Communication-Efficient Online Federated Learning Strategies for Kernel Regression

Vinay Chakravarthi Gogineni, *Member, IEEE*, Stefan Werner, *Senior Member, IEEE*, Yih-Fang Huang, *Life Fellow, IEEE*, and Anthony Kuh, *Fellow, IEEE*

***Abstract*—This paper presents communication-efficient approaches to federated learning for resource-constrained devices with access to streaming data. In particular, we first propose a partial-sharing-based framework for online federated learning, called PSO-Fed, wherein clients update local models from a stream of data and exchange tiny fractions of the model with the server, reducing the communication overhead. In contrast to classical federated learning approaches, the proposed strategy provides clients who are not part of a global iteration with the freedom to update local models whenever new data arrives. Furthermore, by devising a client-side innovation check, we also propose an event-triggered PSO-Fed (ETPSO-Fed) that further reduces the computational burden of clients while enhancing communication efficiency. We implement the abovementioned frameworks in the context of kernel regression, where clients perform local learning employing random Fourier features-based kernel least mean squares. In addition, we examine the mean and mean-square convergence of the proposed PSO-Fed. Finally, we conduct experiments to determine the efficacy of the proposed frameworks. Our results show that PSO-Fed and ETPSO-Fed can compete with Online-Fed while requiring significantly less communication overhead. Simulations demonstrate an 80% reduction in PSO-Fed and an 84.5% reduction in ETPSO-Fed communication overhead compared to Online-Fed. Notably, the proposed partial-sharing-based online FL strategies show good resilience against model-poisoning attacks without involving additional mechanisms.**

***Index Terms*—Online federated learning, communication-efficiency, partial-sharing, set-membership filtering, kernel least mean squares, random Fourier features, Byzantine attacks.**

I. Introduction

In the modern age, geographically dispersed edge devices have access to enormous data volumes. By performing model training on this entire data, the end-user experience can be enhanced in many tasks, such as regression, classification, and clustering. The standard machine learning approaches require edge devices to communicate their data to a centralized server or cloud for further processing. However, privacy concerns prevent edge devices from sharing their private data with the cloud. This concern led to the development of a new distributed learning framework, namely, federated learning (FL) [1]–[8], wherein edge devices connected to a server collaboratively train a global shared model using locally stored data without revealing it to others. The practical implementation of FL, however, poses many challenges. First, client devices own an unbalanced amount of non-IID data [9]–[11]. Second, the training phase of the global shared model requires a significant amount of communication overhead. Third, there is uneven client participation due to battery, bandwidth, memory, and computational constraints [12]–[14]. Lastly, the possible presence of adversarial clients, i.e., malicious clients trying to disrupt the learning and undermine model reliability, poses security and privacy concerns [15], [16]. Throughout this paper, our primary focus is on reducing communication overhead.

The federated average (FedAvg) is among the most popular methods for FL [17]. The FedAvg begins each global iteration round by sharing its aggregated model, the global model, with a fraction of all clients. Typically, the clients are selected uniformly at random; however, other methods exist that can enhance performance, see, e.g., [18]–[21]. Upon receiving the aggregated server model, clients perform several local learning iterations and then share the updated model parameters with the server. The server then fuses the parameters of the local model updates, yielding a new global model. The above update-aggregation procedure is reiterated until convergence, or when a predefined performance criterion is satisfied. Although the training phase of FedAvg summarized above intends to minimize the overall communication overhead, the resulting model accuracy depends heavily on the number of epochs executed by the clients [17], [22].

Modern machine learning models are typically quite large. Consequently, the training phase in FL will involve numerous iterations to finalize the globally shared model. Furthermore, each global round constitutes model exchanges between participating clients and the server, resulting in enormous communication overheads. A variety of solutions have been proposed in the literature for reducing this communication overhead. As an example, the work in [23] proposes a communication-mitigated federated learning scheme that discards irrelevant client updates by employing a model-alignment check. In [24], several strategies are considered to reduce communication overhead in the uplink. The first one is a structured update, where clients update the local model in a restricted space parametrized with fewer variables. Another approach is sketched update, wherein clients utilize 1-bit quantization, random rotations, and subsampling to compress the entire

The Research Council of Norway supported this work.

Vinay Chakravarthi Gogineni and Stefan Werner are with the Department of Electronic Systems, Norwegian University of Science and Technology, Trondheim 7491, Norway (e-mail: {vinay.gogineni, stefan.werner}@ntnu.no).

Yih-Fang Huang is with the Department of Electrical Engineering, University of Notre Dame, Notre Dame, IN 46556 USA (e-mail: huang@nd.edu).

Anthony Kuh is with the Department of Electrical Engineering, University of Hawaii at Manoa, Honolulu, HI 96822 USA (e-mail: kuh@hawaii.edu).

Anthony Kuh acknowledges support in part by NSF Grant 2142987.

locally updated model before sending it to the server. Despite their benefits in cutting down the amount of communication, sketch updates are resource-consuming and incur increased complexity for clients. These aspects further contribute to uneven client participation and limit their use in low-latency applications. The structured communication reduction for federated learning (FedSCR) [25] discards insignificant client updates for global learning when several model parameters remain constant. Recently proposed SignSGD and its variants [26] utilize sign-based gradient compressors to reduce the computational and communication costs of FL.

The above FL approaches assume fixed data batches locally stored at clients, which might be too restrictive in latency-sensitive or dynamic environments [27], [28]. Instead, clients could gain access to new data or even a continuous data stream during the model-training phase [29], [30]. Recently, in [31], online federated learning (Online-Fed) was discussed to cope with such scenarios. In Online-Fed, clients perform real-time and online learning, and the server combines received model parameters shared by the clients. Additionally, the asynchronous online federated learning framework (ASO-Fed) presented in [13] considered model learning in a scenario of uneven client participation and communication delays. Finally, in [32], a communication-efficient online asynchronous FL based on lossy compression was introduced; this approach, however, is associated with the previously discussed challenges encountered with competitive sketched updates.

In contrast to sketch/compressed updates, this paper proposes novel online FL strategies that involve less communication overhead and demand very little memory and computation on the client side. We exploit partial-sharing communication in tandem with online federated learning to develop a partial-sharing-based online federated learning (PSO-Fed) strategy. Clients adapt local models in the proposed PSO-Fed approach using local data streams and share only a fraction of the model updates with the server. Compared to Online-Fed, PSO-Fed allows non-participating clients to update local models whenever they acquire new data. This feature of PSO-Fed can be beneficial for resource-constrained devices, including straggler devices. To further cut down communication costs, we introduce an event-triggered PSO-Fed (ETPSO-Fed) wherein clients only perform local learning and model sharing when the newly available input data has sufficient innovation. Doing so further reduces the communication overhead and the computational burden on clients. To demonstrate the efficacy of the proposed FL schemes, we consider kernel regression in an environment with unbalanced and non-IID data. For this purpose, we update the local nonlinear regression model using the random Fourier features-based kernel LMS (RFF-KLMS) [33]–[35]. We carry out a detailed study on the mean and mean-square convergence of the proposed PSO-Fed strategy employing RFF-KLMS. Our numerical experiments confirm that the PSO-Fed and ETPSO-Fed have comparable performance with the Online-Fed while significantly reducing the overall communication cost. Furthermore, the proposed partial-sharing-based FL strategies demonstrate good resilience to model-poisoning attacks without additional robust aggregation or adversarial detection mechanisms, as in [36], [37].

This paper is organized as follows. Section II introduces the basics of FL and online-Fed for kernel regression. Then, we present communication-efficient online FL strategies, namely, PSO-Fed and ETPSO-Fed, in Section III. Section IV provides the detailed convergence analysis of the PSO-Fed. Results from numerical experiments to validate the performance of the proposed algorithms are presented in Section V. Finally, concluding remarks for this work are presented in Section VI.

II. Preliminaries

In this section, we first review the problem of kernel regression and present online FL for collaboratively training a global shared model in the context of kernel regression. Next, we briefly discuss the communication-efficient version of it called Online-Fed [31].

A. Kernel Regression

In many real-life scenarios, such as time-series prediction, channel equalization in communication systems, and regression, we frequently encounter nonlinear models whose input-output relationships at time index n can be described as

$$y_n = f(\mathbf{x}_n) + \nu_n, \tag{1}$$

where the input signal vector $\mathbf{x}_n = [x_n, x_{n-1}, \cdots, x_{n-L+1}]^{\mathrm{T}}$, y_n is the desired output and ν_n is the observation noise. The function $f : \mathbb{R}^L \to \mathbb{R}$ is a continuous nonlinear function. Linear estimation methods [38], [39] model these sophisticated input-output relationships poorly. Kernel methods that operate in reproducing kernel Hilbert space (RKHS) have been found to be efficient in estimating the nonlinear relationships represented by the function $f(\cdot)$ [40]–[42].

When estimating the nonlinear function $f(\cdot)$ in (1), kernel methods first transform the input regressor $\mathbf{x}_n \in \mathbb{R}^L$ into a high-dimensional feature space as $\boldsymbol{\phi}(\mathbf{x}_n)$, in which the inner products can be evaluated using kernels. A continuous, symmetric, and positive-definite kernel function $\kappa(\cdot,\cdot) : \mathbb{R}^L \times \mathbb{R}^L \to \mathbb{R}$, satisfies the following Mercer's condition [42]:

$$\kappa(\mathbf{x}_i, \mathbf{x}_n) = \boldsymbol{\phi}^{\mathrm{T}}(\mathbf{x}_i)\boldsymbol{\phi}(\mathbf{x}_n). \tag{2}$$

Inner products in higher dimensional space can be obtained via kernel function evaluation even without knowing the mapping $\boldsymbol{\phi}(\cdot)$. A kernel is said to be a reproducing kernel if it satisfies the reproducing property [42], namely,

$$\kappa(\mathbf{x}_i, \mathbf{x}_n) = \langle \kappa(\cdot, \mathbf{x}_i), \kappa(\cdot, \mathbf{x}_n) \rangle_{\mathcal{H}}, \tag{3}$$

where $\mathcal{H}$ is the RKHS in which the reproducing kernel is defined and $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ denotes the corresponding inner product. In (3), $\kappa(\cdot, \mathbf{x}_i)$ is a representer evaluation at $\mathbf{x}_i$. The focus of this paper is exclusively on the Gaussian kernel, which is a well-known Mercer kernel [42].

Given the data pairs $\{\mathbf{x}_i, y_i\}_{i=1}^{n-1} \bigcup \{\mathbf{x}_n\}$, from the representer theorem [42], the estimate of y_n (i.e., $\hat{y}_n$) can be expressed as

$$\hat{y}_n = \sum_{i=1}^{n-1} \alpha_i \, \kappa(\mathbf{x}_i, \mathbf{x}_n). \tag{4}$$

The kernel least mean squares algorithm (KLMS) [33] estimates the coefficients α_is by solving the following optimization problem:

$$\min_{\alpha_1,\ldots,\alpha_{n-1}} \mathbb{E}\left[\left(y_n - \sum_{i=1}^{n-1} \alpha_i\, \kappa(\mathbf{x}_i, \mathbf{x}_n)\right)^2\right]. \tag{5}$$

It can be seen from (5) that the dimensionality of the model increases as time progresses (i.e., the number of kernel evaluations required to obtain the system output increases with n). The growing dimensionality problem can be addressed with sparsification methods [40], [41]. These methods use the coherence check criterion and novelty criterion to discard the redundant input regressors. However, sparsification methods are resource-intensive and unsuitable for decentralized learning due to the need to train and broadcast a dictionary every time the underlying model changes.

It is possible to obtain a computationally efficient and flexible solution for (5) using random Fourier features (RFF) [34], [35], [43]–[46]. In the D-dimensional RFF space, a shift-invariant kernel evaluation, i.e., $\kappa(\mathbf{x}_i, \mathbf{x}_n) = \kappa(\mathbf{x}_i - \mathbf{x}_n)$, can be approximated using inner product. As a consequence, this approximation makes the estimation problem (5) a finite-dimensional linear estimation problem. Furthermore, kernel function evaluations are no longer needed. Suppose the mapping of $\mathbf{x}_n$ into the D-dimensional RFF space is $\mathbf{z}_n$. It is then possible to approximate the kernel evaluation by $\kappa(\mathbf{x}_i, \mathbf{x}_n) \approx \mathbf{z}_i^{\mathrm{T}}\mathbf{z}_n$. Thus, the estimate $\hat{y}_n$ in (4) can be approximated by

$$\hat{y}_n \approx \Big(\sum_{i=1}^{n-1} \alpha_i\, \mathbf{z}_i\Big)^{\mathrm{T}} \mathbf{z}_n = \mathbf{w}^{\mathrm{T}}\mathbf{z}_n, \tag{6}$$

where $\mathbf{w} = \sum_{i=1}^{n-1} \alpha_i\, \mathbf{z}_i$, is the linear representation of the function $f(\cdot)$ in D-dimensional RFF space. Various feature functions such as cosine, exponential, and Gaussian functions can be used to map $\mathbf{x}_n$ into D-dimensional RFF space. The cosine feature function computes $\mathbf{z}_n$ as [34]:

$$\mathbf{z}_n = (D/2)^{-\frac{1}{2}}\big[\cos(\mathbf{v}_1^{\mathrm{T}}\mathbf{x}_n + b_1), \ldots, \cos(\mathbf{v}_D^{\mathrm{T}}\mathbf{x}_n + b_D)\big]^{\mathrm{T}}, \tag{7}$$

where the vectors $\{\mathbf{v}_i\}_{i=1:P}$ are drawn from the probability density function $p(\mathbf{v})$ and phase terms $b_i \in \mathcal{U}\,[0,\, 2\pi]$, for $i = 1, \ldots, D$ (here $\mathcal{U}(\cdot)$ represents the uniform distribution). On the other hand, using the exponential feature function, $\mathbf{z}_n$ can be obtained as [35]:

$$\mathbf{z}_n = \big[\exp\big(-(\mathbf{v}_1^{\mathrm{T}}\mathbf{x}_n + b_1)\big), \ldots, \exp\big(-(\mathbf{v}_D^{\mathrm{T}}\mathbf{x}_n + b_D)\big)\big]^{\mathrm{T}}, \tag{8}$$

where $\{\mathbf{v}_i, b_i\}$ are the same as defined above. For the same D, the cosine feature function exhibits better performance than the exponential feature function but consumes more energy resources. In RFF space, the estimation problem (5) takes the following form:

$$\mathbf{w}_n = \min_{\mathbf{w}} \mathbb{E}\left[(y_n - \mathbf{w}^{\mathrm{T}}\mathbf{z}_n)^2\right]. \tag{9}$$

It is important to note that the usage of RFF avoids the need for maintaining a global dictionary.

B. Online Federated Learning

We consider a scenario wherein K dispersed clients communicate with a server. At time instant n, each client k has access to input signal $x_{k,n}$ and its associated desired output $y_{k,n}$, which are related by the following model:

$$y_{k,n} = f(\mathbf{x}_{k,n}) + \nu_{k,n}, \tag{10}$$

where $f(\cdot)$ specifies a continuous nonlinear model to be estimated, $\mathbf{x}_{k,n} = [x_{k,n}, x_{k,n-1}, \cdots, x_{k,n-L+1}]^{\mathrm{T}}$ and $\nu_{k,n}$ are the data vector and the observation noise local to client k, respectively. Here, the objective is to collaboratively estimate $f(\cdot)$, utilizing locally stored data at clients without leaking it to other clients. In particular, we want to solve the following optimization problem:

$$\min_{\mathbf{w}} \mathcal{J}(\mathbf{w}), \text{ where } \mathcal{J}(\mathbf{w}) = \frac{1}{K}\sum_{k=1}^{K} \mathcal{J}_k(\mathbf{w}). \tag{11}$$

Here, $\mathcal{J}_k(\mathbf{w})$ is the local objective function of kth client, given by

$$\mathcal{J}_k(\mathbf{w}) = \mathbb{E}\left[|y_{k,n} - \hat{y}_{k,n}|^2\right], \tag{12}$$

with $\hat{y}_{k,n} = \mathbf{w}^{\mathrm{T}}\mathbf{z}_{k,n}$, where the local model parameter vector $\mathbf{w} \in \mathbb{R}^D$, is a linear approximation of $f(\cdot)$ in a D-dimensional RFF space, and the mapping of $\mathbf{x}_{k,n}$ into the RFF space is denoted by $\mathbf{z}_{k,n} \in \mathbb{R}^D$.

C. Online-Fed

To achieve energy efficiency, Online-Fed allows the server to randomly select a subset of clients in every global iteration. During the nth global iteration, the set of randomly chosen client indices is denoted by $\mathcal{S}_n$, where $C = |\mathcal{S}_n|$ is the cardinality of $\mathcal{S}_n$. All clients have equal chances of being selected and the probability is $p_c = \frac{C}{K}$. The server shares the global model $\mathbf{w}_n$ with the selected clients. Thereafter, the selected clients $\forall k \in \mathcal{S}_n$ use a stochastic gradient descent rule [47] for minimizing the local risk $\mathcal{J}_k(\mathbf{w})$ as follows:

$$\mathbf{w}_{k,n+1} = \mathbf{w}_n + \mu\, \mathbf{z}_{k,n}\, \epsilon_{k,n}, \tag{13}$$

where $\epsilon_{k,n} = y_{k,n} - \mathbf{w}_n^{\mathrm{T}}\mathbf{z}_{k,n}$, and μ is a step size, controlling convergence rate and steady-state performance. Clients communicate their updated models, obtained via (13), to the server. By aggregating the local models received, the server then produces the global model as

$$\mathbf{w}_{n+1} = \frac{1}{C}\sum_{k\in\mathcal{S}_n} \mathbf{w}_{k,n+1}. \tag{14}$$

We notice that in the Online-Fed, as outlined above, there is no benefit of letting clients perform local learning during the periods when they are not contributing to the global update, even if new data is acquired. In particular, as soon as the server selects new clients, their most recent local model is replaced with the global model, regardless of whether more recent updates are made locally. As a result, performance is hindered. Furthermore, there is still a significant amount of communication within each global round. Our solution to

this problem stems from the partial-sharing-based communication [48], [49], which is appealing for communication-efficient distributed learning. In the following, we present novel communication-efficient online FL strategies using the concepts of partial-sharing. In contrast to sketch updates in [32], the proposed Online FL strategies do not impose any additional computational or memory demands on clients.

III. Partial-Sharing-based Online Federated Learning Strategy (PSO-Fed)

In the following, we propose a federated learning approach that reduces the communication cost by sharing a subset of all model parameters in each update-aggregation round. In every communication round, selection matrices, known by clients and the server, are used to keep track of the exchanged model parameters.

In every global iteration n, a $D \times D$ diagonal selection matrix $\mathbf{S}_{k,n}$ specifies the model parameters that will be exchanged between clients and the server. The principal diagonal of $\mathbf{S}_{k,n}$ has M ones and $D-M$ zeros. The positions of ones in $\mathbf{S}_{k,n}$ specify which parameters to be exchanged. The M model parameters can either be selected stochastically or sequentially as in [48]–[50]. In order to make the implementation simple, our approach considers coordinated and uncoordinated partial-sharing-based communication. The coordinated partial-sharing scheme assigns each client the same selection matrix (i.e., $\mathbf{S}_{1,0} = \mathbf{S}_{2,0} = \cdots = \mathbf{S}_{K,0} = \mathbf{S}_0$). This means that participating clients will share the same portion of the model at every global iteration. In contrast, the uncoordinated partial-sharing scheme assigns random initial selection matrices to clients (i.e., $\mathbf{S}_{1,0} \neq \mathbf{S}_{2,0} \neq \cdots \neq \mathbf{S}_{K,0}$). As a result, clients are not necessarily sharing the same portion of the model with the server in each communication round. A right circular shift operation on the main diagonal elements of the current entry selection matrix $\mathbf{S}_{k,n}$ generates the entry selection matrix for the next global iteration, i.e., $\text{diag}\{\mathbf{S}_{k,n+1}\} = \mathsf{circshift}(\text{diag}\{\mathbf{S}_{k,n}\}, \tau)$, where the integer τ indicates the number of positions. The $\text{diag}\{\cdot\}$ operator returns a column vector that consists of the main diagonal elements of its argument matrix. To keep track of the model parameters being shared, every client must perform this right circular shift operation on the main diagonal elements of the current entry selection matrix. With this procedure, each model parameter will be exchanged M times over D iterations. Thus, each model parameter being exchanged between clients and the server has a probability of $p_e = \frac{M}{D}$.

The Online-Fed workflow can be expressed alternatively using selection matrices as

$$\begin{aligned}\mathbf{w}_{k,n+1} &= \mathbf{w}_n + \mu\, \mathbf{z}_{k,n}\, \epsilon_{k,n} \\ &= \mathbf{S}_{k,n}\mathbf{w}_n + (\mathbf{I}_D - \mathbf{S}_{k,n})\mathbf{w}_n + \mu\, \mathbf{z}_{k,n}\, \epsilon_{k,n},\end{aligned} \tag{15a}$$

with

$$\begin{aligned}\epsilon_{k,n} &= y_{k,n} - \mathbf{w}_n^{\mathrm{T}}\mathbf{z}_{k,n} \\ &= y_{k,n} - (\mathbf{S}_{k,n}\mathbf{w}_n + (\mathbf{I}_D - \mathbf{S}_{k,n})\mathbf{w}_n)^{\mathrm{T}}\mathbf{z}_{k,n}.\end{aligned}$$

Algorithm 1: `PSO-Fed`. There are K clients with learning rate μ, set of all clients is $\mathcal{S}$, and τ is the circular shift variable,

Initialization: Initial global and local models are $\mathbf{w}_0$ and $\mathbf{w}_{k,0}$, respectively. The dimension of RFF is D and selection matrices for partial-sharing-based communication are given by $\mathbf{S}_{k,0}$, $\forall k \in \mathcal{S}$,

For $n = 1$ to N

In every global iteration n, a random subset of clients $\mathcal{S}_n$ (C clients out of K clients) is chosen by the server. Then, the server communicates $\mathbf{S}_{k,n}\mathbf{w}_n$ with the selected clients.

Client Local Update:

If $k \in \mathcal{S}_n$

$$\begin{aligned}\mathbf{w}'_{k,n} &= \mathbf{S}_{k,n}\mathbf{w}_n + (\mathbf{I}_D - \mathbf{S}_{k,n})\mathbf{w}_{k,n}, \\ \epsilon_{k,n} &= y_{k,n} - (\mathbf{w}'_{k,n})^{\mathrm{T}}\mathbf{z}_{k,n}, \\ \mathbf{w}_{k,n+1} &= \mathbf{w}'_{k,n} + \mu\, \mathbf{z}_{k,n}\, \epsilon_{k,n},\end{aligned}$$

Else

$$\begin{aligned}\epsilon_{k,n} &= y_{k,n} - \mathbf{w}_{k,n}^{\mathrm{T}}\mathbf{z}_{k,n}, \\ \mathbf{w}_{k,n+1} &= \mathbf{w}_{k,n} + \mu\, \mathbf{z}_{k,n}\, \epsilon_{k,n},\end{aligned}$$

EndIf

Every client $\forall k \in \mathcal{S}_n$ communicates $\mathbf{S}_{k,n+1}\mathbf{w}_{k,n+1}$ to the server, where $\text{diag}\{\mathbf{S}_{k,n+1}\} = \mathsf{circshift}(\text{diag}\{\mathbf{S}_{k,n}\}, \tau)$.

Aggregation at the Server:

By aggregating the local updated models, the server generated the global shared model as

$$\mathbf{w}_{n+1} = \frac{1}{C}\sum_{k\in\mathcal{S}_n} \mathbf{S}_{k,n+1}\mathbf{w}_{k,n+1} + (\mathbf{I}_D - \mathbf{S}_{k,n+1})\mathbf{w}_n.$$

EndFor

$$\begin{aligned}\mathbf{w}_{n+1} &= \frac{1}{C}\sum_{k\in\mathcal{S}_n} \mathbf{w}_{k,n+1} \\ &= \frac{1}{C}\sum_{k\in\mathcal{S}_n} \mathbf{S}_{k,n+1}\mathbf{w}_{k,n+1} + (\mathbf{I}_D - \mathbf{S}_{k,n+1})\mathbf{w}_{k,n+1}.\end{aligned} \tag{15b}$$

In the above, $\mathbf{I}_D$ is an identity matrix of size $D \times D$. Since in PSO-Fed the server and clients only exchange small portions of the entire models in each round, the remaining portions, i.e., $(\mathbf{I}_D - \mathbf{S}_{k,n})\mathbf{w}_n$ in (15a) and $(\mathbf{I}_D - \mathbf{S}_{k,n+1})\mathbf{w}_{k,n+1}$ in (15b), are unknown. These unknown portions require attention in the local updates and the subsequent aggregation. The best solution is to allow clients and server to utilize their previous model parameters instead of the unknown portions. Therefore, in the proposed PSO-Fed:

- participating clients use $(\mathbf{I}_D - \mathbf{S}_{k,n})\mathbf{w}_{k,n}$ in place of $(\mathbf{I}_D - \mathbf{S}_{k,n})\mathbf{w}_n$, and
- the server uses $(\mathbf{I}_D - \mathbf{S}_{k,n+1})\mathbf{w}_n$ in place of $(\mathbf{I}_D - \mathbf{S}_{k,n+1})\mathbf{w}_{k,n+1}$.

By communicating a portion of the model in each global iteration, the proposed PSO-Fed achieves better communication-efficiency over Online-Fed (since M is much smaller than D) and the reduction in communication overhead is $(\frac{D-M}{D})\%$ during every global iteration n. Aside from enabling efficient communication between clients and the server, PSO-Fed also enables greater control over the local learning. To this end, PSO-Fed allows clients who do not participate in the global iterations to perform local learning whenever they acquire new data. It is worth noting that the participating clients perform only partial-sharing-based information exchange when they do not have access to new data. Algorithm 1 presents a summary of the proposed PSO-Fed.

It is also important to note that, as mentioned above, state-of-the-art approaches replace local models with the global shared model whenever clients get a chance to contribute to the global shared model update, which makes local updates futile during communication-dormant times. Additionally, state-of-the-art techniques require a server-based attack detection mechanism if a few clients with malicious intentions attempt to poison the global shared model. However, model-poisoning cannot spread as quickly with PSO-Fed as with conventional FL strategies due to partial-sharing-based communication. As a result, the proposed PSO-Fed is more robust to Byzantine attacks without requiring additional mechanisms.

IV. Performance Analysis

Throughout this section, we examine the convergence behavior of the proposed PSO-Fed algorithm. In particular, we want to study the impact of partial-sharing-based communication on the convergence behavior of the proposed PSO-Fed. As a preliminary to the analysis, we define the following expanded parameter vectors: global optimal extended model parameter vector $\mathbf{w}_e^\star$, extended estimated global model parameter vector $\mathbf{w}_{e,n}$, extended input data matrix $\mathbf{Z}_{e,n}$ and extended observation noise vector $\boldsymbol{\nu}_{e,n}$ as follows:

$$\begin{aligned}\mathbf{w}_e^\star &= \mathbf{1}_{K+1}\otimes\mathbf{w}^\star,\\ \mathbf{w}_{e,n} &= \mathsf{col}\{\mathbf{w}_n,\mathbf{w}_{1,n},\mathbf{w}_{2,n},\ldots,\mathbf{w}_{K,n}\},\\ \mathbf{Z}_{e,n} &= \mathsf{blockdiag}\{\mathbf{0},\mathbf{z}_{1,n},\mathbf{z}_{2,n},\ldots,\mathbf{z}_{K,n}\},\\ \boldsymbol{\nu}_{e,n} &= \mathsf{col}\big\{0,\nu_{1,n},\nu_{2,n},\ldots,\nu_{K,n}\big\},\end{aligned} \tag{16}$$

where $\mathsf{col}\{\cdot\}$ is the column-wise stacked and $\mathsf{blockdiag}\{\cdot\}$ is the block diagonalized operator. The symbol $\mathbf{1}_{K+1}$ is a $(K+1)$-dimensional column vector, where each element has the value one. These definitions lead us to write the global expanded desired output vector and extended observation noise vector as

$$\begin{aligned}\mathbf{y}_{e,n} &= \mathsf{col}\{0,y_{1,n},y_{2,n},\ldots,y_{K,n}\} = \mathbf{Z}_{e,n}^{\mathrm{T}}\mathbf{w}_e^\star+\boldsymbol{\nu}_{e,n},\\ \boldsymbol{\epsilon}_{e,n} &= \mathsf{col}\big\{0,\epsilon_{1,n},\epsilon_{2,n},\ldots,\epsilon_{K,n}\big\} = \mathbf{y}_{e,n}-\mathbf{Z}_{e,n}^{\mathrm{T}}\boldsymbol{\mathcal{A}}_{\mathbf{S},n}\mathbf{w}_{e,n},\end{aligned} \tag{17}$$

with

$$\boldsymbol{\mathcal{A}}_{\mathbf{S},n} = \begin{bmatrix}\mathbf{I}_D & \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0}\\ a_{1,n}\mathbf{S}_{1,n} & \mathbf{I}_D-a_{1,n}\mathbf{S}_{1,n} & \mathbf{0} & \ldots & \mathbf{0}\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ a_{K,n}\mathbf{S}_{K,n} & \mathbf{0} & \mathbf{0} & \ldots & \mathbf{I}_D-a_{K,n}\mathbf{S}_{K,n}\end{bmatrix}, \tag{18}$$

where $a_{k,n}$ takes the value 1 if the client k is chosen in the current global iteration (i.e., $k\in\mathcal{S}_n$) and zero otherwise. Based on these definitions, the global recursion of PSO-Fed can be expressed as follows:

$$\mathbf{w}_{e,n+1} = \boldsymbol{\mathcal{B}}_{\mathbf{S},n+1}\big(\boldsymbol{\mathcal{A}}_{\mathbf{S},n}\mathbf{w}_{e,n}+\mu\,\mathbf{Z}_{e,n}\,\boldsymbol{\epsilon}_{e,n}\big), \tag{19}$$

where

$$\boldsymbol{\mathcal{B}}_{\mathbf{S},n+1} = \begin{bmatrix}\mathbf{I}_D-\sum\limits_{k\in\mathcal{S}_n}\frac{a_{k,n}}{C}\mathbf{S}_{k,n+1} & \frac{a_{1,n}}{C}\mathbf{S}_{1,n+1} & \ldots & \frac{a_{K,n}}{C}\mathbf{S}_{K,n+1}\\ \mathbf{0} & \mathbf{I}_D & \ldots & \mathbf{0}\\ \vdots & \vdots & \ddots & \vdots\\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{I}_D\end{bmatrix}, \tag{20}$$

In the following, we investigate the mean and mean-square convergence behavior of the proposed PSO-Fed described in (19). To establish the conditions for the convergence of PSO-Fed, we assume the following:

A1: The sequence of local input vectors, $\mathbf{z}_{k,n}$, is modeled as a weakly stationary multivariate random sequence with correlation matrix $\mathbf{R}_k=\mathbb{E}[\mathbf{z}_{k,n}\mathbf{z}_{k,n}^{\mathrm{T}}]$.

A2: The observation noise $\nu_{k,n}$ is taken to be a white process, independent of any other data.

A3: The selection matrices $\mathbf{S}_{k,n}$ are taken to be statistically independent of any other data. Furthermore, $\mathbf{S}_{k,n}$ and $\mathbf{S}_{l,m}$ are assumed independent, for all $k\neq l$ and $m\neq n$.

A4: The higher-order terms of the learning rate μ can be ignored for sufficiently small μ.

A. Mean Convergence Analysis

Denoting $\widetilde{\mathbf{w}}_{e,n}=\mathbf{w}_e^\star-\mathbf{w}_{e,n}$, and recalling the fact that $\mathbf{w}_e^\star=\boldsymbol{\mathcal{B}}_{\mathbf{S},n+1}\boldsymbol{\mathcal{A}}_{\mathbf{S},n}\mathbf{w}_e^\star$ (from (18) and (20), one can see that the row sum of $\boldsymbol{\mathcal{A}}_{\mathbf{S},n}$ and $\boldsymbol{\mathcal{B}}_{\mathbf{S},n+1}$ is equals to 1, so we have $\mathbf{w}_e^\star=\boldsymbol{\mathcal{A}}_{\mathbf{S},n}\mathbf{w}_e^\star$ and $\mathbf{w}_e^\star=\boldsymbol{\mathcal{B}}_{\mathbf{S},n+1}\mathbf{w}_e^\star$), then from (19), the recursive expression for $\widetilde{\mathbf{w}}_{e,n+1}$ is

$$\begin{aligned}\widetilde{\mathbf{w}}_{e,n+1} =&\boldsymbol{\mathcal{B}}_{\mathbf{S},n+1}\big(\mathbf{I}-\mu\mathbf{Z}_{e,n}\mathbf{Z}_{e,n}^{\mathrm{T}}\big)\boldsymbol{\mathcal{A}}_{\mathbf{S},n}\widetilde{\mathbf{w}}_{e,n}\\ &-\mu\boldsymbol{\mathcal{B}}_{\mathbf{S},n+1}\mathbf{Z}_{e,n}\boldsymbol{\nu}_{e,n},\end{aligned} \tag{21}$$

where $\mathbf{I}$ represents the identity matrix of appropriate size.

Theorem 1: Let **A1**-**A3** hold true. Then, the condition for the mean convergence of proposed PSO-Fed is

$$0<\mu<\frac{2}{\max\limits_{\forall k,i}\{\lambda_i(\mathbf{R}_k)\}}, \tag{22}$$

where $\lambda_i(\cdot)$ is the ith eigenvalue of matrix $\mathbf{R}_k$.

Proof: Taking the expectation of (21), and adopting **A1**–**A3**, yields

$$\mathbb{E}[\widetilde{\mathbf{w}}_{e,n+1}] = \overline{\boldsymbol{\mathcal{B}}}_{\mathbf{S}}\big(\mathbf{I} - \mu\boldsymbol{\mathcal{R}}_e\big)\overline{\boldsymbol{\mathcal{A}}}_{\mathbf{S}}\mathbb{E}[\widetilde{\mathbf{w}}_{e,n}], \tag{23}$$

where $\boldsymbol{\mathcal{R}}_e = \mathsf{blockdiag}\{\mathbf{0}, \mathbf{R}_1, \mathbf{R}_2, \ldots, \mathbf{R}_K\}$, $\overline{\boldsymbol{\mathcal{B}}}_{\mathbf{S}} = \mathbb{E}[\boldsymbol{\mathcal{B}}_{\mathbf{S},n+1}]$ and $\overline{\boldsymbol{\mathcal{A}}}_{\mathbf{S}} = \mathbb{E}[\boldsymbol{\mathcal{A}}_{\mathbf{S},n}]$. In Appendix A, we evaluate the quantities $\overline{\boldsymbol{\mathcal{A}}}_{\mathbf{S}}$ and $\overline{\boldsymbol{\mathcal{B}}}_{\mathbf{S}}$ and show that all entries in these matrices are real, non-negative and their row sum is unity. Thus, both of these matrices are right-stochastic matrices. As a result, the spectral radius of these matrices is unity [51]. From (23), one can see that $\mathbb{E}\big[\widetilde{\mathbf{w}}_{e,n}\big]$ converges if and only if $\|\overline{\boldsymbol{\mathcal{B}}}_{\mathbf{S}}\big(\mathbf{I} - \mu\boldsymbol{\mathcal{R}}_e\big)\overline{\boldsymbol{\mathcal{A}}}_{\mathbf{S}}\| < 1$ for every n, where $\|\cdot\|$ is any matrix norm. To obtain the mean convergence condition, we employ the block maximum norm (i.e., $\|\cdot\|_{b,\infty}$) [52]. Since $\overline{\boldsymbol{\mathcal{A}}}_{\mathbf{S}}$ and $\overline{\boldsymbol{\mathcal{B}}}_{\mathbf{S}}$ are right stochastic matrices, we have $\|\overline{\boldsymbol{\mathcal{A}}}_{\mathbf{S}}]\|_{b,\infty} = 1$ and $\|\overline{\boldsymbol{\mathcal{B}}}_{\mathbf{S}}\|_{b,\infty} = 1$. Hence, the convergence condition becomes $\|\mathbf{I} - \mu\boldsymbol{\mathcal{R}}_e\|_{b,\infty} < 1$, or, $\forall k, i : |1 - \mu\lambda_i(\mathbf{R}_k)| < 1$, where $\lambda_i(\cdot)$ is the ith eigenvalue of matrix $\mathbf{R}_k$. Upon solving the convergence condition above, we reach (22). □

Under (22), the proposed PSO-Fed converges in mean and is also asymptotically unbiased in the RFF space.

B. Mean-Square Convergence Analysis

Defining the weighted norm-square of $\widetilde{\mathbf{w}}_{e,n}$ as $\|\widetilde{\mathbf{w}}_{e,n}\|^2_{\boldsymbol{\Sigma}} = \widetilde{\mathbf{w}}^{\mathrm{T}}_{e,n}\boldsymbol{\Sigma}\widetilde{\mathbf{w}}_{e,n}$, where $\boldsymbol{\Sigma}$ is an arbitrary positive semi-definite matrix, then from (21), we have the following weighted variance relation:

$$\mathbb{E}[\|\widetilde{\mathbf{w}}_{e,n+1}\|^2_{\boldsymbol{\Sigma}}] = \mathbb{E}[\|\widetilde{\mathbf{w}}_{e,n}\|^2_{\boldsymbol{\Sigma}'}] + \mu^2\mathbb{E}[\boldsymbol{\nu}^{\mathrm{T}}_{e,n}\mathbf{Y}^{\boldsymbol{\Sigma}}_n\boldsymbol{\nu}_{e,n}], \tag{24}$$

where the cross terms become zero under the assumption **A2**. The matrix $\boldsymbol{\Sigma}'$ is given by

$$\begin{aligned}\boldsymbol{\Sigma}' = \mathbb{E}\big[&\boldsymbol{\mathcal{A}}^{\mathrm{T}}_{\boldsymbol{\mathcal{S}},n}\big(\mathbf{I} - \mu\mathbf{Z}_{e,n}\mathbf{Z}^{\mathrm{T}}_{e,n}\big)\boldsymbol{\mathcal{B}}^{\mathrm{T}}_{\boldsymbol{\mathcal{S}},n+1}\\ &\times\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1}\big(\mathbf{I} - \mu\mathbf{Z}_{e,n}\mathbf{Z}^{\mathrm{T}}_{e,n}\big)\boldsymbol{\mathcal{A}}_{\boldsymbol{\mathcal{S}},n}\big],\end{aligned} \tag{25}$$

and

$$\mathbf{Y}^{\boldsymbol{\Sigma}}_n = \mathbf{Z}^{\mathrm{T}}_{e,n}\boldsymbol{\mathcal{B}}^{\mathrm{T}}_{\boldsymbol{\mathcal{S}},n+1}\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1}\mathbf{Z}_{e,n}. \tag{26}$$

Using **A3**, and from the properties of block vectorization operator $\mathrm{bvec}\{\cdot\}$ and block Kronecker product [53], the relation between $\boldsymbol{\sigma} = \mathrm{bvec}\{\boldsymbol{\Sigma}\}$ and $\boldsymbol{\sigma}' = \mathrm{bvec}\{\boldsymbol{\Sigma}'\}$ can be obtained as

$$\begin{aligned}\boldsymbol{\sigma}' &= \mathrm{bvec}\Big\{\mathbb{E}\big[\boldsymbol{\mathcal{A}}^{\mathrm{T}}_{\boldsymbol{\mathcal{S}},n}\big(\mathbf{I} - \mu\mathbf{Z}_{e,n}\mathbf{Z}^{\mathrm{T}}_{e,n}\big)\boldsymbol{\mathcal{B}}^{\mathrm{T}}_{\boldsymbol{\mathcal{S}},n+1}\\ &\qquad\times\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1}\big(\mathbf{I} - \mu\mathbf{Z}_{e,n}\mathbf{Z}^{\mathrm{T}}_{e,n}\big)\boldsymbol{\mathcal{A}}_{\boldsymbol{\mathcal{S}},n}\big]\Big\}\\ &= \boldsymbol{\mathcal{F}}^{\mathrm{T}}\boldsymbol{\sigma},\end{aligned} \tag{27}$$

where

$$\boldsymbol{\mathcal{F}} = \boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{B}}}\boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{A}}} - \mu\boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{B}}}(\mathbf{I}\otimes_b\boldsymbol{\mathcal{R}}_e)\boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{A}}} - \mu\boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{B}}}(\boldsymbol{\mathcal{R}}_e\otimes_b\mathbf{I})\boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{A}}}, \tag{28}$$

with

$$\begin{aligned}\boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{A}}} &= \mathbb{E}[\boldsymbol{\mathcal{A}}_{\boldsymbol{\mathcal{S}},n}\otimes_b\boldsymbol{\mathcal{A}}_{\boldsymbol{\mathcal{S}},n}],\\ \boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{B}}} &= \mathbb{E}[\boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1}\otimes_b\boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1}].\end{aligned} \tag{29}$$

Under **A4**, the higher-order powers of μ are neglected in (28). In the following, we proceed with this approximation. In Appendix B, we evaluate the quantities $\boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{A}}}$ and $\boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{B}}}$ and show that all entries in these matrices are real, non-negative and their row sum is unity. This implies that both these matrices are right-stochastic matrices. Hence, their spectral radius is equal to one.

Next, we evaluate the second term in the RHS of (24). We can write $\mathbb{E}[\boldsymbol{\nu}^{\mathrm{T}}_{e,n}\mathbf{Y}^{\boldsymbol{\Sigma}}_n\boldsymbol{\nu}_{e,n}] = \mathbb{E}[\boldsymbol{\nu}^{\mathrm{T}}_{e,n}\mathbf{Z}^{\mathrm{T}}_{e,n}\boldsymbol{\mathcal{B}}^{\mathrm{T}}_{\boldsymbol{\mathcal{S}},n+1}\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1}\mathbf{Z}_{e,n}\boldsymbol{\nu}_{e,n}] = \mathbb{E}\big[\mathrm{trace}(\boldsymbol{\nu}^{\mathrm{T}}_{e,n}\mathbf{Z}^{\mathrm{T}}_{e,n}\boldsymbol{\mathcal{B}}^{\mathrm{T}}_{\boldsymbol{\mathcal{S}},n+1}\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1}\mathbf{Z}_{e,n}\boldsymbol{\nu}_{e,n})\big] = \mathrm{trace}\big(\mathbb{E}[\boldsymbol{\nu}^{\mathrm{T}}_{e,n}\mathbf{Z}^{\mathrm{T}}_{e,n}\boldsymbol{\mathcal{B}}^{\mathrm{T}}_{\boldsymbol{\mathcal{S}},n+1}\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1}\mathbf{Z}_{e,n}\boldsymbol{\nu}_{e,n}]\big) = \mathrm{trace}\big(\mathbb{E}[\boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1}\mathbf{Z}_{e,n}\mathbb{E}[\boldsymbol{\nu}^{\mathrm{T}}_{e,n}\boldsymbol{\nu}_{e,n}]\mathbf{Z}^{\mathrm{T}}_{e,n}\boldsymbol{\mathcal{B}}^{\mathrm{T}}_{\boldsymbol{\mathcal{S}},n+1}]\boldsymbol{\Sigma}\big)$ (trace$(\cdot)$ represents the trace of an argument matrix). Under **A2**, one can write

$$\begin{aligned}&\mathrm{trace}\big(\mathbb{E}[\boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1}\mathbf{Z}_{e,n}\mathbb{E}[\boldsymbol{\nu}^{\mathrm{T}}_{e,n}\boldsymbol{\nu}_{e,n}]\mathbf{Z}^{\mathrm{T}}_{e,n}\boldsymbol{\mathcal{B}}^{\mathrm{T}}_{\boldsymbol{\mathcal{S}},n+1}]\boldsymbol{\Sigma}\big)\\ &\qquad= \mathrm{trace}\big(\mathbb{E}[\boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1}\boldsymbol{\Phi}_n\boldsymbol{\mathcal{B}}^{\mathrm{T}}_{\boldsymbol{\mathcal{S}},n+1}]\boldsymbol{\Sigma}\big),\end{aligned} \tag{30}$$

where $\boldsymbol{\Phi}_n = \mathbf{Z}_{e,n}\boldsymbol{\Lambda}_\nu\mathbf{Z}^{\mathrm{T}}_{e,n}$, with $\boldsymbol{\Lambda}_\nu = \mathbb{E}[\boldsymbol{\nu}^{\mathrm{T}}_{e,n}\boldsymbol{\nu}_{e,n}] = \mathsf{diag}\{\mathbf{0}, \sigma^2_{\nu,1}, \sigma^2_{\nu,2}, \cdots, \sigma^2_{\nu,K}\}$, is a diagonal matrix. Using the block Kronecker product properties, we finally have

$$\mathrm{trace}\big(\mathbb{E}[\boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1}\boldsymbol{\Phi}_n\boldsymbol{\mathcal{B}}^{\mathrm{T}}_{\boldsymbol{\mathcal{S}},n+1}]\boldsymbol{\Sigma}\big) = \boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{\sigma}, \tag{31}$$

where

$$\begin{aligned}\boldsymbol{\beta} &= \mathrm{bvec}\{\mathbb{E}[\boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1}\boldsymbol{\Phi}_n\boldsymbol{\mathcal{B}}^{\mathrm{T}}_{\boldsymbol{\mathcal{S}},n+1}]\}\\ &= \boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{B}}}\,\boldsymbol{\beta}_\nu,\end{aligned} \tag{32}$$

with $\boldsymbol{\beta}_\upsilon = \mathrm{bvec}\{\mathbb{E}[\boldsymbol{\Phi}_n]\} = \mathrm{bvec}\{\mathbb{E}[\mathbf{Z}_{e,n}\boldsymbol{\Lambda}_\nu\mathbf{Z}^{\mathrm{T}}_{e,n})]\}$.

Utilizing all these results together, the recursion for the weighted extended MSD of the proposed PSO-Fed can be stated as

$$\mathbb{E}[\|\widetilde{\mathbf{w}}_{e,n+1}\|^2_{\mathsf{bvec}^{-1}\{\boldsymbol{\sigma}\}}] = \mathbb{E}[\|\widetilde{\mathbf{w}}_{e,n}\|^2_{\mathsf{bvec}^{-1}\{\boldsymbol{\mathcal{F}}^{\mathrm{T}}\boldsymbol{\sigma}\}}] + \mu^2\boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{\sigma}, \tag{33}$$

where $\mathsf{bvec}^{-1}\{\cdot\}$ represents the reverse operation of block vectorization.

Theorem 2: Let **A1**-**A4** hold true and (33) represents the dynamics of weighted extended MSD. Then, the proposed PSO-Fed exhibits stable MSD under

$$0 < \mu < \frac{1}{\max\limits_{\forall k,i}\{\lambda_i(\mathbf{R}_k)\}}. \tag{34}$$

Proof: Iterating the recursion (33), backwards down to $n = 0$, we have

$$\begin{aligned}\mathbb{E}[\|\widetilde{\mathbf{w}}_{e,n+1}\|^2_{\mathsf{bvec}^{-1}\{\boldsymbol{\sigma}\}}] &= \mathbb{E}[\|\widetilde{\mathbf{w}}_{e,0}\|^2_{\mathsf{bvec}^{-1}\{(\boldsymbol{\mathcal{F}}^{\mathrm{T}})^{n+1}\ \boldsymbol{\sigma}\}}]\\ &\quad+ \mu^2\ \boldsymbol{\beta}^{\mathrm{T}}\left(\mathbf{I} + \sum_{j=1}^{n}(\boldsymbol{\mathcal{F}}^{\mathrm{T}})^j\right)\boldsymbol{\sigma},\end{aligned} \tag{35}$$

where $\widetilde{\mathbf{w}}_{e,0} = \mathbf{w}^\star_e - \mathbf{w}_{e,0}$. For the convergence of $\mathbb{E}[\|\widetilde{\mathbf{w}}_{e,n}\|^2_{\boldsymbol{\Sigma}}] = E[\|\widetilde{\mathbf{w}}_{e,n}\|^2_{\mathsf{bvec}^{-1}\{\boldsymbol{\sigma}\}}]$, the spectral radius of $\boldsymbol{\mathcal{F}}$ must be less than one, i.e., $\rho\big(\boldsymbol{\mathcal{F}}\big) < 1$. From the properties of block maximum norm, we can write

$$\begin{aligned}&\rho\big(\boldsymbol{\mathcal{F}}\big)\\ &\le \big\|\boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{B}}}\big(\mathbf{I} - \mu(\mathbf{I}\otimes_b\boldsymbol{\mathcal{R}}_e) - \mu(\boldsymbol{\mathcal{R}}_e\otimes_b\mathbf{I})\big)\boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{A}}}\big\|_{b,\infty}\\ &\le \|\boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{B}}}\|_{b,\infty}\|\mathbf{I} - \mu(\mathbf{I}\otimes_b\boldsymbol{\mathcal{R}}_e) - \mu(\boldsymbol{\mathcal{R}}_e\otimes_b\mathbf{I})\|_{b,\infty}\|\boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{A}}}\|_{b,\infty},\end{aligned} \tag{36}$$

Since the matrices $\boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{A}}}$ and $\boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{B}}}$ are right stochastic, their block maximum norm is unity, i.e., $\|\boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{A}}}\|_{b,\infty} =$

$\|\mathcal{Q}_{\mathcal{B}}\|_{b,\infty} = 1$. Therefore, the requirement for the convergence of $E[\|\widetilde{\mathbf{w}}_{e,n}\|^2_{\mathbf{\Sigma}}]$ is $\|\mathbf{I} - \mu(\mathbf{I} \otimes_b \mathcal{R}_e) - \mu(\mathcal{R}_e \otimes_b \mathbf{I})\|_{b,\infty} < 1$, or, equivalently, $|1 - \mu(\lambda_i(\mathcal{R}_e) + \lambda_j(\mathcal{R}_e))| < 1$, $i, j = 1, 2, \cdots, D(K+1)$. Finally, the convergence condition becomes $0 < \mu < \frac{1}{\max\limits_{i=1,\cdots,D(K+1)} \lambda_i(\mathcal{R}_e)}$ that proves (34). □

C. Transient and Steady-State Mean Square Deviation

From (33), we can relate $\mathbb{E}[\|\widetilde{\mathbf{w}}_{e,n+1}\|^2_{\text{bvec}^{-1}\{\boldsymbol{\sigma}\}}]$ and $\mathbb{E}[\|\widetilde{\mathbf{w}}_{e,n}\|^2_{\text{bvec}^{-1}\{\boldsymbol{\sigma}\}}]$ as

$$\begin{aligned}\mathbb{E}[\|\widetilde{\mathbf{w}}_{e,n+1}\|^2_{\text{bvec}^{-1}\{\boldsymbol{\sigma}\}}] = \mathbb{E}[\|\widetilde{\mathbf{w}}_{e,n}\|^2_{\text{bvec}^{-1}\{\boldsymbol{\sigma}\}}] \\ + \mu^2 \boldsymbol{\beta}^{\mathrm{T}} (\boldsymbol{\mathcal{F}}^{\mathrm{T}})^n \, \boldsymbol{\sigma} \\ - \mathbb{E}\big[\|\widetilde{\mathbf{w}}_{e,0}\|^2_{\text{bvec}^{-1}\{(\mathbf{I}-\boldsymbol{\mathcal{F}}^{\mathrm{T}})(\boldsymbol{\mathcal{F}}^{\mathrm{T}})^n \boldsymbol{\sigma}\}}\big].\end{aligned} \tag{37}$$

By selecting $\boldsymbol{\sigma} = \text{bvec}\{\text{blockdiag}\{\mathbf{I}_D, \mathbf{0}, \ldots, \mathbf{0}\}\}$, the mean square deviation of the global estimate at time index n: $\zeta_n = \mathbb{E}[\|\widetilde{\mathbf{w}}_n\|^2] = \mathbb{E}[\|\widetilde{\mathbf{w}}_{e,n}\|^2_{\text{bvec}^{-1}\{\boldsymbol{\sigma}\}}]$ can be obtained.

Under (34), letting $n \to \infty$ on both sides of (33), we obtain

$$\lim_{n\to\infty} \mathbb{E}[\|\widetilde{\mathbf{w}}_{e,n}\|^2_{\text{bvec}^{-1}\{(\mathbf{I}-\boldsymbol{\mathcal{F}}^{\mathrm{T}})\boldsymbol{\sigma}\}}] = \mu^2 \boldsymbol{\beta}^{\mathrm{T}} \boldsymbol{\sigma}. \tag{38}$$

By substituting $\boldsymbol{\sigma} = (\mathbf{I} - \boldsymbol{\mathcal{F}}^{\mathrm{T}})^{-1}\text{bvec}\{\text{blockdiag}\{\mathbf{I}_D, \mathbf{0}, \ldots, \mathbf{0}\}\}$ in (38), the steady-state MSD at the global server can be obtained.

D. ETPSO-Fed

Client local models will hardly change if the newly arrived data is not innovative (i.e., magnitude of the estimation error at client k is less than the predefined threshold). Therefore, updating the local client model and communicating the unchanged model to the server would waste resources. Despite reducing the communication cost between clients and the server, PSO-Fed is agnostic to the innovation of the data arriving at clients. So PSO-Fed forces participating clients to update local models regardless of the benefits of such updates, and then it communicates the updated model to the server. In order to avoid unnecessary processing and communication, it is essential to use data effectively, update parameter estimates and communicate those only when it is beneficial to do so. Below, we consider an FL approach, based on the set-membership filtering (SMF) principles [54], [55], that features data-dependent innovation-triggered updates of parameter estimates at each device. This feature makes naturally embedded selective communication possible among devices and the server. That is, in our context, the selective update feature of SMF algorithms not only reduces local computational complexity for the devices but also provides a systematic mechanism for further reduction of the communication overhead.

The proposed event-triggered PSO-Fed (ETPSO-Fed) scheme implements the partial-sharing strategy and employs the SM-NLMS algorithm [54] in the local adaptation step. In particular, during a global iteration n, each participating client updates the local model if and only if the magnitude of the learning error $\epsilon_{k,n}$ is greater than a presumed bound γ. Otherwise, clients do not need to update and communicate the local model. In this way, the proposed ETPSO-Fed reduces the communication requirements from clients to the server. The recursive formulas for the ETPSO-Fed implementation are as follows:

$$\begin{aligned}\epsilon_{k,n} &= y_{k,n} - (\mathbf{S}_{k,n}\mathbf{w}_n + (\mathbf{I}_D - \mathbf{S}_{k,n})\mathbf{w}_n)^{\mathrm{T}}\mathbf{z}_{k,n}, \\ \mathbf{w}_{k,n+1} &= (\mathbf{S}_{k,n}\mathbf{w}_n + (\mathbf{I}_D - \mathbf{S}_{k,n})\mathbf{w}_n) + \alpha_{k,n}\epsilon_{k,n}\mathbf{z}_{k,n},\end{aligned} \tag{39}$$

Algorithm 2: `ETPSO-Fed`. There are K clients with learning rate μ, set of all clients is $\mathcal{S}$, τ is the circular shift variable, and error threshold γ.

Initialization: Initial global and local models are $\mathbf{w}_0$ and $\mathbf{w}_{k,0}$, respectively. The dimension of RFF is D and selection matrices for partial-sharing-based communication are given by $\mathbf{S}_{k,0}$, $\forall k \in \mathcal{S}$,

For $n = 1$ to N

In every global iteration n, a random subset of clients $\mathcal{S}_n$ (C clients out of K clients) is chosen by the server. Then, the server communicates $\mathbf{S}_{k,n}\mathbf{w}_n$ with the selected clients.

Client Local Update:

If $k \in \mathcal{S}_n$

$$\mathbf{w}'_{k,n} = (\mathbf{S}_{k,,n}\mathbf{w}_n + (\mathbf{I}_D - \mathbf{S}_{k,n})\mathbf{w}_{k,n},$$
$$\epsilon_{k,n} = y_{k,n} - (\mathbf{w}'_{k,n})^{\mathrm{T}}\mathbf{z}_{k,n},$$

If $|\epsilon_{k,n}| > \gamma$

$$\mathbf{w}_{k,n+1} = \mathbf{w}'_{k,n} + (1 - \frac{\gamma}{|\epsilon_{k,n}|})\, \mathbf{z}_{k,n}\, \epsilon_{k,n},$$

Else

$$\mathbf{w}_{k,n+1} = \mathbf{w}'_{k,n}$$

EndIf

Else

$$\epsilon_{k,n} = y_{k,n} - \mathbf{w}^{\mathrm{T}}_{k,n}\mathbf{z}_{k,n},$$

If $|\epsilon_{k,n}| > \gamma$

$$\mathbf{w}_{k,n+1} = \mathbf{w}_{k,n} + (1 - \frac{\gamma}{|\epsilon_{k,n}|})\, \mathbf{z}_{k,n}\, \epsilon_{k,n},$$

Else

$$\mathbf{w}_{k,n+1} = \mathbf{w}_{k,n}$$

EndIf

EndIf

Clients $\forall k \in \mathcal{S}'_n$, i.e., the clients having $|\epsilon_{k,n}| > \gamma$ (with $C' = |\mathcal{S}'_n| \leq C$), share $\mathbf{S}_{k,n+1}\mathbf{w}_{k,n+1}$ with the server, where
$\text{diag}\{\mathbf{S}_{k,n+1}\} = \text{circshift}(\text{diag}\{\mathbf{S}_{k,n}\}, \tau)$.

aggregation at the Server:

If $C' > 0$

$$\mathbf{w}_{n+1} = \frac{1}{C'} \sum_{k\in\mathcal{S}'_n} \mathbf{S}_{k,n+1}\mathbf{w}_{k,n+1} + (\mathbf{I}_D - \mathbf{S}_{k,n+1})\mathbf{w}_n,$$

Else

$$\mathbf{w}_{n+1} = \mathbf{w}_n.$$

EndIf

EndFor

where variable step size $\alpha_{k,n}$ is *optimal* in the sense of "principle of minimum disturbance", and in the sense of "optimal bounding spheroids" [54], and is given by

$$\alpha_{k,n} = \begin{cases} 1 - \frac{\gamma}{|\epsilon_{k,n}|}, & \text{if } |\epsilon_{k,n}| > \gamma \\ 0, & \text{otherwise.} \end{cases} \quad (40)$$

Equation (40) is termed innovation check, i.e., only when the prediction error magnitude of the current data pair with the previous parameter estimate exceeds the error bound, the data is considered innovative, and an update on the parameter estimate is performed. Otherwise, $\alpha_{k,n} = 0$, and no update is needed. Furthermore, non-participating clients can also use SMF principles for local learning when they have access to new data. Using innovation checks, those clients can avoid unnecessary computational and communication burdens. Therefore, by integrating the concepts of partial-sharing and SMF, it is clear that ETPSO-Fed has the potential to achieve higher communication savings than PSO-Fed by utilizing the innovation check along with the partial-sharing-based communication. Compared to Online-Fed and PSO-Fed, ETPSO-Fed offers $\frac{2CD-(C+C')M}{2CD}\%$ and $\frac{2CM-(C+C')M}{2CM}\%$ reduction in communication overhead during each global iteration n. In general, the communication from clients to the server (i.e., uplink) is more resource-intensive than the communication from the server to clients (i.e., downlink). Thus, it is important to note that when only uplink communication is considered, ETPSO-Fed further offers $\frac{C-C'}{C}\%$ reduction in communication overhead compared to PSO-Fed during each global iteration n. Here C' denotes the number of clients fulfilling the innovation check. The proposed ETPSO-Fed is summarized in Algorithm 2.

V. Experimental Results

Experiments are conducted in this section for evaluating the performance of PSO-Fed and ETPSO-Fed. In all experiments, we considered a scenario in which 100 clients are connected to a server. Every client k has access to a synthetic non-IID input signal $x_{k,n}$ and corresponding observed output $y_{k,n}$ that are assumed to be related as

$$f(\mathbf{x}_{k,n}) = \sqrt{x_{k,1,n}^2 + \sin^2(\pi\, x_{k,4,n})} + \big(0.8 - 0.5\exp(-x_{k,2,n}^2)\big)x_{k,3,n} + \nu_{k,n}. \quad (41)$$

At each client k, a first-order autoregressive (AR) model was employed to produce the non-IID input signal $x_{k,n}$: $x_{k,n} = \theta_k\, x_{k,n-1} + \sqrt{1-\theta_k^2}\, u_{k,n}$, $\theta_k \in \mathcal{U}(0.2, 0.9)$, where $u_{k,n} \in \mathcal{N}(\mu_k, \sigma_{u_k}^2)$, with $\mu_k \in \mathcal{U}(-0.2, 0.2)$ and $\sigma_{u_k}^2 \in \mathcal{U}(0.2, 1.2)$. The observation noise $\nu_{k,n}$ was assumed to be white Gaussian with a variance of $\sigma_{\nu_k}^2 \in \mathcal{U}(0.005, 0.03)$. The cosine feature function was used to map $x_{k,n}$ into the 200-dimensional RFF space. Every simulated algorithm was set to the same learning rate of 0.75. In every global iteration n, a uniform random selection procedure was implemented by the server for selecting $C = |\mathcal{S}_n| = 4$ clients. We evaluated the simulation performance by computing the average mean-square-error (MSE) of the test data, i.e.,

$$\text{Testing MSE} = \frac{1}{N_{\text{test}}} \|\mathbf{y}_{\text{test}} - \mathbf{Z}_{\text{test}}^{\text{T}} \mathbf{w}_n\|_2^2, \quad (42)$$

where $\{\mathbf{Z}_{\text{test}}, \mathbf{y}_{\text{test}}\}$ is the test data set (N_{test} examples in total) that contains possible examples of every client. To carry out the task of kernel regression, we simulated the proposed PSO-Fed for several sizes of the shared fraction of the model M. We also simulated SignSGD (adopted to the aforementioned online FL scenario) and Online-Fed for comparison purposes. The resulting learning curves (i.e., testing MSE in dB vss the global iteration index n), averaged over 500 independent experiments, are presented in Figs. 1a-1b for both coordinated and uncoordinated partial-sharing-based online FL schemes.

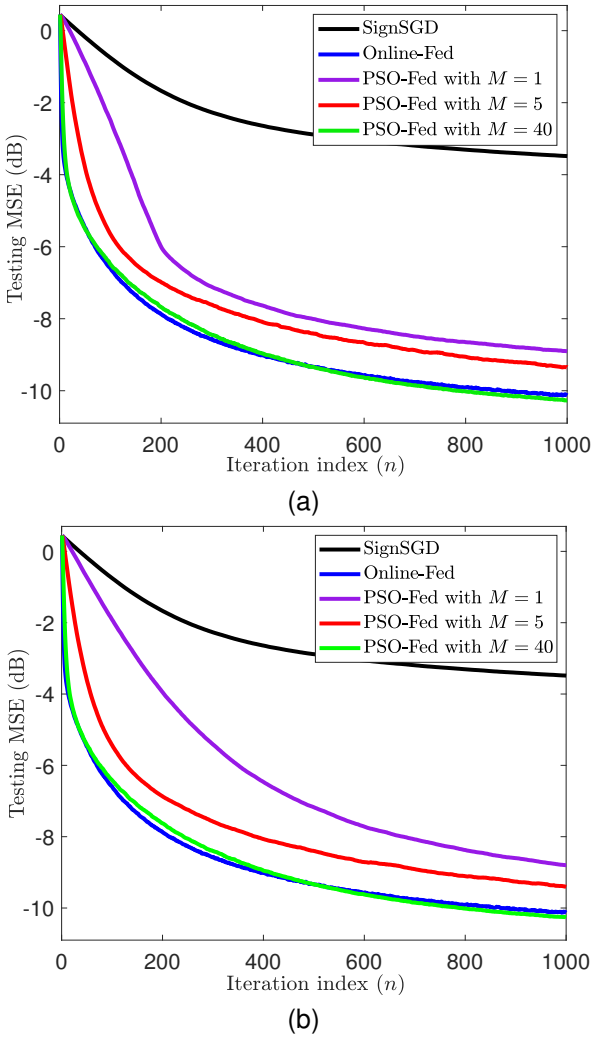

Fig. 1. Learning curves of the proposed PSO-Fed: (a). Coordinated partial-sharing scheme (b). Uncoordinated partial-sharing scheme.

Fig. 1 provides some interesting observations:

1) In comparison with Online-Fed, PSO-Fed offers competitive results at a lower cost of communication. PSO-Fed shows slower convergence with smaller values of M (e.g., 1) than Online-Fed, but with a similar steady-state MSE. As the M value increases (e.g., 5 and 40), its convergence becomes faster. Overall, PSO-Fed shows a similar convergence speed when $M \geq 40$.
2) The communication cost of PSO-Fed is lower than Online-Fed since M is much smaller than D. When $M = 40$, PSO-Fed behaves just like Online-Fed but

with a 80% reduction in communication overhead. We note that the clients that do not contribute to the global model update can perform local model updates whenever new data arrives. Subsequently, when those clients communicate with the server, only a part of those recent local updates are replaced with the aggregated server model. More importantly, a client shares a different portion of the local model with the server based on the most recently acquired data. This partial-parameter-sharing results in improved performance and reduced communication load. It is worthwhile noting that the proposed partial-sharing incurs no additional computational overhead, unlike the sketch updates presented in [24]. In order to maintain partially-shared parameter indices, a small amount of memory is required.

3) To perform local learning, PSO-Fed requires $D+1$ additional multiplication over SignSGD at each client. When comes to communication costs, the SignSGD algorithm needs to communicate $C \times 2D$ bits to complete a global iteration. For $C = 4$, the SignSGD communication cost is 1600 bits. Whereas with 32-bit fixed-point representation, the proposed PSO-Fed requires to communicate $32 \times C \times 2M$ bits to complete a global iteration. For $M = 1$ and 5, the communication costs are 256 bits and 1280 bits, respectively. We see that the proposed PSO-Fed offers much better performance than the SignSGD at lower communication costs. It is important to note that the SignSGD performs better as the value of C increases but also rises the communication costs. According to Sensoria sensors and Berkeley motes, the energy consumption ratio for communication to that for computation per bit ranges from 1000 to 10000 [56]. Furthermore, the communication efficiency of the proposed PSO-Fed over SignSGD is more pronounced under 16-bit fixed-point representation (which is sufficient for implementing RFF-based KLMS algorithm on chip [57]). In light of all this, PSO-Fed offers greater energy-efficiency than SignSGD.
4) Finally, compared to an uncoordinated partial-sharing, a coordinated partial-sharing-based online FL strategy shows a better initial convergence when M is very small (e.g., 1). In the coordinated partial-sharing-based scheme, the server aggregates the same portion of entries from the local model parameter vectors, thus preserving the client's interconnectedness. For large values of M, however, both schemes perform equally well (e.g., ≥ 5 in our experiment).

Next, we simulated ETPSO-Fed to carry out the aforementioned kernel regression task for different values of M. The error bound γ in ETPSO-Fed was set to 0.2 (selected through a grid search ranging from 0 to 2 with an increment of 0.01) and the remaining parameters were the same as in the above experiment. The learning curves are shown in Figs. 2a-2b for both partial-sharing-based ETPSO-Fed schemes. Fig. 2 shows that the proposed ETPSO-Fed exhibits similar performance as that of the PSO-Fed, i.e., same convergence speed and steady-state testing MSE for all values of M. Furthermore, the

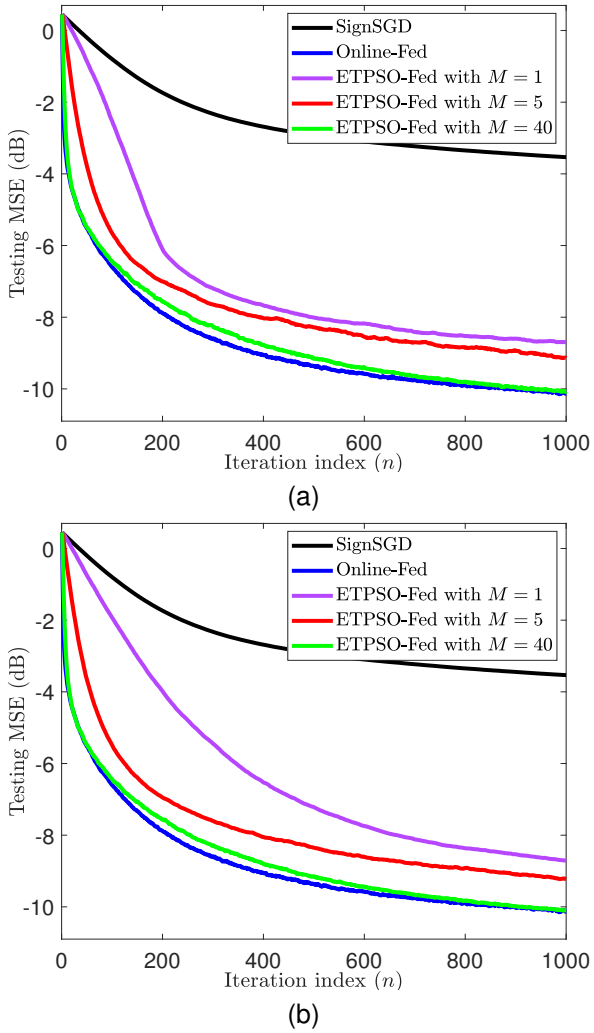

Fig. 2. Learning curves of the proposed ETPSO-Fed: (a). Coordinated partial-sharing. (b). Uncoordinated partial-sharing.

performance degradation is inversely proportional to the value of M. It is important to note that the ETPSO-Fed achieves the same performance as PSO-Fed with slightly lower communication load. When $M = 40$, ETPSO-Fed performs similarly to Online-Fed and PSO-Fed, but with an average reduction of 84.5% and 21.25% in communication overhead, respectively. Additionally, ETPSO-Fed reduces communication overhead by 42.5% on average when only uplink communication is considered, compared to PSO-Fed.

We next compared the communication cost in each global iteration (in terms of number of bits) associated with each FL strategy. For a fair comparison, all strategies were simulated to achieve the same learning performance. We considered 32-bit fixed-point representation to compute the communication costs of Online-Fed, PSO-Fed, and ETPSO-Fed. The corresponding curves (iteration index n vs. communication cost in number of bits) are displayed in Fig. 3. The SignSGD-Fed exhibits the same performance as that of Online-Fed when $C = 100$. For $C = 100$, the communication cost of the Sign-SGD is 40000 bits. The communication costs of Online-Fed and proposed PSO-Fed strategies are 51200 bits and 10240 bits, respectively.

The PSO-Fed communication cost was computed at $M = 40$, where it performs similarly to Online-Fed. Please note that the communication load of ETPSO-Fed is lower than PSO-Fed and varies in each global iteration as it employs an innovation check during the local client learning.

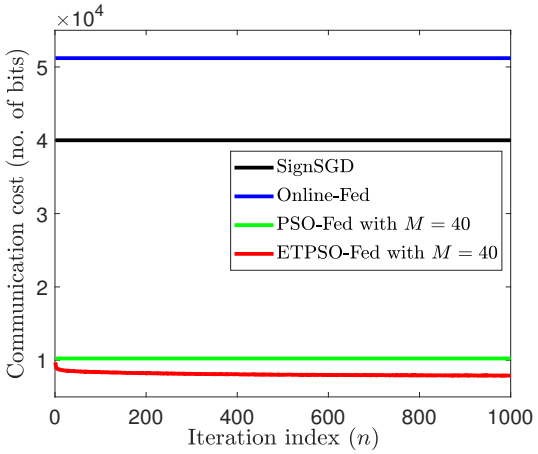

Fig. 3. Communication cost associated with the proposed FL strategies.

From Figs. 1, 2 and 3, we can see that the proposed communication-efficient online FL strategies, such as PSO-Fed and ETPSO-Fed, are able to achieve the same performance as that of Online-Fed with very little communication costs. With the help of an innovation check, ETPSO-Fed not only reduces communication overhead, but it also reduces the computational burden of clients.

As a last step, we carried out experiments to test how partial-sharing-based FL strategies fare under malicious attacks, i.e., when a few clients were trying to disrupt or bias the learning via model poisoning. We considered untargeted attack in which the Byzantine clients send random model updates, i.e., when participating in global iterations, the Byzantine clients share their contaminated local models with the server. For this purpose, Byzantine clients inject random Gaussian noise $\mathcal{N}(0, 1)$ into their local models. Firstly, we selected 10% clients at random to act as Byzantines. Then, the proposed PSO-Fed was simulated to perform the same kernel regression task in the presence of Byzantine clients and the corresponding learning curves for different values of M are presented in Fig. 4a for coordinated partial-sharing scheme and in Fig. 4d for uncoordinated partial-sharing scheme. Furthermore, we also plotted the variance of the noise used by the Byzantine vs. steady-state testing MSE in Figs. 4b and 4e, and percentage of Byzantine clients vs. steady-state testing MSE in Figs. 4c and 4f.

From Fig. 4, it is evident that the proposed PSO-Fed is robust to model-poisoning attacks. The performance of PSO-Fed deteriorates as the noise variance used by the Byzantine clients or the percentage of Byzantine clients increases. However, the performance degradation of PSO-Fed is in an acceptable range due to partial-sharing-based communication between clients and the server. By limiting model-poisoning to a small portion of the model, partial-sharing-based communication can reduce the speed at which model-poisoning spreads throughout the network. Online-Fed, on the other hand, is highly sensitive to the percentage of Byzantine clients or the variance of noise used by Byzantine clients. In addition, uncoordinated partial-sharing performs better than coordinated one in the presence of Byzantines. Model-poisoning becomes easy when all clients, including Byzantines, send the same entries of their local models to the server. Due to this, uncoordinated partial-sharing performed better than coordinated partial-sharing in the presence of Byzantines. The proposed PSO-Fed exhibits this robustness against model-poisoning attacks without requiring additional adversarial detection mechanisms at the server. Lastly, it is worth noting that the ETPSO-Fed also exhibits similar performance to PSO-Fed under malicious attacks.

VI. Conclusions

This paper presented communication-efficient online federated learning strategies based on partial sharing of model parameters. In the basic partial-sharing online federated learning (PSO-Fed) approach, participating clients exchange merely a small portion of the entire model parameters with the server, while the non-participating clients independently perform local learning when accessing new data. As such, partial-sharing-based communication offers clients more control over local learning than standard methods. In addition, within the partial-sharing framework, we also proposed an event-triggered PSO-Fed (ETPSO-Fed) stemming from set-membership filtering principles. Besides improving communication efficiency, ETPSO-Fed reduces the computational burden of the clients by adopting a client-side innovation check on the data. We implemented strategies in the context of kernel regression and provided a detailed study of the mean and mean-square convergence of the PSO-Fed strategy. Simulation results showed that both PSO-Fed and ETPSO-Fed are capable of maintaining competitive performance while significantly reducing communication costs over Online-Fed. In the conducted simulations, we have observed an 80% reduction in PSO-Fed and an 84% reduction in ETPSO-Fed communication overhead compared to Online-Fed. Finally, we showed the proposed partial-sharing-based online FL strategies to be robust against model-poisoning with no additional mechanisms.

Appendix A
Evaluation of $\overline{\boldsymbol{\mathcal{A}}}_{\mathbf{S}}$ and $\overline{\boldsymbol{\mathcal{B}}}_{\mathbf{S}}$

At any given global iteration n, the probability of being selected in each global iteration n is equal for all clients and is given by $p_c = \frac{C}{K}$. Additionally, the probability of being communicated is the same for all model parameters at all clients and is equals to $p_e = \frac{M}{D}$. Using these probabilities and from (18) and (20), we have

$$\begin{aligned}\overline{\boldsymbol{\mathcal{A}}}_{\mathbf{S}} &= \mathbb{E}[\boldsymbol{\mathcal{A}}_{\mathbf{S},n}] \\ &= \begin{bmatrix} \mathbf{I}_D & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \\ p_c p_e \mathbf{I}_D & (1-p_c p_e)\mathbf{I}_D & \mathbf{0} & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ p_c p_e \mathbf{I}_D & \mathbf{0} & \mathbf{0} & \dots & (1-p_c p_e)\mathbf{I}_D \end{bmatrix},\end{aligned} \tag{43}$$

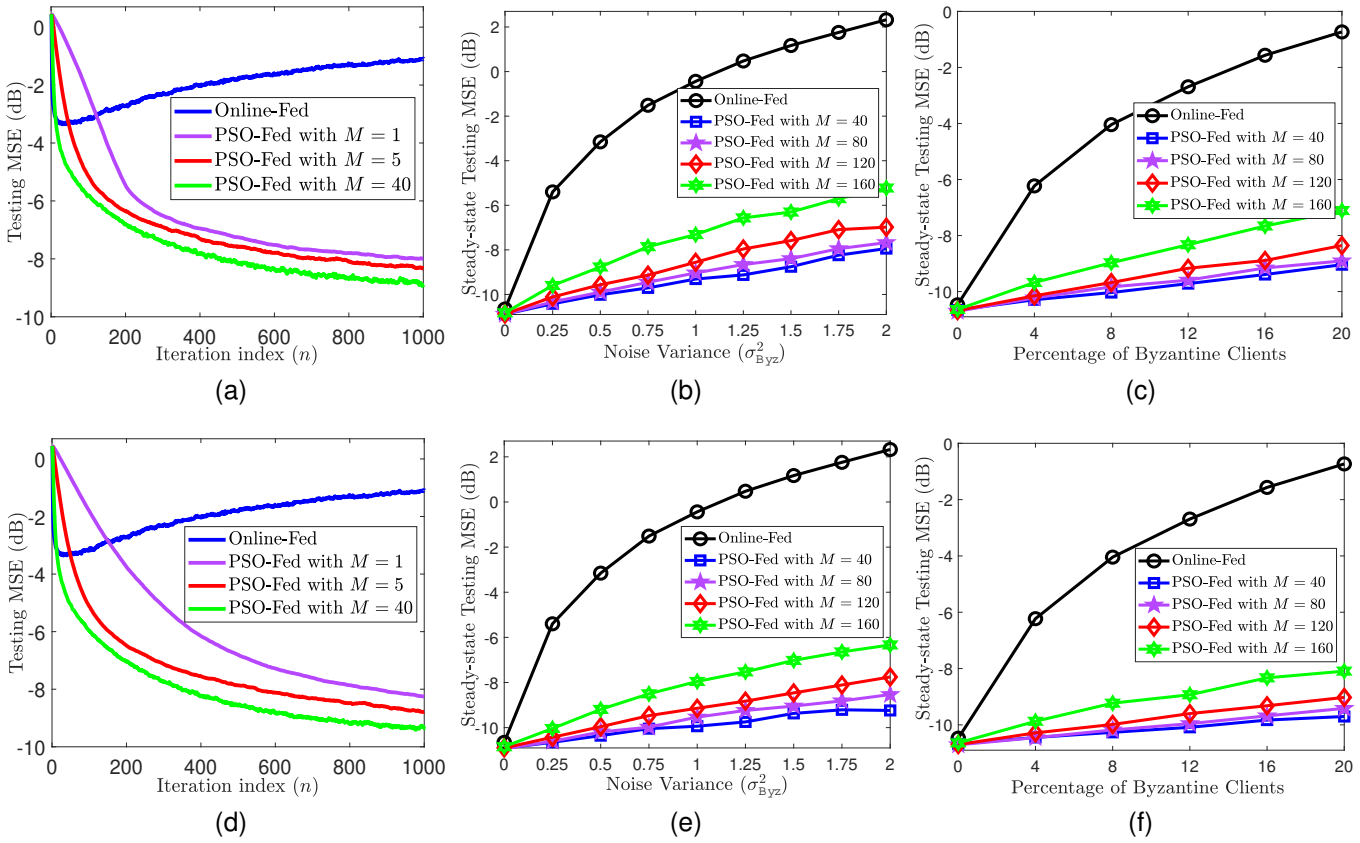

Fig. 4. Performance of PSO-Fed in the presence of malicious attacks. Coordinated partial-sharing: (a). learning curves for 10% Byzantine clients and $\sigma^2_{\text{Byz}} = 1$, (b). σ^2_{Byz} vs steady-state testing MSE for various values of M, (c). Percentage of Byzantine clients vs steady-state testing MSE for various values of M. Uncoordinated partial-sharing: (d). Learning curves for 10% Byzantine clients and $\sigma^2_{\text{Byz}} = 1$, (e). σ^2_{Byz} vs steady-state testing MSE for various values of M, (f). Percentage of Byzantine clients vs steady-state testing MSE for various values of M.

and

$$\overline{\boldsymbol{\mathcal{B}}}_{\mathbf{S}} = \mathbb{E}[\boldsymbol{\mathcal{B}}_{\mathbf{S},n+1}] = \begin{bmatrix} (1-p_c p_e)\mathbf{I}_D & \frac{p_c p_e}{C}\mathbf{I}_D & \dots & \frac{p_c p_e}{C}\mathbf{I}_D \\ \mathbf{0} & \mathbf{I}_D & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{I}_D \end{bmatrix}. \tag{44}$$

Subsequently, $\overline{\boldsymbol{\mathcal{A}}}_{\mathbf{S}}\mathbf{1}_{D(K+1)} = \mathbb{E}[\boldsymbol{\mathcal{A}}_{\mathbf{S},n}\mathbf{1}_{D(K+1)}] = \mathbf{1}_{D(K+1)}$ and $\overline{\boldsymbol{\mathcal{B}}}_{\mathbf{S}}\mathbf{1}_{D(K+1)} = \mathbb{E}[\boldsymbol{\mathcal{B}}_{\mathbf{S},n}\mathbf{1}_{D(K+1)}] = \mathbf{1}_{D(K+1)}$, implying the row sum of these matrices is unity.

Appendix B
Evaluation of $\mathcal{Q}_{\mathcal{A}}$ and $\mathcal{Q}_{\mathcal{B}}$

We have

$$\boldsymbol{\mathcal{A}}_{\boldsymbol{\mathcal{S}},n} = \begin{bmatrix} \mathbf{A}_{1,1,n} & \mathbf{A}_{1,2,n} & \dots & \mathbf{A}_{1,K+1,n} \\ \mathbf{A}_{2,1,n} & \mathbf{A}_{2,2,n} & \dots & \mathbf{A}_{2,K+1,n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{K+1,1,n} & \mathbf{A}_{K+1,2,n} & \dots & \mathbf{A}_{K+1,K+1,n} \end{bmatrix}, \tag{45}$$

with

$$\mathbf{A}_{i,j,n} = \begin{cases} \mathbf{I}_D, & \text{if } i,j = 1 \\ a_{i,n}\mathbf{S}_{i,n}, & \text{if } i = 2,\dots,K+1, j = 1 \\ \mathbf{I}_D - a_{i,n}\mathbf{S}_{i,n}, & \text{if } (i=j) \neq 1 \\ \mathbf{0}, & \text{otherwise.} \end{cases} \tag{46}$$

Then, $\boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{A}}}$ is given by

$$\mathbb{E}[\boldsymbol{\mathcal{A}}_{\boldsymbol{\mathcal{S}},n} \otimes_b \boldsymbol{\mathcal{A}}_{\boldsymbol{\mathcal{S}},n}] = \mathbb{E}\begin{bmatrix} \mathbf{A}_{1,1,n} \otimes_b \boldsymbol{\mathcal{A}}_{\boldsymbol{\mathcal{S}},n} & \dots & \mathbf{A}_{1,K+1,n} \otimes_b \boldsymbol{\mathcal{A}}_{\boldsymbol{\mathcal{S}},n} \\ \mathbf{A}_{2,1,n} \otimes_b \boldsymbol{\mathcal{A}}_{\boldsymbol{\mathcal{S}},n} & \dots & \mathbf{A}_{2,K+1,n} \otimes_b \boldsymbol{\mathcal{A}}_{\boldsymbol{\mathcal{S}},n} \\ \vdots & \ddots & \vdots \\ \mathbf{A}_{K+1,1,n} \otimes_b \boldsymbol{\mathcal{A}}_{\boldsymbol{\mathcal{S}},n} & \dots & \mathbf{A}_{K+1,K+1,n} \otimes_b \boldsymbol{\mathcal{A}}_{\boldsymbol{\mathcal{S}},n} \end{bmatrix}, \tag{47}$$

with

$$\mathbb{E}[\mathbf{A}_{i,j,n} \otimes_b \boldsymbol{\mathcal{A}}_{\boldsymbol{\mathcal{S}},n}] = \mathbb{E}\begin{bmatrix} \mathbf{A}_{i,j,n} \otimes \mathbf{A}_{1,1,n} & \dots & \mathbf{A}_{i,j,n} \otimes \mathbf{A}_{1,K+1,n} \\ \mathbf{A}_{i,j,n} \otimes \mathbf{A}_{2,1,n} & \dots & \mathbf{A}_{i,j,n} \otimes \mathbf{A}_{2,K+1,n} \\ \vdots & \ddots & \vdots \\ \mathbf{A}_{i,j,1} \otimes \mathbf{A}_{K+1,1,n} & \dots & \mathbf{A}_{i,j,n} \otimes_b \mathbf{A}_{K+1,K+1,n} \end{bmatrix}, \tag{48}$$

for $i,j = 1,2,\dots K+1$. To evaluate (48), we need to calculate $\mathbb{E}[s_{i,q,n}s_{l,r,n}]$. During the nth global iteration, the

probability of selecting two entries from the same client to exchange with the server is given by $(\frac{M}{D})(\frac{M-1}{D-1}) = p_e(\frac{M-1}{D-1})$. In uncoordinated partial-sharing scheme, the probability for selecting two entries from two different clients to exchange with server is given by p_e^2. Consequently, for an uncoordinated partial-sharing scheme, we have

$$\mathbb{E}[s_{i,q,n}s_{l,r,n}] = \begin{cases} p_e & \text{if } i = l \text{ and } q = r \\ p_e(\frac{M-1}{D-1}) & \text{if } i = l \text{ and } q \neq r \\ p_e^2 & \text{if } i \neq l. \end{cases} \tag{49}$$

For coordinated partial-sharing scheme, we have

$$\mathbb{E}[s_{i,q,n}s_{l,r,n}] = \begin{cases} p_e & \text{if } i = l \text{ and } q = r \\ p_e(\frac{M-1}{D-1}) & \text{if } i = l \text{ and } q \neq r. \end{cases} \tag{50}$$

Similarly, a special case of coordinated partial-sharing, namely, periodic partial-sharing scheme communicates all the local model parameters to the server efor very $\frac{D}{M}$ iterations. So, we have

$$\mathbb{E}[s_{i,q,n}s_{l,r,n}] = p_e \;\text{ for } \forall i, l, q, r. \tag{51}$$

Using (51),$\mathbb{E}[\mathbf{A}_{i,j,n} \otimes \mathbf{A}_{l,m,n}]$ can be evaluated $\forall i, j, l, m$ as

$$\mathbb{E}[\mathbf{A}_{i,j,n} \otimes \mathbf{A}_{l,m,n}] = \begin{cases} \mathbf{I}_{D^2} & \text{if } i,j = 1 \text{ and } l,m = 1 \\ p_c p_e \mathbf{I}_{D^2} & \text{if } i,j = 1 \\ & \text{and } l = 2,\ldots,K+1, m = 1 \\ p_c p_e \mathbf{I}_{D^2} & \text{if } i = 2,\ldots,K+1, j = 1 \\ & \text{and } l,m = 1 \\ (1-p_c p_e)\mathbf{I}_{D^2} & \text{if } i,j = 1 \text{ and } (l = m) \neq 1 \\ (1-p_c p_e)\mathbf{I}_{D^2} & \text{if } (i = j) \neq 1 \text{ and } l,m = 1 \\ p_c p_e \mathbf{I}_{D^2} & \text{if } (i = l) = 2,\ldots,K+1 \\ & \text{and } (j = m) = 1 \\ p_c p_e \mathbf{I}_{D^2} & \text{if } (i \neq l) = 2,\ldots,K+1 \\ & \text{and } (j = m) = 1 \\ \mathbf{0} & \text{if } (i = l) = 2,\ldots,K+1; j = 1 \\ & \text{and } (l = m) \neq 1 \\ (p_c p_e - p_c p_e)\mathbf{I}_{D^2} & \text{if } (i \neq l) = 2,\ldots,K+1, j = 1 \\ & \text{and } (l = m) \neq 1 \\ \mathbf{0} & \text{if } (i = l) = 2\ldots,K+1; m = 1 \\ & \text{and } (i = j) \neq 1 \\ (p_c p_e - p_c p_e)\mathbf{I}_{D^2} & \text{if } (i \neq l) = 2,\ldots,K+1, m = 1 \\ & \text{and } (i = j) \neq 1 \\ (1-p_c p_e)\mathbf{I}_{D^2} & \text{if } (i = j, l = m) \neq 1, \text{ and } (i = l) \\ (1-2p_c p_e + p_c p_e)\mathbf{I}_{D^2} & \text{if } (i = j, l = m) \neq 1, \text{ and } (i \neq l) \\ \mathbf{0} & \text{otherwise.} \end{cases} \tag{52}$$

Similarly, we have

$$\boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1} = \begin{bmatrix} \mathbf{B}_{1,1,n+1} & \mathbf{B}_{1,2,n+1} & \cdots & \mathbf{B}_{1,K+1,n+1} \\ \mathbf{B}_{2,1,n+1} & \mathbf{B}_{2,2,n+1} & \cdots & \mathbf{B}_{2,K+1,n+1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{B}_{K+1,1,n+1} & \mathbf{B}_{K+1,2,n+1} & \cdots & \mathbf{B}_{K+1,K+1,n+1} \end{bmatrix}, \tag{53}$$

with

$$\mathbf{B}_{i,j,n} = \begin{cases} \mathbf{I}_D - \sum\limits_{k \in \mathcal{S}_n} \frac{a_{k,n}}{C}\mathbf{S}_{k,n} & \text{if } i,j = 1 \\ \frac{a_{j,n}}{C}\mathbf{S}_{j,n} & \text{if } i = 1, j = 2,\ldots,K+1 \\ \mathbf{I}_D & \text{if } (i = j) \neq 1 \\ \mathbf{0}, & \text{otherwise.} \end{cases} \tag{54}$$

Then, $\boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{A}}}$ is given by

$$\mathbb{E}[\boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1} \otimes_b \boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1}] = \mathbb{E}\begin{bmatrix} \mathbf{B}_{1,1,n+1} \otimes_b \boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1} & \cdots & \mathbf{B}_{1,K+1,n+1} \otimes_b \boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1} \\ \mathbf{B}_{2,1,n+1} \otimes_b \boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1} & \cdots & \mathbf{B}_{2,K+1,n+1} \otimes_b \boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1} \\ \vdots & \ddots & \vdots \\ \mathbf{B}_{K+1,1,n+1} \otimes_b \boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1} & \cdots & \mathbf{B}_{K+1,K+1,n+1} \otimes_b \boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1} \end{bmatrix}, \tag{55}$$

with

$$\mathbb{E}[\mathbf{B}_{i,j,n+1} \otimes_b \boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n+1}] = \mathbb{E}\begin{bmatrix} \mathbf{B}_{i,j,n+1} \otimes \mathbf{B}_{1,1,n+1} & \cdots & \mathbf{B}_{i,j,n} \otimes \mathbf{B}_{1,K+1,n+1} \\ \mathbf{B}_{i,j,n+1} \otimes \mathbf{B}_{2,1,n+1} & \cdots & \mathbf{B}_{i,j,n} \otimes \mathbf{B}_{2,K+1,n+1} \\ \vdots & \ddots & \vdots \\ \mathbf{B}_{i,j,n+1} \otimes \mathbf{B}_{K+1,1,n+1} & \cdots & \mathbf{B}_{i,j,n} \otimes_b \mathbf{B}_{K+1,K+1,n+1} \end{bmatrix}. \tag{56}$$

for $i, j = 1, 2, \ldots K+1$. For periodic selection of clients and model coefficients, we have $\mathbb{E}[a_{i,n}a_{l,n}] = p_c$ and $\mathbb{E}[\mathbf{S}_{i,n+1} \otimes \mathbf{S}_{l,n+1}] = p_e\mathbf{I}_{D^2}$, for $i, l = 1, 2, \ldots, K+1$. We can then have

$$\mathbb{E}[\mathbf{B}_{i,j,n+1} \otimes \mathbf{A}_{l,m,n+1}] = \begin{cases} (1-2p_e+\frac{p_e}{p_c})\mathbf{I}_{D^2} & \text{if } i,j = 1 \text{ and } l,m = 1 \\ (\frac{p_c p_e}{C} - \frac{p_e}{C})\mathbf{I}_{D^2} & \text{if } i,j = 1 \\ & \text{and } l = 1, m = 2,\ldots,K+1 \\ (\frac{p_c p_e}{C} - \frac{p_e}{C})\mathbf{I}_{D^2} & \text{if } i = 1, j = 2,\ldots,K+1 \\ & \text{and } l,m = 1 \\ (1-p_c p_e)\mathbf{I}_{D^2} & \text{if } i,j = 1 \text{ and } (l = m) \neq 1 \\ (1-p_c p_e)\mathbf{I}_{D^2} & \text{if } (i = j) \neq 1 \text{ and } l,m = 1 \\ \frac{p_c p_e}{C^2}\mathbf{I}_{D^2} & \text{if } i = 1, j = 2,\ldots,K+1 \\ & \text{and } l = 1, m = 2,\ldots,K+1 \\ (1-p_e)\mathbf{I}_{D^2} & \text{if } i,j = 1, \text{ and } (l = m) \neq 1 \\ (1-p_e)\mathbf{I}_{D^2} & \text{if } (i = j) \neq 1, \text{ and } l,m = 1 \\ \frac{p_c p_e}{C}\mathbf{I}_{D^2} & \text{if } i = 1, j = 2,\ldots,K+1 \\ & \text{and } (l = m) \neq 1 \\ \frac{p_c p_e}{C}\mathbf{I}_{D^2} & \text{if } (i = j) \neq 1 \\ & \text{and } l = 1, m = 2,\ldots,K+1 \\ \mathbf{I}_{D^2} & \text{if } (i = j) \neq 1, \text{ and } (l = m) \neq 1 \\ \mathbf{0} & \text{otherwise.} \end{cases} \tag{57}$$

Subsequently, $\boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{A}}}\mathbf{1}_{D^2(K+1)^2} = \mathbb{E}[\boldsymbol{\mathcal{A}}_{\boldsymbol{\mathcal{S}},n}\mathbf{1}_{D(K+1)} \otimes_b \boldsymbol{\mathcal{A}}_{\boldsymbol{\mathcal{S}},n}\mathbf{1}_{D(K+1)}] = \mathbf{1}_{D^2(K+1)^2}$ and $\boldsymbol{\mathcal{Q}}_{\boldsymbol{\mathcal{B}}}\mathbf{1}_{D^2(K+1)^2} = \mathbb{E}[\boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n}\mathbf{1}_{D(K+1)} \otimes_b \boldsymbol{\mathcal{B}}_{\boldsymbol{\mathcal{S}},n}\mathbf{1}_{D(K+1)}] = \mathbf{1}_{D^2(K+1)^2}$, implying the row sum of these matrices is unity.

REFERENCES

[1] J. Konečnỳ, H. B. McMahan D. Ramage, and Peter Richtárik "Federated optimization: Distributed machine learning for on-device intelligence," in arXiv:1610.02527, 2016, [online]. Available: arXiv: 1610.02527.

[2] Q. Yang, Y. Liu, T. Chen and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 2157-6904, Feb. 2019.

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

[4] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y-C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 2031-2063, 2020.

[5] S. Savazzi, M. Nicoli and V. Rampa, "Federated learning with cooperating devices: A consensus approach for massive IoT networks," *IEEE Internet of Things J.*, vol. 7, no. 5, pp. 4641-4654, May 2020.

[6] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 1, pp. 269-283, Jan. 2021.

[7] S. Abdulrahman, H. Tout, H. Ould-Slimane, A. Mourad, C. Talhi and M. Guizani, "A survey on federated learning: The journey from centralized to distributed on-site learning and beyond," *IEEE Internet of Things J.*, vol. 8, no. 7, pp. 5476-5497, Apr. 2021.

[8] V. Smith, C. K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Proc. Advances in Neural Info. Process. Syst.*, Long Beach, CA, USA, Dec. 2017.

[9] R. Li, F. Ma, W. Jiang and J. Gao, "Online federated multitask learning," *Proc. IEEE Int. Conf. Big Data*, 2019, pp. 215-220.

[10] E. Jeong, S. Oh, H. Kim, J. Park, M. Bennis and S.-L. Kim, "Communication-efficient on-device machine learning: Federated distillation and augmentation under non-IID private data," in arXiv: 1811.11479, 2018, [online]. Available: arXiv: 1811.11479.

[11] F. Sattler, S. Wiedemann, K-R. Müller and W. Samek, "Robust and communication-efficient federated learning from non-i.i.d. data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 9, pp. 3400-3413, Sep. 2020.

[12] Y. Zhou, Q. Ye and J. Lv, "Communication-efficient federated learning with compensated overlap-FedAvg," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 01, pp. 192-205, 2022.

[13] Y. Chen, Y. Ning, M. Slawski and H. Rangwala, "Asynchronous online federated learning for edge devices with non-IID data," in *Proc. IEEE Int. Conf. Big Data,* 2020, pp. 15-24.

[14] S. Niknam, H. S. Dhillon and J. H. Reed, "Federated learning for wireless communications: Motivation, opportunities, and challenges," *IEEE Commun. Mag.*, vol. 58, no. 6, pp. 46-51, Jun. 2020.

[15] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proc. ACM SIGSAC Conf. Comput. and Commun. Security,* 2017, 1175–1191.

[16] Y. Xuefei, Z. Yanming, and H. Jiankun, "A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions," *ACM Comput. Surveys,* vol. 54, no. 6, pp. 46-51, Jul. 2021.

[17] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Int. Conf. Artif. Intell. and Stat.,* 2017, vol. 54, pp. 1273-1282.

[18] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proc. IEEE Int. Conf. Commun.,* 2019, pp. 1-7.

[19] H. H. Yang, Z. Liu, T. Q. S. Quek and H. V. Poor, "Scheduling policies for federated learning in wireless networks," *IEEE Trans. Commun.,* vol. 68, no. 1, pp. 317-333, Jan. 2020.

[20] H. T. Nguyen, V. Sehwag, S. Hosseinalipour, C. G. Brinton, M. Chiang and H. Vincent Poor, "Fast-convergent federated learning," *IEEE J. Sel. Areas Commun,* vol. 39, no. 1, pp. 201-218, Jan. 2021.

[21] E. Rizk, S. Vlaski and A. H. Sayed, "Optimal importance sampling for federated learning," in *Proc. IEEE Int. Conf. Acoust., Speech and Signal Process.,* 2021, pp. 3095-3099.

[22] T. Li, A. Kumar Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar and V. Smith, "Federated optimization in heterogeneous networks," in arXiv: 1812.06127, 2018, [online]. Available: arXiv: 1812.06127.

[23] L. Wang, W. Wang, and B. Li, "CMFL: Mitigating communication overhead for federated learning," in *Proc. IEEE Int. Conf. Distrib. Comput. Syst.,* 2019, pp. 954–964.

[24] J. Konečný H. B. McMahan, F. X. Yu, P. Richtarik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," in *Proc. NIPS Workshop on Private Multi-Party Mach. Learn.,* 2016.

[25] X. Wu, X. Yao and C-L. Wang, "FedSCR: Structure-based communication reduction for federated learning," *IEEE Trans. Parallel Distrib. Syst.,* vol. 32, no. 7, pp. 1565-1577, Jul. 2021.

[26] R. Jin, Y. Huang, X. He, H. Dai and Tianfu Wu, "Stochastic-sign SGD for federated learning with theoretical guarantees," in arXiv:2002.10940, 2020, [online]. Available: arXiv: 2002.10940.

[27] J. Park, S. Samarakoon, M. Bennis and M. Debbah, "Wireless network intelligence at the edge," in *Proc. IEEE,* vol. 107, no. 11, pp. 2204-2239, Nov. 2019.

[28] S. Samarakoon, M. Bennis, W. Saad and M. Debbah, "Distributed federated learning for ultra-reliable low-latency vehicular communications," *IEEE Trans. Commun.,* vol. 68, no. 2, pp. 1146-1159, Feb. 2020.

[29] W. U. Bajwa, V. Cevher, D. Papailiopoulos and A. Scaglione, "Machine learning from distributed, streaming Data," *IEEE Signal Process. Mag.,* vol. 37, no. 3, pp. 11-13, May 2020.

[30] T. Zhang, L. Gao, C. He, M. Zhang, B. Krishnamachari and A. S. Avestimehr, "Federated learning for the internet of things: Applications, challenges, and opportunities," *IEEE Internet of Things Mag.,* vol. 5, no. 1, pp. 24-29, Mar. 2022.

[31] A. Kuh, "Real time kernel learning for sensor networks using principles of federated learning," in *Proc. IEEE Int. Conf. Asia Pacific Signal and Info. Process. Assoc.,* 2021, pp. 2089-2093.

[32] Z. Chai, Y. Chen, L. Zhao, Y. Cheng and H. Rangwala, "FEDAT: A communication-efficient federated learning method with asynchronous tiers under non-iid data," in arXiv: 2010.05958, 2020, [online]. Available: arXiv: 2010.05958.

[33] W. Liu, P. P. Pokharel and J. C. Principe, "The kernel least-mean-square algorithm," *IEEE Trans. Signal Process.,* vol. 56, no. 2, pp. 543-554, Feb. 2008.

[34] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in *Proc. Adv. Neural Inf. Process. Syst.,* vol. 20, 2007, pp. 1177-1184.

[35] P. Bouboulis, S. Chouvardas and S. Theodoridis, "Online distributed learning over networks in RKH spaces using random Fourier features,"*IEEE Trans. Signal Process.,* vol. 66, no. 7, pp. 1920-1932, 1 Apr., 2018.

[36] Z. Wu, Q. Ling, T. Chen and G. B. Giannakis, "Federated variance-reduced stochastic gradient descent with robustness to byzantine attacks," *IEEE Trans. Signal Process.,* vol. 68, pp. 4583-4596, Jul. 2020.

[37] J. So, B. Güler and A. S. Avestimehr, "Byzantine-resilient secure federated learning," *IEEE J. Sel. Areas Commun.,* vol. 39, no. 7, pp. 2168-2181, Jul. 2021.

[38] Y. Yilmaz, G. V. Moustakides and X. Wang, "Sequential and decentralized estimation of linear-regression parameters in wireless sensor networks," *IEEE Transactions on Aerospace and Electronic Systems,* vol. 52, no. 1, pp. 288-306, Feb. 2016.

[39] L. Gispan, A. Leshem, Y. Be'ery, "Decentralized estimation of regression coefficients in sensor networks," *Digital Signal Process.,* Vol. 68, pp. 16-23, Sep. 2017.

[40] C. Richard, J. Bermudez, and P. Honeine, "Online prediction of time series data with kernels," *IEEE Trans. Signal Process.*, vol. 57, no. 3, pp. 1058–1067, Mar. 2009.

[41] A. Koppel, S. Paternain, C. Richard and A. Ribeiro, "Decentralized online learning with kernels," *IEEE Trans. on Signal Process.,* vol. 66, no. 12, pp. 3240-3255, Jun. 2018.

[42] W. Liu, J. C. Príncipe, and S. Haykin, *Kernel Adaptive Filtering: A Comprehensive Introduction*. Wiley, 2010.

[43] V. C. Gogineni, V. R. M. Elias, W. A. Martins and S. Werner, "Graph diffusion kernel LMS using random Fourier features," in Proc. Asilomar Conf. on Signals, Syst., and Comput., 2020, pp. 1528-1532.

[44] V. C. Gogineni, V. Naumova, S. Werner and Y-F. Haung "Graph kernel recursive least-squares algorithms," in Proc. Asia-Pacific Signal Inf. Process. Assoc., 2021, pp. 2072-2076.

[45] V. R. M. Elias, V. C. Gogineni, W. A. Martins and S. Werner, "Kernel regression over graphs using random Fourier features," *IEEE Trans. Signal Process.*, vol. 70, pp. 936-949, Feb. 2022.

[46] K. Prashant, H. Yang, M. Hong, J. Liu, H. T. Wai, S. Liu, "Decentralized learning for overparameterized problems: A multi-agent kernel approximation approach," in *Proc. Int. Conf. Learning Representations,* 2022.

[47] S. Theodoridis, *Machine Learning: A Bayesian and Optimization Perspective,* 2nd Edition, Academic Press, 2020.

[48] R. Arablouei, S. Werner, Y. F. Huang and K. Doğançay, "Distributed least mean-square estimation with partial diffusion," *IEEE Trans. Signal Process.*, Vol. 62, no. 2, pp. 472–484, Jan. 2013.
[49] R. Arablouei,K. Doğançay, S. Werner and Y. F. Huang, "Adaptive distributed estimation based on recursive least-squares and partial diffusion," *IEEE Trans. Signal Process.*, Vol. 62, no. 14, pp. 3510–3522, Jul. 2014.
[50] V. C. Gogineni and M. Chakraborty, "Partial diffusion affine projection algorithm over clustered multitask networks," in *Proc. IEEE Int. Symp. Circuits and Syst.*, 2019, pp. 1-5.
[51] E. B. Saff, A. D. Snider, *Fundamentals of Matrix Analysis with Applications*, Wiley, 2015.
[52] A. H . Sayed, "Diffusion adaptation strategies for distributed optimization and learning over networks," *IEEE Trans. Signal Process.*, vol. 60, no. 8, pp. 4289-4305, Aug. 2012.
[53] R. H. Koning, H. Neudecker and T. Wansbeek, "Block Kronecker products and the vecb operator," *Linear Algebra and its Applications*, vol. 149, pp. 165-184, 1991.
[54] S. Gollamudi, S. Nagaraj, S. Kapoor and Y-F. Huang, "Set-membership filtering and a set-membership normalized LMS algorithm with an adaptive step size," *IEEE Signal Process. Lett.*, vol. 5, no. 5, pp. 111-114, May 1998.
[55] K. Chen, S. Werner, A. Kuh and Y-F. Huang, "Nonlinear adaptive filtering with kernel set-membership approach," *IEEE Trans. Signal Process.*, vol. 68, pp. 1515-1528, 2020.
[56] F. Zhao, J. Liu, J. Liu, L. Guibas and J. Reich, "Collaborative signal and information processing: an information-directed approach," *Proc. of the IEEE*, vol. 91, no. 8, pp. 1199-1209, Aug. 2003.
[57] V. C. Gogineni, R. Sambangi, D. Alex, S. Mula and S. Werner, "Algorithm and architecture design of random Fourier features-based kernel adaptive filters," *IEEE Trans. Circuits. Syst. I: Reg. Papers*, Submitted.

Vinay Chakravarthi Gogineni (Member, IEEE) received the Bachelor's degree in electronics and communication engineering from Jawaharlal Nehru Technological University, Andhra Pradesh, India, in 2005, the Master's degree in communication engineering from VIT University, India, in 2008, and the Ph.D. degree in electronics and electrical communication engineering from Indian Institute of Technology Kharagpur, India in 2019. From 2008 to 2011, he was with a couple of MNCs in India. Currently, he is working as a postdoctoral research fellow at the Department of Electronic Systems, NTNU-Norway. His research interests include statistical signal processing, distributed and federated learning, and geometric deep learning. He was a recipient of the ERCIM Alain Bensoussan Fellowship in 2019 and the Best Paper Award at APSIPA ASC-2021, Tokyo, Japan.

Stefan Werner (Senior Member, IEEE) received the M.Sc. Degree in electrical engineering from the Royal Institute of Technology, Stockholm, Sweden, in 1998, and the D.Sc. degree (Hons.) in electrical engineering from the Signal Processing Laboratory, Helsinki University of Technology, Espoo, Finland, in 2002. He is currently a Professor at the Department of Electronic Systems, Norwegian University of Science and Technology (NTNU), Director of IoT@NTNU, and Adjunct Professor with Aalto University in Finland. He was a visiting Melchor Professor with the University of Notre Dame during the summer of 2019 and an Adjunct Senior Research Fellow with the Institute for Telecommunications Research, University of South Australia, from 2014 to 2020. He held an Academy Research Fellowship, funded by the Academy of Finland, from 2009 to 2014. His research interests include adaptive and statistical signal processing, wireless communications, and security and privacy in cyber-physical systems. He is a member of the editorial boards for the EURASIP Journal of Signal Processing and the IEEE Transactions on Signal and Information Processing over Networks.

Yih-Fang Huang (Life Fellow, IEEE) is Professor of Electrical Engineering and Senior Associate Dean for Education and Undergraduate Programs in the College of Engineering. He received his B.S.E.E. degree from National Taiwan University, M.S.E.E. degree from University of Notre Dame, M.A. and Ph.D. degrees from Princeton University. He served as chair of Notre Dame's Electrical Engineering department from 1998 to 2006. His research lies in the area of statistical and adaptive signal processing and employs principles in mathematical statistics to solve signal detection and estimation problems that arise in various applications, including wireless communications, distributed sensor networks, smart electric power grid, etc.

Dr. Huang received the Golden Jubilee Medal of the IEEE Circuits and Systems Society in 1999. He also served as Vice President in 1997-98 and was a Distinguished Lecturer for the same society in 2000-2001. He served as the lead Guest Editor for a Special Issue on Signal Processing in Smart Electric Power Grid of the IEEE Journal of Selected Topics in Signal Processing, December 2014. At the University of Notre Dame, he received Presidential Award in 2003, the Electrical Engineering department's Outstanding Teacher Award in 1994 and in 2011, the Rev. Edmund P. Joyce, CSC Award for Excellence in Undergraduate Teaching in 2011, and the Engineering College's Outstanding Teacher of the Year Award in 2013.

In Spring 1993, Dr. Huang received the Toshiba Fellowship and was Toshiba Visiting Professor at Waseda University, Tokyo, Japan. From April to July 2007, he was a visiting professor at the Munich University of Technology, Germany. In Fall, 2007, Dr. Huang was awarded the Fulbright-Nokia scholarship for lectures/research at Helsinki University of Technology in Finland. He was appointed Honorary Professor in the College of Electrical Engineering and Computer Science at National Chiao-Tung University, Hsinchu, Taiwan, in 2014. Dr. Huang is a Life Fellow of the IEEE.

Anthony Kuh (Fellow, IEEE) received his B.S. in Electrical Engineering and Computer Science at the University of California, Berkeley in 1979, an M.S. in Electrical Engineering from Stanford University in 1980, and a Ph.D. in Electrical Engineering from Princeton University in 1987. He previously worked at AT&T Bell Laboratories and has been on the faculty in the Department of Electrical and Computer Engineering at the University of Hawai'i since 1986. He is currently a Professor and previously served as Department Chair. His research is in the area of neural networks and machine learning, adaptive signal processing, sensor networks, and renewable energy and smart grid applications. He won a National Science Foundation (NSF) Presidential Young Investigator Award and is an IEEE Fellow. From 2017-2021, he served as program director for NSF in the Electrical, Communications, and Cyber Systems (ECCS) division working in the Energy, Power, Control, and Network (EPCN) group. At NSF he also assisted in initiatives including Harnessing the Data Revolution (HDR), the Mathematics of Deep Learning (MoDL), the AI Institutes, Cyber Physical Systems (CPS), and Smart and Connected Communities. He previously served for the IEEE Signal Processing Society on the Board of Governors as a Regional Director-at-Large Regions1-6, as a senior editor for the IEEE Journal of Selected Topics in Signal Processing, and as a member of the Awards Board. He currently is the President of the Asia Pacific Signal and Information Processing Association.