entropy

MDPI

Article

Multifidelity Model Calibration in Structural Dynamics Using Stochastic Variational Inference on Manifolds

Panagiotis Tsilifis *, Piyush Pandita, Sayan Ghosh and Liping Wang

Probabilistic Design Group, General Electric Research, Niskayuna, NY 12309, USA
* Correspondence: panagiotis.tsilifis@ge.com

Abstract: Bayesian techniques for engineering problems, which rely on Gaussian process (GP) regression, are known for their ability to quantify epistemic and aleatory uncertainties and for being data efficient. The mathematical elegance of applying these methods usually comes at a high computational cost when compared to deterministic and empirical Bayesian methods. Furthermore, using these methods becomes practically infeasible in scenarios characterized by a large number of inputs and thousands of training data. The focus of this work is on enhancing Gaussian process based metamodeling and model calibration tasks, when the size of the training datasets is significantly large. To achieve this goal, we employ a stochastic variational inference algorithm that enables rapid statistical learning of the calibration parameters and hyperparameter tuning, while retaining the rigor of Bayesian inference. The numerical performance of the algorithm is demonstrated on multiple metamodeling and model calibration problems with thousands of training data.

Keywords: Gaussian processes; stochastic variational inference; multifidelity modeling; manifold gradient ascent; structural dynamics; vibration torsion

Citation: Tsilifis, P.; Pandita, P.; Ghosh, S.; Wang, L. Multifidelity Model Calibration in Structural Dynamics Using Stochastic Variational Inference on Manifolds. *Entropy* **2022**, *24*, 1291. https://doi.org/10.3390/e24091291

Academic Editors: Naonori Ueda and Issei Sato

Received: 30 June 2022
Accepted: 8 September 2022
Published: 13 September 2022

1. Introduction

Modern engineering tasks are often characterized by the need to perform large scale expensive laboratory experiments or amortize hours of computation, performing simulations that are based on sophisticated mathematical formulations. While these "high-fidelity" sources of information provide detailed insight into the complex physical process, one usually faces a heavy computational runtime or a massive financial investment. In addition to this, obtaining data by running experiments or simulations needs more advanced insight, that might not always be extricated from the data by applying state-of-the-art methods used to build data-driven metamodels [1]. Finally, with the advent of Industry 4.0 [2], developing digital twins, which are commonly probabilistic surrogate models representing the underlying physical process, is becoming a routine practice across the industry. In a realistic scenario, the paucity of data and noise in the recorded measurements are challenges that also need to be taken into account.

Surrogate modeling methods that have shown promise in dealing with problems of the aforementioned kind, typically include Gaussian process (GP) regression [3–5], probabilistic deep neural networks [6–8] or polynomial chaos expansions [9–11]. The application of these methods has been extended to problems from different domains, such as manufacturing [12,13], flow through porous media [10,14], and combustion mechanics [15]. Classic formulations of these methods provide a meaningful representation of a model from uncertainty and noise, and they demonstrate strong predictive performance on unseen data. However, these approaches are susceptible to challenges such as limited training data, multiple sources of information that model the same process, and the lack of identifiability of model parameters [16]. By referring to the model at-hand that meets the accuracy required by the current application as a "high-fidelity" model, one standard approach in the literature is to employ similar "low-fidelity" models whose characteristic

is that they provide a lower accuracy on the model output, though they are cheaper to evaluate. A detailed review on multifidelity approaches, that is, approaches that employ more than one models to approximate the same physical process with different levels of accuracy, can be found in [17]. In brief, the most common ways in which the low accuracy–low computational cost is achieved are by simplified physics models (coarse-grid PDE solver) [18], reduced-order models [19], data-fit interpolation models [20,21] and machine learning and mathematical surrogates [22,23].

In this work, our focus is on applying GP regression to problems that have thousands of data [24]. Secondly, we focus on the use of GP regression in both the single-fidelity (where only a high-fidelity model is considered) and the multifidelity (both a low- and a high-fidelity model are employed) modeling scenarios. In the second scenario, we focus on the case where data from two sources of varying fidelity are available, and the task involves calibrating the so-called tuners of the low-fidelity source. In all these tasks, we resort to a fully Bayesian formulation of the GP regression, differentiating ourselves from the works of [25–27], the details of which are discussed in Ghosh et al. [28]. This is a critical aspect of this work, as retaining a fully Bayesian treatment for the metamodeling and model calibration tasks with GPs is a major challenge from a computational and numerical perspective. In some of the authors' previous work (see Pandita et al. [29]), it was demonstrated how savings in computational time could be achieved using adaptive sequential Monte Carlo methods fused with a fully Bayesian treatment, applied to tasks of the above kind. However, the utilization of hundreds of computational processing units or cores is not always practically possible, necessitating the need for alternative approaches. Other adaptive algorithms that accelerate Markov chain Monte Carlo methods for Bayesian inference [30–32] and optimal-transport-based approaches that circumvent the need for MCMC methods [33] have shown promise in recent years.

Most of the above-mentioned works rely on computational power and heavy use of large-scale computing in order to overcome the challenges of training the models. Our main contribution in this work is to achieve computational efficiency by leveraging a variational formulation of Bayesian inference, commonly known as black-box variational inference (BBVI) [34], and by improving the performance of the optimization scheme involved using efficient subsampling, rather than resorting to online access to exorbitant computational resources.

Variational methods [35,36] to Bayesian inference have shown promise in various tasks that resort to a Bayesian formalism in order to train surrogate models [37,38], calibrate physical models [39], and more recently across a swathe of deep learning tasks [40–42]. The key ingredient in variational inference (VI) that enables efficient posterior density exploration conditioned on large quantities of data is to perform the required likelihood function evaluations using random batch-sampling. Introducing this additional level of stochasticity in the algorithm, resulting in what is known as stochastic variational inference (SVI) [43], allows for fast likelihood evaluations during the optimization procedure and scales the algorithm, while a full exploration of the available training dataset is still guaranteed. SVI has been previously successfully applied for training deep GP models [44] and sparse GPs in big data scenarios [45]. In this work, we apply SVI to train hybrid Gaussian process models that make use of training data stemming from multiple levels of fidelity, while at the same time they can incorporate calibration parameters. Specifically, we adopt the well-known Kennedy–O'Hagan formulation [46] that relies on an autoregressive GP scheme, and we develop a training algorithm that scales BBVI for big data problems using batch-sampling. We identify the optimal Gaussian approximations to the true posterior densities of the model's hyperparameters by solving the variational problem with respect to full covariance matrices, thus capturing all correlations between the parameters. To achieve this, we make use of a manifold gradient ascent algorithm that performs the optimization directly on the manifold of symmetric positive semidefinite matrices, as opposed to solving complex constrained optimization problems.

The outline of the paper is as follows: We present the mathematical details of the autoregressive multifidelity calibration model in Section 2. In Sections 3 and 4, we expand on the details of the black-box variational inference and its use in scaling up for big data problems, and we introduce the manifold gradient ascent optimization scheme, to be used for carrying out the optimization task. To illustrate the direct applicability of the proposed approach on calibrating models using data from sources of varying fidelity, we use a set of synthetic functions in Section 5.1. We demonstrate the impact of the extended variational formulation on a benchmark machine learning dataset with thousands of training data, in Section 5.2. In Section 5.3, we highlight the impact of the proposed formulation on a challenging multifidelity problem, in the high-sample regime with over ten thousand training data, where the parameters of interest include the uncertain tuners of the low-fidelity simulation model. We summarize our conclusions and directions for future work in Section 6.

2. Multifidelity Gaussian Process Modeling and Calibration

2.1. Autoregressive Gaussian Processes

We consider the Kennedy and O'Hagan's formulation [47], where two simulators are available, namely $y_h(\mathbf{x}), y_l(\mathbf{x}, \theta)$, where y_h represents some high-fidelity computer code and $y_l(\mathbf{x}, \boldsymbol{\theta})$ represents a low-fidelity simulation code. The design variable $\mathbf{x}$ is assumed to take values within a space of feasible designs $\mathcal{X} \subset \mathbb{R}^D$, while $\boldsymbol{\theta}$ is a set of calibration parameters that characterize the low-fidelity simulator.

The relationship between the two codes is assumed to be

$$y_h(\mathbf{x}, \theta) = \rho y_l(\mathbf{x}, \theta) + \delta(\mathbf{x}) + \epsilon(\mathbf{x}), \tag{1}$$

where $\delta(\mathbf{x})$ is a discrepancy term that is statistically independent of $y_l(\mathbf{x}, \theta)$ and $\epsilon(\mathbf{x})$ accounts for measurement noise and is independent of both $y_l(\mathbf{x}, \theta)$ and $\delta(\mathbf{x})$. The coefficient ρ satisfies

$$\rho = \frac{\text{cov}[y_h(\mathbf{x}, \theta), y_l(\mathbf{x}, \theta)]}{\text{var}[y_l(\mathbf{x}, \theta)]} \tag{2}$$

and therefore accounts for the correlation between the models. Although in general, ρ can be considered a function of $\mathbf{x}$ [48,49], we assume for simplicity that it is constant throughout this work. Further, we take $y_l(\mathbf{x}, \theta)$, $\delta(\mathbf{x})$ to be Gaussian processes with zero mean and variances $\sigma_l^2 r_l(\mathbf{x}, \mathbf{x}')$ and $\sigma_\delta^2 r_\delta(\mathbf{x}, \mathbf{x}')$, respectively, where r_l and r_δ are correlation kernels, here to be taken as squared exponential functions

$$r_t(\mathbf{x}, \mathbf{x}') = \exp\left[-\sum_{i=1}^{D} \frac{(x_i - x_i')^2}{\ell_{i,t}^2} \right], \quad t = l, \delta, \tag{3}$$

with $\ell_{i,t}$ being the correlation length or length scale along dimension i, for the two kernels ($t = l, \delta$).

The framework defined above may suffer from issues that pertain to recovering the correct solutions for the parameters being calibrated, also known as identifiability issues. These drawbacks are known in the literature and have been discussed in various works [50–52]. In this work, we limit our focus on improving the computational efficiency in a fully Bayesian formulation, while acknowledging this characteristic of the multifidelity framework.

2.2. Posterior Distribution

Assume a set of observations are available, namely, $\mathcal{D}_l = \{\mathbf{x}_i, \theta_i, y_i\}_{i=1}^{N_l}$ and $\mathcal{D}_h = \{\mathbf{x}_i, y_i\}_{i=1}^{N_h}$ are the input to output sets of points corresponding to the low- and high-fidelity simulators, respectively. Conditioning the distribution of $y_h(\mathbf{x}^*, \theta)$ evaluated at some test point $\mathbf{x}^*$ on the available data $\mathcal{D} := \mathcal{D}_l \cup \mathcal{D}_h$ and taking into account the prior

choices and the independence between $y_l(\cdot)$ and $\delta(\cdot)$, we can write the posterior density as a Gaussian process with mean and variance given by [46]

$$\mu_{y_h}(\mathbf{x}^*, \theta) = t_h(\mathbf{x}^*, \theta) V_h^{-1} \mathbf{y} \tag{4}$$

and

$$\sigma^2_{y_h}(\mathbf{x}^*, \theta) = \sigma^2_h(\mathbf{x}^*) - t_h(\mathbf{x}^*, \theta) V_h^{-1} t_h(\mathbf{x}^*, \theta). \tag{5}$$

In the above expressions, we use $\mathbf{y} = (\mathbf{y}_l^T, \mathbf{y}_h^T)^T$,

$$V_h(\theta) = \begin{bmatrix} V^{(l,l)} & V^{(l,h)}(\theta) \\ V^{(h,l)}(\theta) & V^{(h,h)}(\theta) \end{bmatrix} \tag{6}$$

where the diagonal block matrices are given by

$$\begin{aligned} V^{(l,l)} &= \sigma_l^2 \Big(R_l(\mathcal{D}_l) + \sigma^2_{\epsilon_l} I \Big), \\ V^{(h,h)}(\theta) &= \sigma_\delta^2 \Big(R_\delta(\mathcal{D}_h) + \sigma^2_{\epsilon_h} I \Big) + \sigma_l^2 \rho^2 \Big(R_l(\mathcal{D}_h(\boldsymbol{\theta})) + \sigma^2_{\epsilon_l} I \Big), \end{aligned} \tag{7}$$

and $R_t(\mathcal{D}_t)$ is the correlation matrix with entries $r_t(\mathbf{x}, \mathbf{x}')$ for $\mathbf{x}$, $\mathbf{x}' \in \mathcal{D}_t$, $t = l, \delta$. In the above, $\mathcal{D}_h(\theta) := \{(\mathbf{x}_i, \theta)\}_{i=1}^{N_h}$ for $\mathbf{x}_i \in \mathcal{D}_h$. The off-diagonal blocks are written as

$$V^{(l,h)}(\theta) = \rho V^{(l,l)}(\mathcal{D}_l, \mathcal{D}_h(\theta)). \tag{8}$$

At last, we define the vector

$$t_h(\mathbf{x}^*, \theta) = \begin{pmatrix} \rho \sigma_l^2 R_l((\mathbf{x}^*, \theta), \mathcal{D}_l) \\ \rho^2 \sigma_l^2 R_l((\mathbf{x}^*, \theta), \mathcal{D}_l) + \sigma_\delta^2 R_\delta(\mathbf{x}^*, \mathcal{D}_h) \end{pmatrix}. \tag{9}$$

3. Variational Inference

Throughout this section we present the main ingredients of the variational inference framework for the purpose of training Gaussian process models by means of exploring a Bayesian posterior density. The target distribution in our case is the posterior distribution of the Gaussian process hyperparameters $\boldsymbol{\omega}$, defined as the set of length scales $\ell_{i,t}$, $t = l, h$ along each dimension of $\mathcal{X}$, the variance parameters σ_l^2, σ_h^2, $\sigma^2_{\epsilon_t}$, t, h, and the calibration parameters $\boldsymbol{\theta}$. This posterior density is conditioned on the training data $\mathcal{D}$, which in general consist of the high- and low-fidelity input and output observations. From Bayes' rule

$$p(\boldsymbol{\omega}|\mathcal{D}) = \frac{p(\mathcal{D}|\boldsymbol{\omega}) p(\boldsymbol{\omega})}{p(\mathcal{D})} \tag{10}$$

the posterior density is known as a function of the likelihood term and the prior density, up to a proportionality constant. Variational inference [53,54] bypasses the challenge of sampling from the posterior, by approximating it by an element $q(\boldsymbol{\omega})$ chosen from a parametric family of distributions $\mathcal{Q} = \{q(\boldsymbol{\omega}|\lambda) : \lambda \in \Lambda\}$, where Λ is some set that determines the parameterization of the densities in $\mathcal{Q}$. The criterion for choosing the optimal density from the family is minimizing the Kullback–Leibler (KL) divergence between the candidate and the target densities. We define the KL divergence between the candidate and target densities as follows:

$$\mathrm{KL}[q(\boldsymbol{\omega}|\lambda)||p(\boldsymbol{\omega})] = \int q(\boldsymbol{\omega}|\lambda) \log\left(\frac{q(\boldsymbol{\omega}|\lambda)}{p(\boldsymbol{\omega}|\mathcal{D})} \right) d\boldsymbol{\omega}. \tag{11}$$

Several techniques for solving the optimization problem exist in the literature [35] such as mean-field VI [55] or nonparametric VI [39], and they are typically tailored to problem-

specific choices of prior densities, approximating family of distributions, and the inference problem under investigation.

One common characteristic of the approaches mentioned above is that they all transform the problem of minimizing the KL divergence to an equivalent maximization problem by substituting (10) into (11) to obtain

$$\log p(\mathcal{D}) = \mathrm{KL}[q(\boldsymbol{\omega}|\lambda)||p(\boldsymbol{\omega})] + \mathcal{F}[q], \tag{12}$$

where

$$\mathcal{F}[q] = \mathcal{H}[q] + \int q(\boldsymbol{\omega}|\lambda) \log(p(\mathcal{D}, \boldsymbol{\omega}))d\boldsymbol{\omega} \tag{13}$$

and $\mathcal{H}[q]$ is the entropy of $q(\boldsymbol{\omega}|\lambda)$. Since the left-hand side of (12) is constant, we can conclude that the variational solution can be obtained by maximizing $\mathcal{F}[q]$, which is referred to as the *evidence lower bound (ELBO)*.

Black-Box Variational Inference

One of the most popular choices for optimizing (13) is to directly employ a stochastic gradient descent or ascent algorithm, after observing that the objective function can be written as an expectation

$$\mathcal{F}[q] = \mathbf{E}_q[\log p(\mathcal{D}, \boldsymbol{\theta}) - \log q(\boldsymbol{\omega}|\lambda)], \tag{14}$$

where the expectation is taken with respect to $q(\boldsymbol{\omega}|\lambda)$. The gradient of this expression with respect to the parameters λ that we seek to optimize is

$$\nabla_\lambda \mathcal{F}[q] = \mathbf{E}_q[\nabla_\lambda \log q(\boldsymbol{\omega}|\lambda)(\log p(\mathcal{D}, \boldsymbol{\omega}) - \log q(\boldsymbol{\omega}|\lambda))], \tag{15}$$

where the gradient $\nabla_\lambda \log q(\boldsymbol{\omega}|\lambda)$ is known as the score function for any probability density q and the joint density can be expanded using Bayes' rule to $p(\mathcal{D}, \boldsymbol{\omega}) = p(\mathcal{D}|\boldsymbol{\omega})p(\boldsymbol{\omega})$. A Monte Carlo estimator of (15) can be written as

$$\widehat{\nabla_\lambda \mathcal{F}}[q] = \frac{1}{N} \sum_{i=1}^{N} \nabla_\lambda \log q(\boldsymbol{\omega}^i|\lambda) \Big(\log p(\mathcal{D}, \boldsymbol{\omega}^i) - \log q(\boldsymbol{\omega}^i|\lambda)\Big), \tag{16}$$

where $\boldsymbol{\omega}^i \sim q(\boldsymbol{\omega}|\lambda)$. Note that in the above expression, the gradient appears only on the score function, and can, in general, be computed analytically for certain families of distributions. On the contrary, the log-joint term $\log p(\mathcal{D}, \boldsymbol{\omega})$ which depends on the Bayesian model under investigation, needs not be differentiated. The gradient expression does not make any further assumptions and applies generically on every Bayesian inference problem, justifying the term coined to this approach as *black-box variational inference* [34].

To further scale the algorithm, we perform the log-joint function evaluations $p(\mathcal{D}, \boldsymbol{\omega}^i) = p(\mathcal{D}|\boldsymbol{\omega}^i)p(\boldsymbol{\omega}^i)$ using batch sampling throughout the available dataset $\mathcal{D}$, where each time, a random subset of the dataset is used to form the likelihood term. To put things in a realistic multifidelity context, it is highly unlikely that a big data problem will consist of a large number of high-fidelity observations. Therefore, in this work, we consider the following scenario where the number of training data points in $\mathcal{D}_l$ is significantly larger that the number of high-fidelity observations $\mathcal{D}_h$, that is $|\mathcal{D}_l| \gg |\mathcal{D}_h|$, thus, the batch sampling approach is applied only on $\mathcal{D}_l$. At every evaluation of Equation (17), let $\mathcal{D}_l^i$ be a random subset of $\mathcal{D}_l$ and $\mathcal{D}^i = \mathcal{D}_l^i \cup \mathcal{D}_h$, then Equation (17) is rewritten as follows:

$$\widehat{\nabla_\lambda \mathcal{F}}[q] = \frac{1}{N} \sum_{i=1}^{N} \nabla_\lambda \log q(\boldsymbol{\omega}^i|\lambda) \Big(\log p(\mathcal{D}^i, \boldsymbol{\omega}^i) - \log q(\boldsymbol{\omega}^i|\lambda)\Big), \tag{17}$$

where $\mathcal{D}_l$ is subsampled N times, that is, the number of Monte Carlo samples used to estimate $\widehat{\nabla_\lambda \mathcal{F}}[q]$. This scaling approaching was previously introduced in the literature as stochastic variational inference (SVI) [43].

4. Stochastic Optimization

Manifold Gradient Ascent

For the case where the approximating family of distributions $\mathcal{Q}$ consists of multivariate Gaussian densities, that is, $\mathcal{Q} := \{q(\boldsymbol{\omega}|\lambda) := \mathcal{N}(\boldsymbol{\omega}|\mu, \Sigma)\}$, a suitable optimization scheme needs to be employed over the parameters $\lambda = (\mu, \Sigma)$ such that the symmetric positive semidefiniteness property of the covariance matrix is not violated. Here, we employ a stochastic optimization scheme that is tailored particularly to our problem. The scheme applies a momentum algorithm for updating μ while performing the Σ update using a manifold gradient ascent step. For such a case, we make use of the natural gradient [56] as it is known to be invariant under parameterization [57].

The natural gradient on Riemannian manifolds is defined as

$$\nabla_\lambda^{nat} \mathcal{F}[q] = I_F^{-1} \nabla_\lambda \mathcal{F}[q] \tag{18}$$

where $\nabla_\lambda \mathcal{F}[q]$ is the regular gradient and I_F is the Fisher information for density q that is defined as

$$I_F(\lambda) = \mathbf{E}_q \Big[\nabla_\lambda \log q(\boldsymbol{\omega}|\lambda) (\nabla_\lambda \log q(\boldsymbol{\omega}|\lambda))^T \Big]. \tag{19}$$

In the Gaussian distribution case, the Fisher information matrix becomes

$$I_F(\mu, \Sigma) = \begin{pmatrix} \Sigma^{-1} & 0 \\ 0 & I_F(\Sigma) \end{pmatrix}, \tag{20}$$

where the elements of $I_F(\Sigma)$ are $(I_F(\Sigma))_{\sigma_{ij}, \sigma_{kl}} = \frac{1}{2} \mathrm{tr} \Big(\Sigma^{-1} \frac{\partial \Sigma}{\partial \sigma_{ij}} \Sigma^{-1} \frac{\partial \Sigma}{\partial \sigma_{kl}} \Big)$, and the inverse simplifies to

$$I_F(\lambda)^{-1} \approx \begin{pmatrix} \Sigma & 0 \\ 0 & \Sigma \otimes \Sigma \end{pmatrix}, \tag{21}$$

where "$\otimes$" is the Kronecker product. Finally, the natural gradient of $\mathcal{F}[q]$ can be written as

$$\begin{array}{rcl} \nabla_\mu^{nat} \mathcal{F}[q] & = & \Sigma \nabla_\mu \mathcal{F}[q] \\ \nabla_\Sigma^{nat} \mathcal{F}[q] & = & \Sigma \nabla_\Sigma \mathcal{F}[q] \Sigma \end{array}. \tag{22}$$

In our stochastic gradient ascent scheme, the parameters μ are updated using a momentum algorithm with updating step

$$\mu_{t+1} = \mu_t + \gamma m_{\mu_t} \tag{23}$$

where the momentum term m_{μ_t} is given by

$$m_{\mu_{t+1}} = v m_{\mu_t} + (1 - v) \nabla_\mu^{nat} \mathcal{F}[q]. \tag{24}$$

For the update on Σ, it is necessary to map the point on the tangent space, indicated by the steepest ascent direction, back to the manifold. For that, we use a *retraction mapping* that approximates the exponential map of the manifold of symmetric positive semidefinite matrices [58].

In our case, we use

$$R_\Sigma(\xi) = \Sigma + \xi + \frac{1}{2} \xi \Sigma^{-1} \xi. \tag{25}$$

Further, for the momentum update on the manifolds, we apply a *vector transport* that further projects the translated points back to the tangent space, as was first done in [59]. For our purposes, we apply the following mapping:

$$\Gamma_{\Sigma_1 \to \Sigma_2}(\xi) = U \xi U^T, \;\; U = \left(\Sigma_2 \Sigma_1^{-1}\right)^{1/2}. \tag{26}$$

Finally, our computational algorithm is summarized in Algorithm 1.

Algorithm 1: Manifold gradient ascent

Initialize: Choose μ_0, Σ_0;
Estimate $\nabla_{\mu_0}\mathcal{F}[q]$ and $\nabla_{\Sigma_0}\mathcal{F}[q]$ and the corresponding natural gradients;
Initialize the momentum $m_{\mu_0} = \nabla^{nat}_{\mu_0}\mathcal{F}[q]$ and $m_{\Sigma_0} = \nabla^{nat}_{\Sigma_0}\mathcal{F}[q]$;
for $t = 1$ *to* T **do**
  $\mu_t = \mu_{t-1} + \gamma m_{t-1}$;
  $\Sigma_t = R_{\Sigma_t}(\gamma m_{\Sigma_{t-1}})$;
  Estimate $\nabla_{\mu_t}\mathcal{F}[q]$, $\nabla_{\Sigma_t}\mathcal{F}[q]$;
  Compute natural gradients $\nabla^{nat}_{\mu_t}\mathcal{F}[q] = \Sigma_t \nabla_{\mu_t}\mathcal{F}[q]$, $\nabla^{nat}_{\Sigma_t}\mathcal{F}[q] = \Sigma_t \nabla_{\Sigma_t}\mathcal{F}[q]\Sigma_t$;
  Update momentum terms: $m_{\mu_t} = v m_{\mu_{t-1}} + (1-v)\nabla^{nat}_{\mu_t}\mathcal{F}[q]$ and
  $m_{\Sigma_t} = v\Gamma_{\Sigma_{t-1}\to\Sigma_t}(m_{\Sigma_t}) + (1-v)\nabla^{nat}_{\Sigma_t}\mathcal{F}[q]$;
end
End For

5. Numerical Examples

We studied the performance of the proposed algorithm on three problems. One metamodeling problem and two multifidelity model calibration problems are used in the sections that follow.

5.1. Academic Example

We first considered the following mathematical functions

$$\begin{aligned} f_1(\mathbf{x}, \boldsymbol{\theta}) &= \theta_1(8\mathbf{w}^T\mathbf{x} - 2)\sin(5\mathbf{w}^T\mathbf{x} - 4) + \theta_2(2\mathbf{w}^T\mathbf{x} + \tfrac{1}{2}) \\ f_2(\mathbf{x}, \boldsymbol{\theta}) &= f_1(\mathbf{x}, \boldsymbol{\theta}) + 30(\mathbf{w}^T\mathbf{x})^2, \end{aligned} \tag{27}$$

with the coupling indicating that $f_1(\mathbf{x}, \boldsymbol{\theta})$ can be considered to be a low-fidelity simulator and $f_2(\mathbf{x}, \boldsymbol{\theta})$ the high-fidelity function. We took $\mathbf{x} \in \mathbb{R}^{10}$ and the vector $\mathbf{w}$ was considered a set of known parameters projecting the 10-dimensional vector $\mathbf{x}$ to $\mathbb{R}$. For this example, we took

$$\mathbf{w} = \begin{bmatrix} 0.14042 \\ -0.35474 \\ 0.42674 \\ -0.09312 \\ -0.21463 \\ 0.26425 \\ 0.25603 \\ -0.18959 \\ 0.00467 \\ -0.66800 \end{bmatrix}. \tag{28}$$

A set of 10^4 training points was generated from the low-fidelity function, that is, $\mathcal{D}_l = \{\mathbf{x}_i, \boldsymbol{\theta}_i, y_i\}_{i=1}^{10^4}$ while $\mathcal{D}_h = \{\mathbf{x}_i, y_i\}_{i=1}^{200}$ consisted of 200 points simulated from f_2, where the calibration parameters were fixed to $\boldsymbol{\theta} = (3/2, 30)$. All inputs were generated using a

uniform Latin hypercube sampling on $[-2,2]^{10}$, while the θ_i's were sampled uniformly within $[0.5,2.5] \times [20,40]$. The data are shown in Figure 1.

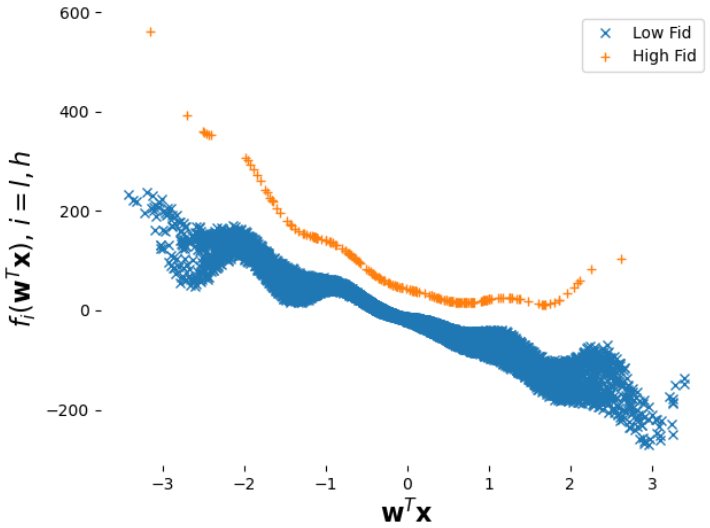

Figure 1. Training data for the academic example. Low-fidelity data are depicted with blue "×" while high-fidelity observations are depicted with orange "+".

To test for the robustness of the approach, we first performed the ELBO optimization corresponding to training an autoregressive GP model on the available training data using a varying number of Monte Carlo samples used to evaluate the ELBO gradient estimate (17), namely $N = 10, 50$, and 100. We ran 5×10^3 iterations of algorithm 1 using an initial learning rate $\gamma_0 = 0.0001$, momentum weight parameters $v = 0.6$, and a random batch size equal to 50 data points (0.5% of the full dataset) to enable the SVI feature. As expected, the runtimes scaled linearly from 13 min for $N = 10$ to 61 min for $N = 50$, and 125 min for $N = 100$. Improving the quality of the MC estimate of the $\mathcal{F}[q]$ function was expected to improve optimization performance. This is highlighted in Table 1 which shows the final ELBO values $\mathcal{F}[q]$ achieved after the optimization procedure was over and the mean predictive standard deviation (std) over all test points. In addition, Figure 2 shows the convergence of the objective function. It can be observed that increasing N resulted in more robust convergence and attaining higher values closer to the true optimum that subsequently resulted in better posterior estimates. The decrease in mean predictive std indicated that the predictive accuracy was improved as well and the confidence increased.

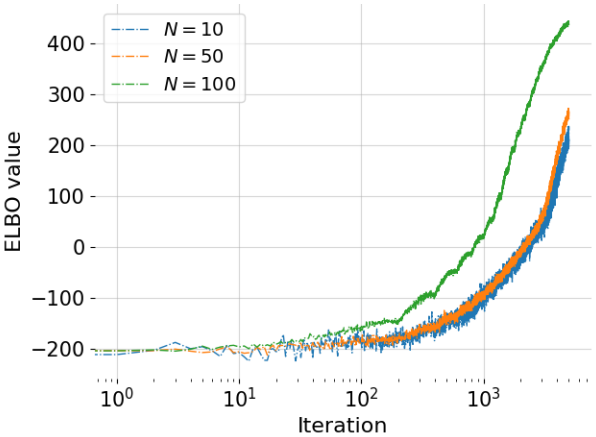

Figure 2. Academic example: ELBO values vs. iterations for each training case ($N = 10, 50$, and 100).

Table 1. Academic example: final ELBO values after 5000 iterations and the computational runtimes for each training case ($N = 10, 50$, and 100).

N	Final $\mathcal{F}[q]$	Mean Predictive Std	Runtime
10	228.77	34.71	13′
50	265.64	27.82	61′
100	439.75	19.28	125′

Next, Figure 3 shows the comparison between the observations and the trained model predictions along with a 45-degree line plot for the case where the number of MC samples was as low as 10. As can be seen, the red "•" marks that correspond to the discrepancy-adjusted prediction match exactly the observations and the variance remains low. The blue "×" marks corresponding to the inferred low-fidelity simulator $\eta(\mathbf{x}, \boldsymbol{\theta})$ fall below the line, which agrees with the observed trends of the true functions as seen in Figure 1. Specifically, the low-fidelity function appears to be the closest possible to the high-fidelity one on design points $\mathbf{x}$ corresponding to values of $\mathbf{w}^T\mathbf{x}$ near the origin, which is when we should expect the discrepancy term points to be the closest to the 45-degree line. When $\eta(\mathbf{x}, \boldsymbol{\theta})$ reaches very low or very high values (near -200 or 200, respectively), the discrepancy is the largest, and indeed the points are far from the 45-degree line.

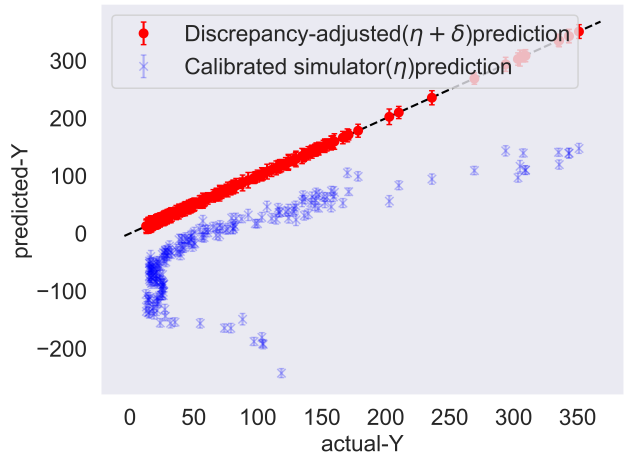

Figure 3. Prediction on the training data for low-fidelity term $\eta(\mathbf{x}, \theta)$ and discrepancy-adjusted high-fidelity output $y_h(\mathbf{x}, \theta)$ versus observations. Model was trained using 10 MC samples for the ELBO evaluation.

Figure 4 shows the prediction versus observations plots for 500 test data points along with a 45-degree line plots again for the three cases where the number of MC samples was 10, 50, and 100, respectively. At last, Figure 5 shows the posterior densities of the two calibration parameters $\boldsymbol{\theta} = (\theta_1, \theta_2)$ obtained using the VI framework. We observe a clear improvement in the accuracy of the θ_1 estimate as the number of Monte Carlo samples increase from 10. To ensure numerical stability in our implementation, the Gaussian approximation was applied on the $\log \boldsymbol{\theta}$ and the resulting density plots were based on a kernel density estimation using 5×10^3 samples from the optimal log-normal approximation that was obtained using the VI approach.

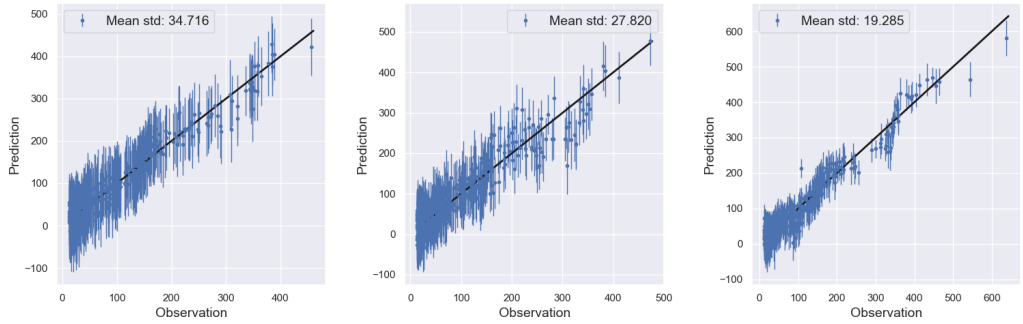

Figure 4. Prediction versus observations for 500 test data points. The model was trained using 10 MC samples for the ELBO gradient evaluation.

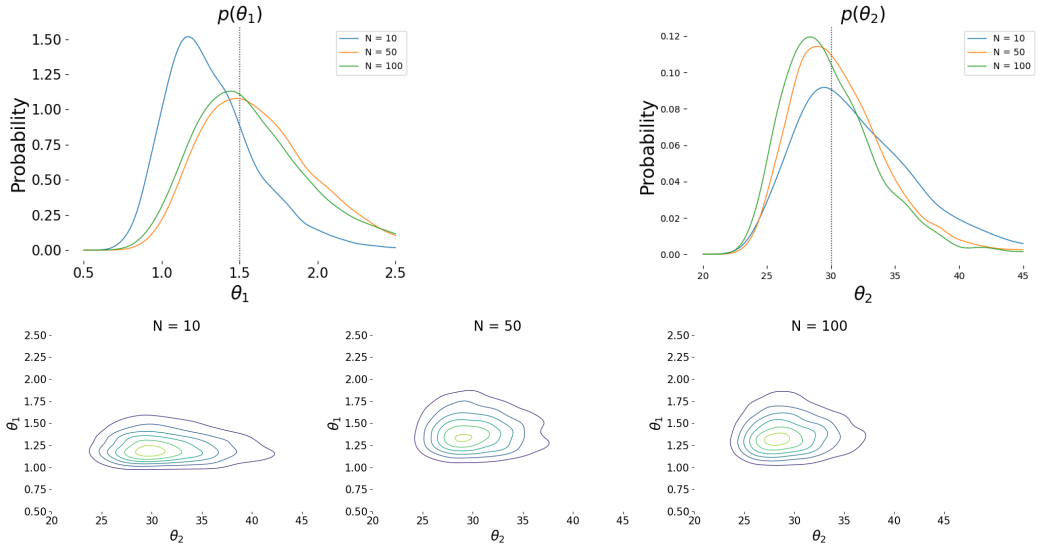

Figure 5. Top: Posterior marginal densities for θ_1 (**left**) and θ_2 (**right**) obtained after the ELBO optimization with a varying number of Monte Carlo samples. **Bottom**: Joint density plots obtain for $N = 10, 50, 100$.

5.2. Chicago Crimes Statistics Dataset

In this section, we demonstrate the applicability of the proposed approach on a metamodeling task. The dataset used for this problem was one of the three datasets under the *Query Analytics Workloads Dataset* section, hosted by the University of California Irvine open-source machine learning data repository (https://archive.ics.uci.edu/ml/datasets/Query+Analytics+Workloads+Dataset accessed on 31 May 2022). This dataset was used in other recent works [60,61] in order to benchmark the performance of the proposed novel machine learning algorithms and has been derived from synthetic query analytics workloads from (https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-Present/ijzp-q8t2 accessed on 31 May 2022). The quantity of interest being modeled was the number of crimes reported or simply the count of crimes in a particular region, in the city of Chicago [60]. The variables used to define the region included the x and y coordinates of the center of the region and the radius of the region. Thus, the problem had three inputs and one output. The dataset had ten thousand pairs of inputs and outputs. We leveraged nine thousand points for training the fully Bayesian metamodel and left out one thousand points as *test data* in order to evaluate the predictive performance of the trained model and we ran 2×10^4 iterations of our optimization scheme.

Two clear observations from Figure 6 are: (a) the predictive performance visibly improves as the batch size of subsampled training data increases from across the three subfigures (here we used batch sizes equal to 45, 67, and 90 samples that corresponded

to using 0.5%, 0.75%, and 1% of the available dataset), and (b) the predictive epistemic uncertainty of the trained model also decreases, indicating a higher confidence in the model. Table 2 shows the improvement of the predictive accuracy of the model as the batch size used for training is increased, as that is illustrated through the root-mean-squared error (RMSE) and the mean std values.

In addition to these, Figure 7 shows the increase in runtime of the algorithm as the batch size of subsampled training data increases. For reference, we also present the runtimes of the sparse GP implementations presented in [4] using the `GPy` package [62] for the same number of iterations and batch size and a latent variable with 80 data points. As can be seen, our approach reduced the runtime significantly for very small batch sizes while the performance of the two algorithms was about the same when batch size became 90.

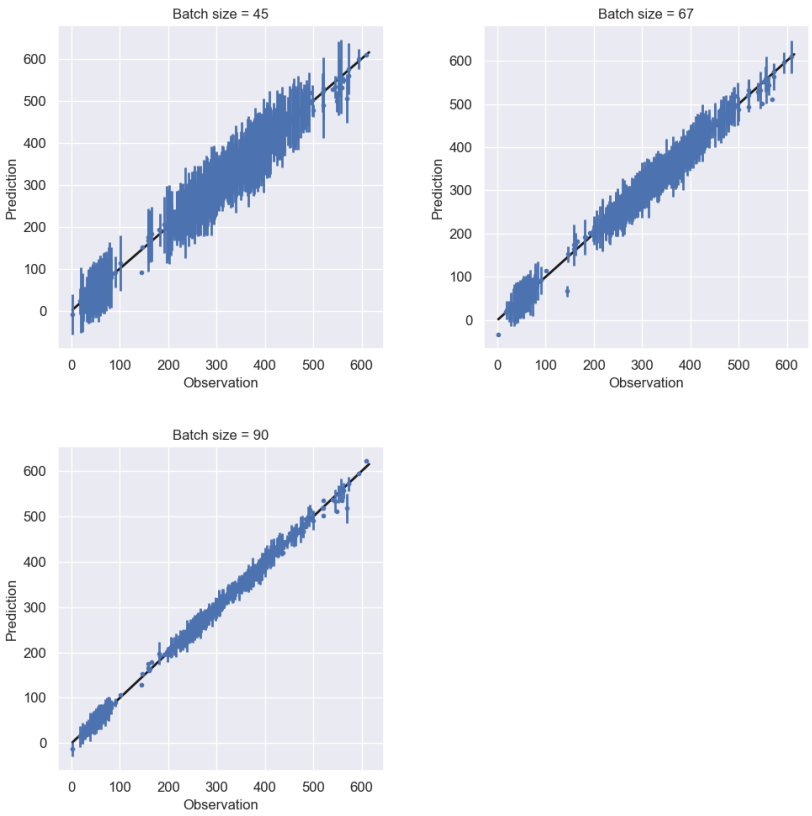

Figure 6. Prediction versus observations for 1000 test data points. The three models were trained using batch sizes equal to 45 (**top left**), 67 (**top right**), and 90 (**bottom**) samples that were resampled from the full dataset.

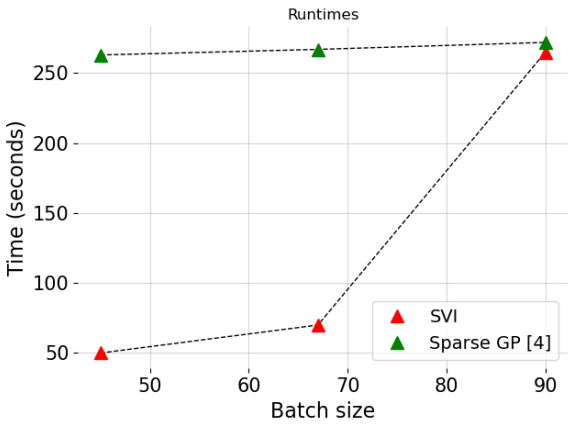

Figure 7. Runtime comparison for three different batch sizes for the Chicago crimes dataset.

Table 2. Chicago crimes dataset: root-mean-squared error (RMSE) values and mean standard deviations for the resulting model trained using different batch sizes.

Batch Size	RMSE Value	Mean Predictive Std
45	6.445	24.626
67	6.315	15.065
90	5.254	7.889

5.3. Torsional Vibration Problem

We considered the torsional vibration problem on the system depicted in Figure 8 consisting of three shafts and two discs of varying geometric characteristics and elasticity properties. Our goal was to built a Gaussian process metamodel on the quantity of interest that expressed the lowest natural frequency, given as

$$Y = \sqrt{\frac{-b - \sqrt{b^2 - 4ac}}{2}} \Big/ 2\pi, \tag{29}$$

where $a = 1$,

$$b = -\left(\frac{K_1+K_2}{J_1} + \frac{K_2+K_3}{J_2}\right), \quad c = \frac{K_1K_2+K_2K_3+K_1K_3}{J_1J_2}. \tag{30}$$

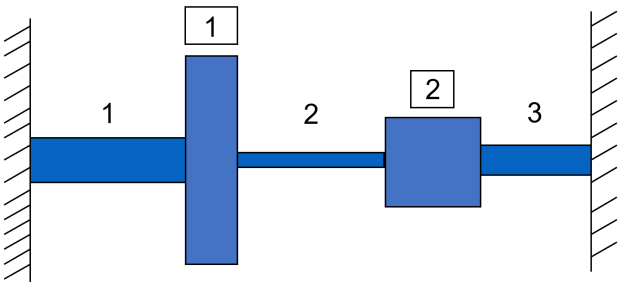

Figure 8. Torsional vibration on system consisting of and three shafts and two discs placed in between successive shafts.

The torsional stiffness values were given by

$$K_i = \theta_1 \frac{\pi G_i d_i}{32 L_i}, \quad i = 1, 2, 3 \tag{31}$$

and the polar moments of inertia were given by

$$J_j = \theta_2 M_j \left(\frac{D_j}{2}\right)^2, \quad i = 1, 2 \tag{32}$$

with $M_j = \frac{\rho_j}{g} \pi t_j \frac{D_j}{4}$, $j = 1, 2$. We considered a high-fidelity simulator where Y_{hf} was evaluated using $\theta_1 = \pi/32$, $\theta_2 = \frac{1}{2}$ and shaft diameters $d_1 = 2$, $d_2 = 1.825$, and $d_3 = 2.25$ in expressions (31) and (32), while data from a low-fidelity Y_{lf} were also used, where θ_1 and θ_2 were considered unknown parameters to be inferred and all diameters were taken equal $d_1 = d_2 = d_3 = 2$. All 12 remaining geometric and elasticity properties of the system were assumed to be design parameters and are described in Table 3.

We considered again an experimental scenario in the big data regime, where 10^4 simulation data points were generated from J_1 and a much smaller number of high-fidelity observations were available from J_2. We tested the robustness of the approach by varying the number of high-fidelity observations from only 50 points up to 250 and we compared the runtimes. Due to the increasing number of data points used to optimize the ELBO, it became necessary to adjust the maximum number of iterations for which the optimization algorithm ran, and therefore, the resulting runtime was affected. For the first three cases, we

performed 1000 iterations; for the case $N_{hf} = 200$, we performed 1500 iterations; and for the remaining case ($N_{hf} = 250$), 2000 iterations were found to be necessary. Figure 9 shows the convergence of the ELBO function along with the root-mean-squared error (RMSE) values obtained for each trained model, based on 100 test data points. As expected, the RMSE goes down with an increasing number of high-fidelity data as shown in Figure 9 bottom.

Table 3. Torsional vibration problem: description of the 12-dimensional input parameters and their values ranges. Length, diameters and thicknesses are given in inches, moduli of rigidity are in lb/sq inch, and weight densities are expressed in lb/cubic inch.

Part	Parameter	Value Range
Shaft 1	Length L_1	[9, 11]
	Modulus of rigidity G_1	$[1053, 1287] \times 10^5$
Shaft 2	Length L_2	[10.8, 13.2]
	Modulus of rigidity G_2	$[558, 682] \times 10^4$
Shaft 3	Length L_3	[7.2, 8.8]
	Modulus of rigidity G_3	$[351, 429] \times 10^4$
Disk 1	Diameter D_1	[10.8, 13.2]
	Thickness t_1	[2.7, 3.3]
	Weight density ρ_1	[0.252, 0.308]
Disk 2	Diameter D_2	[12.6, 15.4]
	Thickness t_2	[3.6, 4.4]
	Weight density ρ_2	[0.09, 0.11]

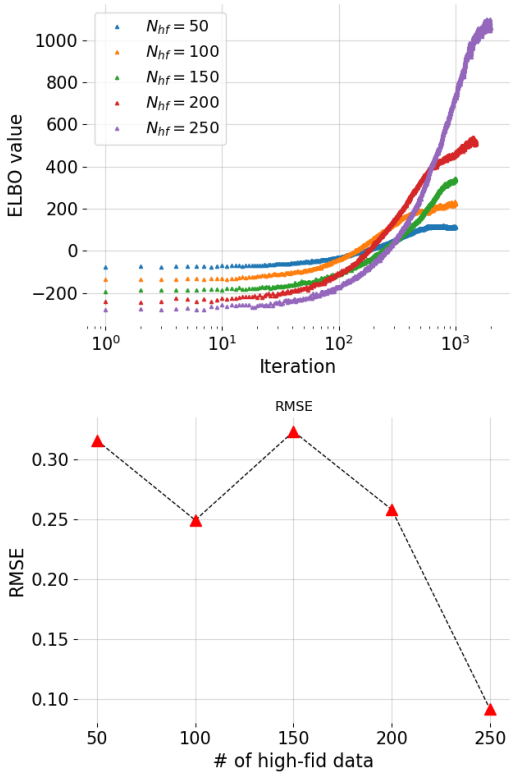

Figure 9. Torsional vibration problem: plots of the ELBO function vs. number of iterations (**top**) and plot of the RMSE values (**bottom**) for different numbers of high-fidelity data points N_{hf}.

The posterior results for the calibrated parameters along with the runtime for each case are shown in Table 4. As can be seen, the true values (0.98 and 0.5) fall within the reported mean values of $\boldsymbol{\theta} \pm 2$ standard deviations for all cases. At last, the comparison of

the model prediction versus observation, along with the 45-degree line plots, is provided for the worse and best cases ($N_{hf} = 50, 250$) in Figure 10.

Table 4. Torsional vibration problem: posterior statistics for the calibration parameters (θ_1, θ_2) and the computational runtimes for each training case.

N_{hf}	θ_1 (Mean, Std)	θ_2 (Mean, Std)	Runtime
50	(0.092, 0.01)	(0.484, 0.042)	9.7′
100	(0.091, 0.01)	(0.487, 0.111)	12.9′
150	(0.132, 0.03)	(0.682, 0.179)	14.1′
200	(0.145, 0.27)	(0.554, 1.614)	44.6′
250	(0.088, 0.0008)	(0.450, 0.0007)	54.4′

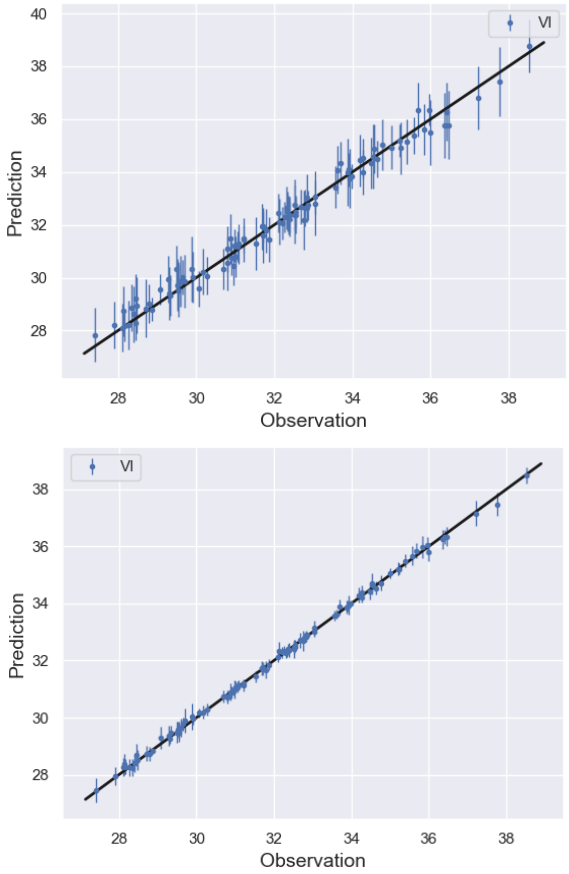

Figure 10. Torsional vibration problem: comparison of trained model prediction vs. observation on 100 test data points along with 45-degree line plots. The high-fidelity points used to train the model were $N_{hf} = 50$ (**top**) and $N_{hf} = 250$ (**bottom**).

6. Conclusions

We enhanced and extended the state-of-the-art stochastic variational Bayesian formulation for tasks that use GPs for multifidelity metamodeling and model calibration tasks, in order to treat problems with tens of thousands of training data and model calibration problems with more than ten inputs. The proposed mathematical formulation extended two classic approaches, the so-called black-box VI and stochastic VI, while utilizing a manifold gradient ascent scheme to accomplish the task of inferring the GP hyperparameters as well as the calibration parameters. The major impact of our work was being able to perform a fully Bayesian uncertainty quantification while training and calibrating models using multifidelity GPs, albeit with large datasets and a moderately large number of inputs. Numerical results on two challenging engineering problems visibly demonstrated a scale up of

classical Bayesian GPs for multifidelity modeling to calibrate *untuned* computer simulators, by enabling savings in computation. This *speed-up* is critical for engineering applications, especially in the industry, where repeated model calibration tasks are a common occurrence and can lead to accumulated savings using the proposed approach.

This work showed promise for accelerating the training procedure in Gaussian process based metamodels without relying on enormous computational power. The key characteristic in our approach was the batch-sampling step used in the stochastic variational inference framework, which allowed the fast computation of the likelihood term and accelerated the optimization task. One key challenge in our approach is that fine tuning of the optimization is required in order to ensure a sufficiently large updating step in the optimization scheme, while at the same time avoiding overshooting. Fine tuning the algorithm heavily depends on the size of the batch samples being used, which is also relative to the original data size that is available. Extremely small batch samples can result in very inaccurate likelihood evaluations and eventually miss the optimum. Another important aspect mentioned above is the number of Monte Carlo samples used for approximating the ELBO function. Very small number of samples can lead to inaccurate estimates with large variance that fail to converge, while on the other hand, a high number of samples makes the algorithm computationally expensive and fails to achieve the desired speed up. Typically, big data problems in Bayesian inference exhibit a well-defined posterior, therefore optimizing the ELBO should always be a feasible task given that some fine tuning has been performed. A limitation of the approach would the case where a big part of the data set is corrupted or contains high noise, in which case, the exploration of the posterior via VI might become challenging due to the complex nature of the true posterior. In such cases, more complex variational approximations need to be considered which could, however, make the algorithm less computationally efficient.

Other general challenges, not associated specifically with our approach, are problems of extremely high input and output dimensions as well as highly nonsmooth response functions. In such cases, a further development of our framework might be necessary such that it aligns with similar approaches in the literature, for instance, enabling covariance matrix sparsity, employing nonsmooth correlation kernels and last, but not least, leveraging parallel computing.

Directions for future work include scaling up the proposed approach to problems with higher input dimensionality, i.e., hundreds of inputs and with more than one sources of information with lower-fidelity and large training data. Additionally, the proposed approach needs more work in order to be applied to problems where the different sources do not share the same inputs.

Author Contributions: Conceptualization, P.T.; methodology, P.T.; software, P.T.; validation, P.T. and P.P.; writing—original draft preparation, P.T. and P.P.; writing—review and editing, P.T., P.P. and S.G.; supervision, L.W.; project administration, S.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All details necessary for regenerating the data used in numerical examples Sections 5.1 and 5.3 is provided in each problem description. Links for the publicly available dataset used in numerical example Section 5.2 is also provided in the relevant section.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Hill, W.J.; Hunter, W.G. A review of response surface methodology: A literature survey. *Technometrics* **1966**, *8*, 571–590. [CrossRef]
2. Vaidya, S.; Ambad, P.; Bhosle, S. Industry 4.0—A glimpse. *Procedia Manuf.* **2018**, *20*, 233–238. [CrossRef]

3. Rasmussen, C.E. *Gaussian Processes in Machine Learning*; Summer School on Machine Learning; Springer: Berlin/Heidelberg, Germany, 2003; pp. 63–71.
4. Hensman, J.; Fusi, N.; Lawrence, N.D. Gaussian processes for Big data. In Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence, Bellevue, WA, USA, 11–15 August 2013; pp. 282–290.
5. Damianou, A.; Lawrence, N.D. Deep gaussian processes. In Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics, Scottsdale, AZ, USA, 29 August 2013; pp. 207–215.
6. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
7. Gal, Y.; Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In Proceedings of the International Conference on Machine Learning, New York, NY, USA, 19–24 June 2016; pp. 1050–1059.
8. Lakshminarayanan, B.; Pritzel, A.; Blundell, C. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 1–12.
9. Ghanem, R.; Spanos, P.D. Polynomial chaos in stochastic finite elements. *J. Appl. Mech.* **1990**, *57*, 197–202. [CrossRef]
10. Tsilifis, P.; Ghanem, R.G. Reduced Wiener chaos representation of random fields via basis adaptation and projection. *J. Comput. Phys.* **2017**, *341*, 102–120. [CrossRef]
11. Sa, G.; Liu, Z.; Qiu, C.; Peng, X.; Tan, J. Novel Performance-Oriented Tolerance Design Method Based on Locally Inferred Sensitivity Analysis and Improved Polynomial Chaos Expansion. *J. Mech. Des.* **2021**, *143*, 022001. [CrossRef]
12. Pandita, P.; Bilionis, I.; Panchal, J.; Gautham, B.; Joshi, A.; Zagade, P. Stochastic multiobjective optimization on a budget: Application to multipass wire drawing with quantified uncertainties. *Int. J. Uncertain. Quantif.* **2018**, *8*, 233–249. [CrossRef]
13. Pandita, P.; Bilionis, I.; Panchal, J. Bayesian optimal design of experiments for inferring the statistical expectation of expensive black-box functions. *J. Mech. Des.* **2019**, *141*, 101404. [CrossRef]
14. Pandita, P.; Bilionis, I.; Panchal, J. Extending expected improvement for high-dimensional stochastic optimization of expensive black-box functions. *J. Mech. Des.* **2016**, *138*, 111412. [CrossRef]
15. Tsilifis, P.; Huan, X.; Safta, C.; Sargsyan, K.; Lacaze, G.; Oefelein, J.C.; Najm, H.N.; Ghanem, R.G. Compressive sensing adaptation for polynomial chaos expansions. *J. Comput. Phys.* **2019**, *380*, 29–47. [CrossRef]
16. Hu, Z.; Hu, C.; Mourelatos, Z.P.; Mahadevan, S. Model discrepancy quantification in simulation-based design of dynamical systems. *J. Mech. Des.* **2019**, *141*, 011401. [CrossRef]
17. Peherstorfer, B.; Willcox, K.; Gunzburger, M. Survey of multifidelity methods in uncertainty propagation, inference, and optimization. *Siam Rev.* **2018**, *60*, 550–591. [CrossRef]
18. Nobile, F.; Tesei, F. A Multi Level Monte Carlo method with control variate for elliptic PDEs with log-normal coefficients. *Stoch. Partial Differ. Equ. Anal. Comput.* **2015**, *3*, 398–444. [CrossRef]
19. Benner, P.; Gugercin, S.; Willcox, K. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM Rev.* **2015**, *57*, 483–531. [CrossRef]
20. Forrester, A.; Keane, A. Recent advances in surrogate-based optimization. *Prog. Aerosp. Sci.* **2009**, *45*, 50–79. [CrossRef]
21. Forrester, A.; Sóbester, A.; Keane, A. Multi-fidelity optimization via surrogate modelling. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **2007**, *463*, 3251–3269. [CrossRef]
22. Huan, X.; Safta, C.; Sargsyan, K.; Vane, Z.; Lacaze, G.; Oefelein, J.; Najm, H. Compressive sensing with cross-validation and stop-sampling for sparse polynomial chaos expansions. *SIAM/ASA J. Uncertain. Quantif.* **2018**, *6*, 907–936. [CrossRef]
23. Tsilifis, P.; Pandita, P.; Ghosh, S.; Andreoli, V.; Vandeputte, T.; Wang, L. Bayesian learning of orthogonal embeddings for multi-fidelity Gaussian Processes. *Comput. Methods Appl. Mech. Eng.* **2021**, *386*, 114147. [CrossRef]
24. Liu, H.; Ong, Y.S.; Shen, X.; Cai, J. When Gaussian process meets big data: A review of scalable GPs. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *31*, 4405–4423. [CrossRef]
25. Wang, K.; Pleiss, G.; Gardner, J.; Tyree, S.; Weinberger, K.Q.; Wilson, A.G. Exact Gaussian processes on a million data points. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 14648–14659.
26. Berns, F.; Beecks, C. Towards Large-scale Gaussian Process Models for Efficient Bayesian Machine Learning. In Proceedings of the 9th International Conference on Data Science, Technology and Applications—DATA, Paris, France, 7–9 July 2020; pp. 275–282. [CrossRef]
27. Tran, A.; Eldred, M.; McCann, S.; Wang, Y. srMO-BO-3GP: A sequential regularized multi-objective Bayesian optimization for constrained design applications using an uncertain Pareto classifier. *J. Mech. Des.* **2022**, *144*, 031705. [CrossRef]
28. Ghosh, S.; Pandita, P.; Atkinson, S.; Subber, W.; Zhang, Y.; Kumar, N.C.; Chakrabarti, S.; Wang, L. Advances in bayesian probabilistic modeling for industrial applications. *ASCE-ASME J. Risk Uncertain. Eng. Syst. Part B Mech. Eng.* **2020**, *6*, 030904. [CrossRef]
29. Pandita, P.; Tsilifis, P.; Ghosh, S.; Wang, L. Scalable Fully Bayesian Gaussian Process Modeling and Calibration with Adaptive Sequential Monte Carlo for Industrial Applications. *J. Mech. Des.* **2021**, *143*, 074502. [CrossRef]
30. Cui, T.; Law, K.J.; Marzouk, Y.M. Dimension-independent likelihood-informed MCMC. *J. Comput. Phys.* **2016**, *304*, 109–137. [CrossRef]
31. Parno, M.D.; Marzouk, Y.M. Transport map accelerated markov chain monte carlo. *SIAM/ASA J. Uncertain. Quantif.* **2018**, *6*, 645–682. [CrossRef]
32. Peherstorfer, B.; Marzouk, Y. A transport-based multifidelity preconditioner for Markov chain Monte Carlo. *Adv. Comput. Math.* **2019**, *45*, 2321–2348. [CrossRef]

33. El Moselhy, T.A.; Marzouk, Y.M. Bayesian inference with optimal maps. *J. Comput. Phys.* **2012**, *231*, 7815–7850. [CrossRef]
34. Ranganath, R.; Gerrish, S.; Blei, D. Black box variational inference. In Proceedings of the Artificial Intelligence and Statistics, Reykjavic, Iceland, 22–25 April 2014; pp. 814–822.
35. Blei, D.; Kucukelbir, A.; McAuliffe, J. Variational inference: A review for statisticians. *J. Am. Stat. Assoc.* **2017**, *112*, 859–877. [CrossRef]
36. Titsias, M.; Lázaro-Gredilla, M. Doubly stochastic variational Bayes for non-conjugate inference. In Proceedings of the International Conference on Machine Learning, Beijing, China, 21–26 June 2014; pp. 1971–1979.
37. Tsilifis, P.; Ghanem, R. Bayesian adaptation of chaos representations using variational inference and sampling on geodesics. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **2018**, *474*, 20180285. [CrossRef]
38. Tsilifis, P.; Papaioannou, I.; Straub, D.; Nobile, F. Sparse Polynomial Chaos expansions using variational relevance vector machines. *J. Comput. Phys.* **2020**, *416*, 109498. [CrossRef]
39. Tsilifis, P.; Bilionis, I.; Katsounaros, I.; Zabaras, N. Computationally efficient variational approximations for Bayesian inverse problems. *J. Verif. Valid. Uncertain. Quantif.* **2016**, *1*, 031004. [CrossRef]
40. Graves, A. Practical variational inference for neural networks. *Adv. Neural Inf. Process. Syst.* **2011**, *24*, 2348–2356.
41. Paisley, J.; Blei, D.M.; Jordan, M.I. Variational Bayesian inference with stochastic search. In Proceedings of the 29th International Coference on International Conference on Machine Learning, Edinburgh, UK, 26 June–1 July 2012; pp. 1363–1370.
42. Deshpande, S.; Purwar, A. Computational creativity via assisted variational synthesis of mechanisms using deep generative models. *J. Mech. Des.* **2019**, *141*, 121402. [CrossRef]
43. Hoffman, M.; Blei, D.; Wang, C.; Paisley, J. Stochastic variational inference. *J. Mach. Learn. Res.* **2013**, *14*, 111401.
44. Salimbeni, H.; Deisenroth, M. Doubly stochastic variational inference for deep Gaussian processes. *arXiv* **2017**, arXiv:1705.08933.
45. Hoang, T.; Hoang, Q.; Low, B. A unifying framework of anytime sparse Gaussian process regression models with stochastic variational inference for big data. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 569–578.
46. Kennedy, M.; O'Hagan, A. Bayesian calibration of computer models. *J. R. Stat. Soc. Ser. B Stat. Methodol.* **2001**, *63*, 425–464. [CrossRef]
47. Kennedy, M.; O'Hagan, A. Predicting the output from a complex computer code when fast approximations are available. *Biometrika* **2000**, *87*, 1–13. [CrossRef]
48. Le Gratiet, L.; Garnier, J. Recursive co-kriging model for design of computer experiments with multiple levels of fidelity. *Int. J. Uncertain. Quantif.* **2014**, *4*, 365–386. [CrossRef]
49. Le Gratiet, L. Bayesian analysis of hierarchical multifidelity codes. *SIAM/ASA J. Uncertain. Quantif.* **2013**, *1*, 244–269. [CrossRef]
50. Arendt, P.D.; Apley, D.W.; Chen, W.; Lamb, D.; Gorsich, D. Improving identifiability in model calibration using multiple responses. *J. Mech. Des.* **2012**, *134*, 100909. [CrossRef]
51. Arendt, P.D.; Apley, D.W.; Chen, W. A preposterior analysis to predict identifiability in the experimental calibration of computer models. *IIE Trans.* **2016**, *48*, 75–88. [CrossRef]
52. Tuo, R.; Jeff Wu, C. A theoretical framework for calibration in computer models: Parametrization, estimation and convergence properties. *SIAM/ASA J. Uncertain. Quantif.* **2016**, *4*, 767–795. [CrossRef]
53. Hoffman, M.; Bach, F.; Blei, D. Online learning for latent Dirichlet allocation. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 6–9 December 2010; pp. 856–864.
54. Wainwright, M.; Jordan, M. Graphical models, exponential families, and variational inference. *Found. Trends® Mach. Learn.* **2008**, *1*, 1–305. [CrossRef]
55. Wang, C.; Blei, D. Variational Inference in Nonconjugate Models. *J. Mach. Learn. Res.* **2013**, *14*, 1005–1031.
56. Amari, S. Natural gradient works efficiently in learning. *Neural Comput.* **1998**, *10*, 251–276. [CrossRef]
57. Martens, J. New insights and perspectives on the natural gradient method. *J. Mach. Learn. Res.* **2020**, *21*, 1–76.
58. Absil, P.; Mahony, R.; Sepulchre, R. *Optimization Algorithms on Matrix Manifolds*; Princeton University Press: Princeton, NJ, USA, 2009.
59. Roy, S.; Harandi, M. Constrained stochastic gradient descent: The good practice. In Proceedings of the 2017 International Conference on Digital Image Computing: Techniques and Applications (DICTA), Sydney, Australia, 29 November–1 December 2017; pp. 1–8.
60. Savva, F.; Anagnostopoulos, C.; Triantafillou, P. Explaining aggregates for exploratory analytics. In Proceedings of the 2018 IEEE International Conference on Big Data (Big Data), Seattle, WA, USA, 10–13 December 2018; pp. 478–487.
61. Anagnostopoulos, C.; Savva, F.; Triantafillou, P. Scalable aggregation predictive analytics. *Appl. Intell.* **2018**, *48*, 2546–2567. [CrossRef]
62. GPy. GPy: A Gaussian Process Framework in Python. Since 2012. Available online: http://github.com/SheffieldML/GPy (accessed on 1 January 2022).