Distributed Gaussian Process Hyperparameter Optimization for Multi-Agent Systems

Zhai, Peiyuan; Rajan, Raj Thilak

DOI
10.1109/ICASSP49357.2023.10096267

Publication date
2023

Document Version
Final published version

Published in
Proceedings of the ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)

ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-7281-6327-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICASSP49357.2023.10096267

DISTRIBUTED GAUSSIAN PROCESS HYPERPARAMETER OPTIMIZATION FOR MULTI-AGENT SYSTEMS

Peiyuan Zhai, Raj Thilak Rajan

Signal Processing and Systems, Delft University of Technology, The Netherlands

ABSTRACT

Gaussian Process (GP) is a flexible non-parametric method which has a wide variety of applications e.g., field estimation using multi-agent systems. However, the training of the hyperparameters suffers from high computational complexity. Recently, distributed hyperparameter optimization with proximal gradients has been proposed to reduce complexity, however only for a network with a central station. In this work, exploiting edge-based constraints, we propose two fully-distributed algorithms pxADMM$_{\text{fd}}$ and pxADMM$_{\text{fd},fast}$ for a network of multi-agent systems, which do not rely on a central station. In addition, asynchronous versions of the algorithms are also proposed to reduce the synchronization overhead in heterogeneous networks. Simulations are conducted for a field estimation problem, using both artificial, and real-world datasets, which show that the proposed fully-distributed algorithms successfully converge, at the cost of an increased number of iterations.

***Index Terms*—** Gaussian Process, Multi-agent Systems, ADMM, Field estimation,

1. INTRODUCTION

A multi-agent system (MAS) consists of identical agents that are able to measure, compute and communicate, and can be deployed for various applications, e.g., source seeking [1, 2], environmental monitoring [3, 4], geographical information systems modeling [5, 6], signal strength mapping [7], etc. In general, the problem can be modeled as learning an unknown function based on noisy measurements of underlying signal, for which a parametric model is hard to be established. As a non-parametric Bayesian model, Gaussian Process (GP) is capable of modeling unknown functions relying on a large number of measurements. However, the application of a large-scale GP regression (GPR) is prohibitive because of its high computational complexity of $\mathcal{O}(N^3)$, where N is the number of measurements. The problem can be alleviated through either low-rank approximation [8] or distributed computing [9]. In the case of distributed computing using a MAS of M agents, the computational complexity is reduced to $\mathcal{O}(\frac{N^3}{M^2})$, by evenly dividing the dataset among the agents.

The performance of GPR is optimized by choosing the appropriate kernel function and hyperparameters that reflect the characteristic of the underlying function, where Bayesian model selection (BMS) [10] can be applied. Early distributed GP models were either trained with full dataset [11] or entirely in computing center [12]. In [13], the computational load is distributed to multiple agents by formulating the problem as consensus optimization problem, which is solved by alternated direction methods of multipliers (ADMM). The state of the art algorithm proposed by Xie et al. [14] adopts a proximal ADMM (pxADMM) approach [15], which minimizes the computational complexity by replacing exact gradient with first-order approximation. However, pxADMM still relies on networks with central stations, which makes it not fully-distributed, and limits the overall network-wide computation speed to the speed of the slowest agent.

In this work, we propose a fully-distributed algorithm pxADMM$_{\text{fd}}$ based on the pxADMM-based algorithm, that does not rely on the assistance of a central station. The proposed algorithm adopts edge-based constraints that can be computed in local agent pairs, thus eliminating the need for a central station and increasing the scalability. By introducing extra auxiliary variables, we also propose pxADMM$_{\text{fd,fast}}$ that shows faster experimental convergence than pxADMM$_{\text{fd}}$. In addition, the asynchronous versions of the aforementioned algorithms are also proposed to ensure a fully-distributed solution, for an unsynchronized MAS. The structure of this paper is as follows. In Section 2, the basics of GPR and state of the art hyperparameter optimization algorithms are introduced. In Section 3, the fully-distributed and asynchronous algorithms are proposed. Numerical simulations are shown in Section 4, followed by conclusions in Section 5.

2. HYPERPARAMETER OPTIMIZATION

2.1. Gaussian Process Regression

Gasussian Process (GP) models the underlying function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ by a prior mean function and kernel function $k(\mathbf{x}, \mathbf{x}')$, where $\mathbf{x}$ and $\mathbf{x}'$ are two input vectors. Without loss of generality, we assume a zero mean function,

This work is partially funded by the European Leadership Joint Undertaking (ECSEL JU), under grant agreement No 876019, the ADACORSA project - "Airborne Data Collection on Resilient System Architectures."

and subsequently GP is typically modelled as $f(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, k(\mathbf{x}, \mathbf{x}'))$. Gaussian Process Regression (GPR) learns the underlying function based on a noisy training dataset $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^N$ with N input and output pairs $(\mathbf{x}_n, y_n)$. The noisy measurement y_n is obtained under regression model $y = f(\mathbf{x}) + w$, where $f(\mathbf{x})$ is assumed to be a GP and $w \sim \mathcal{N}(0, \sigma_n^2)$. To predict the function value y_* at point $\mathbf{x}_*$, a predictive distribution $p(y_*|\mathbf{x}_*, \mathbf{X}, \mathbf{y})$ should be found, where $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N]^T$ and $\mathbf{y} = [y_1, \cdots, y_N]^T$ are the input and output respectively. The posterior distribution is a Gaussian distribution $\mathcal{N}(\mathbf{K}_*\mathbf{K}^{-1}\mathbf{y}, \mathbf{K}_{**} - \mathbf{K}_*\mathbf{K}^{-1}\mathbf{K}_*^T)$. The (m, n)th element of $N \times N$ covariance matrix $\mathbf{K}$ is $[\mathbf{K}]_{m,n} = k(\mathbf{x}_m, \mathbf{x}_n)$, where $\mathbf{x}_m$ and $\mathbf{x}_n$ are the input vectors in the training dataset. The cross-covariance and auto-covariance matrices are given by $\mathbf{K}_* = [k(\mathbf{x}_*, \mathbf{x}_1), \cdots, k(\mathbf{x}_*, \mathbf{x}_N)]$ and $\mathbf{K}_{**} = k(\mathbf{x}_*, \mathbf{x}_*)$, respectively. A commonly used kernel function is the radial basis function (RBF), which is given by

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left(-\boldsymbol{\alpha}^T \boldsymbol{\alpha}/2\right) + \sigma_n^2 \delta(\mathbf{x} - \mathbf{x}'), \quad (1)$$

where $\boldsymbol{\alpha} = \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \mathbf{x}')$, signal variance σ_f^2 indicates the output range of signals, $\boldsymbol{\Sigma} = diag(l_1, l_2, \cdots, l_d)$ is a $d \times d$ diagonal matrix which contains the characteristic lengths of the d input dimensions [10] . These hyperparameters can be collectively denoted by hyperparameter set $\boldsymbol{\theta} = \{\sigma_f, \boldsymbol{\Sigma}, \sigma_n\}$. To estimate the hyperparameters, an ML estimator can be applied to maximize the likelihood $p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})$, which is equivalent to minimizing the negative log-likelihood (NLL) given by $l(\boldsymbol{\theta}) = \mathbf{y}^T\mathbf{K}^{-1}(\boldsymbol{\theta})\mathbf{y} + \log|\mathbf{K}(\boldsymbol{\theta})|$ [10]. The NLL can be directly minimized by gradient-based methods, e.g. gradient descent [14]. We now briefly summarize the state of the art solution to estimate the hyperparameters using a distributed approach.

2.2. pxADMM-based hyperparameter optimization

In a MAS of M agents, the training dataset is divided into M non-overlapping local datasets $\{\mathcal{D}_m\}_{m=1}^M$, where $\mathcal{D}_m = \{\mathbf{X}_m, \mathbf{y}_m\}$ contains the input and output vectors at agent m. The centralized NLL is approximated by the summation of M local NLLs $\{l_m(\boldsymbol{\theta}_m)\}_{m=1}^M$ based on local datasets and hyperparameters. For simplicity, NLL is also used to denote the summation of local NLLs. The distributed optimization problem is

$$\begin{aligned} \mathcal{P}_1 : \min_{\{\boldsymbol{\theta}_m\}, \mathbf{z}} \quad & \sum_{m=1}^M l_m(\boldsymbol{\theta}_m) \\ s.t. \quad & \forall\, \boldsymbol{\theta}_m = \mathbf{z},\ m = 1, 2, \ldots, M, \end{aligned} \quad (2)$$

where $l_m(\boldsymbol{\theta}_m) = \mathbf{y}_m^T\mathbf{K}_m^{-1}(\boldsymbol{\theta}_m)\mathbf{y}_m + \log|\mathbf{K}_m(\boldsymbol{\theta}_m)|$, $\boldsymbol{\theta}_m$ is the local set of hyperparameters which is maintained at agent m, and $\mathbf{z}$ is an auxiliary variable maintained at the central station. The constraints indicate that a unique GP is enough to model the underlying function. The augmented Lagrangian of (2) is given by

$$\begin{aligned} &\mathcal{L}\left(\{\boldsymbol{\theta}_m\}, \mathbf{z}, \{\boldsymbol{\lambda}_m\}\right) = \\ &\sum_{m=1}^M \left(l_m(\boldsymbol{\theta}_m) + \langle \boldsymbol{\lambda}_m, \boldsymbol{\theta}_m - \mathbf{z}\rangle + \frac{\rho}{2}\|\boldsymbol{\theta}_m - \mathbf{z}\|_2^2\right), \end{aligned} \quad (3)$$

where $\boldsymbol{\lambda}_m$ contains the dual variables at agent m, and $\rho > 0$ is a predetermined penalty term [13]. The update of $\boldsymbol{\theta}_m$ at the tth iteration is

$$\boldsymbol{\theta}_m^{t+1} = \arg\min_{\boldsymbol{\theta}_m} \left(l_m(\boldsymbol{\theta}_m) + \boldsymbol{\theta}_{\boldsymbol{m}}^{\boldsymbol{T}}\boldsymbol{\lambda}_m^t + \frac{\rho}{2}\left\|\boldsymbol{\theta}_m - \mathbf{z}^{t+1}\right\|_2^2\right). \quad (4)$$

Note that the exact update of $\boldsymbol{\theta}_m$ in (4) is not in closed-form and requires multiple examinations of gradient $\nabla l_m(\boldsymbol{\theta}_m)$, each of which has a computational complexity of $\mathcal{O}(N^3)$. The pxADMM-based algorithm proposed in [14], further replaces $l_m(\boldsymbol{\theta}_m)$ with first-order approximation $l_m(\boldsymbol{\theta}_m) \approx l_m(\mathbf{z}) + \nabla l_m(\mathbf{z})(\boldsymbol{\theta}_m - \mathbf{z})$, and thus the closed-form $\boldsymbol{\theta}_m$ update is

$$\boldsymbol{\theta}_m^{t+1} = \mathbf{z}^{t+1} - \left(\nabla l_m(\mathbf{z}^{t+1}) + \boldsymbol{\lambda}_m^t\right)/(\rho + L), \quad (5)$$

where L satisfies $\|\nabla l_m(\boldsymbol{\theta}_m) - \nabla l_m(\boldsymbol{\theta}_m')\| \leq L\|\boldsymbol{\theta}_m - \boldsymbol{\theta}_m'\|,\ \forall\ \boldsymbol{\theta}_m, \boldsymbol{\theta}_m'$. Note that a central station is needed to maintain the global variable $\mathbf{z}$, a limitation we overcome with our proposed algorithms in the next section.

3. FULLY DISTRIBUTED OPTIMIZATION

In this section, by reformulating problem $\mathcal{P}_1$, we propose two versions of fully-distributed pxADMM.

3.1. Fully-distributed pxADMM (pxADMM$_{\mathbf{fd}}$)

Inspired by [16], we introduce edge-based auxiliary variables $\mathbf{z}_{mn}$ and replace the constraints in problem $\mathcal{P}_1$ by edge-based constraints, which leads to the following problem $\mathcal{P}_2$

$$\begin{aligned} \mathcal{P}_2 : \min_{\{\boldsymbol{\theta}_m\}, \{\mathbf{z}_{mn}\}} \quad & \sum_{m=1}^M l_m(\boldsymbol{\theta}_m) \\ s.t. \quad & \boldsymbol{\theta}_m = \mathbf{z}_{mn},\ \forall (m, n) \in \mathcal{E}, \end{aligned} \quad (6)$$

where the edge set $\mathcal{E}$ contains all the edges in the MAS. The modified edge-based constraints require that each pair of agents shares the same hyperparameter value. The augmented Lagrangian for the mth agent is then given by

$$\begin{aligned} &\mathcal{L}\left(\{\boldsymbol{\theta}_m\}, \{\mathbf{z}_{mn}, \boldsymbol{\lambda}_{mn}\}\right) = \sum_{m=1}^M \Bigg(l_m(\boldsymbol{\theta}_m) \\ &+ \sum_{n \in \mathcal{N}(m)} \left(\langle \boldsymbol{\lambda}_{mn}, \boldsymbol{\theta}_m - \mathbf{z}_{mn}\rangle + \frac{\rho}{2}\|\boldsymbol{\theta}_m - \mathbf{z}_{mn}\|_2^2\right)\Bigg), \end{aligned} \quad (7)$$

where $\boldsymbol{\lambda}_{mn}$ contains the dual variables associated with edge (m, n) and is stored at agent m, and neighborhood $\mathcal{N}(m)$ is

the set of agents which are directly connected to agent m. Subsequently, the $\boldsymbol{\theta}_m$ update is given by

$$\boldsymbol{\theta}_m^{t+1} = \arg\min_{\boldsymbol{\theta}_m} \Big(l_m(\boldsymbol{\theta}_m) + \sum_{n\in\mathcal{N}(m)} \Big(\boldsymbol{\theta}_m^T \boldsymbol{\lambda}_{mn}^t + \frac{\rho}{2} \left\| \boldsymbol{\theta}_m - \mathbf{z}_{mn}^{t+1} \right\|_2^2 \Big) \Big). \tag{8}$$

To reduce the computational complexity, we apply a first-order approximation of $l_m(\boldsymbol{\theta}_m)$, which gives

$$l_m(\boldsymbol{\theta}_m) \approx l_m\left(\boldsymbol{\theta}_m^t\right) + \nabla l_m\left(\boldsymbol{\theta}_m^t\right)\left(\boldsymbol{\theta}_m - \boldsymbol{\theta}_m^t\right), \tag{9}$$

where $l_m(\boldsymbol{\theta}_m)$ is approximated around the local hyperparameters $\boldsymbol{\theta}_m$ from latest iteration. The update equations at iteration t are

$$\mathbf{z}_{mn}^{t+1} = \frac{1}{2}\left(\rho^{-1}\left(\boldsymbol{\lambda}_{mn}^t + \boldsymbol{\lambda}_{nm}^t\right) + \boldsymbol{\theta}_m^t + \boldsymbol{\theta}_n^t\right), \tag{10a}$$

$$\boldsymbol{\theta}_m^{t+1} = \frac{\sum_{n\in\mathcal{N}(m)} (\rho \mathbf{z}_{mn} - \boldsymbol{\lambda}_{mn}) + L\boldsymbol{\theta}_m^t - \nabla l_m(\boldsymbol{\theta}_m^t)}{L + \rho\left|\mathcal{N}(m)\right|}, \tag{10b}$$

$$\boldsymbol{\lambda}_{mn}^{t+1} = \boldsymbol{\lambda}_{mn}^t + \rho\left(\boldsymbol{\theta}_m^{t+1} - \mathbf{z}_{mn}^{t+1}\right), \tag{10c}$$

where ρ and L satisfy the same conditions as those from (5).

3.2. Faster fully-distributed pxADMM (pxADMM$_{\text{fd,fast}}$)

In pxADMM$_{\text{fd}}$, $l_m(\boldsymbol{\theta}_m)$ is approximated around the most recent $\boldsymbol{\theta}_m$ because the global auxiliary variable $\mathbf{z}$ used in pxADMM-based algorithm is not accessible in fully-distributed networks. To overcome this impediment, we propose another fully-distributed pxADMM-based hyperparameter optimization algorithm by approximating the likelihood around the local auxiliary variables $\{\boldsymbol{\zeta}_m\}_{m=1}^M$, which is inspired by the approximation around global auxiliary variable in [15]. This algorithm shows faster convergence in simulations, and is denoted as pxADMM$_{\text{fd,fast}}$. $\boldsymbol{\zeta}_m$ at agent m is constructed, and is updated as

$$\boldsymbol{\zeta}_m^{t+1} = \frac{1}{1+|\mathcal{N}(m)|}\left(\boldsymbol{\zeta}_m^t + \sum_{n\in\mathcal{N}(m)} \mathbf{z}_{mn}^{t+1}\right), \tag{11}$$

which is a temporally smoothed moving average of $\mathbf{z}_{mn}, n \in \mathcal{N}(m)$. Simulation results that auxiliary $\{\boldsymbol{\Lambda}_m\}_{m=1}^M$ also need to be introduced to replace $\boldsymbol{\lambda}_{mn}$ in (5), otherwise the algorithm can not converge. $\boldsymbol{\Lambda}_m$ is constructed and maintained as

$$\boldsymbol{\Lambda}_m^{t+1} = \frac{\boldsymbol{\Lambda}_m^t + \rho\left(\boldsymbol{\theta}_m^{t+1} - \boldsymbol{\zeta}_m^{t+1}\right) + \sum_{n\in\mathcal{N}(m)} \boldsymbol{\lambda}_{mn}^{t+1}}{1+|\mathcal{N}(m)|}. \tag{12}$$

With the auxiliary variables, the update equation of $\boldsymbol{\theta}_m$ is derived as

$$\boldsymbol{\theta}_m^{t+1} = \boldsymbol{\zeta}^{t+1} - \frac{1}{\rho+L}\left(\nabla l_m(\boldsymbol{\zeta}^{t+1}) + \boldsymbol{\Lambda}_m^t\right). \tag{13}$$

The update iterations of $\mathbf{z}_{mn}$ and $\boldsymbol{\lambda}_{mn}$ follows (10a) and (10c). In the tth iteration, the update of $\boldsymbol{\zeta}_m^{t+1}$ performs an extra step towards the consensus values of the auxiliary variables $\mathbf{z}_{mn}^{t+1}$. Since the most recent information from neighbor agents is stored in $\boldsymbol{\zeta}_m^{t+1}$, this leads to an improved approximation as compared to $\boldsymbol{\theta}_m^t$ from the previous iteration, and thus leads to a faster convergence. In the following simulation section, pxADMM$_{\text{fd,fast}}$ shows faster convergence than pxADMM$_{\text{fd}}$, under the same stop criteria.

To ensure these algorithms operate for an unsynchronized network of MAS, we apply two strategies, partial barrier and bounded delay [17], to introduce the fully-distributed asynchronous algorithms. The partial barrier introduces asynchronous behavior by allowing incomplete variables exchange. An agent m with faster computation capacity can start the next local iteration immediately after receiving updates from at least $1 \le S \le |\mathcal{N}(m)|$ agents. The bounded delay is applied to allow each agent to wait for updates from neighbors who have not communicated with the agent for $\tau \ge 1$ local iterations, which introduce a certain level of alignment that ensures a pattern of convergence across the network. Both pxADMM$_{\text{fd}}$ and pxADMM$_{\text{fd,fast}}$ can adopt the asynchronous strategies, and are respectively named as pxADMM$_{\text{afd}}$ and pxADMM$_{\text{afd,fast}}$.

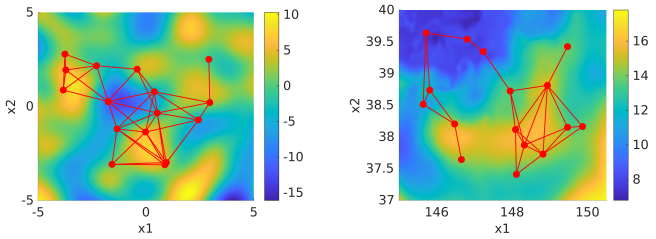
4. SIMULATIONS

We conducted experiments with both artificial and real-world datasets to study the performance of the proposed algorithms in fully-distributed networks. For each simulation, a total number of $N = 4000$ measurements are randomly generated, and then equally distributed to $M = 16$ agents. The simulated algorithms include pxADMM, pxADMM$_{\text{fd}}$, pxADMM$_{\text{fd,fast}}$, pxADMM$_{\text{afd}}$ and pxADMM$_{\text{afd,fast}}$. The centralized ADMM is not compared since the state of the art pxADMM is already the baseline. pxADMM is simulated in a MAS with a central station, and the others are simulated in connected random networks. For all the simulations, the stop criteria for convergence is met when step size $\|\boldsymbol{\theta}_m^{t+1} - \boldsymbol{\theta}_m^t\|_2 \le 10^{-4}, \forall m \in \{1, \cdots, M\}$. Parameters used are $\rho = 400$, $L = 4000$, $S = 2$ and $\tau = 10$. Matlab code can be accessed at https://github.com/hantyou/Distributed-GP-Hyperpar-Optim.

4.1. 2D-GP dataset

Figure 1(a) shows an example 2D GP field generated with a RBF kernel and the following predefined hyperparameters $\{\sigma_f = 5, l_1 = 1, l_2 = 1, \sigma_n = 0.316\}$. All the algorithms are initialized with $\{\sigma_f = 3, l_1 = 2, l_2 = 2, \sigma_n = 1\}$. The converged values of the simulated algorithms are listed in Table 1. The plots of step size and NLL, against iterations for the algorithms are compared in Figure 2.

The results show that all the algorithms converge to the expected values. The conventional pxADMM with a central station converges the fastest, which is expected. Our proposed fully-distributed algorithms pxADMM$_{\text{fd}}$ and pxADMM$_{\text{fd,fast}}$ converge, and are approximately 6.8 and 2.7 times slower than pxADMM for the given experimental setup. This performance is expected in fully-distributed networks since more iterations are needed for agents to exchange information without the assistance of a central station. The proposed asynchronous extensions have similar performance, as their synchronous counterparts.



(a) 2D-GP (b) GHRSST

Fig. 1. The red nodes and lines respectively indicate the agent positions and communication links.

Table 1. Results of hyperparameters optimization

Methods	$\{\sigma_f, l_1, l_2, \sigma_n\}$	Iterations
pxADMM [14]	$\{5.078, 1.007, 1.031, 0.317\}$	433
pxADMM$_{\text{fd}}$	$\{4.934, 0.999, 1.024, 0.316\}$	2952
pxADMM$_{\text{fd,fast}}$	$\{5.070, 1.012, 1.034, 0.316\}$	1098
pxADMM$_{\text{afd}}$	$\{4.949, 1.000, 1.025, 0.316\}$	2624
pxADMM$_{\text{afd,fast}}$	$\{5.076, 1.012, 1.034, 0.316\}$	1112

Table 2. Results of hyperparameters optimization

Methods	$\{\sigma_f, l_1, l_2, \sigma_n\}$	Iterations
pxADMM [14]	$\{8.234, 1.331, 0.652, 0.535\}$	1860
pxADMM$_{\text{fd}}$	$\{7.546, 1.301, 0.640, 0.534\}$	6574
pxADMM$_{\text{fd,fast}}$	$\{8.179, 1.337, 0.656, 0.537\}$	3962
pxADMM$_{\text{afd}}$	$\{7.634, 1.305, 0.641, 0.534\}$	8276
pxADMM$_{\text{afd,fast}}$	$\{8.214, 1.338, 0.657, 0.537\}$	4484

4.2. Real-world dataset - GHRSST

The Group for High Resolution Sea Surface Temperature (GHRSST) [18] is a dataset containing high resolution daily global sea surface temperature maps. The region in longitude [145.0, 150.5] and latitude [37.0, 40.0] is cropped from the map collected on 2nd April, 2022 as the underlying field. Since the true hyperparameters are unknown, the results of pxADMM are used as the baseline that the proposed algorithms should converge to. Convergence plots are omitted since they follow similar patterns as those in Figure 2. We present the final converged valued achieved by the proposed fully-distributed algorithms in Table 2, where we notice that

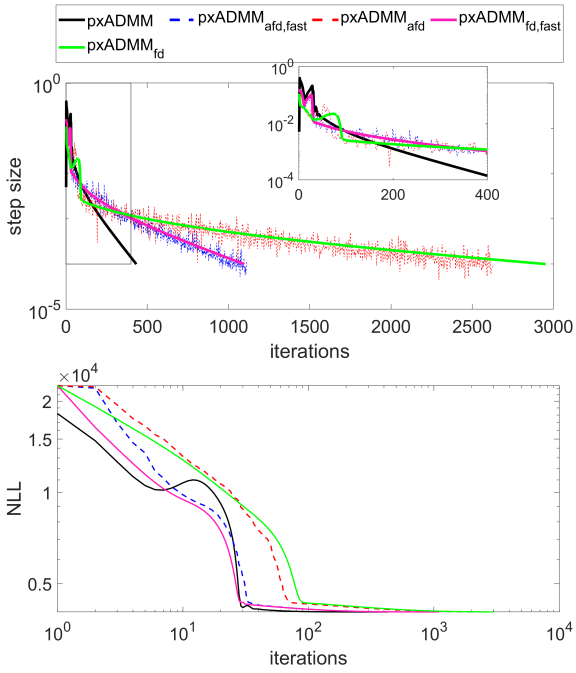


Fig. 2. Convergence curves with 2D-GP dataset. The upper plot shows the hyperparameter step size curve in terms of iterations. The lower plot shows the NLL value curves.

our proposed solutions converge to the expected hyperparameters obtained by pxADMM. The results are also consistent with the field pattern shown in Figure 1(b), where the temperature changes slower in the horizontal direction than the vertical direction.

5. CONCLUSION

In this paper, we proposed two fully-distributed hyperparameter optimization algorithms based on pxADMM through applying edge-based constraints. The proposed pxADMM$_{\text{fd}}$ approximates the local likelihood around the local hyperparameters, while pxADMM$_{\text{fd,fast}}$ approximates the likelihood around local auxiliary variables. We also propose corresponding asynchronous algorithms, which adopt partial barrier and bounded delay strategies. Simulation results show that pxADMM$_{\text{fd}}$ and pxADMM$_{\text{fd,fast}}$ successfully converge to the expected values in a fully-distributed MAS, at the cost of more iterations, as compared to conventional pxADMM which uses a central station. The theoretical proof for the convergence of the proposed algorithms is ongoing, which we aim to present in our follow up work. Further investigation on distributed Bayesian [19] and sparsity-aware hyperparameter optimization [20] can be explored for more resource efficient solutions, in addition to positioning the proposed solutions in the larger context of distributed online learning with kernels [21].

6. REFERENCES

[1] N. A. Atanasov, J. Le Ny, and G. J. Pappas, "Distributed algorithms for stochastic source seeking with mobile robot networks," *Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME*, vol. 137, no. 3, pp. 1–9, 2015.

[2] J. Choi, S. Oh, and R. Horowitz, "Distributed learning and cooperative control for multi-agent systems," *Automatica*, vol. 45, no. 12, pp. 2802–2814, 2009.

[3] K. Tiwari, V. Honore, S. Jeong, N. Y. Chong, and M. P. Deisenroth, "Resource-constrained decentralized active sensing for multi-robot systems using distributed Gaussian processes," *International Conference on Control, Automation and Systems*, vol. 0, no. Iccas, pp. 13–18, 2016.

[4] N. E. Leonard, D. A. Paley, F. Lekien, R. Sepulchre, D. M. Fratantoni, and R. E. Davis, "Collective motion, sensor networks, and ocean sampling," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 48–74, 2007.

[5] A. Datta, S. Banerjee, A. O. Finley, and A. E. Gelfand, "Hierarchical Nearest-Neighbor Gaussian Process Models for Large Geostatistical Datasets," *Journal of the American Statistical Association*, vol. 111, no. 514, pp. 800–812, 2016.

[6] A. Datta, S. Banerjee, A. O. Finley, N. A. S. Hamm, and M. Schaap, "Nonseparable dynamic nearest neighbor Gaussian process models for large spatio-temporal data with an application to particulate matter analysis," *The Annals of Applied Statistics*, vol. 10, no. 3, pp. 1286–1316, 2016.

[7] F. Yin and F. Gunnarsson, "Distributed Recursive Gaussian Processes for RSS Map Applied to Target Tracking," *IEEE Journal on Selected Topics in Signal Processing*, vol. 11, no. 3, pp. 492–503, 2017.

[8] E. Snelson and Z. Ghahramani, "Sparse Gaussian processes using pseudo-inputs," *Advances in Neural Information Processing Systems*, pp. 1257–1264, 2005.

[9] J. Bergstra, R. Bardenet, B. Kégl, and Y. Bengio, "Implementations of algorithms for hyper-parameter optimization," in *NIPS Workshop on Bayesian optimization*, vol. 29, 2011.

[10] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning.* The MIT Press, 2006.

[11] V. Tresp, "A Bayesian Committee Machine," *Neural Computation*, vol. 12, no. 11, pp. 2719–2741, 2000.

[12] M. P. Deisenroth and J. W. Ng, "Distributed Gaussian processes," in *32nd International Conference on Machine Learning, ICML 2015*, vol. 37, pp. 1481–1490, 2015.

[13] Y. Xu, F. Yin, W. Xu, J. Lin, and S. Cui, "Wireless Traffic Prediction with Scalable Gaussian Process: Framework, Algorithms, and Verification," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1291–1306, 2019.

[14] A. Xie, F. Yin, Y. Xu, B. Ai, T. Chen, and S. Cui, "Distributed Gaussian Processes Hyperparameter Optimization for Big Data Using Proximal ADMM," *IEEE Signal Processing Letters*, vol. 26, no. 8, pp. 1197–1201, 2019.

[15] M. Hong, Z. Q. Luo, and M. Razaviyayn, "Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems," *SIAM Journal on Optimization*, vol. 26, no. 1, pp. 337–364, 2016.

[16] G. Zhang and R. Heusdens, "Distributed Optimization Using the Primal-Dual Method of Multipliers," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 4, no. 1, pp. 173–187, 2018.

[17] R. Zhang and J. T. Kwok, "Asynchronous distributed ADMM for consensus optimization," *31st International Conference on Machine Learning, ICML 2014*, vol. 5, no. 2, pp. 3689–3697, 2014.

[18] NASA/JPL, "GHRSST Level 4 MUR Global Foundation Sea Surface Temperature Analysis (v4.1)," 2015, [Online]. Available: https://podaac.jpl.nasa.gov/dataset/MUR-JPL-L4-GLOB-v4.1. [Accessed: 25-Apr.-2022].

[19] M. T. Young, J. Hinkle, A. Ramanathan, and R. Kannan, "HyperSpace: Distributed Bayesian Hyperparameter Optimization," *Proceedings - 2018 30th International Symposium on Computer Architecture and High Performance Computing, SBAC-PAD 2018*, no. 1, pp. 339–347, 2019.

[20] L. Yang, K. Wang, and L. Mihaylova, "Online Sparse Multi-Output Gaussian Process Regression and Learning," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 5, no. 2, pp. 258–272, 2019.

[21] A. Koppel, S. Paternain, C. Richard, and A. Ribeiro, "Decentralized online learning with kernels," *IEEE Transactions on Signal Processing*, vol. 66, no. 12, pp. 3240–3255, 2018.