Dynamic Federated Learning

Elsa Rizk *, Stefan Vlaski †, and Ali H. Sayed ‡
Institute of Electrical Engineering, Ecole Polytechnique Federal de Lausanne
Switzerland
Email: *elsa.rizk@epfl.ch, †stefan.vlaski@epfl.ch, ‡ali.sayed@epfl.ch

***Abstract*—Federated learning has emerged as an umbrella term for centralized coordination strategies in multi-agent environments. While many federated learning architectures process data in an online manner, and are hence adaptive by nature, most performance analyses assume static optimization problems and offer no guarantees in the presence of drifts in the problem solution or data characteristics. We consider a federated learning model where at every iteration, a random subset of available agents perform local updates based on their data. Under a non-stationary random walk model on the true minimizer for the aggregate optimization problem, we establish that the performance of the architecture is determined by three factors, namely, the data variability at each agent, the model variability across all agents, and a tracking term that is inversely proportional to the learning rate of the algorithm. The results clarify the trade-off between convergence and tracking performance.**

***Index Terms*—federated learning, distributed learning, tracking performance, dynamic optimization, asynchronous SGD, non-IID data, heterogeneous agents**

I. Introduction

We consider a collection of K agents dispersed in space. Each agent k has access to N_k data points denoted by $\{x_{k,n}\}$, where the subscript k denotes the agent index and the subscript n denotes the sample index within the dataset. The objective is to seek the minimizer of the aggregate risk:

$$w^o \triangleq \operatorname*{argmin}_{w\in\mathbb{R}^M} \frac{1}{K}\sum_{k=1}^{K} P_k(w), \tag{1}$$

where the individual risks at the local agents are in turn defined as sample averages over their loss values, i.e.,

$$P_k(w) \triangleq \frac{1}{N_k}\sum_{n=1}^{N_k} Q_k(w; x_{k,n}). \tag{2}$$

We will allow the sought-after model w^o to drift with time and often write w_i^o instead of just w^o to highlight this possibility. Here, the subscript i refers to a time index. The drift in w^o is often the result of variations in the statistical properties of the data $\{x_{k,n}\}$ at the agents, which can change with time as well.

Strategies for the pursuit of solutions to (1) can generally be divided into one of two classes: distributed architectures with a fusion center that collects all data centrally for processing [1]–[3], and fully decentralized strategies that rely on local information exchanges among neighbouring agents over a graph [4]–[7]. Federated learning [8]–[20] offers a midterm solution where data is collected locally at the agents and some processing is also performed locally, while global information is shared between a central processor and the dispersed agents. The architecture helps reduce the amount of communication rounds between the central processor and the agents.

A. Related Works

Several recent works have examined the convergence behavior of federated learning. They, however, vary in the assumptions on the number of participating agents (all or partial), nature of data (IID or not), operation (synchronous or asynchronous), nature of risk function (convex or non-convex), and bounded conditions on gradient noise [9], [10], [21]–[31] – see Table 1.

Some other works examine the convergence behaviour of a modified version of the original FedAvg algorithm; in [9] FedAvg is modified to include non-uniform epoch sizes among the agents; the authors then provide a convergence proof of the new algorithm, called FedProx, independent of the local solvers. They assume non-convex cost functions with non-IID data, and they quantify the statistical heterogeneity by a dissimilarity measure based on the randomized Kaczmarz method [32], [33] for solving linear system of equations. However, this proof fails to encapsulate the convergence of FedAvg. In [31], a hierarchical version of FedAvg is developed where model aggregation occurs at several levels, and the convergence under non-IID settings for both convex and non-convex functions is studied. The authors of [10] introduce multi-task federated learning to deal with statistical heterogeneity and show their algorithm converges for convex cost functions. They introduce the MOCHA algorithm that extends CoCoA [34] to deal with the challenges introduced by the federated setting, such as stragglers.

While related works have considered dynamic optimization algorithms under varying algorithmic frameworks [35]–[42], we focus in this work on the original FedAvg algorithm with varying step-sizes and under a non-stationary environment.

II. Algorithm Derivation

Returning to (1), in the absence of communication or computational constraints, the centralized gradient descent step takes the form:

$$w_i = w_{i-1} - \mu\frac{1}{K}\sum_{k=1}^{K} \nabla_{w^\intercal} P_k(w_{i-1}), \tag{3}$$

where i is the iteration index. We can distribute the update by splitting the gradient descent step among the K agents. After

TABLE I: List of references on the convergence analysis of federated learning under different assumptions. This work is the only one to tackle the 3 challenges of federated learning.

References	Algorithm	Function Type	Data Heterogeneity	Operation	Agent Participation	Other Assumptions
[22]	dist. gradient descent	convex	***non-IID***	synchronous	full	smooth
[23]	dist. SGD	convex	IID	synchronous	full	smooth
[24], [25]	dist. SGD	non-convex	IID	synchronous	full	smooth
[26]	dist. SGD	non-convex	***non-IID*** IID	synchronous ***asynchronous***	full	-
[27]	dist. SGD	convex	***non-IID***	synchronous	full	bounded gradients
[28]	dist. momentum SGD	non-convex	***non-IID***	synchronous	full	-
[29]	FedAvg	convex some non-convex	***non-IID***	***asynchronous***	full	-
[30]	FedAvg	convex	***non-IID***	synchronous	***partial***	bounded gradients
This work	FedAvg	convex	***non-IID***	***asynchronous***	***partial***	model drift

introducing local parameters $\boldsymbol{w}_{k,i}$ at each agent, recursion (3) becomes:

$$w_{k,i} = w_{i-1} - \mu \nabla_{w^\mathsf{T}} P_k(w_{i-1}), \tag{4}$$

$$w_i = \frac{1}{K} \sum_{k=1}^{K} w_{k,i-1}. \tag{5}$$

The drawback of this formulation is that evaluation of $\nabla_{w^\mathsf{T}} P_k(w_{i-1})$ at each agent k and for every iteration i requires a total of N_k evaluations of the gradients $\nabla Q_k(w_{i-1}; \boldsymbol{x}_{k,n})$. A popular approach for reducing the per-iteration computational cost is the utilization of stochastic gradient approximations. We will construct the gradient approximation as the average of E_k individual mini-batch approximations, each of size B_k:

$$\widehat{\nabla_{w^\mathsf{T}} P_k}(\boldsymbol{w}_{i-1}) \triangleq \frac{1}{E_k} \sum_{e=0}^{E_k-1} \frac{1}{B_k} \sum_{b \in \mathcal{B}_{k,e}} \nabla_{w^\mathsf{T}} Q_k(\boldsymbol{w}_{i-1}; \boldsymbol{x}_{k,b}), \tag{6}$$

where $\mathcal{B}_{k,e}$ denotes the e-th mini-batch set randomly sampled at time i by agent k. We sample the indices in $\mathcal{B}_{k,e}$ from the set of integers $\{1, \ldots, N_k\}$ *without replacement*. In the above, we are using the boldface notation $\boldsymbol{w}_{i-1}$ to reflect the random nature of the weight iterates. Note that this construction allows for significant heterogeneity in the agents' computational capabilities. In particular, by choosing E_k and B_k appropriately, each agent k is able to contribute to varying degrees, by performing a different number of gradient calculations. Then, the resulting stochastic gradient steps in (4)-(5) become:

$$\boldsymbol{w}_{k,i} = \boldsymbol{w}_{k,i-1} - \frac{\mu}{E_k B_k} \sum_{e=0}^{E_k-1} \sum_{b \in \mathcal{B}_{k,e}} \nabla_{w^\mathsf{T}} Q_k(\boldsymbol{w}_{i-1}; \boldsymbol{x}_{k,b}), \tag{7}$$

$$\boldsymbol{w}_i = \frac{1}{K} \sum_{k=1}^{K} \boldsymbol{w}_{k,i}. \tag{8}$$

An equivalent way of writing (7)–(8) is by introducing an inner iteration over $e = 0, \ldots, E_k - 1$, initialized at $\boldsymbol{w}_{k,-1} = \boldsymbol{w}_{i-1}$:

$$\boldsymbol{w}_{k,e} = \boldsymbol{w}_{k,e-1} - \frac{\mu}{E_k B_k} \sum_{b \in \mathcal{B}_{k,e}} \nabla_{w^\mathsf{T}} Q_k(\boldsymbol{w}_{i-1}; \boldsymbol{x}_{k,b}), \tag{9}$$

followed by:

$$\boldsymbol{w}_i = \frac{1}{K} \sum_{k=1}^{K} \boldsymbol{w}_{k,E_k}. \tag{10}$$

Examination of (9) reveals that, while $\boldsymbol{w}_{k,e}$ evolves with e, all gradients $\nabla_{w^\mathsf{T}} Q_k(\boldsymbol{w}_{i-1}; \boldsymbol{x}_{k,b})$ are evaluated at $\boldsymbol{w}_{k,-1} = \boldsymbol{w}_{i-1}$, which is the starting point of the inner loop. We can instead appeal to an incremental arguement [3] and replace (9) by:

$$\boldsymbol{w}_{k,e} = \boldsymbol{w}_{k,e-1} - \frac{\mu}{E_k B_k} \sum_{b \in \mathcal{B}_{k,e}} \nabla_{w^\mathsf{T}} Q_k(\boldsymbol{w}_{k,e-1}; \boldsymbol{x}_{k,b}), \tag{11}$$

where now all gradients $\nabla_{w^\mathsf{T}} Q_k(\boldsymbol{w}_{k,e-1}; \boldsymbol{x}_b)$ are evaluated at the latest estimate $\boldsymbol{w}_{k,e-1}$ at agent k. Since full participation of all K agents at every time instant i is generally infeasible in a federated learning scenario [8], we allow for a participation of only L agents at every iteration, and sample the set of indices of participating agents $\mathcal{L}_i$ from $\{1, \ldots, L\}$ without replacement, transforming the combination step (10) to:

$$\boldsymbol{w}_i = \frac{1}{L} \sum_{\ell \in \mathcal{L}_i} \boldsymbol{w}_{\ell,E_\ell}. \tag{12}$$

With the local incremental update step (11) and partial-participation combination step (12) we arrive at Algorithm 1.

This algorithm bears similarity to the original FedAvg [8] algorithm, but differs in the fact that we allow for varying local epoch sizes E_k and the normalization of gradient directions by epoch size. The advantage of this construction is in its applicability to settings where agents have varying capabilities so that some agents are able to run multiple epochs while others are not. Instead of forcing capable agents to only run a single epoch, not taking advantage of their computational resources, or forcing slow agents to perform multiple epochs, risking a straggler effect, this model allows every agent to contribute precisely as much as they are able to. The normalization of gradients by the number of local epochs E_k, as our analysis will show, is necessary to ensure an unbiased solution despite asymmetric agent contribution, essentially by ensuring that (6) is an unbiased estimate of the gradient of $P_k(\cdot)$. In the absence of step-size normalization, agents with

more participation would be able to bias the limiting point of the algorithm towards their own local minimizers.

Algorithm 1 (Dynamic Federated Averaging)

initialize w_0
for each iteration $i = 1, 2, \cdots$ **do**
  Select set of participating agents $\mathcal{L}_i$ by sampling L times from $\{1, \ldots, K\}$ without replacement.
  for each agent $k \in \mathcal{L}_i$ **do**
    initialize $\boldsymbol{w}_{k,-1} = \boldsymbol{w}_{i-1}$
    for each epoch $e = 1, 2, \cdots E_k$ **do**
      Find indices of the mini-batch sample $\mathcal{B}_{k,e}$ by sampling B_k times from $\{1, \ldots, N_k\}$ without replacement.
      $\boldsymbol{g} = \frac{1}{B_k} \sum_{b \in \mathcal{B}_{k,e}} \nabla_{w^\mathsf{T}} Q(\boldsymbol{w}_{k,e-1}; \boldsymbol{x}_{k,b})$
      $\boldsymbol{w}_{k,e} = \boldsymbol{w}_{k,e-1} - \mu \frac{1}{E_k} \boldsymbol{g}$
    end for
  end for
  $\boldsymbol{w}_i = \frac{1}{L} \sum_{k \in \mathcal{L}_i} \boldsymbol{w}_{k,E_k}$
end for

III. Convergence Analysis

A. Modeling Conditions

To facilitate the performance analysis of the dynamic federated averaging algorithm in a non-stationary environment, we assume convexity and smoothness of gradients.

Assumption 1. *The functions $P_k(\cdot)$ are $\nu-$strongly convex, and $Q_k(\cdot; \boldsymbol{x}_{k,n})$ are convex:*

$$P_k(w_2) \geq P_k(w_1) + \nabla_{w^\mathsf{T}} P_k(w_1)(w_2 - w_1) + \frac{\nu}{2}\|w_2 - w_1\|^2, \tag{13}$$

$$Q_k(w_2) \geq Q_k(w_1) + \nabla_{w^\mathsf{T}} Q_k(w_1)(w_2 - w_1). \tag{14}$$

They also have $\delta-$Lipschitz gradients:

$$\|\nabla_{w^\mathsf{T}} P_k(w_2) - \nabla_{w^\mathsf{T}} P_k(w_1)\| \leq \delta \|w_2 - w_1\|, \tag{15}$$

$$\|\nabla_{w^\mathsf{T}} Q_k(w_2; \boldsymbol{x})) - \nabla_{w^\mathsf{T}} Q_k(w_1; \boldsymbol{x}))\| \leq \delta \|w_2 - w_1\|. \tag{16}$$

□

We also impose an assumption on the drift of the minimizer $\boldsymbol{w}_i^o$, namely that it follows a random walk model.

Assumption 2. *We assume that the true model $\boldsymbol{w}_i^o$ follows a random walk:*

$$\boldsymbol{w}_i^o = \boldsymbol{w}_{i-1}^o + \boldsymbol{q}_i, \tag{17}$$

where $\boldsymbol{q}_i$ denotes some zero mean random variable independent of $\boldsymbol{w}_j^o$ for any $j < i$ and with bounded variance, i.e., $\mathbb{E}\|\boldsymbol{q}_i\|^2 = \sigma_q^2$. □

As it will turn out, the tracking performance of Algorithm 1 will be largely determined by the drift parameter σ_q^2 on the *global* model $\boldsymbol{w}_i^o$. Nevertheless, we need to additionally assume that the individual minimizers $\boldsymbol{w}_{k,i}^o \triangleq \operatorname{argmin}_{w \in \mathbb{R}^M} P_k(w)$ do not drift too far.

Assumption 3. *For all i, the distance of each local model $\boldsymbol{w}_{k,i}^o$ to the global model $\boldsymbol{w}_i^o$ is bounded, i.e.:*

$$\mathbb{E}\|\boldsymbol{w}_{k,i}^o - \boldsymbol{w}_i^o\|^2 \leq \sigma_{w,k}^2. \tag{18}$$

□

B. Error Recursion

Iterating the local update steps (11) and combining via (12), we find:

$$\begin{aligned}\boldsymbol{w}_i &= \boldsymbol{w}_{i-1} - \mu \frac{1}{L} \sum_{\ell \in \mathcal{L}_i} \frac{1}{E_\ell B_\ell} \sum_{e=0}^{E_\ell - 1} \sum_{b \in \mathcal{B}_{\ell,e}} \nabla_{w^\mathsf{T}} Q_\ell(\boldsymbol{w}_{\ell,e-1}; \boldsymbol{x}_{\ell,b}) \\ &= \boldsymbol{w}_{i-1} - \mu \frac{1}{K} \sum_{k=1}^{K} \nabla_{w^\mathsf{T}} P_k(\boldsymbol{w}_{i-1}) - \mu \boldsymbol{s}_i - \mu \boldsymbol{d}_i, \end{aligned} \tag{19}$$

where we introduced:

$$\boldsymbol{s}_i \triangleq \frac{1}{L} \sum_{\ell \in \mathcal{L}_i} \widehat{\nabla_{w^\mathsf{T}} P_\ell}(\boldsymbol{w}_{i-1}) - \frac{1}{K} \sum_{k=1}^{K} \nabla_{w^\mathsf{T}} P_k(\boldsymbol{w}_{i-1}), \tag{20}$$

$$\begin{aligned}\boldsymbol{d}_i \triangleq \frac{1}{L} \sum_{\ell \in \mathcal{L}_i} \frac{1}{E_\ell B_\ell} \sum_{e=0}^{E_\ell - 1} \sum_{b \in \mathcal{B}_{\ell,e}} \Big[&\nabla_{w^\mathsf{T}} Q_\ell(\boldsymbol{w}_{i-1}; \boldsymbol{x}_{\ell,b}) \\ &- \nabla_{w^\mathsf{T}} Q_\ell(\boldsymbol{w}_{\ell,e}; \boldsymbol{x}_{\ell,b}) \Big]. \end{aligned} \tag{21}$$

The term $\boldsymbol{s}_i$ results from the stochastic approximation of the true gradient of (1) by only utilizing a subset of agents and a subset of data at every iteration. The term $\boldsymbol{d}_i$ results from the incremental implementation of (11). The gradient noise term $\boldsymbol{s}_i$ will turn out to be the dominant factor in the performance of the algorithm, but can be bounded as follows.

Lemma 1. *The gradient noise defined in* (20) *satisfies:*

$$\mathbb{E}\ \{\boldsymbol{s}_i | \boldsymbol{w}_{i-1}\} = 0, \tag{22}$$

$$\mathbb{E}\left\{\|\boldsymbol{s}_i\|^2 | \boldsymbol{w}_{i-1}\right\} \leq \beta_s^2 \mathbb{E}\|\boldsymbol{w}_{i-1}^o - \boldsymbol{w}_{i-1}\|^2 + \sigma_s^2 + \epsilon^2, \tag{23}$$

where we defined:

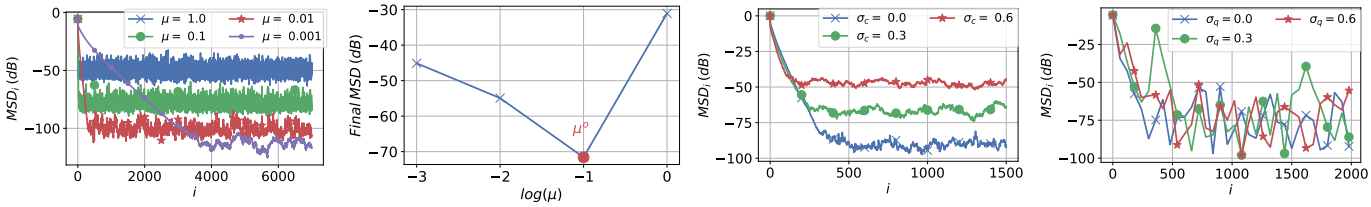
$$\beta_s^2 \ \triangleq\ \frac{1}{KL} \sum_{k=1}^{K} \Big(6\tau_{s,k} + 2\tau_\epsilon\Big)\delta^2, \tag{24}$$

$$\begin{aligned}\sigma_s^2 \ \triangleq\ \frac{3}{KL} \sum_{k=1}^{K} \tau_{s,k} \mathbb{E}\|\nabla_{w^\mathsf{T}} Q_k(\boldsymbol{w}_{i-1}^o; \boldsymbol{x}_k) \\ - \nabla_{w^\mathsf{T}} P_k(\boldsymbol{w}_{i-1}^o)\|^2, \end{aligned} \tag{25}$$

$$\epsilon^2 \ \triangleq\ \frac{2}{KL} \sum_{k=1}^{K} \tau_\epsilon^2 \mathbb{E}\|\nabla_{w^\mathsf{T}} P_k(\boldsymbol{w}_{i-1}^o)\|^2, \tag{26}$$

$$\tau_{s,k} \ \triangleq\ \frac{N_k - B_k}{(N_k - 1) B_k E_k}, \quad \tau_\epsilon \ \triangleq\ \frac{K - L}{K - 1}. \tag{27}$$

Proof. Omitted due to space limitations. □



(a) *Stationary case:* varying μ (b) *Non-stationary case:* varying μ (c) Varying σ_c^2 (d) Varying σ_q^2

Fig. 1: Plots of the MSD across time in dB.

From (23), we observe that the bound on the gradient noise variance consists of two absolute components, σ_s^2 and ϵ^2. The term σ_s^2 corresponds to an average of the *data variability* $\mathbb{E}\|\nabla_{w^\intercal} Q_k(\boldsymbol{w}_{i-1}^o; \boldsymbol{x}_k) - \nabla_{w^\intercal} P_k(\boldsymbol{w}_{i-1}^o)\|^2$ at each agent, weighted by the factor $\tau_{s,k}$. Agents with high data variability will incur a higher variance by employing a mini-batch approximation, instead of a full gradient update, but can mitigate this effect by increasing the mini-batch size B_k, and hence reducing $\tau_{s,k}$. In the limit case where agent k is performing a full gradient update, we have $B_k = N_k$, and hence $\tau_{s,k} = 0$ and no noise contribution from agent k. The second absolute noise term ϵ^2 stems from *model variability* among different agents. In particular, $\mathbb{E}\|\nabla_{w^\intercal} P_k(\boldsymbol{w}_{i-1}^o)\|^2$ measures the suboptimality of the average model $\boldsymbol{w}_{i-1}^o$ for the local cost $P_k(\cdot)$ at agent k, and in light of the Lipschitz gradient condition is proprtional to $\mathbb{E}\|\boldsymbol{w}_{i-1}^o - \boldsymbol{w}_{k,i-1}^o\|^2$; it stems from the incremental step introduced in (11). In this case, the model variability term is multiplied by a participation factor τ_ϵ, which quantifies the fraction of agents participating in the federated update at every iteration, and vanishes whenever all agents participate.

Theorem 1. *Consider the iterates $\boldsymbol{w}_i$ generated by the dynamic federated averaging algorithm. For sufficiently small step-size μ, it holds that $\mathbb{E}\|\boldsymbol{w}_i^o - \boldsymbol{w}_i\|^2$ converges exponentially fast:*

$$\mathbb{E}\|\boldsymbol{w}_i^o - \boldsymbol{w}_i\|^2 \leq O(\gamma^i) + O(\mu)\left(\sigma_s^2 + \epsilon^2\right) + O(\mu^{-1})\sigma_q^2, \tag{28}$$

where $\gamma < 1 - \nu\mu + O(\mu^2) \in [0, 1)$.

Proof. Omitted due to space limitations. □

We observe that, in addition to the model and data variability terms discussed above, the performance of the algorithm is further determined by the drift parameters of the random-walk model σ_q^2. A reduction in the step-size results in slower decay of the transient term $O(\gamma^i)$ and increased tracking loss $O(\mu^{-1})\sigma_q^2$, while reducing the effect of the gradient noise proportional to $O(\mu)$.

IV. Experimental Analysis

A. Experimental Setup

We test the dynamic federated averaging algorithm on the logistic risk function with ℓ_2-norm regularization (29) using synthetic data. The local risk is given by:

$$P_k(w) = \frac{1}{N_k} \sum_{n=0}^{N_k-1} \ln(1 + e^{-\gamma h^T w}) + \rho\|w\|^2. \tag{29}$$

The data is generated as follows: We first generate a random $\boldsymbol{w}_0^\star$ and apply the random walk model (17) with $\boldsymbol{q}_i \sim \mathcal{N}(0, \sigma_q^2)$. Then, we model the change in the true parameters $\boldsymbol{w}_i^\star$ across agents by adding randomly sampled constants $\boldsymbol{c}_k \sim \mathcal{N}(0, \sigma_c^2)$, i.e., $\boldsymbol{w}_{k,i}^\star = \boldsymbol{w}_i^\star + \boldsymbol{c}_k$. Then, for each agent k, N_k random $2-$dimensional features are sampled from a Gaussian distribution with a randomly generated variance, for each time i. We generate random feature vectors $\boldsymbol{h}_{k,i}$ and assign them labels $\boldsymbol{\gamma}_{k,i} = \text{sign}(\boldsymbol{h}_{k,i}^T \boldsymbol{w}_{k,i}^\star)$. We assume we have $K = 20$ agents, with $L = 7$ active agents at each time. We set the batch sizes B_k and epoch sizes E_k to different values in the range $[10, 20]$ and $[1, 10]$, respectively.

We examine the effect of different hyperparameters on the behaviour of the algorithm. We validate the theoretical results by changing the step size; we also look at the effect of varying the variances σ_q^2 and σ_c^2. We plot the averaged mean-square-deviation (MSD) curves in log domain after performing 50 passes over the data (Figure 1). For the first experiment, we set $\sigma_c^2 = 0.1$, and we vary the step size by factor of 10 in the stationary ($\sigma_q^2 = 0$) and non-stationary case ($\sigma_q^2 = 0.01$). We observe in Figure 1a that as μ increases, the MSD increases and the convergence becomes faster; while, in Figure 1b we plot the final MSD value reached for each μ, and we observe that there is an optimal step size $\mu^o = 0.1$ that achieves the best MSD. For the second experiment, we vary σ_c^2 while fixing $\sigma_q^2 = 0$ and $\mu = 0.01$. The plots in Figure 1c indicate that the MSD increases with σ_c^2, but the convergence rate is not effected, which is expected. Finally, in the third experiment we fix $\sigma_c^2 = 0.1$ and vary σ_q^2. A similar trend is observed (Figure 1d); σ_q^2 only affects the MSD and not the convergence.

V. Conclusion

The work presented in this paper consists of developing a convergence analysis for a modified version of the FedAvg algorithm under three major challenges: non-IID data, asynchronous operation under drift, and partial agent participation. We were able to guarantee the convergence of the algorithm and identified three major components that affect its convergence: step size, agent heterogeneity, and drift variance. We were able to illustrate the theoretical results with a series of experiments.

REFERENCES

[1] A. Agarwal and J. C. Duchi, "Distributed delayed stochastic optimization," in *Advances in Neural Information Processing Systems 24*, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2011, pp. 873–881.

[2] M. Zinkevich, M. Weimer, L. Li, and A. J. Smola, "Parallelized stochastic gradient descent," in *Advances in Neural Information Processing Systems 23*, J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, Eds. Curran Associates, Inc., 2010, pp. 2595–2603.

[3] D. P. Bertsekas, "A new class of incremental gradient methods for least squares problems," *SIAM J. Optim*, vol. 7, pp. 913–926, 1996.

[4] A. H. Sayed, "Adaptation, learning, and optimization over networks," *Foundations and Trends® in Machine Learning*, vol. 7, no. 4-5, pp. 311–801, 2014.

[5] J. Chen and A. H. Sayed, "On the limiting behavior of distributed optimization strategies," in *Proc. 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Oct 2012, pp. 1535–1542.

[6] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, Jan 2009.

[7] J. C. Duchi, A. Agarwal, and M. J. Wainwright, "Dual averaging for distributed optimization: Convergence analysis and network scaling," *IEEE Transactions on Automatic Control*, vol. 57, no. 3, pp. 592–606, 2012.

[8] H. B. McMahan, E. Moore, D. Ramage, S. Hampson *et al.*, "Communication-efficient learning of deep networks from decentralized data," *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, vol. 54, pp. 1273–1282, 20–22 April 2017.

[9] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *ICML 2019 Workshop AMTL Workshop*, Long Beach, CA, USA, June 2019, pp. 1–28.

[10] V. Smith, C.-K. Chiang, M. Sanjabi, and A. Talwalkar, "Federated multi-task learning," in *Advances in Neural Information Processing Systems 30*. Long Beach, California, USA: Curran Associates Inc., December 2017, pp. 4424–4434.

[11] S. Caldas, J. Konečny, H. B. McMahan, and A. Talwalkar, "Expanding the reach of federated learning by reducing client resource requirements," *arXiv:1812.07210*, 2018.

[12] F. Sattler, S. Wiedemann, K. Mller, and W. Samek, "Robust and communication-efficient federated learning from non-i.i.d. data," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, November 2019.

[13] J. Konen, H. B. McMahan, F. X. Yu, P. Richtrik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv:1610.05492*, 2016.

[14] M. Mohri, G. Sivek, and A. T. Suresh, "Agnostic federated learning," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. Long Beach, California, USA: PMLR, 09–15 Jun 2019, pp. 4615–4625.

[15] L. Corinzia and J. M. Buhmann, "Variational federated multi-task learning," *arXiv:1906.06268*, 2019.

[16] M. Khodak, M.-F. F. Balcan, and A. S. Talwalkar, "Adaptive gradient-based meta-learning methods," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Vancouver, Canada: Curran Associates, Inc., December 2019, pp. 5915–5926.

[17] F. Chen, M. Luo, Z. Dong, Z. Li, and X. He, "Federated meta-learning with fast convergence and efficient communication," *arXiv:1802.07876*, 2019.

[18] 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 *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS 17. New York, NY, USA: Association for Computing Machinery, oct 2017, p. 11751191.

[19] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," *arXiv:1712.07557*, 2017.

[20] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private recurrent language models," in *International Conference on Learning Representations*, Vancouver, BC, Canada, April 2018, pp. 1–14.

[21] P. Jiang and G. Agrawal, "A linear speedup analysis of distributed deep learning with sparse and quantized communication," in *Advances in Neural Information Processing Systems 31*, Montreal, Canada, December 2018, pp. 2525–2536.

[22] A. Khaled, K. Mishchenko, and P. Richtrik, "First analysis of local gd on heterogeneous data," *arXiv:1909.04715*, 2019.

[23] S. U. Stich, "Local SGD converges fast and communicates little," in *ICLR 2019 - International Conference on Learning Representations*, New Orleans, USA, May 2019, pp. 1–19.

[24] J. Wang and G. Joshi, "Cooperative sgd: A unified framework for the design and analysis of communication-efficient sgd algorithms," *ArXiv:1808.07576*, 2018.

[25] "On the convergence properties of a k-step averaging stochastic gradient descent algorithm for nonconvex optimization."

[26] H. Yu, S. Yang, and S. Zhu, "Parallel restarted sgd with faster convergence and less communication: Demystifying why model averaging works for deep learning," in 33^{rd} *AAAI Conference on Artificial Intelligence*, vol. 33, no. 1. Honolulu, Hawaii, USA: AAAI Press, Palo Alto, California USA, July 2019, pp. 5693–5700.

[27] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1205–1221, 6 2019.

[28] H. Yu, R. Jin, and S. Yang, "On the linear speedup analysis of communication efficient momentum SGD for distributed non-convex optimization," in *Proceedings of the 36th International Conference on Machine Learning*. Long Beach, California, USA: PMLR, 09–15 Jun 2019, pp. 7184–7193.

[29] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization," *arXiv:1903.03934*, 2019.

[30] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," 2020.

[31] L. Liu, J. Zhang, S. H. Song, and K. B. Letaief, "Client-edge-cloud hierarchical federated learning," *arXiv:1905.06641*, 2019.

[32] S. Kaczmarz, "Approximate solution of systems of linear equations," *International Journal of Control*, vol. 57, no. 6, pp. 1269–1271, 1993.

[33] T. Strohmer and R. Vershynin, "A randomized kaczmarz algorithm with exponential convergence," *Journal of Fourier Analysis and Applications*, vol. 15, no. 2, pp. 262–278, Apr 2008.

[34] M. Jaggi, V. Smith, M. Takác, J. Terhorst, S. Krishnan, T. Hofmann, and M. I. Jordan, "Communication-efficient distributed dual coordinate ascent," in *Advances in neural information processing systems*, Montreal, Canada, December 2014, pp. 3068–3076.

[35] Y. Tang, K. Dvijotham, and S. Low, "Real-time optimal power flow," *IEEE Transactions on Smart Grid*, vol. 8, no. 6, pp. 2963–2973, 2017.

[36] C. Enyioha, S. Magnsson, K. Heal, N. Li, C. Fischione, and V. Tarokh, "On variability of renewable energy and online power allocation," *IEEE Transactions on Power Systems*, vol. 33, no. 1, pp. 451–462, 2018.

[37] A. Simonetto, A. Mokhtari, A. Koppel, G. Leus, and A. Ribeiro, "A class of prediction-correction methods for time-varying convex optimization," *IEEE Transactions on Signal Processing*, vol. 64, no. 17, pp. 4576–4591, 2016.

[38] Q. Ling and A. Ribeiro, "Decentralized dynamic optimization through the alternating direction method of multipliers," *IEEE Transactions on Signal Processing*, vol. 62, no. 5, pp. 1185–1197, 2014.

[39] "On distributed online classification in the midst of concept drifts," *Neurocomputing*, vol. 112, pp. 138 – 152, 2013, advances in artificial neural networks, machine learning, and computational intelligence.

[40] W. Xu, K. Yuan, W. Yin, and Q. Ling, "On the comparison between primal and primal-dual methods in decentralized dynamic optimization," in *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, California, USA, November 2019.

[41] E. Dall'Anese, A. Simonetto, S. Becker, and L. Madden, "Optimization and learning with information streams: Time-varying algorithms and applications," *arXiv preprint arXiv:1910.08123*, 2019.

[42] N. Bastianello, A. Ajalloeian, and E. Dall'Anese, "Distributed and inexact proximal gradient method for online convex optimization," *arXiv preprint arXiv:2001.00870*, 2020.