Federated Gaussian Process: Convergence, Automatic Personalization and Multi-fidelity Modeling

Xubo Yue, and Raed Kontar

Abstract—In this paper, we propose FGPR: a Federated Gaussian process ($\mathcal{GP}$) regression framework that uses an averaging strategy for model aggregation and stochastic gradient descent for local computations. Notably, the resulting global model excels in personalization as FGPR jointly learns a shared prior across all devices. The predictive posterior is then obtained by exploiting this shared prior and conditioning on local data, which encodes personalized features from a specific dataset. Theoretically, we show that FGPR converges to a critical point of the full log-marginal likelihood function, subject to statistical errors. This result offers standalone value as it brings federated learning theoretical results to correlated paradigms. Through extensive case studies on several regression tasks, we show that FGPR excels in a wide range of applications and is a promising approach for privacy-preserving multi-fidelity data modeling.

Index Terms—Federated Learning, Gaussian Process Regression, Personalization, Multi-fidelity Modeling, Convergence Rates.

1 Introduction

The modern era of computing is gradually shifting from a centralized regime where data is stored in a centralized location, often a cloud or central server, to a decentralized paradigm that allows devices to collaboratively learn models while keeping their data stored locally [1]. This paradigm shift was set forth by the massive increase in compute resources at the edge device and is based on one simple idea: instead of learning models on a central server, edge devices execute small computations locally and only share the minimum information needed to learn a model. This modern paradigm is often coined as federated learning (FL). Though the prototypical idea of FL dates back decades ago, to the early work of Mangasarian and Solodov [2], it was only brought to the forefront of deep learning after the seminal paper by McMahan et al. [3]. In their work, McMahan et al. [3] propose Federated Averaging (FedAvg) for decentralized learning of a deep learning model. In FedAvg, a central server broadcasts the network architecture and a global model (e.g., initial weights) to selected devices; devices perform local computations (using stochastic gradient descent - SGD) to update the global model based on their local data, and the central server then takes an average of the resulting local models to update the global model. This process is iterated until an accuracy criterion is met.

Despite the simplicity of taking averages of local estimators in deep learning, FedAvg [3] has seen immense success and has since generated an explosive interest in FL. To date, FedAvg for decentralized learning of deep neural networks (NN) was tailored to image classification, text prediction, wireless network analysis, and condition monitoring & failure detection [4, 5, 6, 7, 8, 9]. Besides that, building upon FedAvg's success, literature has been proposed to: (i) tackle adversarial attacks in FL [10, 11]; (ii) allow personalization whereby each device retains its own individualized model [12]; (iii) ensure fairness in performance and participation across devices [13, 14, 15, 16]; (iv) develop more complex aggregation strategies that accommodate deep convolution network [17]; (v) accelerate FL algorithms to improve convergence rate or reduce communication cost [18, 19]; (vi) improve generalization through model ensembling [20].

Despite the aforementioned ubiquitous application of FL, most, if not all, FL literature lies within an empirical risk minimization (ERM) framework - a direct consequence of the focus on deep learning. To date, very few papers study FL beyond ERM, specifically when correlation exists. In this paper, we go beyond ERM and focus on the Gaussian process ($\mathcal{GP}$) regression. We investigate both theoretically and empirically the (i) plausibility of federating model/parameter estimation in $\mathcal{GP}$s and (ii) applications where federated $\mathcal{GP}$s can be of immense value. Needless to say, the inherent capability to encode correlation, quantify uncertainty, and incorporate highly flexible model priors has rendered $\mathcal{GP}$s a key inference tool in various domains such as multi-fidelity modeling, experimental design [21, 22, 23, 24], manufacturing [25, 26], healthcare [27, 28], autonomous vehicles [29] and robotics [30, 31]. Therefore, the success of FL within $\mathcal{GP}$s may help pave the way for FL to infiltrate many new applications and domains.

The central challenge is that, unlike empirical risk minimization (see Sec. 3 for a formal definition), $\mathcal{GP}$s feature correlations across all data points such that any finite collection of which has a joint Gaussian distribution [32, 33]. As a result, the objective function does not simply sum over the loss of individual data points. Adding to that, mini-batch gradients become biased estimators when correlation exists. The performance of FL in such a setting is yet to be understood and explored.

To this end, we propose FGPR: a **F**ederated $\mathcal{GP}$

• *X Yue and R Kontar are with the Industrial & Operations Engineering, University of Michigan, Ann Arbor, MI, 48109. Corresponding Author E-mail: alkontar@umich.edu*

Manuscript received XXX, 20XX; revised XXX, 20XX.

Regression framework that uses FedAvg (i.e., averaging strategy) for model aggregation and SGD for local devices computations. First, we show that, under some conditions, FGPR converges to a critical point of the full log-marginal likelihood function and recovers true parameters (or minimizes the global objective function) up to statistical errors that depend on the device's mini-batch size. Our results hold for kernel functions that exhibit exponential or polynomial eigendecay, which is satisfied by a wide range of kernels commonly used in $\mathcal{GP}$s such as the Matérn and radial basis function (RBF) kernels. Our proof offers standalone value as it is the first to extend the theoretical results of FL beyond ERM and to a correlated paradigm. In turn, this may help researchers further investigate FL within alternative stochastic processes built upon correlations, such as Lévy processes. Second, we explore FGPR within various applications to validate our results. Most notably, we propose FGPR as a privacy-preserving approach for multi-fidelity data modeling and show its advantageous properties compared to the state-of-the-art benchmarks. In addition, we find an interesting yet unsurprising observation. The global model in FGPR excels in personalization. This feature is due to the fact that ultimately FGPR learns a shared prior across all devices. The predictive posterior then is obtained by exploiting this shared prior and conditioning on local data, which encodes personalized features from a specific device. This notion of automatic personalization is closely related to meta-learning, where the goal is to learn a model that can achieve fast personalization.

1.1 Summary of Contributions & Findings

We briefly summarize our contributions below:

- **Convergence:** We explore two data-generating scenarios. (1) Homogeneous setting where local data is generated from the same underlying distribution or stochastic process across all devices; (2) Heterogeneous setting where devices have distributional differences. Under both scenarios and for a large enough batch size M, we prove that the signal variance and noise parameters of FGPR converge to a critical point of the full log-marginal likelihood function (from all data) for kernels that exhibit an exponential or polynomial eigendecay. We also provide uniform error bounds on parameter estimation errors and highlight the ability of FGPR to recover the underlying noise variance.
  - Interestingly, our derived bounds not only depend on iteration T, but also explicitly depend on batch size M, which is a direct consequence of correlation. Our results do not assume any specific functional structure, such as convexity, Lipschitz continuity, or bounded variance.
- **Automatic Personalization Capability:** We demonstrate that FGPR can automatically personalize the shared global model to each local device. Learning a global model by FGPR can be viewed as jointly learning a global $\mathcal{GP}$ prior. On the other hand, the posterior predictive distribution of a $\mathcal{GP}$ depends both on this shared prior and the local training data. The latter can be viewed as a personalized feature encoded in the $\mathcal{GP}$ model. This important personalization feature allows FGPR to excel in the scenario where data among each local device is heterogeneous (Sec. 6 and Sec. 7).
  - In addition to the personalization capability, we find that the prior class learned from FGPR excels in transfer learning (Appendix 0). This idea is similar to meta-learning, where one tries to learn a global model that can quickly adapt to a new task.
- **Multi-fidelity modeling and other applications:** We propose FGPR as a privacy-preserving approach for multi-fidelity data modeling, which combines datasets of varying fidelities into one unified model. We find that in such settings, not only does FGPR preserve privacy but also can improve generalization power across various existing state-of-the-art multi-fidelity and distributed learning (DL) approaches. We also validate FGPR on various simulated datasets and real-world datasets to highlight its advantageous properties.

The remainder of this paper is organized as follows. A detailed literature review can be found in Sec. 2. In Sec. 3, we present the FGPR algorithm. We study the theoretical properties of FGPR in Sec. 4. In Sec. 5-7, we present several empirical results over a range of simulated datasets and real-world datasets. We conclude our paper in Sec. 8 with a brief discussion. Codes are available on the following GitHub link: https://github.com/UMDataScienceLab/Federated_Gaussian_Process.

2 Related Work

2.1 Federated Learning

Most of the existing FL literature has focused on developing deep learning algorithms and their applications in image classification and natural language processing. Please refer to [1] for an in-depth review of FL literature. Here, we briefly review some related papers that tackle data heterogeneity. One popular trend [34, 35] uses regularization techniques to allay heterogeneity. For instance, FedProx [34] adds a quadratic regularizer to the device objective to limit the impact of heterogeneity by penalizing local updates that move far from the global model. Alternatively, personalized models were proposed. Such models usually follow an alternating train-then-personalize approach where a global model is learned, and the personalized model is regularized to stay within its vicinity [12, 36, 37]. Other approaches [38, 39] use different layers of a network to represent global and personalized solutions. More recently, researchers have tried to remove the dependence on a global model for personalization by following a multi-task learning philosophy [40]. Yet, such models can only handle simple convex formulations.

2.2 Distributed Learning

Our work focuses on developing federated models, specifically for $\mathcal{GP}$s, that go beyond deep learning. To date, little to no literature exists along this line. Perhaps the closest

TABLE 1
Comparison between benchmark DL methods and our proposed FL approach. For Modular $\mathcal{GP}$, sparse representation of data entails the pseudo-targets, variational density, model parameters, and lower bound value [41].

Models	Theory	Objective	Comm. Frequency	Comm. Load
DVI [42]	✗	Lower bound	Every Iteration	Gradient Tensor
DGP [43]	✗	PoE Approximate	One-Shot	Predicted Output
Modular $\mathcal{GP}$ [41]	✗	Lower bound	One-Shot	Sparse Representation of Data
FGPR	✓	Exact	Multiple local steps	Model Parameters

field where various regression approaches were investigated is DL for distributed systems. Distributed approaches for MCMC, $\mathcal{GP}$s, PCA, logistic, and quantile regression have been proposed [44, 45, 46, 47, 48, 49, 50, 51]. However, DL and FL have several fundamental differences.

Distributed learning is a centralized computation approach where devices are compute nodes connected by a large bandwidth. Nodes can communicate often and access any part of a dataset, as data partitions can be continuously adjusted. DL aims to parallelize computation tasks across different compute nodes to improve computational efficiency. In FL, data resides at the edge where the goal is to process more of the data at the origin of creation (the edge) and only share updated model parameters rather than entire datasets. In FL, we do not have the luxury to partition, shuffle, and randomize the data. In essence, each device in FL has its own model, and all devices borrow strength from each other to improve model learning. One critical bottleneck in FL is communication [52]. Unlike centralized regimes, aggregation of local models cannot be done after every single optimization iteration, as this incurs huge communication needs between edge devices and the central server. Instead, each device runs multiple local optimization iterates before uploading the data. Indeed, the FedAvg algorithm that we discussed earlier [3] was motivated by the ability to perform multiple optimization iterations locally before updating the global model - hence reducing communication needs. Interestingly, the number of local updates cannot be very large, as we will discuss in Sec. 4.

Along the line of DL, distributed $\mathcal{GP}$s are closely related to our proposed algorithm FGPR. [53] proposed a distributed $\mathcal{GP}$ approach that uses the product-of-experts (PoE) approximation [54] to partition a central dataset into several blocks so that the inference can be made in a distributed fashion. This approach often overestimates predictive variance. [43] proposed a new distributed $\mathcal{GP}$ counterpart (denoted as DGP) that alleviates the aforementioned drawback. The product-of-experts approximation assumes the independence of local experts and, therefore ignores correlation among them. [55] overcame the limitation of PoE using vector quantization to learn correlation among experts. However, the proposed approach requires different nodes to transmit data to each other. [42] resorted to variational inference (VI) to draw a lower bound on the $\mathcal{GP}$ log-marginal likelihood function and developed a distributed variational inference (DVI) framework that parallelizes inference procedures. [41] developed a Modular $\mathcal{GP}$ that extended the VI-based framework into a multi-output scenario where one can model data from multiple sources. Due to space limitation, please refer to [56] for a comprehensive review of the distributed $\mathcal{GP}$ methods.

That said, the methods described above and our approach FGPR feature key differences. The differences are highlighted in Table 1.

First, DGP and Modular $\mathcal{GP}$ are one-shot approaches. Whereas our model FGPR is a collaborative process where the global model is updated over multiple communication rounds. Our model balances communication costs and local training steps. In FL, a one-shot approach, where each device trains till convergence and then model aggregation happens, is sub-optimal. This is due to the well-known "Client-drift" phenomenon [18] where many local steps can push the local solutions to different neighborhoods, and then the aggregation becomes sub-optimal, often giving meaningless predictions. This has also been shown from a theoretical perspective. For instance, in FedAvg, the number of local optimization steps at each communication round should be less than the order of communication rounds for convergence. A similar result is shown for our model in Sec. 4. Whilst Modular $\mathcal{GP}$ and DGP require only one-shot communication, DVI requires communication after every single optimization iterate. This is clearly not viable in FL. Adding to that, DVI needs to send a high-dimension tensor to a central server. This further amplifies communication loads and costs. FGPR, on the other hand, only shares model parameters.

Second, FGPR aims to optimize the exact marginal likelihood function. We alleviate the computational burden by using mini-batch SGD and accordingly show convergence on the exact likelihood. In contrast, DGP resorts to the PoE approximation, and DVI and Modular $\mathcal{GP}$ draw lower bounds on the exact log-marginal likelihood function and optimize the lower bounds. When optimizing the lower bound in a VI-based approach, it does not imply that the resulting solution optimally maximizes the exact log-marginal likelihood.

Third, our paper presents the first successful try at extending the theoretical results of FL beyond ERM and to a correlated paradigm, while existing work [41, 42, 43] did not study the theoretical properties of their proposed algorithms.

More detailed explanations that shed light on the differences between all models can be found in our experiments (Sec. 6 and Sec. 7), where we benchmark our approach with DGP, DVI, and the Modular $\mathcal{GP}$ amongst others.

3 The FGPR Algorithm

In this section, we describe the problem setting in Sec. 3.1 and introduce FGPR - a federated learning scheme for $\mathcal{GP}$s in Sec. 3.2. We then provide insights on the advantages of FGPR in Sec. 3.3. Specifically, we will show that FGPR is capable of

automatically personalizing the global model to each local device. This property allows FGPR to excel in many real-world applications, such as multi-fidelity modeling, where heterogeneity exists. The primary scope of our paper is to focus on regression tasks and provide rigorous theoretical guarantees.

3.1 Background

We consider the Gaussian process regression. We first briefly review the centralized $\mathcal{GP}$ model. Suppose the training dataset is given as $D = \{\boldsymbol{X}, \boldsymbol{y}\}$, where $\boldsymbol{y} = [y_1, ..., y_N]^\intercal$, $\boldsymbol{X} = [x_1^\intercal, ..., x_N^\intercal]$ and N denotes the number of observations. In this paper, we use $|D| = N$ to denote the cardinality of set D. Here, $x \in \mathbb{R}^d$ is a d-dimensional input and $y \in \mathbb{R}$ is the output. We decompose the output as $y_i = f(x_i) + \epsilon_i$, where

$$f \sim \mathcal{GP}(0, \mathcal{K}(\cdot, \cdot; \boldsymbol{\theta}_\mathcal{K})), \quad \epsilon_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2),$$

and $\mathcal{K}(\cdot, \cdot; \boldsymbol{\theta}_\mathcal{K})$ is the prior kernel function parameterized by kernel parameters $\boldsymbol{\theta}_\mathcal{K}$. The prior encodes a belief about the data-generating process and incurs correlations across all data points.

Given a new observation x^*, the goal of $\mathcal{GP}$ regression is to predict $f(x^*)$. By definition, any finite collection of observations from a $\mathcal{GP}$ follows a multivariate normal distribution. Therefore, the joint distribution of $\boldsymbol{y}$ and $f(x^*)$ is given as

$$\begin{bmatrix} \boldsymbol{y} \\ f(x^*) \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2 \boldsymbol{I} & \boldsymbol{K}(\boldsymbol{X}, x^*) \\ \boldsymbol{K}(x^*, \boldsymbol{X}) & \boldsymbol{K}(x^*, x^*) \end{bmatrix}\right)$$

where $\boldsymbol{K}(\cdot, \cdot) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a covariance matrix whose entries are determined by the kernel function $\mathcal{K}(\cdot, \cdot; \boldsymbol{\theta}_\mathcal{K})$. Therefore, the conditional distribution (also known as the posterior predictive distribution) of $f(x^*)$ is given as $\mathcal{N}\left(\mu_{pred}(x^*), \sigma^2_{pred}(x^*)\right)$, where

$$\begin{aligned} &\mu_{pred}(x^*) = \boldsymbol{K}(x^*, \boldsymbol{X}) \left(\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2 \boldsymbol{I}\right)^{-1} \boldsymbol{y}, \\ &\sigma^2_{pred}(x^*) \\ &= \boldsymbol{K}(x^*, x^*) - \boldsymbol{K}(x^*, \boldsymbol{X}) \left(\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2 \boldsymbol{I}\right)^{-1} \boldsymbol{K}(\boldsymbol{X}, x^*). \end{aligned} \tag{1}$$

Here, $\mu_{pred}(x^*)$ is often used as a point estimate of $f(x^*)$ and $\sigma^2_{pred}(x^*)$ quantifies the variance. It can be seen that our predictions will depend on the kernel parameters that parameterize $\boldsymbol{K}(\cdot, \cdot)$ and on the noise parameter σ^2. In this paper, we denote by $\boldsymbol{\theta} := (\boldsymbol{\theta}_\mathcal{K}, \sigma^2)$ the $\mathcal{GP}$ model parameters. Therefore, predicting an accurate output $f(x^*)$ critically depends on finding a good estimate of $\boldsymbol{\theta}$. To estimate $\boldsymbol{\theta}$, the most popular approach is to minimize the negative log-marginal likelihood in the form of

$$\begin{aligned} &-\log p(\boldsymbol{y}|\boldsymbol{X}; \boldsymbol{\theta}) = -\log \int p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{f}; \boldsymbol{\theta}) p(\boldsymbol{f}|\boldsymbol{X}; \boldsymbol{\theta}) df \\ &= \frac{1}{2}[\boldsymbol{y}^\intercal (\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{y} \\ &\quad + \log \left|\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2 \boldsymbol{I}\right| + N \log(2\pi)], \end{aligned} \tag{2}$$

where $\boldsymbol{f} = (f(x_1), \ldots, f(x_N))$, $\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{f} \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I})$ and $p(\boldsymbol{f}|\boldsymbol{X}; \boldsymbol{\theta})$ is a prior density function. There are numerous optimizers that are readily available to minimize $-\log p(\boldsymbol{y}|\boldsymbol{X}; \boldsymbol{\theta})$. In this paper, we resort to stochastic optimization methods such as SGD or Adam [57].

Remark 1. *Needless to say, a current critical challenge in FL is that edge devices have limited compute power. SGD offers an excellent scalability solution to the computational complexity of $\mathcal{GP}$s, which has been a long-standing bottleneck since $\mathcal{GP}$s require inverting a covariance matrix $\boldsymbol{K}(\cdot, \cdot)$ at each iteration of an optimization procedure (see Eq. (2)). This operation, in general, incurs a $\mathcal{O}(N^3)$ time complexity. In SGD, only a mini-batch with a size of $M \ll N$ is taken at each iteration; hence allowing $\mathcal{GP}$s to scale to big data regimes. Besides that, and as will become clear shortly, our approach only requires edge devices to do a few steps on SGD on their local data. Another notable advantage of SGD is that it offers good generalization power [58]. In deep learning, it is well-known that SGD can drive solutions to a flat minimizer that generalizes well [59]. Although this statement is still an open problem in $\mathcal{GP}$, Chen et al. [60] empirically validate that the solution obtained by SGD generalizes better than other deterministic optimizers.*

In the non-federated setting, applying stochastic inference to $\mathcal{GP}$ is not new. Indeed, prior work [61] introduced $N_z < N$ inducing points and employed stochastic VI that optimizes a lower bound of log-marginal likelihood function. As a result, the computation burden is reduced to $\mathcal{O}(N_z^2 N)$. Unfortunately, [62, 63] show that the VI does not work well when the underlying process is not smooth and requires many inducing points to achieve a satisfactory performance. Even for a smooth kernel such as the RBF kernel, $\mathcal{O}(\log^d N)$ inducing points are needed. On the other hand, our work directly applies SGD to the exact log-marginal likelihood function without using an approximation. In Sec. 4, we also support our approach with theoretical guarantees.

Now to use SGD on the exact log-marginal likelihood in Eq. (2) in a centralized regime, we can derive the stochastic gradient given mini-batch of size M as

$$\begin{aligned} &g(\boldsymbol{\theta}; \xi) \\ &= \frac{1}{2}\Bigg[- \boldsymbol{y}_\xi^\intercal \boldsymbol{K}^{-1}(\boldsymbol{X}_\xi, \boldsymbol{X}_\xi) \frac{\partial \boldsymbol{K}(\boldsymbol{X}_\xi, \boldsymbol{X}_\xi)}{\partial \theta} \boldsymbol{K}^{-1}(\boldsymbol{X}_\xi, \boldsymbol{X}_\xi) \boldsymbol{y}_\xi \\ &\quad + \text{Tr}\left(\boldsymbol{K}^{-1}(\boldsymbol{X}_\xi, \boldsymbol{X}_\xi) \frac{\partial \boldsymbol{K}(\boldsymbol{X}_\xi, \boldsymbol{X}_\xi)}{\partial \theta}\right)\Bigg] \\ &= \frac{\text{Tr}\left[\boldsymbol{K}^{-1}(\boldsymbol{X}_\xi, \boldsymbol{X}_\xi)\left(\boldsymbol{I} - \boldsymbol{y}_\xi \boldsymbol{y}_\xi^T \boldsymbol{K}^{-1}(\boldsymbol{X}_\xi, \boldsymbol{X}_\xi)\right) \frac{\partial \boldsymbol{K}(\boldsymbol{X}_\xi, \boldsymbol{X}_\xi)}{\partial \theta}\right]}{2}, \end{aligned}$$

where ξ is the set of indices corresponding to a subset of training data with mini-batch size M and $\boldsymbol{X}_\xi, \boldsymbol{y}_\xi$ is the respective subset of inputs and outputs indexed by ξ. At each iteration t, a subset of training data is taken to update model parameters as

$$\boldsymbol{\theta}^{(t+1)} \leftarrow \boldsymbol{\theta}^{(t)} - \eta^{(t)} g(\boldsymbol{\theta}^{(t)}; \xi^{(t)}),$$

where $\eta^{(t)}$ is the learning rate at iteration t. This step is repeated several times till some exit condition is met.

Although SGD was a key propeller for deep learning, it faces a fundamental challenge in $\mathcal{GP}$s. In deep learning, the empirical risk function is given as $\hat{R}(\boldsymbol{\theta}; D) \approx \frac{1}{N}\sum_{i=1}^N \ell(f_{\boldsymbol{\theta}}(x_i), y_i)$, where $D = \{(x_i, y_i)\}_{i=1}^N$ is the training dataset, $f_{\boldsymbol{\theta}}$ is the neural network to be learned and

$\ell(\cdot,\cdot)$ is a loss function. Therefore, the stochastic gradient of $\hat{R}(\boldsymbol{\theta}; D)$ evaluated using a batch of data, indexed by a set ξ, is $\nabla\hat{R}_k(\boldsymbol{\theta};\xi) = \frac{1}{|\xi|}\sum_{i\in\xi}\ell(f_{\boldsymbol{\theta}}(x_i), y_i)$. As a result, $\mathbb{E}[\nabla\hat{R}(\boldsymbol{\theta};\xi)] = \nabla\hat{R}(\boldsymbol{\theta}; D)$, which means the stochastic gradient is an unbiased estimator of the full gradient. This is a direct consequence of the fact that the objective $\hat{R}(\boldsymbol{\theta}; D)$ is given as a summation over the training data. On the other hand, $\mathcal{GP}$s feature correlations where any finite collection of data points has a joint Gaussian distribution. Therefore, the objective, $-\log p(\boldsymbol{y}|\boldsymbol{X};\boldsymbol{\theta})$, to be minimized in a $\mathcal{GP}$ does not simply sum over individual data points. Consequently, stochastic gradients become biased estimators when correlation exists. Mathematically, this implies $\mathbb{E}[g(\boldsymbol{\theta};\xi)] \neq \nabla(-\log p(\boldsymbol{y}|\boldsymbol{X};\boldsymbol{\theta}))$.

Despite this challenge, we will show in the following sections that our federated SGD approach for learning a $\mathcal{GP}$ converges to a critical point of $\log p(\boldsymbol{y}|\boldsymbol{X};\boldsymbol{\theta})$, subject to statistical errors.

3.2 The FGPR Framework

Suppose there exists $K \geq 2$ local devices. In this paper, we will use (edge) devices and clients interchangeably. For client $k \in [K]$, the local dataset is given as $D_k = \{\boldsymbol{X}_k, \boldsymbol{y}_k\}$ with cardinality N_k. We let $N = \sum_{k=1}^K N_k$. Denote by $L_k(\boldsymbol{\theta}; D_k) := -\log p(\boldsymbol{y}_k|\boldsymbol{X}_k;\boldsymbol{\theta})$ the negative log-marginal likelihood function for device k and $g_k(\boldsymbol{\theta};\xi_k)$ the SG of this negative log-marginal function with respect to a mini-batch of size M indexed by ξ.

In FL, our goal is to collaboratively learn a global parameter $\boldsymbol{\theta}$ that minimizes the global objective function in the form of

$$L(\boldsymbol{\theta}) := \sum_{k=1}^K p_k L_k(\boldsymbol{\theta}; D_k) \tag{3}$$

where $p_k = \frac{N_k}{\sum_{k=1}^K N_k}$ is the weight parameter for device k such that $\sum_{k=1}^K p_k = 1$. To fulfill this goal, during each communication period, each local device k runs E steps of SGD and updates model parameters as

$$\boldsymbol{\theta}_k^{(t+1)} \leftarrow \boldsymbol{\theta}_k^{(t)} - \eta^{(t)} g_k(\boldsymbol{\theta}_k^{(t)}; \xi_k^{(t)}).$$

At the end of each communication round, the central server aggregates model parameters as

$$\bar{\boldsymbol{\theta}} = \sum_{k=1}^K p_k \boldsymbol{\theta}_k.$$

The aggregated parameter $\bar{\boldsymbol{\theta}}$ is then distributed back to local devices. This cycle is repeated several times till convergence. In this training framework, all devices participate during each communication round. We define this framework as synchronous updating. In reality, however, some local devices are frequently offline or reluctant/slow to respond due to various unexpected reasons. To resolve this issue, we develop an asynchronous updating scheme. Specifically, at the beginning of each communication round (c), we select $K_{\text{sample}} \in [1, K)$ clients by sampling probability p_k and denote by $\mathcal{S}$ the indices of these clients. During the communication round, the central server aggregates model parameters as

$$\bar{\boldsymbol{\theta}} = \frac{1}{K_{sample}} \sum_{k\in\mathcal{S}} \boldsymbol{\theta}_k.$$

The detailed procedure is given in Algorithm 1. Although FGPR is primarily a regression framework, it can be extended to handle classification tasks. Due to space limitations, we defer the details to Appendix 1.

Remark 2. *The aggregation strategy used in Algorithm 1 is known as FedAvg [3]. Despite being the first proposed aggregation scheme for FL, FedAvg has stood the test in the past couple of years as one of the most robust and competitive approaches for model aggregation. That being said, it is also possible to extend our algorithm to different strategies, such as different sampling or weighting schemes.*

Algorithm 1: The FGPR algorithm

Data: number of sampled devices K_{sample}, number of communication rounds R, initial model parameter $\boldsymbol{\theta}$, number of local SGD steps E

for $c = 0 : (R-1)$ **do**
- Select K_{sample} clients by sampling probability p_k and denote by $\mathcal{S}$ the indices of these clients;
- Server broadcasts $\boldsymbol{\theta}$;
- **for** $k \in \mathcal{S}$ **do**
  - $\boldsymbol{\theta}_k^{(0)} = \boldsymbol{\theta}$;
  - Update model parameter (e.g., using Algorithm 2);
- **end**
- Aggregation $\bar{\boldsymbol{\theta}}_c = \frac{1}{K_{\text{sample}}}\sum_{k\in\mathcal{S}} \boldsymbol{\theta}_k^{(E)}$, Set $\boldsymbol{\theta} = \bar{\boldsymbol{\theta}}_c$;

end

Return $\bar{\boldsymbol{\theta}}_R$.

Algorithm 2: Local update using SGD

Data: index of device k, number of local updates E, SGD learning rate schedule $\{\eta^{(t)}\}_{t=1}^E$, initial model parameter $\boldsymbol{\theta}_k^{(0)}$

for $t = 0 : (E-1)$ **do**
- Randomly sample a subset of data from D_k and denote it as $\xi_k^{(t)}$;
- $\boldsymbol{\theta}_k^{(t+1)} = \boldsymbol{\theta}_k^{(t)} - \eta^{(t)} g_k(\boldsymbol{\theta}_k^{(t)}; \xi_k^{(t)})$;

end

Return $\boldsymbol{\theta}_k^{(E)}$;

3.3 Why a Single Global $\mathcal{GP}$ Model Works?

In this paper, we will demonstrate the viability of FGPR in cases where data across devices are both homogeneous or heterogeneous. In heterogeneous settings, it is often the case that personalized FL approaches are developed where clients eventually retain their own models while borrowing strength from one another. Popular personalization methods usually fine-tune the global model based on local data while encouraging local weights to stay in a small region in the parameter space of the global model [12]. This allows a balance between the client's shared knowledge and unique

characteristics. This literature, however, is mainly focused on deep learning.

One natural question is: why does a single global model, learned by Algorithm 1, work in FGPR? Here, it is critical to note that, unlike deep learning, estimating $\boldsymbol{\theta}$ in a $\mathcal{GP}$ is equivalent to learning a prior through which predictions are obtained by conditioning on the observed data, and the learned prior. By "learning a prior", we refer to estimating hyper-parameters of $\mathcal{GP}$s by maximizing the global objective.

More specifically, in the $\mathcal{GP}$, we impose a prior on f_k such that $f_k \sim \mathcal{GP}(0, \mathcal{K}(\cdot,\cdot;\boldsymbol{\theta}_{\mathcal{K}}))$. The kernel function is parameterized by $\boldsymbol{\theta}_{\mathcal{K}}$. Therefore, learning a global model by FGPR can be viewed as learning a common model prior over $f_k, \forall k$. On the other hand, the posterior predictive distribution at a testing point x^* is given as

$$
\begin{aligned}
p(f_k^*|\boldsymbol{X}_k, \boldsymbol{y}_k, x^*) &= \int p(f_k^*|x^*, \boldsymbol{f}_k) p(\boldsymbol{f}_k|\boldsymbol{X}_k, \boldsymbol{y}_k) d\boldsymbol{f}_k \\
&= \int p(f_k^*|x^*, \boldsymbol{f}_k) \frac{p(\boldsymbol{y}_k|\boldsymbol{X}_k, \boldsymbol{f}_k) \overbrace{p(\boldsymbol{f}_k)}^{\text{prior}}}{p(\boldsymbol{y}_k|\boldsymbol{X}_k)} d\boldsymbol{f}_k \\
&= \mathcal{N}(\mu_{k,pred}(x^*), \sigma^2_{k,pred}(x^*)),
\end{aligned}
$$

where $\boldsymbol{f}_k$ is defined in Eq. (2), the predictive mean $\mu_{k,pred}(x^*)$ and the predictive variance $\sigma^2_{k,pred}(x^*)$ are defined in Eq. (1). From this posterior predictive equation, one can see that the predicted trajectory (and variance) of $\mathcal{GP}$ in device k is affected by both prior distribution and training data $(\boldsymbol{X}_k, \boldsymbol{y}_k)$ explicitly. For a specific device, the local data themselves embody the personalization role. Therefore, FGPR can automatically tailor a shared global model to a personalized model for each local device. This idea is similar to meta-learning, where one tries to learn a global model that can quickly adapt to a new task.

To see this, we create a simple and stylized numerical example. Another example can be found in the Appendix 2. Suppose there are two local devices. Device 1 has data that follows $y = \sin(x)$ while device 2 has data that follows $y = -\sin(x)$. Each device has 100 training points uniformly spread in $[0, 10]$. We use FedAvg to train a 2-layer neural network. Unfortunately, a single global model of a neural network simply returns a line, as shown in Figure 1. Mathematically, this example solves

$$
\min_{\boldsymbol{\theta}} \left(\|f_{\boldsymbol{\theta}} - \sin(x)\|_2^2 + \|f_{\boldsymbol{\theta}} + \sin(x)\|_2^2 \right),
$$

where $f_{\boldsymbol{\theta}}$ is a global neural network parametrized by $\boldsymbol{\theta}$ and $\|\cdot\|_2^2$ is a functional on $[0, 10]$ defined as $\|f\|_2^2 = \int_0^{10} f(x)^2 dx$.

By taking the derivative of the above objective and setting it to zero, we can find that the solution is $f_{\boldsymbol{\theta}} = 0$. This implies that the global model cannot provide meaningful predictions on both devices.

To remedy this issue, one needs to implement an additional personalization step that fine-tunes the global model from local data. This comes with its own challenges, such as starting with a bad global model (as is the case above) and introducing extra computational costs and parameters. On the other hand, a single $\mathcal{GP}$ model learned from FGPR can provide good interpolation performance for both devices. This demonstrates the advantage of automatic personalization intrinsic to FGPR.

Remark 3. *Despite FGPR being a global modeling approach, in our empirical section, we will compare with personalized FL using NNs when the data distributions are heterogeneous.*

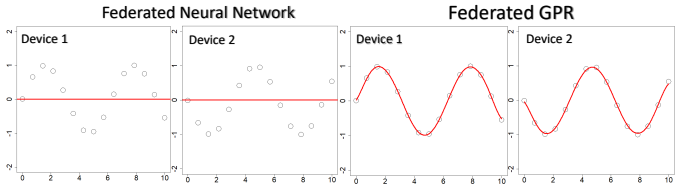

Fig. 1. A simple example used to demonstrate the automatic personalization ability of FGPR. In the plot, the black dots are original data, and the red lines are fitted curves.

4 Theoretical Results

Proving convergence of FGPR introduces new challenges due to correlation and the decentralized nature of model estimation.

In $\mathcal{GP}$s, the objective function cannot be approximated by a summation form since all data points are correlated. This correlation renders the stochastic gradient a biased estimator of the full gradient. To the best of our knowledge, only a recent work from [60] has shown theoretical convergence results of centralized $\mathcal{GP}$ in a correlated setting. Adding to that, FGPR aggregates parameters that are estimated on only a partial dataset.

In this section, we take a step forward in understanding the theoretical properties of $\mathcal{GP}$ estimated in a federated fashion. Specifically, we provide several probabilistic convergence results of FGPR under both homogeneous and heterogeneous clients and under both full and partial device participation settings. Our theoretical results are built upon [60], yet it is not a simple extension of [60] since in the federated setting, partial device participation, non-$i.i.d.$ data, infrequently communication (multiple steps of SGD), parameter aggregation, and the inherent bias stochastic gradient from $\mathcal{GP}$ will further complicate the theoretical analysis. To our knowledge, this is the first paper that theoretically studies FL when correlation exists.

To proceed, we define $\boldsymbol{\theta}_{\mathcal{K}} = (\theta_1, l)$ such that $\boldsymbol{\theta} = (\theta_1, \theta_2, l)$. Here, θ_1 is the signal variance parameter, $\theta_2 = \sigma$ is the noise parameter, and l is the length parameter. Denote by $\boldsymbol{\theta}^* := (\theta_1^*, \theta_2^*, l^*)$ the true data-generating parameter. We impose a structure on the kernel function such that $\mathcal{K}(\cdot,\cdot;\boldsymbol{\theta}_{\mathcal{K}}) = \theta_1^2 \mathrm{k}_f(\cdot,\cdot)$ where $\mathrm{k}_f(\cdot,\cdot)$ is a known function. Now, we define $\mathcal{C}(x_1, x_2) = \mathcal{K}(x_1, x_2; \boldsymbol{\theta}_{\mathcal{K}}) + \sigma_2^2 \mathbb{I}_{x_1 = x_2}$ as a covariance function, where $\mathbb{I}$ is an indicator function. This form of covariance function is ubiquitous and widely adopted. For instance, the Matérn covariance is in the form of

$$
\begin{aligned}
&\mathcal{C}_v(x_1, x_2) \\
&= \theta_1^2 \frac{2^{1-v}}{\Gamma(v)} \left(\sqrt{2v} \frac{\|x_1 - x_2\|}{l} \right)^v K_v \left(\sqrt{2v} \frac{\|x_1 - x_2\|}{l} \right) \\
&\quad + \theta_2^2 \mathbb{I}_{x_1 = x_2}
\end{aligned}
$$

where v is a positive scalar and K_v is the modified Bessel function of the second kind. In this example, $\mathrm{k}_f(x_1, x_2) =$

$\frac{2^{1-v}}{\Gamma(v)}\left(\sqrt{2v}\frac{\|x_1-x_2\|}{l}\right)^v K_v\left(\sqrt{2v}\frac{\|x_1-x_2\|}{l}\right)$. Another example is the RBF covariance:

$$\mathcal{C}_{RBF}(x_1,x_2)=\theta_1^2\exp\left(\frac{\|x_1-x_2\|^2}{2l^2}\right)+\theta_2^2\mathbb{I}_{x_1=x_2}.$$

There are also many other examples, such as the Ornstein–Uhlenbeck covariance and the periodic covariance [64].

Remark 4. *A more general setting is to consider the compound kernel function that is in the form of* $\mathcal{K}(\cdot,\cdot;\boldsymbol{\theta}_{\mathcal{K}})=\sum_{i=1}^{A}\theta_{1i}^2\mathrm{k}_{f_i}(\cdot,\cdot)$. *For simplicity, in the theoretical analysis, we assume* $A=1$. *However, our proof techniques can be easily extended to the scenario where* $A>1$.

In the theoretical analysis, we will show the explicit convergence bounds on θ_1 and θ_2. The convergence behavior of the length parameter l is still an open problem [60]. The key reason is that one needs to apply the eigendecomposition technique to the kernel function and carefully analyze the lower and upper bounds of eigenfunctions. The length parameter l, however, lies in the denominator of a kernel function. In this case, it is extremely challenging to write the kernel function in the form of eigenvalues and bound them. To the best of our knowledge, the work that studies convergence results of l is still vacant even in centralized regimes.

4.1 Assumptions

To derive our convergence results, we make the following assumptions.

Assumption 1. *The parameter space* $\boldsymbol{\Theta}$ *is a compact and convex subset of* $\mathbb{R}^2$. *Moreover,* $(\theta_1^*,\theta_2^*)^\intercal\in\boldsymbol{\Theta}^{\mathrm{o}}$ *and* $\sup_{(\theta_1,\theta_2)^\intercal\in\boldsymbol{\Theta}}\|(\theta_1,\theta_2)^\intercal-(\theta_1^*,\theta_2^*)^\intercal\|>0$, *where* $\boldsymbol{\Theta}^{\mathrm{o}}$ *is the interior of set* $\boldsymbol{\Theta}$.

This assumption indicates that all parameter iterates are bounded, and the global minimizer $(\theta_1^*,\theta_2^*)^\intercal$ exists. Without loss of generality, assume the lower (or upper) bound of the parameter space on each dimension is θ_{min} (or θ_{max}). The convexity of the parameter space (not the objective function) implies that for any parameters $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$ within a bounded region, their convex combination also falls within the same boundary.

Assumption 2. *The norm of the stochastic gradient is bounded. Specifically,*

$$0\le\left\|g_k(\cdot;\xi_k^{(t)})\right\|\le G,\ \textit{for all } k\in[K],t\in[T].$$

Here T is defined as the total number of iteration indices on each device. Mathematically, $T=RE-1$ and $[T]=\{0,\ldots,T\}$.

Remark 5. *It is very common to assume the local functions are L-smooth, (strongly-)convex, or the variance of the stochastic gradient is bounded. However, we do not make those assumptions. More specifically, in Assumption 1, we only assume that the parameter space (rather than the objective function) is a compact and convex subset. This assumption implies that the SGD parameter iterates are bounded within a specific region. Otherwise, the minimizer does not exist, and SGD will not converge. Assumptions 2 introduces conditions related to the boundedness of stochastic gradients. This assumption does not pertain to the L-smoothness of the objective function.*

In the $\mathcal{GP}$ setting, the explicit convergence bound depends on the rate of decay of eigenvalues from a specific type of kernel function. In this paper, we study two types of kernel functions: (1) kernel functions with exponential eigendecay rates; and (2) kernel functions with polynomial eigendecay rates. Those translate to the following assumptions.

Assumption 3a. *For each* $k\in[K]$, *the eigenvalues of function* k_f *with respect to probability measure* μ *are* $\{\lambda_{1j}\}_{j=1}^{\infty}=\{C_k e^{-b_k j}\}_{j=1}^{\infty}$, *where* $b_k>0$ *and* $C_k<\infty$. *Without loss of generality, assume* $C_k\le 1$.

Assumption 3b. *For each* $k\in[K]$, *the eigenvalues of function* k_f *with respect to probability measure* μ *are* $\{\lambda_{1j}\}_{j=1}^{\infty}=\{C_k j^{-2b_k}\}_{j=1}^{\infty}$, *where* $b_k>\frac{\sqrt{21}+3}{4}$ *and* $C_k<\infty$. *Without loss of generality, assume* $C_k\le 1$.

Remark 6. *Assumption 3a is satisfied by smooth kernels such as RBF kernels and Assumption 3b is satisfied by the non-smooth kernels such as Matérn kernels.*

4.2 Homogeneous Setting

We first assume that data across all devices are generated from the same underlying process or distribution (i.e., homogeneous data). Mathematically, it indicates [65]

$$\lim_{N_1,\ldots,N_k\to\infty}\left|\sum_{k=1}^{K}p_kL_k(\boldsymbol{\theta}^*;D_k)-\sum_{k=1}^{K}p_kL_k(\boldsymbol{\theta}_k^*;D_k)\right|=0.$$

We briefly parse this expression. Since the data distribution across all devices is homogeneous, we know, for each k, $\boldsymbol{\theta}_k^*=\boldsymbol{\theta}^*$ as $N_k\to\infty$. Therefore, $\sum_{k=1}^{K}p_kL_k(\boldsymbol{\theta}^*;D_k)=\sum_{k=1}^{K}p_kL_k(\boldsymbol{\theta}_k^*;D_k)$. In Sec. 4.3 and Appendix 3, we will consider the heterogeneous data settings, which are often more realistic in real-world applications.

To derive the convergence result, we divide $[g_k(\boldsymbol{\theta};\xi_k)]_1$ by a constant factor $s_1(M_k)=\tau\log M_k$ and $[g_k(\boldsymbol{\theta};\xi_k)]_2$ by $s_2(M_k)=M_k$, where $[g_k(\boldsymbol{\theta};\xi_k)]_i$ is the i-th component in the stochastic gradient. Those scaling factors are introduced to ensure $[g_k(\boldsymbol{\theta};\xi_k)]_1$ and $[g_k(\boldsymbol{\theta};\xi_k)]_2$ have the same scale in the theoretical analysis.

Remark 7. *The aforementioned scaling factors are only needed for convergence results. In practice, we observe that those factors* $s_1(M_k),s_2(M_k)$ *have minimal influence on the model performance.*

Our first Theorem shows that FGPR using RBF kernels converges if all devices participated in the training.

Theorem 1. *(RBF kernels, synchronous update) Suppose Assumptions 1-3a hold. At each communication round, assume* $|\mathcal{S}|=K$. *If* $\eta^{(t)}=\mathcal{O}(\frac{1}{t})$ *(i.e., a decay learning rate scheduler), then for some constants* $\beta_1,C_{\boldsymbol{\theta}},c_{\boldsymbol{\theta}}>0,\epsilon_k\in(0,\frac{1}{2})$, *when* $M_k>C_{\boldsymbol{\theta}}$, *at iteration* T, *with probability at least*

$\min_k \left(1 - C_{\boldsymbol{\theta}} T \exp\left\{-c_{\boldsymbol{\theta}} \left(\log M_k\right)^{2\epsilon_k}\right\}\right)$,

$$\left|\bar{\theta}_1^{(T)} - \theta_1^*\right|^2 + \left|\bar{\theta}_2^{(T)} - \theta_2^*\right|^2 \leq \frac{2\beta_1^2\left(8(E-1)^2+2\right)G^2}{T+1} + \mathcal{O}\left(\max_k \frac{\log M_k}{M_k} + \sum_{k=1}^K p_k (\log M_k)^{\epsilon_k - \frac{1}{2}}\right),$$

and with probability at least

$$\min_k \left(1 - C_{\boldsymbol{\theta}} \left(\log\left(M_k^{\epsilon_k - \frac{1}{2}}\right)\right)^4 T \exp\left\{-c_{\boldsymbol{\theta}} M_k^{2\epsilon_k}\right\}\right),$$

$$\left\|\bar{\theta}_2^{(T)} - \theta_2^*\right\|_2^2 \leq \frac{2\beta_1^2\left(8(E-1)^2+2\right)G^2}{T+1} + \mathcal{O}\left(\max_k \frac{\log M_k}{M_k} + \sum_{k=1}^K p_k M_k^{\epsilon_k - \frac{1}{2}}\right).$$

Here, constants $\beta_1, C_{\boldsymbol{\theta}}, c_{\boldsymbol{\theta}}$ only depend on $\theta_{min}, \theta_{max}$ and $\{b_k\}_{k=1}^K$.

Remark 8. *Recall that T is the number of iterations. Theorem 1 implies that, when the batch size is large enough, then with a high probability, the parameter iterate converges to the global optimal parameter at a rate of $\mathcal{O}(\frac{1}{T})$. This is credited to the unique structure of the $\mathcal{GP}$ objective function, which we refer to as relaxed convexity (See Lemma 4 and Lemma 5 in the Appendix 4).*

Remark 9. *In the upper bound, there is a term $\frac{2\beta_1^2\left(8(E-1)^2+2\right)G^2}{T+1} \sim \frac{(E-1)^2}{T+1}$, where E is the number of local SGD steps. To ensure this term decreases with respect to T, one needs to ensure E does not exceed $\Omega(\sqrt{T})$. Otherwise, the FGPR will not converge. For instance, if $E = T$, then the FGPR is equivalent to the one-shot communication approach [44]. Furthermore, since $T = RE - 1$, we also know $\frac{(E-1)^2}{T+1} = \frac{(E-1)^2}{RE-1+1} \sim \frac{E-1}{R}$. This hints that the number of communication round R should be greater than E.*

Remark 10. *In addition to the $\mathcal{O}(\frac{1}{T})$ term, there is also a statistical error term $\mathcal{O}(\max_k \frac{\log M_k}{M_k} + \sum_{k=1}^K p_k M_k^{\epsilon_k - \frac{1}{2}})$ that appeared in the upper bound. Theoretically, it indicates that a large batch size is capable of reducing errors in parameter estimation.*

Remark 11. *From Theorem 1, it can be seen that $\left\|\bar{\theta}_2^{(T)} - \theta_2^*\right\|_2^2$ has smaller error term $\mathcal{O}\left(\sum_{k=1}^K p_k M_k^{\epsilon_k - \frac{1}{2}}\right)$ than $\mathcal{O}\left(\sum_{k=1}^K p_k (\log M_k)^{\epsilon_k - \frac{1}{2}}\right)$. This implies that the noise parameter θ_2 is easier to estimate than θ_1. This is intuitively understandable due to the different eigenvalue structures dictated by k_f compared to $\mathbb{I}_{x_1 = x_2}$.*

Next, we study the convergence behavior under the asynchronous update (i.e., partial device participation) framework. In this scenario, only a portion of devices is actively sending their model parameters to the central server at each communication round.

Theorem 2. *(RBF kernels, asynchronous update) Suppose Assumptions 1-3a hold. At each communication round, assume $|\mathcal{S}| = K_{sample} < K$ number of devices are sampled according to the sampling probability p_k. If $\eta^{(t)} = \mathcal{O}(\frac{1}{t})$, then for some constants $C_{\boldsymbol{\theta}}, c_{\boldsymbol{\theta}} > 0, \epsilon_k \in (0, \frac{1}{2})$, when $M_k > C_{\boldsymbol{\theta}}$, at iteration T, with probability at least $\min_k \left(1 - C_{\boldsymbol{\theta}} T \exp\left(-c_{\boldsymbol{\theta}} \left(\log M_k\right)^{2\epsilon_k}\right)\right)$,*

$$\mathbb{E}_{\mathcal{S}}\left\{\left|\bar{\theta}_1^{(T)} - \theta_1^*\right|^2 + \left|\bar{\theta}_2^{(T)} - \theta_2^*\right|^2\right\} \leq \frac{2\beta_1^2\left(\frac{1}{|\mathcal{S}|}4E^2 + 8(E-1)^2 + 2\right)G^2}{T+1} + \mathcal{O}\left(\max_k \frac{\log M_k}{M_k} + \sum_{k=1}^K p_k (\log M_k)^{\epsilon_k - \frac{1}{2}}\right),$$

and with probability at least

$$\min_k \left(1 - C_{\boldsymbol{\theta}} \left(\log\left(M_k^{\epsilon_k - \frac{1}{2}}\right)\right)^4 T \exp\left\{-c_{\boldsymbol{\theta}} M_k^{2\epsilon_k}\right\}\right),$$

$$\mathbb{E}_{\mathcal{S}}\left\{\left\|\bar{\theta}_2^{(T)} - \theta_2^*\right\|_2^2\right\} \leq \frac{2\beta_1^2\left(\frac{1}{|\mathcal{S}|}4E^2 + 8(E-1)^2 + 2\right)G^2}{T+1} + \mathcal{O}\left(\max_k \frac{\log M_k}{M_k} + \sum_{k=1}^K p_k M_k^{\epsilon_k - \frac{1}{2}}\right),$$

where the expectation is taken over the set $\mathcal{S}$, and please refer to Appendix 5.3 for a rigorous definition.

Remark 12. *Under the asynchronous update setting, a similar convergence guarantee holds. The only difference is that the number of active devices $|\mathcal{S}|$ plays a role in the upper bound. Numerically, the ratio $\frac{E^2}{|\mathcal{S}|}$ enlarges the upper bound and impedes the convergence rate. As $|\mathcal{S}|$ grows (i.e., more devices participate in the training), the ratio $\frac{E^2}{|\mathcal{S}|}$ decreases.*

Our next theorem provides explicit convergences rate for FGPR with Matérn kernels under both a synchronous and asynchronous update scheme.

Theorem 3. *(Matérn kernels) Suppose Assumptions 1-2 and 3b hold,*

(1) At each communication round, assume $|\mathcal{S}| = K$. If $\eta^{(t)} = \mathcal{O}(\frac{1}{t})$, then for some constants $C_{\boldsymbol{\theta}}, c_{\boldsymbol{\theta}} > 0, \beta_1 > 0, b_k > \frac{(\sqrt{21}+3)}{4}$ and $0 < \alpha_k < \frac{1}{2}$, when $M_k > C_{\boldsymbol{\theta}}$, with probability at least $\min_k \left(1 - C_{\boldsymbol{\theta}} T (\log(M_k^{\epsilon_k - \frac{1}{2}}))^4 \exp\{-c_{\boldsymbol{\theta}} M_k^{2\epsilon_k}\}\right)$,

$$\left\|\bar{\theta}_2^{(T)} - \theta_2^*\right\|_2^2 \leq \frac{2\beta_1^2\left(8(E-1)^2+2\right)G^2}{T+1} + \mathcal{O}\left(\max_k M_k^{-\frac{8b_k^2 - 12b_k - 6 - 3\alpha_k - 4\alpha_k b_k}{8b_k^2 - 4b_k}}\right) + \mathcal{O}\left(\sum_{k=1}^K p_k M_k^{\epsilon_k - \frac{1}{2}}\right).$$

Additionally,

$$\left\|\nabla L(\bar{\boldsymbol{\theta}}^{(T)})\right\|_2^2 \leq \frac{2\beta_1^2\left(8(E-1)^2+2\right)G^2}{4\theta_{min}^4(T+1)} + \mathcal{O}\left(\max_k \left\{M_k^{\frac{(2+\alpha_k)(4b_k+3)}{4b_k(2b_k-1)} - 1} + \sum_{k=1}^K p_k M_k^{\epsilon_k - \frac{1}{2}}\right\}\right).$$

(2) At each communication round, assume $|\mathcal{S}| = K_{sample}$, number of devices are sampled according to the sampling

probability p_k. *If* $\eta^{(t)} = \mathcal{O}(\frac{1}{t})$, *then for some constants* $C_{\boldsymbol{\theta}}, c_{\boldsymbol{\theta}} > 0$, $\beta_1 > 0, b_k > \frac{(\sqrt{21}+3)}{4}$ *and* $0 < \alpha_k < \frac{1}{2}$, *when* $M_k > C_{\boldsymbol{\theta}}$, *with probability at least* $\min_k \left(1 - C_{\boldsymbol{\theta}} T \left(\log\left(M_k^{\epsilon_k - \frac{1}{2}}\right)\right)^4 \exp\{-c_{\boldsymbol{\theta}} M_k^{2\epsilon_k}\}\right)$,

$$\begin{aligned}
&\mathbb{E}_{\mathcal{S}}\left\{\left\|\bar{\theta}_2^{(T)} - \theta_2^*\right\|_2^2\right\} \\
&\leq \frac{2\beta_1^2\left(\frac{4E^2}{|\mathcal{S}|} + 8(E-1)^2 + 2\right)G^2}{T+1} \\
&\quad + \mathcal{O}\left(\max_k M_k^{-\frac{8b_k^2 - 12b_k - 6 - 3\alpha_k - 4\alpha_k b_k}{8b_k^2 - 4b_k}}\right) \\
&\quad + \mathcal{O}\left(\sum_{k=1}^K p_k M_k^{\epsilon_k - \frac{1}{2}}\right).
\end{aligned}$$

Additionally,

$$\begin{aligned}
&\mathbb{E}_{\mathcal{S}}\left\{\left\|\nabla L(\bar{\boldsymbol{\theta}}^{(T)})\right\|_2^2\right\} \\
&\leq \frac{2\beta_1^2\left(\frac{4E^2}{|\mathcal{S}|} + 8(E-1)^2 + 2\right)G^2}{4\theta_{min}^4(T+1)} \\
&\quad + \mathcal{O}\left(\max_k\left\{M_k^{\frac{(2+\alpha_k)(4b_k+3)}{4b_k(2b_k-1)}-1} + \sum_{k=1}^K p_k M_k^{\epsilon_k - \frac{1}{2}}\right\}\right).
\end{aligned}$$

Remark 13. *It can be seen that the* FGPR *using Matérn kernel has a larger statistical error than the one using RBF kernel. In the RBF kernel, the statistical error is partially affected by* $\mathcal{O}\left(\max_k \frac{logM_k}{M_k}\right)$ *(Theorems 1,2) while this term becomes* $\mathcal{O}\left(\max_k M_k^{-\frac{8b_k^2 - 12b_k - 6 - 3\alpha_k - 4\alpha_k b_k}{8b_k^2 - 4b_k}}\right)$ *in the Matérn kernel. The latter one is larger since* $b_k > \frac{(\sqrt{21}+3)}{4}$ *and* $\alpha_k \in (0, 0.5)$. *This difference arises from the fact that the Matérn kernel has a slower eigenvalue decay rate (determined by* b_k*) than the RBF kernel (i.e., polynomial vs. exponential). This slow decay rate leads to slower convergence and larger statistical error. When* b_k *becomes larger, the decay rate becomes faster, and the influence of* $\mathcal{O}\left(\max_k M_k^{-\frac{8b_k^2 - 12b_k - 6 - 3\alpha_k - 4\alpha_k b_k}{8b_k^2 - 4b_k}}\right)$ *gets smaller. In this case, the statistical error is dominated by* $\mathcal{O}\left(\sum_{k=1}^K p_k M_k^{\epsilon_k - \frac{1}{2}}\right)$, *which is the same as the one in the RBF kernel.*

Remark 14. *In addition to the convergence bound on parameter iterates, we also provide an upper bound on the full gradient* $\left\|\nabla L(\bar{\boldsymbol{\theta}}^{(T)})\right\|_2^2$. *This bound scales the same as the bound for* $\left\|\bar{\theta}_2^{(T+1)} - \theta_2^*\right\|_2^2$.

Remark 15. *For Matérn kernel, there is no explicit convergence guarantee for parameter* $\bar{\theta}_1$. *The reason is that it is very hard to derive the lower and upper bounds for the SG for Matérn kernel. However, Theorem 3 shows that both* $\bar{\theta}_2$ *and the full gradient converge at rates of* $\mathcal{O}(\frac{1}{T})$ *subject to statistical errors.*

4.3 Heterogeneous Setting

Besides the homogeneous setting, we further consider the scenario where data from all devices are generated from several different processes or distributions. Equivalently, this indicates

$$\mathbb{P}\left(\left|\sum_{k=1}^K p_k L_k(\boldsymbol{\theta}^*; D_k) - \sum_{k=1}^K p_k L_k(\boldsymbol{\theta}_k^*; D_k)\right| = 0\right) = 0.$$

Since the data are heterogeneous, we know $\boldsymbol{\theta}_k^* \neq \boldsymbol{\theta}^*$. As a result, the weighted average of $L_k(\boldsymbol{\theta}_k^*; D_k)$ can be very different from $L(\boldsymbol{\theta}^*)$. We here note that convergence results for the heterogeneous setting are moved to Appendix 5.5, due to space limitations.

Overall, in this theoretical section, we show that the FGPR is guaranteed to converge under both homogeneous setting (Sec. 4.2) and heterogeneous setting (Appendix 5), regardless of the synchronous updating or the asynchronous updating.

5 Proof of Concept

We start by validating the theoretical results obtained in Sec. 4.2. We also provide sample experiments that shed light on key properties of FGPR.

Example 1: Homogeneous Setting with Balanced Data. We generate data from a $\mathcal{GP}$ with zero-mean and both a RBF and Matérn$-3/2$ kernel. We consider $\theta_1 \in [0.1, 10]$, $\theta_2 \in [0.01, 1]$ and a length parameter $\boldsymbol{l} \in [0.01, 1]^d$. The input space is a d-dimensional unit cube $[0,1]^d$ in $\mathbb{R}^d$ with $d \in \{1, \ldots, 10\}$ and the dimension of the output is one. We conduct 20 independent experiments. In each experiment, we first randomly sample $\theta_1, \theta_2, \boldsymbol{l}$ and d to generate data samples from the $\mathcal{GP}$. In each scenario, we set $N_k = \frac{N}{K}$. This setting is homogeneous and balanced as the number of data points across K clients is equal, and they all come from the same underlying stochastic process. We consider three scenarios: (1) $K = 20, N = 5000$, (2) $K = 50, N = 2000$, (3) $K = 100, N = 800$. Results from the RBF kernel are provided in Figure 2. Due to space limitation, we move plots of the Matérn Kernel into Appendix 7. It can be seen that the convergence rate follows a $\mathcal{O}(\frac{1}{T})$ pattern. In some runs, the values of $\left\|\bar{\theta} - \theta^*\right\|_2^2$ are very large at the beginning. Those imply that initial parameters are far away from true parameters. However, after 20-40 communication rounds, those values quickly diminish. In 2, we also observe that plots in (c) are more dispersed and fluctuated than (a) and (b). This is because each device only has fewer data points ($N/K = 2000/100 = 20$).

Example 2: Homogeneous Setting with Unbalanced Data. We use the same data-generating strategy as Example 1, but the sample sizes are unbalanced. Specifically, the number of data points in each device ranges from 10 to 10,000. The histogram of data distribution from one experiment is given in Figure 4. The convergence curves are plotted in Figure 3. Again, the convergence rate agrees with our theoretical finding. This simple example reveals a critical property of FGPR: FGPR can help devices with few observations recover true parameters (subject to statistical errors) or reduce prediction errors. We will further demonstrate this advantage in the heterogeneous setting in Sec. 6.

Example 3: The Ability to Recover Accurate Predictions for a Badly Initialized $\mathcal{GP}$. When training an FL algorithm, it is not uncommon to initialize the model parameters $\boldsymbol{\theta}$ near a bad stationary point. Here, we provide one toy

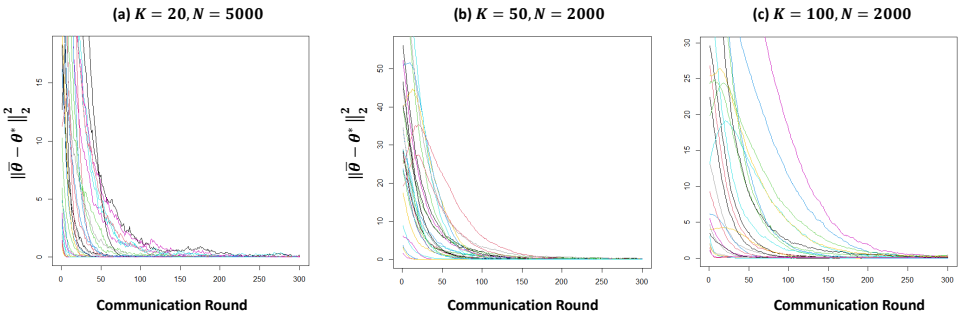

Fig. 2. (RBF kernel) Evolution of $\left\|\bar{\theta} - \theta^*\right\|_2^2$ over training epochs. In the plot, each color represents an independent run. The input dimension d is different for each run and $d \in \{1, \ldots, 10\}$.

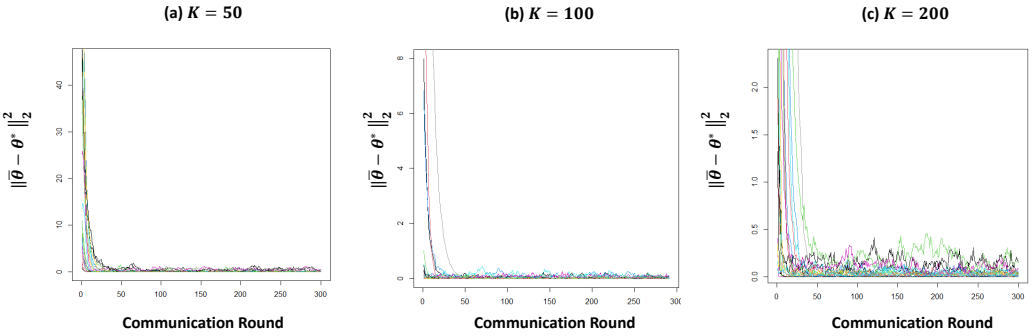

Fig. 3. (RBF kernel) Evolution of $\left\|\bar{\theta} - \theta^*\right\|_2^2$ over training epochs using unbalanced data. In the plot, each color represents an independent run. The input dimension d is different for each run and $d \in \{1, \ldots, 10\}$.

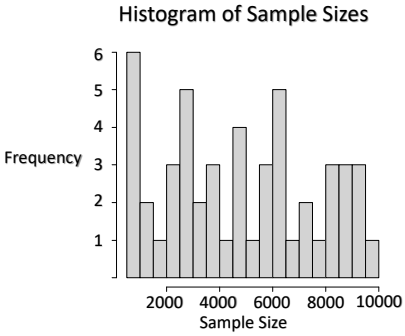

Fig. 4. Histogram of Sample Sizes (Example 2).

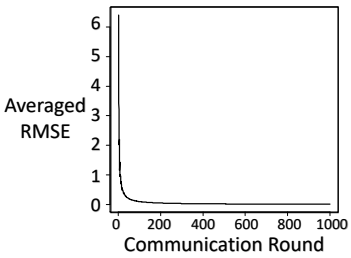

Fig. 5. Evolution of the averaged RMSE in Example 3.

example. We simulate data from $y = sin(x) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, 0.2)$ and create two clients ($K = 2$). Each client has 100 training data points and $1,000$ testing data points that are uniformly sampled from $[0, 1]$. We artificially find a bad initial parameter $\boldsymbol{\theta}$ such that the fitted curve is just a flat line. This can be achieved by finding a $\boldsymbol{\theta}$ whose noise parameter θ_2 is large. In this case, $\boldsymbol{\theta} = (1, 10, 1)$ where the $\mathcal{GP}$ interprets all data as noise and simply returns a flat line.

We evaluate the predictive performance of `FGPR` using the averaged root-mean-square error (RMSE) metric. The RMSE for each device is evaluated on the local testing data, and the averaged RMSE averages RMSEs across all devices. We find that `FGPR` is robust to parameter initialization. We plot the evolution of averaged RMSE versus training epoch in Figure 5. It can be seen that even when the parameter is poorly initialized, `FGPR` can still correct the wrong initialization after several communication rounds. This credits to the stochasticity in the SGD method. It is known that, in ERM, SGD can escape bad stationary solutions and converge to solutions with good generalization (often flat ones) [59].

Example 4: Scalability to big data. We followed the setup presented in Sec. 4.1 (Regression) of Moreno-Muñoz et al. [41]. More specifically, Moreno-Muñoz et al. [41] define a generating function

$$f(x) = \frac{9}{2}\cos(2\pi x + \frac{3\pi}{2}) - 3\sin(\frac{43\pi}{10}x + \frac{3\pi}{10})$$

and define the client-specific data generating function $f_k(x) = f(x) + \epsilon$ where $\epsilon \sim \mathcal{N}(0, 2)$. The input domain is $x \in [0, 5.5]$. We generated **one million data points** for each device $k \in \{1, \ldots, 50\}$. For each device, we randomly

selected 60% of the data points for the training dataset (0.6 million data points), while the remaining data points were included in the testing data (0.4 million data points). This results in a total of **30 million data points.** We set the batch size to 32, the learning rate to 0.1, and the number of communication rounds to 50. We repeated the experiment 30 times. The Averaged RMSEs (line 1) and the standard deviation of RMSEs (line 2) across all devices, along with running times, are reported in Table 2.

Model	FGPR	Modular $\mathcal{GP}$
Averaged RMSEs	1.56 (0.07) 0.12 (0.01)	1.95 (0.09) 0.13 (0.01)
Total Running Times (in seconds)	3846($\pm$495)	5085($\pm$605)

TABLE 2
Experimental results on 30 million data

It can be seen that FGPR can efficiently handle large datasets (30 million data points in total) while generating satisfactory prediction performance. This credits to two properties of FGPR: first, FGPR employs mini-batch SGD that only takes a subset of data during each iteration. This circumvents the need to invert a huge matrix. Second, FGPR avoids using inducing points that scale with sample size.

6 Application I: Multi-fidelity Modeling

For many computer experiments, high-fidelity numerical simulations of complex physical processes typically require a significant amount of time and budget. This limits the number of data points researchers can collect and affects the modeling accuracy due to insufficient data. A major work trend has been proposed to augment the expensive data source with cheaper surrogates to overcome this hindrance. Multi-fidelity models are designed to fuse scant but accurate observations (i.e., high-fidelity, HF) with cheap and biased approximations (i.e., low-fidelity, LF) to improve the HF model performance.

Denote by f_h a high-fidelity function and f_l a low-fidelity function. Multi-fidelity approaches [66, 67, 68] aim to use f_l to better predict f_h. During the past decades, many multi-fidelity models have been proposed to fulfill this goal. We refer to [69] and [70] for detailed literature reviews. Among all the methods, $\mathcal{GP}$-based approaches have caught the most attention due to their ability to incorporate prior beliefs, interpolate complex functional patterns, and quantify uncertainties [69]. The last ability is critical to fuse observations across different fidelities effectively.

Within many applications, two specific models have been shown to be very competitive [68]; the auto-regressive (AR) and the Deep $\mathcal{GP}$ (Deep) approaches. Both approaches model f_h as shown below

$$f_h(x) = \rho(f_l(x), x) + \Delta(x),$$

where $\rho(\cdot,\cdot)$ is a space-dependent non-linear transformation and $\Delta(x)$ is a bias term modeled through a $\mathcal{GP}$.

More specifically, the AR model [71] sets the transformation as a linear mapping such that $\rho(f_l(x), x) = \rho_c f_l(x)$, where ρ_c is a constant. It then imposes a $\mathcal{GP}$ prior on f_l and accordingly obtains its posterior f_l^*. As a result, one can derive the closed-form posterior distribution $p(f_h|f_l^*, x, y)$ and obtain the posterior predictive equation of the high-fidelity model. On the other hand, the Deep model [67] treats $\rho(f_l(x), x)$ as a deep Gaussian process to uncover highly complex relationships among f_l and f_h . Deep is one of the state-of-the-art multi-fidelity models. For more details, please refer to [68].

Nowadays, as data privacy gains increased importance, having access to data across multiple fidelities is often impractical as multiple clients can own data. This imposes a key challenge in multi-fidelity modeling approaches as effective inference on expensive high-fidelity models often necessitates the need to borrow strength from other information sources. Fortunately, in such a case, FGPR is a potential candidate that learns a $\mathcal{GP}$ prior without sharing data.

In this section, we test the viability of FGPR in multi-fidelity modeling. We test our approach using settings where local devices contain data with different fidelities. We then use Algorithm 1 to train our FGPR algorithm. Specifically, each device runs several steps of SGD and then sends its model parameter to the central orchestrator. The orchestrator then aggregates model parameters and sends the aggregated parameter back to each device. This procedure is repeated several times till some exit condition is met. Upon estimating the model parameters, we then test the local predictive accuracy using the predictive equation (1) for device k.

We benchmark FGPR with several state-of-the-art models. Interestingly, our results (Table 3) show that FGPR not only preserves privacy but also can provide superior performance than centralized multi-fidelity approaches.

Below we detail the benchmark models: (1) Separate which fits a single $\mathcal{GP}$ to the HF dataset without any communication. This means the HF dataset does not use any information from the LF dataset; (2) the AR method [71]. AR is the most classical and widely-used multi-fidelity modeling approach [66, 69, 72]; (3) the Deep model [73] highlighted above; (4) Modular $\mathcal{GP}$ [41] that models each fidelity-level as an output. For this method, we introduce 20 inducing points for each device and 3 global latent variables. All output values are standardized to mean 0 and variance 1.

We start with two simple illustrative examples from [67] and then benchmark all models on five well-known models in the multi-fidelity literature.

Example 1: Linear Example - We first present a simple one dimensional linear example where $x \in [0, 1]$. The low and high-fidelity models are given by [67]

$$y_l(x) = \frac{1}{2} y_h(x) + 10(x - \frac{1}{2}) + 5,$$
$$y_h(x) = (6x - 2)^2 \sin(12x - 4),$$

where $y_l(\cdot)$ is the output from the LF model and $y_h(\cdot)$ is the output from the HF model. We simulate 100 data points from the LF model and 20 data points from the HF model. The number of testing data points is 1,000.

Example 2: Nonlinear Example - The one dimensional non-linear example for $x \in [0, 2]$ is given as

$$y_l(x) = \cos(15x),$$
$$y_h(x) = x \exp^{y_l(2x-0.2)} - 1.$$

We use the same data-generating strategy in Example 1.

The results from both examples are plotted in Figure

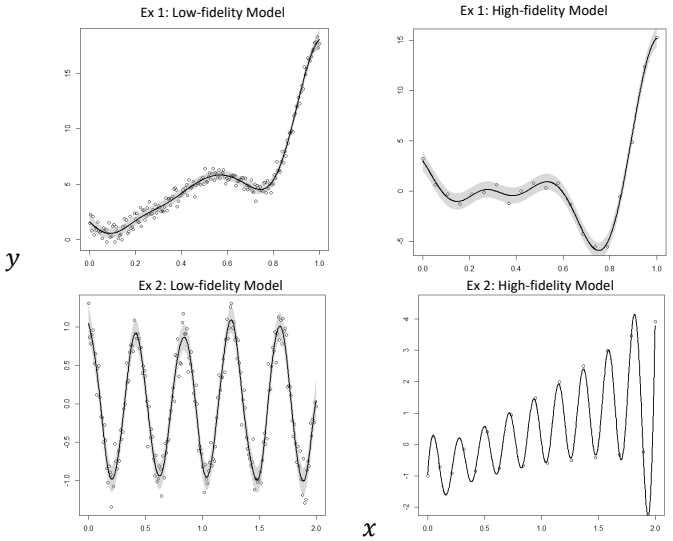

Fig. 6. Results of Example 1 and 2. The solid black line denotes the predicted mean, and the grey area is a 95% confidence interval.

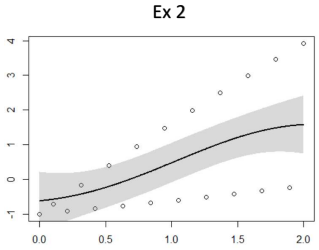

Fig. 7. Results of Example 2 using `separate` on the HF data only.

6. The results provide a simple proof-of-concept that the learned FGPR is able to accurately predict the HF model despite sparse observations. Additionally, FGPR can also adequately capture uncertainties (grey areas in Figure 6) in predictions. The results also confirm our insights on automatic personalization in Sec. 3.3 whereby a single global model was able to adequately fit both HF and LF datasets. Here, we conduct one additional comparison study on Example 2. We train a $\mathcal{GP}$ model solely using a high-fidelity dataset. The fitted curve is plotted in Figure 7. It can be seen that, without borrowing any information from the LF dataset, the fitted $\mathcal{GP}$ curve fails to recover the true underlying pattern. This example further demonstrates the advantage of FGPR: the shared global model parameter encodes key information (e.g., trend, pattern) from the low-fidelity dataset such that the high-fidelity dataset can exploit this information to fit a more accurate surrogate model.

Next, we consider a range of benchmark problems widely used in the multi-fidelity literature [67, 68]. We defer the full specifications of those problems to the Appendix 8. For each experiment, we generate 1,000 testing points uniformly on the input domain.

- **CURRIN:** CURRIN [33, 74] is a two-dimensional function that is widely used for multi-fidelity computer simulation models.
- **PARK:** The PARK function [74, 75] lies in a four-dimensional space ($\boldsymbol{x} \in (0,1]^4$). This function is often used in testing for parameter calibration and design of experiments.
- **BRANIN:** BRANIN is widely used as a test function for metamodeling in computer experiments. In this example, there are three fidelity levels [67, 76].
- **Hartmann-3D:** Similar to BRANIN, this is a 3-level multi-fidelity dataset where the input space is $[0,1]^3$.
- **Borehole Model:** The Borehole model is an 8-dimensional physical model that simulates water flow through a borehole [74, 77, 78].

Each experiment is repeated 30 times, and we report RMSEs of the model performance on the true HF model, along with the standard deviations in Table 3. The training data size is highlighted in the table.

First, it can be seen in Table 3, FGPR consistently yields smaller RMSE than Separate. This confirms that FGPR is able to borrow strength across multi-fidelity datasets. More importantly, we find that FGPR can even achieve superior performance compared to the AR and Deep benchmarks. This implies that one can avoid centralized approaches without compromising accuracy. Finally, the inferior performance of Modular $\mathcal{GP}$s is because: (1) Modular $\mathcal{GP}$ optimizes a lower bound of log-marginal likelihood instead of the exact likelihood. FGPR, on the other hand, directly performs stochastic optimization on the exact likelihood; (2) Modular $\mathcal{GP}$ is a one-shot approach. For instance, the convergence bound of FedAvg follows $\mathcal{O}(E^2/T)$ where E is the number of local steps and $T = RE - 1$ where R is the number of communication rounds. Clearly, E should be small (less than the order of $\mathcal{O}(R)$) to guarantee convergence. A similar result is shown in FGPR in Sec. 4. Whereas our model FGPR is a collaborative process where the global model is updated over R communication rounds; (3) Modular $\mathcal{GP}$ require one additional layer of approximation that sacrifices accuracy [41]. As a side note, as mentioned in Table 1, in Modular $\mathcal{GP}$, a sparse representation of the local data is shared, which entails the pseudo-targets, variational density, model parameters, and lower bound value. Clearly, if the sparse approximation is close to the true local posterior, there is an infringement on local privacy. FGPR, on the other hand, only shares model parameters.

In summary, the results show that FGPR can serve as a compelling candidate for privacy-preserving multi-fidelity modeling in the modern era of statistics and machine learning.

Below, we also detail an interesting technical observation.

Remark 16. *In our settings, the weight coefficient p_k for the HF is low compared to LF, as HF clients have fewer data. For instance, in the CURRIN example, the HF coefficient is $p_1 = \frac{40}{240} = 0.17$. Therefore, the global parameter is averaged with higher weights for the LF model. Yet, the model excels in predicting the HF model. This again goes back to the fact that, unlike deep learning-based FL approaches, FGPR is learning a joint prior on the functional space. The scarce HF data alone cannot learn a strong prior, yet with the help of the LF data, such prior can be learned effectively. That being said, it may be interesting to investigate the adaptive assignment of p_k, yet this requires additional theoretical analysis.*

On par with Remark 17, we conduct an ablation study on p_i using the CURRIN function. Specifically, we use the

TABLE 3
RMSEs and standard deviations compared to the true HF model. Each experiment is repeated 30 times. The sample size is in a format of HF/MF/LF, where MF represents a medium-fidelity model.

RMSE-HF	Sample Size	**FGPR**	Separate	AR	Deep	Modular $\mathcal{GP}$
CURRIN	40/0/200	$\mathbf{0.148 \pm 0.056}$	0.301 ± 0.080	0.295 ± 0.052	0.252 ± 0.064	0.243 ± 0.033
PARK	50/0/300	$\mathbf{0.012 \pm 0.002}$	0.052 ± 0.006	0.035 ± 0.001	0.013 ± 0.001	0.039 ± 0.001
BRANIN	20/40/200	0.260 ± 0.065	0.374 ± 0.089	0.335 ± 0.070	$\mathbf{0.213 \pm 0.085}$	0.365 ± 0.076
Hartmann-3D	50/100/200	$\mathbf{0.365 \pm 0.074}$	0.456 ± 0.087	0.412 ± 0.067	0.383 ± 0.092	0.438 ± 0.085
Borehole	50/0/200	$\mathbf{0.604 \pm 0.006}$	0.633 ± 0.006	0.615 ± 0.005	0.622 ± 0.007	0.621 ± 0.004

same sample size (i.e., $N_1 = 40$, $N_2 = 200$), but we gradually increase p_1 from 0.17 to 1 and decrease p_2 from 0.83 to 0. We plot the RMSE versus p_1 in Figure 8. It can be seen that the RMSE remains consistent when we moderately increase p_1. However, once p_1 passes a threshold, the RMSE increases sharply. Again this is because the increased weight to HF can be misleading due to the scarcity of HF data.

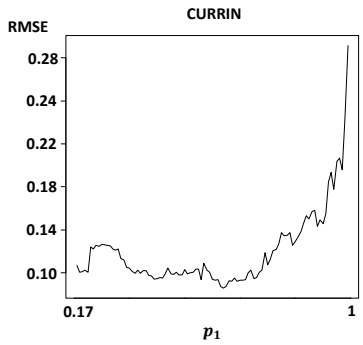

Fig. 8. Ablation Study (CURRIN).

7 Application II: Robotics

We now test the performance of FGPR on a robotic dataset [Link].

To enable accurate robot movement, one needs to control the joint torques [79]. Joint torques can be computed by many existing inverse dynamics models. However, in real-world applications, the underlying physical process is highly complex and often hard to derive using first principles. Data-driven models were proposed as an appealing alternative to handle complex functional patterns and, more importantly, quantify uncertainties [80]. The goal of this section is to test FGPR as a data-driven approach to accurately compute joint torques at different joint positions, velocities, and accelerations.

To this end, we test FGPR using a Matérn-3/2 kernel on learning an inverse dynamics problem for a seven degrees-of-freedom SARCOS anthropomorphic robot arm [64, 81]. This task contains $d = 21$ dimensional input and 7 dimensional output with 44,484 points for training and 4,449 points for testing. Since FGPR is a single-output FL framework, we only use one output each time (See Table 4). Our goal is to accurately predict the forces used at different joints given the joints' input information. We randomly partition the data into 25 devices. Overall, each device has around 1850 training points and 180 testing points each.

We benchmark FGPR with (1) neural network; (2) DGP [43] that uses the product-of-experts approximation and distributes learning tasks to different experts (i.e., nodes); (3) DVI [42] that performs distributed variational inference.

We found that a neural network trained from a simple FedAvg failed. This is due to the large heterogeneity. To resolve this issue, we train Neural using a state-of-the-art personalized FL framework Ditto [12]. In Ditto, each local device solves two optimization problems. The first is the same as FedAvg and to find $\boldsymbol{\theta}$, while the second derives personalized parameters $\boldsymbol{v}_k$ for each client k by solving

$$\min_{\boldsymbol{v}_k} h_k(\boldsymbol{v}_k; \boldsymbol{\theta}) := \hat{R}_k(\boldsymbol{v}_k; D_k) + \frac{\lambda}{2} \|\boldsymbol{v}_k - \boldsymbol{\theta}\|_2^2$$

where λ is a regularization parameter and $\boldsymbol{\theta}$ is the shared global parameter. The idea behind Ditto is clear: in addition to updating a shared global parameter $\boldsymbol{\theta}$, each device also maintains its own personalized solution $\boldsymbol{v}_k$. Yet, the regularization term ensures that this $\boldsymbol{v}_k$ should be close to $\boldsymbol{\theta}$ such that one can retain useful information learned from a global model.

For DVI and DGP, we use Matérn-3/2 kernels and introduce 1024 inducing points for the former method.

In Table 4, we present results for outputs 1, 3, 5, and 7. Here, note that the RMSEs of DVI and DGP are evaluated on the central location using all testing data rather than on each node. This is because the goal of DVI or DGP is to distribute learning tasks and speed up training rather than improve the model performance on each local node. Whilst for FGPR and Neural, we can additionally obtain the standard error of RMSEs across devices since predictions are performed on local devices.

Under the heterogeneous setting, FGPR still provides lower averaged RMSE than the personalized Neural, DGP, and DVI benchmark models. This credits to (1) the flexible prior regularization in the $\mathcal{GP}$ regression that can avoid potential model over-fitting; (2) the intrinsic personalization capability of FGPR; (3) FGPR does exact inference whereas DVI and DGP use lower bound and PoE approximate objectives, respectively. Recall that DGP uses the product-of-experts approximation that induces a notion of independence across local experts (devices). DVI uses VI that faces several drawbacks, per our earlier discussion in Sec. 3; (4) DGP is a one-shot approach that is not optimal, as discussed earlier. Here, we note that DVI requires each device to send an $N_z \times N_z \times d$ dimension tensor to the server after every single optimization step. This incurs very heavy communication loads and high costs. Also, DGP shares local predicted output to a central server, and the server can re-construct the data pattern from each device. This clearly leaks the local data information.

An additional case study on NASA Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) tools dataset

TABLE 4
For FGPR & Neural we report averaged RMSE and the standard deviation (std) of RMSEs across all testing devices for the robotics data. Each experiment is repeated 30 times. The standard deviation of each performance measure is reported in brackets. For DVI/DGP, we report the RMSE on a central server

Averaged RMSE ×10 std of RMSE ×10	Output 1	Output 3	Output 5	Output 7
FGPR	**2.75 (0.00)** **1.84 (0.01)**	**2.42 (0.03)** **1.57 (0.01)**	**2.20 (0.05)** **1.29 (0.02)**	**2.38 (0.01)** **1.44 (0.02)**
Neural	3.01 (0.01) 1.70 (0.00)	3.05 (0.06) 2.11 (0.02)	2.89 (0.09) 1.37 (0.02)	2.90 (0.02) 1.50 (0.01)
DVI	2.85 (0.02)	3.32 (0.06)	2.57 (0.03)	2.98 (0.02)
DGP	2.99 (0.03)	3.17 (0.04)	2.62 (0.01)	2.77 (0.02)

[82] that involves multiple engines is deferred to the Appendix 0 due to space limitation. In this case study, we also benchmarked with federated polynomial regression models.

8 Conclusion

In this paper, we extend the standard $\mathcal{GP}$ regression model to a federated setting, FGPR. We use both theory and a wide range of experiments to justify the viability of our proposed framework. We highlight the unique capability of FGPR to provide automatic personalization and strong transferability on untrained devices.

FGPR may find value in meta-learning as it provides an inherent Bayesian perspective on this topic. Other interesting research directions include: (1) Extending the current framework to a multi-output $\mathcal{GP}$ model. The challenge lies in capturing the correlation across output in a federated paradigm. (2) Enlightening theoretical perspective of FGPR. In this work, we only provide theoretical guarantees on noise/variance parameters and the gradient norm. Studying the convergence behavior of length parameters is another crucial but challenging future research direction. (3) Exploring differential private FGPR. Despite federated learning circumventing the need to share data, recent work [83] has demonstrated the potential risk of data reconstruction. In the future, we intend to systematically explore privacy preservation and data leakage prevention within the training process of federated $\mathcal{GP}$s. Our goal is to establish robust theoretical guarantees for these privacy-preserving techniques.

References

[1] R. Kontar, N. Shi, X. Yue, S. Chung, E. Byon, M. Chowdhury, J. Jin, W. Kontar, N. Masoud, M. Noueihed *et al.*, "The internet of federated things (ioft): A vision for the future and in-depth survey of data-driven approaches for federated learning," *arXiv preprint arXiv:2111.05326*, 2021.

[2] O. L. Mangasarian and M. V. Solodov, "Backpropagation convergence via deterministic nonmonotone perturbed minimization," *Advances in Neural Information Processing Systems*, pp. 383–383, 1994.

[3] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[4] V. Smith, S. Forte, M. Chenxin, M. Takáč, M. I. Jordan, and M. Jaggi, "Cocoa: A general framework for communication-efficient distributed optimization," *Journal of Machine Learning Research*, vol. 18, p. 230, 2018.

[5] T. S. Brisimi, R. Chen, T. Mela, A. Olshevsky, I. C. Paschalidis, and W. Shi, "Federated learning of predictive models from federated electronic health records," *International journal of medical informatics*, vol. 112, pp. 59–67, 2018.

[6] N. H. Tran, W. Bao, A. Zomaya, M. N. Nguyen, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 1387–1395.

[7] H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Blockchained on-device federated learning," *IEEE Communications Letters*, vol. 24, no. 6, pp. 1279–1283, 2019.

[8] X. Yue and R. Kontar, "The renyi gaussian process: Towards improved generalization," *arXiv preprint arXiv:1910.06990*, 2019.

[9] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[10] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens," in *International Conference on Machine Learning*. PMLR, 2019, pp. 634–643.

[11] H. Wang, K. Sreenivasan, S. Rajput, H. Vishwakarma, S. Agarwal, J.-y. Sohn, K. Lee, and D. Papailiopoulos, "Attack of the tails: Yes, you really can backdoor federated learning," *arXiv preprint arXiv:2007.05084*, 2020.

[12] T. Li, S. Hu, A. Beirami, and V. Smith, "Ditto: Fair and robust federated learning through personalization," in *International Conference on Machine Learning*. PMLR, 2021, pp. 6357–6368.

[13] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," *International Conference on Learning Representations*, 2019.

[14] M. Mohri, G. Sivek, and A. T. Suresh, "Agnostic federated learning," in *International Conference on Machine Learning*. PMLR, 2019, pp. 4615–4625.

[15] X. Yue, M. Nouiehed, and R. A. Kontar, "Gifair-fl: An approach for group and individual fairness in federated learning," *arXiv preprint arXiv:2108.02741*, 2021.

[16] Y. Zeng, H. Chen, and K. Lee, "Improving fairness via federated learning," *arXiv preprint arXiv:2110.15545*, 2021.

[17] H. Wang, M. Yurochkin, Y. Sun, D. Papailiopoulos, and Y. Khazaeni, "Federated learning with matched averaging," *International Conference on Learning Representations.*, 2020.

[18] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich,

and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *International Conference on Machine Learning*. PMLR, 2020, pp. 5132–5143.

[19] H. Yuan and T. Ma, "Federated accelerated stochastic gradient descent," *arXiv preprint arXiv:2006.08950*, 2020.

[20] N. Shi, F. Lai, R. A. Kontar, and M. Chowdhury, "Fed-ensemble: Improving generalization through model ensembling in federated learning," *arXiv preprint arXiv:2107.10663*, 2021.

[21] S. Rana, C. Li, S. Gupta, V. Nguyen, and S. Venkatesh, "High dimensional bayesian optimization with elastic gaussian process," in *International conference on machine learning*. PMLR, 2017, pp. 2883–2891.

[22] X. Yue and R. A. Kontar, "Why non-myopic bayesian optimization is promising and how far should we look-ahead? a study via rollout," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 2808–2818.

[23] S. Jiang, H. Chai, J. Gonzalez, and R. Garnett, "Binoculars for efficient, nonmyopic sequential experimental design," in *International Conference on Machine Learning*. PMLR, 2020, pp. 4794–4803.

[24] A. Krishna, V. R. Joseph, S. Ba, W. A. Brenneman, and W. R. Myers, "Robust experimental designs for model calibration," *Journal of Quality Technology*, pp. 1–12, 2021.

[25] G. Tapia, A. H. Elwany, and H. Sang, "Prediction of porosity in metal-based additive manufacturing using spatial gaussian process models," *Additive Manufacturing*, vol. 12, pp. 282–290, 2016.

[26] W. Peng, Y.-F. Li, Y.-J. Yang, J. Mi, and H.-Z. Huang, "Bayesian degradation analysis with inverse gaussian process models under time-varying degradation rates," *IEEE Transactions on Reliability*, vol. 66, no. 1, pp. 84–96, 2017.

[27] F. Imani, C. Cheng, R. Chen, and H. Yang, "Nested gaussian process modeling for high-dimensional data imputation in healthcare systems," in *IISE 2018 Conference & Expo, Orlando, FL, May*, 2018, pp. 19–22.

[28] S. Ketu and P. K. Mishra, "Enhanced gaussian process regression-based forecasting model for covid-19 outbreak and significance of iot for its detection," *Applied Intelligence*, vol. 51, no. 3, pp. 1492–1512, 2021.

[29] S. A. Goli, B. H. Far, and A. O. Fapojuwo, "Vehicle trajectory prediction with gaussian process regression in connected vehicle environment," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 550–555.

[30] M. P. Deisenroth, D. Fox, and C. E. Rasmussen, "Gaussian processes for data-efficient learning in robotics and control," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 2, pp. 408–423, 2013.

[31] D. Jang, J. Yoo, C. Y. Son, D. Kim, and H. J. Kim, "Multi-robot active sensing and environmental model learning with distributed gaussian process," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5905–5912, 2020.

[32] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, "Design and analysis of computer experiments," *Statistical science*, vol. 4, no. 4, pp. 409–423, 1989.

[33] C. Currin, T. Mitchell, M. Morris, and D. Ylvisaker, "Bayesian prediction of deterministic functions, with applications to the design and analysis of computer experiments," *Journal of the American Statistical Association*, vol. 86, no. 416, pp. 953–963, 1991.

[34] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *arXiv preprint arXiv:1812.06127*, 2018.

[35] X. Zhang, M. Hong, S. Dhople, W. Yin, and Y. Liu, "Fedpd: A federated learning framework with optimal rates and adaptivity to non-iid data," *arXiv preprint arXiv:2005.11418*, 2020.

[36] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[37] C. T. Dinh, N. H. Tran, and T. D. Nguyen, "Personalized federated learning with moreau envelopes," *arXiv preprint arXiv:2006.08848*, 2020.

[38] M. G. Arivazhagan, V. Aggarwal, A. K. Singh, and S. Choudhary, "Federated learning with personalization layers," *arXiv preprint arXiv:1912.00818*, 2019.

[39] P. P. Liang, T. Liu, L. Ziyin, N. B. Allen, R. P. Auerbach, D. Brent, R. Salakhutdinov, and L.-P. Morency, "Think locally, act globally: Federated learning with local and global representations," *arXiv preprint arXiv:2001.01523*, 2020.

[40] V. Smith, C.-K. Chiang, M. Sanjabi, and A. Talwalkar, "Federated multi-task learning," *arXiv preprint arXiv:1705.10467*, 2017.

[41] P. Moreno-Muñoz, A. Artés, and M. Álvarez, "Modular gaussian processes for transfer learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 24 730–24 740, 2021.

[42] Y. Gal, M. Van Der Wilk, and C. E. Rasmussen, "Distributed variational inference in sparse gaussian process regression and latent variable models," *Advances in neural information processing systems*, vol. 27, 2014.

[43] M. Deisenroth and J. W. Ng, "Distributed gaussian processes," in *International Conference on Machine Learning*. PMLR, 2015, pp. 1481–1490.

[44] Y. Zhang, J. C. Duchi, and M. J. Wainwright, "Communication-efficient algorithms for statistical optimization," *The Journal of Machine Learning Research*, vol. 14, no. 1, pp. 3321–3363, 2013.

[45] X. Wang and D. B. Dunson, "Parallelizing mcmc via weierstrass sampler," *arXiv preprint arXiv:1312.4605*, 2013.

[46] J. D. Lee, Q. Liu, Y. Sun, and J. E. Taylor, "Communication-efficient sparse regression," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 115–144, 2017.

[47] S.-B. Lin, X. Guo, and D.-X. Zhou, "Distributed learning with regularized least squares," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 3202–3232, 2017.

[48] X. Chen, W. Liu, and Y. Zhang, "Quantile regression under memory constraint," *The Annals of Statistics*, vol. 47, no. 6, pp. 3244–3273, 2019.

[49] X. Chen, J. D. Lee, H. Li, and Y. Yang, "Distributed estimation for principal component analysis: An enlarged eigenspace analysis," *Journal of the American Statistical Association*, pp. 1–12, 2021.

[50] X. Chen, W. Liu, and Y. Zhang, "First-order newton-type estimator for distributed estimation and inference,"

Journal of the American Statistical Association, pp. 1–17, 2021.

[51] J. Fan, Y. Guo, and K. Wang, "Communication-efficient accurate statistical estimation," *Journal of the American Statistical Association*, no. just-accepted, pp. 1–29, 2021.

[52] S. Zheng, C. Shen, and X. Chen, "Design and analysis of uplink and downlink communications for federated learning," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 7, pp. 2150–2167, 2020.

[53] Y. Cao and D. J. Fleet, "Generalized product of experts for automatic and principled fusion of gaussian process predictions," *arXiv preprint arXiv:1410.7827*, 2014.

[54] J. W. Ng and M. P. Deisenroth, "Hierarchical mixture-of-experts model for large-scale gaussian process regression," *arXiv preprint arXiv:1412.3078*, 2014.

[55] M. Tavassolipour, S. A. Motahari, and M. T. M. Shalmani, "Learning of gaussian processes in distributed and communication limited systems," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 8, pp. 1928–1941, 2019.

[56] K. Chen, Q. Kong, Y. Dai, Y. Xu, F. Yin, L. Xu, and S. Cui, "Recent advances in data-driven wireless communication using gaussian processes: A comprehensive survey," *China Communications*, vol. 19, no. 1, pp. 218–237, 2022.

[57] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[58] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, "On large-batch training for deep learning: Generalization gap and sharp minima," *arXiv preprint arXiv:1609.04836*, 2016.

[59] L. Wu, C. Ma *et al.*, "How sgd selects the global minima in over-parameterized learning: A dynamical stability perspective," *Advances in Neural Information Processing Systems*, vol. 31, pp. 8279–8288, 2018.

[60] H. Chen, L. Zheng, R. Al Kontar, and G. Raskutti, "Stochastic gradient descent in correlated settings: A study on gaussian processes," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[61] J. Hensman, N. Fusi, and N. D. Lawrence, "Gaussian processes for big data," *arXiv preprint arXiv:1309.6835*, 2013.

[62] M. L. Stein, "Limitations on low rank approximations for covariance matrices of spatial data," *Spatial Statistics*, vol. 8, pp. 1–19, 2014.

[63] D. Burt, C. E. Rasmussen, and M. Van Der Wilk, "Rates of convergence for sparse variational gaussian process regression," in *International Conference on Machine Learning*. PMLR, 2019, pp. 862–871.

[64] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006, vol. 2, no. 3.

[65] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," *arXiv preprint arXiv:1907.02189*, 2019.

[66] J. Bailly and D. Bailly, "Multifidelity aerodynamic optimization of a helicopter rotor blade," *AIAA Journal*, vol. 57, no. 8, pp. 3132–3144, 2019.

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

[68] L. Brevault, M. Balesdent, and A. Hebbal, "Overview of gaussian process based multi-fidelity techniques with variable relationship between fidelities, application to aerospace systems," *Aerospace Science and Technology*, vol. 107, p. 106339, 2020.

[69] M. G. Fernández-Godino, C. Park, N.-H. Kim, and R. T. Haftka, "Review of multi-fidelity models," *arXiv preprint arXiv:1609.07196*, 2016.

[70] B. Peherstorfer, K. Willcox, and M. Gunzburger, "Survey of multifidelity methods in uncertainty propagation, inference, and optimization," *Siam Review*, vol. 60, no. 3, pp. 550–591, 2018.

[71] M. C. Kennedy and A. O'Hagan, "Predicting the output from a complex computer code when fast approximations are available," *Biometrika*, vol. 87, no. 1, pp. 1–13, 2000.

[72] J. Laurenceau and P. Sagaut, "Building efficient response surfaces of aerodynamic functions with kriging and cokriging," *AIAA journal*, vol. 46, no. 2, pp. 498–507, 2008.

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

[74] S. Xiong, P. Z. Qian, and C. J. Wu, "Sequential design and analysis of high-accuracy and low-accuracy computer codes," *Technometrics*, vol. 55, no. 1, pp. 37–46, 2013.

[75] D. D. Cox, J.-S. Park, and C. E. Singer, "A statistical method for tuning a computer code to a data base," *Computational statistics & data analysis*, vol. 37, no. 1, pp. 77–92, 2001.

[76] P. Perdikaris, M. Raissi, A. Damianou, N. D. Lawrence, and G. E. Karniadakis, "Nonlinear information fusion algorithms for data-efficient multi-fidelity modelling," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 473, no. 2198, p. 20160751, 2017.

[77] H. Moon, A. M. Dean, and T. J. Santner, "Two-stage sensitivity-based group screening in computer experiments," *Technometrics*, vol. 54, no. 4, pp. 376–387, 2012.

[78] R. B. Gramacy and H. Lian, "Gaussian process single-index models as emulators for computer experiments," *Technometrics*, vol. 54, no. 1, pp. 30–41, 2012.

[79] D. Nguyen-Tuong, M. Seeger, and J. Peters, "Computed torque control with nonparametric regression models," in *2008 American Control Conference*. IEEE, 2008, pp. 212–217.

[80] D. Nguyen-Tuong and J. Peters, "Model learning for robot control: a survey," *Cognitive processing*, vol. 12, no. 4, pp. 319–340, 2011.

[81] T. D. Bui, C. V. Nguyen, S. Swaroop, and R. E. Turner, "Partitioned variational inference: A unified framework encompassing federated and continual learning," *arXiv preprint arXiv:1811.11206*, 2018.

[82] A. Saxena and K. Goebel, "Turbofan engine degradation simulation data set," *NASA Ames Prognostics Data Repository*, pp. 878–887, 2008.

[83] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients-how easy is it to break privacy in federated learning?" *Advances in Neural Information Processing Systems*, vol. 33, pp. 16 937–16 947, 2020.