2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Online Distributed Learning with Quantized Finite-Time Coordination

Nicola Bastianello, Apostolos I. Rikos, Karl H. Johansson

Abstract— **In this paper we consider online distributed learning problems. Online distributed learning refers to the process of training learning models on distributed data sources. In our setting a set of agents need to cooperatively train a learning model from streaming data. Differently from federated learning, the proposed approach does not rely on a central server but only on peer-to-peer communications among the agents. This approach is often used in scenarios where data cannot be moved to a centralized location due to privacy, security, or cost reasons. In order to overcome the absence of a central server, we propose a distributed algorithm that relies on a *quantized, finite-time coordination* protocol to aggregate the locally trained models. Furthermore, our algorithm allows for the use of *stochastic gradients* during local training. Stochastic gradients are computed using a randomly sampled subset of the local training data, which makes the proposed algorithm more efficient and scalable than traditional gradient descent. In our paper, we analyze the performance of the proposed algorithm in terms of the mean distance from the online solution. Finally, we present numerical results for a logistic regression task.**

I. Introduction

Recent technological advances have led to the widespread implementation of multi-agent systems in several applications, ranging from power grids to robotics, from traffic to sensor networks, to name a few [1], [2]. These systems, composed of interconnected agents equipped with computational and communication resources, enable the collection of data on many different phenomena at an unprecedented level of detail [3]. Processing these data in order to train machine learning models has therefore become a research objective of central importance, around which *federated learning* (FL) was born [3], [4].

Differently from traditional machine learning approaches in which data are collected and processed at a single location, the goal of FL is to enable a cooperative learning process that does not require the agents to directly share private data. The usual architecture of federated learning set-ups includes a set of agents that store and locally process data, and a *fusion center* that communicates with them and aggregates the results of their computations [5]. However, in many applications it may not be possible (or desirable) to implement such an architecture. Indeed, the fusion center is a single point of failure, and (temporarily) removing it halts the learning process entirely. For this reason, researchers have recently focused on developing fully distributed solutions. In these solutions agents aggregate their results without needing a fusion center [6]. This paper follows the same line of research and focuses on designing *distributed* learning algorithms. In these algorithms agents employ peer-to-peer communications to share and aggregate the results of local processing. This set-up builds the foundation of this paper, and is commonly assumed in *distributed optimization*, [2], [7].

Relying on peer-to-peer communications instead of a fusion center introduces many advantages for FL algorithms (e.g., enhances robustness) but poses bigger design challenges. Indeed, a fusion center enables dissemination of information (*e.g.* local computations) to all agents in a single communication round. On the contrary, peer-to-peer communications require multiple communication rounds. Several classes of distributed algorithms have been proposed to overcome this challenge. In particular, distributed (sub-)gradient methods, gradient tracking methods, and primal-dual methods [8]. While the gradient-based approaches usually rely on average consensus as a coordination and aggregation technique, in this paper we explore the use of *finite-time coordination* (FTC) [9] as a peer-to-peer substitute for the fusion center. In particular, the proposed algorithm employs FTC to aggregate local gradient descent steps (as computed by the agents on the data they store) through multiple rounds of communications. This approach is similar to Near-DGD [10] (which however employs average consensus).

Decentralized learning algorithms (either relying on a fusion center or peer-to-peer coordination among agents) face the major challenge of operating over networks with limited communication constraints. Specifically, in practice communication between agents is done via bandwidth limited channels. For example, the usage of wireless networks [11], or the training high dimensional models, would significantly increase the communication burden [12]. Therefore, guaranteeing efficient communication among agents is one of the main challenges for FL algorithms [4]. One of the approaches for guaranteeing efficient communication is via *quantization*. The basic idea of quantization is to represent the numerical values used in the model's parameters using fewer bits than their full precision, while still preserving a reasonable level of accuracy. Addressing the constraint of limited communication by utilizing quantized communication among agents is built into our proposed algorithm. Specifically, the finite-time coordination protocol only employs quantized communication. We remark that, differently

This work was partially supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 101070162, and partially by Swedish Research Council Distinguished Professor Grant 2017-01078 Knut and Alice Wallenberg Foundation Wallenberg Scholar Grant.

N. Bastianello and K. H. Johansson are with the School of Electrical Engineering and Computer Science and Digital Futures, KTH Royal Institute of Technology, Sweden, { nicolba | kallej}@kth.se.

Apostolos I. Rikos is with the Department of Electrical and Computer Engineering, Division of Systems Engineering, Boston University, Boston, MA 02215, US. E-mail: arikos@bu.edu.

from distributed algorithms that use average consensus-based aggregation [13] (as also Near-DGD does), the proposed algorithm is robust to quantized communications, reaching an approximate solution to the learning problem. This approximation depends on the utilized quantization level. Moreover, our algorithm can be deployed on *directed networks*. This deployment usually requires specialized reformulation of gradient-based algorithms [14], [15].

Combining the aforementioned characteristics, in our paper we present a distributed learning algorithm. During the operation of our algorithm agents coordinate by exchanging quantized messages via a finite-time coordination protocol. We also present a convergence analysis for the proposed algorithm which characterizes the effect of quantization on the accuracy of the learned model, providing a guide to tune the quantization level. Additionally, our convergence analysis accounts for the effect of *stochastic gradients*. Stochastic gradients are inexact gradients computed by the agents using only a portion of the local data [15]. This approach reduces the computational burden and speeds up the training process. In this paper we also characterize how stochastic gradients degrade the accuracy of the proposed method. Finally, our analysis in this paper also accounts for time-variability of the local costs. Time-variability of local costs refers to the fact that the agents' local cost functions associated with different training examples in a dataset can vary over time. Indeed, in many applications, as agents continuously collect new data, local cost functions may vary over time. Thus, the goal of every agent in the network becomes that of solving an *online learning* problem [16], [17], [18].

The main contributions of our paper are the following:

1) We present a distributed projected gradient-based learning algorithm. Differently from federated learning, the agents do not rely on a central coordinator. Coordination is achieved by a quantized finite-time coordination protocol. Our proposed algorithm employs quantized communication between agents and accounts for the effect of stochastic gradients (see Algorithm 1).
2) We analyze our algorithm's convergence to the optimal solution. Specifically, in our analysis we illustrate the effect on the performance that quantized communications, stochastic gradient, and time-varying costs have (see Proposition 1).

Outline: In section II we formulate the problem and describe the set-up of interest. Section III presents and discussed the proposed algorithm, whose convergence is analyzed in section IV. Section V presents numerical results for a logistic regression task.

Notation: We denote with $\mathbb{N}$ and $\mathbb{R}$ the set of integer and real numbers, respectively. For any $a \in \mathbb{R}$, the greatest integer less than or equal to a is denoted $\lfloor a \rfloor$, while the smallest integer greater than or equal to a is denoted as $\lceil a \rceil$. We denote a directed graph (digraph) of N agents by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with $\mathcal{V} = \{1, \dots, N\}$ and $(i, j) \in \mathcal{E}$ if i can transmit information to j. We denote the in-neighbors of i by $\mathcal{N}_i^- = \{j \in \mathcal{V} \mid (i, j) \in \mathcal{E}\}$, and the out-neighbors by $\mathcal{N}_i^+ = \{l \in \mathcal{V} \mid (l, i) \in \mathcal{E}\}$. A directed path from i to j of length t exists if we can find a sequence of agents $i \equiv j_0, j_1, \dots, j_t \equiv j$ such that $(j_{\tau+1}, j_\tau) \in \mathcal{E}$ for $\tau = 0, 1, \dots, t-1$. A digraph is *strongly connected* if there exists a directed path from every agent i to every agent j in the network, for every $i, j \in \mathcal{V}$. The diameter D of a digraph is the longest shortest path between any two agents $i, j \in \mathcal{V}$ in the network.

II. Problem Formulation

Given the digraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with N agents, our goal is to solve the online optimization problem

$$x_k^* = \arg\min_{x \in \mathbb{R}^n} \sum_{i=1}^{N} f_{i,k}(x) \tag{1}$$

where $f_{i,k} : \mathbb{R}^n \to \mathbb{R}$ are the time-varying local costs, each privately held by one of the agents. Since we are specifically interested in learning applications, in the following we consider local costs of the form

$$f_{i,k}(x) = \frac{1}{m_i} \sum_{h=1}^{m_i} \ell(x; d_{i,k}^h) \tag{2}$$

where $\ell : \mathbb{R}^n \to \mathbb{R}$ is a loss function (*e.g.* quadratic or logistic loss) and $\{d_{i,k}^h\}_{h=1}^{m_i}$ are the data points available to agent i at time k (*e.g.* pairs of feature vector and label for a classification task). The problem we face then is time-varying as the local costs depend on time-varying data.

Let now $x_i \in \mathbb{R}^n$ be the local state of $i \in \mathcal{V}$, and let $\boldsymbol{x} = [x_1^\top, \dots, x_N^\top]^\top \in \mathbb{R}^{nN}$. Then we can define the *consensus set* as

$$\mathcal{C} = \{\boldsymbol{x} \in \mathbb{R}^{nN} \mid x_i = x_j \ \forall i, j \in \mathcal{V}\},$$

and we can rewrite (1) as the following *distributed online optimization* problem

$$\boldsymbol{x}_k^* = \arg\min_{\boldsymbol{x} \in \mathbb{R}^{nN}} \sum_{i=1}^{N} f_{i,k}(x_i) \quad \text{s.t. } \boldsymbol{x} \in \mathcal{C}. \tag{3}$$

If the digraph is strongly connected (cf. Assumption 1 below), then it is clear that problems (1) and (3) are equivalent, and $\boldsymbol{x}_k^* = \mathbf{1}_N \otimes x_k^*$. Our goal in the following will be to design an online distributed algorithm that solves (3) making use of a *finite-time coordination routine*.

We introduce now some assumptions that will hold throughout our paper.

Assumption 1 (Network): The digraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is strongly connected. Every agent in the network knows the diameter (or an upper bound thereof) D of the digraph.

Assumption 2 (Costs): The local cost $f_{i,k} : \mathbb{R}^n \to \mathbb{R}$ is $\underline{\lambda}$-strongly convex and $\bar{\lambda}$-smooth for each agent $i \in \mathcal{V}$ and time $k \in \mathbb{N}$.

Assumption 3 (Bounded time-variability): Problem (3) is such that there exists $\sigma \geq 0$ such that

$$\left\| \boldsymbol{x}_k^* - \boldsymbol{x}_{k-1}^* \right\| \leq \sigma, \quad \forall k \in \mathbb{N}.$$

As mentioned above, Assumption 1 guarantees the equivalence of (1) and (3), so that the former can be solved in a distributed fashion. Moreover, assuming that the agents know an (upper bound to) the diameter D of the digraph enables them to deploy the finite time coordination algorithm. We

remark that it is possible to compute D in a distributed fashion via max-consensus protocols, see *e.g.* [19], [20].

Assumptions 2 and 3 are fairly standard in online optimization, see *e.g.* [16], [17], and guarantee existence and uniqueness of the optimal trajectory $\{\boldsymbol{x}_k^*\}_{k\in\mathbb{N}}$ and a bounded distance between consecutive points in the trajectory, respectively. Intuitively, Assumption 3 implies that consecutive problems are "similar" so that computation performed at time k can be repurposed towards an improved solution of the problem at $k+1$.

We remark that, due to the dynamic nature of the problem (3), convergence to the exact solution trajectory in general cannot be achieved [16]. Rather, the proposed algorithm will converge to a neighborhood thereof, as characterized by the results presented in section IV.

III. Algorithm

Conceptually, one could solve problem (3) by applying the *online projected gradient* method [21] characterized by

$$\boldsymbol{x}_{k+1} = \mathrm{proj}_{\mathcal{C}}\left(\boldsymbol{x}_k - \alpha\nabla f_k(\boldsymbol{x}_k)\right), \quad k\in\mathbb{N}, \tag{4}$$

where $\nabla f_k(\boldsymbol{x}) = [\nabla f_{1,k}(x_1)^\top, \dots, \nabla f_{N,k}(x_N)^\top]^\top$ collects the local gradients, and the projection onto the consensus space is $\mathrm{proj}_{\mathcal{C}}(\boldsymbol{x}) = \frac{1}{N}\sum_{i=1}^N x_i$. However, in order for the algorithm we design to be distributed, we need to design a distributed implementation of the projection $\mathrm{proj}_{\mathcal{C}}$. The idea, following [9], is to implement such projection using the finite-time coordination routine outlined in Algorithm 2. Before delving into the details, we outline in the following section two challenges to which the proposed algorithm is subject: *(i) limited communication*, and *(ii) inexact gradients*.

A. Challenges

Learning problems are often *large-scale*, both due to the fact that the (local) costs depend on a huge number of data points, and because the model to be learned is high-dimensional. These features of the problem pose very stringent constraints on the design of solution algorithms, as discussed in the following.

(i) Limited communication: Distributedly learning a high-dimensional model (such as deep neural networks) translates into the solution of problem (3) with a large $n \gg 1$. But the agents need to perform consensus on the local gradient steps $\mathbb{R}^n \ni y_{i,k} = x_{i,k} - \alpha\nabla f_{i,k}(x_{i,k})$, demanding high communication rates. To alleviate the communication cost, different schemes can be implemented, foremost of which is *quantization* [3], [4], which will be employed by the finite-time coordination protocol of Algorithm 2. As a consequence, the agents compute only an approximate projection onto the consensus set.

(ii) Inexact gradients: Besides the high-dimensionality of the learning problem (3), the local costs are also defined on high-dimensional data-sets [4], recalling (2):

$$f_{i,k}(x) = \frac{1}{m_i}\sum_{h=1}^{m_i} \ell(x; d_{i,k}^h)$$

where the number of data points available at any time k, m_i, can be very large. Computing the exact local gradient $\nabla f_{i,k}(x)$ may have too high a computational cost, and in learning applications they are replaced by *stochastic gradients*, as computed on a randomly selected sub-set of the local data [22]. In the following we suppose that agents compute only approximate gradients $\hat{\nabla} f_{i,k}(x)$, whose inexactness is characterized by the following assumption [22].

Assumption 4 (Stochastic gradients): The agents compute the approximate local gradient $\hat{\nabla} f_{i,k}(x)$, which satisfies

$$\mathbb{E}\left[\left\|\hat{\nabla} f_{i,k}(x) - \nabla f_{i,k}(x)\right\|\right] \le \tau.$$

B. Algorithm

Accounting for the challenges discussed above, we can finally characterize the distributed implementation of the online projected gradient (4) with Algorithm 1.

Algorithm 1 FTQC-DGD

Input: For each agent $i\in\mathcal{V}$ initialize $x_{i,0}$; choose the step-size α.

1: **for** $k = 0, 1, \dots$ each agent i **do**

// local update

2: receive a new local cost $f_{i,k}$

3: apply the local (possibly inexact) gradient step

$$y_{i,k} = x_{i,k} - \alpha\hat{\nabla} f_{i,k}(x_{i,k})$$

// coordination

4: apply the finite-time coordination Algorithm 2 to approximately project onto the consensus space

$$\boldsymbol{x}_{k+1} = \hat{\mathrm{proj}}_{\mathcal{C}}(\boldsymbol{y}_k)$$

with $\boldsymbol{y}_k = [y_{1,k}^\top, \dots, y_{N,k}^\top]^\top$

5: **end for**

The finite-time coordination routine employed to approximate the projection onto the consensus set is described in Algorithm 2. Note here that Algorithm 2 is executed until it converges to the quantized average of each agent's estimate. Therefore, by calculating the quantized average we are able to approximate the projection onto the consensus set. For simplicity we report it for scalar states, while in practice the agents apply Algorithm 2 to each component of the vectors $\{y_{i,k}\}_{i\in\mathcal{V}}$ in parallel.

a) Algorithm 1: First of all, we remark that by design Algorithm 1 can be interpreted as an inexact projected gradient descent method, and be written in the abstract form

$$\boldsymbol{x}_{k+1} = \mathrm{proj}_{\mathcal{C}}\left(\boldsymbol{x}_k - \alpha\nabla f_k(\boldsymbol{x}_k)\right) + \boldsymbol{e}_k. \tag{5}$$

The (random) vector $\boldsymbol{e}_k$ accounts for all sources of inexactness discussed in section III-A, namely the quantization error introduced by Algorithm 2 and the error due to inexact gradients. This interpretation of Algorithm 1 will prove to be particularly suited to studying its convergence in section IV. We further remark that – following the literature on *online learning* [23] – the proposed algorithm operates

Algorithm 2 Finite-time quantized coordination (FTQC)

Input: The states to be averaged $\{x_i\}_{i\in\mathcal{V}}$, quantization level Δ, and upper bound to the diameter D.

// initialization

each agent $i \in \mathcal{V}$ sets $y_i = 2\lfloor x_i/\Delta \rfloor$ and $z_i = 2$

for $\ell = 1, 2, \dots$ each agent i **do**

// local update and transmission

while $z_i > 1$ **do**

compute $t = \lfloor y_i/z_i \rfloor$, and update $y_i = y_i - t$ and $z_i = z_i - 1$

select an out-neighbor $j \in \mathcal{N}_i^+$ uniformly at random

transmit t to j, which updates $y_j = y_j + t$ and $z_j = z_j + 1$

end while

// vote on termination

if $\ell \mod D = 1$ **then**

compute $\bar{m}_i = \lceil y_i/z_i \rceil$ and $\underline{m}_i = \lfloor y_i/z_i \rfloor$, and broadcast them to the out-neighbors $j \in \mathcal{N}_i^+$

receive $\bar{m}_j$, $\underline{m}_j$ from the in-neighbors $j \in \mathcal{N}_i^-$ and set $\bar{m}_i = \max_{j\in\mathcal{N}_j^-\cup\{i\}} \bar{m}_j$, $\underline{m}_i = \min_{j\in\mathcal{N}_j^-\cup\{i\}} \underline{m}_j$

if $\bar{m}_i - \underline{m}_i \leq 1$ **then**

set $x_i = \Delta \underline{m}_i$ and terminate

end if

end if

end for

in a *predictive* fashion. Indeed, the local costs $\{f_{i,k}\}_{k\in\mathbb{N}}$ are used to *predict the solution to the problem at time* $k+1$ [16, p. 73].

b) Algorithm 2: It is important to notice that the finite-time coordination routine of Algorithm 2 is designed to be deployed over quantized communications – indeed the agents only share integers. As a consequence, the accuracy of the projection onto $\mathcal{C}$ that it returns only depends on the quantization level Δ.

Notice that the agents perform a vote on the termination of the routine every D iterations, during which the agents perform a max- and a min-consensus to assess if their states have converged on an integer, thus signifying that the quantized consensus has been reached. Indeed, Algorithm 2 enables the agents to compute the asymmetrically quantized average of their initial states, that is

$$\left\lfloor \frac{\bar{x}}{\Delta} \right\rfloor \Delta, \quad \bar{x} := \frac{1}{N}\sum_{i=1}^{N} x_i.$$

This is achieved with high probability, as proved in [9], [24] by showing that Algorithm 2 can be modeled as a random walk of multiple tokes around the digraph.

Lemma 1 (Coordination error): Let $\boldsymbol{x}_{k+1} = \mathbf{1}_N \otimes x_{k+1}$, be the output of Algorithm 2 as applied in Algorithm 1 (line 5) to distributedly approximate $\mathrm{proj}_{\mathcal{C}}$. Then it holds that

$$\|\mathrm{proj}_{\mathcal{C}}(\boldsymbol{y}_k) - \boldsymbol{x}_{k+1}\| \leq 2\Delta\sqrt{nN} =: \gamma.$$

Proof: By definition $\mathrm{proj}_{\mathcal{C}}(\boldsymbol{y}_k) = \frac{1}{N}\sum_{i=1}^{N} y_{i,k}$ and, since Algorithm 2 converges to the asymmetrically quantized average consensus, we can write

$$\begin{aligned}\|\mathrm{proj}_{\mathcal{C}}(\boldsymbol{y}_k) - \boldsymbol{x}_{k+1}\|^2 &= \sum_{i=1}^{N} \left\| \frac{1}{N}\sum_{i=1}^{N} y_{i,k} - x_{k+1} \right\|^2 \\ &\leq \sum_{i=1}^{N} n(2\Delta)^2 = nN(2\Delta)^2\end{aligned}$$

where the inequality holds the fact that each of the n components of the vector $\frac{1}{N}\sum_{i=1}^{N} y_{i,k} - x_{k+1}$ are at most 2Δ apart. The thesis follows by taking the square root. ■

C. Alternative $\mathrm{proj}_{\mathcal{C}}$ implementations

In this work we have resorted to a finite-time coordination routine to (approximately) implement the projection $\mathrm{proj}_{\mathcal{C}}$. However, different approaches have been explored in the literature on distributed optimization and federated learning, and it is instructive to briefly discuss them.

a) Average consensus: Assuming that the graph is undirected[1], and letting $\boldsymbol{W}$ be a doubly stochastic matrix for the network, one may choose to approximate $\mathrm{proj}_{\mathcal{C}}$ with a number $t \in \mathbb{N}$ of consensus steps for each time k, yielding the *online version of Near-DGD* [10]

$$\boldsymbol{x}_{k+1} = \boldsymbol{W}^t \left(\boldsymbol{x}_k - \alpha\nabla f_k(\boldsymbol{x}_k)\right), \quad k \in \mathbb{N}. \tag{6}$$

As characterized in [10, Lemma V.2], $\boldsymbol{W}^t$ is indeed an inexact projection onto $\mathcal{C}$, whose error decreases as t increases (as a consequence of the well known convergence of the average consensus).

b) Gradient tracking: Rather than (approximately) projecting local gradient steps onto the consensus set, the extensively studied approach of *gradient tracking* can be used instead, see *e.g.* [15]. This paradigm performs a projection onto $\mathcal{C}$ asymptotically, by introducing additional variables and using average consensus. In the context of this paper, it is interesting to notice that *online versions of gradient tracking methods may actually be less performant than "DGD-style" methods* such as (4) and (6), see [18], [26].

c) Centralized aggregation: In *federated learning* a widely used assumption is that a central server is available, with which all agents can communicate, and which has the task of aggregating the local updates (*e.g.* via weighted averaging as in FedAvg [5]). When all the agents provide a local gradient at time k, the server can then perform an exact projection onto $\mathcal{C}$. However, this approach relies on the central server, which becomes a single point of failure. In this paper we are interested in fully distributed algorithms that rely on peer-to-peer communications.

IV. Convergence Analysis

In section III-A we introduced the abstract characterization of Algorithm 1 as the inexact online projected gradient (5). It is straightforward to see that, by adding and subtracting

[1]If this is not the case, a *push-sum* or *ratio* consensus algorithm can be employed instead [25].

the right vectors, we can indeed write (5) with $\boldsymbol{e}_k = \boldsymbol{e}_k^p + \boldsymbol{e}_k^g$ where

$$\boldsymbol{e}_k^p := \hat{\text{proj}}_{\mathcal{C}}(\boldsymbol{x}_k - \alpha\hat{\nabla} f_k(\boldsymbol{x}_k)) - \text{proj}_{\mathcal{C}}(\boldsymbol{x}_k - \alpha\hat{\nabla} f_k(\boldsymbol{x}_k))$$
$$\boldsymbol{e}_k^g := \text{proj}_{\mathcal{C}}(\boldsymbol{x}_k - \alpha\hat{\nabla} f_k(\boldsymbol{x}_k)) - \text{proj}_{\mathcal{C}}(\boldsymbol{x}_k - \alpha\nabla f_k(\boldsymbol{x}_k))$$

with $\boldsymbol{e}_k^p$ modeling the inexact projection onto the consensus set $\mathcal{C}$ due to the use of the FTQC scheme in Algorithm 2, and $\boldsymbol{e}_k^g$ the use of approximate local gradients. The following result analyzes the convergence of (5) in mean, accounting for the effects of time-variability of the costs and of $\boldsymbol{e}_k$.

Proposition 1 (Convergence in mean): Consider the online distributed problem (3), and suppose that Assumptions 1–4 hold. Let $\{\boldsymbol{x}_k\}_{k\in\mathbb{N}}$ be the trajectory generated by Algorithm 1, then it holds that for all $k > 0$:

$$\mathbb{E}\left[\|\boldsymbol{x}_k - \boldsymbol{x}_k^*\|\right] \leq \zeta^k \|\boldsymbol{x}_0 - \boldsymbol{x}_0^*\| + (\sigma + \gamma + \alpha\tau)\frac{1-\zeta^{k+1}}{1-\zeta}$$

where $\zeta = \max\{|1-\alpha\underline{\lambda}|, |1-\alpha\bar{\lambda}|\} \in (0,1)$, and σ, γ, τ are defined in Assumption 3, Lemma 1, and Assumption 4, respectively.

Proof: It is straightforward to see that we can write (5)

$$\begin{aligned}\boldsymbol{x}_{k+1} - \boldsymbol{x}_{k+1}^* = \text{proj}_{\mathcal{C}}\left(\boldsymbol{x}_k - \alpha\nabla f_k(\boldsymbol{x}_k)\right) - \boldsymbol{x}_k^* + \\ + \left(\boldsymbol{x}_k^* - \boldsymbol{x}_{k+1}^* + \boldsymbol{e}_k^p + \boldsymbol{e}_k^g\right)\end{aligned}$$

where we introduced an additional error $(\boldsymbol{x}_k^* - \boldsymbol{x}_{k+1}^*)$ due to the fact that the algorithm is predictive and approximates $\boldsymbol{x}_{k+1}^*$ using the costs at time k. Taking the norm, redefining $\boldsymbol{e}_k = \boldsymbol{x}_k^* - \boldsymbol{x}_{k+1}^* + \boldsymbol{e}_k^p + \boldsymbol{e}_k^g$, and using the triangle inequality we write

$$\begin{aligned}\left\|\boldsymbol{x}_{k+1} - \boldsymbol{x}_{k+1}^*\right\| &\leq \|\text{proj}_{\mathcal{C}}\left(\boldsymbol{x}_k - \alpha\nabla f_k(\boldsymbol{x}_k)\right) - \boldsymbol{x}_k^*\| + \|\boldsymbol{e}_k\| \\ &\overset{(i)}{\leq} \zeta\|\boldsymbol{x}_k - \boldsymbol{x}_k^*\| + \|\boldsymbol{e}_k\|, \end{aligned} \tag{7}$$

with $\zeta = \max\{|1-\alpha\underline{\lambda}|, |1-\alpha\bar{\lambda}|\} \in (0,1)$, where (i) follows since by Assumption 2 problem (3) is $\underline{\lambda}$-strongly convex and $\bar{\lambda}$-smooth [27] and the fact that $\boldsymbol{x}_k^* = \text{proj}_{\mathcal{C}}(\boldsymbol{x}_k^* - \alpha\nabla f_k(\boldsymbol{x}_k^*))$. Taking the expected value we have $\mathbb{E}\left[\left\|\boldsymbol{x}_{k+1} - \boldsymbol{x}_{k+1}^*\right\|\right] \leq \zeta\mathbb{E}\left[\|\boldsymbol{x}_k - \boldsymbol{x}_k^*\|\right] + \mathbb{E}\left[\|\boldsymbol{e}_k\|\right]$; we bound now the three terms in $\mathbb{E}\left[\|\boldsymbol{e}_k\|\right]$ separately.

By Assumption 3 we know that $\left\|\boldsymbol{x}_k^* - \boldsymbol{x}_{k+1}^*\right\| \leq \sigma$, and by Lemma 1 we know that $\mathbb{E}\left[\|\boldsymbol{e}_k^p\|\right] \leq \gamma$. Finally, by Assumption 4 we get

$$\begin{aligned}\mathbb{E}\left[\boldsymbol{e}_k^g\right] &\overset{(i)}{\leq} \mathbb{E}\left[\left\|\boldsymbol{x}_k - \alpha\hat{\nabla} f_k(\boldsymbol{x}_k) - (\boldsymbol{x}_k - \alpha\nabla f_k(\boldsymbol{x}_k))\right\|\right] = \\ &= \alpha\mathbb{E}\left[\left\|\hat{\nabla} f_k(\boldsymbol{x}_k) - \nabla f_k(\boldsymbol{x}_k)\right\|\right] \leq \alpha\tau\end{aligned}$$

where (i) holds since the projection is 1-Lipschitz continuous [28, Proposition 4.16]. Iterating (7) and using the geometric series yields the thesis. ∎

Notice that taking the limit for $k \to \infty$ from Proposition 1 we get

$$\lim_{k\to\infty} \mathbb{E}\left[\|\boldsymbol{x}_k - \boldsymbol{x}_k^*\|\right] \leq \frac{\sigma + \gamma + \alpha\tau}{1-\zeta}. \tag{8}$$

We can then see how the different sources of "inexactness" – time-variability of the costs, inexact coordination due to the use of Algorithm 2, approximated gradients – play a role in the bound for the asymptotic tracking error $\lim_{k\to\infty}\mathbb{E}\left[\|\boldsymbol{x}_k - \boldsymbol{x}_k^*\|\right]$. We remark in particular that the bound τ is multiplied by the step-size α, which implies that the smaller we choose it, the less the effect of inexact gradients; however, the smaller α is the larger ζ, implying a trade-off[2].

Remark 1: We remark that the results can be easily extended to allow for a different bound $\left\|\boldsymbol{x}_k^* - \boldsymbol{x}_{k-1}^*\right\| \leq \sigma_k$ at each time $k \in \mathbb{N}$.

V. Numerical Results

In this section we consider the *(online) logistic regression* problem characterized by problem (1) with the local cost functions

$$f_{i,k}(x) = \sum_{h=1}^{m_i} \log\left(1 + \exp\left(-b_{i,k}^h a_{i,k}^h x\right)\right) + \frac{\epsilon}{2}\|x\|^2, \tag{9}$$

with agent $i \in \mathcal{V}$ at time $k \in \mathbb{N}$ storing the private data $\{(a_{i,k}^h, b_{i,k}^h)\}_{h=1}^{m_i}$, with $a_{i,k}^h \in \mathbb{R}^{1\times n}$ and $b_{i,k}^h \in \{-1, 1\}$, and $\epsilon = 5$. The network is composed of $N = 10$ agents, the number of unknowns (including the intercept) is $n = 16$, and each agent stores $m_i = 20$ data points.

Following the discussion of section III-C, we will compare FTQC-DGD with Near-DGD [10] and the distributed gradient tracking (DGT) method [18]. All three methods use quantized communications, and are given the same communication budget to ensure a fair comparison[3].

A. Quantization

We start evaluating the performance of the three algorithms by applying them to a static logistic regression problem (the data in (9) do not change) and for different quantization levels Δ.

First of all, in Figure 1 we report the error trajectory $\{\|\boldsymbol{x}_k - \boldsymbol{x}^*\|\}_{k\in\mathbb{N}}$ generated by the three algorithms when the communications are quantized with $\Delta = 0.01$. As previously observed in *e.g.* [13], gradient tracking methods are not robust to quantization, which leads DGT to diverge. FTQC-DGD and Near-DGD both converge to a neighborhood of the optimal solution, in accordance with the results of section IV, with the neighborhood reached by the former being smaller.

In Table I we report the asymptotic error[4] for Near-DGD and FTQC-DGD, varying the quantization level Δ. As we can see, the proposed algorithm consistently achieves smaller errors, with the error being one order of magnitude smaller for values $\Delta \leq 0.1$.

[2]One option to choose the step-size then is to find the value of α that minimizes the asymptotic error bound given in (8).

[3]In practice, this means that Near-DGD and DGT perform the same number of communication rounds as required in FTQC-DGD. For DGT these communications are split among the two steps characterizing the algorithm.

[4]Computed as the maximum error in the last $4/5$ of the simulation.

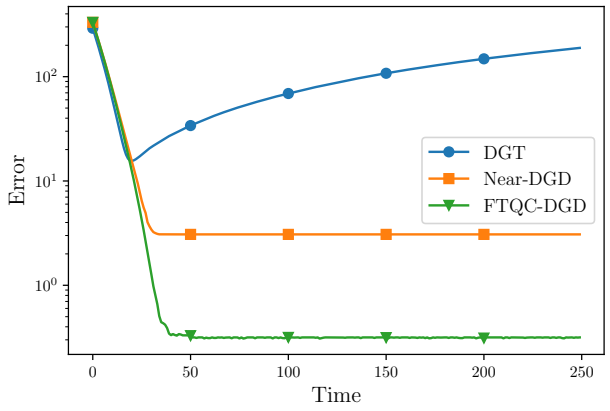

Fig. 1. Comparison of the proposed FTQC-DGD with Near-DGD and DGT, using quantized communications with $\Delta = 0.01$.

TABLE I

ASYMPTOTIC ERROR ACHIEVED BY NEAR-DGD AND FTQC-DGD FOR DIFFERENT QUANTIZATION LEVELS.

Quantization	Near-DGD	FTQC-DGD
$\Delta = 10^{-5}$	4.49×10^{-3}	3.79×10^{-4}
$\Delta = 10^{-4}$	3.51×10^{-2}	3.65×10^{-3}
$\Delta = 10^{-3}$	2.18×10^{-1}	3.43×10^{-2}
$\Delta = 10^{-2}$	3.07	3.16×10^{-1}
$\Delta = 10^{-1}$	41.91	3.72
$\Delta = 1$	236.58	124.20
$\Delta = 10$	566.37	414.18

B. Stochastic gradients

In this section we apply the proposed algorithm to the static logistic regression problem, with quantization level fixed to $\Delta = 10^{-4}$, and are interested in evaluating its performance when the agents can only access inexact local gradient. As discussed in section III-A, when the local cost (9) depends on a large number of data points ($m_i \gg 1$), then evaluating a local gradient may be too computationally expensive. Instead, an agent i each time selects a subset of its data, $\mathcal{H} \subset \{1, \dots, m_i\}$, uniformly at random, with $b := |\mathcal{H}| < m_i$ being the batch size. The agent can then approximately evaluate the gradient by computing [15] $\hat{\nabla} f_{i,k}(x) = \sum_{h \in \mathcal{H}} \nabla \log \left(1 + \exp\left(-b^h_{i,k} a^h_{i,k} x\right)\right) + \epsilon x$.

In Table II we report the asymptotic error attained by the proposed algorithm for different batch sizes. Clearly, the smaller the batch size the worse $\hat{\nabla} f_{i,k}(x)$ approximates the true gradient, compounding the quantization error with a second source of inexactness.

C. Online optimization

We conclude this section by evaluating the performance of the proposed algorithm when applied to an online logistic regression problem, in which the local cost functions change every 100 iterations.

In Figure 2 we report the tracking error $\{\|\boldsymbol{x}_k - \boldsymbol{x}^*_k\|\}_{k \in \mathbb{N}}$ of FTQC-DGD for different values of the quantization level Δ. As we can see, while the problem does not change, the algorithm converges to a neighborhood of the optimal solution, whose radius depends on the quantization error (as discussed in section V-A). Each time the problem changes, the tracking error spikes, due to the fact that the new optimal solution is different from the previous one and the algorithm undergoes a new transient. As characterized in the results of section IV, the upper bound to the tracking error indeed depends on the maximum distance between consecutive optima.

TABLE II

ASYMPTOTIC ERROR ACHIEVED BY FTQC-DGD FOR DIFFERENT BATCH SIZES.

Batch size	As. error
$b = m_i/2$	2.37
$b = 11$	2.05
$b = 12$	1.82
$b = 13$	1.68
$b = 14$	1.38
$b = 15$	1.19
$b = 16$	9.78×10^{-1}
$b = 17$	7.51×10^{-1}
$b = 18$	6.05×10^{-1}
$b = 19$	3.66×10^{-1}
$b = m_i$	3.45×10^{-3}

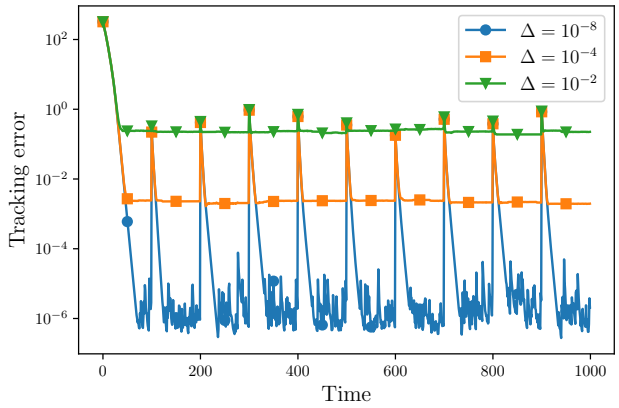

Fig. 2. Tracking error of FTQC-DGD applied to an online logistic regression problem, with different quantization levels.

VI. CONCLUSIONS AND FUTURE DIRECTIONS

In this paper we proposed a novel gradient-based algorithm for distributed learning over fully decentralized, directed networks. In particular, the algorithm substitutes a fusion center (employed in federated learning) with a finite-time coordination protocol. We analyzed the performance of the algorithm accounting for the use of quantized communications and stochastic gradients, as well as the impact of time-varying costs. Future work will look into comparing the use of finite-time coordination with different quantization schemes, and applying the proposed approach to more general problems such as composite and constrained optimization problems.

REFERENCES

[1] D. K. Molzahn, F. Dorfler, H. Sandberg, S. H. Low, S. Chakrabarti, R. Baldick, and J. Lavaei, "A Survey of Distributed Optimization and

Control Algorithms for Electric Power Systems," *IEEE Transactions on Smart Grid*, vol. 8, no. 6, pp. 2941–2962, Nov. 2017.

[2] A. Nedić and J. Liu, "Distributed Optimization for Control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, pp. 77–103, May 2018.

[3] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated Learning: Challenges, Methods, and Future Directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, May 2020.

[4] T. Gafni, N. Shlezinger, K. Cohen, Y. C. Eldar, and H. V. Poor, "Federated Learning: A signal processing perspective," *IEEE Signal Processing Magazine*, vol. 39, no. 3, pp. 14–41, May 2022.

[5] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, A. Singh and J. Zhu, Eds., vol. 54. Fort Lauderdale, FL, USA: PMLR, Apr. 2017, pp. 1273–1282.

[6] S. A. Alghunaim and K. Yuan, "A unified and refined convergence analysis for non-convex decentralized learning," *IEEE Transactions on Signal Processing*, vol. 70, pp. 3264–3279, 2022.

[7] T. Yang, X. Yi, J. Wu, Y. Yuan, D. Wu, Z. Meng, Y. Hong, H. Wang, Z. Lin, and K. H. Johansson, "A survey of distributed optimization," *Annual Reviews in Control*, vol. 47, pp. 278–305, 2019.

[8] G. Notarstefano, I. Notarnicola, and A. Camisa, "Distributed Optimization for Smart Cyber-Physical Networks," *Foundations and Trends® in Systems and Control*, vol. 7, no. 3, pp. 253–383, 2019.

[9] A. I. Rikos, W. Jiang, T. Charalambous, and K. H. Johansson, "Distributed optimization with gradient descent and quantized communication," in *Proceedings of* 22^{nd} *IFAC World Congress*, 2023 (Accepted for publication).

[10] A. S. Berahas, R. Bollapragada, N. S. Keskar, and E. Wei, "Balancing Communication and Computation in Distributed Optimization," *IEEE Transactions on Automatic Control*, vol. 64, no. 8, pp. 3141–3155, Aug. 2019.

[11] L. Qian, P. Yang, M. Xiao, O. A. Dobre, M. D. Renzo, J. Li, Z. Han, Q. Yi, and J. Zhao, "Distributed Learning for Wireless Communications: Methods, Applications and Challenges," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 3, pp. 326–342, Apr. 2022.

[12] P. Richtarik, I. Sokolov, E. Gasanov, I. Fatkhullin, Z. Li, and E. Gorbunov, "3PC: Three Point Compressors for Communication-Efficient Distributed Training and a Better Theory for Lazy Aggregation," in *Proceedings of the 39th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., vol. 162. PMLR, Jul. 2022, pp. 18 596–18 648.

[13] S. Pu, "A Robust Gradient Tracking Method for Distributed Optimization over Directed Networks," in *2020 59th IEEE Conference on*

[20] D. Deplano, M. Franceschelli, and A. Giua, "Distributed tracking of graph parameters in anonymous networks with time-varying topology," in *2021 60th IEEE Conference on Decision and Control (CDC)*. Austin, TX, USA: IEEE, Dec. 2021, pp. 6258–6263.

Decision and Control (CDC). Jeju Island, Korea (South): IEEE, Dec. 2020, pp. 2335–2341.

[14] C. Xi, R. Xin, and U. A. Khan, "ADD-OPT: Accelerated Distributed Directed Optimization," *IEEE Transactions on Automatic Control*, vol. 63, no. 5, pp. 1329–1339, May 2018.

[15] R. Xin, S. Kar, and U. A. Khan, "Decentralized Stochastic Optimization and Machine Learning: A Unified Variance-Reduction Framework for Robust Performance and Fast Convergence," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 102–113, May 2020.

[16] E. Dall'Anese, A. Simonetto, S. Becker, and L. Madden, "Optimization and Learning With Information Streams: Time-varying algorithms and applications," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 71–83, May 2020.

[17] A. Simonetto, E. Dall'Anese, S. Paternain, G. Leus, and G. B. Giannakis, "Time-Varying Convex Optimization: Time-Structured Algorithms and Applications," *Proceedings of the IEEE*, vol. 108, no. 11, pp. 2032–2048, Nov. 2020.

[18] K. Yuan, W. Xu, and Q. Ling, "Can Primal Methods Outperform Primal-Dual Methods in Decentralized Dynamic Optimization?" *IEEE Transactions on Signal Processing*, vol. 68, pp. 4466–4480, 2020.

[19] G. Oliva, R. Setola, and C. N. Hadjicostis, "Distributed finite-time calculation of node eccentricities, graph radius and graph diameter," *Systems & Control Letters*, vol. 92, pp. 20–27, Jun. 2016.

[21] R. Dixit, A. S. Bedi, R. Tripathi, and K. Rajawat, "Online Learning with Inexact Proximal Online Gradient Descent Algorithms," *IEEE Transactions on Signal Processing*, vol. 67, no. 5, pp. 1338 – 1352, 2019.

[22] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the Convergence of FedAvg on Non-IID Data," in *International Conference on Learning Representations*, 2020.

[23] S. Shalev-Shwartz, "Online Learning and Online Convex Optimization," *Foundations and Trends® in Machine Learning*, vol. 4, no. 2, pp. 107–194, 2011.

[24] A. I. Rikos, C. N. Hadjicostis, and K. H. Johansson, "Non-oscillating quantized average consensus over dynamic directed topologies," *Automatica*, vol. 146, 2022.

[25] K. I. Tsianos, S. Lawlor, and M. G. Rabbat, "Push-Sum Distributed Dual Averaging for convex optimization," in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*. Maui, HI: IEEE, Dec. 2012, pp. 5453–5458.

[26] N. Bastianello and E. Dall'Anese, "Distributed and Inexact Proximal Gradient Method for Online Convex Optimization," in *2021 European Control Conference (ECC)*, Delft, Netherlands, 2021, pp. 2432–2437.

[27] A. B. Taylor, J. M. Hendrickx, and F. Glineur, "Exact Worst-Case Convergence Rates of the Proximal Gradient Method for Composite Convex Minimization," *Journal of Optimization Theory and Applications*, vol. 178, no. 2, pp. 455–476, Aug. 2018.

[28] H. H. Bauschke and P. L. Combettes, *Convex analysis and monotone operator theory in Hilbert spaces*, 2nd ed., ser. CMS books in mathematics. Cham: Springer, 2017.