Journal of Machine Learning Research 22 (2021) 1-35 Submitted 1/20; Published 5/21

COKE: Communication-Censored Decentralized Kernel Learning

Ping Xu PXU3@GMU.EDU
Yue Wang YWANG56@GMU.EDU
Xiang Chen XCHEN26@GMU.EDU
Zhi Tian ZTIAN1@GMU.EDU
Department of Electrical and Computer Engineering, George Mason University
Fairfax, VA 22030, USA

Editor: Corinna Cortes

Abstract

This paper studies the decentralized optimization and learning problem where multiple interconnected agents aim to learn an optimal decision function defined over a reproducing kernel Hilbert space by jointly minimizing a global objective function, with access to their own locally observed dataset. As a non-parametric approach, kernel learning faces a major challenge in distributed implementation: the decision variables of local objective functions are data-dependent and thus cannot be optimized under the decentralized consensus framework without any raw data exchange among agents. To circumvent this major challenge, we leverage the random feature (RF) approximation approach to enable consensus on the function modeled in the RF space by data-independent parameters across different agents. We then design an iterative algorithm, termed DKLA, for fast-convergent implementation via ADMM. Based on DKLA, we further develop a communication-censored kernel learning (COKE) algorithm that reduces the communication load of DKLA by preventing an agent from transmitting at every iteration unless its local updates are deemed informative. Theoretical results in terms of linear convergence guarantee and generalization performance analysis of DKLA and COKE are provided. Comprehensive tests on both synthetic and real datasets are conducted to verify the communication efficiency and learning effectiveness of COKE.[1]

Keywords: Decentralized nonparametric learning, reproducing kernel Hilbert space, random features, ADMM, communication censoring.

1. Introduction

Decentralized learning has attracted extensive interest in recent years, largely due to the explosion of data generated everyday from mobile sensors, social media services, and other networked multi-agent applications (Worden and Manson, 2006; Ilyas et al., 2013; Facchinei et al., 2015; Demarie and Sabia, 2019). In many of these applications, the observed data are usually kept private at local sites without being aggregated to a fusion center, either due to the prohibitively high cost of raw data transmission or privacy concerns. Meanwhile,

1. Preliminary results in this paper were presented in part at the 2019 IEEE Data Science Workshop (Xu et al., 2019).

each agent in the network only communicates with its one-hop neighbors within its local area to save transmission power. Such localized data processing and transmission obviate the implementation of any centralized learning techniques. Under this circumstance, this article focuses on the decentralized learning problem where a network of distributed agents aim to collaboratively learn a functional model describing the global data with only access to their own locally observed datasets.

To learn the functional model that is often nonlinear and complex, nonparametric kernel methods are widely appreciated thanks to the "kernel trick" that makes some well-behaved linear learning algorithms applicable in a high-dimensional implicit feature space, without explicit mapping from data to that feature space (Shawe-Taylor et al., 2004; Hofmann et al., 2008; Pérez-Cruz and Bousquet, 2004). However, in the absence of any raw data sharing or aggregation, it is challenging to directly apply them to a decentralized multi-agent setting and solve them under the consensus optimization framework using algorithms such as decentralized alternating direction method of multipliers (ADMM) (Shi et al., 2014). This is because decentralized learning relies on solving local optimization problems and then aggregating the updates on the local decision variables over the network through one-hop communications in an iterative manner (Nedić et al., 2016). Unfortunately, these decision variables of local objective functions resulted from the kernel trick are data-dependent and thus cannot be optimized in the absence of raw data exchange under the decentralized consensus framework.

There are several works applying kernel methods in decentralized learning for various applications under different settings (Predd et al., 2006; Mitra and Bhatia, 2014; Gao et al., 2015; Chouvardas and Draief, 2016; Shin et al., 2016, 2018; Koppel et al., 2018). These works, however, either assume that agents have access to their neighbors' observed raw data (Predd et al., 2006) or require agents to transmit their raw data to their neighbors (Koppel et al., 2018) to ensure consensus through collaborative learning. These assumptions may not be valid in many practical applications that involve users' private data. Moreover, standard kernel learning for big data faces the curse of dimensionality when the number of training examples increases (Shawe-Taylor et al., 2004). For example, in (Mitra and Bhatia, 2014; Chouvardas and Draief, 2016), the nonlinear function learned at each node is represented as a weighted combination of kernel functions centered on its local observed data. As a result, each agent needs to transmit both the weights of kernel functions and its local data to its neighbors at every iterative step to guarantee consensus of the common prediction function. Thus, both the computation and communication resources are demanding in the distributed implementation. To alleviate the curse of dimensionality problem, Gao et al. (2015) and Koppel et al. (2018) have developed compression techniques such as data selection and sparse subspace projection, respectively, but these techniques typically incur considerable extra computation, and still involve raw data exchange with no alleviation to the data privacy concern. Furthermore, when computation cost is more affordable than the communication in the big data scenario, communication cost of the iterative learning algorithms becomes the bottleneck for efficient distributed learning (McMahan et al., 2016). Therefore, it is crucial to design communication-efficient distributed kernel learning algorithms with data privacy protection.

1.1 Related work

This work lies at the intersection of non-parametric kernel methods, decentralized learning with batch-form data, and communication-efficient iterative implementation. Related work to these three subjects is reviewed below.

Centralized kernel learning. Centralized kernel methods assume data are collected and processed by a single server and are known to suffer from the curse of dimensionality for large-scale learning tasks. To mitigate their computational complexity, various dimensionality reduction techniques are developed for both batch-form or online streaming learning, including stochastic approximation (Bucak et al., 2010; Gu et al., 2018), restricting the number of function parameters (Gomes and Krause, 2010; Wang et al., 2012; Zhang et al., 2013; Le et al., 2016; Koppel et al., 2017), and approximating the kernel during training (Honeine, 2015; Engel et al., 2004; Richard et al., 2008; Drineas and Mahoney, 2005; Dai et al., 2014; Lu et al., 2016; Sheikholeslami et al., 2018; Rahimi and Recht, 2008; Băzăvan et al., 2012; Nguyen et al., 2017). Among them, random feature (RF) mapping methods have gained popularity thanks to their ability to map the large-scale data into a RF space of much reduced dimension by approximating the kernel with a fixed (small) number of random features, which thus circumvents the curse of dimensionality problem (Rahimi and Recht, 2008; Dai et al., 2014; Băzăvan et al., 2012; Nguyen et al., 2017). Enforcing orthogonality on random features can greatly reduce the error in kernel approximation (Yu et al., 2016; Shen et al., 2018), and the learning performance of RF-based methods is evaluated in (Bach, 2017; Rudi and Rosasco, 2017; Li et al., 2018).

Decentralized kernel learning. For the decentralized kernel learning problem relevant to our work (Mitra and Bhatia, 2014; Gao et al., 2015; Chouvardas and Draief, 2016; Koppel et al., 2018), gradient descent is conducted locally at each agent to update its learning model, followed by diffusion-based information exchange among agents. However, these methods either assume that agents have access to their neighbors' observed raw data or require agents to transmit their raw data to their neighbors to ensure convergence on the prediction function. For the problem studied in this article where the observed data are only locally available, these methods are not applicable since there are no common decision parameters for consensus without any raw data exchange. Moreover, these methods operate in the kernel space parameterized by training data, and still encounter the curse of dimensionality when the local dataset goes large. Though data selection (Gao et al., 2015) and subspace projection (Koppel et al., 2018) are adopted to alleviate the curse of dimensionality problem, they typically require significant extra computational resources. RF mapping (Rahimi and Recht, 2008) offers a viable approach to overcome these issues, by having all agents map their datasets of various sizes onto the same RF space. For instance, Bouboulis et al. (2018) proposes a diffusion-based combine-then-adapt (CTA) method that achieves consensus on the model parameters in the RF space for the online learning problem, without the exchange of raw data. Though the batch-form counterpart of online CTA can be developed for off-line learning, the convergence speed of the diffusion-based method is relatively slow compared with higher-order methods such as ADMM (Liu et al., 2019).

Communication-efficient optimization. Communication-efficient algorithms for decentralized optimization and learning problems have attracted attention when data movement among computing nodes becomes a bottleneck due to the high latency and limited band-

width of decentralized networks. To reduce the communication cost, one way is to transmit the compressed information by quantization (Zhu et al., 2016; Alistarh et al., 2017; Zhang et al., 2019) or sparsification (Stich et al., 2018; Alistarh et al., 2018; Wangni et al., 2018; Harrane et al., 2018). However, these methods only reduce the required bandwidth at each communication round, not the number of rounds or the number of transmissions. Alternatively, some works randomly select a number of nodes for broadcasting/communication and operate asynchronous updating to reduce the number of transmissions per iteration (Mota et al., 2013; Li et al., 2014; Jaggi et al., 2014; Arablouei et al., 2015; McMahan et al., 2016; Yin et al., 2018; Yu et al., 2019). In contrast to random node selection, a more intuitive way is to evaluate the importance of a message in order to avoid unnecessary transmissions (Chen et al., 2018; Liu et al., 2019; Li et al., 2019b). This is usually implemented by adopting a censoring scheme to adaptively decide if a message is informative enough to be transmitted during the iterative optimization process. Other efforts to improve the communication efficiency are made by accelerating the convergence speed of the iterative algorithm implementation (Shamir et al., 2014; Reddi et al., 2016; Li et al., 2019a).

1.2 Contributions

This paper develops communication-efficient decentralized kernel learning algorithms under the consensus optimization framework without any central coordination or raw data exchange among agents for built-in privacy protection. Relative to prior art, our contributions are summarized as follows.

- We first formulate the decentralized multi-agent kernel learning problem as a decentralized consensus optimization problem in the RF space. Since most machine learning scenarios can afford plenty computational capability but limited communication resources, we solve this problem with ADMM, which has shown fast convergence at the expense of relatively high computation cost per iteration (Shi et al., 2014). To the best of our knowledge, this is the first work to solve decentralized kernel learning in the RF space by ADMM without any raw data exchange. The key of our proposed Decentralized Kernel Learning via ADMM (DKLA) algorithm is to apply RF mapping, which not only reduces the computational complexity but also enables consensus on a set of model parameters of fixed size in the RF space. In addition, since no raw data is exchanged among agents and the mapping from the original data space to the RF space is not one-to-one mapping, data privacy is protected to a certain level.

- To increase the communication efficiency, we further develop a COmmunication-censored KErnel learning (COKE) algorithm, which achieves desired learning performance given limited communication resources and energy supply. Specifically, we devise a simple yet powerful censoring strategy to allow each user to autonomously skip unnecessary communications when its local update is not informative enough for transmission, without aid of a central coordinator. In this way, the communication efficiency can be boosted at almost no sacrifice of the learning performance. When the censoring strategy is absent, COKE degenerates to DKLA.

- In addition, we conduct theoretical analysis in terms of both functional convergence and generalization performance to provide guidelines for practical implementations of

our proposed algorithms. We show that the individually learned functional at each agent through DKLA and COKE both converges to the optimal one at a linear rate under mild conditions. For the generalization performance, we show that $O(\sqrt{T}\log d_{\mathbf{K}}^{\lambda})$ features are sufficient to ensure $O(1/\sqrt{T})$ learning risk for the decentralized kernel ridge regression problem, where $d_{\mathbf{K}}^{\lambda}$ is the number of effective degrees of freedom that will be defined in Section 4.2 and T is the total number of samples.

- Finally, we test the performance of our proposed DKLA and COKE algorithms on both synthetic and real datasets. The results corroborate that both DKLA and COKE exhibit attractive learning performance and COKE is highly communication-efficient.

1.3 Organization and notation of the paper

Organization. Section 2 formulates the problem of non-parametric learning and highlights the challenges in applying traditional kernel methods in the decentralized setting. Section 3 develops the decentralized kernel learning algorithms, including both DKLA and COKE. Section 4 presents the theoretical results and Section 5 reports the numerical tests using both synthetic data and real datasets. Concluding remarks are provided in Section 6.
Notation. $\mathbb{R}$ denotes the set of real numbers. $\|\cdot\|_2$ denotes the Euclidean norm of vectors and $\|\cdot\|_F$ denotes the Frobenius norm of matrices. $|\cdot|$ denotes the cardinality of a set. $\mathbf{A}$, $\mathbf{a}$, and a denotes a matrix, a vector, and a scalar, respectively.

2. Problem Statement

This section reviews basics of kernel-based learning and decentralized optimization, introduces notation, and provides background needed for our novel DKLA and COKE schemes.

Consider a network of N agents interconnected over a fixed topology $\mathcal{G} = (\mathcal{N}, \mathcal{C}, \mathbf{A})$, where $\mathcal{N} = \{1, 2 \ldots, N\}$, $\mathcal{C} \subseteq \mathcal{N} \times \mathcal{N}$, and $\mathbf{A} \in \mathbb{R}^{N\times N}$ denote the agent set, the edge set and the adjacency matrix, respectively. The elements of $\mathbf{A}$ are $a_{in} = a_{ni} = 1$ when the unordered pair of distinct agents $(i, n) \in \mathcal{C}$, and $a_{in} = a_{ni} = 0$ otherwise. For agent i, its one-hop neighbors are in the set $\mathcal{N}_i = \{n|(n, i) \in \mathcal{C}\}$. The term *agent* used here can be a single computational system (e.g. a smart phone, a database, etc.) or a collection of co-located computational systems (e.g. data centers, computer clusters, etc.). Each agent only has access to its locally observed data composed of independently and identically distributed (i.i.d) input-label pairs $\{\mathbf{x}_{i,t}, y_{i,t}\}_{t=1}^{T_i}$ obeying an unknown probability distribution p on $\mathcal{X} \times \mathcal{Y}$, with $\mathbf{x}_{i,t} \in \mathbb{R}^d$ and $y_{i,t} \in \mathbb{R}$. The kernel learning task is to find a prediction function f that best describes the ensemble of all data from all agents. Suppose that f belongs to the reproducing kernel Hilbert space (RKHS) $\mathcal{H} := \{f|f(\mathbf{x}) = \sum_{t=1}^{\infty} \alpha_t \kappa(\mathbf{x}, \mathbf{x}_t)\}$ induced by a positive semidefinite kernel $\kappa(\mathbf{x}, \mathbf{x}_t) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ that measures the similarity between $\mathbf{x}$ and $\mathbf{x}_t$, for all $\mathbf{x}, \mathbf{x}_t \in \mathcal{X}$. In a decentralized setting with privacy concern, this means that each agent has to be able to learn the global function $f \in \mathcal{H}$ such that $y_{i,t} = f(\mathbf{x}_{i,t}) + e_{i,t}$ for $\{\{\mathbf{x}_{i,t}, y_{i,t}\}_{t=1}^{T_i}\}_{i=1}^{N}$, without exchange of any raw data and in the absence of a fusion center, where the error terms $e_{i,t}$ are minimized according to certain optimality metric.

To evaluate the learning performance, a nonnegative loss function $\ell(y, \hat{y})$ is utilized to measure the difference between the true label value y and the predicted value $\hat{y} = f(\mathbf{x})$. Some common loss functions include the quadratic loss $\ell(y, \hat{y}) = (y - \hat{y})^2$ for regression

tasks, the hinge loss $\ell(y,\hat{y}) = \max(0, 1-y\hat{y})$ and the logistic loss $\ell(y,\hat{y}) = \log(1+e^{-y\hat{y}})$ for binary classification tasks. The above mentioned loss functions are all convex with respect to $\hat{y}$. The learning problem is then to minimize the expected risk of the prediction function:

$$R(f) = \int_{\mathcal{X}\times\mathcal{Y}} \ell(f(\mathbf{x}), y) dp(\mathbf{x}, y), \tag{1}$$

which indicates the generalization ability of f to new data.

However, the distribution p is unknown in most learning tasks. Therefore, minimizing $R(f)$ is not applicable. Instead, given the finite number of training examples, the problem turns to minimizing the empirical risk:

$$\min_{f\in\mathcal{H}} \quad \hat{R}(f) := \sum_{i=1}^{N} \hat{R}_i(f), \tag{2}$$

where $\hat{R}_i(f)$ is the local empirical risk for agent i given by

$$\hat{R}_i(f) = \frac{1}{T_i}\sum_{t=1}^{T_i} \ell(f(\mathbf{x}_{i,t}), y_{i,t}) + \lambda_i \|f\|_{\mathcal{H}}^2, \tag{3}$$

with $\|\cdot\|_{\mathcal{H}}$ being the norm associated with $\mathcal{H}$, and $\lambda_i > 0$ being a regularization parameter that controls over-fitting.

The representer theorem states that the minimizer of a regularized empirical risk functional defined over a RKHS can be represented as a finite linear combination of kernel functions evaluated on the data pairs from the training dataset (Schölkopf et al., 2001). If $\{\{\mathbf{x}_{i,t}, y_{i,t}\}_{t=1}^{T_i}\}_{i=1}^{N}$ are centrally available at a fusion center, the minimizer of (2) admits

$$f^\star(\mathbf{x}) = \sum_{i=1}^{N}\sum_{t=1}^{T_i} \alpha_{i,t}\kappa(\mathbf{x}, \mathbf{x}_{i,t}) := \boldsymbol{\alpha}^\top \boldsymbol{\kappa}(\mathbf{x}), \tag{4}$$

where $\boldsymbol{\alpha} = [\alpha_{1,1}, \ldots, \alpha_{N,T_N}]^\top \in \mathbb{R}^T$ is the coefficient vector to be learned, $T = \sum_{i=1}^{N} T_i$ is the total number of samples, and $\boldsymbol{\kappa}(\mathbf{x}) = [\kappa(\mathbf{x}, \mathbf{x}_{1,1}), \ldots, \kappa(\mathbf{x}, \mathbf{x}_{N,T_N})]^\top \in \mathbb{R}^T$ is the kernel function parameterized by the global data $\mathbf{X}_T := \{\{\mathbf{x}_{i,t}\}_{t=1}^{T_i}\}_{i=1}^{N}$ from all agents, for any $\mathbf{x}$. In RKHS, since $\langle \kappa(\mathbf{x}_t, \mathbf{x}), \kappa(\mathbf{x}_\tau, \mathbf{x})\rangle_{\mathcal{H}} = \kappa(\mathbf{x}_t, \mathbf{x}_\tau)$, it yields $\|f\|_{\mathcal{H}}^2 = \boldsymbol{\alpha}^\top \mathbf{K}\boldsymbol{\alpha}$, where $\mathbf{K}$ is the $T \times T$ kernel matrix that measures the similarity between any two data points in $\mathbf{X}_T$. In this way, the local empirical risk (3) can be reformulated as a function of $\boldsymbol{\alpha}$:

$$\hat{R}_i(\boldsymbol{\alpha}) := \frac{1}{T_i}\sum_{t=1}^{T_i} \ell(f^\star(\mathbf{x}_{i,t}), y_{i,t}) + \lambda_i \|f^\star\|_{\mathcal{H}}^2 = \frac{1}{T_i}\sum_{t=1}^{T_i} \ell(\boldsymbol{\alpha}^\top \boldsymbol{\kappa}(\mathbf{x}_{i,t}), y_{i,t}) + \lambda_i \boldsymbol{\alpha}^\top \mathbf{K}\boldsymbol{\alpha}. \tag{5}$$

Accordingly, (2) becomes

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^T} \quad \sum_{i=1}^{N} \hat{R}_i(\boldsymbol{\alpha}). \tag{6}$$

Relating the decentralized kernel learning problem with the decentralized consensus optimization problem, solving (6) is equivalent to solving

$$\begin{aligned} \min_{\{\boldsymbol{\alpha}_i\in\mathbb{R}^T\}_{i=1}^N} \quad & \sum_{i=1}^N \hat{R}_i(\boldsymbol{\alpha}_i) \\ \text{s.t.} \quad & \boldsymbol{\alpha}_i = \boldsymbol{\alpha}_n, \qquad \forall i, \quad \forall n \in \mathcal{N}_i, \end{aligned} \tag{7}$$

where $\boldsymbol{\alpha}_i$ and $\boldsymbol{\alpha}_n$ are the local copies of the global decision variable $\boldsymbol{\alpha}$ at agent i and agent n, respectively. The problem can then be solved by ADMM (Shi et al., 2014) or other primal dual methods (Terelius et al., 2011). However, it is worth noting that (7) reveals a subtle yet profound difference from a general optimization problem for parametric learning. That is, each local function $\hat{R}_i$ depends on not only the global decision variable $\boldsymbol{\alpha}$, but also the global data $\mathbf{X}_T$ because of the kernel terms $\boldsymbol{\kappa}(\mathbf{x}_{i,t})$ and $\mathbf{K}$. As a result, solving the local objective for agent i requires raw data from all other agents to obtain $\boldsymbol{\kappa}(\mathbf{x}_{i,t})$ and $\mathbf{K}$, which contradicts the situation that private raw data are only locally available. Moreover, notice that $\boldsymbol{\alpha}_i$ is of the same size T as that of the ensemble dataset, which incurs the curse of dimensionality and insurmountable computational cost when T becomes large, even when the obstacle of making all the data available to all agents is not of concern.

To resolve this issue, an alternative formulation is to associate a local prediction model $\bar{f}_i \in \mathcal{H}$ with each agent i, with $\bar{f}_i^\star = \sum_{t=1}^{T_i} \bar{\alpha}_{i,t}\kappa(\mathbf{x}, \mathbf{x}_{i,t}) = \bar{\boldsymbol{\alpha}}_i^\top \boldsymbol{\kappa}_i(\mathbf{x})$ being the local optimal solution that only involves local data $\{\mathbf{x}_{i,t}\}_{t=1}^{T_i}$ (Ji et al., 2016). Specifically, $\bar{\boldsymbol{\alpha}}_i = [\bar{\alpha}_{i,1}, \ldots, \bar{\alpha}_{i,T_i}] \in \mathbb{R}^{T_i}$, and $\boldsymbol{\kappa}_i(\mathbf{x}) = [\kappa(\mathbf{x}, \mathbf{x}_{i,1}), \ldots, \kappa(\mathbf{x}, \mathbf{x}_{i,T_i})]^\top \in \mathbb{R}^{T_i}$ is parameterized by the local data $\{\mathbf{x}_{i,t}\}_{t=1}^{T_i}$ only. In this way, the local cost function becomes

$$\hat{R}_i(\bar{\boldsymbol{\alpha}}_i) := \frac{1}{T_i}\sum_{t=1}^{T_i} \ell(\bar{f}_i^\star(\mathbf{x}_{i,t}), y_{i,t}) + \lambda_i \|\bar{f}_i^\star\|_{\mathcal{H}}^2 = \frac{1}{T_i}\sum_{t=1}^{T_i} \ell(\bar{\boldsymbol{\alpha}}_i^\top \boldsymbol{\kappa}_i(\mathbf{x}_{i,t}), y_{i,t}) + \lambda_i \bar{\boldsymbol{\alpha}}_i^\top \mathbf{K}_i \bar{\boldsymbol{\alpha}}_i, \tag{8}$$

where $\mathbf{K}_i$ is of size $T_i \times T_i$ and depends on local data only. With (8), the optimization problem (7) is then modified to

$$\begin{aligned} \min_{\{\bar{\boldsymbol{\alpha}}_i\in\mathbb{R}^{T_i}\}_{i=1}^N} \quad & \sum_{i=1}^N \hat{R}_i(\bar{\boldsymbol{\alpha}}_i) \\ \text{s.t.} \quad & \bar{f}_n(\mathbf{x}_{i,t}) = \bar{f}_i(\mathbf{x}_{i,t}), \qquad \forall i, \quad \forall n \in \mathcal{N}_i, \quad t = 1, \ldots, T_i, \end{aligned} \tag{9}$$

and can be solved distributedly by ADMM. Note that the consensus constraint is the learned prediction values $\bar{f}_i(\mathbf{x})$, not the parameters $\bar{\boldsymbol{\alpha}}_i$. This is because $\bar{\boldsymbol{\alpha}}_i$ are data-dependent and may have different sizes at different agents (the dimension of $\bar{\boldsymbol{\alpha}}_i$ equals to the number of training samples at agent i), and cannot be directly optimized through consensus.

Still, this method has four drawbacks. Firstly, it is necessary to associate a local learning model $\bar{f}_i$ to each agent i for the decentralized implementation. However, the local learning model $\bar{f}_i$ and the global optimal model f in (2) may not be the same because different local training data are used. Therefore, the optimization problem (9) is only an approximation of (2). Even with the equality constraint to minimize the gap between the decentralized learning output and the optimal centralized one, the approximation performance is not guaranteed. Besides, the functional consensus constraint still requires raw data exchange

among agents in order for agent $n \in \mathcal{N}_i$ to be able to compute the values $\bar{f}_n(\mathbf{x}_{i,t})$ from agent i's data $\mathbf{x}_{i,t}$, for $i \neq n$. Apparently, this violates the privacy-protection requirement for practical applications. In addition, when T_i is large, both the storage and computational costs are high for each agent due to the curse of dimensionality problem at the local sites. Lastly, the frequent local communication is resource-consuming under communication constraints. To circumvent all these obstacles, the goal of this paper is to develop efficient decentralized algorithms that protect privacy and conserve communication resources.

3. Algorithm Development

In this section, we leverage the RF approximation and ADMM to develop our algorithms. We first introduce the RF mapping method. Then, we devise the DKLA algorithm that globally optimizes a shared learning model for the multi-agent system. Finally, we take into consideration of the limited communication resources in large-scale decentralized networks and develop the COKE algorithm. Both DKLA and COKE are computationally efficient and protect data privacy at the same time. Further, COKE is communication efficient.

3.1 RF-based kernel learning

As stated in previous sections, standard kernel methods incur the curse of dimensionality issue when the data size grows large. To make kernel methods scalable for a large dataset, RF mapping is adopted for approximation by using the shift-invariance property of kernel functions (Rahimi and Recht, 2008).

For a shift-invariant kernel that satisfies $\kappa(\mathbf{x}_t, \mathbf{x}_\tau) = \kappa(\mathbf{x}_t - \mathbf{x}_\tau),\ \forall t,\ \forall \tau$, if $\kappa(\mathbf{x}_t - \mathbf{x}_\tau)$ is absolutely integrable, then its Fourier transform $p_\kappa(\boldsymbol{\omega})$ is guaranteed to be nonnegative ($p_\kappa(\boldsymbol{\omega}) \geq 0$), and hence can be viewed as its probability density function (pdf) when κ is scaled to satisfy $\kappa(0) = 1$ (Bochner, 2005). Therefore, we have

$$\kappa(\mathbf{x}_t, \mathbf{x}_\tau) = \int p_\kappa(\boldsymbol{\omega}) e^{j\boldsymbol{\omega}^\top(\mathbf{x}_t - \mathbf{x}_\tau)} d\boldsymbol{\omega} := \mathbb{E}_{\boldsymbol{\omega}}[e^{j\boldsymbol{\omega}^\top(\mathbf{x}_t - \mathbf{x}_\tau)}] = \mathbb{E}_{\boldsymbol{\omega}}[\phi(\mathbf{x}_t, \boldsymbol{\omega})\phi^*(\mathbf{x}_\tau, \boldsymbol{\omega})], \tag{10}$$

where $\mathbb{E}$ denotes the expectation operator, $\phi(\mathbf{x}, \boldsymbol{\omega}) := e^{j\boldsymbol{\omega}^\top \mathbf{x}}$ with $\boldsymbol{\omega} \in \mathbb{R}^d$, and $*$ is the complex conjugate operator. In (10), the first equality is the result of the Fourier inversion theorem, and the second equality arises by viewing $p_\kappa(\boldsymbol{\omega})$ as the pdf of $\boldsymbol{\omega}$. In this paper, we adopt a Gaussian kernel $\kappa(\mathbf{x}_t, \mathbf{x}_\tau) = \exp(-\|\mathbf{x}_\mathrm{t} - \mathbf{x}_\tau\|_2^2/(2\sigma^2))$, whose pdf is a normal distribution with $p_\kappa(\boldsymbol{\omega}) \sim \mathbf{N}(\mathbf{0}, \sigma^{-2}\mathbf{I})$.

The main idea of RF mapping is to randomly generate $\{\boldsymbol{\omega}_l\}_{l=1}^L$ from the distribution $p_\kappa(\boldsymbol{\omega})$ and approximate the kernel function $\kappa(\mathbf{x}_t, \mathbf{x}_\tau)$ by the sample average

$$\hat{\kappa}_L(\mathbf{x}_t, \mathbf{x}_\tau) := \frac{1}{L}\sum_{l=1}^{L} \phi(\mathbf{x}_t, \boldsymbol{\omega}_l)\phi^*(\mathbf{x}_\tau, \boldsymbol{\omega}_l) := \boldsymbol{\phi}_L^\dagger(\mathbf{x}_\tau)\boldsymbol{\phi}_L(\mathbf{x}_t), \tag{11}$$

where $\boldsymbol{\phi}_L(\mathbf{x}) := \sqrt{\frac{1}{L}}[\phi(\mathbf{x}, \boldsymbol{\omega}_1), \ldots, \phi(\mathbf{x}, \boldsymbol{\omega}_L)]^\top$ and $\dagger$ is the conjugate transpose operator.

The following real-valued mappings can be adopted to approximate $\kappa(\mathbf{x}_t, \mathbf{x}_\tau)$, both satisfying the condition $\mathbb{E}_{\boldsymbol{\omega}}[\phi_r(\mathbf{x}_t, \boldsymbol{\omega})^\top \phi_r(\mathbf{x}_\tau, \boldsymbol{\omega})] = \kappa(\mathbf{x}_t, \mathbf{x}_\tau)$ (Rahimi and Recht, 2008):

$$\phi_r(\mathbf{x}, \boldsymbol{\omega}) = [\cos(\boldsymbol{\omega}^\top \mathbf{x}), \sin(\boldsymbol{\omega}^\top \mathbf{x})]^\top, \tag{12}$$

$$\phi_r(\mathbf{x}, \boldsymbol{\omega}) = \sqrt{2}\cos(\boldsymbol{\omega}^\top \mathbf{x} + b), \tag{13}$$

where b is drawn uniformly from $[0, 2\pi]$.

With the real-valued RF mapping, the minimizer of (2) then admits the following form:

$$\hat{f}^\star(\mathbf{x}) = \sum_{i=1}^{N}\sum_{t=1}^{T_i} \alpha_{i,t} \boldsymbol{\phi}_L^\top(\mathbf{x}_{i,t}) \boldsymbol{\phi}_L(\mathbf{x}) = \boldsymbol{\theta}^\top \boldsymbol{\phi}_L(\mathbf{x}), \tag{14}$$

where $\boldsymbol{\theta}^\top := \sum_{i=1}^{N}\sum_{t=1}^{T_i} \alpha_{i,t} \boldsymbol{\phi}_L^\top(\mathbf{x}_{i,t})$ denotes the new decision vector to be learned in the RF space and $\boldsymbol{\phi}_L(\mathbf{x}) = \sqrt{\frac{1}{L}}[\phi_r(\mathbf{x}, \boldsymbol{\omega}_1), \ldots, \phi_r(\mathbf{x}, \boldsymbol{\omega}_L)]^\top$. If (12) is adopted, then $\boldsymbol{\phi}_L(\mathbf{x})$ and $\boldsymbol{\theta}$ are of size $2L$. Otherwise, if (13) is adopted, then $\boldsymbol{\phi}_L(\mathbf{x})$ and $\boldsymbol{\theta}$ are of size L. In either case, the size of $\boldsymbol{\theta}$ is fixed and does not increase with the number of data samples.

3.2 DKLA: Decentralized kernel learning via ADMM

Consider the decentralized kernel learning problem described in Section 2 and adopt the RF mapping described in Section 3.1. Let all agents in the network have the same set of random features, i.e., $\{\boldsymbol{\omega}_l\}_{l=1}^L$. Plugging (14) into the local cost function $\hat{R}_i(f)$ in (3) gives

$$\hat{R}_i(\boldsymbol{\theta}) := \frac{1}{T_i}\sum_{t=1}^{T_i} \ell(\hat{f}^\star(\mathbf{x}_{i,t}), y_{i,t}) + \lambda_i \|\hat{f}^\star\|_{\mathcal{H}}^2 = \frac{1}{T_i}\sum_{t=1}^{T_i} \ell(\boldsymbol{\theta}^\top \boldsymbol{\phi}_L(\mathbf{x}_{i,t}), y_{i,t}) + \lambda_i \|\boldsymbol{\theta}\|_2^2. \tag{15}$$

In (15), we have

$$\begin{aligned}\|\boldsymbol{\theta}\|_2^2 :&= (\sum_{i=1}^{N}\sum_{t=1}^{T_i} \alpha_{i,t}\boldsymbol{\phi}_L^\top(\mathbf{x}_{i,t}))(\sum_{n=1}^{N}\sum_{\tau=1}^{T_i} \alpha_{n,\tau}\boldsymbol{\phi}_L(\mathbf{x}_{n,\tau})) \\ &= \sum_{i=1}^{N}\sum_{t=1}^{T_i}\sum_{n=1}^{N}\sum_{\tau=1}^{T_i} \alpha_{i,t}\alpha_{n,\tau}\kappa(\mathbf{x}_{i,t}, \mathbf{x}_{n,\tau}) := \|\hat{f}^\star\|_{\mathcal{H}}^2.\end{aligned}$$

Therefore, with RF mapping, the centralized benchmark (2) becomes

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^L} \quad \sum_{i=1}^{N} \hat{R}_i(\boldsymbol{\theta}). \tag{16}$$

Here for notation simplicity, we denote the size of $\boldsymbol{\theta}$ by $L \times 1$, which can be achieved by adopting the real-valued mapping in (13). Adopting an alternative mapping such as (12) only changes the size of $\boldsymbol{\theta}$ while the algorithm development is the same. RF mapping is essential because it results in a common optimization parameter $\boldsymbol{\theta}$ of fixed size for all agents.

To solve (16) in a decentralized manner, we associate a model parameter $\boldsymbol{\theta}_i$ with each agent i and enforce the consensus constraint on neighboring agents i and n using an auxiliary

variable $\boldsymbol{\vartheta}_{in}$. Specifically, the RF-based decentralized kernel learning problem is formulated to jointly minimize the following objective function:

$$\begin{aligned} \min_{\{\boldsymbol{\theta}_i \in \mathbb{R}^L\}, \{\boldsymbol{\vartheta}_{in} \in \mathbb{R}^L\}} \quad & \sum_{i=1}^{N} \hat{R}_i(\boldsymbol{\theta}_i) \\ \text{s.t.} \quad & \boldsymbol{\theta}_i = \boldsymbol{\vartheta}_{in},\ \boldsymbol{\theta}_n = \boldsymbol{\vartheta}_{in}, \qquad \forall (i,n) \in \mathcal{C}. \end{aligned} \tag{17}$$

Note that the new decision variables $\boldsymbol{\theta}_i$ to be optimized are local copies of the global optimization parameter $\boldsymbol{\theta}$ and are of the same size for all agents. On the contrary, the decision variables $\bar{\boldsymbol{\alpha}}_i$ in (9) are data-dependent and may have different sizes. In addition, the size of $\boldsymbol{\theta}$ is L, which can be much smaller than that of $\boldsymbol{\alpha}$ (whose size equals to T) in (6). For big data scenarios where $L \ll T$, RF mapping greatly reduces the computational complexity. Moreover, as shown in the following, the updating of $\boldsymbol{\theta}$ does not involve any raw data exchange and the RF mapping from $\mathbf{x}$ to $\boldsymbol{\phi}_L(\mathbf{x})$ is not one-to-one mapping, therefore provides raw data privacy protection. Further, it is easy to set the regularization parameters λ_i to control over-fitting. Specifically, since the parameters $\boldsymbol{\theta}_i$ are of the same length among agents, we can set them to be $\lambda_i = \frac{1}{N}\lambda, \forall i$, where λ is the corresponding over-fitting control parameter assuming all data are collected at a center. In contrast, the regularization parameters λ_i in (5) depend on local data and need to satisfy $\lambda = \sum_{i=1}^{N} \lambda_i$, which is relatively difficult to tune in a large-scale network.

In the constraint, $\boldsymbol{\theta}_i$ are separable when $\boldsymbol{\vartheta}_{in}$ are fixed, and vice versa. Therefore, (17) can be solved by ADMM. Following (Shi et al., 2014), we develop the DKLA algorithm where each agent updates its local primal variable $\boldsymbol{\theta}_i$ and local dual variable $\boldsymbol{\gamma}_i$ by

$$\boldsymbol{\theta}_i^k := \arg\min_{\boldsymbol{\theta}_i} \left\{ \hat{R}_i(\boldsymbol{\theta}_i) + \rho |\mathcal{N}_i| \|\boldsymbol{\theta}_i\|_2^2 + \boldsymbol{\theta}_i^\top \left[\boldsymbol{\gamma}_i^{k-1} - \rho \sum_{n \in \mathcal{N}_i} \left(\boldsymbol{\theta}_i^{k-1} + \boldsymbol{\theta}_n^{k-1} \right) \right] \right\}, \tag{18a}$$

$$\boldsymbol{\gamma}_i^k = \boldsymbol{\gamma}_i^{k-1} + \rho \sum_{n \in \mathcal{N}_i} \left(\boldsymbol{\theta}_i^k - \boldsymbol{\theta}_n^k \right), \tag{18b}$$

where $|\mathcal{N}_i|$ is the cardinality of $\mathcal{N}_i$. The auxiliary variable $\boldsymbol{\vartheta}_{in}$ can be written as a function of $\boldsymbol{\theta}_i$ and then canceled out. Interested readers are referred to (Shi et al., 2014) for detailed derivation. The learning algorithm DKLA is outlined in Algorithm 1. Note that the random features need to be common to all agents, hence, in step 1, we restrict them to be drawn according to a common random seed. Algorithm 1 is fully decentralized since the updates of $\boldsymbol{\theta}_i$ and $\boldsymbol{\gamma}_i$ depend only on local and neighboring information.

3.3 COKE: Communication-censored decentralized kernel learning

From Sections 3.1 and 3.2, we can see that decentralized kernel learning in the RF space under the consensus optimization framework has much reduced computational complexity, thanks to the RF mapping technique that transforms the learning model into a smaller RF space. In this subsection, we consider the case when the communication resource is limited and we aim to further reduce the communication cost of DKLA. To start, we notice that in Algorithm 1, each agent i ($i \in \mathcal{N}$) maintains $2 + |\mathcal{N}_i|$ local variables at iteration k,

Algorithm 1 DKLA Run at Agent i

Require: Kernel κ, the number of random features L, and λ to control over-fitting; initialize local variables to $\boldsymbol{\theta}_i^0 = \mathbf{0}$, $\boldsymbol{\gamma}_i^0 = \mathbf{0}$; set step size $\rho > 0$;

1: Draw L i.i.d. samples $\{\boldsymbol{\omega}_l\}_{l=1}^L$ from $p_\kappa(\boldsymbol{\omega})$ according to a common random seed.
2: Construct $\{\boldsymbol{\phi}_L(\mathbf{x}_{i,t})\}_{t=1}^{T_i}$ using the random features $\{\boldsymbol{\omega}_l\}_{l=1}^L$ via (12) or (13).
3: **for** iterations $k = 1, 2, \cdots$ **do**
4: Update local variable $\boldsymbol{\theta}_i^k$ by (18a);
5: Transmit $\boldsymbol{\theta}_i^k$ to all neighbor n ($n \in \mathcal{N}_i$) and receive $\boldsymbol{\theta}_n^k$ from all neighbor n;
6: Update local dual variable $\boldsymbol{\gamma}_i^k$ by (18b).
7: **end for**

i.e., its local primal variable $\boldsymbol{\theta}_i^k$, local dual variable $\boldsymbol{\gamma}_i^k$ and $|\mathcal{N}_i|$ state variables $\boldsymbol{\theta}_n^k$ received from its neighbors. While the dual variable $\boldsymbol{\gamma}_i^k$ is kept locally for agent i, the transmission of its updated local variable $\boldsymbol{\theta}_i^k$ to its one-hop neighbors happens in every iteration, which consumes a large amount of communication bandwidth and energy along iterations for large-scale networks. In order to improve the communication efficiency, we develop the COKE algorithm by employing a censoring function at each agent to decide if a local update is informative enough to be transmitted.

To evaluate the importance of a local update at iteration k for agent i ($i \in \mathcal{N}$), we introduce a new state variable $\hat{\boldsymbol{\theta}}_i^{k-1}$ to record agent i's latest broadcast primal variable up to time $k-1$. Then, at iteration k, we define the difference between agent i's current state $\boldsymbol{\theta}_i^k$ and its previously transmitted state $\hat{\boldsymbol{\theta}}_i^{k-1}$ as

$$\boldsymbol{\xi}_i^k = \hat{\boldsymbol{\theta}}_i^{k-1} - \boldsymbol{\theta}_i^k, \tag{19}$$

and choose a censoring function as

$$H_i(k, \boldsymbol{\xi}_i^k) = \|\boldsymbol{\xi}_i^k\|_2 - h_i(k), \tag{20}$$

where $\{h_i(k)\}$ is a non-increasing non-negative sequence. A typical choice for the censoring function is $H_i(k, \boldsymbol{\xi}_i^k) = \|\boldsymbol{\xi}_i^k\|_2 - v\mu^k$, where $\mu \in (0, 1)$ and $v > 0$ are constants. When $H_i(k, \boldsymbol{\xi}_i^k) < 0$, $\boldsymbol{\theta}_i^k$ is deemed not informative enough, hence will be censored and will not be transmitted to its neighbors.

When executing the COKE algorithm, each agent i maintains $3 + |\mathcal{N}_i|$ local variables at each iteration k. Comparing with the DKLA update in (18), the additional local variable is the state variable $\hat{\boldsymbol{\theta}}_i^k$ that records its latest broadcast primal variable up to time k. Moreover, the $|\mathcal{N}_i|$ state variables from its neighbors are $\hat{\boldsymbol{\theta}}_n^k$ that record the latest received primal variables from its neighbors, instead of the timely updated and broadcast variables $\boldsymbol{\theta}_n^k$ of its neighbors $n \in \mathcal{N}_i$. Though each agent still computes local updates at every step, its transmission to neighbors does not always occur, but is determined by the censoring criterion (20). To be specific, at each iteration k, if $H_i(k, \boldsymbol{\xi}_i^k) \geq 0$, then $\hat{\boldsymbol{\theta}}_i^k = \boldsymbol{\theta}_i^k$, and agent i is allowed to transmit its local primal variable $\boldsymbol{\theta}_i^k$ to its neighbors. Otherwise, $\hat{\boldsymbol{\theta}}_i^k = \hat{\boldsymbol{\theta}}_i^{k-1}$ and no information is transmitted. If agent i receives $\boldsymbol{\theta}_n^k$ from any neighbor n, then that neighbor's state variable kept by agent i becomes $\hat{\boldsymbol{\theta}}_n^k = \boldsymbol{\theta}_n^k$, otherwise, $\hat{\boldsymbol{\theta}}_n^k = \hat{\boldsymbol{\theta}}_n^{k-1}$.

Algorithm 2 COKE Run at Agent i

Require: Kernel κ, the number of random features L, the censoring thresholds $\{h_i(k)\}$, and λ to control over-fitting; initialize local variables to $\boldsymbol{\theta}_i^0 = \mathbf{0}$, $\hat{\boldsymbol{\theta}}_i^0 = \mathbf{0}$, $\boldsymbol{\gamma}_i^0 = \mathbf{0}$; set step size $\rho > 0$;

1: Draw L i.i.d. samples $\{\boldsymbol{\omega}_l\}_{l=1}^L$ from $p_\kappa(\boldsymbol{\omega})$ according to a common random seed.
2: Construct $\{\boldsymbol{\phi}_L(\mathbf{x}_{i,t})\}_{t=1}^{T_i}$ using the random features $\{\boldsymbol{\omega}_l\}_{l=1}^L$ via (12) or (13).
3: **for** iterations $k = 1, 2, \cdots$ **do**
4: Update local variable $\boldsymbol{\theta}_i^k$ by (21a);
5: Compute $\boldsymbol{\xi}_i^k = \hat{\boldsymbol{\theta}}_i^{k-1} - \boldsymbol{\theta}_i^k$;
6: If $H_i(k, \boldsymbol{\xi}_i^k) = \|\boldsymbol{\xi}_i^k\|_2 - h_i(k) \geq 0$, transmit $\boldsymbol{\theta}_i^k$ to neighbors and let $\hat{\boldsymbol{\theta}}_i^k = \boldsymbol{\theta}_i^k$; else do not transmit and let $\hat{\boldsymbol{\theta}}_i^k = \hat{\boldsymbol{\theta}}_i^{k-1}$;
7: If receives $\boldsymbol{\theta}_n^k$ from neighbor n, let $\hat{\boldsymbol{\theta}}_n^k = \boldsymbol{\theta}_n^k$; else let $\hat{\boldsymbol{\theta}}_n^k = \hat{\boldsymbol{\theta}}_n^{k-1}$;
8: Update local dual variable $\boldsymbol{\gamma}_i^k$ by (21b).
9: **end for**

Consequently, agent i's local parameters are updated as follows:

$$\boldsymbol{\theta}_i^k := \arg\min_{\boldsymbol{\theta}_i} \left\{ \hat{R}_i(\boldsymbol{\theta}_i) + \rho|\mathcal{N}_i|\|\boldsymbol{\theta}_i\|_2^2 + \boldsymbol{\theta}_i^\top \left[\boldsymbol{\gamma}_i^{k-1} - \rho \sum_{n\in\mathcal{N}_i} \left(\hat{\boldsymbol{\theta}}_i^{k-1} + \hat{\boldsymbol{\theta}}_n^{k-1} \right) \right] \right\}, \tag{21a}$$

$$\boldsymbol{\gamma}_i^k = \boldsymbol{\gamma}_i^{k-1} + \rho \sum_{n\in\mathcal{N}_i} \left(\hat{\boldsymbol{\theta}}_i^k - \hat{\boldsymbol{\theta}}_n^k \right), \tag{21b}$$

with a censoring step conducted between (21a) and (21b). We outline the COKE algorithm in Algorithm 2.

The key feature of COKE is that agent i's local variables $\boldsymbol{\theta}_i^k$ and $\boldsymbol{\gamma}_i^k$ are updated all the time, but the transmission of $\boldsymbol{\theta}_i^k$ occurs only when the censoring condition is met. By skipping unnecessary transmissions, the communication efficiency of COKE is improved. It is obvious that large $\{h_i(k)\}$ saves more communication but may lead to divergence from the optimal solution $\boldsymbol{\theta}^*$ of (16), while small $\{h_i(k)\}$ does not contribute much to communication saving. Noticeably, DKLA is a special case of COKE when the communication censoring strategy is absent by setting $h_i(k) = 0, \forall i, k$.

4. Theoretical Guarantees

In this section, we perform theoretical analyses to address two questions related to the convergence properties of DKLA and COKE algorithms. First, do they converge to the globally optimal point, and if so, at what rate? Second, what is their achieved generalization performance in learning? Since DKLA is a special case of COKE, the analytic results of COKE, especially the second one, extend to DKLA straightforwardly. For theoretical analysis, we make the following assumptions.

Assumption 1 *The network with topology $\mathcal{G} = (\mathcal{N}, \mathcal{C}, \mathbf{A})$ is undirected and connected.*

Assumption 2 *The local cost functions $\hat{R}_i$ are strongly convex with constants $m_{\hat{R}_i} > 0$ such that $\forall i \in \mathcal{N}$, $\langle \nabla \hat{R}_i(\tilde{\boldsymbol{\theta}}_a) - \nabla \hat{R}_i(\tilde{\boldsymbol{\theta}}_b), \tilde{\boldsymbol{\theta}}_a - \tilde{\boldsymbol{\theta}}_b \rangle \geq m_{\hat{R}_i} \|\tilde{\boldsymbol{\theta}}_a - \tilde{\boldsymbol{\theta}}_b\|_2^2$, for any $\tilde{\boldsymbol{\theta}}_a, \tilde{\boldsymbol{\theta}}_b \in \mathbb{R}^L$. The minimum convexity constant is $m_{\hat{R}} := \min_i m_{\hat{R}_i}$. The gradients of the local cost functions are Lipschitz continuous with constants $M_{\hat{R}_i} > 0, \forall\ i$. That is, $\|\nabla \hat{R}_i(\tilde{\boldsymbol{\theta}}_a) - \nabla \hat{R}_i(\tilde{\boldsymbol{\theta}}_b)\|_2 \leq M_{\hat{R}_i} \|\tilde{\boldsymbol{\theta}}_a - \tilde{\boldsymbol{\theta}}_b\|_2$ for any agent i given any $\tilde{\boldsymbol{\theta}}_a, \tilde{\boldsymbol{\theta}}_b \in \mathbb{R}^L$. The maximum Lipschitz constant is $M_{\hat{R}} := \max_i M_{\hat{R}_i}$.*

Assumption 3 *The number of training samples of different agents is of the same order of magnitude, that is, $\frac{\max_i T_i - \min_i T_i}{\min_i T_i} < 10, \forall i \in \mathcal{N}$.*

Assumption 4 *There exists $f_{\mathcal{H}} \in \mathcal{H}$, such that for all estimators $f \in \mathcal{H}$, $\mathcal{E}(f_{\mathcal{H}}) \leq \mathcal{E}(f)$, where $\mathcal{E}(f) := \mathbb{E}_p\left[\ell(f(\mathbf{x}), y)\right]$ is the expected risk to measure the generalization ability of the estimator f.*

Assumption 1 and 2 are standard for decentralized optimization over decentralized networks (Shi et al., 2014), Assumption 4 is standard in generalization performance analysis of kernel learning (Li et al., 2018), and Assumption 3 is enforced to exclude the case of extremely unbalanced data distributed over the network.

4.1 Linear convergence of DKLA and COKE

We first establish that DKLA enables agents in the decentralized network to reach consensus on the prediction function at a linear rate. We then show that when the censoring function is properly chosen and the penalty parameter satisfies certain conditions, COKE also guarantees that the individually learned functional on the same sample linearly converges to the optimal solution.

Theorem 1 ***[Linear convergence of DKLA]*** *Initialize the dual variables as $\boldsymbol{\gamma}_i^0 = \mathbf{0}$, $\forall i$, with Assumptions 1 - 3, the learned functional at each agent through DKLA is R-linearly convergent to the optimal functional $\hat{f}_{\boldsymbol{\theta}^*}(\mathbf{x}) := (\boldsymbol{\theta}^*)^\top \boldsymbol{\phi}_L(\mathbf{x})$ for any $\mathbf{x} \in \mathcal{X}$, where $\boldsymbol{\theta}^*$ denotes the optimal solution to* (16) *obtained in the centralized case. That is,*

$$\lim_{k \to \infty} \hat{f}_{\boldsymbol{\theta}_i^k}(\mathbf{x}) = \hat{f}_{\boldsymbol{\theta}^*}(\mathbf{x}), \forall i. \tag{22}$$

Proof. See Appendix A. ■

Theorem 2 ***[Linear convergence of COKE]*** *Initialize the dual variables as $\boldsymbol{\gamma}_i^0 = \mathbf{0}$, $\forall i$, set the censoring thresholds to be $h(k) = v\mu^k$, with $v > 0$ and $\mu \in (0, 1)$, and choose the penalty parameter ρ such that*

$$0 < \rho < \min\left\{ \frac{4m_{\hat{R}}}{\eta_1}, \frac{(\nu-1)\tilde{\sigma}_{\min}^2(\mathbf{S}_-)}{\nu\eta_3\tilde{\sigma}_{\max}^2(\mathbf{S}_+)}, \left(\frac{\eta_1}{4} + \frac{\eta_2\tilde{\sigma}_{\max}^2(\mathbf{S}_+)}{8}\right)^{-1} \left(m_{\hat{R}} - \frac{\eta_3\nu M_{\hat{R}}^2}{\tilde{\sigma}_{\min}^2(\mathbf{S}_-)}\right) \right\}, \tag{23}$$

where $\eta_1 > 0$, $\eta_2 > 0$, $\eta_3 > 0$ and $\nu > 1$ are arbitrary constants, $m_{\hat{R}}$ and $M_{\hat{R}}$ are the minimum strong convexity constant of the local cost functions and the maximum Lipschitz constant of the local gradients, respectively. $\tilde{\sigma}_{\max}(\mathbf{S}_+)$ and $\tilde{\sigma}_{\min}(\mathbf{S}_-)$ are the maximum singular value of the unsigned incidence matrix $\mathbf{S}_+$ and the minimum non-zero singular value of the signed incidence matrix $\mathbf{S}_-$ of the network, respectively. Then, with Assumptions 1

- 3, the learned functional at each agent through COKE is R-linearly convergent to the optimal one $\hat{f}_{\boldsymbol{\theta}^*}(\mathbf{x}) := (\boldsymbol{\theta}^*)^\top \boldsymbol{\phi}_L(\mathbf{x})$ *for any* $\mathbf{x} \in \mathcal{X}$*, where* $\boldsymbol{\theta}^*$ *denotes the optimal solution to* (16) *obtained in the centralized case. That is,*

$$\lim_{k\to\infty} \hat{f}_{\boldsymbol{\theta}_i^k}(\mathbf{x}) = \hat{f}_{\boldsymbol{\theta}^*}(\mathbf{x}), \forall i. \tag{24}$$

Proof. See Appendix A. ■

Remark 1. It should be noted that the kernel transformation with RF mapping is essential in enabling convex consensus formulation with convergence guarantee. For example, in a regular optimization problem with a local cost function $(y - f(\mathbf{x}))^2$, even if it is quadratic, the nonlinear function $f(\mathbf{x})$ inside destroys the convexity. In contrast, with RF mapping, $f(\mathbf{x})$ of any form is expressed as a linear function of $\boldsymbol{\theta}$, and hence the local cost function is guaranteed to be convex. For decentralized kernel learning, many widely-adopted loss functions result in (strongly) convex local objective functions in the RF space, such as the quadratic loss in a regression problem and logistic loss in a classification problem.

Remark 2. For Theorem 2, notice that choosing larger v and μ in the design of the censoring thresholds in COKE leads to less communication per iteration at the expense of possible performance degradation, whereas smaller v and μ may not contribute much to communication saving. However, it is challenging to acquire an explicit tradeoff between communication cost and steady-state accuracy, since the designed censoring thresholds do not have an explicit relationship with the update of the model parameter.

The above theorems establish the exact convergence of the functional learned in the multi-agent system for the decentralized kernel regression problem via DKLA and COKE. Different from previous works (Koppel et al., 2018; Shin et al., 2018), our analytic results are obtained by converting the non-parametric data-dependent learning model into a parametric data-independent model in the RF space and solved under the consensus optimization framework. In this way, we not only reduce the computational complexity of the standard kernel methods and make the RF-based kernel methods scalable to large-size datasets, but also protect data privacy since no raw data exchange among agents is required and the RF mapping is not one-to-one mapping. RF mapping is crucial in our algorithms, with which we are able to show the linear convergence of the functional by showing the linear convergence of the iteratively updated decision variables in the RF space; see Appendix A for more details.

4.2 Generalization property of COKE

The ultimate goal of decentralized learning is to find a function that generalizes well for the ensemble of all data from all agents. To evaluate the generalization property of the predictive function learned by COKE, we are then interested in bounding the difference between the expected risk of the predictive function learned by COKE at the k-th iteration, defined as $\mathcal{E}(\hat{f}^k) := \sum_{i=1}^N \mathcal{E}_i(\hat{f}_{\boldsymbol{\theta}_i^k}) := \sum_{i=1}^N \mathbb{E}_p[(y - (\boldsymbol{\theta}_i^k)^\top \boldsymbol{\phi}_L(\mathbf{x}))^2]$, and the expected risk $\mathcal{E}(f_{\mathcal{H}})$ in the RKHS. This is different from bounding the approximation error between the kernel κ and the approximated $\hat{\kappa}_L$ by L random features as in the literature (Rahimi and Recht, 2008; Sutherland and Schneider, 2015; Sriperumbudur and Szabó, 2015). As DKLA is a special case of COKE, the generalization performance of COKE can be extended to DKLA straightforwardly.

To illustrate our finding, we focus on the kernel regression problem whose loss function is least squares, i.e., $\ell(y, f(\mathbf{x})) = (y - f(\mathbf{x}))^2$. With RF mapping, the objective function (16) of the regression problem can be formulated as

$$\hat{R}(\boldsymbol{\theta}) = \sum_{i=1}^{N} \hat{R}_i(\boldsymbol{\theta}) = \sum_{i=1}^{N} \left(\frac{1}{T_i} \|\mathbf{y}_i - (\boldsymbol{\Phi}_L^i)^\top \boldsymbol{\theta}\|_2^2 + \frac{\lambda}{N} \|\boldsymbol{\theta}\|_2^2 \right), \tag{25}$$

where $\mathbf{y}_i = [y_{i,1}, \ldots, y_{i,T_i}]^\top \in \mathbb{R}^{T_i \times 1}$, $\boldsymbol{\Phi}_L^i = [\boldsymbol{\phi}_L(\mathbf{x}_{i,1}), \ldots, \boldsymbol{\phi}_L(\mathbf{x}_{i,T_i})] \in \mathbb{R}^{L \times T_i}$, and $\boldsymbol{\phi}_L(\mathbf{x}_{i,t})$ is the data mapped to the RF space.

The optimal solution of (25) is given in closed form by

$$\boldsymbol{\theta}^* = (\tilde{\boldsymbol{\Phi}}^\top \tilde{\boldsymbol{\Phi}} + \lambda \mathbf{I})^{-1} \tilde{\boldsymbol{\Phi}}^\top \tilde{\mathbf{y}}, \tag{26}$$

where $\tilde{\boldsymbol{\Phi}} = [\tilde{\boldsymbol{\Phi}}_L^1, \ldots, \tilde{\boldsymbol{\Phi}}_L^N]^\top \in \mathbb{R}^{T \times L}$ with $\tilde{\boldsymbol{\Phi}}_L^i = \frac{1}{\sqrt{T_i}} \boldsymbol{\Phi}_L^i, \forall i \in \mathcal{N}$, and $\tilde{\mathbf{y}} = [\tilde{\mathbf{y}}_1; \ldots; \tilde{\mathbf{y}}_N] \in \mathbb{R}^{T \times 1}$ with $\tilde{\mathbf{y}}_i = \frac{1}{\sqrt{T_i}} \mathbf{y}_i, \forall i \in \mathcal{N}$. The optimal prediction model is then expressed by

$$\hat{f}_{\boldsymbol{\theta}^*}(\mathbf{x}) = (\boldsymbol{\theta}^*)^\top \boldsymbol{\phi}_L(\mathbf{x}). \tag{27}$$

In the following theorem, we give a general result of the generalization performance of the predictive function learned by COKE for the kernel regression problem, which is built on the linear convergence result given in Theorem 3 and taking into account of the number of random features adopted.

Theorem 3 *Let $\lambda_{\mathbf{K}}$ be the largest eigenvalue of the kernel matrix $\mathbf{K}$ constructed by all data, $\mathbf{X}_T$, and choose the regularization parameter $\lambda < \lambda_{\mathbf{K}}/T$ so as to control overfitting. Under the Assumptions 1 - 4, with the censoring function and other parameters given in Theorem 2, for all $\delta_p \in (0,1)$ and $\|f\|_{\mathcal{H}} \leq 1$, if the number of random features L satisfies*

$$L \geq \frac{1}{\lambda} \left(\frac{1}{\epsilon^2} + \frac{2}{3\epsilon} \right) \log \frac{16 d_{\mathbf{K}}^{\lambda}}{\delta_p},$$

then with probability at least $1 - \delta_p$, the excess risk of $\mathcal{E}(\hat{f}^k)$ obtained by Algorithm 2 converges to an upper bound, i.e.,

$$\lim_{k \to \infty} \left(\mathcal{E}(\hat{f}^k) - \mathcal{E}(f_{\mathcal{H}}) \right) \leq 3\lambda + O\left(\frac{1}{\sqrt{T}} \right), \tag{28}$$

where $\epsilon \in (0,1)$, and $d_{\mathbf{K}}^{\lambda} := \mathrm{Tr}(\mathbf{K}(\mathbf{K} + \lambda T \mathbf{I})^{-1})$ is the number of effective degrees of freedom that is known to be an indicator of the number of independent parameters in a learning problem (Avron et al., 2017).

Proof. See Appendix B. ■

Theorem 3 states the tradeoff between the computational efficiency and the statistical efficiency through the regularization parameter λ, effective dimension $d_{\mathbf{K}}^{\lambda}$, and the number of random features adopted. We can see that to bound the excess risk with a higher probability, we need more random features, which results in a higher computational complexity. The regularization parameter is usually determined by the number of training data and one common practice is to set $\lambda = O(1/\sqrt{T})$ for the regression problem (Caponnetto and

De Vito, 2007). Therefore, with $O(\sqrt{T}\log d_{\mathbf{K}}^{\lambda})$ features, COKE achieves a learning risk of $O(1/\sqrt{T})$ at a linear rate. We also notice that different sampling strategies affect the number of random features required to achieve a given generalization error. For example, importance sampling is studied for the centralized kernel learning in RF space in (Li et al., 2018). Interested readers are referred to (Li et al., 2018) and references therein.

5. Experiments

This section evaluates the performance of our COKE algorithm in regression tasks using both synthetic and real-world datasets. Since we consider the case that data are only locally available and cannot be shared among agents, the following RF-based methods are used to benchmark our COKE algorithm.

CTA. This is a form of diffusion-based technique where all agents first construct their RF-mapped data $\{\boldsymbol{\phi}_L(\mathbf{x}_{i,t})\}_{t=1}^{T_i}$, for $t = 1, \ldots, T_i, \forall i$, using the same random features $\{\boldsymbol{\omega}_l\}_{l=1}^{L}$ as DKLA and COKE. Then at each iteration k, each agent i first combines information from its neighbors, i.e., $\boldsymbol{\theta_n}, \forall n \in \mathcal{N}_i$ with its own parameter $\boldsymbol{\theta}_i$ by aggregation. Then, it updates its own parameter $\boldsymbol{\theta}_i$ using the gradient descent method with the aggregated information (Sayed, 2014). The cost function for agent i is given in (15). Note that this method has not been formally proposed in existing works for RF-based decentralized kernel learning with batch-form data, but we introduced it here only for comparison purpose. An online version that deals with streaming data is available in (Bouboulis et al., 2018). The batch version of CTA introduced here is expected to converge faster than the online version.

DKLA. Algorithm 1 proposed in Section 3.2 where ADMM is applied and the communication among agents happen at every iteration without being censored.

The performance of all algorithms is evaluated using both synthetic and real-world datasets, where the entries of data samples are normalized to lie in $[0, 1]$ and each agent uses 70% of its data for training and the rest for testing. The learning performance at each iteration is evaluated using mean-squared-error (MSE) given by $\text{MSE}(k) = \frac{1}{T}\sum_{i=1}^{N}\sum_{t=1}^{T_i}(y_{i,t} - (\boldsymbol{\theta}_i^k)^\top \boldsymbol{\phi}_L(\mathbf{x}_{i,t}))^2$. The decision variable $\boldsymbol{\theta}_i$ for CTA is initialized as $\boldsymbol{\theta}_i^0 = \mathbf{0}, \forall i$ as that in DKLA and COKE. For COKE, it should be noted that the design of the censoring function is crucial. For the censoring thresholds adopted in Theorem 2, choosing larger v and μ to design the censor thresholds leads to less communication per iteration but may result in performance degradation. For all simulations, the kernel bandwidth is fine-tuned for each dataset individually via cross-validation. The parameters of the censoring function are tuned to achieve the best learning performance at nearly no performance loss.

5.1 Synthetic dataset

In this setup, the connected graph is randomly generated with $N = 20$ nodes and 95 edges. The probability of attachment per node equals to 0.3, that is, any pair of two nodes are connected with a probability of 0.3. Each agent has $T_i \in (4000, 6000)$ data pairs generated following the model $y_{i,t} = \sum_{m=1}^{50} b_m \kappa(\mathbf{c}_m, \mathbf{x}_{i,t}) + e_{i,t}$, where b_m are uniformly drawn from $[0, 1]$, $\mathbf{c}_m \sim \mathbf{N}(\mathbf{0}, \mathbf{I}_5)$, $\mathbf{x}_{i,t} \sim \mathbf{N}(\mathbf{0}, \mathbf{I}_5)$, and $e_{i,t} \sim \mathbf{N}(0, 0.1)$. The kernel κ in the model is Gaussian with a bandwidth $\sigma = 5$.

5.2 Real datasets

To further evaluate our algorithms, the following popular real-world datasets from UCI machine learning repository are chosen (Asuncion and Newman, 2007).

Tom's hardware. This dataset contains $T = 11000$ samples with $\mathbf{x}_t \in \mathbb{R}^{96}$ whose features include the number of created discussions and authors interacting on a topic and $y_t \in \mathbb{R}$ representing the average number of displays to a visitor about that topic (Kawala et al., 2013).

Twitter. This dataset consists of $T = 13800$ samples with $\mathbf{x}_t \in \mathbb{R}^{77}$ being a feature vector reflecting the number of new interactive authors and the length of discussions on a given topic, etc., and $y_t \in \mathbb{R}$ representing the average number of active discussion on a certain topic. The learning task is to predict the popularity of these topics. We also include a larger Twitter dataset for testing which has $T = 98704$ samples (Kawala et al., 2013).

Energy. This dataset contains $T = 19735$ samples with $\mathbf{x}_t \in \mathbb{R}^{28}$ describing the humidity and temperature in different areas of the houses, pressure, wind speed and viability outside, while y_t denotes the total energy consumption in the house (Candanedo et al., 2017).

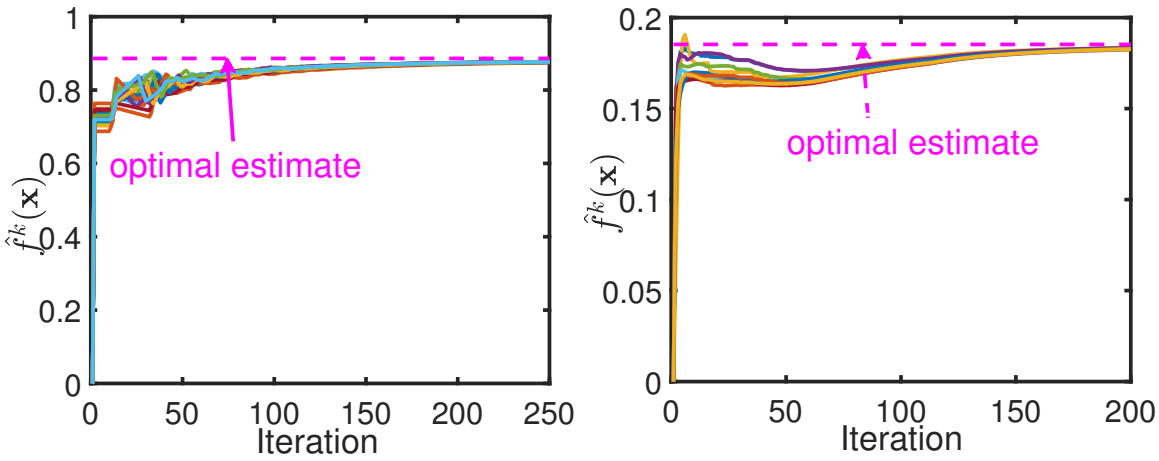
Air quality. This dataset contains dataset collects $T = 9358$ samples measured by a gas multi-sensor device in an Italian city, where $\mathbf{x}_t \in \mathbb{R}^{13}$ represents the hourly concentration of CO, NOx, NO2, etc, while y_t denotes the concentration of polluting chemicals in the air (De Vito et al., 2008).

5.3 Parameter setting and performance analysis

For synthetic data, we adopt a Gaussian kernel with a bandwidth $\sigma = 1$ for training and use $L = 100$ random features for kernel approximation. Note that the chosen σ differs from that of the actual data model. The censoring thresholds are $h(k) = 0.95^k$, the regularization parameter λ and stepsize ρ of DKLA and COKE are set to be 5×10^{-5} and 10^{-2}, respectively. The stepsize of CTA is set to be $\eta = 0.99$, which is tuned to achieve the same level of learning performance as COKE and DKLA at its fastest speed.

To show the performance of all algorithms on real datasets concisely and comprehensively, we present the experimental results on the Twitter dataset with $T = 13800$ samples by figures and record the experimental results on the remaining datasets by tables. For the Twitter dataset with $T = 13800$ samples, we randomly split it into 10 mini-batches each with $T_i \in (1200, 1400)$ data pairs while $\sum_{i=1}^{10} T_i = T$. The 10 mini-batches are distributed to 10 agents connected by a random network with 28 edges. We use 100 random features to approximate a Gaussian kernel with a bandwidth $\sigma = 1$ during the training process. The parameters λ and ρ are set to be 10^{-3} and 10^{-2}, respectively. The censoring thresholds are $h(k) = 0.97^k$. The stepsize of CTA is set to be $\eta = 0.99$ to balance the learning performance and the convergence speed.

In Fig.1, we show that the individually learned functional at each agent via COKE reaches consensus to the optimal estimate for both synthetic and real datasets. In Fig. 2, we compare the MSE performance of COKE, DKLA, and CTA. Both figures show that COKE converges slower than DKLA due to the communications skipped by the censoring step. However, the learning performance of COKE eventually is the same as DKLA. For the diffusion-based CTA algorithm, it converges the slowest. In Fig. 3, we show the MSE performance versus the communication cost (in terms of the number of transmissions). As



(a) Synthetic data. (b) Twitter data.

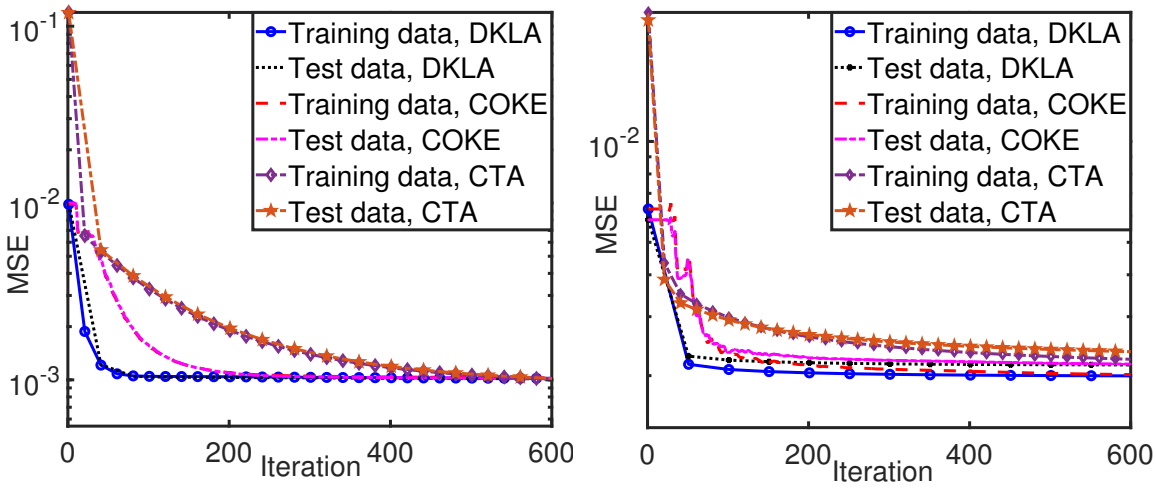
Figure 1: Functional convergence via COKE for synthetic data (Figure 1 (a)) and the real dataset (Figure 1 (b)). The learned functionals of all distributed agents converge to the optimal estimate where data are assumed to be centrally available.

	Training error (MSE (10^{-3})) / Commun. cost			Test error (MSE(10^{-3}))		
Iteration	CTA	DKLA	COKE	CTA	DKLA	COKE
$k = 50$	3.9/500	2.4/500	4.5/**13**	4.0	2.6	4.2
$k = 100$	3.3/1000	2.4/1000	2.6/**100**	3.4	2.6	2.8
$k = 200$	3.0/2000	2.3/2000	2.4/**298**	3.2	2.5	2.6
$k = 500$	2.7/5000	2.3/5000	**2.3/902**	2.9	2.5	**2.5**
$k = 1000$	2.5/10000	2.2/10000	**2.2/4648**	2.7	2.5	**2.5**
$k = 1500$	2.5/15000	2.2/15000	**2.2/9648**	2.7	2.5	**2.5**
$k = 2000$	2.4/20000	2.2/20000	**2.2/14648**	2.6	2.5	**2.5**

Table 1: MSE performance on the Twitter dataset (large), $\sigma = 1$, $L = 100$, $\lambda = 10^{-3}$, stepsize $\eta = 0.99$ for CTA, stepsize $\rho = 10^{-2}$ for DKLA and COKE, censoring thresholds $h(k) = 0.5 \times 0.98^k$. DKLA and COKE achieve better MSE performance than CTA while COKE requires the least communication resource than DKLA.

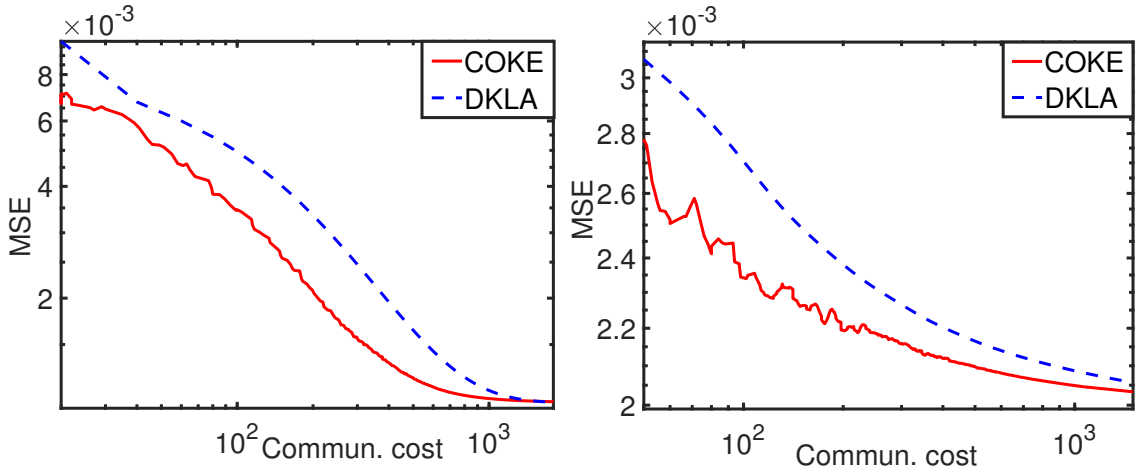
CTA converges the slowest and communicates all the time, its communication cost is much higher than that of DKLA, and thus we do not include it in Fig. 3 but rather focus on the communication-saving of COKE over DKLA. We can see that to achieve the same level of learning performance, COKE requires much less communication cost than DKLA. Both the synthetic data and the real dataset show communication saving of around 50% in Fig. 3 for a given learning accuracy, which corroborate the communication-efficiency of COKE.

The performance of all three algorithms on the rest four datasets is listed in Table 1 - 6. All results show that COKE saves much communication (almost 50%) within a negligible



(a) Synthetic data. (b) Twitter data.

Figure 2: MSE performance for synthetic data (Figure 2 (a)) and the real dataset (Figure 2 (b)). ADMM-based algorithms (COKE and DKLA) converge faster than the diffusion-based algorithm (CTA) for both synthetic data (Figure 2 (a)) and the real dataset (Figure 2 (b)). Furthermore, DKLA and COKE achieve better learning performance than CTA in terms of MSE on the real dataset.



(a) Synthetic data. (b) Twitter data.

Figure 3: MSE performance versus communication cost for synthetic data (Figure 3 (a)) and the real dataset (Figure 3 (b)). Compared with DKLA, COKE achieves around 50% communication saving on the same level of MSE performance for both synthetic data (Figure 3 (a)) and the real dataset (Figure 3 (b)).

	Training error (MSE (10^{-4})) / Commun. cost			Test error (MSE(10^{-4}))		
Iteration	CTA	DKLA	COKE	CTA	DKLA	COKE
$k = 50$	20.02/500	10.01/500	17.40/**10**	20.16	11.20	18.82
$k = 100$	16.6/1000	9.91/1000	10.67/**112**	17.09	11.10	11.86
$k = 200$	13.68/2000	9.90/2000	9.97/**331**	14.58	11.10	11.15
$k = 500$	11.19/5000	9.90/5000	**9.90/1114**	12.35	11.10	**11.10**
$k = 1000$	10.27/10000	9.90/10000	**9.90/5600**	11.47	11.10	**11.10**
$k = 1500$	10.01/15000	9.90/15000	**9.90/10600**	11.22	11.10	**11.10**
$k = 2000$	9.92/20000	9.90/20000	**9.90/15600**	11.13	11.10	**11.10**

Table 2: MSE performance on the Tom's hardware dataset, $\sigma = 1$, $L = 100$, $\lambda = 10^{-2}$, stepsize $\eta = 0.99$ for CTA, stepsize $\rho = 10^{-2}$ for DKLA and COKE, censoring thresholds $h(k) = 0.5 \times 0.95^k$. DKLA and COKE achieve better MSE performance than CTA while COKE requires the least communication resource than DKLA.

Twitter (large)				Tom's hardware			
MSE (10^{-3})	Commun. cost			MSE (10^{-4})	Commun. cost		
	CTA	DKLA	COKE		CTA	DKLA	COKE
5	360	20	**10**	18	680	20	**3**
4	480	30	**10**	16	1020	30	**22**
3	1860	60	**48**	14	1610	60	**28**
2.8	3250	100	**55**	12	2880	110	**51**
2.6	6120	180	**100**	10	7950	250	**128**
2.3	-	1080	**577**	9.95	17620	640	**361**
2.2	-	5660	**4428**	9.90	-	1550	**984**

Table 3: MSE performance (training error) versus communication cost on the Twitter dataset (large) and the Tom's hardware dataset. For both datasets, COKE saves around 50% communication resource than DKLA to achieve the same level of learning performance.

learning gap from DKLA, and both DKLA and COKE require much less communication resources than CTA. For example, the number of transmissions required to reach a training estimation error of 2.3×10^{-3} on Twitter dataset by COKE is 577, which is only 53% of that required by DKLA to reach the same level of learning performance. For Tom's hardware dataset, COKE requires 361 total transmissions to reach a learning error of 9.95×10^{-4}, which is 56.4% of DKLA and 0.02% of CTA. Note that much of the censoring occurs at the beginning update iterations. While at the later stage, COKE nearly transmits all parameters at every iteration since the censoring thresholds are smaller than the difference between two consecutive updates.

	Training error (MSE (10^{-3})) / Commun. cost			Test error (MSE(10^{-3}))		
Iteration	CTA	DKLA	COKE	CTA	DKLA	COKE
$k = 50$	25.65/500	22.52/500	25.22/**0**	26.45	22.97	26.02
$k = 100$	24.88/1000	22.12/1000	23.65/**57**	25.57	22.50	24.2
$k = 200$	24.17/2000	21.81/2000	22.57/**254**	24.77	22.15	23.02
$k = 500$	23.40/5000	21.55/5000	21.88/**987**	23.92	21.86	22.22
$k = 1000$	22.84/10000	21.48/10000	21.51/**5752**	23.31	21.79	21.82
$k = 1500$	22.54/15000	21.47/15000	**21.47/10752**	22.97	21.78	**21.78**
$k = 2000$	22.35/20000	21.47/20000	**21.47/15752**	22.75	21.78	**21.78**

Table 4: MSE performance on the Energy dataset, $\sigma = 0.1$, $L = 100$, $\lambda = 10^{-3}$, stepsize $\eta = 0.99$ for CTA, $\rho = 10^{-2}$ for DKLA and COKE, censoring thresholds $h(k) = 0.5 \times 0.98^k$. DKLA and COKE achieve better MSE performance than CTA while COKE requires the least communication resource.

	Training error (MSE (10^{-3})) / Commun. cost			Test error (MSE(10^{-3}))		
Iteration	CTA	DKLA	COKE	CTA	DKLA	COKE
$k = 50$	6.4/500	1.8/500	3.7/**72**	6.7	2.1	4.0
$k = 100$	4.5/1000	1.6/1000	2.2/**172**	4.8	1.8	2.5
$k = 200$	3.2/2000	1.4/2000	1.7/**384**	3.5	1.7	2.0
$k = 500$	2.2/5000	1.3/5000	**1.3/2263**	2.5	1.6	**1.6**
$k = 1000$	1.7/10000	1.2/10000	**1.2/7263**	2.0	1.6	**1.6**
$k = 1500$	1.6/15000	1.2/15000	**1.2/12263**	1.8	1.6	**1.6**
$k = 2000$	1.5/20000	1.2/20000	**1.2/17263**	1.8	1.6	**1.6**

Table 5: MSE performance on the Air quality dataset, $\sigma = 0.1$, $L = 200$, $\lambda = 10^{-5}$, stepsize $\eta = 0.99$ for CTA, $\rho = 10^{-2}$ for DKLA and COKE, censoring thresholds $h(k) = 0.9 \times 0.97^k$. DKLA and COKE achieve better MSE performance than CTA while COKE requires the least communication resource than DKLA.

Energy				Air quality			
MSE (10^{-3})	Commun. cost			MSE (10^{-3})	Commun. cost		
	CTA	DKLA	COKE		CTA	DKLA	COKE
25	860	20	**11**	5.0	810	60	**49**
24	2290	70	**48**	3.0	2290	180	**81**
23.5	4160	140	**76**	2.0	6010	360	**211**
23	7690	250	**134**	1.8	8160	490	**285**
22.5	14750	480	**258**	1.6	12300	750	**424**
22	-	1160	**652**	1.5	16190	1010	**586**
21.5	-	4950	**4062**	1.2	-	5990	**5383**

Table 6: MSE performance (training error) versus communication cost on the Energy dataset and the Air quality dataset. For both datasets, COKE saves around 45%-55% communication resource than DKLA to achieve the same level of learning performance.

6. Concluding Remarks

This paper studies the decentralized kernel learning problem under privacy concern and communication constraints for multi-agent systems. Leveraging the random feature mapping, we convert the non-parametric kernel learning problem into a parametric one in the RF space and solve it under the consensus optimization framework by the alternating direction method of multipliers. A censoring strategy is applied to conserve communication resources. Through both theoretical analysis and simulations, we establish that the proposed algorithms not only achieve linear convergence rate but also exhibit effective generalization performance. Thanks to the fixed-size parametric learning model, the proposed algorithms circumvent the curse of dimensionality problem and do not involve raw data exchange among agents. Hence, they can be applied in distributed learning that involve big-data and offer some level of data privacy protection. To cope with dynamic environments and enhance the learning performance, future work will be devoted to decentralized online kernel learning and multi-kernel learning.

Acknowledgments

We would like to acknowledge support for this project from the National Science Foundation (NSF grants #1741338 and #1939553).

Appendix A. Proof of Theorem 1 and Theorem 2

Proof. As discussed in Section 3.2, solving the decentralized kernel learning problem in the RF space (17) is equivalent to solving the problem (16). From (14), it is evident that the convergence of the optimal functional f in (16) hinges on the convergence of the decision variables $\boldsymbol{\theta}$ in the RF space. Since in the RF space, the decision variables are data-independent, the convergence proof of DKLA boils down to proving the convergence of a convex optimization problem solved by ADMM. However, the convergence proof of COKE is nontrivial because of the error caused by the outdated information introduced by the communication censoring strategy. Our proof for both theorems consists of two steps. The first step is to show linear convergence of decision parameters $\boldsymbol{\theta}$ for DKLA via Theorem 4 and for COKE via Theorem 5 below, which are derived straightforwardly from (Shi et al., 2014) and (Liu et al., 2019), respectively. The second step is to show how the convergence of $\boldsymbol{\theta}$ translates to the convergence of the learned functional, which are the same for both algorithms.

Compared to (Shi et al., 2014) and (Liu et al., 2019) that deal with general optimization problems for parametric learning, this work focuses on specific decentralized kernel learning problem which is more challenging in both solution development and theoretical analysis. By leveraging the RF mapping technique, we successfully develop the DKLA algorithm and the COKE algorithm. Noticeably, a direct application of ADMM as in (Shi et al., 2014) on decentralized kernel learning is infeasible without raw data exchanges. Moreover, we analyze the convergence of the nonlinear functional to be learned and the generalization performance of kernel learning in the decentralized setting. The analysis is built on the work of (Shi et al., 2014) and (Liu et al., 2019) but goes further, and it is only attainable because of the adoption of the RF mapping.

For both algorithms, the linear convergence of decision variables in the RF space is based on matrix reformulation of (17). Define $\boldsymbol{\Theta}^* := [\boldsymbol{\theta}^*, \boldsymbol{\theta}^*, \ldots, \boldsymbol{\theta}^*]^\top \in \mathbb{R}^{N\times L}$ and $\boldsymbol{\Theta}^* := [\boldsymbol{\vartheta}^*, \boldsymbol{\vartheta}^*, \ldots, \boldsymbol{\vartheta}^*]^\top \in \mathbb{R}^{N\times L}$ be the optimal primal variables, and $\boldsymbol{\beta}^*$ be the optimal dual variable. Then, for DKLA, Theorem 4 states that $\{\boldsymbol{\Theta}^k\}$ ($\boldsymbol{\Theta}^k := [\boldsymbol{\theta}_1^k, \boldsymbol{\theta}_2^k, \ldots, \boldsymbol{\theta}_N^k]^\top \in \mathbb{R}^{N\times L}$) is R-linear convergent to the optimal $\boldsymbol{\Theta}^*$. For detailed proof, see (Shi et al., 2014).

Theorem 4 ***[Linear convergence of decision variables in DKLA]*** *For the optimization problem* (16)*, initialize the dual variables as* $\boldsymbol{\gamma}_i^0 = \mathbf{0}$, $\forall i$*, with Assumptions 1 - 2, then* $\{\boldsymbol{\Theta}^k\}$ *is R-linearly convergent to the optimal* $\boldsymbol{\Theta}^*$ *when* k *goes to infinity following from*

$$\|\boldsymbol{\Theta}^k - \boldsymbol{\Theta}^*\|_F^2 \leq \frac{1}{m_{\hat{R}}}\left[\rho\|\boldsymbol{\Theta}^{k-1} - \boldsymbol{\Theta}^*\|_F^2 + \frac{1}{\rho}\|\boldsymbol{\beta}^{k-1} - \boldsymbol{\beta}^*\|_F^2\right], \tag{29}$$

where $\{(\boldsymbol{\Theta}^k, \boldsymbol{\beta}^k)\}$ *is Q-linearly convergent to its optimal* $\{(\boldsymbol{\Theta}^*, \boldsymbol{\beta}^*)\}$*:*

$$\rho\|\boldsymbol{\Theta}^k - \boldsymbol{\Theta}^*\|_F^2 + \frac{1}{\rho}\|\boldsymbol{\beta}^k - \boldsymbol{\beta}^*\|_F^2 \leq \frac{1}{1+\delta_d}\left[\rho\|\boldsymbol{\Theta}^{k-1} - \boldsymbol{\Theta}^*\|_F^2 + \frac{1}{\rho}\|\boldsymbol{\beta}^{k-1} - \boldsymbol{\beta}^*\|_F^2\right] \tag{30}$$

with

$$\delta_d = \min\left\{\frac{(\nu-1)\tilde{\sigma}_{\min}^2(\mathbf{S}_-)}{\nu\tilde{\sigma}_{\max}^2(\mathbf{S}_+)}, \frac{m_{\hat{R}}}{\frac{\rho}{4}\tilde{\sigma}_{\max}^2(\mathbf{S}_+) + \frac{\nu}{\rho}M_{\hat{R}}^2\tilde{\sigma}_{\min}^2(\mathbf{S}_-)}\right\},$$

where $\nu > 1$ *is an arbitrary constant,* $\tilde{\sigma}_{\max}(\mathbf{S}_+)$ *is the maximum singular value of the unsigned incidence matrix* $\mathbf{S}_+$ *of the network, and* $\tilde{\sigma}_{\min}^2(\mathbf{S}_-)$ *is the minimum non-zero singular value of the signed incidence matrix* $\mathbf{S}_-$ *of the network,* $m_{\hat{R}}$ *and* $M_{\hat{R}}$ *are the minimum*

strong convexity constant of the local cost functions and the maximum Lipschitz constant of the local gradients, respectively. The Q-linear convergence rate of $\{(\boldsymbol{\Theta}^k, \boldsymbol{\beta}^k)\}$ *to* $\{(\boldsymbol{\Theta}^*, \boldsymbol{\beta}^*)\}$ *satisfies*

$$r_c \leq \sqrt{\frac{1}{1+\delta_d}}. \tag{31}$$

To achieve linear convergence of decision variables in COKE, choosing appropriate censoring functions is crucial. Moreover, the penalty parameter ρ also needs to satisfy certain conditions, see Theorem 5 for details (Liu et al., 2019).

Theorem 5 ***[Linear convergence of decision variables in COKE]*** *For the optimization problem* (16) *with strongly convex local cost functions whose gradients are Lipschitz continuous, initialize the dual variables as* $\boldsymbol{\gamma}_i^0 = \mathbf{0}$, $\forall i$, *set the censoring thresholds to be* $h(k) = v\mu^k$, *with* $v > 0$ *and* $\mu \in (0,1)$, *and choose the penalty parameter* ρ *such that*

$$0 < \rho < \min\left\{\frac{4m_{\hat{R}}}{\eta_1}, \frac{(\nu-1)\tilde{\sigma}^2_{\min}(\mathbf{S}_-)}{\nu\eta_3\tilde{\sigma}^2_{\max}(\mathbf{S}_+)}, \left(\frac{\eta_1}{4} + \frac{\eta_2\tilde{\sigma}^2_{\max}(\mathbf{S}_+)}{8}\right)^{-1}\left(m_{\hat{R}} - \frac{\eta_3\nu M^2_{\hat{R}}}{\tilde{\sigma}^2_{\min}(\mathbf{S}_-)}\right)\right\}, \tag{32}$$

where $\eta_1 > 0$, $\eta_2 > 0$, $\eta_3 > 0$ *and* $\nu > 1$ *are arbitrary constants,* $m_{\hat{R}}$ *and* $M_{\hat{R}}$ *are the minimum strong convexity constant of the local cost functions and the maximum Lipschitz constant of the local gradients, respectively.* $\tilde{\sigma}_{\max}(\mathbf{S}_+)$ *and* $\tilde{\sigma}^2_{\min}(\mathbf{S}_-)$ *are the maximum singular value of the unsigned incidence matrix* $\mathbf{S}_+$ *and the minimum non-zero singular value of the signed incidence matrix* $\mathbf{S}_-$ *of the network, respectively. Then,* $\{\boldsymbol{\Theta}^k\}$ *is R-linearly convergent to the optimal* $\boldsymbol{\Theta}^*$ *when* k *goes to infinity following from.*

Remark 3. For the kernel ridge regression problem (25), the minimum strong convexity constant of the local cost functions and the maximum Lipschitz constant of the local gradients are $m_{\hat{R}} := \min_i \tilde{\sigma}^2_{\min}(\frac{1}{T_i}\boldsymbol{\Phi}^i_L(\boldsymbol{\Phi}^i_L)^\top + \frac{2\lambda}{N}\mathbf{I})$ and $M_{\hat{R}} := \max_i \tilde{\sigma}^2_{\max}(\frac{1}{T_i}\boldsymbol{\Phi}^i_L(\boldsymbol{\Phi}^i_L)^\top + \frac{2\lambda}{N}\mathbf{I})$, respectively.

With the convergence of decision variables in the RF space given in Theorem 4 and Theorem 5, the second step is to prove the linear convergence of the learned functional $\hat{f}_{\boldsymbol{\theta}_i^k}(\mathbf{x})$ to the optimal $\hat{f}_{\boldsymbol{\theta}^*}(\mathbf{x})$, which is straightforward for both algorithms.

Denote $\hat{\boldsymbol{f}}_{\boldsymbol{\Theta}^k}(\mathbf{x}) = [\hat{f}_{\boldsymbol{\theta}_1^k}(\mathbf{x}), \ldots, \hat{f}_{\boldsymbol{\theta}_N^k}(\mathbf{x})]^\top = \boldsymbol{\Theta}^k\boldsymbol{\phi}_L(\mathbf{x})$ and $\hat{\boldsymbol{f}}_{\boldsymbol{\Theta}^*}(\mathbf{x}) = [\hat{f}_{\boldsymbol{\theta}^*}(\mathbf{x}), \ldots, \hat{f}_{\boldsymbol{\theta}^*}(\mathbf{x})]^\top = \boldsymbol{\Theta}^*\boldsymbol{\phi}_L(\mathbf{x})$, then we have

$$\begin{aligned}\|\hat{\boldsymbol{f}}_{\boldsymbol{\Theta}^k}(\mathbf{x}) - \hat{\boldsymbol{f}}_{\boldsymbol{\Theta}^*}(\mathbf{x})\|_2 &= \|\boldsymbol{\Theta}^k\boldsymbol{\phi}_L(\mathbf{x}) - \boldsymbol{\Theta}^*\boldsymbol{\phi}_L(\mathbf{x})\|_2 \\ &\leq \|\boldsymbol{\Theta}^k - \boldsymbol{\Theta}^*\|_2\|\boldsymbol{\phi}_L(\mathbf{x})\|_2 \\ &\leq \|\boldsymbol{\Theta}^k - \boldsymbol{\Theta}^*\|_2,\end{aligned} \tag{33}$$

where the second inequality comes from the fact that $\|\boldsymbol{\phi}_L(\mathbf{x})\|_2 \leq 1$ with the adopted RF mapping.

For DKLA, we have

$$\|\hat{\boldsymbol{f}}_{\boldsymbol{\Theta}^k}(\mathbf{x}) - \hat{\boldsymbol{f}}_{\boldsymbol{\Theta}^*}(\mathbf{x})\|_2 \leq \|\boldsymbol{\Theta}^k - \boldsymbol{\Theta}^*\|_2 \leq \frac{1}{m_{\hat{R}}}\left[\rho\|\boldsymbol{\Theta}^{k-1} - \boldsymbol{\Theta}^*\|_F^2 + \frac{1}{\rho}\|\boldsymbol{\beta}^{k-1} - \boldsymbol{\beta}^*\|_F^2\right]. \tag{34}$$

Therefore, the Q-linear convergence of $\{\boldsymbol{\Theta}^k, \boldsymbol{\beta}^k\}$ to the optimal $(\boldsymbol{\Theta}^*, \boldsymbol{\beta}^*)$ translates to the R-linear convergence of $\{\hat{\boldsymbol{f}}_{\boldsymbol{\Theta}^k}(\mathbf{x})\}$. Similarly, the R-linear convergence of $\{\boldsymbol{\Theta}^k\}$ to the optimal

$\boldsymbol{\Theta}^*$ of COKE can be translated from the Q-linear convergence of $\{\boldsymbol{\Theta}^k, \boldsymbol{\beta}^k\}$ to the optimal $(\boldsymbol{\Theta}^*, \boldsymbol{\beta}^*)$, see Liu et al. (2019) for detailed proof.

It is then straightforward to see that the individually learned functionals converge to the optimal one when k goes to infinity, that is, for $i \in \mathcal{N}$,

$$\begin{aligned}\lim_{k\to\infty} |\hat{f}_{\boldsymbol{\theta}_i^k}(\mathbf{x}) - \hat{f}_{\boldsymbol{\theta}^*}(\mathbf{x})| &= \lim_{k\to\infty} |(\boldsymbol{\theta}_i^k)^\top \boldsymbol{\phi}_L(\mathbf{x}) - (\boldsymbol{\theta}^*)^\top \boldsymbol{\phi}_L(\mathbf{x})| \\ &\leq \lim_{k\to\infty} \|\boldsymbol{\theta}_i^k - \boldsymbol{\theta}^*\|_2 \|\boldsymbol{\phi}_L(\mathbf{x})\|_2 \\ &\leq \lim_{k\to\infty} \|\boldsymbol{\theta}_i^k - \boldsymbol{\theta}^*\|_2 \\ &= 0.\end{aligned} \tag{35}$$

■

Appendix B. Proof of Theorem 3

Proof. The empirical risk (6) to be minimized for the kernel regression problem in the RKHS is

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^T} \hat{R}(\boldsymbol{\alpha}) = \sum_{i=1}^N \hat{R}_i(\boldsymbol{\alpha}) = \sum_{i=1}^N \left(\frac{1}{T_i} \|\mathbf{y}_i - \mathbf{K}_i^\top \boldsymbol{\alpha}\|_2^2 + \lambda_i \boldsymbol{\alpha}^\top \mathbf{K} \boldsymbol{\alpha} \right), \tag{36}$$

where $\mathbf{y}_i = [y_{i,1}, \ldots, y_{i,T_i}]^\top \in \mathbb{R}^{T_i \times 1}$, the matrices $\mathbf{K}_i \in \mathbb{R}^{T\times T_i}$ and $\mathbf{K} \in \mathbb{R}^{T\times T}$ are used to store the similarity of the total data and data from agent i, and the similarity of all data, respectively, with the assumption that all data are available to all agents. The optimal solution is given in closed form by

$$\boldsymbol{\alpha}^* = (\tilde{\mathbf{K}}^\top \tilde{\mathbf{K}} + \lambda \mathbf{K})^{-1} \tilde{\mathbf{K}} \tilde{\mathbf{y}}, \tag{37}$$

where $\tilde{\mathbf{K}} = [\tilde{\mathbf{K}}_1, \ldots, \tilde{\mathbf{K}}_N] \in \mathbb{R}^{T\times T}$ with $\tilde{\mathbf{K}}_i = \frac{1}{\sqrt{T_i}} \mathbf{K}_i$, $\forall i \in \mathcal{N}$, $\tilde{\mathbf{y}} = [\tilde{\mathbf{y}}_1; \ldots; \tilde{\mathbf{y}}_N] \in \mathbb{R}^{T\times 1}$ with $\tilde{\mathbf{y}}_i = \frac{1}{\sqrt{T_i}} \mathbf{y}_i$, $\forall i \in \mathcal{N}$, and $\lambda = \sum_{i=1}^N \lambda_i$. Denote the predicted values on the training examples using $\boldsymbol{\alpha}^*$ as $\mathbf{f}_{\boldsymbol{\alpha}^*}^i \in \mathbb{R}^{T_i}$ for node i and the overall predictions as $\mathbf{f}_{\boldsymbol{\alpha}^*} = [\mathbf{f}_{\boldsymbol{\alpha}^*}^1; \ldots; \mathbf{f}_{\boldsymbol{\alpha}^*}^N] \in \mathbb{R}^T$. In the corresponding RF space, we can denote the predicted values obtained for node i by $\boldsymbol{\theta}^*$ in (26) as $\mathbf{f}_{\boldsymbol{\theta}^*}^i \in \mathbb{R}^{T_i}$ and the overall prediction by $\mathbf{f}_{\boldsymbol{\theta}^*} = [\mathbf{f}_{\boldsymbol{\theta}^*}^1; \ldots; \mathbf{f}_{\boldsymbol{\theta}^*}^N] \in \mathbb{R}^T$.

To prove Theorem 3, we start by customizing several lemmas and theorems from the literature, which facilitate proving our main results.

Definition 1 *(Bartlett and Mendelson, 2002, Definition 2) Let $\{\mathbf{x}_q\}_{q=1}^Q$ be i.i.d samples drawn from the probability distribution $p_{\mathcal{X}}$. Let $\mathcal{F}$ be a class of functions that map $\mathcal{X}$ to $\mathbb{R}$. Define the random variable*

$$\hat{\mathfrak{R}}_Q(\mathcal{F}) := \mathbb{E}_\epsilon \left[\sup_{f\in\mathcal{F}} \left| \frac{2}{Q} \sum_{q=1}^Q \epsilon_q f(\mathbf{x}_q) \right| \Big| \mathbf{x}_1, \ldots, \mathbf{x}_Q \right], \tag{38}$$

where $\{\epsilon_q\}_{q=1}^Q$ are i.i.d. $\{\pm 1\}$-valued random variables with $\mathbb{P}(\epsilon_q = 1) = \mathbb{P}(\epsilon_q = -1) = \frac{1}{2}$. Then, the Rademacher complexity of $\mathcal{F}$ is defined as

$$\mathfrak{R}_Q(\mathcal{F}) := \mathbb{E}\left[\hat{\mathfrak{R}}_Q(\mathcal{F}) \right]. \tag{39}$$

Rademacher complexity is adopted in machine learning and theory of computation to measure the richness of a class of real-valued functions with respect to a probability distribution. Here we adopt it to measure the richness of functions defined in the RKHS induced by the positive definite kernel κ with respect to the sample distribution p.

Lemma 2 *(Bartlett and Mendelson, 2002, Lemma 22) Let $\mathcal{H}$ be a RKHS associated with a positive definite kernel κ that maps $\mathcal{X}$ to $\mathbb{R}$. Then, we have $\hat{\mathfrak{R}}_Q(\mathcal{H}) \leq \frac{2}{Q}\sqrt{Tr(\mathbf{K})}$, where $\mathbf{K}$ is the kernel matrix for kernel κ over the i.i.d. sample set $\{\mathbf{x}_q\}_{q=1}^Q$. Correspondingly, the Rademacher complexity satisfies $\mathfrak{R}_Q(\mathcal{H}) \leq \frac{2}{Q}\mathbb{E}\left[\sqrt{Tr(\mathbf{K})}\right]$.*

The next theorems state that the generalization performance of a particular estimator in $\mathcal{H}$ not only depends on the number of data points, but also depends on the complexity of $\mathcal{H}$.

Theorem 6 *(Bartlett and Mendelson, 2002, Theorem 8, Theorem 12) Let $\{\mathbf{x}_q, y_q\}_{q=1}^Q$ be i.i.d samples drawn from the distribution p defined on $\mathcal{X} \times \mathcal{Y}$. Assume the loss function $\ell : \mathcal{Y} \times \mathbb{R} \to [0,1]$ is Lipschitz continuous with a Lipschitz constant M_ℓ. Define the expected risk for all $f \in \mathcal{H}$ be $\mathcal{E}(f) = \mathbb{E}_p[\ell(f(\mathbf{x}), y)]$, and its corresponding empirical risk be $\hat{\mathcal{E}}(f) = \frac{1}{Q}\sum_{q=1}^Q \ell(y_q, f(\mathbf{x}_q))$. Then, for $\delta_p \in (0,1)$, with probability at least $1-\delta_p$, every $f \in \mathcal{H}$ satisfies*

$$\mathcal{E}(f) \leq \hat{\mathcal{E}}(f) + \mathfrak{R}_Q(\tilde{\ell} \circ \mathcal{H}) + \sqrt{\frac{8\log(2/\delta_p)}{Q}}, \tag{40}$$

where $\tilde{\ell} \circ \mathcal{H} = \{(\mathbf{x}, y) \to \ell(y, f(\mathbf{x})) - \ell(y, 0) | f \in \mathcal{H}\}$.

Theorem 7 *(Bartlett and Mendelson, 2002, Theorem 12) If $\ell : \mathcal{Y} \times \mathbb{R} \to [0,1]$ is Lipschitz with constant M_ℓ and satisfies $\ell(0) = 0$, then $\mathfrak{R}_Q(\tilde{\ell} \circ \mathcal{H}) \leq 2M_\ell \mathfrak{R}_Q(\mathcal{H})$.*

Lemma 3 *(Li et al., 2018, Modified Proposition 1) For the RKHS induced by the kernel κ with expression (10), define $\hat{\mathcal{H}}^k := \{\hat{f}^k : \hat{f}^k = (\boldsymbol{\theta}^k)^\top \boldsymbol{\phi}_L(\mathbf{x}) = \sum_{l=1}^L \theta_l^k \phi(\mathbf{x}, \boldsymbol{\omega}_l)$, then we have $\forall \hat{f}^k \in \hat{\mathcal{H}}^k, \|\hat{f}^k\|_{\hat{\mathcal{H}}^k}^2 \leq \|\boldsymbol{\theta}^k\|_2^2$, where $\hat{\mathcal{H}}^k$ is the RKHS of functions $\hat{f}^k$ at the k-th step. The kernel that induces $\hat{\mathcal{H}}^k$ is the approximated kernel $\hat{\kappa}_L$ defined in (11).*

Lemma 4 *(Li et al., 2018, Lemma 6) For the decentralized kernel regression problem defined in Section 2, let $\mathbf{f}_{\boldsymbol{\alpha}^*}$, $\mathbf{f}_{\boldsymbol{\theta}^*}$ be the predictions obtained by (37) and (26), respectively. Then, we have*

$$\langle \mathbf{y} - \mathbf{f}_{\boldsymbol{\alpha}^*}, \mathbf{f}_{\boldsymbol{\theta}^*} - \mathbf{f}_{\boldsymbol{\alpha}^*} \rangle = 0. \tag{41}$$

Theorem 8 *(Li et al., 2018, Modified Theorem 5) For the decentralized kernel regression problem defined in Section 2, let $\lambda_{\mathbf{K}}$ be the largest eigenvalue of the kernel matrix $\mathbf{K}$, and choose the regularization parameter $\lambda < \lambda_{\mathbf{K}}/T$ so as to control overfitting. Then, for all $\delta_p \in (0,1)$ and $\|f\|_{\mathcal{H}} \leq 1$, if the number of random features L satisfies*

$$L \geq \frac{1}{\lambda}\left(\frac{1}{\epsilon^2} + \frac{2}{3\epsilon}\right)\log\frac{16 d_{\mathbf{K}}^{\lambda}}{\delta_p},$$

then with probability at least $1-\delta_p$*, the following equation holds*

$$\sup_{\|f\|_{\mathcal{H}}\leq 1}\inf_{\|\boldsymbol{\theta}\|\leq\sqrt{2/L}}\frac{1}{T}\|\mathbf{f}_{\mathbf{x}}-\mathbf{f}_{\boldsymbol{\theta}^*}\|_2^2\leq 2\lambda, \tag{42}$$

where $\mathbf{f}_{\mathbf{x}}\in\mathbb{R}^T$ *is the predictions evaluated by* $f_{\mathcal{H}}$ *on all samples and* $\epsilon\in(0,1)$*.*

With the above lemmas and theorems, we are ready to prove Theorem 3, which relies on the following decomposition:

$$\begin{aligned}\mathcal{E}(\hat{f}^k)-\mathcal{E}(f_{\mathcal{H}})=&\underbrace{\mathcal{E}(\hat{f}^k)-\hat{\mathcal{E}}(\hat{f}^k)}_{\text{(1) estimation error}}+\underbrace{\hat{\mathcal{E}}(\hat{f}^k)-\hat{\mathcal{E}}(\hat{f}_{\boldsymbol{\theta}^*})}_{\text{(2) convergence error}}+\underbrace{\hat{\mathcal{E}}(\hat{f}_{\boldsymbol{\theta}^*})-\hat{\mathcal{E}}(\hat{f}_{\boldsymbol{\alpha}^*})}_{\text{(3) approximation error of RF mapping}}\\&+\underbrace{\hat{\mathcal{E}}(\hat{f}_{\boldsymbol{\alpha}^*})-\mathcal{E}(\hat{f}_{\boldsymbol{\alpha}^*})}_{\text{(4) estimation error}}+\underbrace{\mathcal{E}(\hat{f}_{\boldsymbol{\alpha}^*})-\mathcal{E}(f_{\mathcal{H}})}_{\text{(5) approximation error of kernel representation}},\end{aligned} \tag{43}$$

where $\mathcal{E}(\hat{f}^k)$, $\hat{\mathcal{E}}(\hat{f}^k)$, $\mathcal{E}(\hat{f}_{\boldsymbol{\theta}^*})$, $\hat{\mathcal{E}}(\hat{f}_{\boldsymbol{\theta}^*})$, $\hat{\mathcal{E}}(\hat{f}_{\boldsymbol{\alpha}^*})$, $\mathcal{E}(\hat{f}_{\boldsymbol{\alpha}^*})$ are defined as follows for the kernel regression problem:

$$\begin{aligned}\mathcal{E}(\hat{f}^k)&:=\sum_{i=1}^N\mathcal{E}_i(\hat{f}_{\boldsymbol{\theta}_i^k})=\sum_{i=1}^N\mathbb{E}_p[(y-(\boldsymbol{\theta}_i^k)^\top\boldsymbol{\phi}_L(\mathbf{x}))^2]:=\mathbb{E}_p[\|\mathbf{y}_N-\boldsymbol{\Phi}_N\tilde{\boldsymbol{\Theta}}^k\|_2^2],\\
\hat{\mathcal{E}}(\hat{f}^k)&:=\sum_{i=1}^N\hat{\mathcal{E}}_i(\hat{f}_{\boldsymbol{\theta}_i^k})=\sum_{i=1}^N\frac{1}{T_i}\sum_{t=1}^{T_i}\|\mathbf{y}_i-(\boldsymbol{\Phi}_L^i)^\top\boldsymbol{\theta}_i^k\|_2^2=\sum_{i=1}^N\|\tilde{\mathbf{y}}_i-(\tilde{\boldsymbol{\Phi}}_L^i)^\top\boldsymbol{\theta}_i^k\|_2^2=\|\tilde{\mathbf{y}}-\tilde{\boldsymbol{\Phi}}_B\tilde{\boldsymbol{\Theta}}^k\|_2^2,\\
\hat{\mathcal{E}}(\hat{f}_{\boldsymbol{\theta}^*})&:=\sum_{i=1}^N\hat{\mathcal{E}}_i(\hat{f}_{\boldsymbol{\theta}^*})=\sum_{i=1}^N\frac{1}{T_i}\sum_{t=1}^{T_i}\|\mathbf{y}_i-(\boldsymbol{\Phi}_L^i)^\top\boldsymbol{\theta}^*\|_2^2=\sum_{i=1}^N\|\tilde{\mathbf{y}}_i-(\tilde{\boldsymbol{\Phi}}_L^i)^\top\boldsymbol{\theta}^*\|_2^2=\|\tilde{\mathbf{y}}-\tilde{\boldsymbol{\Phi}}_B\tilde{\boldsymbol{\Theta}}^*\|_2^2,\\
\mathcal{E}(\hat{f}_{\boldsymbol{\alpha}^*})&:=\sum_{i=1}^N\hat{\mathcal{E}}_i(\hat{f}_{\boldsymbol{\alpha}^*})=\sum_{i=1}^N\mathbb{E}_p[(y-(\boldsymbol{\alpha}^*)^\top\boldsymbol{\kappa}(\mathbf{x}))^2]:=\mathbb{E}_p[(y-(\boldsymbol{\alpha}^*)^\top\boldsymbol{\kappa}(\mathbf{x}))^2],\\
\hat{\mathcal{E}}(\hat{f}_{\boldsymbol{\alpha}^*})&:=\sum_{i=1}^N\frac{1}{T_i}\|\mathbf{y}_i-\mathbf{K}_i\boldsymbol{\alpha}^*\|_2^2=\sum_{i=1}^N\|\tilde{\mathbf{y}}_i-\tilde{\mathbf{K}}_i\boldsymbol{\alpha}^*\|_2^2,\end{aligned}$$

where $\mathbf{y}_N=y\mathbf{1}_N$, $\boldsymbol{\Phi}_N=\begin{bmatrix}\boldsymbol{\phi}_L(\mathbf{x})&\cdots&\mathbf{0}\\\vdots&\ddots&\vdots\\\mathbf{0}&\cdots&\boldsymbol{\phi}_L(\mathbf{x})\end{bmatrix}\in\mathbb{R}^{N\times NL}$, $\tilde{\boldsymbol{\Phi}}_B=\begin{bmatrix}(\tilde{\boldsymbol{\Phi}}_L^1)^\top&\cdots&\mathbf{0}\\\vdots&\ddots&\vdots\\\mathbf{0}&\cdots&(\tilde{\boldsymbol{\Phi}}_L^N)^\top\end{bmatrix}\in\mathbb{R}^{T\times NL}$, $\tilde{\boldsymbol{\Theta}}^k=[\boldsymbol{\theta}_1^k;\ldots;\boldsymbol{\theta}_i^k]\in\mathbb{R}^{NL}$, and $\tilde{\boldsymbol{\Theta}}^*=[\boldsymbol{\theta}^*;\ldots;\boldsymbol{\theta}^*]\in\mathbb{R}^{NL}$.

Then, we upper bound the excessive risk of $\mathcal{E}(\hat{f}^k)$ learned by COKE by upper bounding the decomposed five terms. For term (1), for $\delta_{p_1}\in(0,1)$, with probability at least $1-\delta_{p_1}$,

we have

$$\begin{aligned}\mathcal{E}(\hat{f}^k) - \hat{\mathcal{E}}(\hat{f}^k) &\leq 2M_{\ell_1}\mathfrak{R}_T(\tilde{\ell}_1 \circ \hat{\mathcal{H}}^k) + \sqrt{\frac{8\log(2/\delta_{p_1})}{T}} \\ &\leq 2M_{\ell_1}\mathfrak{R}_T(\hat{\mathcal{H}}^k) + \sqrt{\frac{8\log(2/\delta_{p_1})}{T}} \\ &\leq \frac{4M_{\ell_1}}{T}\mathbb{E}[Tr(\hat{\mathbf{K}})] + \sqrt{\frac{8\log(2/\delta_{p_1})}{T}} \\ &\leq \frac{4M_{\ell_1}}{T}\sqrt{T} + \sqrt{\frac{8\log(2/\delta_{p_1})}{T}} \\ &= \frac{C_1}{\sqrt{T}},\end{aligned} \tag{44}$$

where $C_1 := 4M_{\ell_1} + \sqrt{8\log(2/\delta_{p_1})}$, and M_{ℓ_1} is the Lipschitz constant for loss function $\ell_1(\hat{f}_{\boldsymbol{\theta}_i^k}, y) = ((\boldsymbol{\theta}_i^k)^\top \boldsymbol{\phi}_L(\mathbf{x}) - y)^2$. The first inequality comes from Theorem 6, the second inequality comes from Theorem 7, and the third inequality comes from Lemma 2. For the last inequality, each element in the Gram matrix $\hat{\mathbf{K}} \in \mathbb{R}^{T\times T}$ is given by (11), thus $Tr(\hat{\mathbf{K}}) \leq T\|\boldsymbol{\phi}_L(\mathbf{x})\|_2^2 \leq T$ with the adopted RF mapping such that $\|\boldsymbol{\phi}_L(\mathbf{x})\|_2^2 \leq 1$.

Similarly, for term (4), with probability at least $1 - \delta_{p_2}$ for $\delta_{p_2} \in (0, 1)$, the following holds,

$$\begin{aligned}\hat{\mathcal{E}}(\hat{f}_{\boldsymbol{\alpha}^*}) - \mathcal{E}(\hat{f}_{\boldsymbol{\alpha}^*}) &\leq 2M_{\ell_2}\mathfrak{R}_T(\tilde{\ell}_2 \circ \mathcal{H}) + \sqrt{\frac{8\log(2/\delta_{p_2})}{T}} \\ &\leq 2M_{\ell_2}\mathfrak{R}_T(\mathcal{H}) + \sqrt{\frac{8\log(2/\delta_{p_2})}{T}} \\ &\leq \frac{4M_{\ell_2}}{T}\mathbb{E}[Tr(\mathbf{K})] + \sqrt{\frac{8\log(2/\delta_{p_2})}{T}} \\ &\leq \frac{4M_{\ell_2}}{T}\sqrt{T} + \sqrt{\frac{8\log(2/\delta_{p_2})}{T}} \\ &= \frac{C_2}{\sqrt{T}},\end{aligned} \tag{45}$$

where $C_2 := 4M_{\ell_2} + \sqrt{8\log(2/\delta_{p_2})}$, and M_{ℓ_2} is the Lipschitz constant for the loss function $\ell_2(\hat{f}_{\boldsymbol{\alpha}^*}, y) = ((\boldsymbol{\alpha}^*)^\top \boldsymbol{\kappa}(\mathbf{x}) - y)^2$.

For term (2), we have

$$\begin{aligned}\hat{\mathcal{E}}(\hat{f}^k) - \hat{\mathcal{E}}(\hat{f}_{\boldsymbol{\theta}^*}) &= \|\tilde{\mathbf{y}} - \tilde{\boldsymbol{\Phi}}_B\tilde{\boldsymbol{\Theta}}^k\|_2^2 - \|\tilde{\mathbf{y}} - \tilde{\boldsymbol{\Phi}}_B\tilde{\boldsymbol{\Theta}}^*\|_2^2 \\ &\leq \nabla\left(\|\tilde{\mathbf{y}} - \tilde{\boldsymbol{\Phi}}_B\tilde{\boldsymbol{\Theta}}^*\|_2^2\right)\|\tilde{\boldsymbol{\Theta}}^k - \tilde{\boldsymbol{\Theta}}^*\|_2 + \frac{M_{\ell_3}}{2}\|\tilde{\boldsymbol{\Theta}}^k - \tilde{\boldsymbol{\Theta}}^*\|_2 \\ &\leq \left(\|\tilde{\boldsymbol{\Phi}}_B^\top(\tilde{\boldsymbol{\Phi}}_B\tilde{\boldsymbol{\Theta}}^* - \tilde{\mathbf{y}})\|_2 + \frac{M_{\ell_3}}{2}\right)\|\tilde{\boldsymbol{\Theta}}^k - \tilde{\boldsymbol{\Theta}}^*\|_2 \\ &= C_3\|\tilde{\boldsymbol{\Theta}}^k - \tilde{\boldsymbol{\Theta}}^*\|_2,\end{aligned} \tag{46}$$

where $C_3 := \|\tilde{\boldsymbol{\Phi}}_B^\top(\tilde{\boldsymbol{\Phi}}_B\tilde{\boldsymbol{\Theta}}^* - \tilde{\mathbf{y}})\|_2 + \frac{M_{\ell_3}}{2}$, and M_{ℓ_3} is the Lipschitz constant of the loss function $\ell_3(\tilde{\mathbf{y}}, \tilde{\boldsymbol{\Theta}}) = \|\tilde{\mathbf{y}} - \tilde{\boldsymbol{\Phi}}_B\tilde{\boldsymbol{\Theta}}\|_2^2$. From Theorem 4 and 5, we conclude $\{\tilde{\boldsymbol{\Theta}}^k\}$ converges linearly to $\tilde{\boldsymbol{\Theta}}^*$.

Term (3) is the approximation error caused by RF mapping, which is bounded by

$$\begin{aligned}
\hat{\mathcal{E}}(\hat{f}_{\boldsymbol{\theta}^*}) - \hat{\mathcal{E}}(\hat{f}_{\boldsymbol{\alpha}^*}) &= \sum_{i=1}^{N} \|\tilde{\mathbf{y}}_i - (\tilde{\boldsymbol{\Phi}}_L^i)^\top \boldsymbol{\theta}^*\|_2^2 - \sum_{i=1}^{N} \|\tilde{\mathbf{y}}_i - \tilde{\mathbf{K}}_i \boldsymbol{\alpha}^*\|_2^2 \\
&= \|\tilde{\mathbf{y}} - \tilde{\mathbf{f}}_{\boldsymbol{\theta}^*}\|_2^2 - \|\tilde{\mathbf{y}} - \tilde{\mathbf{f}}_{\boldsymbol{\alpha}^*}\|_2^2 \\
&= \|(\tilde{\mathbf{y}} - \tilde{\mathbf{f}}_{\boldsymbol{\alpha}^*}) + (\tilde{\mathbf{f}}_{\boldsymbol{\alpha}^*} - \tilde{\mathbf{f}}_{\boldsymbol{\theta}^*})\|_2^2 - \|\tilde{\mathbf{y}} - \tilde{\mathbf{f}}_{\boldsymbol{\alpha}^*}\|_2^2 \\
&= \inf_{\|\tilde{\mathbf{f}}_{\boldsymbol{\theta}}\|} \Big(\|\tilde{\mathbf{y}} - \tilde{\mathbf{f}}_{\boldsymbol{\alpha}^*}\|_2^2 + \|\tilde{\mathbf{f}}_{\boldsymbol{\alpha}^*} - \tilde{\mathbf{f}}_{\boldsymbol{\theta}}\|_2^2 + 2\langle \tilde{\mathbf{y}} - \tilde{\mathbf{f}}_{\boldsymbol{\alpha}^*}, \tilde{\mathbf{f}}_{\boldsymbol{\alpha}^*} - \tilde{\mathbf{f}}_{\boldsymbol{\theta}} \rangle \Big) - \|\tilde{\mathbf{y}} - \tilde{\mathbf{f}}_{\boldsymbol{\alpha}^*}\|_2^2 \\
&= \inf_{\|\tilde{\mathbf{f}}_{\boldsymbol{\theta}}\|} \|\tilde{\mathbf{f}}_{\boldsymbol{\alpha}^*} - \tilde{\mathbf{f}}_{\boldsymbol{\theta}}\|_2^2 + 2 \inf_{\|\tilde{\mathbf{f}}_{\boldsymbol{\theta}}\|} \langle \tilde{\mathbf{y}} - \tilde{\mathbf{f}}_{\boldsymbol{\alpha}^*}, \tilde{\mathbf{f}}_{\boldsymbol{\alpha}^*} - \tilde{\mathbf{f}}_{\boldsymbol{\theta}} \rangle \\
&\leq \inf_{\|\tilde{\mathbf{f}}_{\boldsymbol{\theta}}\|} \|\tilde{\mathbf{f}}_{\boldsymbol{\alpha}^*} - \tilde{\mathbf{f}}_{\boldsymbol{\theta}}\|_2^2 + 2\langle \tilde{\mathbf{y}} - \tilde{\mathbf{f}}_{\boldsymbol{\alpha}^*}, \tilde{\mathbf{f}}_{\boldsymbol{\alpha}^*} - \tilde{\mathbf{f}}_{\boldsymbol{\theta}^*} \rangle \\
&= \inf_{\|\tilde{\mathbf{f}}_{\boldsymbol{\theta}}\|} \|\tilde{\mathbf{f}}_{\boldsymbol{\alpha}^*} - \tilde{\mathbf{f}}_{\boldsymbol{\theta}}\|_2^2 \\
&\leq \sup_{\|\tilde{\mathbf{f}}_{\mathbf{x}}\|} \inf_{\|\tilde{\mathbf{f}}_{\boldsymbol{\theta}}\|} \|\tilde{\mathbf{f}}_{\mathbf{x}} - \tilde{\mathbf{f}}_{\boldsymbol{\theta}}\|_2^2 \\
&\leq 2\lambda,
\end{aligned} \tag{47}$$

where the seventh equality comes from Lemma 4 while the last inequality comes from Theorem 8 with $\tilde{\mathbf{f}}_{\mathbf{x}} := [\frac{1}{\sqrt{T_1}}\mathbf{f}_1; \ldots; \frac{1}{\sqrt{T_N}}\mathbf{f}_N] \in \mathbb{R}^T$ and $\mathbf{f}_i = [f(\mathbf{x}_{i,1}), \ldots, f(\mathbf{x}_{i,T_i})]^\top \in \mathbb{R}^{T_i}$ for $f \in \mathcal{H}$.

To bound term (5) of the approximation error of the models in the RKHS $\mathcal{H}$, we refer to the following Lemma.

Lemma 5 *(Rudi and Rosasco, 2017, Modified Lemma 5) For the kernel κ that can be represented as* (10) *and bounded RF mapping, that is $\|\phi(\mathbf{x}, \boldsymbol{\omega})\| \leq 1$ for any $\mathbf{x} \in \mathcal{X}$, under Assumption 4, the following holds for any regularization parameter $\lambda > 0$,*

$$\mathcal{E}(\hat{f}_{\boldsymbol{\alpha}^*}) - \mathcal{E}(f_{\mathcal{H}}) = \|\hat{f}_{\boldsymbol{\alpha}^*} - P f_p\|_{p_{\mathcal{X}}}^2 \leq (R\lambda^r)^2.$$

In Lemma 5, f_p is the ideal minimizer given the prior knowledge of the marginal distribution $p_{\mathcal{X}}$ of $\mathbf{x}$ and P is a projection operator on f_p so that Pf_p is the optimal minimizer in RKHS. The parameter $r \in [1/2, 1)$ is equivalent to assuming $f_{\mathcal{H}}$ exits, and R can take value as either 1 or $\|\hat{f}_{\boldsymbol{\alpha}^*}\|_{p_{\mathcal{X}}}$. Setting $r = 1/2$ and $R = 1$, we have

$$\mathcal{E}(\hat{f}_{\boldsymbol{\alpha}^*}) - \mathcal{E}(f_{\mathcal{H}}) \leq \lambda. \tag{48}$$

Combining (44)-(48) gives

$$\begin{aligned}
\lim_{k\to\infty} \mathcal{E}(\hat{f}^k) - \mathcal{E}(f_{\mathcal{H}})) &\leq \lim_{k\to\infty} \left[\frac{C_1}{\sqrt{T}} + C_3 \|\tilde{\boldsymbol{\Theta}}^k - \tilde{\boldsymbol{\Theta}}^*\|_2 + 2\lambda + \frac{C_2}{\sqrt{T}} + \lambda \right] \\
&= \lim_{k\to\infty} \left[3\lambda + \frac{C_1 + C_2}{\sqrt{T}} + C_3 \|\tilde{\boldsymbol{\Theta}}^k - \tilde{\boldsymbol{\Theta}}^*\|_2 \right] \\
&= 3\lambda + O(\frac{1}{\sqrt{T}}),
\end{aligned} \tag{49}$$

and completes the proof. ■

References

Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. Qsgd: Communication-efficient sgd via gradient quantization and encoding. In *Advances in Neural Information Processing Systems*, pages 1709–1720. 2017.

Dan Alistarh, Torsten Hoefler, Mikael Johansson, Nikola Konstantinov, Sarit Khirirat, and Cedric Renggli. The convergence of sparsified gradient methods. In *Advances in Neural Information Processing Systems*, pages 5973–5983. 2018.

Reza Arablouei, Stefan Werner, Kutluyıl Doğançay, and Yih-Fang Huang. Analysis of a reduced-communication diffusion lms algorithm. *Signal Processing*, 117:355–361, 2015.

Arthur Asuncion and David Newman. UCI machine learning repository, 2007.

Haim Avron, Michael Kapralov, Cameron Musco, Christopher Musco, Ameya Velingker, and Amir Zandieh. Random Fourier features for kernel ridge regression: Approximation bounds and statistical guarantees. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 253–262. JMLR.org, 2017.

Francis Bach. On the equivalence between kernel quadrature rules and random feature expansions. *Journal of Machine Learning Research*, 18(1):714–751, 2017.

Peter L Bartlett and Shahar Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

Eduard Gabriel Băzăvan, Fuxin Li, and Cristian Sminchisescu. Fourier kernel learning. In *European Conference on Computer Vision*, pages 459–473. Springer, 2012.

Salomon Bochner. *Harmonic analysis and the theory of probability*. Courier Corporation, 2005.

Pantelis Bouboulis, Symeon Chouvardas, and Sergios Theodoridis. Online distributed learning over networks in RKH spaces using random fourier features. *IEEE Transactions on Signal Processing*, 66(7):1920–1932, 2018.

Serhat Bucak, Rong Jin, and Anil K Jain. Multi-label multiple kernel learning by stochastic approximation: Application to visual object recognition. In *Advances in Neural Information Processing Systems*, pages 325–333, 2010.

Luis M Candanedo, Véronique Feldheim, and Dominique Deramaix. Data driven prediction models of energy use of appliances in a low-energy house. *Energy and Buildings*, 140:81–97, 2017.

Andrea Caponnetto and Ernesto De Vito. Optimal rates for the regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7(3):331–368, 2007.

Tianyi Chen, Georgios Giannakis, Tao Sun, and Wotao Yin. Lag: Lazily aggregated gradient for communication-efficient distributed learning. In *Advances in Neural Information Processing Systems*, pages 5050–5060, 2018.

Symeon Chouvardas and Moez Draief. A diffusion kernel LMS algorithm for nonlinear adaptive networks. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4164–4168. IEEE, 2016.

Bo Dai, Bo Xie, Niao He, Yingyu Liang, Anant Raj, Maria-Florina F Balcan, and Le Song. Scalable kernel methods via doubly stochastic gradients. In *Advances in Neural Information Processing Systems*, pages 3041–3049, 2014.

Saverio De Vito, Ettore Massera, Marco Piga, Luca Martinotto, and Girolamo Di Francia. On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario. *Sensors and Actuators B: Chemical*, 129(2):750–757, 2008.

Giacomo Vincenzo Demarie and Donato Sabia. A machine learning approach for the automatic long-term structural health monitoring. *Structural Health Monitoring*, 18(3): 819–837, 2019.

Petros Drineas and Michael W Mahoney. On the Nyström method for approximating a Gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, 6:2153–2175, 2005.

Yaakov Engel, Shie Mannor, and Ron Meir. The kernel recursive least-squares algorithm. *IEEE Transactions on Signal Processing*, 52(8):2275–2285, 2004.

Francisco Facchinei, Gesualdo Scutari, and Simone Sagratella. Parallel selective algorithms for nonconvex big data optimization. *IEEE Transactions on Signal Processing*, 63(7): 1874–1889, 2015.

Wei Gao, Jie Chen, Cédric Richard, and Jianguo Huang. Diffusion adaptation over networks with kernel least-mean-square. In *2015 IEEE 6th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 217–220. IEEE, 2015.

Ryan Gomes and Andreas Krause. Budgeted nonparametric learning from data streams. In *International Conference on Machine Learning (ICML)*, 2010.

Bin Gu, Miao Xin, Zhouyuan Huo, and Heng Huang. Asynchronous doubly stochastic sparse kernel learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Ibrahim El Khalil Harrane, Rémi Flamary, and Cédric Richard. On reducing the communication cost of the diffusion lms algorithm. *IEEE Transactions on Signal and Information Processing over Networks*, 5(1):100–112, 2018.

Thomas Hofmann, Bernhard Schölkopf, and Alexander J Smola. Kernel methods in machine learning. *The Annals of Statistics*, pages 1171–1220, 2008.

Paul Honeine. Analyzing sparse dictionaries for online learning with kernels. *IEEE Transactions on Signal Processing*, 63(23):6343–6353, 2015.

Muhammad U Ilyas, M Zubair Shafiq, Alex X Liu, and Hayder Radha. A distributed algorithm for identifying information hubs in social networks. *IEEE Journal on Selected Areas in Communications*, 31(9):629–640, 2013.

Martin Jaggi, Virginia Smith, Martin Takác, Jonathan Terhorst, Sanjay Krishnan, Thomas Hofmann, and Michael I Jordan. Communication-efficient distributed dual coordinate ascent. In *Advances in Neural Information Processing Systems*, pages 3068–3076, 2014.

Xinrong Ji, Cuiqin Hou, Yibin Hou, Fang Gao, and Shulong Wang. A distributed learning method for ℓ-1 regularized kernel machine over wireless sensor networks. *Sensors*, 16(7): 1021, 2016.

François Kawala, Ahlame Douzal-Chouakria, Eric Gaussier, and Eustache Dimert. Prédictions d'activité dans les réseaux sociaux en ligne. 2013.

Alec Koppel, Garrett Warnell, Ethan Stump, and Alejandro Ribeiro. Parsimonious online learning with kernels via sparse projections in function space. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4671–4675. IEEE, 2017.

Alec Koppel, Santiago Paternain, Cédric Richard, and Alejandro Ribeiro. Decentralized online learning with kernels. *IEEE Transactions on Signal Processing*, 66(12):3240–3255, 2018.

Trung Le, Vu Nguyen, Tu Dinh Nguyen, and Dinh Phung. Nonparametric budgeted stochastic gradient descent. In *Artificial Intelligence and Statistics*, pages 654–572, 2016.

Boyue Li, Shicong Cen, Yuxin Chen, and Yuejie Chi. Communication-efficient distributed optimization in networks with gradient tracking. *arXiv preprint arXiv:1909.05844*, 2019a.

Mu Li, David G Andersen, Alexander J Smola, and Kai Yu. Communication efficient distributed machine learning with the parameter server. In *Advances in Neural Information Processing Systems*, pages 19–27, 2014.

Weiyu Li, Yaohua Liu, Zhi Tian, and Qing Ling. COLA: Communication-censored linearized ADMM for decentralized consensus optimization. In *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5237–5241. IEEE, 2019b.

Zhu Li, Jean-Francois Ton, Dino Oglic, and Dino Sejdinovic. Towards a unified analysis of random Fourier features. *arXiv preprint arXiv:1806.09178*, 2018.

Yaohua Liu, Wei Xu, Gang Wu, Zhi Tian, and Qing Ling. Communication-censored ADMM for decentralized consensus optimization. *IEEE Transactions on Signal Processing*, 67 (10):2565–2579, 2019.

Jing Lu, Steven CH Hoi, Jialei Wang, Peilin Zhao, and Zhi-Yong Liu. Large scale online kernel learning. *Journal of Machine Learning Research*, 17(1):1613–1655, 2016.

H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. Communication-efficient learning of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629*, 2016.

Rangeet Mitra and Vimal Bhatia. The diffusion-KLMS algorithm. In *2014 International Conference on Information Technology*, pages 256–259. IEEE, 2014.

Joao FC Mota, Joao MF Xavier, Pedro MQ Aguiar, and Markus Püschel. D-ADMM: A communication-efficient distributed algorithm for separable optimization. *IEEE Transactions on Signal Processing*, 61(10):2718–2723, 2013.

Angelia Nedić, Alex Olshevsky, and César A Uribe. A tutorial on distributed (non-bayesian) learning: Problem, algorithms and results. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 6795–6801. IEEE, 2016.

Tu Dinh Nguyen, Trung Le, Hung Bui, and Dinh Phung. Large-scale online kernel learning with random feature reparameterization. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2543–2549. AAAI Press, 2017.

Fernando Pérez-Cruz and Olivier Bousquet. Kernel methods and their potential use in signal processing. *IEEE Signal Processing Magazine*, 21(3):57–65, 2004.

Joel B Predd, Sanjeev R Kulkarni, and H Vincent Poor. Distributed kernel regression: An algorithm for training collaboratively. In *2006 IEEE Information Theory Workshop (ITW)*, pages 332–336. IEEE, 2006.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, pages 1177–1184, 2008.

Sashank J Reddi, Jakub Konečnỳ, Peter Richtárik, Barnabás Póczós, and Alex Smola. Aide: Fast and communication efficient distributed optimization. *arXiv preprint arXiv:1608.06879*, 2016.

Cédric Richard, José Carlos M Bermudez, and Paul Honeine. Online prediction of time series data with kernels. *IEEE Transactions on Signal Processing*, 57(3):1058–1067, 2008.

Alessandro Rudi and Lorenzo Rosasco. Generalization properties of learning with random features. In *Advances in Neural Information Processing Systems*, pages 3215–3225, 2017.

Ali H Sayed. Adaptation, learning, and optimization over networks. *Foundations and Trends in Machine Learning*, 7(ARTICLE):311–801, 2014.

Bernhard Schölkopf, Ralf Herbrich, and Alex J Smola. A generalized representer theorem. In *International Conference on Computational Learning Theory*, pages 416–426. Springer, 2001.

Ohad Shamir, Nathan Srebro, and Tong Zhang. Communication-efficient distributed optimization using an approximate newton-type method. In *International Conference on Machine Learning (ICML)*, pages 1000–1008, 2014.

John Shawe-Taylor, Nello Cristianini, et al. *Kernel methods for pattern analysis*. Cambridge university press, 2004.

Fatemeh Sheikholeslami, Dimitris Berberidis, and Georgios B Giannakis. Large-scale kernel-based feature extraction via low-rank subspace tracking on a budget. *IEEE Transactions on Signal Processing*, 66(8):1967–1981, 2018.

Yanning Shen, Tianyi Chen, and Georgios B Giannakis. Online multi-kernel learning with orthogonal random features. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6289–6293, 2018.

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

Ban-Sok Shin, Henning Paul, and Armin Dekorsy. Distributed kernel least squares for nonlinear regression applied to sensor networks. In *2016 24th European Signal Processing Conference (EUSIPCO)*, pages 1588–1592. IEEE, 2016.

Ban-Sok Shin, Masahiro Yukawa, Renato Luis Garrido Cavalcante, and Armin Dekorsy. Distributed adaptive learning with multiple kernels in diffusion networks. *IEEE Transactions on Signal Processing*, 66(21):5505–5519, 2018.

Bharath Sriperumbudur and Zoltán Szabó. Optimal rates for random Fourier features. In *Advances in Neural Information Processing Systems*, pages 1144–1152, 2015.

Sebastian U Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified sgd with memory. In *Advances in Neural Information Processing Systems*, pages 4447–4458. 2018.

Dougal J Sutherland and Jeff Schneider. On the error of random Fourier features. *arXiv preprint arXiv:1506.02785*, 2015.

Håkan Terelius, Ufuk Topcu, and Richard M Murray. Decentralized multi-agent optimization via dual decomposition. *IFAC Proceedings Volumes*, 44(1):11245–11251, 2011.

Zhuang Wang, Koby Crammer, and Slobodan Vucetic. Breaking the curse of kernelization: Budgeted stochastic gradient descent for large-scale svm training. *Journal of Machine Learning Research*, 13:3103–3131, 2012.

Jianqiao Wangni, Jialei Wang, Ji Liu, and Tong Zhang. Gradient sparsification for communication-efficient distributed optimization. In *Advances in Neural Information Processing Systems*, pages 1299–1309. 2018.

Keith Worden and Graeme Manson. The application of machine learning to structural health monitoring. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 365(1851):515–537, 2006.

Ping Xu, Zhi Tian, Zhe Zhang, and Yue Wang. COKE: Communication-censored kernel learning via random features. In *2019 IEEE Data Science Workshop (DSW)*, pages 32–36. IEEE, 2019.

Wotao Yin, Xianghui Mao, Kun Yuan, Yuantao Gu, and Ali H Sayed. A communication-efficient random-walk algorithm for decentralized optimization. *arXiv preprint arXiv:1804.06568*, 2018.

Felix Xinnan X Yu, Ananda Theertha Suresh, Krzysztof M Choromanski, Daniel N Holtmann-Rice, and Sanjiv Kumar. Orthogonal random features. In *Advances in Neural Information Processing Systems*, pages 1975–1983, 2016.

Yue Yu, Jiaxiang Wu, and Longbo Huang. Double quantization for communication-efficient distributed optimization. In *Advances in Neural Information Processing Systems*, pages 4440–4451. 2019.

Lijun Zhang, Jinfeng Yi, Rong Jin, Ming Lin, and Xiaofei He. Online kernel learning with a near optimal sparsity bound. In *International Conference on Machine Learning (ICML)*, pages 621–629, 2013.

Mingrui Zhang, Lin Chen, Aryan Mokhtari, Hamed Hassani, and Amin Karbasi. Quantized frank-wolfe: Communication-efficient distributed optimization. *arXiv preprint arXiv:1902.06332*, 2019.

Shengyu Zhu, Mingyi Hong, and Biao Chen. Quantized consensus ADMM for multi-agent distributed optimization. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4134–4138. IEEE, 2016.