified by its mean and covariance function [71]. The covariance function, also called a kernel, describes the similarity between different inputs. Common choices of the kernel include the squared exponential function

$$k(\boldsymbol{\theta}, \boldsymbol{\theta}') = \sigma^2 \exp\left(-\frac{\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|^2}{2l^2}\right), \tag{1.6}$$

and the Matérn covariance functions

$$k(\boldsymbol{\theta}, \boldsymbol{\theta}') = \frac{\sigma^2}{\Gamma(\nu)2^{\nu-1}} \left(\sqrt{2\nu}\frac{\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|}{l}\right)^{\nu} B_\nu \left(\sqrt{2\nu}\frac{\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|}{l}\right), \tag{1.7}$$

where $\|\cdot\|$ represents the Euclidean norm, B_ν is the modified Bessel function and $\Gamma(\nu)$ is the Gamma function [34]. σ and l are two hyperparameters commonly appear in covariance functions, where σ^2 is the variance that determines how far the function $g(\boldsymbol{\theta})$ can be away from the mean $m(\boldsymbol{\theta})$, and l is the length-scale which describes how rapidly the function value can change. ν in the Matérn kernels controls the differentiability of the function. When ν tends to infinity, the Matérn kernel tends to the squared exponential kernel. In Figure 1.1, we illustrate how the choice of hyperparameters and kernels affects the shape of samples. Compared to the top left plot (the squared exponential kernel, $l = 1$ and $\sigma = 1$), a smaller variance ($\sigma^2 = 0.04$) makes samples closer to the mean (top right); a shorter length-scale ($l = 0.2$) make samples more wiggly (bottom left); the Matérn kernel with hyperparameters $l = 1, \sigma^2 = 1$ and $\nu = \frac{5}{2}$ leads to less smooth samples (top left). In addition, both of the example covariance functions depend only on $(\boldsymbol{\theta} - \boldsymbol{\theta}')$, so they are invariant under translation and are defined as the *stationary* covariance function. Moreover, a variety of modelling assumptions can be made in covariance functions, such as symmetry, periodicity, and anisotropy. These assumptions give Gaussian processes strong flexibility to model the specific properties of real-world processes, enabling them to capture the underlying behaviour more accurately while maintaining interpretability. A detailed discussion of their properties can be found in [71].

The flexibility and interpretability of Gaussian processes make them highly suitable for solving regression problems, which are concerned with predicting continuous quantities based on given data. In a typical regression problem, we aim to find a function that fits a set of data points. We denote the set of data by $\mathbf{g}(\Theta) = \{g(\boldsymbol{\theta}_i)\}_{i=1}^N$ with the set of corresponding inputs $\Theta = \{\boldsymbol{\theta}_i\}_{i=1}^N$. Traditional parametric regression models such as linear regression, employ a fixed functional form to fit the available data, whose performance largely depends on whether the functional form can capture the dependencies in data. On the other hand, Gaussian process regression is a flexible non-parametric regression method, whose performance is more data-dependent. In Gaussian process regression, we assume the function $g : \mathcal{T} \to \mathbb{R}$ that fits data is a realisation from a Gaussian process prior $g_0(\boldsymbol{\theta}) \sim \mathrm{GP}(0, k(\boldsymbol{\theta}, \boldsymbol{\theta}'))$. It is common to assume that the prior mean is

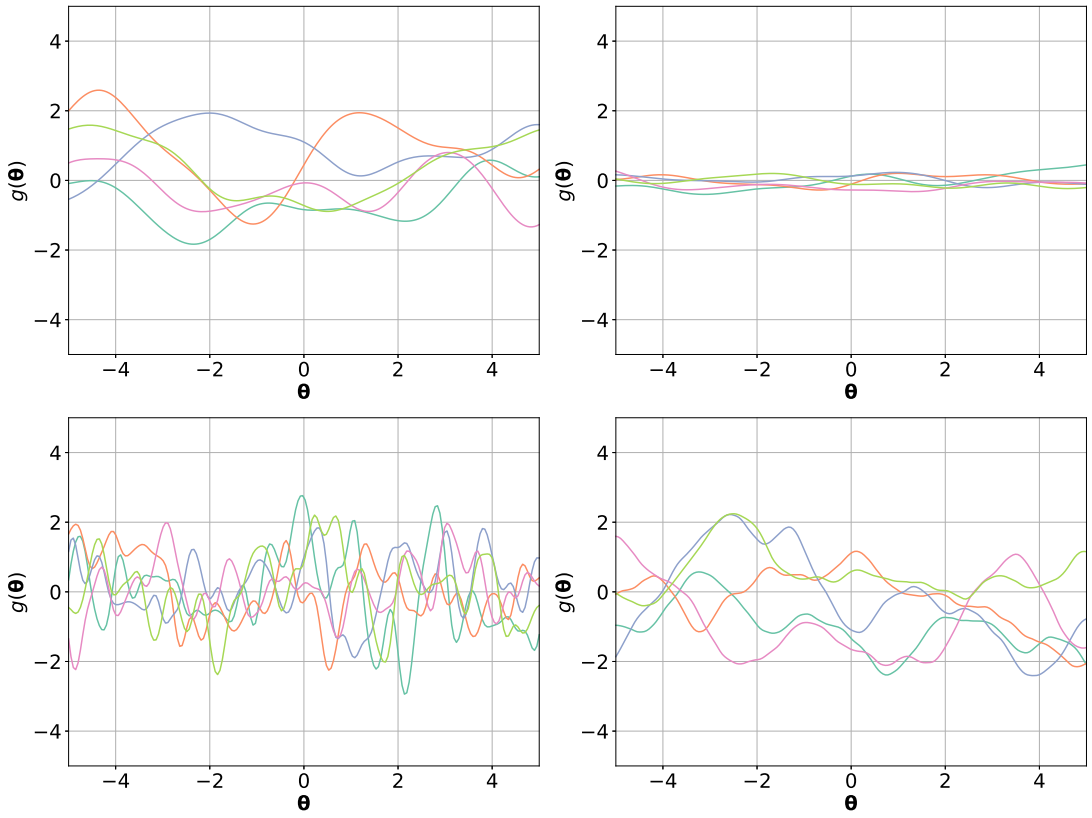

Figure 1.1: Samples from Gaussian processes with different covariance functions and hyperparameters: squared exponential with $\sigma = 1, l = 1$ (*top left plot*); squared exponential with $\sigma = 0.2, l = 1$ (*top right plot*); squared exponential with $\sigma = 1, l = 0.2$ (*bottom left plot*) and Matérn kernel with $\nu = 2, \sigma = 1, l = 1$ (*bottom right plot*).

zero, particularly when no other prior information is given. Then, given a set of points $\Theta^* = \{\boldsymbol{\theta}_i^*\}_{i=1}^{N^*}$ that we want to predict at, we can build a joint Gaussian distribution between them as

$$\begin{bmatrix} \mathbf{g}(\Theta^*) \\ \mathbf{g}(\Theta) \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K(\Theta^*, \Theta^*) & K(\Theta^*, \Theta) \\ K(\Theta, \Theta^*) & K(\Theta, \Theta) \end{bmatrix}\right),$$

where $K(\Theta^*, \Theta)$ and $K(\Theta, \Theta^*)^T$ are $N^* \times N$ matrices whose $(i,j)_{th}$ entries are $k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$ with $\boldsymbol{\theta}_i \in \Theta^*$ and $\boldsymbol{\theta}_j \in \Theta$, and $K(\Theta, \Theta)$ is $N \times N$ covariance matrix (also called the Gram matrix) whose $(i,j)_{th}$ entries are $k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$. By conditioning this joint Gaussian distribution on the observation points, we obtain the predictive Gaussian process

$$\begin{aligned} &\mathbf{g}(\Theta^*)|\mathbf{g}(\Theta) \sim \mathcal{N}\left(\mathbf{m}_N^g(\Theta^*), K_N(\Theta^*, \Theta^*)\right) \\ &\mathbf{m}_N^g(\Theta^*) = K(\Theta^*, \Theta)K(\Theta, \Theta)^{-1}\mathbf{g}(\Theta) \\ &K_N(\Theta^*, \Theta^*) = K(\Theta^*, \Theta^*) - K(\Theta^*, \Theta)K(\Theta, \Theta)^{-1}K(\Theta, \Theta^*). \end{aligned} \tag{1.8}$$

If given noisy data $\mathbf{y} = \mathbf{g}(\Theta) + \xi$, where $\xi \sim \mathcal{N}(0, \Sigma)$ is the noise term, then the joint distribution becomes

$$\begin{bmatrix} \mathbf{g}(\Theta^*) \\ \mathbf{y} \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K(\Theta^*, \Theta^*) & K(\Theta^*, \Theta) \\ K(\Theta, \Theta^*) & K(\Theta, \Theta) + \Sigma \end{bmatrix}\right),$$

and the predictive Gaussian process is

$$\begin{aligned} &\mathbf{g}(\Theta^*)|\mathbf{y} \sim \mathcal{N}\left(\mathbf{m}_N^g(\Theta^*), K_N(\Theta^*, \Theta^*)\right) \\ &\mathbf{m}_N^g(\Theta^*) = K(\Theta^*, \Theta)(K(\Theta, \Theta) + \Sigma)^{-1}\mathbf{y} \\ &K_N(\Theta^*, \Theta^*) = K(\Theta^*, \Theta^*) - K(\Theta^*, \Theta)(K(\Theta, \Theta) + \Sigma)^{-1}K(\Theta, \Theta^*). \end{aligned} \tag{1.9}$$

To understand the non-parametric nature of the Gaussian process regression, we can view each row of the matrix $K(\Theta^*, \Theta)(K(\Theta, \Theta) + \Sigma)^{-1}$ in the predictive mean to be the parameter of the linear combination of the data $\mathbf{y}$. Hence, the prediction accuracy typically increases as the number of data points N increases, although there are exceptions, such as when the additional data points

are noisy. A theoretical investigation of the convergence property of Gaussian process regression can be found in [88]. In addition, once the data $\mathbf{y}$ is fixed, the vector $(K(\Theta,\Theta)+\Sigma)^{-1}\mathbf{y}$ can be pre-computed. Thus, when predicting at a single point $\boldsymbol{\theta}^*$, we can write the predictive mean as $\mathbf{m}_N^g(\boldsymbol{\theta}^*) = \sum_{i=1}^N \alpha_i k(\boldsymbol{\theta}^*, \boldsymbol{\theta}_i)$, where $\boldsymbol{\alpha} = (K(\Theta,\Theta)+\Sigma)^{-1}\mathbf{y}$ is the pre-computed weights. We see that the computation of predictions is simplified to evaluating the vector $\mathbf{k}(\boldsymbol{\theta}^*, \Theta)$ and computing the inner product between them. Therefore, the predictive mean of the conditional Gaussian process is computationally efficient for making repeated predictions on different points. The computation of the predictive covariance will be discussed later on. Along with these advantages, another significant benefit of Gaussian process regression is its inherent capability to quantify uncertainty in its predictions. This feature is particularly valuable in scenarios where understanding the confidence in model predictions is as crucial as the predictions themselves.

Due to the advantages discussed above, Gaussian process regression has remained a popular statistical model, particularly in emulating complex systems[47]. It is widely used in black box modelling when the model is over-complex, for example, in emulating human heart contraction [51]. Additionally, it can also be used to build physics-informed models for problems well-studied through partial differential equations [67]. Although Gaussian process regression is popular nowadays, it was not widely applied to practical problems in history due to the computational intensity of matrix operations. The computation of the predictive covariance involves the inversion of the matrix, whose costs increase cubically with the number of training points. The history of the Gaussian process dates back to the 1940s, beginning with the development of the Wiener-Kolmogorov predictor. This predictor was proposed independently by Wiener in 1942[90], Wold in 1938[91], and Kolmogorov in 1941[49]. In geophysics, these ideas were further refined into what is known as 'kriging', a statistical method developed from the work of Matheron (1962) based on Daniel Krige's research. The resurgence of interest in Gaussian process regression appears in the 1990s, with the development of modern computers and efficient numerical linear algebra techniques. Notably, Neal [57] demonstrated that the Bayesian neural network tends to Gaussian process as the number of hidden neurons approaching infinity. Rasmussen [70] showed

that Gaussian process regression outperforms several other non-parametric regression methods, such as k-nearest neighbours and multivariate adaptive regression splines, in a range of prediction tasks. These works demonstrated its potential in practical applications and established a theoretical foundation for future research. In upcoming chapters, we will explore how to use Gaussian process regression, particularly physical-informed Gaussian process regression, as an emulator for solving Bayesian inverse problems.

1.3 Markov chain Monte Carlo

Many quantities of interest are in the form of integrals. For example, in solving Bayesian inverse problems, such quantities often have the form $\int_{\mathcal{T}} f(\boldsymbol{\theta})\pi(\boldsymbol{\theta}|\mathbf{y})d\boldsymbol{\theta}$. In general, these integrals cannot be computed analytically and one needs to resort to numerical approximations. In low dimensional numerical integration, numerical quadrature is commonly used, which discretizes the domain into sub-intervals and calculates a weighted sum of function values at specific points within each sub-interval. However, the computational cost of the quadrature methods often increases exponentially with dimensions, which is a phenomenon known as the curse of dimensionality. On the other hand, another type of method called the Monte Carlo method, uses independent and identically distributed (i.i.d.) samples and the law of large numbers to estimate the integral. To estimate the integral with respect to a general target density $\pi(\boldsymbol{\theta})$, we need to generate a series of i.i.d. samples $\boldsymbol{\theta}_i$ from $\pi(\boldsymbol{\theta})$, then the integral is estimated by

$$\int_{\mathcal{T}} f(\boldsymbol{\theta})\pi(\boldsymbol{\theta})d\boldsymbol{\theta} \approx \frac{1}{N}\sum_{i=1}^{N} f(\boldsymbol{\theta}_i),$$

where N is the number of samples. When $N \to \infty$, we have $\frac{1}{N}\sum_{i=1}^{N} f(\boldsymbol{\theta}_i)$ converges almost surely (with probability equal to one) to $\mathbb{E}_{\pi}[f]$. In most applications of interest, $\pi(\boldsymbol{\theta})$ is known up to the normalization constant, so direct sampling from $\pi(\boldsymbol{\theta})$ is not possible.

Several types of methods are proposed to draw i.i.d. samples, such as rejection sampling [59, 15], which generates i.i.d. samples following the desired distribution,

and importance sampling [48] which uses i.i.d. samples drawn from a different distribution to approximate integrals. A third type of sampling method, Markov chain Monte Carlo (MCMC) [55, 27] methods, generates correlated samples with a Markov chain whose equilibrium distribution is the desired distribution. MCMC methods are particularly useful when it is difficult or computationally expensive to draw direct i.i.d. samples from the desired distribution, which is often the case in Bayesian inference and high-dimensional settings. In this report, we will focus on one class of MCMC methods, called the Metropolis-Hastings (MH) algorithm, which was first proposed by Nicholas Metropolis [55], and then generalised by W. K. Hastings [40].

Algorithm 1 Metropolis-Hastings algorithm

Require: Target distribution $\pi(\boldsymbol{\theta})$, proposal distribution $q(\boldsymbol{\theta}'|\boldsymbol{\theta})$, initial value $\boldsymbol{\theta}_0$, total number of iterations N

Ensure: Sequence of samples $\{\boldsymbol{\theta}_n\}_{n=0}^{N}$

Choose an initial value $\boldsymbol{\theta}_0$.

for $n = 0$ to $N - 1$ **do**

Generate a candidate $\boldsymbol{\theta}'$ from the proposal distribution $q(\cdot|\boldsymbol{\theta}_n)$.

Compute the acceptance probability

$$\alpha := \max\left(1, \frac{\pi(\boldsymbol{\theta}')q(\boldsymbol{\theta}_n|\boldsymbol{\theta}')}{\pi(\boldsymbol{\theta}_n)q(\boldsymbol{\theta}'|\boldsymbol{\theta}_n)}\right).$$

Sample $r \sim \text{Uniform}[0, 1]$.

if $r \leq \alpha$ **then**

$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}'$ ▷ acceptance

else

$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n$ ▷ rejection

end if

end for

The algorithm (see Algorithm 1) starts at an initial state $\boldsymbol{\theta}_0$, then at n_{th} state, a proposal distribution q proposes a transition from the current state $\boldsymbol{\theta}_n$ to a new state $\boldsymbol{\theta}'$, and then the new state will be accepted with a probability. The choice of proposal distribution q is very flexible. The simplest proposal does not take into account the properties of the target distribution and it is based on a random walk. In this case, q is the density related to a Gaussian distribution centred at the current state $\boldsymbol{\theta}_n$. In addition, this q is symmetric in the sense of $q(\boldsymbol{\theta}_1|\boldsymbol{\theta}_2) = q(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)$, so the computation of the acceptance probability simplifies

to $\alpha = \max\left(1, \frac{\pi(\boldsymbol{\theta}')}{\pi(\boldsymbol{\theta}_n)}\right)$. This accept-reject step avoids computing the normalization constant, which is often difficult in high dimensions. However, this does not imply the acquisition of samples is free, evaluation of unnormalised $\pi(\boldsymbol{\theta})$ could still be computationally expensive in practical problems. In addition, samples generated by the Markov chain are correlated, therefore are less effective when used for the estimation. An important aspect that characterises the effectiveness of this approach is the proposal variance. In particular, if it is taken to be very large this will lead to a number of rejections, while if it is taken too small this will lead to slow exploration. Thus tuning the proposal distribution is very important in terms of the algorithmic behaviour. Several studies have been done to find the best way of tuning the algorithm. In the case of random walk applied to targets of i.i.d components, the optimal acceptance rate was found to be 0.234 [32, 33]. Heuristically, this has been used as the desirable acceptance rate in the general case, which in turn is obtained by an adaptive proposal variance [3].

A suitable choice of the proposal function can greatly improve the sampling efficiency. However, a more efficient transition kernel often requires additional information from the target density $\pi(\boldsymbol{\theta})$. For example, the Metropolis-adjusted Langevin algorithm (MALA) needs to evaluate $\nabla \log(\pi(\boldsymbol{\theta}))$. In this thesis, the use of the Gaussian process for emulating the forward map in (1.5), allows us to calculate the gradient of the approximate posterior explicitly. We will leverage this advantage by using MALA to sample from the posterior in solving the Bayesian inverse problem. Additionally, Hamiltonian Monte Carlo (HMC) [58] is another powerful alternative, using Hamiltonian dynamics to explore the parameter space efficiently in high dimensions by leveraging both the gradient and momentum. However, it comes with the cost of more computational effort per iteration. While HMC could be a promising next step for improving sampling efficiency, in this work, we opt to use the simpler MALA algorithm for its ease of implementation.

1.4 Outline of thesis

The structure of this thesis is as follows: In Chapter 2 we mainly focus on building the Gaussian process emulator as a solver of linear differential equations. In Chapter 3 we use the Gaussian process emulator for solving Bayesian inverse problems with a partial differential equation background and propose an induced MALA sampling algorithm. The results presented in this chapter appeared in Statistics and Computing [7] and are joint work with Aretha L. Teckentrup and Konstantinos C. Zygalakis. This work constitutes the primary focus of my PhD research, for which I conducted the majority of the development, implementation, and analysis. Chapter 4 introduces a delayed acceptance Metropolis-Hastings sampling algorithm.

Chapter 2

Gaussian processes constrained by linear operators

2.1 Introduction

Differential equations describe how physical quantities change over time and space, which is crucial for modelling continuous systems in fields such as physics and engineering. Partial differential equations (PDEs) are particularly crucial in modelling phenomena where changes occur in more than one variable and direction, such as heat conduction, fluid flow, and electromagnetic fields. The ability to understand and solve these equations is important for predicting and analyzing complex systems in the natural and social sciences. However, getting closed-form solutions for general PDEs is typically impossible, therefore numerical methods are needed. The main idea behind numerical methods for PDEs is that the continuous equation is discretised. A series of works have been proposed in the early 20th century, including the Ritz method in 1909 and the Galerkin method in 1915. These foundational methods paved the way for advanced techniques like the Finite Element Method (FEM) and spectral methods, significantly expanding the application of PDEs across disciplines like applied mathematics, engineering, and physics.

The error of numerical methods is systematically studied in numerical analysis. Theoretical results from numerical analysis provide the bounds on approxima-

tion error as a function of discretization parameters. Typically, the more accurate the approximation the higher the computational cost. High computational cost makes traditional numerical methods like FEM face challenges in contemporary applications, particularly in scenarios involving numerous evaluations of the system, such as Bayesian inverse problems. Therefore, practitioners often resort to coarse discretizations in real-world applications, which may introduce significant approximation errors, thus uncertainty quantification becomes important. Probabilistic numerics has emerged as a pivotal field that integrates principles from probability theory with numerical analysis to quantify uncertainty in numerical approximations. Unlike traditional numerical methods that provide deterministic solutions, probabilistic numerics treats numerical tasks—such as solving PDEs as inference problems, thereby enabling the incorporation of prior knowledge and assessing uncertainty in the results [43]. For example, Statistical Finite Element Methods (StatFEM) [37, 62] introduces probabilistic interpretations to the finite element framework. and Latent Force Models [2], which combine Gaussian processes with differential equations to model complex, dynamic systems. These approaches not only enhance the robustness of numerical solutions but also facilitate more informed decision-making by providing uncertainty estimates alongside predictions. The advent of probabilistic numerics complements machine learning-based methods like PI-GPR, as both seek to leverage probabilistic frameworks to improve the accuracy and reliability of solutions to complex physical systems.

On the other hand, in the past two decades, with the development of machine learning (ML) techniques, ML-based emulators have been used as alternatives to traditional numerical methods in response to the computational limitations in specific contexts. Once trained, ML emulators can provide predictions rapidly, often in orders of magnitude less time when compared with traditional numerical methods for solving PDEs. Moreover, ML methods have a strong capability in approximation. They can directly incorporate experimental or observational data, learning complex relationships from inputs without the need for analytical forms. This can be particularly useful for systems where the physics is not fully understood or is too complicated to model accurately. Two main approaches are currently popular in emulating numerical PDE solvers, including Physics-

informed Gaussian process regression (PI-GPR) [67, 19] and Physics-informed neural networks (PINNs) [66]. They are designed for the situation that only a limited volume of data is provided and purely data-driven models may generate physically inconsistent or implausible predictions. Both of the methods aim at incorporating physical laws, such as conservation laws and boundary conditions, directly into the models, improving the prediction accuracy and reliability. While PINNs are often highlighted for their flexibility and fast evaluation, it is less interpretable as a deep-learning-based method, compared to the kernel-based Gaussian process regression method. Another advantage of PI-GPR is its inherent capability in uncertainty quantification as a statistical model. The prediction error induced by coarse discretization can be quantified by the covariance of the Gaussian process.

PI-GPR takes advantage of the fact that the application of a bounded linear operator to a Gaussian process results in another Gaussian process. This allows for constructing joint and conditional Gaussian processes between different quantities in linear differential equations. The theoretical foundation for PI-GPR was laid early in the 20th century, with significant contributions in the study of Gaussian process differentiability and stochastic integration [1]. Related methods first emerged in the early 2000s, with the combination of observation of derivatives in the learning of dynamic systems [81], and in 2003, initial and boundary value problems in linear differential equations were solved [38]. Then, in 2016 and 2017, this approach attracted interest again and was applied to solve PDEs [19, 67], which led to numerous applications in many fields [25, 86], such as structural health monitoring [24, 23] and wave loading prediction [65].

In this chapter, we will first introduce the PI-GPR method, and illustrate its performance through several numerical experiments. In particular, we discuss different behaviour of PI-GPR when it is applied to: a PDE with boundary conditions, a PDE without boundary conditions and a PDE with boundary conditions and additional observation data. We then provide a way to understand PI-GPR from a linear system perspective. We show that PI-GPR provides a statistical model to the solutions when it is applied to a system of linear equations. We then extend the result to the infinite-dimensional case.

2.2 Physics-informed Gaussian process regression

Consider a system of equations

$$\mathcal{L}u(\mathbf{x}) = f(\mathbf{x}), \qquad \mathbf{x} \in D, \tag{2.1a}$$

$$\mathcal{B}u(\mathbf{x}) = g(\mathbf{x}), \qquad \mathbf{x} \in \partial D, \tag{2.1b}$$

posed on a domain $D \subseteq \mathbb{R}^{d_{\mathbf{x}}}$, where $\mathcal{L}$ denotes a bounded linear differential operator, $f : D \to \mathbb{R}$ is the source function and the linear operator $\mathcal{B}$ and function $g : \partial D \to \mathbb{R}$ incorporate boundary conditions. We assume the operators $\mathcal{L}, \mathcal{B}$ and functions f, g are known, and the target is to find the solution u. For clarity, we note that $\mathcal{L}$ is linear with respect to the function u. We first assume a Gaussian prior on u as

$$u(\mathbf{x}) \sim \mathrm{GP}(0, k(\mathbf{x}, \mathbf{x}')). \tag{2.2}$$

Using the property that the linear transformation of a Gaussian process with the squared exponential kernel or the Matérn kernel results in a Gaussian process [64], we therefore have the prior on f as $f(\mathbf{x}) \sim \mathrm{GP}(0, \mathcal{L}\mathcal{L}'k(\mathbf{x}, \mathbf{x}'))$, where $\mathcal{L}'$ is applied to the second variable $\mathbf{x}'$ of $k(\mathbf{x}, \mathbf{x}')$.

Given the point evaluations of f at a set of spatial points $X_f = \{\mathbf{x}_i\}_{i=1}^{N_f}$ denoted by $\mathbf{f}(X_f)$, the conditional Gaussian process is (see Corollary 2 in [64])

$$u(\mathbf{x})|\mathbf{f}(X_f) \sim \mathrm{GP}(m|_f(\mathbf{x}), k|_f(\mathbf{x}, \mathbf{x}')) \tag{2.3}$$

with the conditional mean and covariance function given by

$$m|_f(\mathbf{x}) = \mathcal{L}'\mathbf{k}(\mathbf{x}, X_f)^T \mathcal{L}\mathcal{L}' K(X_f, X_f)^+ \mathbf{f}(X_f),$$

and

$$k|_f(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{x}') - \mathcal{L}'\mathbf{k}(\mathbf{x}, X_f)^T \mathcal{L}\mathcal{L}' K(X_f, X_f)^+ \mathcal{L}\mathbf{k}(X_f, \mathbf{x}),$$

where the superscript "+" denotes the Moore-Penrose pseudo-inverse [28, 56]. Since applying bounded linear operators on kernels preserves only positive semi-

definiteness, the Moore-Penrose pseudo-inverse is used for the possibility that $\mathcal{L}\mathcal{L}'K(X_f, X_f)$ is singular [79]. The vectors $\mathcal{L}'\mathbf{k}(\mathbf{x}, X_f)^T$ and $\mathcal{L}\mathbf{k}(X_f, \mathbf{x})$ of size N_f have respective i_{th} entries $\mathcal{L}'k(\mathbf{x}, \mathbf{x}_i)$ and $\mathcal{L}k(\mathbf{x}_i, \mathbf{x})$. The matrix $\mathcal{L}\mathcal{L}'K(X_f, X_f)$ of size $N_f \times N_f$ has its $(i, j)_{th}$ entry $\mathcal{L}\mathcal{L}'k(\mathbf{x}_i, \mathbf{x}_j)$.

To further constrain the Gaussian process by the boundary conditions, similarly, we have the prior on g as $g(\mathbf{x}) \sim \mathrm{GP}(0, \mathcal{B}\mathcal{B}'k(\mathbf{x}, \mathbf{x}'))$. Let $\mathbf{g}(X_g)$ denote the point evaluations of g at a set of spatial points $X_g = \{\mathbf{x}_i\}_{i=1}^{N_g}$, we can incorporate the boundary conditions into the Gaussian process and the conditional Gaussian process is

$$
\begin{aligned}
&u(\mathbf{x})|\mathbf{g}(X_g), \mathbf{f}(X_f) \sim \mathrm{GP}(m|_{g,f}(\mathbf{x}), k|_{g,f}(\mathbf{x}, \mathbf{x}')), \\
&m|_{g,f}(\mathbf{x}) = \begin{bmatrix} \mathcal{B}'\mathbf{k}(\mathbf{x}, X_g) & \mathcal{L}'\mathbf{k}(\mathbf{x}, X_f) \end{bmatrix} K_{gf}^{+} \begin{bmatrix} \mathbf{g}(X_g) \\ \mathbf{f}(X_f) \end{bmatrix}, \\
&k|_{g,f}(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{x}') - \begin{bmatrix} \mathcal{B}'\mathbf{k}(\mathbf{x}, X_g) & \mathcal{L}'\mathbf{k}(\mathbf{x}, X_f) \end{bmatrix} K_{gf}^{+} \begin{bmatrix} \mathcal{B}\mathbf{k}(X_g, \mathbf{x}) \\ \mathcal{L}\mathbf{k}(X_f, \mathbf{x}) \end{bmatrix},
\end{aligned}
\tag{2.4}
$$

where

$$
K_{gf} = \begin{bmatrix} \mathcal{B}\mathcal{B}'K(X_g, X_g) & \mathcal{B}\mathcal{L}'K(X_g, X_f) \\ \mathcal{L}\mathcal{B}'K(X_f, X_g) & \mathcal{L}\mathcal{L}'K(X_f, X_f) \end{bmatrix}, \qquad K_{gf} \in \mathbb{R}^{(N_f+N_g)^2}.
$$

The computation of the covariance matrices in (2.4) involves applying the differential operators $\mathcal{B}$ and $\mathcal{L}$ to the covariance function $k(\cdot, \cdot)$ and subsequently evaluating the result at the designated points X_g and X_f. Here, in order to introduce the method analogous to a PDE solver, we only incorporate data from f into the Gaussian process. However, as a data-driven model, the above formula can be easily extended to incorporate direct observations of u into the prediction. In addition, the method is applicable to equations with any bounded linear operator [64]. When applied to equations with unbounded linear operators, certain conditions must be met [54]. Moreover, this method is also termed "Probabilistic Meshless Method" [19] and its predictive mean $m|_{g,f}(\mathbf{x})$ is the same as the numerical PDE solution given by the method of symmetric collocation [19, 29].

2.2.1 Numerical illustration

We now illustrate the method with simple numerical experiments. We use (1.2) as the PDE we want to solve and further simplify it to

$$\begin{aligned} -\frac{\mathrm{d}^2}{\mathrm{d}x^2}u(x) &= 10, \quad x \in (0,1), \\ u(0) &= 0, \quad u(1) = 0. \end{aligned} \tag{2.5}$$

In this case, this boundary value problem has an analytical solution

$$u(x) = -5x^2 + 5x. \tag{2.6}$$

We can hence compare the approximated solution given by PI-GPR with this solution.

We now use PI-GPR to approximate this solution. First, we assume a Gaussian process prior on u as in (2.2), with zero mean and the squared exponential covariance function. The hyperparameters of the covariance function, $\sigma = 1$ and $l = 0.2$, were chosen ad hoc. We illustrate the samples of the Gaussian process prior in Figure 2.1 (top left). We then constrain the prior by incorporating the boundary conditions. Since the equation has a Dirichlet boundary condition, that is, the operator $\mathcal{B}$ in (2.1b) is an identity operator, the calculation of the predictive Gaussian process is the same as in (1.8). After conditioning on the boundary points, the uncertainty at the ends of the domain is reduced to zero, and the samples always go through the boundary points (top right). We further constrain the prior by incorporating the PDE information with X_f being equally spaced points in $[0, 1]$ (excluding the boundary points). We see that the predictive mean gets closer to the solution as we increase N_f from 0 (top right) to 3 (bottom left). In the bottom right plot, when $N_f = 8$, the samples and the mean overlap with the solution.

In practical applications, it is possible that only partial information of the equation system is given. For example, boundary conditions could be unknown. In our previous example (2.5), the solution without incorporating the boundary

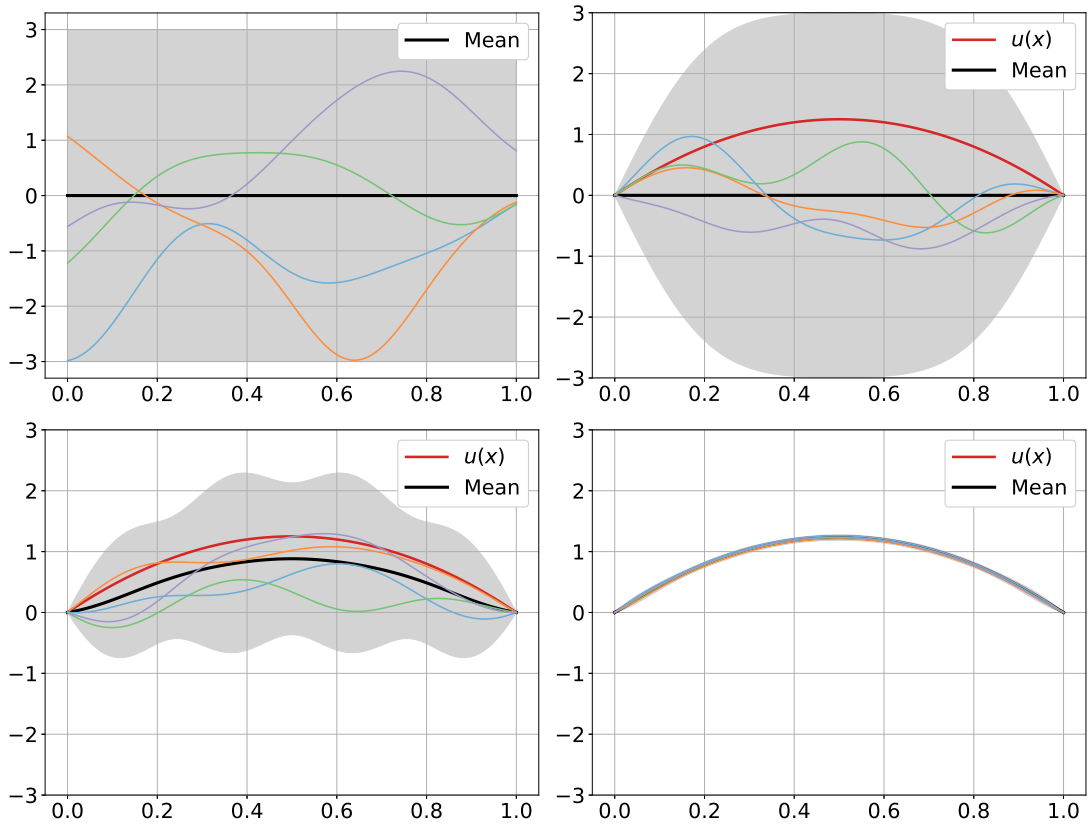

Figure 2.1: The mean and the samples of Gaussian processes: Prior (*top left plot*); Conditioned on two boundary points (*top right plot*); Conditioned on two boundary points and $N_f = 3$ from f (*bottom left plot*); Conditioned on the two boundary points and $N_f = 8$ from f (*bottom right plot*). The red line is the solution (2.6) and the grey region is the 99.7% confidence interval.

conditions is

$$u(x) = -5x^2 + cx + c', \tag{2.7}$$

where c and c' are two constants. Though the system is under-determined, we can still apply PI-GPR to obtain a physics-constrained Gaussian process posterior. In Figure 2.2, we illustrate the mean and the samples of the Gaussian process conditioned on $N_f = 8$ points from f and without boundary conditions. We see that the shape of the mean and the samples are like a parabola, which coincides with the general solution (2.7). This predictive Gaussian process can also be understood as a physics-constrained prior, which will converge faster than the original prior when incorporating the direct observations of u.

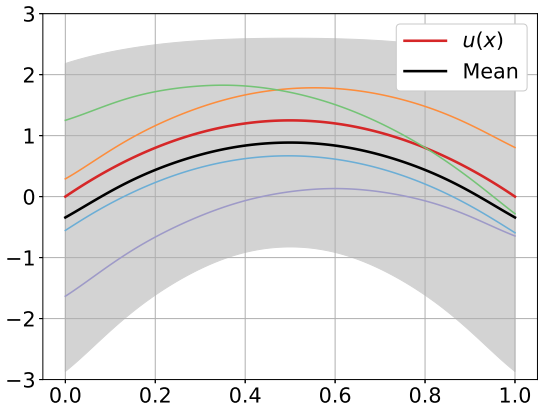

Figure 2.2: The mean and the samples of PI-GPR, with $N_f = 8$ and no boundary conditions. The red line is the solution (2.6) and the grey region is the 99.7% confidence interval.

On the other hand, when predicting or simulating a real-world phenomenon, the physical model described by PDEs could be imperfect, which means even the exact solution of PDEs may fail to provide accurate predictions. In this case, we want to use experimental or observed data to improve the accuracy or robustness of the prediction. One significant advantage of PI-GPR is that, as a data-driven model, it can easily integrate known data points into the predictive Gaussian process. For example, along with (2.5), assume that we are given a data point $y = 1.5$ at $x = 0.5$. We notice that the data is inconsistent with the equation, since $u(0.5) = 1.25 \neq y$. The inconsistency may be due to an imperfect model or noise in the data, which is common in real applications. We

now employ PI-GPR to make the prediction. We assume that the noise ξ in data follows a Gaussian distribution, that is, $\xi \sim \mathcal{N}(0, 10^{-2})$, so we have $y = u(0.5)+\xi$. The calculation of the predictive Gaussian process, conditioned on the data y and physics information, can be done by a combination of (1.9) and (2.4). Comparing the top left plot of Figure 2.3 to the bottom left plot of Figure 2.1, we see that the additional data point makes the mean and the samples move upwards to $y = 1.5$ and reduce the uncertainty in the middle to the level of noise in the data. If we increase N_f from 3 to 8 (top right plot in Figure 2.3), more points from the PDE revert the predictive Gaussian process to the analytical solution of the PDE. In addition, if we assume the data is noise-free, the predictive Gaussian process will be forced to go through the data point and the uncertainty at the point will be reduced to zero, as shown in the bottom plots of Figure 2.3.

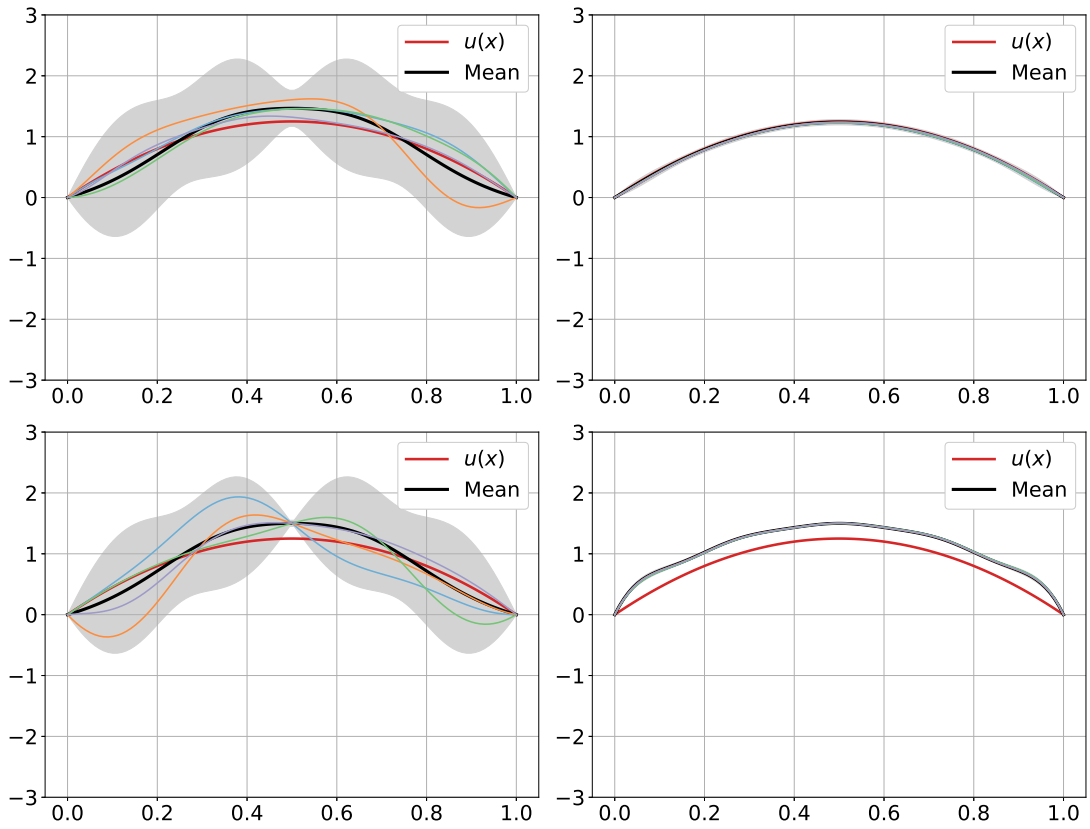

Figure 2.3: The mean and the samples of PI-GPR, conditioned on PDE, boundary conditions and additional data point $y = 1.5$ at $x = 0.5$: $N_f = 3$, noisy data y (*top left plot*); $N_f = 8$, noisy data y (*top right plot*); $N_f = 3$, noise-free data y (*bottom left plot*); $N_f = 8$, noise-free data y (*bottom right plot*). The red line is the solution (2.6) and the grey region is the 99.7% confidence interval.

Therefore, when the PDE does not fully capture the real physical process, for example, due to model simplifications or unknown boundary conditions, PI-GPR provides a framework to leverage the observational data in predictions, allowing it to produce a solution that is not strictly a solution of the PDE but could possibly be instead a more realistic representation of the physical system. Thus, it is ambiguous to only describe PI-GPR as an approximation of the PDE solution. In the next section, we aim to further explore the meaning of solution provided by the method from a linear system perspective.

2.3 Linear system perspective of PI-GPR

In this section, we discuss and try to understand PI-GPR from a linear system perspective, using a simplified setting where the system is finite-dimensional, specifically, solving a system of linear equations. We will later build a connection with the infinite-dimensional case.

We start by considering $A\mathbf{u} = \mathbf{f}$, where $A \in \mathbb{R}^{m\times n}$ is of full row rank. In this case, A has m linearly independent columns, so $\mathbf{f}$ is in the column space of A. Therefore, there exists at least one solution $\mathbf{u}$ such that $A\mathbf{u} - \mathbf{f} = 0$. We assume a multivariate Gaussian prior on $\mathbf{u}$ as $\mathbf{u} \sim \mathcal{N}(0, K)$, where $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ and k is a covariance function. Then, we have the covariance $\mathrm{Cov}(\mathbf{u}, \mathbf{f}) = KA^T$ and $\mathrm{Cov}(\mathbf{f}, \mathbf{f}) = AKA^T$, so the joint distribution between $\mathbf{u}$ and $\mathbf{f}$ is

$$\begin{bmatrix}\mathbf{u}\\ \mathbf{f}\end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix}K & KA^T\\ AK & AKA^T\end{bmatrix}\right).$$

When the covariance matrix K is positive definite, we can compute the conditional distribution as

$$\mathbf{u}|\mathbf{f} \sim \mathcal{N}\left(KA^T(AKA^T)^{-1}\mathbf{f}, K - KA^T(AKA^T)^{-1}AK\right),$$

where AKA^T is invertible since A is full row rank and K is full rank. We demonstrate that every sample from the conditional Gaussian $\mathbf{u}|\mathbf{f}$ is a solution to the system and its predictive mean is the minimum norm solution. Formally, we have

Theorem 1. *Given a linear equation system* $A\mathbf{u} = \mathbf{f}$*, where* $A \in \mathbb{R}^{m\times n}$ *is of full row rank, for any positive definite covariance matrix* $K \in \mathbb{R}^{n\times n}$*, we have the conditional distribution*

$$\mathbf{u}|\mathbf{f} \sim \mathcal{N}\left(KA^T(AKA^T)^{-1}\mathbf{f}, K - KA^T(AKA^T)^{-1}AK\right),$$

such that, any sample obtained from the conditional distribution is a solution to the system of linear equations. Furthermore, the predictive mean of the conditional distribution is the minimum norm solution, where the norm is defined by: $\|\mathbf{x}\|_K := \sqrt{\mathbf{x}^T K^{-1}\mathbf{x}}$.

The proof of the theorem largely depends on the following results from [68]:

Definition 1. *Let* $A \in \mathbb{R}^{m\times n}$*, then* $A^\dagger$ *is a* ***generalized inverse*** *of* A *if* $AA^\dagger A = A$*. (Note that the Moore-Penrose inverse additionally requires that* $A^+AA^+ = A^+$*,* $(A^+A)^T = A^+A$ *and* $(AA^+)^T = AA^+$*.)*

Lemma 2. *Let* $A\mathbf{u} = \mathbf{f}$ *be a consistent equation (i.e. exists at least one solution) and* $\mathbf{u} = G\mathbf{f}$ *a minimum norm solution, where the norm is defined by* $\|\mathbf{x}\|_{N^{-1}} := \sqrt{\mathbf{x}^T N\mathbf{x}}$*. Then it is necessary and sufficient that* $AGA = A$*,* $(GA)^T N = NGA$*.*

Lemma 3. *Let* $A\mathbf{u} = \mathbf{f}$ *be a consistent equation and* $A^\dagger$ *be a generalised inverse of* A*, then the class of all solution is provided by* $A^\dagger\mathbf{f} + (I - A^\dagger A)\mathbf{z}$*, where* $\mathbf{z} \in \mathrm{R}^n$ *is an arbitrary vector.*

Proof of Theorem 1. We first prove that $KA^T(AKA^T)^{-1}$ in the predictive mean is a generalized inverse of A. This generalized inverse was defined by Rao [68] (see Definition 1). Unlike the more commonly used Moore-Penrose pseudo-inverse, this definition does not satisfy all the restrictions imposed by Moore-Penrose inverse. Let $G = KA^T(AKA^T)^{-1}$, we have $AGA = A(KA^T)(AKA^T)^{-1}A = (AKA^T)(AKA^T)^{-1}A = A$, and hence G is a generalised inverse of A.

Now we prove that $\mathbf{u} = G\mathbf{f}$ is the minimum norm solution by using Lemma 2. We have verified the first condition above, secondly, we have $(GA)^T N = (KA^T(AKA^T)^{-1}A)^T N = (A^T(AKA^T)^{-1}AK)N$ and $NGA = N(KA^T(AKA^T)^{-1}A)$. When $N = K$, we have $(GA)^T N = NGA$. Hence, by Lemma 2, we show that the

predictive mean of the conditional Gaussian distribution is the minimum norm solution, where the norm is defined by $\|\mathbf{x}\|_{K^{-1}} := \sqrt{\mathbf{x}^T K \mathbf{x}}$.

Then we prove that any sample from the predictive Gaussian process is a solution to the system. We write the conditional Gaussian process as $\mathbf{u}|\mathbf{f} = KA^T(AKA^T)^{-1}\mathbf{f} + C^T\xi$, where $CC^T = K - KA^T(AKA^T)^{-1}AK$ and $\xi \in \mathcal{N}(0,1)$. We have $K - KA^T(AKA^T)^{-1}AK = K - GAK = (I - GA)K$. Meanwhile, since C and $(I - GA)K$ have the same column space, C can be expressed in the form of $C = (I - GA)C'$, where C' is an arbitrary matrix. Therefore, by Lemma 3, samples from $\mathbf{u}|\mathbf{f} = G\mathbf{f} + (I - GA)C'\xi$ are the solutions of the linear system. □

Therefore, we see that in the finite-dimensional case, the conditional Gaussian distribution $\mathbf{u}|\mathbf{f}$ provides a statistical model for the solutions of the linear system. When we solve the differential equation with PI-GPR, to understand the infinite-dimensional case in analogue to the finite-dimensional case intuitively, we let the dimension of $\mathbf{u}$ tend to infinity. In this case, the prior we put on $\mathbf{u}$ aligns with the Gaussian process prior we put on u in (2.2) by the definition of Gaussian process. Meanwhile, the matrix A represents the operator $\delta_{X_f} \circ \mathcal{L}$, where δ_{X_f} denotes the point evaluation operator that maps a function to its values at points X_f. In addition, we have $\delta_{X_f}(f) = \mathbf{f}(X_f)$. Therefore, when applying PI-GPR to solve a linear operator equation, the predictive Gaussian process (2.3) can be understood as providing a statistical model for the solutions of $\delta_{X_f}(\mathcal{L}(u) - f) = 0$.

2.4 Conclusion and future work

In this chapter, we introduced the physics-informed Gaussian process regression (PI-GPR) method. In the numerical experiments, we showed that PI-GPR can be used to approximate the solution of linear operator equations with uncertainty quantification. When the system solely depends on the PDE constraints without additional observational data, the predictive mean of the PI-GPR is the same as the numerical solution obtained by the symmetric collocation method [19]. An discussion of the convergence rate of the symmetric collocation method can be found in [31]. When the system of equations is partially known, PI-GPR can

provide a physics-constrained prior. In addition, when a real-world phenomenon is modelled by an imperfect physics model, its capability of integrating experimental or observed data can enhance the prediction accuracy and robustness. In the third section, we discussed the nature of the method from a linear system perspective. We started the exploration from the case of a system of linear equations $A\mathbf{u} = \mathbf{f}$. We assumed Gaussian priors on the solution $\mathbf{u}$ and right-hand-side $\mathbf{f}$ and showed that the conditional distribution $\mathbf{u}|\mathbf{f}$ provides a statistical model for the solutions of the system. We then extended the discussion into an infinite-dimensional case in a non-rigorous way and provided insight into PI-GPR. In future, we will further explore and formalise the connection between the finite-dimensional case and the infinite-dimensional case.

Chapter 3

PDE-constrained GP emulator in Bayesian Inverse Problem and MCMC

3.1 Introduction

When solving Bayesian inverse problems, a major challenge in the application of MCMC methods to problems of practical interest is the large computational cost associated with numerically solving the mathematical model for a given set of input parameters. Since the generation of each sample by an MCMC method requires a solve of the governing equations, and often millions of samples are required in practical applications, this process can quickly become very costly. One way to deal with the challenge of full Bayesian inference for complex models is the use of surrogate models, also known as emulators, meta-models or reduced-order models. Instead of using the complex (and computationally expensive) model, one can use a simpler and computationally more efficient model to approximate the solution of the governing equations, which in turn is used to approximate the likelihood function. Within the statistics literature, the most commonly used type of surrogate model is a Gaussian process emulator [69, 83, 76, 47, 61, 44], but other types of surrogate models can also be used including projection-based methods [13], generalised Polynomial Chaos [92, 53], sparse grid collocation [6, 52]

and adaptive subspace methods [21, 22].

In this chapter, we focus on the use of Gaussian process surrogate models for approximating the posterior distribution in inverse problems, where the forward model is related to the solution of a linear partial differential equation (PDE). In particular, we consider two different ways of using the surrogate model, emulating either the parameter-to-observation map or the negative log-likelihood. Convergence properties of the corresponding posterior approximations, as the number of design points N used to construct the surrogate model goes to infinity, have recently been studied in [84, 88, 41]. These results put the methodology on a firm theoretical footing, and show that the error in the approximate posterior distribution can be bounded by the corresponding error in the surrogate model. Furthermore, the error in the approximate posteriors tends to zero as N tends to infinity. However, when the forward model of interest is given by a complex model such as a PDE, one normally operates in a regime where only a very limited number of design points N can be used due to constraints on computational cost. This setting is less understood and is the main setting of practical interest.

With a small number of design points, different modelling choices made in the derivation of the approximate posterior can have a large effect on its accuracy. In particular, the choice of Gaussian prior distribution in the emulator is crucial, as it heavily influences its accuracy. Intuitively, we want to make the prior distribution as informative as possible, by incorporating known information about the underlying forward model. For example, an informed prior specially tailored to solving the forward problem in linear PDEs can be found in [67]. For incorporating more general constraints, we refer the reader to the recent review [87]. Other modelling choices that require careful consideration are whether we build a surrogate model for the parameter-to-observation map or the log-likelihood directly, and whether we use the full distribution of the emulator or only the mean (see e.g. [84, 50]).

Our focus is on computational aspects of the use of Gaussian process surrogate models in PDE inverse problems, with particular emphasis on the setting where the number of design points is limited by computational constraints. The main contributions of this chapter are the following:

1. We extend the PDE-informed Gaussian process priors from [67] to enable their use in inverse problems, which requires a Gaussian process prior as a function of both the spatial variable of the PDE and the unknown parameter(s).

2. By showing that the required gradients can be computed explicitly, we establish that gradient-based MCMC samplers such as the Metropolis-adjusted Langevin algorithm (MALA) can be used to efficiently sample from the approximate posterior distributions.

3. Using a range of numerical examples, we demonstrate the isolated effects of various modelling choices made, and thus offer valuable insights and guidance for practitioners. This includes choices of posterior approximation in the inverse problem (e.g. emulating the parameter-to-observation map or the log-likelihood) and on prior distributions for the Gaussian process emulator (e.g. black-box or PDE-constrained).

   The rest of the chapter is organised as follows. In Section 3.2 we set up notation with respect to the inverse problems of interest and discuss the different kinds of posterior approximations that result from using Gaussian surrogate models for the data-likelihood. We then proceed in Section 3.3 to present our main methodology, discussing how one can blend better-informed Gaussian surrogate models with inverse problems as well as presenting the MCMC algorithm that we use. A number of different numerical experiments that illustrate the computational benefits of our approach are then presented in Section 3.4, and finally Section 3.5 provides a summary and discussion of the main results.

3.2 Preliminaries

We now give more details about the type of inverse problems considered in this chapter and discuss different aspects of Gaussian emulators, as well as the corresponding type of approximate posteriors considered in this work.

3.2.1 PDE inverse problems

Consider the linear PDE with a boundary condition

$$\begin{aligned} \mathcal{L}^{\boldsymbol{\theta}} u(\mathbf{x}) &= f(\mathbf{x}), \qquad \mathbf{x} \in D, \\ \mathcal{B} u(\mathbf{x}) &= g(\mathbf{x}), \qquad \mathbf{x} \in \partial D, \end{aligned} \tag{3.1}$$

posed on a domain $D \subseteq \mathbb{R}^{d_{\mathbf{x}}}$. Comparing to (2.1a) in the previous section, we use notation $\mathcal{L}^{\boldsymbol{\theta}}$ here to emphasise that the linear differential operator depends on parameters $\boldsymbol{\theta} \in \mathcal{T} \subseteq \mathbb{R}^{d_{\boldsymbol{\theta}}}$. The linear operator $\mathcal{B}$ incorporates boundary conditions. The inverse problem of interest is to infer the parameters $\boldsymbol{\theta}$ from the noisy data $\mathbf{y} \in \mathbb{R}^{d_{\mathbf{y}}}$ given by

$$\mathbf{y} = \mathcal{G}_X(\boldsymbol{\theta}) + \boldsymbol{\eta}, \tag{3.2}$$

where $X = \{\mathbf{x}_1, \cdots, \mathbf{x}_{d_{\mathbf{y}}}\} \subset \overline{D}$ are the spatial points where we observe the solution u of our PDE, $\mathcal{G}_X : \mathcal{T} \to \mathbb{R}^{d_{\mathbf{y}}}$ is the *parameter-to-observation map* defined by $\mathcal{G}_X(\boldsymbol{\theta}) = \{u(\mathbf{x}_j; \boldsymbol{\theta})\}_{j=1}^{d_{\mathbf{y}}}$, and $\boldsymbol{\eta} \sim \mathcal{N}(0, \Gamma_\eta)$ is an additive Gaussian noise term with known covariance matrix. Note that the assumption of Gaussianity and diagonal noise covariance is made for simplicity, but these assumptions can be relaxed [50]. Likewise, the methodology generalises straightforwardly to general bounded linear observation operators applied to the PDE solution u, such as integral operators. From (3.2) we have $\mathbf{y}|\boldsymbol{\theta} \sim \mathcal{N}(\mathcal{G}_X(\boldsymbol{\theta}), \Gamma_\eta)$, so the *likelihood* is

$$L(\mathbf{y}|\boldsymbol{\theta}) \propto \exp\left(-\frac{1}{2}\|\mathcal{G}_X(\boldsymbol{\theta}) - \mathbf{y}\|_{\Gamma_\eta}^2\right) := \exp\left(-\Phi(\boldsymbol{\theta}, \mathbf{y})\right), \tag{3.3}$$

where the function $\Phi : \mathcal{T} \times \mathbb{R}^{d_{\mathbf{y}}} \to \mathbb{R}$ is called the *negative log-likelihood* or *potential* and $\|\mathbf{z}\|_{\Gamma_\eta} = \mathbf{z}^{\mathrm{T}} \Gamma_\eta^{-1} \mathbf{z}$ denotes the norm weighted by Γ_η^{-1}. Then by Bayes' formula we have the posterior

$$\pi(\boldsymbol{\theta}|\mathbf{y}) \propto L(\mathbf{y}|\boldsymbol{\theta}) \pi_0(\boldsymbol{\theta}). \tag{3.4}$$

3.2.2 Gaussian emulators and approximate posteriors

We now discuss two different approaches for constructing a Gaussian emulator and using it for approximating the posterior (3.4). The first approach constructs an emulator for the forward map $\mathcal{G}_X$, while the second approach is based on constructing an emulator directly for the log-likelihood [80, 45].

Emulating the forward map

First, we assume a multi-dimensional Gaussian process prior

$$\mathcal{G}_X^0(\boldsymbol{\theta}) \sim \mathrm{GP}(\mathbf{m}(\boldsymbol{\theta}), K(\boldsymbol{\theta}, \boldsymbol{\theta}')),$$

where $\mathbf{m} : \mathcal{T} \to \mathbb{R}^{d_\mathbf{y}}$ is a mean function and $K(\boldsymbol{\theta}, \boldsymbol{\theta}') : \mathcal{T} \times \mathcal{T} \to \mathbb{R}^{d_\mathbf{y} \times d_\mathbf{y}}$ is the matrix-valued covariance function which represents the covariance between the different entries of $\mathcal{G}_X(\boldsymbol{\theta})$ evaluated at $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$. For simplicity, we assume that the matrix $K(\boldsymbol{\theta}, \boldsymbol{\theta}')$ takes the form

$$K(\boldsymbol{\theta}, \boldsymbol{\theta}') = k(\boldsymbol{\theta}, \boldsymbol{\theta}') I_{d_\mathbf{y}}$$

for some scalar-valued covariance function $k(\boldsymbol{\theta}, \boldsymbol{\theta}') : \mathcal{T} \times \mathcal{T} \to \mathbb{R}$, implying that the entries of $\mathcal{G}_X(\boldsymbol{\theta})$ are independent. We call the model based on this assumption the *baseline* model. As we will see later, better emulators can be constructed by relaxing this independence assumption.

Given the data set $\mathcal{G}_X(\Theta) = \{\mathcal{G}_X(\boldsymbol{\theta}^i)\}_{i=1}^N$, we can now proceed with building our Gaussian process emulator for the forward map $\mathcal{G}_X$. We derive the predictive Gaussian process $\mathcal{G}_X^N$ in the same way as in (1.8) and also use similar notation to (1.8) for its mean and covariance matrix. One then needs to decide how to incorporate the emulation for the construction of an approximate posterior. In particular, depending on what type of information we plan to utilize, different approximations will be obtained. If we use its predictive mean $\mathbf{m}_N^{\mathcal{G}_X}$ as a point estimator of the forward map $\mathcal{G}_X$, we obtain

$$\pi_{\mathrm{mean}}^{N,\mathcal{G}_X}(\boldsymbol{\theta}|\mathbf{y}) \propto \exp\left(-\frac{1}{2}\|\mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta}) - \mathbf{y}\|_{\Gamma_\eta}^2\right) \pi_0(\boldsymbol{\theta}). \tag{3.5}$$

Alternatively, we can try to exploit the full information given by the Gaussian process by incorporating its variance in the posterior approximation. A natural way to do this is to consider the following approximation [1]:

$$\begin{aligned}\pi^{N,\mathcal{G}_X}_{\text{marginal}}(\boldsymbol{\theta}|\mathbf{y}) &\propto \mathbb{E}\left(\exp\left(-\frac{1}{2}\|\mathcal{G}^N_X(\boldsymbol{\theta})-\mathbf{y}\|^2_{\Gamma_\eta}\right)\pi_0(\boldsymbol{\theta})\right)\\ &\propto \left(\frac{\exp\left(-\frac{1}{2}\|\mathbf{m}^{\mathcal{G}_X}_N(\boldsymbol{\theta})-\mathbf{y}\|^2_{(K_N(\boldsymbol{\theta},\boldsymbol{\theta})+\Gamma_\eta)}\right)}{\sqrt{(2\pi)^{d_\mathbf{y}}\det\left(K_N(\boldsymbol{\theta},\boldsymbol{\theta})+\Gamma_\eta\right)}}\right)\pi_0(\boldsymbol{\theta}),\end{aligned} \tag{3.6}$$

where the expectation is taken over the probability space of the Gaussian process posterior. Note that the marginal approximation is not exactly a Gaussian distribution, since the covariance also depends on the parameter $\boldsymbol{\theta}$.

The derivation of (3.6) results from the fact that the convolution of two Gaussian measures is Gaussian. We first simplify the notation by letting $\mathbf{m}_{\boldsymbol{\theta}} = \mathbf{m}^{\mathcal{G}_X}_N(\boldsymbol{\theta})$, $K_{\boldsymbol{\theta}} = K_N(\boldsymbol{\theta},\boldsymbol{\theta})$ and $\Gamma_\eta = \sigma_\eta^{-2} I_{d_\mathbf{y}}$. Since $\mathcal{G}^N_X(\boldsymbol{\theta}) = \mathbf{m}_{\boldsymbol{\theta}} + \boldsymbol{\xi}$, where $\boldsymbol{\xi} \sim \mathcal{N}(0, K_{\boldsymbol{\theta}})$, using the definition of the expectation we obtain

$$\begin{aligned}&\mathbb{E}\left(\exp\left(-\frac{1}{2}\|\mathcal{G}^N_X(\boldsymbol{\theta})-\mathbf{y}\|^2_{\Gamma_\eta}\right)\pi_0(\boldsymbol{\theta})\right)\\ &= \frac{1}{\sqrt{(2\pi)^{d_\mathbf{y}}\det\left(K_{\boldsymbol{\theta}}\right)}}\int_{\mathbb{R}^{d_\mathbf{y}}}\exp\left(-\frac{\|\mathbf{m}_{\boldsymbol{\theta}}+\boldsymbol{\xi}-\mathbf{y}\|^2_{\Gamma_\eta}}{2}\right)\exp\left(-\frac{\|\boldsymbol{\xi}\|^2_{K_{\boldsymbol{\theta}}}}{2}\right)d\boldsymbol{\xi}.\end{aligned}$$

We then rewrite and simplify the exponent part in the formula. Let $\bar{\mathbf{y}} = \mathbf{y} - \mathbf{m}_{\boldsymbol{\theta}}$, then

$$\begin{aligned}&-\frac{1}{2}\left(\|\boldsymbol{\xi}-(\mathbf{y}-\mathbf{m}_{\boldsymbol{\theta}})\|^2_{\Gamma_\eta}+\|\boldsymbol{\xi}\|^2_{K_{\boldsymbol{\theta}}}\right)\\ &= -\frac{1}{2}\left(\|\boldsymbol{\xi}-\bar{\mathbf{y}}\|^2_{\Gamma_\eta}+\|\boldsymbol{\xi}\|^2_{K_{\boldsymbol{\theta}}}\right)\\ &= -\frac{1}{2}\left((\boldsymbol{\xi}-\bar{\mathbf{y}})^T\Gamma_\eta^{-1}(\boldsymbol{\xi}-\bar{\mathbf{y}})+\boldsymbol{\xi}^T K_{\boldsymbol{\theta}}^{-1}\boldsymbol{\xi}\right)\\ &= -\frac{1}{2}\left(\boldsymbol{\xi}^T(\Gamma_\eta^{-1}+K_{\boldsymbol{\theta}}^{-1})\boldsymbol{\xi}.-2\bar{\mathbf{y}}^T\Gamma_\eta^{-1}\boldsymbol{\xi}+\bar{\mathbf{y}}^T\Gamma_\eta^{-1}\bar{\mathbf{y}}\right)\end{aligned}$$

[1] For completeness, the formula was also derived in [14, 19].

Since Γ_η and $K_{\boldsymbol{\theta}}$ are symmetric matrices, we have

$$\begin{aligned}
&\bar{\mathbf{y}}^T\Gamma_\eta^{-1}\boldsymbol{\xi} \\
&= \bar{\mathbf{y}}^T((K_{\boldsymbol{\theta}}+\Gamma_\eta)^{-1}K_{\boldsymbol{\theta}})(K_{\boldsymbol{\theta}}^{-1}(K_{\boldsymbol{\theta}}+\Gamma_\eta))\Gamma_\eta^{-1}\boldsymbol{\xi} \\
&= (K_{\boldsymbol{\theta}}(K_{\boldsymbol{\theta}}+\Gamma_\eta)^{-1}\bar{\mathbf{y}})^T K_{\boldsymbol{\theta}}^{-1}(K_{\boldsymbol{\theta}}+\Gamma_\eta)\Gamma_\eta^{-1}\boldsymbol{\xi} \\
&= \tilde{\mathbf{y}}^T C^{-1}\boldsymbol{\xi},
\end{aligned}$$

where $C = K_{\boldsymbol{\theta}}(K_{\boldsymbol{\theta}}+\Gamma_\eta)^{-1}\Gamma_\eta$ and $\tilde{\mathbf{y}} = C\Gamma_\eta^{-1}\bar{\mathbf{y}}$. Substituting it into the formula above, we have

$$= -\frac{1}{2}\left(\boldsymbol{\xi}^T C^{-1}\boldsymbol{\xi} - 2\tilde{\mathbf{y}}^T C^{-1}\boldsymbol{\xi} + \bar{\mathbf{y}}^T\Gamma_\eta^{-1}\bar{\mathbf{y}}\right)$$

We can then complete the square

$$\begin{aligned}
&= -\frac{1}{2}\left(\|\boldsymbol{\xi}-\tilde{\mathbf{y}}\|_C^2 - \tilde{\mathbf{y}}^T C^{-1}\tilde{\mathbf{y}} + \bar{\mathbf{y}}^T\Gamma_\eta^{-1}\bar{\mathbf{y}}\right) \\
&= -\frac{1}{2}\left(\|\boldsymbol{\xi}-\tilde{\mathbf{y}}\|_C^2 - (C\Gamma_\eta^{-1}\bar{\mathbf{y}})^T C^{-1}(C\Gamma_\eta^{-1}\bar{\mathbf{y}}) + \bar{\mathbf{y}}^T\Gamma_\eta^{-1}\bar{\mathbf{y}}\right) \\
&= -\frac{1}{2}\left(\|\boldsymbol{\xi}-\tilde{\mathbf{y}}\|_C^2 - \bar{\mathbf{y}}^T\Gamma_\eta^{-1}K_{\boldsymbol{\theta}}(K_{\boldsymbol{\theta}}+\Gamma_\eta)^{-1}\bar{\mathbf{y}}) + \bar{\mathbf{y}}^T\Gamma_\eta^{-1}\bar{\mathbf{y}}\right) \\
&= -\frac{1}{2}\left(\|\boldsymbol{\xi}-\tilde{\mathbf{y}}\|_C^2 - \bar{\mathbf{y}}^T(\Gamma_\eta^{-1}K_{\boldsymbol{\theta}}(K_{\boldsymbol{\theta}}+\Gamma_\eta)^{-1} - \Gamma_\eta^{-1})\bar{\mathbf{y}})\right) \\
&= -\frac{1}{2}\left(\|\boldsymbol{\xi}-\tilde{\mathbf{y}}\|_C^2 + \bar{\mathbf{y}}^T(K_{\boldsymbol{\theta}}+\Gamma_\eta)^{-1}\bar{\mathbf{y}}\right) \\
&= -\frac{1}{2}\|\bar{\mathbf{y}}\|_{(K_{\boldsymbol{\theta}}+\Gamma_\eta)}^2 - \frac{1}{2}\left(\|\boldsymbol{\xi}-\tilde{\mathbf{y}}\|_C^2\right)
\end{aligned}$$

We now factor out $\exp\left(-\frac{1}{2}\left(\|\boldsymbol{\xi}-\tilde{\mathbf{y}}\|_C^2\right)\right)$ from the integral, and the remaining part matches the form of Gaussian distribution up to a constant.

$$\begin{aligned}
&= \frac{\sqrt{\det(C)}}{\sqrt{\det(K_{\boldsymbol{\theta}})}}\exp\left(-\frac{1}{2}\|\bar{\mathbf{y}}\|_{(K_{\boldsymbol{\theta}}+\Gamma_\eta)}^2\right) \\
&\qquad \int_{\mathbb{R}^{d_{\mathbf{y}}}} \frac{1}{\sqrt{(2\pi)^{d_{\mathbf{y}}}\det(C)}}\exp\left(-\frac{1}{2}\left(\|\boldsymbol{\xi}-\tilde{\mathbf{y}}\|_C^2\right)\right)d\boldsymbol{\xi}
\end{aligned}$$

Hence, we obtain the explicit form of the marginal approximation.

$$\mathbb{E}\left(\exp\left(-\frac{1}{2}\|\mathcal{G}_X^N(\boldsymbol{\theta})-\mathbf{y}\|_{\Gamma_\eta}^2\right)\pi_0(\boldsymbol{\theta})\right)$$
$$\propto \frac{1}{\sqrt{(2\pi)^{d_\mathbf{y}}\det(K_{\boldsymbol{\theta}}+\Gamma_\eta)}}\exp\left(-\frac{1}{2}\|\mathbf{y}-\mathbf{m}_{\boldsymbol{\theta}}\|_{(K_{\boldsymbol{\theta}}+\Gamma_\eta)}^2\right)\pi_0(\boldsymbol{\theta})$$

Comparing (3.6) with (3.5), the likelihood function in the marginal approximation is Gaussian with additional uncertainty $K_N(\boldsymbol{\theta},\boldsymbol{\theta})$ from the emulator included into its covariance matrix. Hence, for a fixed parameter $\boldsymbol{\theta}$, the likelihood function in (3.6) will be less concentrated due to variance inflation. When the magnitude of $K_N(\boldsymbol{\theta},\boldsymbol{\theta})$ is small compared to that of Γ_η, the marginal approximation will be similar to the mean-based approximation.

Comparing (3.6) with (3.5), the term $K_N(\boldsymbol{\theta},\boldsymbol{\theta})$ in the likelihood function of the marginal approximation represents the variance contribution from the emulator at a fixed parameter $\boldsymbol{\theta}$, indicating the additional uncertainty introduced due to the approximation error of the forward model. When the magnitude of each entry of $K_N(\boldsymbol{\theta},\boldsymbol{\theta})$ is much smaller than Γ_η, it means that the emulator's uncertainty is negligible compared to the observational noise variance. In such cases, the total covariance matrix in the likelihood function is dominated by Γ_η and the marginal approximation behaves similarly to the mean-based approximation. Conversely, if $K_N(\boldsymbol{\theta},\boldsymbol{\theta})$ is significant relative to Γ_η, the covariance matrix will reflect more uncertainty, leading to a larger value of the likelihood function and the posterior. Therefore, the uncertainty of the emulator can be propagated to the approximation of the posterior. The marginal approximation will put more weight on the areas that have larger uncertainty and will be less likely to be highly concentrated.

Emulating the log-likelihood

Another way of building the emulator is to model the potential function Φ in (3.3) directly. We can convert the data set $\mathcal{G}_X(\Theta)$ into a data set of negative log-likelihood $\boldsymbol{\Phi}(\Theta)=\{\Phi(\boldsymbol{\theta}^i,\mathbf{y})\}_{i=1}^N$. We denote the predictive Gaussian process by Φ^N and use a similar notation to (1.8) again for its mean and covariance matrix. If we only include the mean of the Gaussian process emulator, the posterior

approximation becomes

$$\pi^{N,\Phi}_{\text{mean}}(\boldsymbol{\theta}|\mathbf{y}) \propto \exp\left(-m^{\Phi}_{N}(\boldsymbol{\theta})\right)\pi_0(\boldsymbol{\theta}), \tag{3.7}$$

while, in a similar fashion to the forward map emulation, we can take into account the covariance of our emulator to obtain the approximate posterior

$$\begin{aligned}\pi^{N,\Phi}_{\text{marginal}}(\boldsymbol{\theta}|\mathbf{y}) &\propto \mathbb{E}\left((\exp\left(-\Phi^{N}(\boldsymbol{\theta})\right))\pi_0(\boldsymbol{\theta})\right) \\ &\propto \exp\left(-m^{\Phi}_{N}(\boldsymbol{\theta}) + \frac{1}{2}k_N(\boldsymbol{\theta},\boldsymbol{\theta})\right)\pi_0(\boldsymbol{\theta}).\end{aligned} \tag{3.8}$$

The derivation of (3.8) is similar to that of (3.6). Let $m_{\boldsymbol{\theta}} = m^{\Phi}_{N}(\boldsymbol{\theta})$ and $k_{\boldsymbol{\theta}} = k_N(\boldsymbol{\theta},\boldsymbol{\theta})$. Since $\Phi^N(\boldsymbol{\theta}) = m_{\boldsymbol{\theta}} + \xi$, where $\xi \sim N(0, k_{\boldsymbol{\theta}})$, by the definition of the expectation and completing the square we obtain

$$\begin{aligned}&\mathbb{E}\left((\exp\left(-\Phi^{N}(\boldsymbol{\theta})\right))\pi_0(\boldsymbol{\theta})\right) \\ &= \frac{1}{\sqrt{2\pi k_{\boldsymbol{\theta}}}}\int_{\mathbb{R}} \exp\left(-m_{\boldsymbol{\theta}} - \xi\right)\exp\left(-\frac{\xi^2}{2k_{\boldsymbol{\theta}}}\right)d\xi \\ &= \frac{\exp(-m_{\boldsymbol{\theta}})}{\sqrt{2\pi k_{\boldsymbol{\theta}}}}\int_{\mathbb{R}} \exp\left(\xi - \frac{\xi^2}{2k_{\boldsymbol{\theta}}}\right)d\xi \\ &= \frac{\exp(-m_{\boldsymbol{\theta}} + \frac{1}{2}k_{\boldsymbol{\theta}})}{\sqrt{2\pi k_{\boldsymbol{\theta}}}}\int_{\mathbb{R}} \exp\left(-\frac{(\xi + k_{\boldsymbol{\theta}})^2}{2k_{\boldsymbol{\theta}}}\right)d\xi \\ &\propto \exp(-m_{\boldsymbol{\theta}} + \frac{1}{2}k_{\boldsymbol{\theta}}).\end{aligned}$$

Note that in this case, the following relationship holds between the two approximate posteriors

$$\pi^{N,\Phi}_{\text{marginal}}(\boldsymbol{\theta}|\mathbf{y}) \propto \pi^{N,\Phi}_{\text{mean}}(\boldsymbol{\theta}|\mathbf{y})\exp\left(\frac{1}{2}k_N(\boldsymbol{\theta},\boldsymbol{\theta})\right),$$

which again illustrates a form of variance inflation for the marginal posterior approximation. In summary, we have two methods for approximating the true posterior: the mean-based approximation and the marginal approximation; and we have two types of emulators: the forward map emulator and the potential function emulator; thus by combination we have four types of approximations in total. The convergence properties of all these approximate posteriors where the

subject of study in [84, 88, 41], where it was proved under suitable assumptions that all of them converge to the true posterior as $N \to \infty$. However, in the case of small N, the difference between the approximate posteriors could be large and which one we choose is important. Furthermore, the type of Gaussian process emulator used plays an even bigger role in this case, and one would like to use a Gaussian prior that is as informative as possible. We discuss how to do this in the next section.

3.3 Methodology

Having described the different types of posterior approximations we will consider, in this section we discuss different modelling approaches for the prior distribution used in our Gaussian emulators. In doing this it is important to note that the function that we are interested to emulate, in this case, the forward map $\mathcal{G}_X(\boldsymbol{\theta})$, depends not only on the parameters $\boldsymbol{\theta}$ of our PDE, but also on the locations of the spatial observations. Thus in terms of modelling, one would like to take this into account and build spatial correlation explicitly into the prior covariance. Note that when emulating the potential Φ instead of the forward map $\mathcal{G}_X$, we are emulating a scalar-valued function. Since Φ is a non-linear function of $\mathcal{G}_X$, it is not possible to extend the ideas of spatial correlation presented in this section to emulating Φ, and in particular, it is not possible to construct a PDE-informed emulator in the same way.

Introducing spatial correlation when emulating $\mathcal{G}_X(\boldsymbol{\theta})$ can be done in two different ways, the first by prescribing some explicit form of spatial correlation, and the second by using the fact that we know that our forward map is associated with the solution of a linear PDE. We do this in Section 3.3.1. It is important to note that in both cases it is possible to calculate the gradients with respect to the parameters $\boldsymbol{\theta}$ in a closed form, which can then be used to sample from the approximate posterior distributions using gradient-based MCMC methods such as MALA. We discuss this in more detail in Section 3.3.3.

3.3.1 Correlated and PDE-informed priors

We now discuss two different approaches to incorporate spatial correlation into our prior covariance function for the forward map $\mathcal{G}_X(\boldsymbol{\theta})$. Even though this is a function from the parameter space $\mathcal{T}$ to the observation space $\mathbb{R}^{d_{\mathbf{y}}}$, for introducing more complicated spatial correlation it is useful to think first about the PDE solution $u(\boldsymbol{\theta}, \mathbf{x})$ as a function from $\mathcal{T} \times \overline{D}$ to $\mathbb{R}$. We introduce the prior covariance function $k((\boldsymbol{\theta}, \mathbf{x}), (\boldsymbol{\theta}', \mathbf{x}'))$ for $u(\boldsymbol{\theta}, \mathbf{x})$, and choose a separable model

$$k((\boldsymbol{\theta}, \mathbf{x}), (\boldsymbol{\theta}', \mathbf{x}')) = k_p(\boldsymbol{\theta}, \boldsymbol{\theta}')k_s(\mathbf{x}, \mathbf{x}'), \tag{3.9}$$

where k_p and k_s are the covariance functions for the parameters $\boldsymbol{\theta}$ and the spatial points $\mathbf{x}$ respectively.

Using the fact that the forward map $\mathcal{G}_X$ relates to the point-wise evaluation of the function $u(\boldsymbol{\theta}, \mathbf{x})$ for $\mathbf{x} \in X$, and assuming zero mean, we then obtain the Gaussian prior

$$\mathcal{G}_X(\boldsymbol{\theta}) \sim \mathrm{GP}(0, K_{uu}(\boldsymbol{\theta}, \boldsymbol{\theta}')), \tag{3.10}$$

with

$$K_{uu}(\boldsymbol{\theta}, \boldsymbol{\theta}') = k_p(\boldsymbol{\theta}, \boldsymbol{\theta}')K_s(X, X),$$

where K_s is the covariance matrix and its entries are the evaluation of k_s at points X, that is, $(K_s(X, X))_{i,j} = k_s(\mathbf{x}_i, \mathbf{x}_j)$, $\mathbf{x}_i, \mathbf{x}_j \in X$. This prior can then be conditioned on training data $\mathcal{G}_X(\Theta)$, and due to the separable structure in (3.9), the predictive mean $\mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta})$ is in fact the same as for the baseline model in Section 3.2.2, where we have

$$\mathcal{G}_X(\boldsymbol{\theta})|\mathcal{G}_X(\Theta) \sim \mathrm{GP}(\mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta}), K_N(\boldsymbol{\theta}, \boldsymbol{\theta}'))$$

with

$$\begin{aligned}
\mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta}) &= K_{uu}(\boldsymbol{\theta}, \Theta)K_{uu}(\Theta, \Theta)^{-1}\mathcal{G}_X(\Theta), \\
K_N(\boldsymbol{\theta}, \boldsymbol{\theta}') &= K(\boldsymbol{\theta}, \boldsymbol{\theta}') - K_{uu}(\boldsymbol{\theta}, \Theta)K_{uu}(\Theta, \Theta)^{-1}K(\boldsymbol{\theta}', \Theta)^T
\end{aligned}$$

and

$$K_{uu}(\Theta,\Theta) = \{k_p(\boldsymbol{\theta}^i,\boldsymbol{\theta}^j)K_s(X,X)\} \in \mathbb{R}^{Nd_{\mathbf{y}}\times Nd_{\mathbf{y}}},$$
$$K_{uu}(\boldsymbol{\theta},\Theta) = \{k_p(\boldsymbol{\theta},\boldsymbol{\theta}^j)K_s(X,X)\} \in \mathbb{R}^{d_{\mathbf{y}}\times Nd_{\mathbf{y}}}.$$

The second way of introducing spatial correlation is explicitly taking into account that the forward map is related to a PDE solution. Given the PDE system

$$\begin{aligned}\mathcal{L}^{\boldsymbol{\theta}}u(\mathbf{x}) &= f(\mathbf{x}), \qquad \mathbf{x}\in D,\\ \mathcal{B}u(\mathbf{x}) &= g(\mathbf{x}), \qquad \mathbf{x}\in \partial D,\end{aligned}$$

as described in Section 3.2, we can build a joint prior between u, f and g. In particular, if we take fixed points $\mathbf{x}, \mathbf{x}_f \in D$ and $\mathbf{x}_b \in \partial D$ we have that

$$\begin{bmatrix} u(\boldsymbol{\theta},\mathbf{x}) \\ g(\boldsymbol{\theta},\mathbf{x}_b) \\ f(\boldsymbol{\theta},\mathbf{x}_f)\end{bmatrix} \sim \mathrm{GP}\left(\mathbf{0}, k_p(\boldsymbol{\theta},\boldsymbol{\theta}')\begin{bmatrix} k_s(\mathbf{x},\mathbf{x}) & \mathcal{B}k_s(\mathbf{x},\mathbf{x}_b) & \mathcal{L}^{\boldsymbol{\theta}'}k_s(\mathbf{x},\mathbf{x}_f)\\ \mathcal{B}k_s(\mathbf{x}_b,\mathbf{x}) & \mathcal{B}\mathcal{B}k_s(\mathbf{x}_b,\mathbf{x}_b) & \mathcal{B}\mathcal{L}^{\boldsymbol{\theta}'}k_s(\mathbf{x}_b,\mathbf{x}_f)\\ \mathcal{L}^{\boldsymbol{\theta}}k_s(\mathbf{x}_f,\mathbf{x}_b) & \mathcal{L}^{\boldsymbol{\theta}}\mathcal{B}k_s(\mathbf{x}_f,\mathbf{x}_b) & \mathcal{L}^{\boldsymbol{\theta}}\mathcal{L}^{\boldsymbol{\theta}'}k_s(\mathbf{x}_f,\mathbf{x}_f)\end{bmatrix}\right), \tag{3.11}$$

where the above is a Gaussian process as a function of $\boldsymbol{\theta}$, and we have used known properties of linear operators applied to Gaussian processes (see e.g. [54]) in the derivation. The idea of a joint prior between u and f was also used in [67, 82, 19, 64], while [17] uses this explicitly in an inverse problem setting. The crucial difference is that in these works u and f were considered as functions of the spatial variable $\mathbf{x}$ only, while here we instead explicitly model the dependency of u on $\boldsymbol{\theta}$.

We then have

$$\begin{bmatrix} \mathcal{G}_X(\boldsymbol{\theta}) \\ g(\boldsymbol{\theta},X_g) \\ f(\boldsymbol{\theta},X_f)\end{bmatrix} \sim \mathrm{GP}\left(\mathbf{0}, K(\boldsymbol{\theta},\boldsymbol{\theta}')\right), \tag{3.12}$$

where

$$K(\boldsymbol{\theta}, \boldsymbol{\theta}') = k_p(\boldsymbol{\theta}, \boldsymbol{\theta}') \begin{bmatrix} K_s(X, X) & \mathcal{B}K_s(X, X_g) & \mathcal{L}^{\boldsymbol{\theta}'}K_s(X, X_f) \\ \mathcal{B}K_s(X_g, X) & \mathcal{B}\mathcal{B}K_s(X_g, X_g) & \mathcal{B}\mathcal{L}^{\boldsymbol{\theta}'}K_s(X_g, X_f) \\ \mathcal{L}^{\boldsymbol{\theta}}K_s(X_f, X) & \mathcal{L}^{\boldsymbol{\theta}}\mathcal{B}K_s(X_f, X_g) & \mathcal{L}^{\boldsymbol{\theta}}\mathcal{L}^{\boldsymbol{\theta}'}K_s(X_f, X_f) \end{bmatrix}$$

and $X_g \subset \partial D$ and $X_f \subset D$ are collections of d_g and d_f points at which g and f have been evaluated, respectively. The formula (3.12) is derived from (3.11), where each block matrix of the spatial correlation matrix is obtained by: first apply the operator $\mathcal{L}$, $\mathcal{B}$ over the covariance function k_s, then evaluate at pointwise observation points X, X_f or X_g. For example, $(\mathcal{B}\mathcal{B}K_s(X_g, X_g))_{i,j} = \mathcal{B}\mathcal{B}k_s(\mathbf{x}_i, \mathbf{x}_j), \mathbf{x}_i, \mathbf{x}_j \in X_g$. Note that the marginal prior placed on $\mathcal{G}_X$ is the same as in (3.10).

The prior (3.12) can then again be conditioned on training data as in Section 3.2.2 on the observations $\mathbf{g}(\Theta)$, where now

$$\mathbf{g} = \begin{bmatrix} \mathcal{G}_X(\cdot) \\ g(\cdot, X_g) \\ f(\cdot, X_f) \end{bmatrix} : \mathcal{T} \to \mathbb{R}^{d_{\mathbf{y}}+d_g+d_f}.$$

After a re-ordering of the observations $\mathbf{g}(\Theta)$, this results in the conditional distribution

$$\mathbf{g}(\boldsymbol{\theta})|\mathbf{g}(\Theta) \sim \mathrm{GP}\left(\mathbf{m}_N^{\mathbf{g}}(\boldsymbol{\theta}), K_N(\boldsymbol{\theta}, \boldsymbol{\theta}')\right),$$

where

$$\begin{aligned} \mathbf{m}_N^{\mathbf{g}}(\boldsymbol{\theta}) &= \tilde{K}(\boldsymbol{\theta}, \Theta)\tilde{K}(\Theta, \Theta)^{-1}\mathbf{g}(\Theta), \\ K_N^{\mathbf{g}}(\boldsymbol{\theta}, \boldsymbol{\theta}') &= K(\boldsymbol{\theta}, \boldsymbol{\theta}') - \tilde{K}(\boldsymbol{\theta}, \Theta)\tilde{K}(\Theta, \Theta)^{-1}\tilde{K}(\boldsymbol{\theta}', \Theta)^{\mathrm{T}}, \end{aligned}$$

with $K(\boldsymbol{\theta}, \boldsymbol{\theta}') = k_p(\boldsymbol{\theta}, \boldsymbol{\theta}')K_s(X, X)$ as before and

$$\tilde{K}(\boldsymbol{\theta}, \Theta) = \begin{bmatrix} K_{uu}(\boldsymbol{\theta}, \Theta) & K_{ug}(\boldsymbol{\theta}, \Theta) & K_{uf}(\boldsymbol{\theta}, \Theta) \\ K_{ug}^{\mathrm{T}}(\boldsymbol{\theta}, \Theta) & K_{gg}(\boldsymbol{\theta}, \Theta) & K_{gf}(\boldsymbol{\theta}, \Theta) \\ K_{uf}^{\mathrm{T}}(\boldsymbol{\theta}, \Theta) & K_{gf}^{T}(\boldsymbol{\theta}, \Theta) & K_{ff}(\boldsymbol{\theta}, \Theta) \end{bmatrix} \in \mathbb{R}^{(d_{\mathbf{y}}+d_f+d_g)\times N(d_{\mathbf{y}}+d_f+d_g)},$$

$$\tilde{K}(\Theta, \Theta) = \begin{bmatrix} K_{uu}(\Theta, \Theta) & K_{ug}(\Theta, \Theta) & K_{uf}(\Theta, \Theta) \\ K_{ug}^{\mathrm{T}}(\Theta, \Theta) & K_{gg}(\Theta, \Theta) & K_{gf}(\Theta, \Theta) \\ K_{uf}^{\mathrm{T}}(\Theta, \Theta) & K_{gf}^{T}(\Theta, \Theta) & K_{ff}(\Theta, \Theta) \end{bmatrix} \in \mathbb{R}^{N(d_{\mathbf{y}}+d_f+d_g)\times N(d_{\mathbf{y}}+d_f+d_g)},$$

$$\mathbf{g}(\Theta) = \begin{bmatrix} \mathcal{G}_X(\Theta) \\ g(\Theta, X_g) \\ f(\Theta, X_f) \end{bmatrix} \in \mathbb{R}^{N(d_{\mathbf{y}}+d_f+d_g)},$$

and

$$\begin{aligned}
&K_{uu}(\Theta, \Theta) = \{k_p(\boldsymbol{\theta}^i, \boldsymbol{\theta}^j)K_s(X, X)\} \in \mathbb{R}^{Nd_{\mathbf{y}}\times Nd_{\mathbf{y}}}, \\
&K_{uu}(\boldsymbol{\theta}, \Theta) \in \mathbb{R}^{d_{\mathbf{y}}\times Nd_{\mathbf{y}}}, \\
&K_{ug}(\Theta, \Theta) = \{k_p(\boldsymbol{\theta}^i, \boldsymbol{\theta}^j)\mathcal{B}K_s(X, X_g)\} \in \mathbb{R}^{Nd_{\mathbf{y}}\times Nd_g}, \\
&K_{ug}(\boldsymbol{\theta}, \Theta) \in \mathbb{R}^{d_{\mathbf{y}}\times Nd_g}, \\
&K_{uf}(\Theta, \Theta) = \{k_p(\boldsymbol{\theta}^i, \boldsymbol{\theta}^j)\mathcal{L}^{\boldsymbol{\theta}^j}K_s(X, X_f)\} \in \mathbb{R}^{Nd_{\mathbf{y}}\times Nd_f}, \\
&K_{uf}(\boldsymbol{\theta}, \Theta) \in \mathbb{R}^{d_{\mathbf{y}}\times Nd_f}, \\
&K_{gg}(\Theta, \Theta) = \{k_p(\boldsymbol{\theta}^i, \boldsymbol{\theta}^j)\mathcal{B}\mathcal{B}K_s(X_g, X_g)\} \in \mathbb{R}^{Nd_g\times Nd_g}, \\
&K_{gg}(\boldsymbol{\theta}, \Theta) \in \mathbb{R}^{d_g\times Nd_g}, \\
&K_{gf}(\Theta, \Theta) = \{k_p(\boldsymbol{\theta}^i, \boldsymbol{\theta}^j)\mathcal{B}\mathcal{L}^{\boldsymbol{\theta}^j}K_s(X_g, X_f)\} \in \mathbb{R}^{Nd_g\times Nd_f}, \\
&K_{gf}(\boldsymbol{\theta}, \Theta) \in \mathbb{R}^{d_g\times Nd_f}, \\
&K_{ff}(\Theta, \Theta) = \{k_p(\boldsymbol{\theta}^i, \boldsymbol{\theta}^j)\mathcal{L}^{\boldsymbol{\theta}^i}\mathcal{L}^{\boldsymbol{\theta}^j}K_s(X_f, X_f)\} \in \mathbb{R}^{Nd_f\times Nd_f}, \\
&K_{ff}(\boldsymbol{\theta}, \Theta) \in \mathbb{R}^{d_f\times Nd_f}, \\
&g(\Theta, X_g) = \{g(\boldsymbol{\theta}^i, X_g)\} \in \mathbb{R}^{Nd_g}, \\
&f(\Theta, X_f) = \{f(\boldsymbol{\theta}^i, X_f)\} \in \mathbb{R}^{Nd_f}.
\end{aligned}$$

The marginal posterior distribution on $\mathcal{G}_X(\boldsymbol{\theta})$ can then be extracted from the above joint posterior by taking the first $d_{\mathbf{y}}$ rows of $\mathbf{m}_N^{\mathbf{g}}$ and the first $d_{\mathbf{y}}$ rows and

columns of $K_N^{\mathbf{g}}$, which gives

$$\mathcal{G}_X(\boldsymbol{\theta})|\mathcal{G}_X(\Theta), g(\Theta, X_g), f(\Theta, X_f) \sim \mathrm{GP}(\mathbf{m}^{\mathcal{G}_X}_{N,X_f,X_g}(\boldsymbol{\theta}), K_{N,X_f,X_g}(\boldsymbol{\theta}, \boldsymbol{\theta}')),$$

where

$$\begin{aligned}
&\mathbf{m}^{\mathcal{G}_X}_{N,X_f,X_g}(\boldsymbol{\theta}) = \begin{bmatrix} K_{uu}(\boldsymbol{\theta}, \Theta), & K_{ug}(\boldsymbol{\theta}, \Theta), & K_{uf}(\boldsymbol{\theta}, \Theta) \end{bmatrix} \tilde{K}(\Theta, \Theta)^{-1} \mathbf{g}(\Theta), \\
&K_{N,X_f,X_g}(\boldsymbol{\theta}, \boldsymbol{\theta}') = \\
&K(\boldsymbol{\theta}, \boldsymbol{\theta}') - \begin{bmatrix} K_{uu}(\boldsymbol{\theta}, \Theta), & K_{ug}(\boldsymbol{\theta}, \Theta), & K_{uf}(\boldsymbol{\theta}, \Theta) \end{bmatrix} \tilde{K}(\Theta, \Theta)^{-1} \begin{bmatrix} K_{uu}(\boldsymbol{\theta}', \Theta) \\ K_{ug}(\boldsymbol{\theta}', \Theta) \\ K_{uf}(\boldsymbol{\theta}', \Theta) \end{bmatrix}.
\end{aligned}$$

Note that in this case we are updating our prior on $\mathcal{G}_X(\boldsymbol{\theta})$ using the observations $g(\Theta, X_g)$ and $f(\Theta, X_f)$ as well as $\mathcal{G}_X(\Theta)$, essentially augmenting the space on which the emulator $\mathcal{G}_X^N(\boldsymbol{\theta})$ is trained. Since g and f are assumed known, these additional observations are cheap to obtain. It is also possible to condition on training data $g(\Theta_g, X_g)$ and $f(\Theta_f, X_f)$, for point sets Θ_g and Θ_f different to Θ, and this has been found to be beneficial in some of the numerical experiments (see Section 3.4).

3.3.2 Computational implementation

We have three different approaches for emulating the forward map and defining the correlation between its components. We will refer to these as the independent, spatially correlated, and PDE-constrained models, respectively. Each of them can be combined with the mean-based or the marginal approximation of the posterior. We note here that for the computational implementation of the spatially correlated model, the introduction of the correlation matrix does not change the predictive mean of the Gaussian process, it only affects the predictive covariance (see Theorem 4 below). This was already noted in [10], but we give a proof for this for completeness. Since the spatial correlation matrix is independent of $\boldsymbol{\theta}$, the covariance matrix between two sets of parameters Θ_1 and Θ_2 can

be computed by the Kronecker product, that is,

$$\underbrace{K(\Theta_1, \Theta_2)}_{N_1 d_{\mathbf{y}} \times N_2 d_{\mathbf{y}}} = \underbrace{K_p(\Theta_1, \Theta_2)}_{(N_1 \times N_2)} \otimes \underbrace{K_s(X, X)}_{(d_{\mathbf{y}} \times d_{\mathbf{y}})}. \tag{3.13}$$

Hence, assuming a spatial correlation of the type (3.9) only affects approximate posteriors that take into account the uncertainty of the emulator.

Theorem 4. *Consider two Gaussian processes* $\mathbf{g}_0(\boldsymbol{\theta}) \sim GP(\mathbf{m}(\boldsymbol{\theta}), k_p(\boldsymbol{\theta}, \boldsymbol{\theta}')I_{d_{\mathbf{y}}})$, $\mathbf{g}_{0,s}(\boldsymbol{\theta}) \sim GP(\mathbf{m}(\boldsymbol{\theta}), k_p(\boldsymbol{\theta}, \boldsymbol{\theta}')K_s(X, X))$, *where* $K_s(X, X)$ *is the covariance matrix on the set of spatial points* $X = \{\mathbf{x}_i\}_{i=1}^{d_{\mathbf{y}}}$ *and* $k_p(\boldsymbol{\theta}, \boldsymbol{\theta}')$ *is scalar-valued. Conditioning both Gaussian processes on a set of training points* $\mathbf{g}(\Theta) = \{\mathbf{g}(\boldsymbol{\theta}_i)\}_{i=1}^{N}$, *denote the corresponding conditional Gaussian processes by* $\mathbf{g}^N(\boldsymbol{\theta}) \sim GP(\mathbf{m}_N^{\mathbf{g}}(\boldsymbol{\theta}), K_N(\boldsymbol{\theta}, \boldsymbol{\theta}'))$ *and* $\mathbf{g}_s^N(\boldsymbol{\theta}) \sim GP(\mathbf{m}_{N,s}^{\mathbf{g}}(\boldsymbol{\theta}), K_{N,s}(\boldsymbol{\theta}, \boldsymbol{\theta}'))$, *respectively. Then we have,*

$$\begin{aligned}
\mathbf{m}_{N,s}^{\mathbf{g}}(\boldsymbol{\theta}) &= \mathbf{m}_N^{\mathbf{g}}(\boldsymbol{\theta}), \\
K_N(\boldsymbol{\theta}, \boldsymbol{\theta}') &= k_{N,p}(\boldsymbol{\theta}, \boldsymbol{\theta}')I_{d_{\mathbf{y}}}, \\
K_{N,s}(\boldsymbol{\theta}, \boldsymbol{\theta}') &= k_{N,p}(\boldsymbol{\theta}, \boldsymbol{\theta}')K_s(X, X),
\end{aligned}$$

where $k_{N,p}(\boldsymbol{\theta}, \boldsymbol{\theta}')$ *is scalar-valued.*

Proof. Let $k_p(\boldsymbol{\theta}, \Theta) := [k_p(\boldsymbol{\theta}, \boldsymbol{\theta}^1); \ldots; k_p(\boldsymbol{\theta}, \boldsymbol{\theta}^N)] \in \mathbb{R}^{d_{\mathbf{y}}}$, and denote by $K_p(\Theta, \Theta) \in \mathbb{R}^{d_{\mathbf{y}} \times d_{\mathbf{y}}}$ the matrix with entries $(K_p(\Theta, \Theta))_{i,j} = k_p(\boldsymbol{\theta}^i, \boldsymbol{\theta}^j)$. Then by (1.8) we have

$$\mathbf{m}_{N,s}^{\mathbf{g}}(\boldsymbol{\theta}) = \mathbf{m}(\boldsymbol{\theta}) + \left(k_p(\boldsymbol{\theta}, \Theta) \otimes K_s(X, X)\right)^T \left(K_p(\Theta, \Theta) \otimes K_s(X, X)\right)^{-1} \left(\mathbf{g}(\Theta) - \mathbf{m}(\Theta)\right),$$

where $\otimes$ denotes the Kronecker product. Using properties of products and inverses of Kronecker products and the fact that $K_s(X, X)$ is symmetric positive definite, we then have

$$\begin{aligned}
&\mathbf{m}_{N,s}^{\mathbf{g}}(\boldsymbol{\theta}) \\
&= \mathbf{m}(\boldsymbol{\theta}) + \left(k_p(\boldsymbol{\theta}, \Theta)^T \otimes K_s(X, X)^T\right) \left(K_p(\Theta, \Theta)^{-1} \otimes K_s(X, X)^{-1}\right) \left(\mathbf{g}(\Theta) - \mathbf{m}(\Theta)\right) \\
&= \mathbf{m}(\boldsymbol{\theta}) + \left(k_p(\boldsymbol{\theta}, \Theta)^T K_p(\Theta, \Theta)^{-1} \otimes K_s(X, X)^T K_s(X, X)^{-1}\right) \left(\mathbf{g}(\Theta) - \mathbf{m}(\Theta)\right) \\
&= \mathbf{m}(\boldsymbol{\theta}) + \left(k_p(\boldsymbol{\theta}, \Theta)^T K_p(\Theta, \Theta)^{-1} \otimes I_{d_{\mathbf{y}}}\right) \left(\mathbf{g}(\Theta) - \mathbf{m}(\Theta)\right) \\
&= \mathbf{m}_N^{\mathbf{g}}(\boldsymbol{\theta}).
\end{aligned}$$

The relationship between $K_{N,s}(\boldsymbol{\theta}, \boldsymbol{\theta}')$ and $K_N(\boldsymbol{\theta}, \boldsymbol{\theta}')$ can be shown in a similar way. □

For the PDE-constrained model, since the covariance functions related to f are obtained by applying the differential operator, the spatially correlated matrix in the joint prior (3.11) also depends explicitly on the parameters $\boldsymbol{\theta}$. Therefore, its covariance matrix cannot be written in a Kronecker product structure as in (3.13) and Theorem 4 does not apply. Thus, incorporating the PDE constraints into the model also affects the predictive mean and hence the mean-based posterior is also changed.

3.3.3 MCMC algorithms

To extract information from the posterior, MCMC algorithms are powerful and popular tools [74, 11]. In this work, we will consider the Metropolis-Adjusted Langevin Algorithm (MALA) [75], which is a type of MCMC algorithm that uses gradient information to accelerate the convergence of the sampling chain. Central to the idea of MALA, we have

$$d\boldsymbol{\theta} = \nabla \log \pi(\boldsymbol{\theta}|\mathbf{y})dt + \sqrt{2}dW, \tag{3.14}$$

where W is a standard $d_{\boldsymbol{\theta}}$-dimensional Brownian motion. This is an Itô SDE, which is used to model the continuous evolution of $\boldsymbol{\theta}$ over time with a drift term $\nabla \log \pi(\boldsymbol{\theta}|\mathbf{y})$ and a diffusion term $\sqrt{2}\, dW$. Under mild conditions on the posterior π, (3.14) is ergodic and has π as its stationary distribution [74], so that the probability density function of $\boldsymbol{\theta}(t)$ tends to $\pi(\boldsymbol{\theta}|\mathbf{y})$ as $t \to \infty$.

In practice, (3.14) is discretised with a simple Euler-Maruyama method with a time step γ:

$$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n + \gamma \nabla \log \pi(\boldsymbol{\theta}|\mathbf{y}) + \sqrt{2\gamma}\xi_n, \tag{3.15}$$

with $\xi_n \sim \mathcal{N}(0, 1)$. Assuming that the dynamics of (3.15) remain ergodic the corresponding numerical invariant measure would not necessarily coincide with the posterior. To alleviate this bias, one needs to incorporate an accept-reject mechanism [78]. This gives rise to MALA as described in Algorithm 2. Note

that in the MCMC algorithm, we use an adaptive time-stepping scheme[3]. This adaptive scheme adjusts the time step based on how the observed acceptance rate compares to the desired α_{opt}. The time step is increased when the acceptance rate is larger than α_{opt}, and conversely, the time step is decreased when the acceptance rate is small.

Algorithm 2 Metropolis-Adjusted Langevin Algorithm

Require: initial value $\boldsymbol{\theta}_0$, initial acceptance rate $\alpha_0 = 0$, number of samples N, initial time-step γ_0, posterior $\pi(\boldsymbol{\theta}|\mathbf{y})$, optimal acceptance rate α_{opt}

Ensure: Sequence of samples $\{\boldsymbol{\theta}_n\}_{n=0}^{N}$

while $n < N$ **do**

1. Generate $\xi_n \sim \mathcal{N}(0, 1)$.
2. Generate a candidate

$$\boldsymbol{\theta}' = \boldsymbol{\theta}_n + \gamma_n \nabla \log \pi(\boldsymbol{\theta}_n|\mathbf{y}) + \sqrt{2\gamma_n}\xi_n.$$

3. Compute the acceptance probability

$$\alpha := \max\left(1, \frac{\pi(\boldsymbol{\theta}'|\mathbf{y})q(\boldsymbol{\theta}_n|\boldsymbol{\theta}')}{\pi(\boldsymbol{\theta}_n|\mathbf{y})q(\boldsymbol{\theta}'|\boldsymbol{\theta}_n)}\right),$$

where $q(\boldsymbol{\theta}|\tilde{\boldsymbol{\theta}}) \propto \exp\left(-\frac{1}{4\gamma_n}\|\boldsymbol{\theta} - \tilde{\boldsymbol{\theta}} - \gamma_n \nabla \log \pi(\tilde{\boldsymbol{\theta}}|\mathbf{y})\|^2\right)$.

4. Generate $r \sim U[0, 1]$.

if $r \leq \alpha_n$ **then**

$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}'$

$\mathbb{I}_n = 1$

else

$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n$.

$\mathbb{I}_n = 0$

end if

5. Update the time-step $\gamma_{n+1} = \gamma_n(1 + \frac{\mathbb{I}_n - \alpha_{opt}}{n+1})$

end while

An advantage of using the Gaussian process emulator in the posterior is that, assuming the prior is differentiable, $\nabla \log \pi^N(\boldsymbol{\theta}|\mathbf{y})$ can be computed analytically for the mean-based and marginal approximations introduced in Section 3.2.2, which enables us to easily implement the MALA algorithm. Note that in contrast since the true posterior involves the (analytical or numerical) solution u to the PDE (3.1), it is usually impossible to compute these gradients analytically and one needs to resort to their numerical approximation. The following Lemma gives us the gradient of the different approximate posteriors.

Lemma 5. *Given the Gaussian process* $\mathcal{G}_X^N \sim GP(\mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta}), K_N(\boldsymbol{\theta}, \boldsymbol{\theta}))$ *that emulates the forward map* $\mathcal{G}_X$ *with data* $\mathcal{G}_X(\Theta)$*, we have the gradient of the mean-based approximation of the posterior*

$$\nabla \log(\pi_{\text{mean}}^{N,\mathcal{G}_X}(\boldsymbol{\theta}|\mathbf{y})) = -\frac{1}{\sigma_\eta^2}\nabla \mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta})^T(\mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta}) - \mathbf{y}) + \nabla \log \pi_0(\boldsymbol{\theta}),$$

and the gradient of the marginal approximation of the posterior

$$\begin{aligned}\nabla \log(\pi_{\text{marginal}}^{N,\mathcal{G}_X}(\boldsymbol{\theta}|\mathbf{y})) = &- \nabla \mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta})^T(K_N(\boldsymbol{\theta}, \boldsymbol{\theta}) + \Gamma_\eta)^{-1}(\mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta}) - \mathbf{y}) \\ &- \frac{1}{2}(\mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta}) - \mathbf{y})^T \nabla \left((K_N(\boldsymbol{\theta}, \boldsymbol{\theta}) + \Gamma_\eta)^{-1}\right)(\mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta}) - \mathbf{y}) \\ &- \frac{1}{2}\left(\text{Tr}\left((K_N(\boldsymbol{\theta}, \boldsymbol{\theta}) + \Gamma_\eta)^{-1}\right)\nabla(K_N(\boldsymbol{\theta}, \boldsymbol{\theta}))\right) + \nabla \log \pi_0(\boldsymbol{\theta}),\end{aligned}$$

where

$$\nabla\left((K_N(\boldsymbol{\theta}, \boldsymbol{\theta}) + \Gamma_\eta)^{-1}\right) = -(K_N(\boldsymbol{\theta}, \boldsymbol{\theta}) + \Gamma_\eta)^{-1}\nabla\left(K_N(\boldsymbol{\theta}, \boldsymbol{\theta})\right)(K_N(\boldsymbol{\theta}, \boldsymbol{\theta}) + \Gamma_\eta)^{-1},$$

and

$$\nabla K_N(\boldsymbol{\theta}, \boldsymbol{\theta}) = 2\nabla K(\boldsymbol{\theta}, \Theta)K(\Theta, \Theta)^{-1}K(\Theta, \boldsymbol{\theta})$$

Proof of Lemma 5. For the mean-based posterior:

$$\begin{aligned}&\nabla \log \pi_{\text{mean}}^{N,\mathcal{G}_X}(\boldsymbol{\theta}|\mathbf{y}) \\ &= \nabla \log\left(\exp\left(-\frac{1}{2\sigma_\eta^2}\|\mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta}) - \mathbf{y}\|^2\right)\right) \\ &= -\frac{1}{2\sigma_\eta^2}\nabla\left(\|\mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta}) - \mathbf{y}\|^2\right) \\ &= -\frac{1}{\sigma_\eta^2}\left(\nabla \mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta})\right)^T\left(\mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta}) - \mathbf{y}\right) \\ &= -\frac{1}{\sigma_\eta^2}\left(\nabla K(\boldsymbol{\theta}, \Theta)K(\Theta, \Theta)^{-1}\mathbf{y}\right)^T\left(\mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta}) - \mathbf{y}\right)\end{aligned}$$

For the marginal posterior:

$$\nabla \log \pi^{N,\mathcal{G}_X}_{\text{marginal}}(\boldsymbol{\theta}|\mathbf{y}) = \nabla \log \left(\frac{\exp\left(-\frac{1}{2}\|\mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta}) - \mathbf{y}\|^2_{(K_N(\boldsymbol{\theta},\boldsymbol{\theta})+\Gamma_\eta)}\right)}{\sqrt{(2\pi)^{d_\mathbf{y}} \det\left(K_N(\boldsymbol{\theta},\boldsymbol{\theta}) + \Gamma_\eta\right)}} \right)$$
$$= -\frac{1}{2}\nabla\left(\|\mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta}) - \mathbf{y}\|^2_{(K_N(\boldsymbol{\theta},\boldsymbol{\theta})+\Gamma_\eta)}\right) - \frac{1}{2}\nabla \log\left((2\pi)^n \det\left(K_N(\boldsymbol{\theta},\boldsymbol{\theta}) + \Gamma_\eta\right)\right)$$
$$= -(\nabla K(\boldsymbol{\theta},\Theta)K(\Theta,\Theta)^{-1}\mathbf{y})^T(K_N(\boldsymbol{\theta},\boldsymbol{\theta}) + \Gamma_\eta)^{-1}(\mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta}) - \mathbf{y})$$
$$- \frac{1}{2}(\mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta}) - \mathbf{y})^T\nabla\left((K_N(\boldsymbol{\theta},\boldsymbol{\theta}) + \Gamma_\eta)^{-1}\right)(\mathbf{m}_N^{\mathcal{G}_X}(\boldsymbol{\theta}) - \mathbf{y})$$
$$- \frac{1}{2}\left(\text{Tr}\left((K_N(\boldsymbol{\theta},\boldsymbol{\theta}) + \Gamma_\eta)^{-1}\right)\nabla(K_N(\boldsymbol{\theta},\boldsymbol{\theta}))\right),$$

where

$$\nabla\left((K_N(\boldsymbol{\theta},\boldsymbol{\theta}) + \Gamma_\eta)^{-1}\right) =$$
$$-(K_N(\boldsymbol{\theta},\boldsymbol{\theta}) + \Gamma_\eta)^{-1}\nabla\left(K_N(\boldsymbol{\theta},\boldsymbol{\theta})\right)(K_N(\boldsymbol{\theta},\boldsymbol{\theta}) + \Gamma_\eta)^{-1}$$
$$\text{and } \nabla K_N(\boldsymbol{\theta},\boldsymbol{\theta}) = 2\nabla K(\boldsymbol{\theta},\Theta)K(\Theta,\Theta)^{-1}K(\Theta,\boldsymbol{\theta}).$$

□

3.4 Numerical experiments

We now discuss a number of different numerical experiments related to inverse problems for linear PDES of the form (3.1) in various set-ups in terms of the number of spatial and parameter dimensions as well as for different types of forward models. A common theme in all our experiments is that the number of training points N is small, as this would be the case in large-scale applications in practice where increasing the number of training points is often either very costly or infeasible. The number of training points N used will serve as a benchmark for comparing different methodologies. Throughout all our numerical experiments in Sections 3.4.1-3.4.3 when comparing the different approaches we keep N fixed. The value of N is chosen in such a way to ensure that significant uncertainty remains present in the emulator, which is typically the case in applications. Alternatively, one could ask what number of training points for each model is needed to reach a certain accuracy, however as explained above, this is not the viewpoint

taken here. Precise timings for each of the approaches are reported in Section 3.4.4.

In cases where the PDE solution is not available in closed form, we use the finite element software Firedrake [72] to obtain the "true" solution. Furthermore, when using MALA we adaptively tune the step size to achieve an average acceptance probability 0.573 [11]. In all our numerical experiments we replace the uniform prior with a smooth approximation given by the $\lambda-$Moreau-Yoshida envelope [8] with $\lambda = 10^{-3}$ (see Figure 3.1). The parameter λ controls the level of smoothness in the approximation: smaller values of λ result in an approximation that more closely resembles the original non-smooth uniform prior, while larger values lead to a smoother, but less accurate, representation. We selected $\lambda = 10^{-3}$ as it provides a good approximation of the original prior and ensures numerical stability in computations.

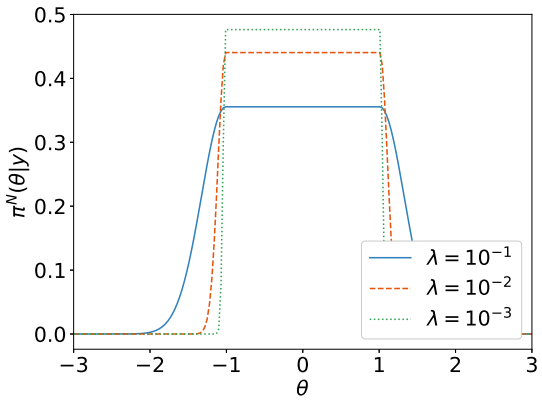

Figure 3.1: Pirors with different λ

The selection of hyperparameters is crucial for the application of Gaussian process regression. In this chapter, we optimise the hyperparameters by minimizing the negative log marginal likelihood (NLML). The expression for the NLML is given by:

$$-p(\mathbf{y}^*|\sigma^2, l) = \frac{1}{2}\mathbf{y}^{*T}K^{*-1}\mathbf{y}^* + \frac{1}{2}\log|K^*| + \frac{N^*}{2}\log 2\pi,$$

where K^* is the covariance matrix parameterized by the hyperparameters σ^2 and l, $\mathbf{y}^*$ is the observational data, N^* is the total number of observations, and $|K^*|$ denotes the determinant of the covariance matrix. For the baseline model,

$K^* = K(\Theta, \Theta)$, $\mathbf{y}^* = \mathcal{G}_X(\Theta)$ and $N^* = Nd_{\mathbf{y}}$. For the spatial correlation model, $K^* = K_{uu}(\Theta, \Theta)$, $\mathbf{y}^*$ and N^* are the same as the baseline model. For the PDE-constrained model, $K^* = \bar{K}(\Theta, \Theta)$, $\mathbf{y}^* = \mathbf{g}(\Theta)$ and $N^* = N(d_{\mathbf{y}} + d_f + d_g)$. The minimization problem can thus be formulated as:

$$\sigma^*, l^* = \arg\min_{\sigma,l} -p(\mathbf{y}^*|\sigma^2, l),$$

where σ^*, l^* represents the set of optimal hyperparameters that minimize the negative log marginal likelihood. The optimisation method used is L-BFGS. This optimization can be computationally intensive, particularly when the covariance matrix K^* is large, as it involves matrix inversion and determinant computation. We simplify this process by assuming isotropy for the length-scale in the covariance function for $\boldsymbol{\theta}$ as in (1.6) and (1.7), so the optimisation of the hyperparameters becomes a two-dimensional problem. Additionally, since we are operating in a small training data regime, the computational cost of evaluating the log-likelihood is small. To emphasise the improvement brought by the structure and also for simplicity, in the case of the spatially correlated and the PDE-constrained models, we use the same hyperparameters for the covariance function of the unknown parameter $\boldsymbol{\theta}$ as in the baseline model and only optimise the hyperparameters of the spatial covariance function. In principle, these assumptions can be relaxed to achieve potentially higher accuracy in the regression. The computational timings for optimisation of the hyperparameters can be found in Section 3.4.4.

To clarify the notation we use in our numerical experiments, we recall some of it again in Table 3.1.

Table 3.1: Symbols and notations used in numerical experiments.

Symbol	Description
$\mathcal{G}_X(\Theta)$	Training data set: point-wise evaluation of the PDE solution $u(\boldsymbol{\theta}, \mathbf{x})$ for $\mathbf{x} \in X = \{\mathbf{x}_i\}_{i=1}^{d_\mathbf{y}}$, $\boldsymbol{\theta} \in \Theta = \{\boldsymbol{\theta}^i\}_{i=1}^{N}$
$g(\Theta_g, X_g)$	Additional training data for boundary condition: point-wise evaluation of the function $g(\boldsymbol{\theta}, \mathbf{x})$ for $\mathbf{x} \in X_g = \{\mathbf{x}_i\}_{i=1}^{d_g}$, $\boldsymbol{\theta} \in \Theta = \{\boldsymbol{\theta}^i\}_{i=1}^{N_g}$
$f(\Theta_f, X_f)$	Additional training data for the source function: point-wise evaluation of the function $f(\boldsymbol{\theta}, \mathbf{x})$ for $\mathbf{x} \in X = \{\mathbf{x}_i\}_{i=1}^{d_f}$, $\boldsymbol{\theta} \in \Theta = \{\boldsymbol{\theta}^i\}_{i=1}^{N_f}$
$\bar{N}$	We use $N_g = N_f = \bar{N}$

3.4.1 Examples in one spatial dimension

Constant diffusion coefficient

We consider the following PDE in one spatial dimension

$$-\frac{\mathrm{d}}{\mathrm{d}x}\left(e^{\theta}\frac{\mathrm{d}u(x)}{\mathrm{d}x}\right) = 1, \qquad x \in (0,1),\, \theta \in [-1,1], \tag{3.16}$$
$$u(0) = 0, u(1) = 0.$$

In this case, the dimension of the parameter space is $d_{\boldsymbol{\theta}} = 1$, and the solution is available in closed form. More precisely, we have

$$u(x) = \frac{(x - x^2)}{2e^{\theta}}.$$

Given this explicit solution and the low dimension of the parameter space, we calculate the true and approximate posteriors on a fine grid without having to resort to Markov Chain Monte Carlo sampling. We now generate our observations $\mathbf{y}$ according to equation (3.2) for the value of $\boldsymbol{\theta}^\dagger = 0.314$ for a varying number of spatial points $d_\mathbf{y}$ (equally spaced in $[0,1]$) and for noise level $\sigma_\eta^2 = 10^{-5}$. The true posterior is not necessarily centred on the true parameter value $\theta^\dagger$ due to the presence of noise in the observations and the nature of the data being only point-wise measurements. As we can see in Figure 3.2 as we increase $d_\mathbf{y}$ the true posterior $\pi(\theta|\mathbf{y})$ tends to get more and more concentrated around the value of $\boldsymbol{\theta}^\dagger$

which is consistent with what the theory would predict by a Bernstein-von-Mises theorem (see e.g. [36] for related results).

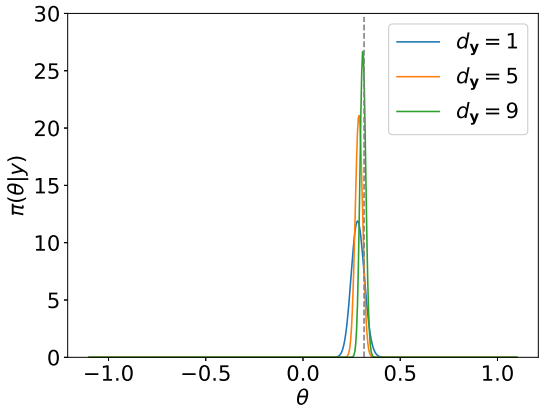

Figure 3.2: True posterior with different $d_{\mathbf{y}}$. The grey line denotes the true posterior and the grey dashed line denotes the true value $\boldsymbol{\theta}^{\dagger}$.

We now turn our attention to the different approximate posteriors discussed in Section 2.3 obtained for different Gaussian priors (independent, spatially correlated, and PDE-constrained). The distance between the approximated posterior $\pi^*(\boldsymbol{\theta}|\mathbf{y})$ and the true posterior $\pi(\boldsymbol{\theta}|\mathbf{y})$ is measured by the Hellinger distance [42]. The infinite-dimensional Hellinger distance $H(\pi, \pi^*)$ is defined as:

$$H(\pi, \pi^*) = \sqrt{\frac{1}{2}\int_{\mathcal{T}} \left(\sqrt{\pi(\boldsymbol{\theta}|\mathbf{y})} - \sqrt{\pi^*(\boldsymbol{\theta}|\mathbf{y})}\right)^2 d\boldsymbol{\theta}},$$

where Θ is the parameter space. To approximate this integral numerically, we used the trapezoidal rule over a discrete grid of points $\{\boldsymbol{\theta}_i\}_{i=1}^N$ in $\mathcal{T}$, giving the following approximation:

$$H(\pi, \pi^*) \approx \sqrt{\frac{1}{2}\sum_{i=1}^{N-1} \left(\sqrt{\pi(\boldsymbol{\theta}_i|\mathbf{y})} - \sqrt{\pi^*(\boldsymbol{\theta}_i|\mathbf{y})}\right)^2 \Delta\boldsymbol{\theta}_i},$$

where $\Delta\boldsymbol{\theta}_i$ represents the spacing between consecutive grid points.

Baseline model: In the case of the simplest emulator with independent entries, we illustrate in Figure 3.3, how the mean-based posterior $\pi_{\text{mean}}^{N,\mathcal{G}_X}(\theta|\mathbf{y})$ and the marginal posterior $\pi_{\text{marginal}}^{N,\mathcal{G}_X}(\theta|\mathbf{y})$ behave as a function of the number of training

points N (here $d_{\mathbf{y}} = 5$). The location of the training points is chosen from the Halton sequence [60]. Now, when comparing Figure 3.3 (top left and top right) we see that the marginal posterior is more spread than the mean-based posterior. This is due to the variance inflation associated with the marginal posterior which reflects better the uncertainty of the emulator. For example, in the case $N = 1$ the mean-based posterior has negligible posterior probability mass near $\boldsymbol{\theta}^\dagger$ and exhibits bimodality since it so happens that the training point used is not near the true $\boldsymbol{\theta}^\dagger$. However, due to the variance inflation this is not the case for the marginal approximation. Furthermore, in Figure 3.3 (bottom left) we plot the Hellinger distance between the approximate posteriors and the true posterior as a function of the number of training points N. As we can see the error for the marginal approximation is slightly smaller than the error for the mean-based posterior for small N. The two errors are equal as N increases, which can be further understood by Figure 3.3 (bottom right). Here, we plot the average variance $k_N(\boldsymbol{\theta}, \boldsymbol{\theta})$ (averaged over $\boldsymbol{\theta}$) of our emulator for different values of N, and see that as expected from (3.6) the marginal approximation behaves in the same manner as the mean-based approximation once the average variance is of the same order as the observational noise σ_η^2.

Spatially correlated model: As discussed in Section 3.3.1 the introduction of spatial correlation doesn't change the predictive mean of the Gaussian processes. We hence now compare in Figure 3.4 the two different marginal posteriors $\pi^{N,\mathcal{G}_X}_{\text{marginal}}$, $\pi^{N,\mathcal{G}_X,s}_{\text{marginal}}$, where the latter includes spatial correlation. We again choose $d_{\mathbf{y}} = 5$. In particular, as we can see in Figure 3.4 (left) (here $N = 2$), introducing spatial correlation seems to improve the accuracy of the approximate posterior and place more mass near $\theta^\dagger$. The fact that the spatially correlated model has an increased variance at $N = 2$ (see Figure 3.4) leads to similar behaviour as in Figure 3.3 with $\pi^{N,\mathcal{G}_X,s}_{\text{marginal}}$ being more spread than $\pi^{N,\mathcal{G}_X}_{\text{marginal}}$. Furthermore, as we can see in Figure 3.4 (middle) as we increase the number of training points for our Gaussian process the Hellinger distance between the true posterior and $\pi^{N,\mathcal{G}_X,s}_{\text{marginal}}$ is smaller than the one of the baseline model.

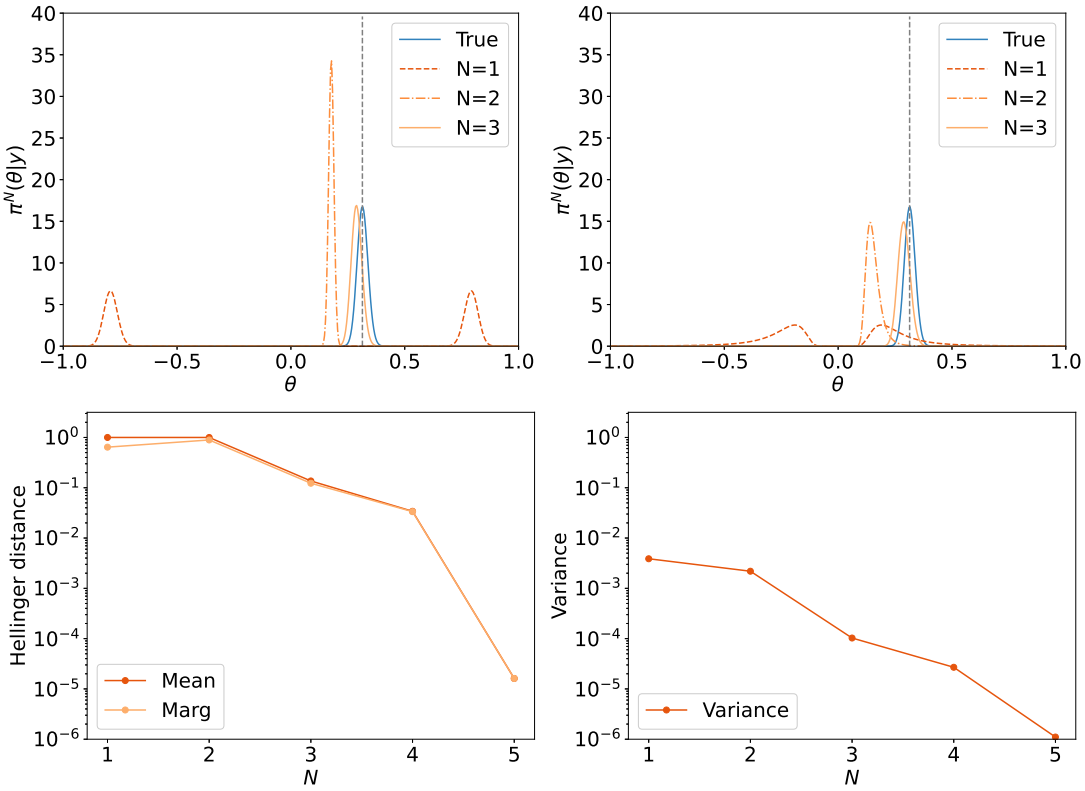

Figure 3.3: Baseline model mean-based posterior (*top left plot*) and marginal posterior (*top right plot*) with different N. Hellinger distance between approximate posteriors and true posteriors (*bottom left plot*) and average predictive variance of the Gaussian process emulator (*bottom right plot*) as N increases. $\mathcal{G}_X$ is the pointwise observation of u in (3.16). The grey line denotes the true posterior and the grey dashed line denotes the true value $\boldsymbol{\theta}^\dagger$.

PDE-constrained model: We now compare the behaviour of the PDE-constrained model with the other two models, both for mean-based approximate posterior, as well as for marginal posterior (again here $d_{\mathbf{y}} = 5$). In particular, as we can see in Figures 3.5 for $N = 2$, $\pi^{N,\mathcal{G}_X,\text{PDE}}_{\text{mean}}$ and $\pi^{N,\mathcal{G}_X,\text{PDE}}_{\text{marginal}}$ are indistinguishable from the true posterior when using $\bar{N} = 10$, $d_f = 5$ showing much better approximation properties than the other two models. This is consistent with what we observe in terms of Hellinger distance, since both $\pi^{N,\mathcal{G}_X,\text{PDE}}_{\text{mean}}$ and $\pi^{N,\mathcal{G}_X,\text{PDE}}_{\text{marginal}}$ have similar errors over a different range of values for d_f. It is also worth noting that when comparing with the Hellinger distance from Figures 3.3 (bottom left) and 3.4 (middle) we see that the PDE-based model achieves the same order of error with only using half of the training points ($N = 2$ instead of $N = 4$). Furthermore, as we can see in Figure 3.5 (bottom right) the average variance of the PDE-constrained emulator converges to zero very fast as the number of extra training points for f increases, implying that at least in this simple example adding the PDE knowledge leads to an extremely good approximation of the forward map.

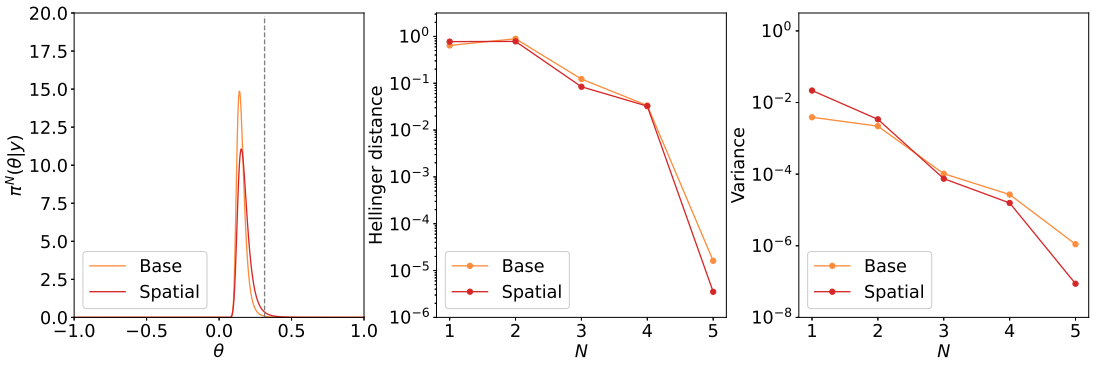

Figure 3.4: Baseline and spatially correlated model marginal posterior for $N = 2$ (*left plot*). Hellinger distance between approximate posteriors and true posterior (*middle plot*) and average predictive variance of the Gaussian process emulator (*right plot*) as N increases. $\mathcal{G}_X$ is the pointwise observation of u in (3.16). The grey line denotes the true posterior and the grey dashed line denotes the true value $\boldsymbol{\theta}^\dagger$.

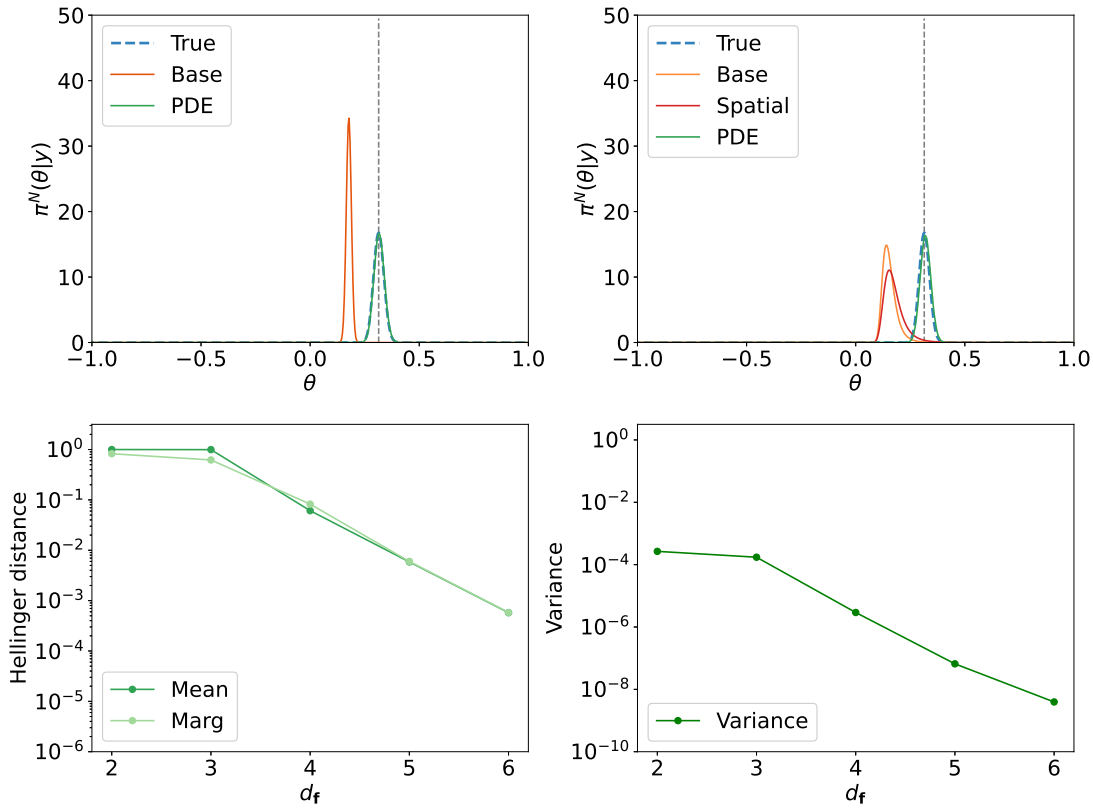

Figure 3.5: Comparison of different models when $N = 2$, for PDE model $d_f = 5$: mean-based posteriors (*top left plot*) and marginal posteriors (*top right plot*). Hellinger distance between approximate posteriors and true posterior (*bottom left plot*) and average predictive variance of the emulator (*bottom right plot*) as d_f increases. $\mathcal{G}_X$ is the pointwise observation of u in (3.16). The grey line denotes the true posterior and the grey dashed line denotes the true value $\boldsymbol{\theta}^\dagger$.

Two-dimensional piece-wise constant diffusion coefficient

We consider an elliptic equation with a 2-dimensional piece-wise constant diffusion coefficient; we have the following equation

$$\begin{aligned} -\frac{\mathrm{d}}{\mathrm{d}x}(\exp(\kappa(x,\boldsymbol{\theta}))\frac{\mathrm{d}}{\mathrm{d}x}u(x)) &= 4x, \\ x \in (0,1), \quad \boldsymbol{\theta} &\in [-1,1]^2, \\ u(0) = 0, \qquad u(1) &= 2, \end{aligned} \tag{3.17}$$

where κ is defined as a piece-wise constant over four equally spaced intervals. More precisely, we consider

$$\kappa(x,\boldsymbol{\theta}) = \begin{cases} 0, & \text{for } x \in [0,\frac{1}{4}) \\ \theta_1, & \text{for } x \in [\frac{1}{4},\frac{1}{2}) \\ \theta_2, & \text{for } x \in [\frac{1}{2},\frac{3}{4}) \\ 1 & \text{for } x \in [\frac{3}{4},1] \end{cases} \tag{3.18}$$

Since it is not possible to solve (3.17) explicitly, we use Firedrake to obtain its solution.

Throughout this numerical experiment, we take the prior of the parameters to be the uniform distribution on $[-1,1]^2$, and we generate our data $\mathbf{y}$ according to equation (3.2) for $\boldsymbol{\theta}^\dagger = [0.098, 0.430]$, $d_{\mathbf{y}} = 6$ (equally spaced points in [0,1]) and noise level $\sigma_\eta^2 = 10^{-4}$. For the covariance kernels, we choose k_p to be the squared exponential kernel and k_s to be the Matèrn kernel with $\nu = \frac{5}{2}$.

For the PDE-constrained model, we first test the effect of additional training data $g(\Theta_g, X_g)$ and $f(\Theta_f, X_f)$ on the accuracy of the emulator. In Figure 3.6, we see that as d_f and $\bar{N}$ increase, the accuracy of emulators gradually increases.

We now use MALA to obtain samples for all our approximate posteriors using 10^6 samples. For all models, we have used $N = 4$ training points (chosen to be the first 4 points in the Halton sequence), while additionally for the PDE-constrained model, we have used $\bar{N} = 10$ (chosen to be the next 10 points in the Halton sequence), $d_f = 20$ and $d_g = 2$. Since we do not have access to the true posterior, we consider the results obtained from a mean-based approximation

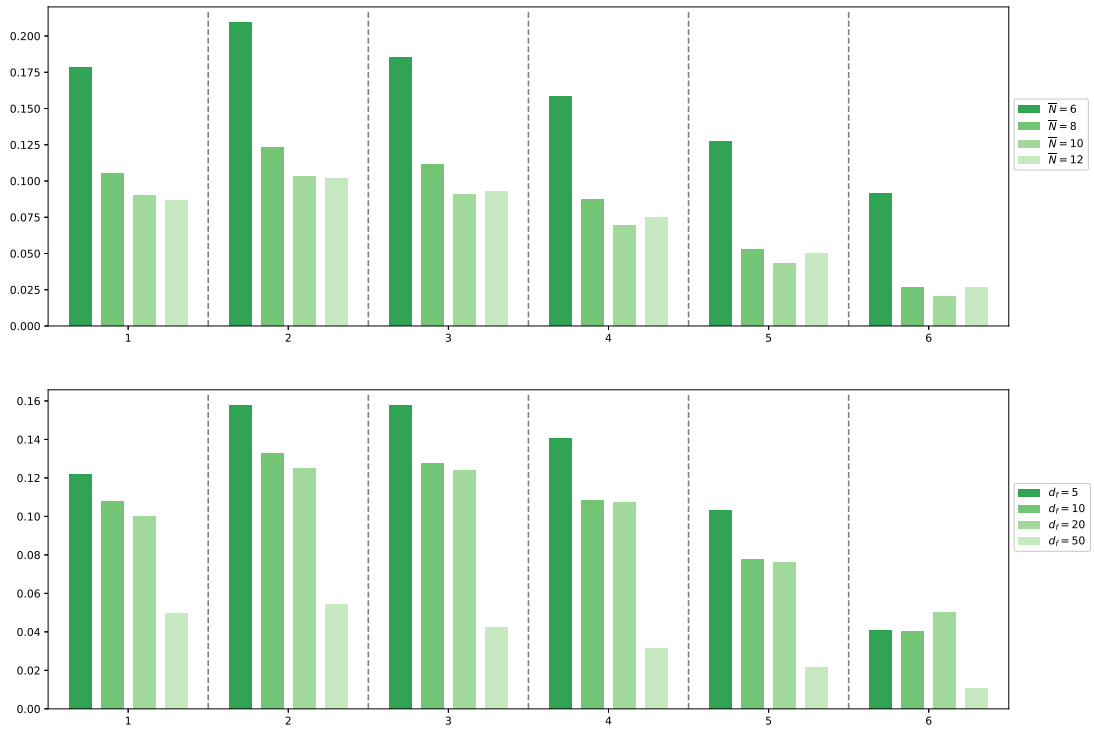

Figure 3.6: Absolute error between the predictive mean of the PDE-constrained emulator and the ground truth ($\boldsymbol{\theta} = \boldsymbol{\theta}^{\dagger}$) at the $d_{\mathbf{y}} = 6$ observation points. The x-axis represents the index of the 6 observation points, and the y-axis shows the absolute error between the predictive mean and the ground truth. The *top plot* shows results for different $\bar{N}$ values with $d_f = 20$ fixed, while the *bottom plot* displays results for different d_f values with $\bar{N} = 10$ fixed.

with the baseline model for $N = 10^2$ training points as the ground truth.

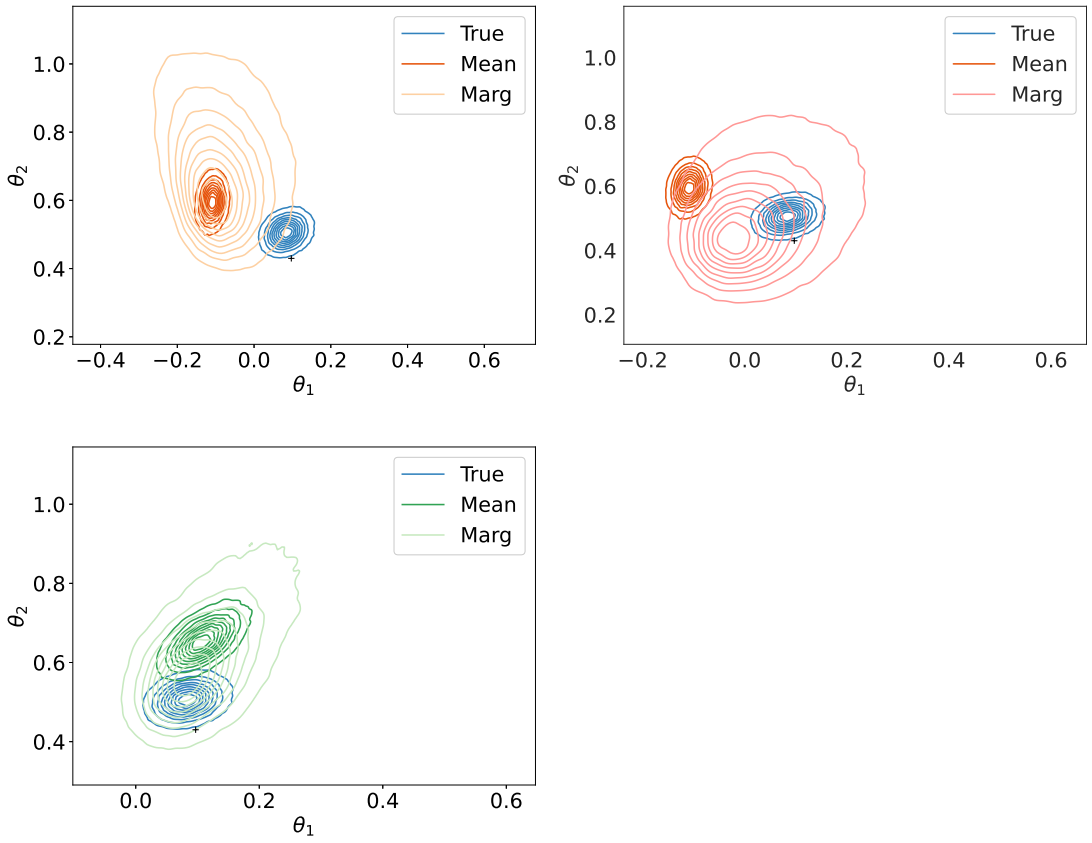

Figure 3.7: Contour plots of the approximate mean-based and marginal posteriors: baseline model (*top left plot*), spatially correlated (*top right plot*), PDE-constrained (*bottom plot*). The symbol "+" denotes $\boldsymbol{\theta}^\dagger$. $\mathcal{G}_X$ is the pointwise observation of u in (3.17).

As we can see in Figure 3.7, all the mean-based posteriors fail to put significant posterior mass near the true parameter value $\boldsymbol{\theta}^\dagger$. The situation improves when the uncertainty of the emulator is taken into account as we can see for the marginal approximations. Out of the three different models, the PDE-constrained one seems to be performing best since it is placing the most posterior mass around the true value $\boldsymbol{\theta}^\dagger$. This is further illustrated in Figure 3.8 where we plot the θ_1 and θ_2 marginals for all the mean-based posterior approximations $\pi_{\text{mean}}^{N,\mathcal{G}_X}$, $\pi_{\text{mean}}^{N,\mathcal{G}_X,s}$, $\pi_{\text{mean}}^{N,\mathcal{G}_X,\text{PDE}}$ and the marginal posterior approximations $\pi_{\text{marginal}}^{N,\mathcal{G}_X}$, $\pi_{\text{marginal}}^{N,\mathcal{G}_X,s}$, $\pi_{\text{marginal}}^{N,\mathcal{G}_X,\text{PDE}}$. Note that the marginal plot could be misleading regarding the overall performance of the approximations, for example in Figure 3.8 (top right) the baseline model seems to be better than the PDE-constrained model, but from Figure 3.7 we know that this is not true. In other words, the marginal

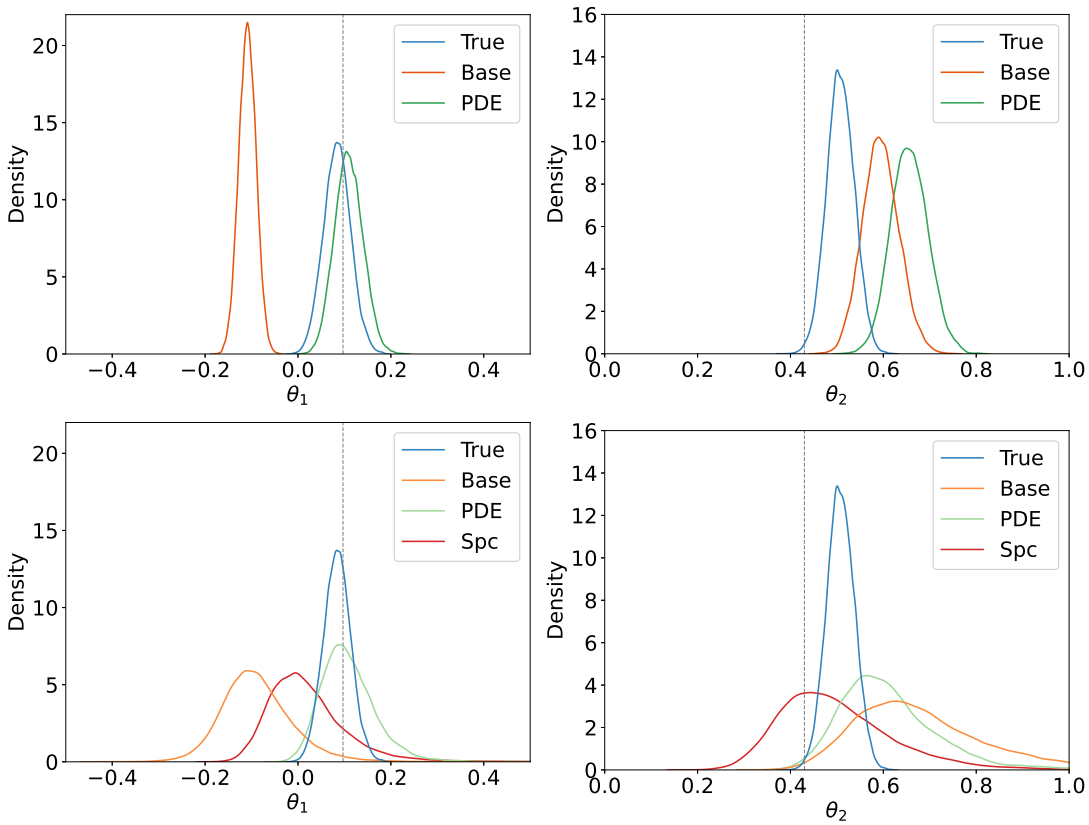

Figure 3.8: Comparison of different models when $N = 4$, for PDE model $d_f = 20$ and $\bar{N} = 20$: mean-based approximation of the θ_1 marginal (*top left plot*) and θ_2 marginal (*top right plot*), marginal approximation of the θ_1 marginal (*bottom left plot*) and θ_2 marginal (*bottom right plot*). $\mathcal{G}_X$ is the pointwise observation of u in (3.17) with diffusion coefficient (3.18). The grey line denotes the true posterior and the grey dashed line denotes the true value $\boldsymbol{\theta}^\dagger$.

posteriors are better approximations than the joint posterior. When we increase d_f from 20 to 50, the accuracy of the approximation improves as we can see in Figure 3.9 where we compare PDE-constrained approximations for the two different values of d_f.

Integral observation operator

We now investigate the proposed method with a different form of observation operator. In terms of the PDE problem, we study again (3.17). However, instead of point-wise observations $\mathcal{G}_X(\boldsymbol{\theta}) = \{u(x_j;\boldsymbol{\theta})\}_{j=1}^{d_\mathbf{y}}$ as in (3.2), we observe local averages $\mathcal{G}_X(\boldsymbol{\theta}) = \{\int_{a_j}^{b_j} u(x;\boldsymbol{\theta})dx\}_{j=1}^{d_\mathbf{y}}$ for non-overlapping intervals $[a_j, b_j] \subset [0, 1]$.

For the inverse problem setting, we have $\boldsymbol{\theta}^\dagger = [0.098, 0.430]$ which is the same

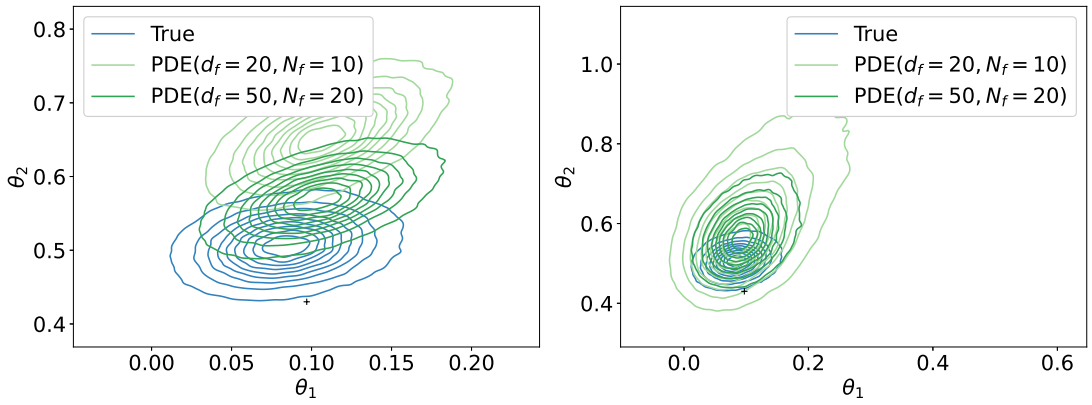

Figure 3.9: Comparison of different models when $N = 4$: mean-based approximation (*left plot*) and marginal approximation (*right plot*).

as before, $d_{\mathbf{y}} = 16$ (equally spaced sub-intervals of $[0, 1]$) and $\sigma_\eta^2 = 10^{-6}$. We again do not conduct precise integration as in (3.16), but use the MALA algorithm to obtain our samples. We utilize 10^6 samples for all our approximate posterior. We treat the sampling results obtained by a mean-based approximation with the baseline model for $N = 10^2$ training points as the ground truth. In Figure 3.10, we plot again the θ_1 and θ_2 marginals for all the mean-based posterior approximations and the marginal posterior approximations. The result is similar to the example in Section 3.4.2 that the PDE-constrained model performs better than the other two models.

Parametric expansion for the diffusion coefficient

In this example, we study again (3.17), but this time instead of working with a piece-wise constant diffusion coefficient we assume that the diffusion coefficient satisfies the following parametric expansion

$$\kappa(\boldsymbol{\theta}, x) = \sum_{n=1}^{d_{\boldsymbol{\theta}}} \sqrt{a_n}\theta_n b_n(x) \tag{3.19}$$

where $a_n = \frac{8}{\omega_n^2+16}$, $b_n(x) = A_n(\sin(\omega_n x) + \frac{\omega_n}{4}\cos(\omega_n x))$, ω_n is the n_{th} solution of the equation $\tan(\omega_n) = \frac{8\omega_n}{\omega_n^2-16}$ and A_n is a normalisation constant which makes $\|b_n\|_{L^2(0,1)} = 1$. This choice is motivated by the fact that for $\{\theta_n\}_{n=1}^{d_{\boldsymbol{\theta}}}$ i.i.d. standard normal random variables, this is a truncated Karhunen-Loève ex-

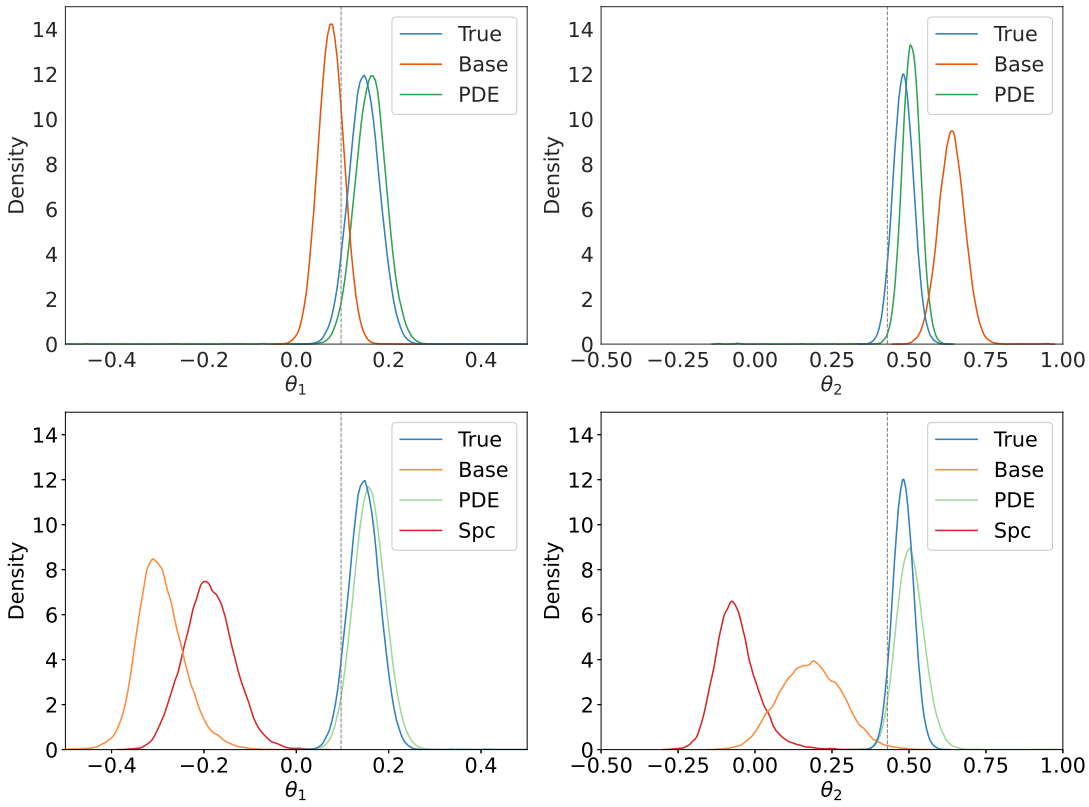

Figure 3.10: Comparison of different models' marginal distribution when $N = 4$, for PDE model $\bar{N} = 10$ and $d_f = 50$: mean-based approximation of the θ_1 marginal (*top left plot*) and θ_2 marginal (*top right plot*), marginal approximation of the θ_1 marginal (*bottom left plot*) and θ_2 marginal (*bottom right plot*). $\mathcal{G}_X$ is the integrals of solution u in (3.17) with diffusion coefficient (3.18). The grey line denotes the true posterior and the grey dashed line denotes the true value $\boldsymbol{\theta}^\dagger$.

pansion of $\log(\kappa(\boldsymbol{\theta}, x)) \sim \mathrm{GP}(0, \exp(-\|x - x'\|_1^2))$ [35]. The weights a_n and the functions $b_n(x)$ in the parametric expansion (3.19) are the eigenvalues and the normalised eigenfunctions of the covariance operator with the covariance function $\exp(-\|x - x'\|_1^2)$. In this work, we adopt this expansion to parametrize the diffusion coefficient $\kappa(\boldsymbol{\theta}, x)$, where θ_n are not random variables but unknown parameters to be inferred in the inverse problem.

In terms of the inverse problem setting, we are using the same parameters as before ($\boldsymbol{\theta}^\dagger = [0.098, 0.430]$, $d_{\mathbf{y}} = 6$, noise level $\sigma_\eta^2 = 10^{-4}$). The number of training points for all the emulators has been set to $N = 4$ (chosen using the Halton sequence), while in the case of the PDE-constrained emulator we have used $\bar{N} = 10$ and $d_f = 8$. Furthermore, throughout this numerical experiment, we take the prior of the parameters to be the uniform distribution on $[-1, 1]^2$. For the choices of kernels, we use the squared exponential kernel for both k_p and k_s.

As in the previous experiments, we produce 10^6 samples of the posteriors using MALA and use the results obtained by a mean-based approximation with the baseline model for $N = 10^2$ training points as the ground truth.

We now plot in Figure 3.11 the θ_1 and θ_2 marginals for the different Gaussian emulators both in the case of mean-based and marginal posterior approximations. As we can see in Figure 3.11 for the mean-based posterior approximations, the baseline and spatially correlated models fail to capture the true posterior while this is not the case for the PDE-constrained model since the agreement with the true posterior is excellent. When looking at the marginal approximations in Figure 3.11 (bottom left and bottom right) we can see that the marginals for the baseline and spatially correlated models move closer towards the true value $\boldsymbol{\theta}^\dagger$ and exhibit variance inflation. This is, however, not the case for the PDE-constrained model since again it is in excellent agreement with the true posterior.

Ten-dimensional parametric expansion diffusion coefficient

We will now increase the dimension of the diffusion coefficient from $d_{\boldsymbol{\theta}} = 2$ to $d_{\boldsymbol{\theta}} = 10$ in (3.19), to test the proposed method in a relatively high dimensional

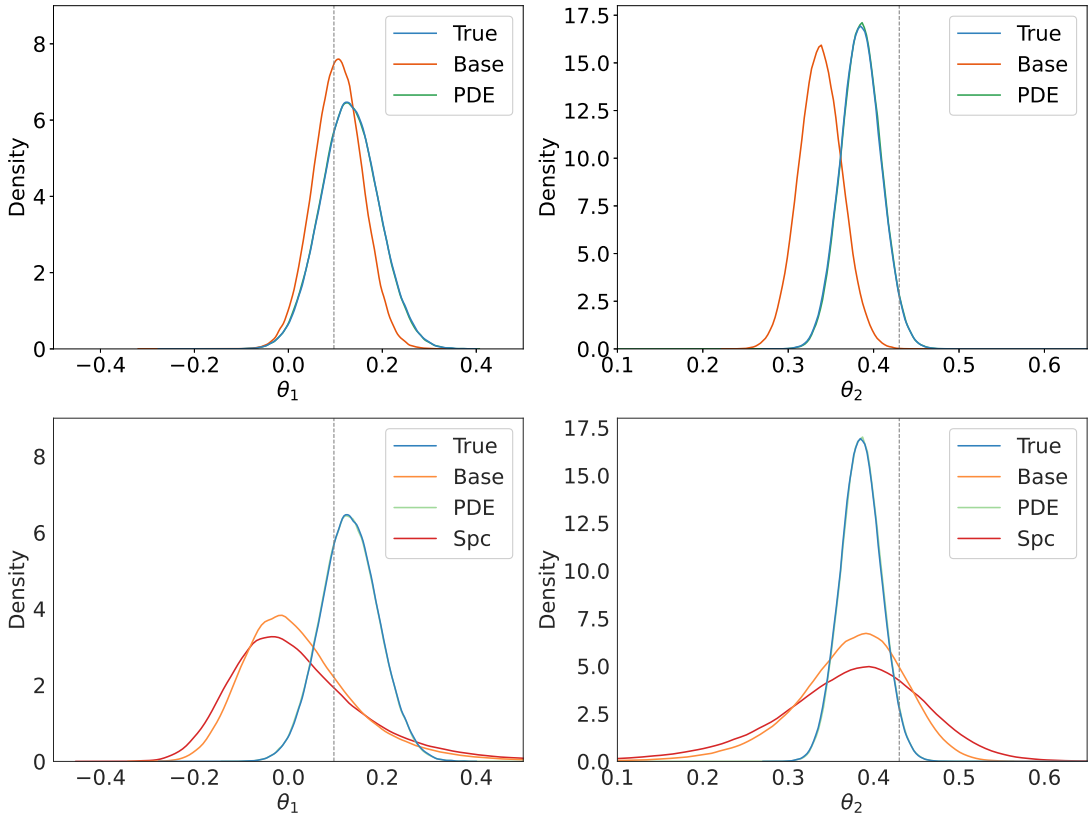

Figure 3.11: Comparison of different models' marginal distribution when $N = 4$, for PDE model $\bar{N} = 10$ and $d_f = 8$: mean-based approximation of the θ_1 marginal (*top left plot*) and θ_2 marginal (*top right plot*), marginal approximation of the θ_1 marginal (*bottom left plot*) and θ_2 marginal (*bottom right plot*). $\mathcal{G}_X$ is the pointwise observation of solution u in (3.17) with diffusion coefficient (3.19) and $d_{\boldsymbol{\theta}} = 2$. The grey line denotes the true posterior and the grey dashed line denotes the true value $\boldsymbol{\theta}^\dagger$.

space. With regard to the inverse problem setting, we set

$$\boldsymbol{\theta}^\dagger = [0.098, 0.430, 0.206, 0.090, -0.153, 0.292, -0.125, 0.784, 0.927, -0.233]$$

and we increase the number of observation points to $d_{\mathbf{y}} = 20$. The level of noise is the same as before ($\sigma_\eta^2 = 10^{-4}$). The number of training points for all emulators is again set to be $N = 4$, and for the PDE-constrained emulator we use $\bar{N} = 50$, $d_f = 25$ and $d_g = 2$. For the choices of kernels, we use the squared exponential kernel for both k_p and k_s.

We now use the MALA algorithm to obtain 10^7 samples of the approximate posteriors. In this relatively high-dimensional setting, we need longer chains for the sampling algorithm to converge. Meanwhile, computation of a suitable "ground truth" is prohibitively expensive, so we only compare the sampling result with the true parameter $\boldsymbol{\theta}^\dagger$. The number of training points $N = 4$ is far from enough for the baseline Gaussian process model to give an accurate prediction. From Figure 3.6 (left), we can see that the mean-based posterior approximation with the baseline model can only give a reasonable approximation for the first few variables, for the rest of the variables the approximation could not put any density around the true value. Adding spatial correlation into the model helps the approximation move toward the true value (Figure 3.6 (right)), but it still cannot correctly approximate the posterior for the last few variables. The performance of the PDE-constrained model is much better than the other models, it places the posterior mass around the true value for all variables.

3.4.2 Two spatial dimensions

In this example, we increase the spatial dimension from $d_{\mathbf{x}} = 1$ to $d_{\mathbf{x}} = 2$ and use a 2-dimensional piece-wise constant as the diffusion coefficient. The values of the diffusion coefficient are set in a similar way to the first example, but depending

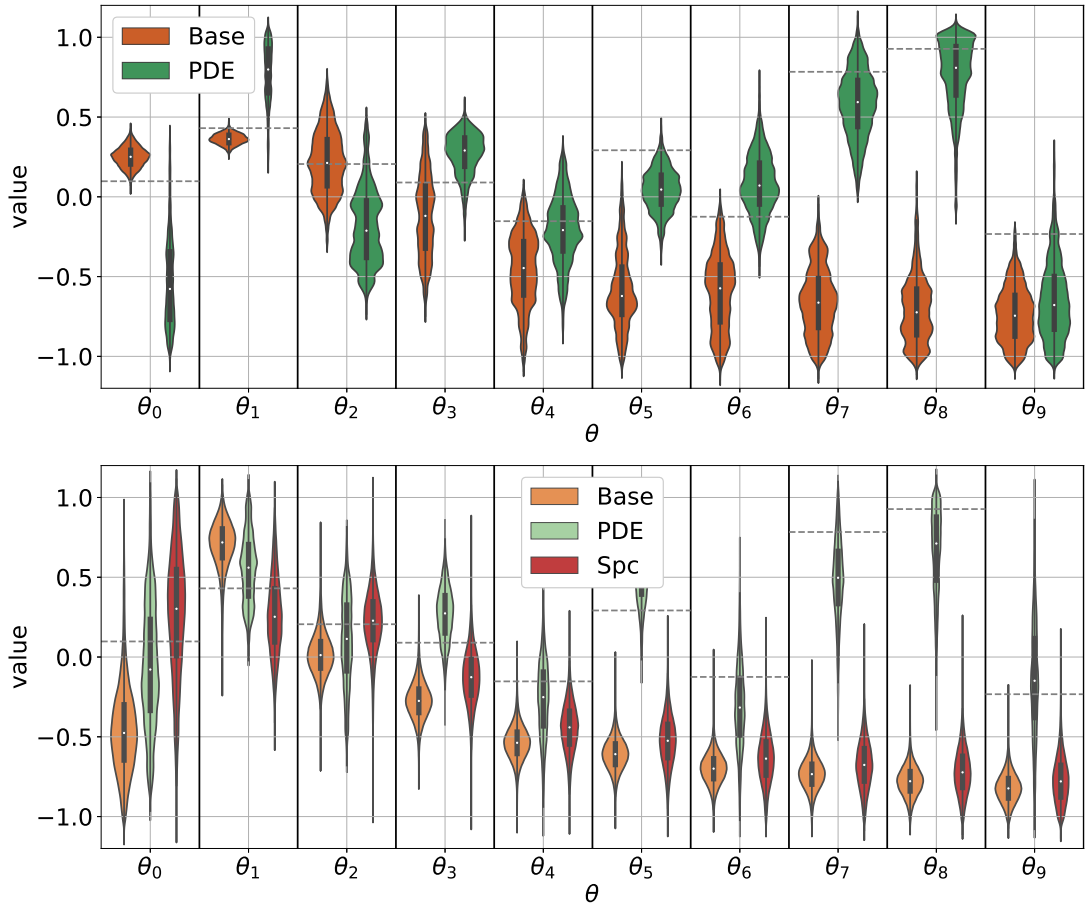

Figure 3.12: Comparison of different models' marginal distribution when $N = 4$, for PDE model $\bar{N} = 50$ and $d_f = 25$: mean-based approximation (*top plot*) and marginal approximation (*bottom plot*). $\mathcal{G}_X$ is the pointwise observation of solution u in (3.17) with diffusion coefficient (3.19) and $d_{\boldsymbol{\theta}} = 10$. The grey line denotes the true posterior and the grey dashed line denotes the true value $\boldsymbol{\theta}^\dagger$.

only on the first dimension of $\mathbf{x}$:

$$\kappa(\mathbf{x}, \boldsymbol{\theta}) = \begin{cases} 0, & \text{for } x_1 \in [0, \frac{1}{4}), \\ \theta_1, & \text{for } x_1 \in [\frac{1}{4}, \frac{1}{2}), \\ \theta_2, & \text{for } x_1 \in [\frac{1}{2}, \frac{3}{4}), \\ 1, & \text{for } x_1 \in [\frac{3}{4}, 1]. \end{cases} \tag{3.20}$$

The boundary conditions are a mixture of Neumann and Dirichlet conditions, given by

$$\begin{aligned} &\partial_{x_1} u(x_1, 0) = \partial_{x_1} u(x_1, 1) = 0, && \text{for } x_1 \in [0, 1], \\ &u(0, x_2) = 1, \quad u(1, x_2) = 0, && \text{for } x_2 \in [0, 1]. \end{aligned}$$

These boundary conditions define a *flow cell*, with no flux at the top and bottom boundary ($x_2 = 0, 1$) and flow from left to right induced by the higher value of u at $x_1 = 0$.

For the observation, we generate our data $\mathbf{y}$ according to equation (3.2) for $\boldsymbol{\theta}^\dagger = [0.098, 0.430]$ with $d_{\mathbf{y}} = 6$ (chosen to be the first 6 points in the Halton sequence) and a noise level $\sigma_\eta^2 = 10^{-5}$. In addition, for the baseline and spatially correlated models, we have used $N = 4$ training points (chosen to be the first 4 points in the Halton sequence), while additionally for the PDE-constrained model, we have used $\bar{N} = 30$, $d_f = 30$ and $d_g = 8$, corresponding to 2 equally spaced points on each boundary. For the covariance kernels, we let k_p be the squared exponential kernel and k_s be the Matèrn kernel with $\nu = \frac{5}{2}$.

We plot the mean-based approximate marginal posteriors in Figure 3.13 (top left and top right). We can see that in this case, the PDE-constrained model significantly improves the approximation accuracy, which is different from the previous piece-wise constant diffusion coefficient example in one spatial dimension. In Figures 3.13 (bottom left and bottom right), we compare the marginal approximation for the three models, and we see again that the PDE-constrained model performs best.

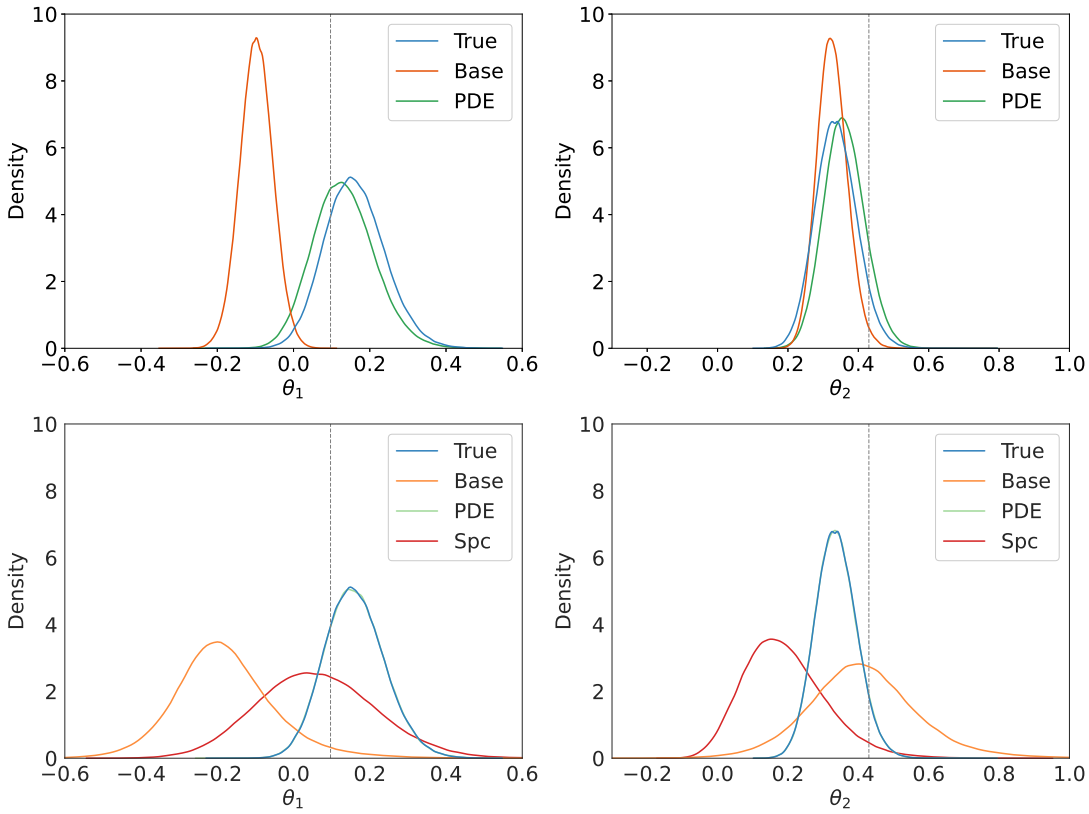

Figure 3.13: Comparison of different models' marginal distribution when $N = 4$, for PDE model $\bar{N} = 30$ and $d_f = 30$: mean-based approximation of the θ_1 marginal (*top left plot*) and the θ_2 marginal (*top right plot*), and marginal approximation of the θ_1 marginal (*bottom left plot*) and the θ_2 marginal (*bottom right plot*). $\mathcal{G}_X$ is the pointwise observation of u with $d_{\mathbf{x}} = 2$ and diffusion coefficient (3.20). The grey line denotes the true posterior and the grey dashed line denotes the true value $\boldsymbol{\theta}^\dagger$.

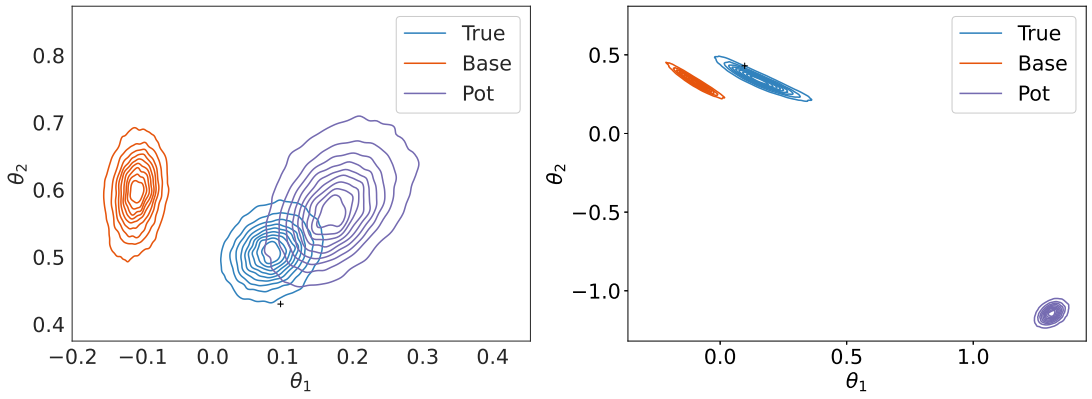

Figure 3.14: Comparison of emulating log-likelihood function and emulating observations when $N = 4$. Both approximations are mean-based approximations. $\mathcal{G}_X$ is the negative log-likelihood function in problem (3.17) with diffusion coefficient (3.18) with $d_{\mathbf{x}} = 1$ (*left plot*) and $d_{\mathbf{x}} = 2$ (*right plot*). "Pot" refers to the posterior distribution approximated using an emulated "potential" function.

3.4.3 Emulating the negative log-likelihood

As discussed in Section 3.2.2, we can emulate the negative log-likelihood directly with Gaussian process regression instead of emulating the forward map. Since emulation of the log-likelihood involves emulating a non-linear functional of the PDE solution u, we are not able to incorporate spatial correlation or PDE constraints in the same way. We test the performance of the mean-based approximation (3.7) and the marginal approximation (3.8) using the previous examples: problem (3.17) with diffusion coefficient (3.18) with $d_{\mathbf{x}} = 1$ and $d_{\mathbf{x}} = 2$. All parameters are kept the same as in Section 3.4.1 and Section 3.4.2.

In Figure 3.14, we compare the mean-based approximation with the emulation of the log-likelihood Φ and the observation operator $\mathcal{G}_X$ using the baseline model. We see that the results are very different in both examples. For the $d_{\mathbf{x}} = 1$, emulating the log-likelihood function performs better than the emulation of the observation with the baseline model, the approximate posterior is closer to the true posterior. For the $d_{\mathbf{x}} = 2$, its performance is much worse. Hence, emulating the log-likelihood with a small amount of data could be less reliable compared to emulating the forward map. If we increase the number of training data to $N = 10$ for the $d_{\mathbf{x}} = 2$ case, we can see an improvement of accuracy in Figure 3.15, but it is still worse than emulating the forward map with the baseline model.

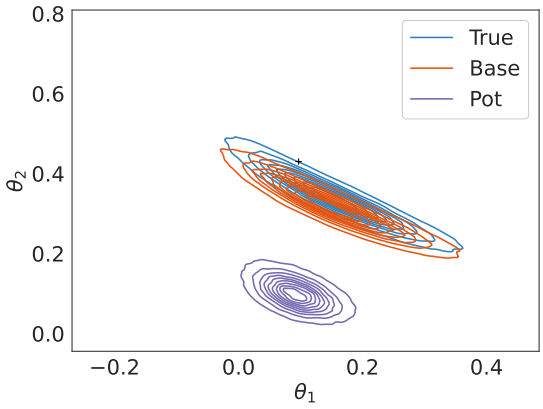

Figure 3.15: Accuracy of emulator is improved when N increases ($N = 10$). $\mathcal{G}_X$ is the negative log-likelihood function in problem (3.17) with diffusion coefficient (3.18) with $d_{\mathbf{x}} = 2$ and mean-based approximation. "Pot" refers to the posterior distribution approximated using an emulated "potential" function.

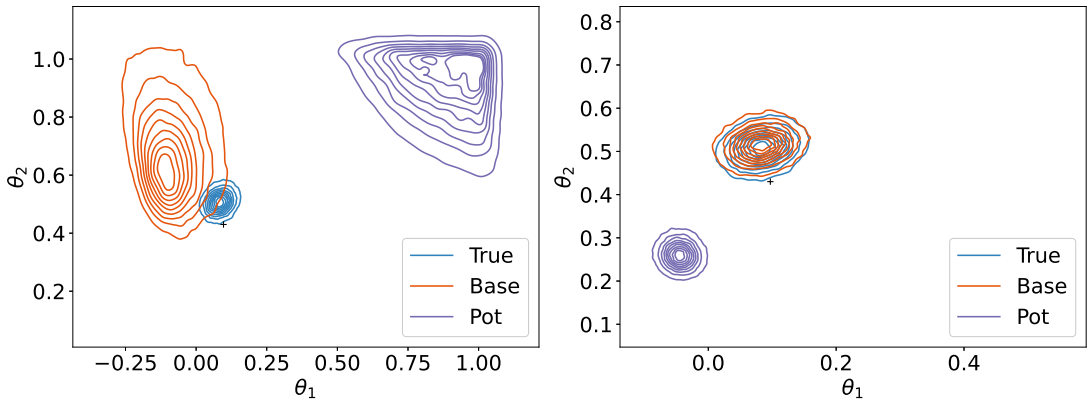

Figure 3.16: Marginal approximation with $N = 4$ (*left plot*) and marginal approximation with $N = 10$ (*right plot*). $\mathcal{G}_X$ is the negative log-likelihood function in problem (3.17) with diffusion coefficient (3.18) with $d_{\mathbf{x}} = 1$. "Pot" refers to the posterior distribution approximated using an emulated "potential" function.

Similarly, we see in Figure 3.16 that marginal approximations of the log-likelihood based emulation appear to be less reliable, but including more training points can again improve the performance.

3.4.4 Computational timings

In this section, we discuss computational timings. The computational timings reported in this section were obtained using a standard laptop equipped with an Intel Core i7-10710U CPU without the use of parallel computing libraries. We focus on the computational gains resulting from using Gaussian process emulators instead of the PDE solution in the posterior (see Table 3.2) and the relative costs of sampling from the various approximate posteriors (see Tables 3.3, 3.4, 3.5 and 3.6).

Table 3.2 gives average computational timings comparing the evaluation of the solution of the PDE using Firedrake with using the Gaussian process surrogate model. For the baseline surrogate model, the two primary costs are (i) computing the coefficients $\boldsymbol{\alpha} = K(\Theta, \Theta)^{-1}\mathcal{G}_X(\Theta)$, which is an *offline* cost and only needs to be done once, and (ii) computing the predictive mean $m_N^f(\boldsymbol{\theta}) = K(\boldsymbol{\theta}, \Theta)\boldsymbol{\alpha}$, which is the *online* cost and needs to be done for every new test point $\boldsymbol{\theta}$. We see that evaluating $m_N^f(\boldsymbol{\theta})$ is orders of magnitude faster than evaluating $\mathcal{G}_X(\boldsymbol{\theta})$.

Table 3.2: Timings of PDE solution vs baseline Gaussian process emulator

Set-up	$\mathcal{G}_X(\boldsymbol{\theta})$	$m_N^{\mathcal{G}_X}(\boldsymbol{\theta})$	$\boldsymbol{\alpha}$
$d_{\boldsymbol{\theta}} = 2, d_{\mathbf{y}} = 6, D = (0,1), N = 4$	1.1×10^{-1}s	1.0×10^{-4}s	2.5×10^{-4}s
$d_{\boldsymbol{\theta}} = 2, d_{\mathbf{y}} = 6, D = (0,1), N = 20$	1.1×10^{-1}s	1.3×10^{-4}s	6.8×10^{-4}s
$d_{\boldsymbol{\theta}} = 10, d_{\mathbf{y}} = 18, D = (0,1), N = 4$	1.1×10^{-1}s	1.6×10^{-4}s	4.5×10^{-4}s
$d_{\boldsymbol{\theta}} = 2, d_{\mathbf{y}} = 6, D = (0,1)^2, N = 4$	$6.7 \times 10^{0}s$	$1.0 \times 10^{-4}s$	$5.3 \times 10^{-4}s$

In Tables 3.3, 3.4 and 3.5, we compare average computational timings of drawing one sample from the approximate posterior with different models. In Table 3.3, we see that the mean-based approximation with the PDE-informed prior is more expensive than the one with the baseline prior, by a factor of 2-4 depending on the setting. This is to be expected, since the PDE-informed posterior mean $\mathbf{m}_{N,X_f,X_g}^{\mathcal{G}_X}$ involves matrices of larger dimensions than the baseline

posterior mean $\mathbf{m}_N^{\mathcal{G}_X}$.

Table 3.4 investigates the different marginal approximations. Compared to the mean-based approximations in Table 3.3, we see that the marginal approximations are more expensive by a factor of around 2 for the baseline model and around 3-10 for the PDE-constrained model. Within the different marginal approximations, the spatially correlated model is not much more expensive than the baseline model, whereas, depending on the setting, the PDE-constrained model is 2-30 times more expensive.

In Table 3.5, we can see that emulating the log-likelihood significantly reduces the cost of sampling from the mean-based and marginal approximations, by around a factor of 20 compared to the baseline model for emulating the observations.

Table 3.6 shows the effective sample sizes (ESSs) obtained for the different posterior approximations with MALA. We can see that the ESSs are all comparable, implying that it is meaningful to look at the cost per sample to compare the different approximate posteriors in terms of computational cost.

Finally, Table 3.7 gives computational timings for computing the hyperparameters in the covariance functions. The optimisation of the hyperparameters for k_s involves repeatedly computing the inverse of the Gram matrix. In the PDE-constrained model, since the matrix is significantly augmented by the data of PDE, the computation timing is therefore much longer than that of the other two models.

Table 3.3: Timings of different mean-based approximations (baseline and PDE-constrained)

Set-up	$\pi_{\text{mean}}^{N,\mathcal{G}_X}$	$\pi_{\text{mean}}^{N,\mathcal{G}_X,\text{PDE}}$
$d_{\boldsymbol{\theta}} = 2, d_{\mathbf{y}} = 6, D = (0,1), N = 4$	8.5×10^{-4}s	1.2×10^{-3}s ($\bar{N} = 10, d_f = 20$)
$d_{\boldsymbol{\theta}} = 2, d_{\mathbf{y}} = 6, D = (0,1), N = 20$	9.3×10^{-4}s	1.4×10^{-3}s ($\bar{N} = 10, d_f = 20$)
$d_{\boldsymbol{\theta}} = 10, d_{\mathbf{y}} = 18, D = (0,1), N = 4$	2.6×10^{-3}s	1.2×10^{-2}s ($\bar{N} = 50, d_f = 25$)
$d_{\boldsymbol{\theta}} = 2, d_{\mathbf{y}} = 6, D = (0,1)^2, N = 4$	8.5×10^{-4}s	1.6×10^{-3}s ($\bar{N} = 30, d_f = 30$)

Table 3.4: Timings of different marginal approximations (baseline, spatially correlated and PDE-constrained); $\overline{N}$ and d_f are as in Table 3.3

Set-up	$\pi^{N,\mathcal{G}_X}_{\text{marginal}}$	$\pi^{N,\mathcal{G}_X,s}_{\text{marginal}}$	$\pi^{N,\mathcal{G}_X,\text{PDE}}_{\text{marginal}}$
$d_{\boldsymbol{\theta}} = 2, d_{\mathbf{y}} = 6, D = (0,1), N = 4$	1.7×10^{-3}s	2.2×10^{-3}s	3.2×10^{-3}s
$d_{\boldsymbol{\theta}} = 2, d_{\mathbf{y}} = 6, D = (0,1), N = 20$	2.0×10^{-3}s	2.6×10^{-3}s	5.6×10^{-3}s
$d_{\boldsymbol{\theta}} = 10, d_{\mathbf{y}} = 18, D = (0,1), N = 4$	3.4×10^{-3}s	3.6×10^{-3}s	1.1×10^{-1}s
$d_{\boldsymbol{\theta}} = 2, d_{\mathbf{y}} = 6, D = (0,1)^2, N = 4$	1.7×10^{-3}s	2.2×10^{-3}s	4.8×10^{-2}s

Table 3.5: Timings of mean-based and marginal approximation when emulating the log-likelihood

Set-up	$\pi^{N,\Phi}_{\text{mean}}$	$\pi^{N,\mathcal{G}_X,\Phi}_{\text{marginal}}$
$d_{\boldsymbol{\theta}} = 2, d_{\mathbf{y}} = 6, D = (0,1), N = 4$	3.4×10^{-5}s	5.8×10^{-5}s
$d_{\boldsymbol{\theta}} = 2, d_{\mathbf{y}} = 6, D = (0,1)^2, N = 4$	3.4×10^{-5}s	5.8×10^{-5}s

Table 3.6: Effective sample size for 10^6 samples

Model	ESS
Baseline mean	7274
Baseline marginal	10337
Spatially correlated marginal	9249
PDE-constrained mean	9637
PDE-constrained marginal	9982

Table 3.7: Timings of hyperparameter optimisation

Model	Time
Baseline model (σ, l for k_s)	$0.18s$
Spatially correlated model (l for k_s)	$0.04s$
PDE-constrained model (l for k_s)	$45s$

3.5 Conclusions, discussion and actionable advice

Bayesian inverse problems for PDEs pose significant computational challenges. The application of state-of-the-art sampling methods, including MCMC methods, is typically computationally infeasible due to the large computational cost of simulating the underlying mathematical model for a given value of the unknown parameters. A solution to alleviate this problem is to use a surrogate model to

approximate the PDE solution within the Bayesian posterior distribution. In this work, we considered the use of Gaussian process surrogate models, which are frequently used in engineering and geo-statistics applications and offer the benefit of built-in uncertainty quantification in the variance of the emulator.

The focus of this work was on the practical aspects of using Gaussian process emulators in this context, providing efficient MCMC methods and studying the effect of various modelling choices in the derivation of the approximate posterior on its accuracy and computational efficiency. We now summarise the main conclusions of our investigation.

1. **Emulating log-likelihood vs emulating observations.** We can construct an emulator for the negative log-likelihood Φ or the parameter-to-observation map $\mathcal{G}_X$ in the likelihood (3.3).

    - *Computational efficiency.* The log-likelihood Φ is always scalar-valued, independent of the number of observations $d_{\mathbf{y}}$, which makes the computation of the approximate likelihood for a given value of the parameters $\boldsymbol{\theta}$ much cheaper than the approximate likelihood with emulated $\mathcal{G}_X$. The relative cost will depend on $d_{\mathbf{y}}$.
    - *Accuracy.* When only limited training data are provided, emulating $\mathcal{G}_X$ appears more reliable than emulating Φ, even with the baseline model. The major advantage of emulating $\mathcal{G}_X$ is that it allows us to include correlation between different observations, i.e. between the different entries of $\mathcal{G}_X$. This substantially increases the accuracy of the approximate posteriors, in particular, if we use the PDE structure to define the correlations (see point 3 below).

2. **Mean-based vs marginal posterior approximations.** We can use only the mean of the Gaussian process emulator to define the approximate posterior as in (3.5) and (3.7), or we can make use of its full distribution to define the marginal approximate posteriors as in (3.6) and (3.8).

    - *Computational efficiency.* The mean-based approximations are faster to sample from using MALA. This is due to the simpler structure

of the gradient required for the proposals. The difference in computational times depends on the prior chosen, and is greater for the PDE-constrained model.

- *Accuracy.* The marginal approximations correspond to a form of variance inflation in the approximate posterior (see Section 3.2.2), representing our incomplete knowledge about the PDE solution. They thus combat over-confident predictions. In our experiments, we confirm that they typically allocate larger mass to regions around the true parameter value than the mean-based approximations.

3. **Spatial correlation and PDE-constrained priors.**

   - *Computational efficiency.* Introducing the spatially correlated model only affects the marginal approximation, and sampling from the marginal approximate posterior with the spatially correlated model is slightly slower than with the baseline model. The PDE-constrained model significantly increases the computational times for both the mean-based and marginal approximations, with the extent depending on the size of the additional training data.

   - *Accuracy.* Introducing spatial correlation improves the accuracy of the marginal approximation compared to the baseline model. The most accurate results are obtained with the PDE-constrained riors, which are problem-specific and more informative. A benefit of the spatially correlated model is that it does not rely on the underlying PDE being linear, and easily extends to non-linear settings.

In summary, the marginal posterior approximations and the spatially correlated/ PDE-constrained prior distributions provide mechanisms for increasing the accuracy of the inference and avoiding over-confident biased predictions, without the need to increase N. This is particularly useful in practical applications, where the number of model runs N available to train the surrogate model may be very small due to constraints in time and/or cost. This does result in higher computational cost compared to mean-based approximations based on black-box

priors, but may still be the preferable option if obtaining another training point is impossible or computationally very costly.

Variance inflation, as exhibited in the marginal posterior approximations considered in this work, is a known tool to improve Bayesian inference in complex models, see e.g. [20, 14, 30]. Conceptually, it is also related to including model discrepancy [47, 12]. The approach to variance inflation presented in this work has several advantages. First, the variance inflation being equal to the predictive variance of the emulator means that the amount of variance inflation included depends on the location $\boldsymbol{\theta}$ in the parameter space. We introduce more uncertainty in parts of the parameter space where we have fewer training points and the emulator is possibly less accurate. Second, the amount of variance inflation can be tuned in a principled way using standard techniques for hyperparameter estimation in Gaussian process emulators. There is no need to choose a model for the variance inflation separately from choosing the emulator, since this is determined automatically as part of the emulator.

We did not apply optimal experimental design in this work, i.e. how we should optimally choose the locations Θ of the training data. One would expect that using optimal design will have a large influence on the accuracy of the approximate posteriors, especially for small N. In the context of inverse problems, one usually wants to place the training points in regions of parameter space where the (approximate) posterior places significant mass (see e.g. [41] and the references therein). For a fair comparison between all scenarios, and to eliminate the interplay between optimal experimental design and other modelling choices, we have chosen the training points as a space-filling design in our experiments. We expect the same conclusions to hold with optimally placed points.

Chapter 4

Delayed acceptance Metropolis-Hastings algorithm

4.1 Introduction

The delayed acceptance Metropolis-Hastings (DA-MH) algorithm [18] is an extension of the MH algorithm aiming to reduce the computational cost of the MH algorithm to sample posterior distributions arising in complex inverse problems. In this setting, one would like to avoid *wasteful* evaluations of the forward operator to reduce the overall computational complexity. This is achieved by adding an extra accept-reject step to MH for which the acceptance probability is calculated using an approximate posterior density. If this step is accepted then there is a second, appropriately defined, accept-reject step for ensuring overall ergodicity that makes use of the true posterior distribution. In this way, if the approximate posterior distribution is indeed a good approximation to the true posterior distribution the samples that are accepted through the first step have a high probability of being accepted in the second step reducing thus the *wasteful* evaluations of the forward model. Of course, for the DA-MH to be computationally more efficient than the standard MH algorithm one needs the approximate posterior density to be much cheaper to compute than the true one. This is the case for example, when the approximate posterior is based on a coarse grid numerical model.

In this chapter, we explore the use of the DA-MH algorithm in solving the

Bayesian inverse problem (3.2), particularly in sampling from the true posterior (3.4). We again employ the Gaussian process to emulate the forward map as in the previous chapter and construct the approximate posterior for the first accept-reject step. In Section 4.2, we present the DA-MH algorithm, while in Section 4.3 we perform a number of numerical experiments using the DA-MH algorithm with a number of different approximate posteriors based on the different Gaussian process emulators introduced in Chapter 3.

4.2 DA-MH with GP Emulator

We present the delayed acceptance Metropolis-Hastings (DA-MH) algorithm in Algorithm 3 [18]. Contrary to the standard Metropolis-Hastings (MH) algorithm [55, 40], the DA-MH algorithm incorporates two accept-reject steps. The first step corresponds to the accept-reject step of MH for the approximate posterior $\pi^*(\boldsymbol{\theta}|\mathbf{y})$, while the second step is appropriately defined to ensure ergodicity with respect to the true posterior $\pi(\boldsymbol{\theta}|\mathbf{y})$ [89]. As illustrated in Algorithm 3, the true posterior $\pi(\boldsymbol{\theta}|\mathbf{y})$ is used only in the second accept-reject step. Therefore, employing a suitable approximate posterior can help reject candidates that are unlikely to be accepted in the second step, significantly reducing computational costs by avoiding the evaluation of the true posterior for these candidates.

The quality of the posterior approximation is crucial for the efficiency of the DA-MH algorithm. One effective approach to enhance the quality of the approximate posterior is to explicitly account for the approximation error. Specifically, when the forward model $\mathcal{G}_X$ in the inverse problem (3.2) is emulated by $\mathcal{G}_X^*$, the error can be modeled as

$$\mathbf{y} = \mathcal{G}_X^*(\boldsymbol{\theta}) + E(\boldsymbol{\theta}) + \boldsymbol{\eta},$$

where $E(\boldsymbol{\theta})$ represents the error introduced by the approximation of the forward model, and $\boldsymbol{\eta}$ denotes observational noise [46] .

In the context of the Enhanced Error Model (EEM) technique [5], $E(\boldsymbol{\theta})$ is typically assumed to follow a multivariate Gaussian distribution. The mean and

Algorithm 3 Delayed-acceptance Metropolis-Hastings Algorithm

Require: initial value $\boldsymbol{\theta}_0$, number of samples N, initial time-step γ_0, posterior $\pi(\boldsymbol{\theta}|\mathbf{y})$, approximate posterior $\pi^*(\boldsymbol{\theta}|\mathbf{y})$, optimal acceptance rate α_{opt}

Ensure: Sequence of samples $\{\boldsymbol{\theta}_n\}_{n=0}^N$

while $n < N$ **do**

1. Generate a candidate $\boldsymbol{\theta}'$ from a proposal distribution $q(\cdot|\boldsymbol{\theta}_n)$.
2. Compute the acceptance probability with the approximate posterior

$$\alpha^* := \max\left(1, \frac{\pi^*(\boldsymbol{\theta}'|\mathbf{y})q(\boldsymbol{\theta}_n|\boldsymbol{\theta}')}{\pi^*(\boldsymbol{\theta}_n|\mathbf{y})q(\boldsymbol{\theta}'|\boldsymbol{\theta}_n)}\right) \tag{4.1}$$

3. Generate $r \sim U[0,1]$.

if $r \leq \alpha^*$ **then**

Compute the acceptance probability with the true posterior

$$\alpha := \max\left(1, \frac{\pi(\boldsymbol{\theta}'|\mathbf{y})q(\boldsymbol{\theta}_n|\boldsymbol{\theta}')}{\pi(\boldsymbol{\theta}_n|\mathbf{y})q(\boldsymbol{\theta}'|\boldsymbol{\theta}_n)\alpha^*}\right) \tag{4.2}$$

$\mathbb{I}_n = 1$

Generate $r' \sim U[0,1]$

if $r' \leq \alpha$ **then**

$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}'$

else

$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n$.

end if

else

$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n$

$\mathbb{I}_n = 0$

end if

4. Update the time-step $\gamma_{n+1} = \gamma_n(1 + \frac{\mathbb{I}_n - \alpha_{opt}}{n+1})$

end while

covariance of $E(\boldsymbol{\theta})$ are estimated empirically from the discrepancies between $\mathcal{G}_X^*$ and the true forward model $\mathcal{G}_X$ at a set of points sampled from the prior distribution of $\boldsymbol{\theta}$. This empirical estimation enhances the approximation by capturing the systematic errors of the emulator.

When a Gaussian Process (GP) is employed to emulate the forward map, the error $E(\boldsymbol{\theta})$ does not need to be estimated empirically. Instead, the GP provides an explicit model for $E(\boldsymbol{\theta})$, where the covariance of the error is given by the predictive covariance $K^N(\boldsymbol{\theta}, \boldsymbol{\theta})$ of the GP emulator. Consequently, the marginal posterior approximation (3.6) discussed in Chapter 3, when utilized as the approximate posterior in the DA-MH algorithm, can be regarded as a specific instance of

the EEM technique. By integrating the GP-based error model into the DA-MH algorithm, we effectively account for the uncertainty in the emulator, leading to a more robust and efficient sampling process. We thus anticipate that this approach will outperform the mean-based posterior approximations within the DA-MH framework.

4.3 Numerical experiments

In this section, we present two numerical examples, which are in the same set-up as in Section 3.4.1 and Section 3.4.2. We test the DA-MH algorithm with three different Gaussian process emulators: the baseline model (Section 3.2.2), the spatially correlated model and the PDE-constrained model (Section 3.3.1). Similarly, we assume that in all our experiments the number of training points N is small. We utilize 10^5 samples for all our sampling chains.

As already discussed the main aim of using DA-MH is to reduce the number of wasteful evaluations of the true posterior distribution density. In the ideal situation if the approximate posterior used in (4.1) was perfect then the acceptance probability in (4.2) would always be equal to 1. We thus use the acceptance probability (4.2), which from now on is denoted with α_2, as a way of measuring the efficiency of the DA-MH. The closer to one it is, the better the algorithm is behaving in terms of reducing the wasteful evaluations using the true posterior.

We also discuss the performance of DA-MH through the additional quantity $\#E$ which denotes the number of true posterior evaluations required to get one effective sample. In particular, if the calculation of the approximate posterior density is fast when compared to the true one, the computational cost of the DA-MH algorithm only depends on the number of evaluations of the true posterior. Therefore, when the budget of computation is fixed, a smaller $\#E$ leads to a larger effective sample size (ESS) for the sampling chain, which indicates a higher efficiency of the sampling algorithm.

4.3.1 Constant diffusion coefficient

This example follows the same setup as in Section 3.4.1. In particular, we have the analytical solution of the PDE as well as the analytical form of the true posterior.

We start our investigation utilising the mean-based approximation for the baseline model when $N = 2$ in the first accept reject step in DA-MH. In Table 4.1, we show α_2 for this particular choice of approximation. As we can see using the mean-based approximate posterior in DA-MH random walk leads to an extremely low acceptance rate ($\alpha_2 = 0.013$), which means samples accepted in the first accept-reject step can hardly be accepted by the second step. This results in a chain that is mixing very poorly as we can see in Figure 4.1 (left plot). This of course should not be a surprise since as we have already seen in Figure 3.3, for this choice of parameters, the approximate posterior has almost no overlap with the true posterior. As we can also see in Table 4.1 α_2 is even lower when using the MALA algorithm. This again is because the approximate posterior is not suitable, since MALA drives the chain towards the high-probability regions of the approximate posterior faster, so the proposed candidates are even less likely to be accepted in the second step. When the marginal approximate posterior is used the situation improves both for RW and MALA with α_2 being 0.187 and 0.169 respectively. Again this can be explained from the point of view of variance inflation of the marginal posterior, since as we have already seen in Figure 3.3 contrary to the mean-based posterior, the marginal posterior overlaps with the true posterior even when $N = 2$.

Table 4.1: α_2 for DA-MH algorithm with the baseline model ($N = 2$)

Model \ Algorithm	RW	MALA
Mean-based	0.013	0.001
Marginal	0.187	0.169

We now discuss the performance of the DA-MH random walk algorithm (DA-MHRW) for different approximate models using the quantity $\#E$. The results are summarised in Table 4.2. We first run the MH algorithm to sample the true

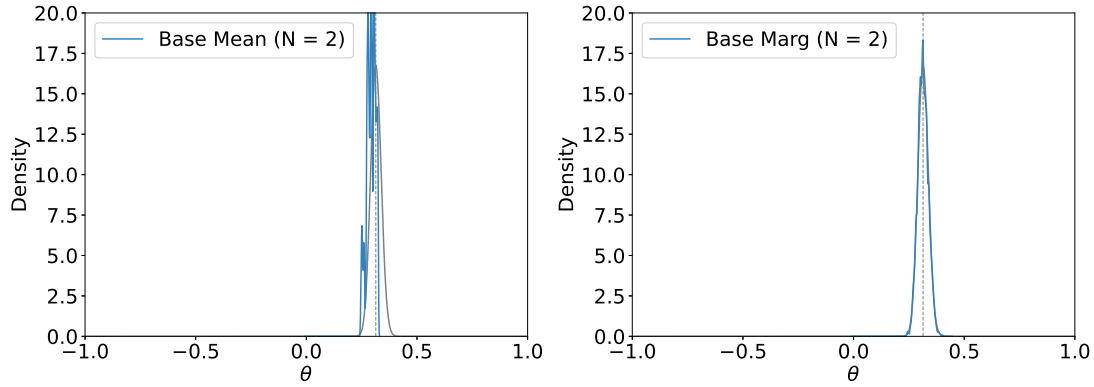

Figure 4.1: Sampling results of the DA-MHRW algorithm. Baseline model ($N = 2$): mean-based posterior (*left plot*) and marginal posterior (*right plot*). The grey line denotes the true posterior and the grey dashed line denotes the true value $\boldsymbol{\theta}^{\dagger}$.

posterior which gives the reference value of 7.4 for $\#E$. As discussed before, posterior approximations that give a lower value than this for $\#E$ are desirable. Again, it is clear that for both the mean-based and the marginal posteriors the type of Gaussian prior used plays an important role. In particular, we see that the PDE-constrained model outperforms both the baseline as well as the spatially correlated model, both for the mean-based and the marginal approximate posteriors. Overall, in the best performing case (PDE constrained model, $N = 2$, $d_f = 5$, $\bar{N} = 10$ and marginal approximate posterior) the DA-MH is roughly 4 times more efficient than the standard MH algorithm applied directly to the true posterior.

Table 4.2: α_2 and $\#E$ for DA-MHRW algorithm with different models ($N = 2$, for PDE-constrained model $d_f = 5$, $d_g = 2$, $\bar{N} = 10$)

Model	Approximate posterior	N	α_2	$\#E$
True	—	—	—	7.4
Baseline	Mean-based	2	0.01	—
Baseline	Mean-based	3	0.49	8.2
PDE-constrained	Mean-based	2	0.88	2.1
Baseline	Marginal	2	0.19	14.4
Baseline	Marginal	3	0.55	6.6
Spatially correlated	Marginal	2	0.20	12.5
PDE-constrained	Marginal	2	0.96	1.7

4.3.2 Two dimensional piece-wise constant coefficient

The example we use here is the same as the example in Section 3.4.1, which is an elliptic equation with a 2-dimensional piece-wise constant. The settings of the Bayesian inverse problem and the Gaussian process emulators are also the same as in Section 3.4.1, hence the shape of the approximate posteriors are the same as plotted in Figure 3.7. We now run DA-MHRW with different approximate posteriors and compare their sampling efficiency in Table 4.3 both in terms of the acceptance rate α_2 as well as the quantity $\#E$. We first run the DA-MHRW algorithm to sample the true posterior which gives the reference value of 10.1 and 8.14 for $\#E$ in two dimensions. Again, the PDE-constrained model outperforms both the baseline as well as the spatially correlated model, both for the mean-based and the marginal approximate posteriors. Overall, in the best performing case (PDE constrained model, $N = 4, d_f = 20$, $d_g = 2$, $\bar{N} = 10$ and marginal approximate posterior) the DA-MH is roughly 2 times more efficient than sampling from the true posterior with the standard MH algorithm.

Table 4.3: α_2 and $\#E$ for DA-MHRW algorithm with different models ($N = 4$, for PDE-constrained model $d_f = 10$, $d_g = 2$, $\bar{N} = 10$)

Model	Approximate posterior	α_2	$\#E(\theta_1)$	$\#E(\theta_2)$
True	—	—	10.1	8.14
Baseline	Mean-based	0.003	—	—
PDE-constrained	Mean-based	0.68	4.7	3.9
Baseline	Marginal	0.11	20.6	17.0
Spatially correlated	Marginal	0.30	5.8	7.8
PDE-constrained	Marginal	0.57	4.6	3.5

4.4 Conclusion

In this chapter, we introduced the delayed-acceptance Metropolis-Hastings (DA-MH) algorithm for efficient sampling from the posterior of the Bayesian inverse problem. By introducing an initial accept-reject step using an approximate posterior density, the algorithm reduces the number of evaluations of the costly true

posterior density. The efficiency of the DA-MH algorithm heavily depends on the quality of the approximate posterior. To enhance this, the error of approximation can be explicitly modelled, as shown in the enhanced error model (EEM) technique. When employing a Gaussian process to emulate the forward map, the approximation error is directly provided by the predictive covariance of the Gaussian process emulator. Consequently, the marginal posterior approximation from Chapter 3 serves as an effective EEM technique within the DA-MH algorithm. In numerical experiments, we showed that the marginal approximate posterior significantly improved performance over the mean-based approximation. Meanwhile, the PDE-constrained model outperforms the baseline model and the spatially correlated model. In the best performing case of DA-MH, that is, the PDE-constrained model with marginal approximate posterior, we showed the computational efficiency is significantly improved compared the the standard MH algorithm. However, when the approximation of posterior is highly biased, for example, when using the baseline model with mean-based approximation and a small size of training data, the DA-MH algorithm might be less efficient than using MH random walk sampling from the true posterior.

Chapter 5

Conclusion

This thesis focuses on the use of Gaussian surrogate models for Bayesian inverse problems associated with linear partial differential equations. A major challenge is how to build a well-performed emulator when only a small amount of training data is available. The physics-informed Gaussian process regression (PI-GPR) method provides a flexible framework for integrating physical information into the Gaussian process, enhancing the accuracy and robustness of prediction without requiring additional observational data. We investigated the application of the PI-GPR method for solving the Bayesian inverse problems.

In chapter 2, we introduced the PI-GPR method and demonstrated its capability to approximate solutions of linear operator equations with uncertainty quantification. We also discussed the method from a linear system perspective, starting with a system of linear equations and extending to an infinite-dimensional case, providing insights into the PI-GPR approach.

In chapter 3, we investigated the use of the Gaussian process emulator for solving the Bayesian inverse problem involving PDEs, particularly in a circumstance where only very limited amount of data is provided. As a statistical model, the covariance matrix of Gaussian process regression inherently provides a natural way of quantifying the uncertainty in the emulator. It conceptually relates to the discrepancy between the observational data [47]. In our work, the amount of variance inflation included also depends on the location $\boldsymbol{\theta}$ in the parameter space. We introduce more uncertainty in parts of the parameter space where we have

fewer training points and the emulator is possibly less accurate. The marginal approximation of the posterior, which incorporates the predictive variance of the emulator, can help mitigate over-confidence and enhance the robustness of inference. We illustrated this result with a series of numerical experiments.

Additionally, we proposed three types of Gaussian surrogate models: Baseline model, spatially correlated model and PDE-constrained model. The PDE-constrained model is based on the extension of the PI-GPR method, which embeds physical constraints directly into the Gaussian process regression. The PDE-constrained model offers significant accuracy improvements in posterior approximation, especially when only limited training data is available and obtaining additional data is prohibitively expensive. Our numerical experiments consistently demonstrated that the best posterior approximations were obtained by combining the PDE-constrained model with the marginal approximation.

Meanwhile, we showed that the gradient of the approximate posteriors based on the Gaussian surrogate model can be computed explicitly, which enables the use of gradient-based sampling algorithms such as MALA without introducing more approximation errors. We showed the performance of the combinations of different Gaussian surrogate models and different approximate posteriors in a series of numerical experiments and also logged their computational timings. The computational cost of mean-based approximation is much slower than the marginal approximation. The PDE-constrained model significantly increases the computational cost with the extent depending on the size of the additional training data.

The primary limitation of the proposed approach is the challenge of scaling GP models to high-dimensional parameter spaces. The covariance matrix $\bar{K}(\Theta, \Theta)$ can quickly increase in size, potentially becoming ill-conditioned and difficult to invert. Even when inversion is feasible, the computational cost scales cubically with the number of parameters, leading to substantial increases in the cost of computing the marginal approximation of the posterior. Thus, future work could focus on exploring advanced dimensionality reduction techniques or developing scalable GP methods to address this limitation.

In chapter 4, we introduced the delayed-acceptance Metropolis-Hastings (DA-

MH) algorithm for efficient sampling from the true posterior. The algorithm adds an initial accept-reject step using an approximate posterior density, which reduces the number of evaluations of the costly true posterior density. The efficiency of the DA-MH algorithm heavily depends on the quality of the approximate posterior. We showed that the marginal approximate posterior introduced in Chapter 3 can be regarded as a type of enhanced error model (EEM) technique, where the approximation error is provided directly by the predictive covariance of the Gaussian process emulator. In our numerical experiments, we demonstrated that the marginal approximate posterior significantly outperformed the mean-based approximation, and the PDE-constrained model outperformed both the baseline model and the spatially correlated model in terms of reducing the number of true posterior density evaluations. The most effective configuration is the DA-MH algorithm using the PDE-constrained model with the marginal approximate posterior. However, when the posterior approximation is highly biased, such as when using the baseline model with a mean-based approximation and a small training dataset, the DA-MH algorithm may be less efficient than employing MH random walk sampling from the true posterior.

Bibliography

[1] Robert J Adler and Jonathan E Taylor. *Random fields and geometry*. Springer Science & Business Media, 2009.

[2] Mauricio Alvarez, David Luengo, and Neil D Lawrence. Latent force models. In *Artificial intelligence and statistics*, pages 9–16. PMLR, 2009.

[3] Christophe Andrieu and Johannes Thoms. A tutorial on adaptive mcmc. *Statistics and computing*, 18:343–373, 2008.

[4] Simon Arridge, Peter Maass, Ozan Öktem, and Carola-Bibiane Schönlieb. Solving inverse problems using data-driven models. *Acta Numerica*, 28:1–174, 2019.

[5] Simon R Arridge, Jari P Kaipio, Ville Kolehmainen, Martin Schweiger, Erkki Somersalo, Tanja Tarvainen, and Marko Vauhkonen. Approximation errors and model reduction with an application in optical diffusion tomography. *Inverse problems*, 22(1):175, 2006.

[6] Ivo Babuska, Fabio Nobile, and Raul Tempone. A stochastic collocation method for elliptic partial differential equations with random input data. *SIAM J. Numerical Analysis*, 45:1005–1034, 01 2007.

[7] Tianming Bai, Aretha L Teckentrup, and Konstantinos C Zygalakis. Gaussian processes for bayesian inverse problems associated with linear partial differential equations. *Statistics and Computing*, 34, 2024.

[8] Heinz H Bauschke, Regina S Burachik, Patrick L Combettes, Veit Elser, D Russell Luke, and Henry Wolkowicz. *Fixed-point algorithms for inverse*

problems in science and engineering, volume 49. Springer Science & Business Media, 2011.

[9] Mario Bertero, Patrizia Boccacci, and Christine De Mol. *Introduction to inverse problems in imaging*. CRC press, 2021.

[10] Edwin V Bonilla, Kian Chai, and Christopher Williams. Multi-task gaussian process prediction. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2007.

[11] Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng. *Handbook of Markov chain Monte Carlo*. CRC press, 2011.

[12] Jennỳ Brynjarsdóttir and Anthony O'Hagan. Learning about physical parameters: The importance of model discrepancy. *Inverse problems*, 30(11):114007, 2014.

[13] T. Bui-Thanh, K. Willcox, and O. Ghattas. Model reduction for large-scale systems with high-dimensional parametric input space. *SIAM Journal on Scientific Computing*, 30(6):3270–3288, 2008.

[14] Daniela Calvetti, Matthew Dunlop, Erkki Somersalo, and Andrew Stuart. Iterative updating of model error for bayesian inversion. *Inverse Problems*, 34(2):025008, 2018.

[15] George Casella, Christian P Robert, and Martin T Wells. Generalized accept-reject sampling schemes. *Lecture Notes-Monograph Series*, pages 342–347, 2004.

[16] Tony F Chan, Gene H Golub, and Pep Mulet. A nonlinear primal-dual method for total variation-based image restoration. *SIAM journal on scientific computing*, 20(6):1964–1977, 1999.

[17] Robert G. Aykroyd Chris J. Oates, Jon Cockayne and Mark Girolami. Bayesian probabilistic numerical methods in time-dependent state estima-

tion for industrial hydrocyclone equipment. *Journal of the American Statistical Association*, 114(528):1518–1531, 2019.

[18] J Andrés Christen and Colin Fox. Markov chain monte carlo using an approximation. *Journal of Computational and Graphical statistics*, 14(4):795–810, 2005.

[19] Jon Cockayne, Chris Oates, Tim Sullivan, and Mark Girolami. Probabilistic numerical methods for PDE-constrained Bayesian inverse problems. *AIP Conference Proceedings*, 1853(1):060001, 2017.

[20] Patrick R Conrad, Mark Girolami, Simo Särkkä, Andrew Stuart, and Konstantinos Zygalakis. Statistical analysis of differential equations: introducing probability measures on numerical solutions. *Statistics and Computing*, 27:1065–1082, 2017.

[21] Paul G. Constantine. *Active Subspaces*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2015.

[22] Paul G. Constantine, E. Dow, and Q. Wang. Active subspace methods in theory and practice: Applications to kriging surfaces. *SIAM Journal on Scientific Computing*, 36(4):A1500–A1524, 2014.

[23] Elizabeth J Cross and Timothy J Rogers. Physics-derived covariance functions for machine learning in structural dynamics. *IFAC-PapersOnLine*, 54(7):168–173, 2021.

[24] ELIZABETH J CROSS, TIMOTHY J ROGERS, and THOMAS J GIBBONS. Grey-box modelling for structural health monitoring: physical constraints on machine learning algorithms. *Structural Health Monitoring 2019*, 2019.

[25] Elizabeth J Cross, Timothy J Rogers, Daniel J Pitchforth, Samuel J Gibson, Sikai Zhang, and Matthew R Jones. A spectrum of physics-informed gaussian processes for regression in engineering. *Data-Centric Engineering*, 5:e8, 2024.

[26] Henry Darcy. *Les fontaines publiques de la ville de Dijon: exposition et application des principes à suivre et des formules à employer dans les questions de distribution d'eau*, volume 1. Victor dalmont, 1856.

[27] Persi Diaconis. The markov chain monte carlo revolution. *Bulletin of the American Mathematical Society*, 46(2):179–205, 2009.

[28] Morris L Eaton and ML Eaton. *Multivariate statistics: a vector space approach*, volume 512. Wiley New York, 1983.

[29] Gregory E Fasshauer. Solving differential equations with radial basis functions: multilevel methods and smoothing. *Advances in computational mathematics*, 11(2):139–159, 1999.

[30] Colin Fox, Tiangang Cui, and Markus Neumayer. Randomized reduced forward models for efficient metropolis–hastings mcmc, with application to subsurface fluid flow and capacitance tomography. *GEM-International Journal on Geomathematics*, 11:1–38, 2020.

[31] Carsten Franke and Robert Schaback. Convergence order estimates of meshless collocation methods using radial basis functions. *Advances in computational mathematics*, 8:381–399, 1998.

[32] Andrew Gelman, Walter R Gilks, and Gareth O Roberts. Weak convergence and optimal scaling of random walk metropolis algorithms. *The annals of applied probability*, 7(1):110–120, 1997.

[33] Andrew Gelman, Gareth O Roberts, Walter R Gilks, et al. Efficient metropolis jumping rules. *Bayesian statistics*, 5(599-608):42, 1996.

[34] Marc G Genton. Classes of kernels for machine learning: a statistics perspective. *Journal of machine learning research*, 2(Dec):299–312, 2001.

[35] Roger G Ghanem and Pol D Spanos. *Stochastic finite elements: a spectral approach*. Springer, 1991.

[36] Matteo Giordano and Richard Nickl. Consistency of bayesian inference with gaussian process priors in an elliptic inverse problem. *Inverse Problems*, 36(8):085001, 2020.

[37] Mark Girolami, Eky Febrianto, Ge Yin, and Fehmi Cirak. The statistical finite element method (statfem) for coherent synthesis of observation data and model predictions. *Computer Methods in Applied Mechanics and Engineering*, 375:113533, 2021.

[38] Thore Graepel. Solving noisy linear operator equations by gaussian processes: Application to ordinary and partial differential equations. In *ICML*, volume 3, pages 234–241, 2003.

[39] Jacques Hadamard. Sur les problèmes aux dérivées partielles et leur signification physique. *Princeton university bulletin*, pages 49–52, 1902.

[40] W Keith Hastings. *Monte Carlo sampling methods using Markov chains and their applications*. Oxford University Press, 1970.

[41] Tapio Helin, Andrew M Stuart, Aretha L Teckentrup, and Konstantinos C Zygalakis. Introduction To Gaussian Process Regression In Bayesian Inverse Problems, With New Results On Experimental Design For Weighted Error Measures. *arXiv preprint arXiv:2302.04518*, 2023.

[42] Ernst Hellinger. Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen. *Journal für die reine und angewandte Mathematik*, 1909(136):210–271, 1909.

[43] Philipp Hennig, Michael A Osborne, and Mark Girolami. Probabilistic numerics and uncertainty in computations. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 471(2179):20150142, 2015.

[44] Dave Higdon, Marc Kennedy, James C. Cavendish, John A. Cafeo, and Robert D. Ryne. Combining field data and computer simulations for calibration and prediction. *SIAM Journal on Scientific Computing*, 26(2):448–466, 2004.

[45] Marko Järvenpää, Michael U Gutmann, Aki Vehtari, and Pekka Marttinen. Parallel gaussian process surrogate bayesian inference with noisy likelihood evaluations. 2021.

[46] Jari Kaipio and Erkki Somersalo. *Statistical and computational inverse problems*, volume 160. Springer Science & Business Media, 2006.

[47] Marc C. Kennedy and Anthony O'Hagan. Bayesian calibration of computer models. *Journal of the Royal Statistical Society, Series B, Methodological*, 63:425–464, 2000.

[48] Teun Kloek and Herman K Van Dijk. Bayesian estimates of equation system parameters: an application of integration by monte carlo. *Econometrica: Journal of the Econometric Society*, pages 1–19, 1978.

[49] A Kolmogorov. Interpolation and extrapolation of stationary random sequences. *Izvestiya Rossiiskoi Akademii Nauk. Seriya Matematicheskaya*, 5:3, 1941.

[50] Han Cheng Lie, Timothy John Sullivan, and Aretha L Teckentrup. Random forward models and log-likelihoods in Bayesian inverse problems. *SIAM/ASA Journal on Uncertainty Quantification*, 6(4):1600–1629, 2018.

[51] Stefano Longobardi, Alexandre Lewalle, Sam Coveney, Ivar Sjaastad, Emil KS Espe, William E Louch, Cynthia J Musante, Anna Sher, and Steven A Niederer. Predicting left ventricular contractile function via gaussian process emulation in aortic-banded rats. *Philosophical Transactions of the Royal Society A*, 378(2173):20190334, 2020.

[52] Youssef Marzouk and Dongbin Xiu. A stochastic collocation approach to bayesian inference in inverse problems. *PRISM: NNSA Center for Prediction of Reliability, Integrity and Survivability of Microsystems*, 6, 10 2009.

[53] Youssef M. Marzouk, Habib N. Najm, and Larry A. Rahn. Stochastic spectral methods for efficient bayesian solution of inverse problems. *Journal of Computational Physics*, 224(2):560–586, 2007.

[54] Tadashi Matsumoto and TJ Sullivan. Images of Gaussian and other stochastic processes under closed, densely-defined, unbounded linear operators. *arXiv preprint arXiv:2305.03594*, 2023.

[55] Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

[56] Eliakim H Moore. On the reciprocal of the general algebraic matrix. *Bulletin of the american mathematical society*, 26:294–295, 1920.

[57] Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

[58] Radford M Neal. Mcmc using hamiltonian dynamics. *arXiv preprint arXiv:1206.1901*, 2012.

[59] Von Neumann. Various techniques used in connection with random digits. *Notes by GE Forsythe*, pages 36–38, 1951.

[60] Harald Niederreiter. *Random number generation and quasi-Monte Carlo methods*. SIAM, 1992.

[61] A. O'Hagan. Bayesian analysis of computer code outputs: A tutorial. *Reliability Engineering & System Safety*, 91(10):1290–1300, 2006.

[62] Yanni Papandreou, Jon Cockayne, Mark Girolami, and Andrew Duncan. Theoretical guarantees for the statistical finite element method. *SIAM/ASA Journal on Uncertainty Quantification*, 11(4):1278–1307, 2023.

[63] Robert L Parker. *Geophysical inverse theory*, volume 1. Princeton university press, 1994.

[64] Marvin Pförtner, Ingo Steinwart, Philipp Hennig, and Jonathan Wenger. Physics-informed gaussian process regression generalizes linear pde solvers, 2022.

[65] Daniel J Pitchforth, Timothy J Rogers, Ulf T Tygesen, and Elizabeth J Cross. Grey-box models for wave loading prediction. *Mechanical Systems and Signal Processing*, 159:107741, 2021.

[66] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

[67] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Machine learning of linear differential equations using gaussian processes. *Journal of Computational Physics*, 348:683–693, 2017.

[68] C Radhakrishna Rao, Sujit Kumar Mitra, et al. Generalized inverse of a matrix and its applications. In *Proceedings of the sixth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 601–620. University of California Press Oakland, CA, USA, 1972.

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

[70] Carl Edward Rasmussen. *Evaluation of Gaussian processes and other methods for non-linear regression*. PhD thesis, University of Toronto Toronto, Canada, 1997.

[71] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pages 63–71. Springer, 2003.

[72] Florian Rathgeber, David A Ham, Lawrence Mitchell, Michael Lange, Fabio Luporini, Andrew TT McRae, Gheorghe-Teodor Bercea, Graham R Markall, and Paul HJ Kelly. Firedrake: automating the finite element method by composing abstractions. *ACM Transactions on Mathematical Software (TOMS)*, 43(3):1–27, 2016.

[73] Alejandro Ribes and Francis Schmitt. Linear inverse problems in imaging. *IEEE Signal Processing Magazine*, 25(4):84–99, 2008.

[74] C.P. Robert and G. Casella. *Monte Carlo statistical methods*. Springer Verlag, 2004.

[75] Gareth O Roberts and Richard L Tweedie. Exponential convergence of langevin distributions and their discrete approximations. *Bernoulli*, pages 341–363, 1996.

[76] Jerome Sacks, William J. Welch, Toby J. Mitchell, and Henry P. Wynn. Design and Analysis of Computer Experiments. *Statistical Science*, 4(4):409 – 423, 1989.

[77] Daniel Sanz-Alonso, Andrew M Stuart, and Armeen Taeb. Inverse problems and data assimilation. *arXiv preprint arXiv:1810.06191*, 2018.

[78] J. M. Sanz-Serna. *Markov Chain Monte Carlo and Numerical Differential Equations*, pages 39–88. Springer International Publishing, Cham, 2014.

[79] B Schölkopf. Learning with kernels: support vector machines, regularization, optimization, and beyond, 2002.

[80] Michael Sinsbeck and Wolfgang Nowak. Sequential design of computer experiments for the solution of bayesian inverse problems. *SIAM/ASA Journal on Uncertainty Quantification*, 5(1):640–664, 2017.

[81] Ercan Solak, Roderick Murray-Smith, WE Leithead, Douglas Leith, and Carl Rasmussen. Derivative observations in gaussian process models of dynamic systems. *Advances in neural information processing systems*, 15, 2002.

[82] Michail Spitieris and Ingelin Steinsland. Bayesian Calibration of Imperfect Computer Models using Physics-Informed Priors. *Journal of Machine Learning Research*, 24(108):1–39, 2023.

[83] Michael L. Stein. *Interpolation of spatial data*. Springer Series in Statistics. Springer-Verlag, New York, 1999.

[84] Andrew Stuart and Aretha Teckentrup. Posterior consistency for gaussian process approximations of bayesian posterior distributions. *Mathematics of Computation*, 87, 03 2018.

[85] Andrew M Stuart. Inverse problems: a bayesian perspective. *Acta numerica*, 19:451–559, 2010.

[86] Laura P Swiler, Mamikon Gulian, Ari L Frankel, Cosmin Safta, and John D Jakeman. A survey of constrained gaussian process regression: Approaches and implementation challenges. *Journal of Machine Learning for Modeling and Computing*, 1(2), 2020.

[87] Laura Painton Swiler, Mamikon Gulian, Ari Louis Frankel, Cosmin Safta, and John Davis Jakeman. Constrained gaussian processes: A survey. Technical report, Sandia National Lab.(SNL-NM), Albuquerque, NM (United States); Sandia ..., 2021.

[88] Aretha L Teckentrup. Convergence of gaussian process regression with estimated hyper-parameters and applications in bayesian inverse problems. *SIAM/ASA Journal on Uncertainty Quantification*, 8(4):1310–1337, 2020.

[89] Luke Tierney. A note on metropolis-hastings kernels for general state spaces. *Annals of applied probability*, pages 1–9, 1998.

[90] Norbert Wiener. Extrapolation, interpolation, and smoothing of stationary time series, with engineering applications. 1964.

[91] Herman Wold. *A study in the analysis of stationary time series*. PhD thesis, Almqvist & Wiksell, 1938.

[92] Dongbin Xiu and George Em Karniadakis. Modeling uncertainty in flow simulations via generalized polynomial chaos. *Journal of Computational Physics*, 187(1):137–167, 2003.

[93] Yunan Yang, Björn Engquist, Junzhe Sun, and Brittany F Hamfeldt. Application of optimal transport and the quadratic wasserstein metric to full-waveform inversion. *Geophysics*, 83(1):R43–R62, 2018.

[94] Michael S Zhdanov. *Geophysical inverse theory and regularization problems*, volume 36. Elsevier, 2002.

[95] Michael S Zhdanov. *Inverse theory and applications in geophysics*, volume 36. Elsevier, 2015.