Federated Gaussian Process Learning via Pseudo-Representations for Large-Scale Multi-Robot Systems

Sanket A. Salunkhe
Colorado School of Mines
Golden, CO, USA
sanket_salunkhe@mines.edu

George P. Kontoudis
Colorado School of Mines
Golden, CO, USA
george.kontoudis@mines.edu

ABSTRACT

Multi-robot systems require scalable and federated methods to model complex environments under computational and communication constraints. Gaussian Processes (GPs) offer robust probabilistic modeling, but suffer from cubic computational complexity, limiting their applicability in large-scale deployments. To address this challenge, we introduce the pxpGP, a novel distributed GP framework tailored for both centralized and decentralized large-scale multi-robot networks. Our approach leverages sparse variational inference to generate a local compact pseudo-representation. We introduce a sparse variational optimization scheme that bounds local pseudo-datasets and formulate a global scaled proximal-inexact consensus alternating direction method of multipliers (ADMM) with adaptive parameter updates and warm-start initialization. Experiments on synthetic and real-world datasets demonstrate that pxpGP and its decentralized variant, dec-pxpGP, outperform existing distributed GP methods in hyperparameter estimation and prediction accuracy, particularly in large-scale networks.

KEYWORDS

Gaussian Processes, Multi-Robot Systems, Distributed Optimization, Sparse Methods, Federated Learning, Large-Scale Networks

ACM Reference Format:
Sanket A. Salunkhe and George P. Kontoudis. 2026. Federated Gaussian Process Learning via Pseudo-Representations for Large-Scale Multi-Robot Systems . In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 9 pages. https://doi.org/10.65109/YQEA8075

Code: github.com/mpala-lab/distributed-gaussian-processes

1 INTRODUCTION

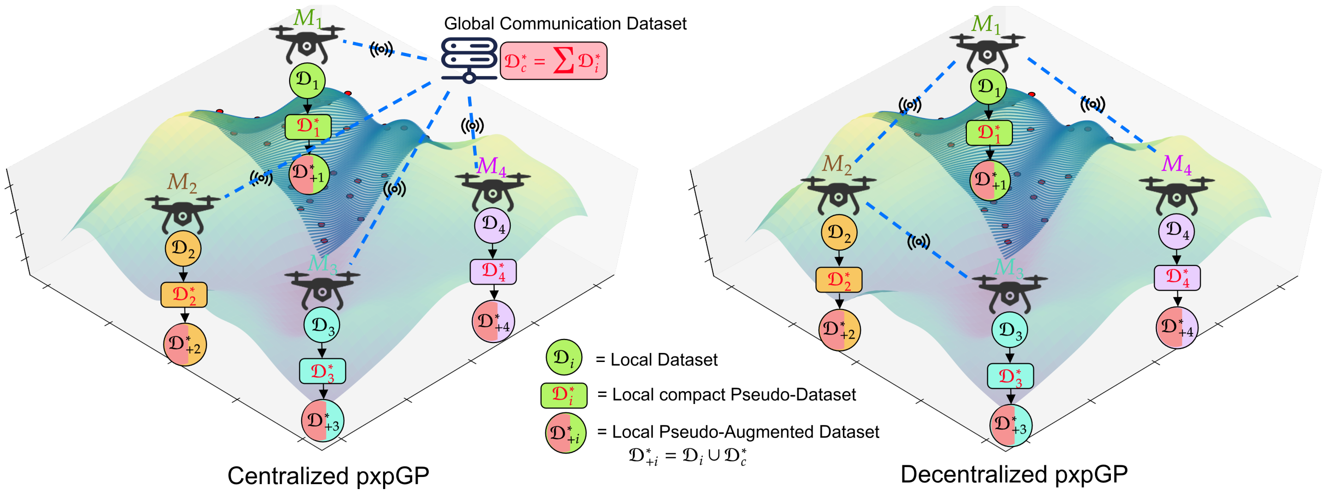
Multi-robot systems are increasingly used in executing complex, cooperative tasks such as environmental monitoring [5], search-and-rescue [27], autonomous exploration [4], and surveillance [33]. These applications require accurate modeling and prediction of environment or task-specific phenomena under uncertainty. Gaussian Processes (GPs) are well suited to these challenges, combining accurate function approximation with explicit uncertainty quantification [11, 28]. They have been successfully applied to distributed mapping [9, 29] and collaborative exploration tasks [16, 19, 34] due to their ability to provide uncertainty estimates that guide their decision-making process [15, 22].

However, using GPs in multi-robot teams is constrained by practical challenges such as limited onboard computation, privacy requirements, and communication bandwidth [10]. At the same time, GP training entails cubic complexity, which poses a major barrier with large datasets [17]. GP surrogate models are governed by a set of hyperparameters $\boldsymbol{\theta}$, learned using maximum likelihood estimation (MLE) methods over a given dataset $\mathcal{D}$. Accurate hyperparameter estimation is essential to ensure reliable predictions [28]. Our objective in this work is to develop distributed GP learning methods that can accurately estimate GP hyperparameters in large-scale multi-robot systems without sharing local raw datasets.

GP approximation techniques can be broadly classified into global aggregation methods and local inducing point-based methods [20]. Global approximation methods such as cGP [37], apxGP [36], and gapxGP [18] perform GP training across agents with Alternating Direction Method of Multipliers (ADMM) algorithms [2]. These approaches reduce computational and communication costs but require direct data sharing which can compromise data privacy and the quality of representation. Moreover, their performance degrades in networks larger than approximately 40 agents due to the independent assumption of distributed optimization [18, 19].

Local sparse variational methods reduce the cubic computational complexity of exact GPs by introducing a compact set of inducing variables that approximate the full covariance [35]. This method is infeasible in multi-robot systems, where data are inherently partitioned. To address this, [26] proposed a decentralized SGP framework, where each agent maintains a local variational posterior and fuses with neighboring models through maximum-consensus. While effective for small-scale deployments, this fusion mechanism is heuristic and lacks theoretical convergence guarantees, limiting scalability and robustness in large-scale networks.

Recent efforts have explored adaptive sampling in multi-agent systems. In [3], authors proposed a waypoint selection strategy where heterogeneous robots collaboratively estimate a stationary GP under dynamic constraints, and sensor noise. In [24], the authors presented a centralized GP method using variational inference. Similarly, [21, 38] develop decentralized GP approaches using random-feature GPs (RF-GPs), where agents share compact random-feature statistics with neighbors. However, random features can introduce systematic kernel approximation bias and yield poor covariance estimates, with accuracy strongly dependent on the number and quality of random features, in contrast to optimized pseudo-datasets. Other works, such as COOL-GP [13] and mixture-of-experts-based adaptive sampling [23], enable distributed GP learning and scalable



Global Communication Dataset
$\mathcal{D}_c^* = \sum \mathcal{D}_i^*$
M_1
M_2
M_3
M_4
$\mathcal{D}_i$ = Local Dataset
$\mathcal{D}_i^*$ = Local compact Pseudo-Dataset
$\mathcal{D}_{+i}^*$ = Local Pseudo-Augmented Dataset
$\mathcal{D}_{+i}^* = \mathcal{D}_i \cup \mathcal{D}_c^*$
Centralized pxpGP
Decentralized pxpGP

Figure 1: Overview of the proposed pxpGP framework in centralized and decentralized multi-robot networks. Each agent M_i generates a compact pseudo-dataset $\mathcal{D}_i^*$ and forms a pseudo-augmented dataset $\mathcal{D}_{+i}^*$. Centralized networks aggregate pseudo-datasets via a central node, while decentralized networks exchange data through neighbors via flooding.

modeling of non-stationary fields. While these methods advance distributed inference and adaptive data collection, they do not address the challenge of privacy-preserving hyperparameter optimization in GP training.

In this work, we propose the **Proximal Inexact Pseudo Gaussian Process (pxpGP)**, a distributed GP training framework designed for large-scale centralized and decentralized multi-robot networks. The method lies at the intersection of global aggregation and local sparse approaches to achieve scalability and data privacy by exchanging only optimized pseudo-datasets among agents, rather than raw or random observations. An overview of the proposed methods for both centralized and decentralized settings is illustrated in Fig. 1.

Contribution. The contribution of this work is twofold. First, we extend sparse variational inference techniques [12, 25, 32] to generate *compact pseudo-datasets* confined to each agent's region, improving informativeness and scalability to large-scale networks. Federated learning is promoted by sharing only compact pseudo-representations and optimization iterates instead of raw data. Second, we formulated pxpGP as a *scaled proximal-inexact consensus ADMM* algorithm initialized with warm-start hyperparameters and adaptive residual balancing that accelerates convergence and reduces communication rounds.

2 GAUSSIAN PROCESS TRAINING

Gaussian Processes (GPs) are non-parametric Bayesian models that define distributions over functions with a Gaussian prior. A GP over a latent function $f(\boldsymbol{x})$ is defined as,

$$f(\boldsymbol{x}) \sim GP(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')),$$

where $m(\boldsymbol{x})$ is the mean function and $k(\boldsymbol{x}, \boldsymbol{x}')$ is the covariance function (i.e., kernel), parameterized by a set of hyperparameters $\boldsymbol{\theta}$, that govern the smoothness, variability, and predictive accuracy of the GP model.

We model observations as $y(\boldsymbol{x}) = f(\boldsymbol{x}) + \epsilon$, where $\boldsymbol{x} \in \mathbb{R}^D$ is the input with dimension D, $y(\boldsymbol{x}) \in \mathbb{R}$ is the scalar output, and $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$ is zero-mean Gaussian measurement noise with variance σ_ϵ^2. We employ the Separable Squared Exponential (SSE) kernel,

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sigma_f^2 \exp\left\{-\frac{1}{2}\sum_{d=1}^{D}\frac{\left(x_d - x_d'\right)^2}{l_d^2}\right\},$$

with signal variance $\sigma_f > 0$ and length-scale $l_d > 0$. The GP hyperparameters $\boldsymbol{\theta} = \left[l_1, l_2, \cdots, l_d, \sigma_f, \sigma_\epsilon\right]^T \in \mathbb{R}^{D+2}$ are trained by maximizing the log-likelihood function,

$$\mathcal{L}(\boldsymbol{X}, \boldsymbol{y}; \boldsymbol{\theta}) = -\frac{1}{2}\left(\boldsymbol{y}^\top \boldsymbol{C}_\theta^{-1}\boldsymbol{y} + \log|\boldsymbol{C}_\theta| + N\log 2\pi\right),$$

where $\boldsymbol{C}_\theta = \boldsymbol{K} + \sigma_\epsilon^2 I_N$ is the positive definite covariance matrix and $\boldsymbol{K} = k(\boldsymbol{X}, \boldsymbol{X})$ is the kernel matrix, with $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n\}_{n=1}^N \subset \mathbb{R}^{N\times D}$ the input locations, $\boldsymbol{y} = \{y_1, y_2, \cdots, y_n\}_{n=1}^N \subset \mathbb{R}^N$ the corresponding scalar outputs, and N the dataset size. Thus, the negative log-likelihood (NLL) optimization problem yields,

$$\begin{aligned}\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \quad & \boldsymbol{y}^\top \boldsymbol{C}_\theta^{-1}\boldsymbol{y} + \log|\boldsymbol{C}_\theta| \\ \text{s.t.} \quad & \boldsymbol{\theta} > \boldsymbol{0}_{D+2}.\end{aligned} \tag{1}$$

The positivity constraint keeps $\boldsymbol{C}_\theta$ well-conditioned and positive definite. The optimization (1) requires computing $\boldsymbol{C}_\theta^{-1}$ at each iteration, with $O(N^3)$ computations and $O(N^2 + DN)$ storage.

2.1 Centralized Factorized GP Training (fact-GP)

To reduce the complexity of GP training, factorized GP (fact-GP) [6] partitions the global dataset $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{y}\}$ across multiple agents M disjoint subsets, $\mathcal{D} = \{\mathcal{D}_i\}_{i=1}^M$, where $i = \{1, 2, \cdots, M\}$ and

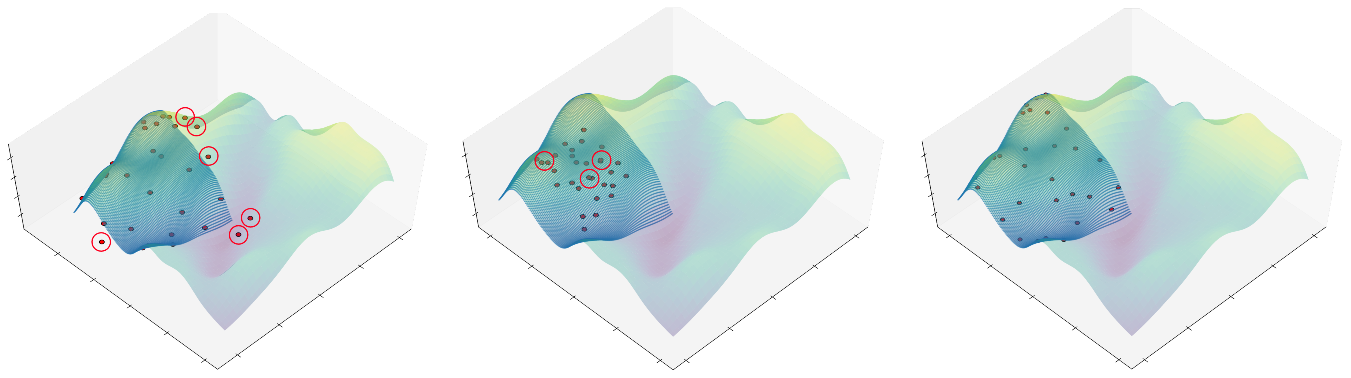


Figure 2: Effect of pxpGP regularization. (a) Pseudo-points drift beyond local bounds without boundary penalty (highlighted red circles). (b) Without the repulsive penalty, points cluster densely in a local region (highlighted red circles). (c) Combined boundary ($\mathfrak{L}_b$) and repulsive ($\mathfrak{L}_r$) penalties yield a well-distributed local pseudo-representations.

$\mathcal{D}_i = \{X_i, \boldsymbol{y}_i\}$. The global objective is approximated by the sum of local objectives with independent local datasets, $\mathcal{L} \approx \sum_{i=1}^{M} \mathcal{L}_i$. Each agent i trains local hyperparameters $\boldsymbol{\theta}_i$ and enforces global consensus through a shared parameter $\boldsymbol{z}$,

$$\begin{aligned} \hat{\boldsymbol{\theta}} &= \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{M} \mathcal{L}_i(\boldsymbol{\theta}_i) \\ \text{s.t.} \quad \boldsymbol{\theta}_i &= \boldsymbol{z}, \quad \forall i = 1, 2, \cdots, M, \end{aligned} \tag{2}$$

where $\mathcal{L}_i(\boldsymbol{\theta}_i) = \boldsymbol{y}_i^\top C_{\theta,i}^{-1} \boldsymbol{y}_i + \log |C_{\theta,i}|$ is the local NLL and $C_{\theta,i}$ is the local covariance matrix. To formulate the proposed distributed training algorithms, we introduce two specific assumptions about data distribution and communication structure among agents.

Assumption 1. *Each agent i trains a local sub-model on a statistically independent dataset that corresponds to a distinct region of the input space.*

Assumption 2. *Communication between agents is restricted to parameter or summary exchange and does not involve sharing raw datasets to preserve data privacy.*

The approximate proximal GP (apx-GP) [36] uses proximal inexact consensus ADMM to solve (2), reducing local complexity to $O((N/M)^3)$ with convergence guarantees for the non-convex optimization. It enables GP models to scale over large datasets, but as the number of agents M increases, Assumption 1 weakens, degrading hyperparameter estimates. On the other hand, gapx-GP [18] addresses this challenge by augmenting each local dataset $\mathcal{D}_{+i}$ with randomly sampled data from other agents. However, the latter violates Assumption 2 about privacy and does not scale beyond networks of approximately 40 agents.

2.2 Decentralized GP Training

In practical scenarios, a central coordinator is infeasible due to communication constraints, which motivates decentralized GP training where each agent collaborates only with its immediate neighbors $\mathcal{N}_i$. We model the decentralized network of M agents as a connected undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with $\mathcal{V} = \{v_1, v_2, \cdots, v_M\}$ as a set of agents i.e nodes in the network, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ the communication links i.e edges between them. For each agent i, the set of neighbors is defined as $\mathcal{N}_i = \{v_j \in \mathcal{V} | (v_i, v_j) \in \mathcal{E}\}$.

Prior work [18] introduced dec-cGP, dec-apxGP, and dec-gapxGP, to decentralized networks using edge-based ADMM [31] to solve,

$$\begin{aligned} \hat{\boldsymbol{\theta}} &= \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{M} \mathcal{L}_i(\boldsymbol{\theta}_i) \\ \text{s.t.} \quad \boldsymbol{\theta}_i &= \boldsymbol{z}_{ij}, \quad \forall i \in \mathcal{V}, j \in \mathcal{N}_i \\ \boldsymbol{\theta}_j &= \boldsymbol{z}_{ij}, \quad \forall i \in \mathcal{V}, j \in \mathcal{N}_i, \end{aligned} \tag{3}$$

where each agent optimizes its local hyperparameters $\boldsymbol{\theta}_i$ while maintaining consensus with neighbors via shared auxiliary variables $\boldsymbol{z}_{ij}$. The per-agent complexity remains $O\left((N/M)^3\right)$. Similar to the gapxGP, dec-gapxGP also shares raw data with neighboring agents, violating Assumption 2.

Problem 1. *Consider a large network of M robots that collaboratively model an unknown latent function using GPs. Each agent i holds a local dataset $\mathcal{D}_i$ and communicates with its one-hop neighbors. Under Assumption 1 (Independence) and 2 (Federated Constraints), the goal is to estimate the global GP hyperparameters $\hat{\boldsymbol{\theta}}$ by solving the centralized optimization problem* (2) *and its decentralized counterpart* (3)*, while minimizing communication rounds by ensuring fast convergence.*

3 PSEUDO INEXACT PROXIMAL GP (PXP-GP) TRAINING

In this section, we present the formulation of the **Proximal Inexact Pseudo GP (pxpGP)** training method for centralized and decentralized networks. Existing distributed GP training methods, such as gapxGP and dec-gapxGP [18], reduce approximation error of factorized GP methods by augmenting each agent's dataset $\mathcal{D}_{+i}$ with randomly sampled data from other agents. While these approaches are effective for small networks, they i) yield poorly representative augmented datasets in large-scale networks and ii) require raw-data exchange that violates the federated constraint (Assumption 2). pxpGP addresses these issues by letting each agent build a local pseudo-augmented dataset $\mathcal{D}_{+i}^*$ from a local sparse GP model trained over its local dataset $\mathcal{D}_i$, rather than from random samples.

Sparse GP approximations use a compact set of P inducing points $\boldsymbol{X}_p = \{x_{p_1}, x_{p_2}, \cdots, x_{p_i}\}_{i=1}^P \subset \mathbb{R}^{P\times D}$, where typically $P << (N/M)$. Thus, the training and prediction complexity reduces to $O(NP^2)$ and $O(P^2)$, respectively [32, 35]. The inducing points $\boldsymbol{X}_p$, variational parameters $\boldsymbol{\mu}_P$, and $\boldsymbol{A}_p$ are optimized by minimizing the negative Evidence Lower Bound (ELBO) $\mathfrak{L}_{\text{ELBO}}$ that yields,

$$\begin{aligned} q(f_P) &= \min_{q(f_P), X_P} -\mathfrak{L}_{\text{ELBO}} \\ &= \min_{q(f_P), X_P} -\left(\mathbb{E}_{q(f)}\left[\log p(\boldsymbol{y}|\boldsymbol{f})\right] - \text{KL}\left(q(\boldsymbol{f}_P) \mid\mid p(\boldsymbol{f}_P)\right)\right), \end{aligned} \quad (4)$$

where $q(f_P) = \mathcal{N}(\boldsymbol{\mu}_P, \boldsymbol{A}_P)$ is a Gaussian distribution with $\boldsymbol{\mu}_P$ variational mean and $\boldsymbol{A}_P$ variational covariance matrix, and KL $(q \mid\mid p)$ is the Kullback-Leibler (KL) divergence between the variational distribution q and GP prior p over the inducing variables $\boldsymbol{f}_P$. The first term of $\mathfrak{L}_{\text{ELBO}}$ encourages accurate data fitting, and the second term regularizes the variational approximation by penalizing divergence from the true posterior. The optimization starts with K-means initialization of the inducing points $\boldsymbol{X}_P$, followed by variational inference to minimize the negative ELBO (4).

Once each agent i generates a local pseudo-dataset $\mathcal{D}_i^*$ by solving (4), then all agents transmit $\mathcal{D}_i^*$ to create a shared communication dataset $\mathcal{D}_c^* = \cup_{i=1}^M \mathcal{D}_i^*$. Next, each agent constructs its local pseudo-augmented dataset $\mathcal{D}_{+i}^* = \mathcal{D}_i \cup \mathcal{D}_c^*$ by merging the original local dataset with the shared communication dataset, providing a richer global representation for GP training. The quality of the pseudo-dataset largely depends on the placement of inducing points. Without constraints, inducing points may drift beyond local data boundaries or cluster in dense regions as shown in Fig. 2a, 2b, respectively, leading to poor generalization, ill-conditioned covariance matrices, and occasionally Cholesky decomposition failures. To mitigate this, we introduce two regularization terms in the variational ELBO: 1) a boundary penalty ($\mathfrak{L}_b$) to confine points within data bounds; and 2) a repulsive penalty ($\mathfrak{L}r$) to ensure well-spread inducing points.

3.0.1 Boundary Penalty ($\mathfrak{L}_b$). This penalty constrains inducing points to remain within the bounds of the local dataset,

$$\mathfrak{L}_b = \sum_{i=1}^{P} \text{ReLU}\left(\boldsymbol{x}_{\min} - \boldsymbol{x}_i^*\right)^2 + \text{ReLU}\left(\boldsymbol{x}_i^* - \boldsymbol{x}_{\max}\right)^2, \quad (5)$$

where $\boldsymbol{x}_i^*$ represents the i-th inducing pseudo-input point among P points, $\boldsymbol{x}_{\min}$ and $\boldsymbol{x}_{\max}$ denote the minimum and maximum boundaries of the local dataset, respectively. The Rectified Linear Unit (ReLU) function ensures zero penalty inside the valid region, but applies a quadratic cost when points stray beyond the boundaries.

3.0.2 Repulsive Penalty ($\mathfrak{L}_r$). The variational sparse GP objective (4) [35] inherently discourages redundant overlapping inducing points through its complexity and trace terms. However, this implicit repulsive and non-overlapping effect is soft and data-dependent. Thus, the variational sparse GP objective (4) does not explicitly prevent local clustering, especially in distributed, data-partitioned, or non-stationary multi-agent setups where data exhibit spatial bias. As a result, the standard ELBO may yield clustered or poorly spaced inducing points, as seen in Fig. 2b. To address this limitation, the proposed repulsive penalty $\mathfrak{L}_r$ introduces an explicit geometric prior to enforce a minimum separation distance $d_{\min}$ between pseudo-inputs and improve spatial coverage,

$$\mathfrak{L}_r = \sum_{i=1}^{P}\sum_{j=1}^{P} \text{ReLU}\left(d_{\min} - \left\|\boldsymbol{x}_i^* - \boldsymbol{x}_j^*\right\|\right)^2, \quad (6)$$

where $\|\cdot\|$ denotes the Euclidean norm between two inducing points. The ReLU function ensures zero penalty when points are sufficiently separated, but applies a quadratic cost distance between points that fall below the threshold $d_{\min}$.

Together, these penalties produce a compact, well-distributed, and privacy-preserving local pseudo-augmented dataset $\mathcal{D}_{+i}^*$ that enhances global GP approximation and improves numerical conditioning of covariance matrices. The final objective function for Sparse GP combines ELBO (4) with the boundary (5) and repulsive penalties (6) as,

$$\boldsymbol{X}_P = \underset{q(f_P), X_P}{\arg\min} -\mathfrak{L}_{\text{ELBO}} + \mathfrak{L}_b + \mathfrak{L}_r. \quad (7)$$

3.1 Centralized pxpGP training

In the proposed centralized pxpGP framework, each agent i optimizes the hyperparameters $\boldsymbol{\theta}_i$ of its local GP model using the local pseudo-augmented dataset $\mathcal{D}_{+i}^*$. This is formulated as a scaled proximal-inexact consensus ADMM (pxADMM) problem, with analytical synchronous iterates [14] coordinated by a central node, with a fixed set of participating agents. By introducing a scaled dual variable $u_i^k = \frac{1}{\rho}\lambda_i^k$, we simplify the update rules, improve numerical stability, and enable adaptive penalty updates and warm-start initialization.

Consistent with existing distributed GP training methods such as cGP [37], apxGP [36], and gapxGP [18], pxpGP also enables each local agent to train independently while maintaining global consensus (2). The pxADMM linearizes the augmented Lagrangian around a stationary point $\boldsymbol{v}_i = \boldsymbol{z} + \boldsymbol{u}_i$ that yields,

$$\begin{aligned} \mathscr{L}\left(\boldsymbol{\theta}_i, z, \boldsymbol{v}_i\right) = \sum_{i=1}^{M} & \mathcal{L}_i(z) + \nabla_{\boldsymbol{\theta}}^{\intercal}\mathcal{L}_i(\boldsymbol{v}_i)\left(\boldsymbol{\theta}_i - \boldsymbol{v}_i\right) \\ &+ \frac{L_i + \rho_i}{2}\|\boldsymbol{\theta}_i - \boldsymbol{v}_i\|^2, \end{aligned} \quad (8)$$

where $L_i > 0$ is a positive Lipschitz parameter and ρ_i a regularization penalty parameter. The iterative updates for the pxpGP are provided by,

$$\boldsymbol{\theta}_i^{(s+1)} = \boldsymbol{v}_i^{(s)} - \frac{1}{L_i^{(s)} + \rho_i^{(s)}}\nabla_{\theta}\mathcal{L}_i\left(\boldsymbol{v}_i^{(s)}\right) \quad (9a)$$

$$\boldsymbol{z}^{(s+1)} = \frac{1}{M}\sum_{i=1}^{M}\left(\boldsymbol{\theta}_i^{(s+1)} + \boldsymbol{u}_i^{(s)}\right) \quad (9b)$$

$$\boldsymbol{u}_i^{(s+1)} = \boldsymbol{u}_i^{(s)} + \boldsymbol{\theta}_i^{(s+1)} - \boldsymbol{z}^{(s+1)}. \quad (9c)$$

While optimizing the global hyperparameters $\boldsymbol{\theta}$, the proposed pxpGP framework leverages the locally learned variational hyperparameters $\boldsymbol{\theta}_i^*$ from each sparse GP model to initialize subsequent

Algorithm 1 pxpGP

```
    Input: $\mathcal{D}_i = (X_i, \boldsymbol{y}_i)$, $k(\cdot,\cdot)$, $\rho_i$, $L_i$, $\epsilon_{\text{abs}}$, $\epsilon_{\text{rel}}$
    Output: $\hat{\boldsymbol{\theta}}$, $\mathcal{D}^*_{+i}$
 1: for $i = 1$ to $M$ do                                          ▷ Sparse Modeling
 2:     $\mathcal{D}^*_i, \boldsymbol{\theta}^*_i \leftarrow$ SparseModel($\mathcal{D}_i$) (7)
 3:     Communicate $\mathcal{D}^*_i$ to central node.
 4: end for
 5: Aggregate $\mathcal{D}^*_c = \cup_{i=1}^{M} \mathcal{D}^*_i$ at central node.
 6: Broadcast $\mathcal{D}^*_c$ to all agents $i \in M$ from central node.
 7: for $i = 1$ to $M$ do
 8:     $\mathcal{D}^*_{+i} = \mathcal{D}_i \cup \mathcal{D}^*_c$                     ▷ Local Augmented Dataset
 9:     Initialize $\boldsymbol{\theta}_i^{(1)} = \boldsymbol{\theta}^*_i$                          ▷ Warm Start
10: end for
11: repeat                                                  ▷ ADMM optimization
12:     Communicate $\boldsymbol{\theta}_i^{(s)}$ to central node.
13:     $\boldsymbol{z}^{(s+1)} \leftarrow$ primal-2($\boldsymbol{\theta}_i^{(s)}, \boldsymbol{u}_i^{(s)}$) (9b)
14:     Broadcast $\boldsymbol{z}^{(s+1)}$ to all agents from central node.
15:     for $i = 1$ to $M$ do
16:         $\boldsymbol{\theta}_i^{(s+1)} \leftarrow$ primal-1($\boldsymbol{z}^{(s+1)}, \boldsymbol{u}_i^{(s)}, \mathcal{D}^*_{+i}$) (9a)
17:         $\boldsymbol{u}_i^{(s+1)} \leftarrow$ dual($\boldsymbol{u}_i^{(s)}, \boldsymbol{\theta}_i^{(s+1)}, \boldsymbol{z}^{(s+1)}$) (9c)
18:         Update $\rho_i^{(s+1)}$ (11), $L_i^{(s+1)}$ (12)
19:     end for
20: until $\left\|\boldsymbol{r}_i^{(s+1)}\right\| \leq \epsilon_{\text{primal}}$ (10a), $\left\|\boldsymbol{s}_i^{(s+1)}\right\| \leq \epsilon_{\text{dual}}$ (10b)
21: return $\hat{\boldsymbol{\theta}}$
```

global training rounds. This warm-start mechanism preserves posterior information from local models and accelerates convergence by providing informed initial estimates for $\boldsymbol{\theta}_i^{(1)} = \boldsymbol{\theta}^*_i$.

We monitor the convergence of each agent using primal $\boldsymbol{r}_i^{(s+1)} = \boldsymbol{\theta}_i^{(s+1)} - \boldsymbol{z}^{(s+1)}$ and the dual residual $\boldsymbol{s}_i^{(s+1)} = \rho_i \left(\boldsymbol{z}^{(s+1)} - \boldsymbol{z}^{(s)}\right)$. These residuals must satisfy the conditions, $\left\|\boldsymbol{r}_i^{(s+1)}\right\| \leq \epsilon_{\text{primal}}$ and $\left\|\boldsymbol{s}_i^{(s+1)}\right\| \leq \epsilon_{\text{dual}}$ with tolerances,

$$\epsilon_{\text{primal}} = \sqrt{n_{\text{p}}}\epsilon_{\text{abs}} + \epsilon_{\text{rel}} \max\left\{\left\|\boldsymbol{\theta}_i^{(s+1)}\right\|, \left\|\boldsymbol{z}^{(s+1)}\right\|\right\} \tag{10a}$$

$$\epsilon_{\text{dual}} = \sqrt{n_{\text{d}}}\epsilon_{\text{abs}} + \epsilon_{\text{rel}} \left\|\rho_i \boldsymbol{u}_i^{(s+1)}\right\|, \tag{10b}$$

where n_{p} and n_{d} are the dimensions of the primal $\boldsymbol{\theta}$ and dual $\boldsymbol{u}$ variables, and ϵ_{abs}, ϵ_{rel} are absolute and relative tolerances.

The penalty parameter ρ_i balances the consensus constraint and the local objective in the augmented Lagrangian (8). Unlike prior works that fix ρ_i to a heuristic value, we adopt a residual-balancing strategy [2] that adjusts ρ_i based on primal and dual residuals,

$$\rho_i^{(s+1)} = \begin{cases} \tau_{\text{incr}}\rho_i^{(s)}, & \text{if } \left\|r^{(s)}\right\| > \beta \left\|s^{(s)}\right\| \\ \frac{\rho^{(s)}}{\tau_{\text{decr}}}, & \text{if } \left\|s^{(s)}\right\| > \beta \left\|r^{(s)}\right\| \\ \rho^{(s)}, & \text{otherwise,} \end{cases} \tag{11}$$

where $\beta, \tau_{\text{iccr}}, \tau_{\text{decr}} > 1$. In addition, we also adjust the local Lipschitz variable $L_i^{(s+1)}$ using a backtracking line search based on the Armijo condition [1]. In addition, at each iteration, we ensure that the local

Algorithm 2 dec-pxpGP

```
    Input: $\mathcal{D}_i = (X_i, \boldsymbol{y}_i)$, $k(\cdot,\cdot)$, $\rho_i$, $L_i$, $\mathcal{N}_i$ , $s^{\text{end}}_{\text{dec-pxpGP}}$
    Output: $\hat{\boldsymbol{\theta}}$, $\mathcal{D}^*_{+i}$
 1: for $i = 1$ to $M$ do                                          ▷ Sparse Modeling
 2:     $\mathcal{D}^*_i, \boldsymbol{\theta}^*_i \leftarrow$ SparseModel($\mathcal{D}_i$) (7)
 3:     $\mathcal{D}^*_c \leftarrow$ Flooding($\mathcal{D}^*_i, \mathcal{N}_i$)
 4:     $\mathcal{D}^*_{+i} = \mathcal{D}_i \cup \mathcal{D}^*_c$                     ▷ Local Augmented Dataset
 5:     Initialize $\boldsymbol{\theta}_i^{(1)} = \boldsymbol{\theta}^*_i$                          ▷ Warm Start
 6: end for
 7: for $s = 1$ to $s^{\text{end}}_{\text{dec-pxpGP}}$ do                           ▷ dec-ADMM optimization
 8:     for each $i \in \mathcal{V}$ do
 9:         Communicate $\boldsymbol{\theta}_i^{(s)}$ with $\mathcal{N}_i$
10:         $\boldsymbol{\alpha}_i^{(s+1)} \leftarrow$ dual($\boldsymbol{\alpha}_i^{(s)}, \boldsymbol{\theta}_i^{(s)}, \boldsymbol{\theta}_j^{(s)}$) (13b)
11:         $\boldsymbol{\theta}_i^{(s+1)} \leftarrow$ primal($\boldsymbol{\alpha}_i^{(s+1)}, \boldsymbol{\theta}_i^{(s)}, \boldsymbol{\theta}_j^{(s)} \mathcal{D}^*_{+i}$) (13a)
12:         Update $\rho_i^{(s+1)}$ (11), $L_i^{(s+1)}$ (12)
13:     end for
14: end for
15: return $\hat{\boldsymbol{\theta}}$
```

objective satisfies the condition,

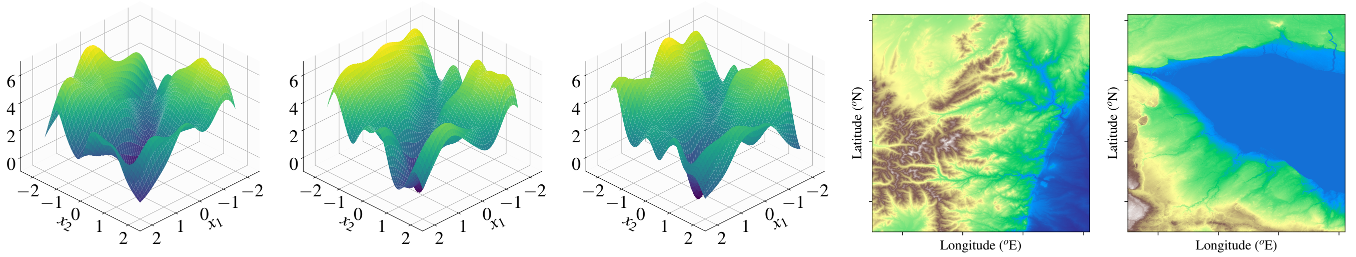
$$\mathcal{L}_i(\boldsymbol{\theta}_i^{(s+1)}) \leq \mathcal{L}_i(\boldsymbol{v}_i) - \frac{c \left\|\nabla_{\boldsymbol{\theta}} \mathcal{L}_i(\boldsymbol{v}_i)\right\|^2}{L_i^{(s+1)} + \rho_i^{(s+1)}}, \tag{12}$$

where $c \in (0, 1)$. If the condition fails, $L_i^{(s+1)}$ is reduced by a factor $\tau_{\text{lip}} \in (0, 1)$ until satisfied or a retry limit is reached.

The details of pxpGP implementation are presented in Algorithm 1. First, each agent computes locally a compact sparse GP model by solving (7) and communicates the local compact pseudo-dataset $\mathcal{D}^*_i$ to the central node. Next, the central node aggregates and broadcasts the communication dataset $\mathcal{D}^*_c$ to all agents. Then, each agent forms the local pseudo-augmented dataset $\mathcal{D}^*_{+i}$ and initializes the local hyperparameter vector based on the local compact sparse GP model $\boldsymbol{\theta}^*_i$ to warm-start the optimization. The ADMM optimization begins by communicating $\boldsymbol{\theta}_i^{(s)}$ to the central node to update the primal variable $\boldsymbol{z}^{(s+1)}$ (9b) and broadcast the computed value to all agents. Algorithm 1 requires coordination with a central node; asynchronous federated GP formulations that rely on different update rules and convergence assumptions are studied in [30]. Then, each agent computes the primal variable $\boldsymbol{\theta}_i^{(s+1)}$ (9a), the dual variable $\boldsymbol{u}_i^{(s+1)}$ (9c), while updating the penalty parameter $\rho_i^{(s+1)}$ (11) and Lipschitz parameter $L_i^{(s+1)}$ (12). The optimization iterates until the primal (10a) and dual residual (10b) converge.

3.2 Decentralized pxpGP training

We extend the pxpGP framework to decentralized network topologies by addressing the optimization problem (3). In decentralized pxpGP (dec-pxpGP), each agent i independently optimizes its local GP hyperparameters $\boldsymbol{\theta}_i$ using its local pseudo-augmented dataset $\mathcal{D}^*_{+i}$, while communicating only with its immediate neighbors $\mathcal{N}_i$ over a static, connected undirected graph $\mathcal{G}$. To distribute the pseudo-datasets across the network, we adopt a flooding mechanism that ensures all agents receive the shared communication



(a) GP Generative 1 (b) GP Generative 2 (c) GP Generative 3 (d) N39W106 (e) N43W080

Figure 3: Visualization of the datasets used for experimentation. Figures (3a), (3b), and (3c) depict synthetic generative GP datasets used for hyperparameter accuracy evaluation experiments, while Figure (3d) and (3e) show real-world NASA SRTM terrain datasets [7] used for assessing prediction performance.

dataset $\mathcal{D}_c^*$. Similarly to the centralized pxpGP, dec-pxpGP leverages locally learned variational hyperparameters $\boldsymbol{\theta}_i^*$ for warm-start initialization of the local hyperparameters $\boldsymbol{\theta}_i^{(1)} = \boldsymbol{\theta}_i^*$ in the global training stage. Each agent optimizes $\boldsymbol{\theta}_i$ while maintaining consensus on shared variables $\boldsymbol{z}_{ij}$ with its neighbors. Through synchronous iterative updates, all agents progressively converge to a consistent global hyperparameter estimate by following [18] that yields,

$$\boldsymbol{\theta}_i^{(s+1)} = \frac{1}{L_i^{(s)} + 2\rho_i\,|\mathcal{N}_i|}\left(\rho_i^{(s)} \sum_{j\in\mathcal{N}_i} \boldsymbol{\theta}_j^{(s)} - \nabla_{\boldsymbol{\theta}} \mathcal{L}_i\left(\boldsymbol{\theta}_i^{(s)}\right) - \boldsymbol{\alpha}_i^{(s)} + \left(\rho_i^{(s)}\,|\mathcal{N}_i| + L_i^{(s)}\right)\boldsymbol{\theta}_i^{(s)}\right) \tag{13a}$$

$$\boldsymbol{\alpha}_i^{(s+1)} = \boldsymbol{\alpha}_i^{(s)} + \rho_i^{(s)}\left(|\mathcal{N}_i|\,\boldsymbol{\theta}_i^{(s+1)} - \sum_{j\in\mathcal{N}_i} \boldsymbol{\theta}_j^{(s+1)}\right), \tag{13b}$$

where $|\mathcal{N}_i|$ denotes the number of neighbors of agent i (cardinality) and $\boldsymbol{\alpha}_i$ represents the dual variable. This formulation enables parallel, neighbor-only communication updates, resulting in a scalable and robust approach across varying network topologies.

The dec-pxpGP employs the adaptive residual-balancing strategy for ρ_i (11) and tunes the Lipschitz parameter L_i (12) using the Armijo condition (Algorithm 2).

4 NUMERICAL EXPERIMENTS AND RESULTS

To illustrate the efficacy of the proposed **pxpGP** and **dec-pxpGP** training method, we conduct numerical experiments with both synthetic and real-world datasets, and compare against existing distributed GP methods [36], [18]. For our experiments, we generate 2D synthetic datasets using generative GP functions with known hyperparameters $\boldsymbol{\theta} = (l_1, l_2, \sigma_f, \sigma_\epsilon)^\intercal = (0.7, 0.5, 1.8, 0.1)^\intercal$ for controlled benchmarking of GP hyperparameters, and used the NASA Shuttle Radar Topography Mission (SRTM) terrain elevation dataset [7] to evaluate the prediction performance.

We perform synthetic dataset experiments with two different dataset sizes, $N = 16,900$ and $N = 34,900$. For the real-world SRTM dataset, we use 3 tiles (N39W106, N37W120, N43W080), each with $N = 30,000$ training samples divided among agents and assign $N_{\text{test}} = 300$ test samples to each agent. Each training dataset is spatially and sequentially partitioned into equal-sized local datasets, satisfying Assumption 1, across varying fleet sizes $M \in \{16, 49, 64, 100\}$. For real-world dataset experiments, we use a single consistent global test dataset N_{test} among all agents to evaluate prediction performance. In all experiments, the number of inducing points per agent is chosen as $P = \max\{(N_i/M), 4\}$, where N_i is the dataset size of agent i. All experiments are implemented in Python using PyTorch and GPyTorch [8], running on a workstation with an Intel Core i7-14700 CPU, 62 GB RAM, and an NVIDIA GeForce RTX 4080 GPU with 16 GB VRAM.

The proposed pxpGP and dec-pxpGP frameworks are evaluated by benchmarking their hyperparameter estimation accuracy against several baseline methods, including the global full GP, centralized variants (apxGP [36] and gapxGP [18]), and their decentralized counterparts (dec-apxGP [18] and dec-gapxGP [18]). The predictive performance of proposed pxpGP and dec-pxpGP is compared against gapxGP and dec-gapxGP. The apxGP method is excluded from this comparison, since the gapxGP formulation provides a superior and more scalable alternative. For the centralized methods, the penalty and Lipschitz parameters are fixed at $\rho = 5$, $L_i = 10$ respectively, with convergence tolerances set to $\epsilon_{\text{abs}} = 10^{-5}$ and a maximum of 1,000 ADMM iterations. In contrast, pxpGP uses adaptive updates of ρ and L_i, initialized with $\rho^{(1)} = 1.0$ and $L_i^{(1)} = 5.0$, with the outer ADMM iterations are capped at $s^{\text{end}} = 500$. For the decentralized experiments, we adopt a minimal connected graph topology in which each agent has a maximum neighborhood degree of $|\mathcal{N}| = 2$, yielding a 1-connected network that satisfies the standard connectivity assumption for consensus while providing a lower-bound scenario on communication redundancy and mixing speed. This graph topology and data distribution represent the worst-case connectivity and worst-case data allocation conditions. As network connectivity increases or local regions begin to overlap, all methods demonstrate improved performance.

The evaluation focuses on three aspects: i) **hyperparameter estimation accuracy** relative to the ground-truth using the synthetic dataset; ii) **prediction performance** relative to the ground-truth on the real-world SRTM dataset; and iii) **computational and communication complexity** compared to baseline methods.

4.1 Hyperparameter Accuracy Estimate

The accuracy of the hyperparameter estimation is a key indicator of model consistency and scalability across agents. For the smaller

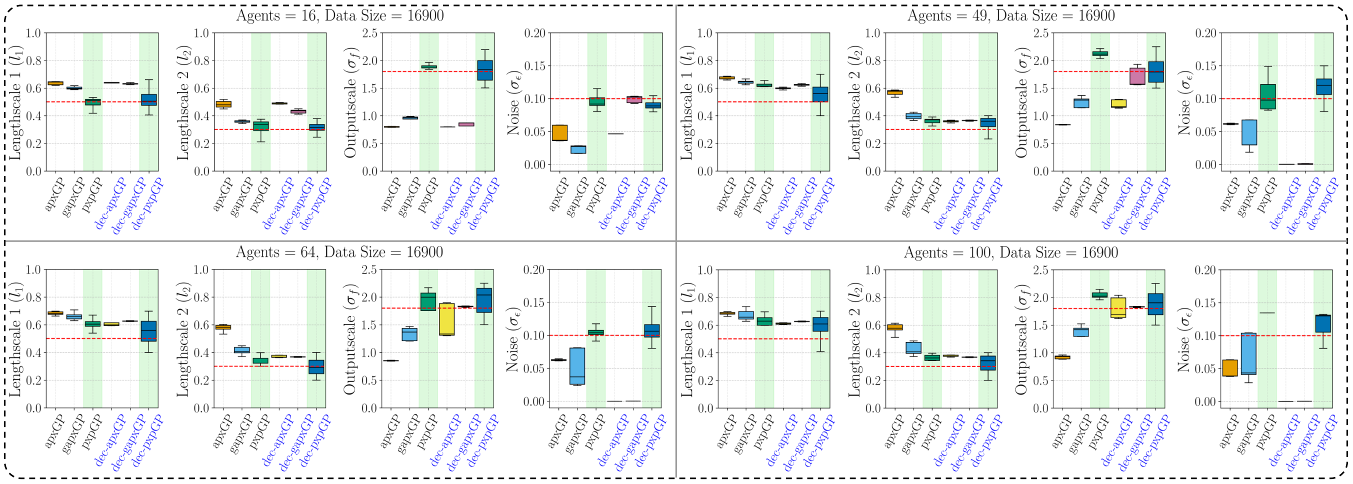


Figure 4: Hyperparameter estimation accuracy of baseline GP methods and proposed pxpGP (highlighted with green background) for centralized (black) and decentralized (blue) setups across fleet sizes $M = \{16, 49, 64, 100\}$ on a dataset with $N = 16{,}900$. Red dashed lines indicate ground-truth hyperparameters.

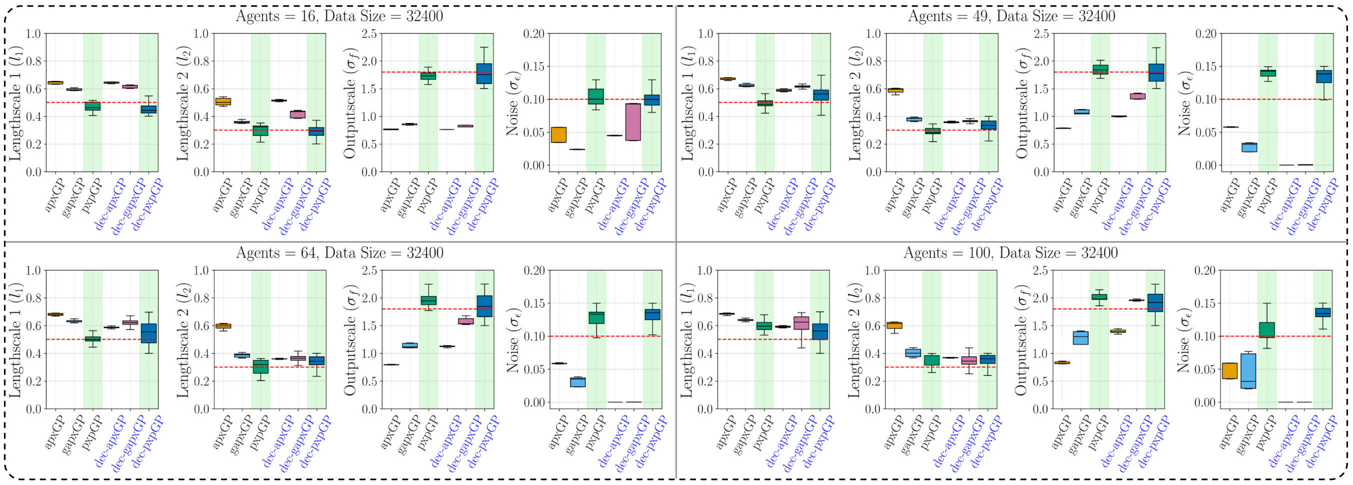


Figure 5: Hyperparameter estimation accuracy of baseline GP methods and proposed pxpGP (highlighted with green background) for centralized (black) and decentralized (blue) setups across fleet sizes $M = \{16, 49, 64, 100\}$ on a dataset with $N = 32{,}400$. Red dashed lines indicate ground-truth hyperparameters.

synthetic dataset ($N = 16,900$), Fig. 4, pxpGP and dec-pxpGP remain close to the ground-truth hyperparameters for all fleet sizes M, while the accuracy of baseline methods degrades noticeably as the number of agents M increases. For the larger synthetic dataset ($N = 34,900$), Fig. 5, all methods benefit from the increased data volume, but the pxpGP and dec-pxpGP still provide the most accurate and stable estimates across all fleet sizes, particularly in larger networks.

4.2 Prediction Performance

Reliable prediction and uncertainty estimation are key to evaluating model performance in distributed multi-robot learning. To evaluate these parameters, we assess the predictive performance of the proposed pxpGP and dec-pxpGP frameworks on three tiles of the real-world SRTM terrain dataset, comparing them with baseline gapxGP and dec-gapxGP methods. As summarized in Table 1, both pxpGP and dec-pxpGP achieve comparable or nearly identical Normalized Root Mean Square Error (NRMSE) values relative to their respective baselines.

More importantly, the proposed methods consistently yield substantially lower Negative Log Predictive Density (NLPD) values, which measure predictive uncertainty. Lower NLPD corresponds to more accurate and confident predictions, particularly for larger fleet sizes and non-stationary datasets such as tiles N37W120 and N43W080. Thus, the reduced NLPD values demonstrate that the proposed methods provide superior uncertainty quantification and higher model confidence compared to baseline approaches. Overall, both pxpGP and dec-pxpGP maintain stable prediction accuracy and well-calibrated uncertainty across diverse datasets and large-scale networks.

Table 1: Prediction accuracy of the proposed pxpGP and dec-pxpGP frameworks across fleet size $M = 16, 49, 64, 100$, using a training dataset of size $N = 30,000$ equally distributed among agents and a test dataset size $N_{test} = 300$ per agent, compared with the baseline models (gapxGP and dec-gapxGP [18]) on the SRTM dataset [7].

		Centralized GPs				Decentralized GPs			
		pxpGP		gapxGP [18]		**dec-pxpGP**		dec-gapxGP [18]	
Dataset	M	NRMSE ↓	NLPD ↓	NRMSE ↓	NLPD ↓	NRMSE ↓	NLPD ↓	NRMSE ↓	NLPD ↓
N39W106	16	**0.058 ± 0.012**	**0.265 ± 0.057**	0.071 ± 0.001	0.437 ± 0.009	**0.067 ± 0.001**	**0.305 ± 0.009**	0.072 ± 0.007	0.311 ± 0.093
	49	0.080 ± 0.002	**0.414 ± 0.033**	**0.076 ± 0.002**	0.487 ± 0.014	**0.061 ± 0.002**	**0.153 ± 0.038**	0.062 ± 0.003	0.213 ± 0.076
	64	0.081 ± 0.003	**0.422 ± 0.037**	**0.076 ± 0.002**	0.492 ± 0.018	**0.062 ± 0.002**	**0.170 ± 0.029**	0.062 ± 0.003	0.204 ± 0.056
	100	0.086 ± 0.003	0.562 ± 0.021	**0.081 ± 0.003**	**0.521 ± 0.019**	**0.067 ± 0.008**	**0.219 ± 0.084**	0.068 ± 0.006	0.272 ± 0.139
N37W120	16	**0.067 ± 0.012**	0.387 ± 0.130	0.069 ± 0.002	**0.381 ± 0.057**	**0.067 ± 0.005**	**0.202 ± 0.029**	0.071 ± 0.001	0.235 ± 0.070
	49	0.076 ± 0.004	**0.281 ± 0.135**	**0.074 ± 0.003**	0.421 ± 0.067	0.062 ± 0.002	**0.061 ± 0.101**	**0.061 ± 0.003**	0.151 ± 0.114
	64	0.077 ± 0.004	**0.292 ± 0.133**	**0.075 ± 0.003**	0.432 ± 0.062	0.064 ± 0.004	**0.084 ± 0.094**	**0.063 ± 0.003**	0.133 ± 0.093
	100	0.083 ± 0.004	**0.392 ± 0.040**	**0.078 ± 0.003**	0.457 ± 0.066	**0.066 ± 0.005**	**0.106 ± 0.129**	0.068 ± 0.006	0.204 ± 0.123
N43W080	16	**0.057 ± 0.016**	0.327 ± 0.139	0.065 ± 0.005	**0.314 ± 0.105**	0.065 ± 0.008	0.123 ± 0.055	**0.062 ± 0.005**	**0.161 ± 0.091**
	49	**0.071 ± 0.005**	**0.193 ± 0.169**	0.071 ± 0.008	0.371 ± 0.095	**0.054 ± 0.008**	**-0.086 ± 0.178**	0.061 ± 0.004	0.091 ± 0.131
	64	**0.072 ± 0.004**	**0.206 ± 0.165**	0.072 ± 0.009	0.397 ± 0.077	0.060 ± 0.006	**0.005 ± 0.130**	**0.057 ± 0.007**	0.067 ± 0.124
	100	0.082 ± 0.004	**0.343 ± 0.035**	**0.075 ± 0.005**	0.432 ± 0.065	**0.060 ± 0.008**	**0.005 ± 0.182**	0.067 ± 0.006	0.178 ± 0.128

Table 2: Computational and Communication Complexity Comparison of Distributed GP Methods

	apxGP [36]	gapxGP [18]	**pxpGP (Ours)**	dec-gapxGP [18]	**dec-pxpGP (Ours)**
Time	$O\left(\frac{N^3}{M^3}\right)$	$O\left(8\frac{N^3}{M^3}\right)$	$O\left(8\frac{N^3}{M^3}\right) + O\left(BP^2 + P^3\right)$	$O\left(8\frac{N^3}{M^3}\right)$	$O\left(8\frac{N^3}{M^3}\right) + O\left(BP^2 + P^3\right)$
Space	$O(\xi)$	$O\left(2\xi + \frac{2N^2}{M^2}\right)$	$O\left(2\xi + \frac{2N^2}{M^2} + P^2\right)$	$O\left(2\xi + \frac{2N^2}{M^2}\right)$	$O\left(2\xi + \frac{2N^2}{M^2} + P^2\right)$
Comms	$O\left(s^{\text{end}}M(D+2)\right)$	$O\left(s^{\text{end}}M(D+2)\right)$	$O\left(s^{\text{end}}M(D+2)\right)$	$O\left(s^{\text{end}}M(D+2)\right)$	$O\left(s^{\text{end}}M(D+2)\right)$

4.3 Computational Analysis

We compare the computational efficiency and scalability of all distributed GP training methods in Table 2, which summarizes their time, space, and communication complexity, where $\xi = N^2/M^2 + D(N/M)$. While pxpGP and dec-pxpGP add a one-time computational overhead from local sparse GP training ($O\left(BP^2 + P^3\right)$) to generate the local compact pseudo-dataset $\mathcal{D}_i^*$, where B is the batch size. This overhead is offset by: i) improved initialization, which requires fewer ADMM iterations to converge; and ii) adaptive tuning of ρ_i, and L_i, which minimizes manual adjustments and accelerates convergence.

Unlike gapxGP and dec-gapxGP [18], which rely on random sampling from local datasets, the proposed pxpGP and dec-pxpGP share compact pseudo-representations that enhance data privacy, and reduce communication overhead. Moreover, the dataset heterogeneity in gapxGP often leads to ill-conditioned covariance matrices, resulting in numerical instability during inversion, especially for large-scale networks. In contrast, pxpGP maintains numerical robustness through the use of optimized sparse datasets, warm-start initialization, and adaptive parameter selection, all of which contribute to faster convergence. Additionally, the proposed pxpGP employs warm-start initialization with near-optimal hyperparameters, reducing the need for strict convergence tolerances and requiring fewer ADMM iterations. The latter leads to improved hyperparameter estimation accuracy for large-scale networks, while lowering computational and communication costs.

5 CONCLUSION

In this work, we introduced **pxpGP** and **dec-pxpGP**, scalable and federated GP methods for large-scale multi-robot learning. By combining sparse variational inference with boundary and repulsive penalty terms, pxpGP constructs informative and well-distributed shared pseudo-datasets that enhance global data representation. The proposed centralized and decentralized variants demonstrate efficient and accurate hyperparameter estimation, and superior predictive performance in numerical experiments, spanning fleet sizes from 16 to 100 agents, while preserving data privacy.

These results underscore the scalability, efficiency, and robustness of the proposed method, positioning pxpGP as a strong candidate for real-world federated learning applications in large-scale multi-robot systems. Beyond scalable model learning, the proposed framework establishes a foundation for model-based control and decision-making, maintaining consistent uncertainty estimates under limited communication. This makes it particularly suited for cooperative exploration, where reliable modeling directly influences control and coordination strategies.

REFERENCES

[1] Dimitri P Bertsekas. 1999. *Nonlinear programming*. Athena Scientific.

[2] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. 2011. *Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers*. Vol. 3.

[3] Michael Brancato and Artur Wolek. 2024. Adaptive sampling of a stationary Gaussian spatial process by a team of robots with heterogeneous dynamics and measurement noise variance. *IEEE Access* 12 (2024), 94407–94423.

[4] Micah Corah, Cormac O'Meadhra, Kshitij Goel, and Nathan Michael. 2019. Communication-efficient planning and mapping for multi-robot exploration in large environments. *IEEE Robotics and Automation Letters* 4, 2 (2019), 1715–1721.

[5] Jnaneshwar Das, Frédéric Py, Julio BJ Harvey, John P Ryan, Alyssa Gellene, Rishi Graham, David A Caron, Kanna Rajan, and Gaurav S Sukhatme. 2015. Data-driven robotic sampling for marine ecosystem monitoring. *The International Journal of Robotics Research* 34, 12 (2015), 1435–1452.

[6] Marc Deisenroth and Jun Wei Ng. 2015. Distributed Gaussian processes. In *International Conference on Machine Learning*. 1481–1490.

[7] Tom G Farr, Paul A Rosen, Edward Caro, Robert Crippen, Riley Duren, Scott Hensley, Michael Kobrick, Mimi Paller, Ernesto Rodriguez, Ladislav Roth, et al. 2007. The shuttle radar topography mission. *Reviews of geophysics* 45, 2 (2007).

[8] Jacob Gardner, Geoff Pleiss, Kilian Q Weinberger, David Bindel, and Andrew G Wilson. 2018. GPyTorch: Blackbox matrix-matrix Gaussian process inference with GPU acceleration. *Advances in Neural Information Processing Systems* 31 (2018).

[9] Maani Ghaffari Jadidi, Jaime Valls Miro, and Gamini Dissanayake. 2018. Gaussian processes autonomous mapping and exploration for range-sensing mobile robots. *Autonomous Robots* 42 (2018), 273–290.

[10] Jennifer Gielis, Ajay Shankar, and Amanda Prorok. 2022. A critical review of communications in multi-robot systems. *Current Robotics Reports* 3, 4 (2022), 213–225.

[11] Robert B. Gramacy. 2020. *Surrogates: Gaussian Process Modeling, Design and Optimization for the Applied Sciences*. Chapman Hall/CRC, Boca Raton, Florida.

[12] Peter L Green. 2024. Distributed Gaussian Processes with Uncertain Inputs. *IEEE Access* 12 (2024), 176087–176093.

[13] Trong Nghia Hoang, Quang Minh Hoang, Kian Hsiang Low, and Jonathan How. 2019. Collective online learning of Gaussian processes in massive multi-agent systems. In *AAAI Conference on Artificial Intelligence*, Vol. 33. 7850–7857.

[14] Mingyi Hong, Zhi-Quan Luo, and Meisam Razaviyayn. 2016. Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems. *SIAM Journal on Optimization* 26, 1 (2016), 337–364.

[15] Dohyun Jang, Jaehyun Yoo, Clark Youngdong Son, Dabin Kim, and H Jin Kim. 2020. Multi-Robot Active Sensing and Environmental Model Learning With Distributed Gaussian Process. *IEEE Robotics and Automation Letters* 5, 4 (2020), 5905–5912.

[16] George P Kontoudis and Daniel J Stilwell. 2021. Decentralized nested Gaussian processes for multi-robot systems. In *IEEE International Conference on Robotics and Automation*. 8881–8887.

[17] George P Kontoudis and Daniel J Stilwell. 2023. Decentralized federated learning using Gaussian processes. In *IEEE International Symposium on Multi-Robot and Multi-Agent Systems*. 1–7.

[18] George P Kontoudis and Daniel J Stilwell. 2024. Scalable, Federated Gaussian Process Training for Decentralized Multi-Agent Systems. *IEEE Access* 12 (2024), 77800–77815.

[19] George P Kontoudis and Daniel J Stilwell. 2025. Multi-Agent Federated Learning Using Covariance-Based Nearest Neighbor Gaussian Processes. *IEEE Transactions on Machine Learning in Communications and Networking* 4 (2025), 115–138.

[20] Haitao Liu, Yew-Soon Ong, Xiaobo Shen, and Jianfei Cai. 2020. When Gaussian process meets big data: A review of scalable GPs. *IEEE Transactions on Neural Networks and Learning Systems* 31, 11 (2020), 4405–4423.

[21] Fernando Llorente, Daniel Waxman, and Petar M Djurić. 2025. Decentralized Online Ensembles of Gaussian Processes for Multi-Agent Systems. In *IEEE International Conference on Acoustics, Speech and Signal Processing*. 1–5.

[22] Wenhao Luo and Katia Sycara. 2018. Adaptive sampling and online learning in multi-robot sensor coverage with mixture of Gaussian processes. In *IEEE International Conference on Robotics and Automation*. 6359–6364.

[23] Kizito Masaba and Alberto Quattrini Li. 2023. Multi-robot adaptive sampling based on mixture of experts approach to modeling non-stationary spatial fields. In *IEEE International Symposium on Multi-Robot and Multi-Agent Systems*. 191–198.

[24] Pablo Moreno-Muñoz, Antonio Artés, and Mauricio Alvarez. 2021. Modular Gaussian processes for transfer learning. In *Advances in Neural Information Processing Systems*, Vol. 34. 24730–24740.

[25] Tanner Norton, Grant Stagg, Derek Ward, and Cameron K Peterson. 2023. Decentralized sparse Gaussian process regression with event-triggered adaptive inducing points. *Journal of Intelligent & Robotic Systems* 108, 4 (2023), 72.

[26] Tanner A Norton. 2022. *Efficient and Adaptive Decentralized Sparse Gaussian Process Regression for Environmental Sampling Using Autonomous Vehicles*. Master's thesis. Brigham Young University.

[27] Jorge Pena Queralta, Jussi Taipalmaa, Bilge Can Pullinen, Victor Kathan Sarker, Tuan Nguyen Gia, Hannu Tenhunen, Moncef Gabbouj, Jenni Raitoharju, and Tomi Westerlund. 2020. Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision. *IEEE Access* 8 (2020), 191617–191643.

[28] Carl Edward Rasmussen and Christopher KI Williams. 2006. *Gaussian Processes for Machine Learning* (2 ed.). Cambridge, MA, USA: MIT Press.

[29] Maria Santos, Udari Madhushani, Alessia Benevento, and Naomi Ehrich Leonard. 2021. Multi-robot learning and coverage of unknown spatial fields. In *IEEE International Symposium on Multi-Robot and Multi-Agent Systems*. 137–145.

[30] Jianwei Shi, Sameh Abdulah, Ying Sun, and Marc G Genton. 2025. Scalable Asynchronous Federated Modeling for Spatial Data. *arXiv preprint arXiv:2510.01771* (2025).

[31] Wei Shi, Qing Ling, Kun Yuan, Gang Wu, and Wotao Yin. 2014. On the linear convergence of the ADMM in decentralized consensus optimization. *IEEE Transactions on Signal Processing* 62, 7 (2014), 1750–1761.

[32] Edward Snelson and Zoubin Ghahramani. 2005. Sparse Gaussian processes using pseudo-inputs. *Advances in Neural Information Processing Systems* 18 (2005).

[33] Ethan Stump and Nathan Michael. 2011. Multi-robot persistent surveillance planning as a vehicle routing problem. In *IEEE International Conference on Automation Science and Engineering*. 569–575.

[34] Varun Suryan and Pratap Tokekar. 2020. Learning a Spatial Field in Minimum Time with a Team of Robots. *IEEE Transactions on Robotics* 36, 5 (2020), 1562–1576.

[35] Michalis Titsias. 2009. Variational learning of inducing variables in sparse Gaussian processes. In *Artificial intelligence and statistics*. PMLR, 567–574.

[36] Ang Xie, Feng Yin, Yue Xu, Bo Ai, Tianshi Chen, and Shuguang Cui. 2019. Distributed Gaussian processes hyperparameter optimization for big data using proximal ADMM. *IEEE Signal Processing Letters* 26, 8 (2019), 1197–1201.

[37] Yue Xu, Feng Yin, Wenjun Xu, Jiaru Lin, and Shuguang Cui. 2019. Wireless traffic prediction with scalable Gaussian process: Framework, algorithms, and verification. *IEEE Journal on Selected Areas in Communication* 37, 6 (2019), 1291–1306.

[38] Xubo Yue and Raed Kontar. 2024. Federated Gaussian Process: Convergence, Automatic Personalization and Multi-fidelity Modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 46, 6 (2024), 4246–4261.