2024 IEEE 63rd Conference on Decision and Control (CDC)
December 16-19, 2024. MiCo, Milan, Italy

Asynchronous Distributed Learning with Quantized Finite-Time Coordination

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

2024 IEEE 63rd Conference on Decision and Control (CDC) | 979-8-3503-1633-9/24/$31.00 ©2024 IEEE | DOI: 10.1109/CDC56724.2024.10885917

***Abstract*— In this paper we address distributed learning problems over peer-to-peer networks. In particular, we focus on the challenges of quantized communications, asynchrony, and stochastic gradients that arise in this set-up. We first discuss how to turn the presence of quantized communications into an advantage, by resorting to a *finite-time, quantized coordination* scheme. This scheme is combined with a distributed gradient descent method to derive the proposed algorithm. Secondly, we show how this algorithm can be adapted to allow asynchronous operations of the agents, as well as the use of stochastic gradients. Finally, we propose a variant of the algorithm which employs zooming-in quantization. We analyze the convergence of the proposed methods and compare them to state-of-the-art alternatives.**

I. Introduction

In recent years, multi-agent systems have become ubiquitous in a broad range of applications, *e.g.* robotics, power grids, traffic networks [1], [2]. A multi-agent system consists of autonomous agents with communication and computation capabilities, cooperating to accomplish a specific goal, *e.g.* learning, decision-making, navigation. In this paper, we will focus on developing algorithms to enable decentralized learning. In decentralized learning, the agents in the system collect and locally store data, with the goal being to collectively train a model without sharing these raw data [3]. To enable this objective, the design of distributed learning (or optimization) algorithms has been extensively studied in the past decades [2], [4]. In particular, different classes of algorithms have been proposed, with the main ones being gradient methods (*e.g.* DGD), gradient tracking, and dual methods (*e.g.* ADMM) [5]. In this paper, we will focus on gradient-based methods.

Learning over a multi-agent system, however, presents a number of practical challenges, with communication constraints being a central one. These constraints may arise due to the reliance on communication channels with limited bandwidth (*e.g.* wireless) [6], or the necessity to share high dimensional models (*e.g.* neural networks) [7]. A common solution to reduce the communication burden is the use of *quantization*, which however may result in lower accuracy of the trained model. In this paper we address distributed learning with quantized communication, and aim at showing how to turn quantization from a design constraint into an opportunity. The central idea is that agents employing quantized communications can *reach an inexact consensus in finite time*. Thus, in this paper we combine a Finite-Time, Quantized Coordination (FTQC) scheme with gradient descent, to design efficient learning algorithms that only require quantized communications. In particular, the FTQC scheme we analyze is based on the consensus ADMM [8], differently from previous alternatives [9], [10].

Besides limited communications, in this paper we address two additional challenges that arise in distributed learning: *asynchrony* [11] and *stochastic gradients* [12]. Indeed, the cooperating agents may have *access to different, and limited, hardware resources*. On the one hand, different resources result in the agents having different computation speeds, which make asynchronous completion of local training steps inevitable. In this paper, therefore, we follow the literature [13], [14], [15] in designing a gradient-based learning algorithm that enables asynchronous local training. On the other hand, limited hardware resources have the consequence that agents may find the computation of local gradients prohibitive (e.g., due to potentially lengthy computation times or memory constraints). For this reason, the agents may resort to computing inexact stochastic gradients [12], [16], by only using a subset of the available data. In the following we design an algorithm that relies on stochastic gradients.

The main contributions of the paper are as follows:

1) We propose an asynchronous distributed learning algorithm which relies on finite-time, quantized coordination. The novel FTQC scheme we propose is based on consensus ADMM, and the algorithm allows for the use of stochastic gradients.
2) We further propose an alternative version of the algorithm which employs *zooming-in quantization*, which progressively reduces the loss of accuracy due to quantized communications.
3) We analyze the convergence of the proposed FTQC scheme, and of the complete algorithm, highlighting the effect of (i) quantization, (ii) stochastic gradients, (iii) asynchrony. We further analyze the effect of zooming-in quantization on the convergence.
4) We conclude with numerical results comparing the proposed FTQC scheme and algorithms against state-

This work was partially supported by the European Union's Horizon Research and Innovation Actions 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 Artificial Intelligence Thrust of the Information Hub, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China. He is also affiliated with the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China. E-mail: apostolosr@hkust-gz.edu.cn.

of-the-art alternatives.

II. Problem Formulation

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

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

where $\boldsymbol{x} = [x_1^\top, \dots, x_N^\top]^\top$, $f_i : \mathbb{R}^n \to \mathbb{R}$ are local costs, each privately held by one of the agents, and we define the *consensus set* $\mathcal{C} = \{\boldsymbol{x} \in \mathbb{R}^{nN} \mid x_i = x_j \ \forall i, j \in \mathcal{V}\}$. In the following we are interested in the *finite-sum local costs* that arise in learning applications, hence we assume that

$$f_i(x) = \frac{1}{m_i} \sum_{h=1}^{m_i} \ell(x; d_i^h) \tag{2}$$

where $\ell : \mathbb{R}^n \to \mathbb{R}$ is a loss function (*e.g.* logistic) and $\{d_i^h\}_{h=1}^{m_i}$ are the data points stored by agent i (*e.g.* pairs of label and feature vector).

The following assumptions will hold throughout the paper.

Assumption 1 (Network): The graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is undirected and connected.

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

By Assumption 1, the graph is connected, which ensures that problem (1) can be solved in a distributed fashion. Moreover, Assumption 2 implies that there is a unique solution to the problem, which we can write as

$$\boldsymbol{x}^* = \mathbf{1}_N \otimes x^*, \quad x^* = \arg\min_{x\in\mathbb{R}^n} \sum_{i=1}^{N} f_i(x).$$

We are now ready to discuss the objectives that will guide the algorithm design in Section III:

1) *Quantized communications*: learning problems are *high-dimensional*, as the model being trained may have a large number of parameters $n \gg 1$ [17], [18]. However, distributed learning requires the agents to share their local models, which may cause a large communication overhead. The idea is to design an algorithm that uses quantized/compressed communications [19].
2) *Stochastic gradients*: in order to train an accurate model, the local costs (2) of the learning problem are often defined over a large data-set, with $m_i \gg 1$ [17], [18]. However, computing the gradients of such costs may be excessively time consuming. Hence, we are interested in designing an algorithm that uses less computationally expensive gradients, called stochastic gradients.
3) *Asynchrony*: synchronizing all agents in the network $\mathcal{G}$ may not be feasible, especially when $N \gg 1$ [11]. The goal is to design an algorithm that allows the agents to perform computations asynchronously.

III. Algorithm Design

In this section we design the proposed distributed learning algorithm tailored to the objectives detailed in section II. Conceptually, one could think of solving problem (1) by applying the *projected gradient* method [20] characterized by

$$\boldsymbol{x}_{k+1} = \mathrm{proj}_{\mathcal{C}} \left(\boldsymbol{x}_k - \alpha \nabla f(\boldsymbol{x}_k)\right), \quad k \in \mathbb{N}, \tag{3}$$

where $\nabla f(\boldsymbol{x}) = [\nabla f_1(x_1)^\top, \dots, \nabla f_N(x_N)^\top]^\top$ collects the local gradients, and the projection onto the consensus space is $\mathrm{proj}_{\mathcal{C}}(\boldsymbol{x}) = \frac{1}{N}\sum_{i=1}^{N} x_i$.

Clearly, the computation of $\mathrm{proj}_{\mathcal{C}}(\boldsymbol{x})$ cannot be performed in a distributed fashion, except with specific architectures such as federated learning [18]. The objective therefore is to propose a distributed (and approximate) implementation of the consensus projection. Different techniques have been explored to this end, foremost of which is averaged consensus. In particular, we can replace $\mathrm{proj}_{\mathcal{C}}(\boldsymbol{x})$ with one or more consensus steps, giving rise to Near-DGD [21]

$$\boldsymbol{x}_{k+1} = \boldsymbol{W}^t \left(\boldsymbol{x}_k - \alpha \nabla f(\boldsymbol{x}_k)\right), \quad k \in \mathbb{N},\ t \geq 1, \tag{4}$$

where $\boldsymbol{W}$ is a symmetric, doubly stochastic matrix. Alternatively, the average consensus can be replaced with dynamic average consensus, which gives rise to gradient tracking algorithms [12].

In this paper we take a different approach by using, similarly to [9], [10], a *finite-time, quantized coordination* (FTQC) scheme to approximate $\mathrm{proj}_{\mathcal{C}}(\boldsymbol{x})$. Indeed, as discussed in section II, in learning applications we may need to use quantized communications, and the idea is to use this fact to our advantage.

A. Finite-time, quantized coordination

The main insight guiding our design is that *specific consensus schemes achieve convergence in finite-time when the communications are quantized*. Employing such a scheme therefore allows the agents to approximate $\mathrm{proj}_{\mathcal{C}}(\boldsymbol{x})$ in a finite number of iterations. The algorithm proposed in [22], for example, is specifically tailored to achieve this goal. However, we explore a different strategy by showing how the consensus ADMM [8] satisfies the requirements of a FTQC scheme.

Let $\{y_i\}_{i\in\mathcal{V}}$ be local states that need to be averaged. We can formulate this as the distributed optimization problem

$$\min_{\boldsymbol{x}\in\mathbb{R}^{nN}} \frac{1}{2} \sum_{i=1}^{N} \|x_i - y_i\|^2 \quad \text{s.t. } \boldsymbol{x} \in \mathcal{C}, \tag{5}$$

to which we apply the distributed ADMM [8], yielding Algorithm 1 [1].

The following lemma shows how Algorithm 1 can indeed serve as a FTQC scheme. The proof is reported in Appendix I.

Lemma 1 (Consensus ADMM as FTQC scheme): Let $\{w_i^\ell\}_{\ell\in\mathbb{N}}$ be the trajectory generated by Algorithm 1

[1] To be precise, Algorithm 1 is derived from [8] by setting $\alpha = 0.5$, and excluding the termination step, discussed in the following.

Algorithm 1 Finite-time quantized coordination (FTQC)

Input: The states to be averaged $\{y_i\}_{i\in\mathcal{V}}$, $z_{ij}^0 = 0$ for all $i \in \mathcal{V}$, $j \in \mathcal{N}_i$, penalty $\rho > 0$, quantizer $q(\cdot)$, termination threshold $\theta > 0$.

// initialization

each agent $i \in \mathcal{V}$ picks z_{ij}^0 for all $j \in \mathcal{N}_i$

for $\ell = 1, 2, \dots$ **do**

// local update and transmission

if agent i is active

computes $w_i^\ell = \frac{1}{1+\rho|\mathcal{N}_i|}\left(y_i + \sum_{j\in\mathcal{N}_i} z_{ij}^\ell\right)$

and transmits $t_{i\to j} = q\left(-z_{ij}^\ell + 2\rho w_i^\ell\right)$ to each neighbor $j \in \mathcal{N}_i$

end if

// auxiliary update

if agent i is active and receives $t_{j\to i}$

computes $z_{ij}^{\ell+1} = \frac{1}{2}\left(z_{ij}^\ell + t_{j\to i}\right)$

end if

// termination

if $\left\|z_{ij}^{\ell+1} - z_{ij}^\ell\right\| \le \theta$ for all $j \in \mathcal{N}_i$

agent i terminates

end if

end for

applied to average $\{y_i\}_{i\in\mathcal{V}}$, with a given (auxiliary) initial condition $\{z_{ij}^0\}_{i\in\mathcal{V},\ j\in\mathcal{N}_i}$, and penalty $\rho > 0$. Assume that communications are quantized according to

$$q(x) = \Delta\left\lfloor \frac{x}{\Delta}\right\rceil, \ \Delta > 0 \tag{6}$$

where $\lfloor\cdot\rceil$ rounds to the nearest integer. Then there exist $\mu \in (0,1)$ and $C > 0$ such that for each $i \in \mathcal{V}$

$$\left\|w_i^\ell - \frac{1}{N}\sum_{i\in\mathcal{V}} y_i\right\| \le C\left(\mu^\ell d(z_0) + \frac{\Delta}{2}\sqrt{n\sum_{i\in\mathcal{V}}|\mathcal{N}_i|}\frac{1-\mu^\ell}{1-\mu}\right)$$

with $d(z_0)$ being a function of the initial conditions. Additionally, convergence is achieved after a finite number of iterations; that is, for $\ell \ge \bar{\ell}$ we have $w_i^\ell = w_i^{\ell+1}$, with

$$\bar{\ell} \ge \left|\frac{1}{\log(\mu)}\log\left(\frac{\frac{\Delta}{2}\sqrt{n\sum_{i\in\mathcal{V}}|\mathcal{N}_i|}}{(1-\mu)d(z_0)}\right)\right|.$$

We can now use the finite-time convergence result of Lemma 1 to design the termination technique in Algorithm 1. The idea is for each agent $i \in \mathcal{V}$ to detect when the difference $z_{ij}^{\ell+1} - z_{ij}^\ell$ is below a threshold θ, identifying that their values have stopped changing significantly. In practice, we can choose $\theta = c\Delta$ for some $c \ge 1$. Notice that the agents do not need to know $\bar{\ell}$ to apply the termination.

Remark 1 (Speed and accuracy trade-off): Lemma 1 shows how the smaller the quantization level Δ is, the smaller the consensus error. On the other hand, smaller values of Δ imply that a larger number of iterations is required to reach convergence, thus presenting a trade-off between speed and accuracy.

Remark 2 (Why choose ADMM?): Why choose consensus ADMM as an FTQC scheme, as opposed to the average consensus of Near-DGD, or the FTQC [22]? The answer is that ADMM has been proved to be robust to many different challenges, ranging from asynchrony and packet losses [8], to quantization and other additive errors [23]. Alternative schemes instead lack such theoretical robustness guarantees.

Remark 3 (Extensions of Algorithm 1): Besides allowing for asynchronous activations and packet losses (cf. Remark 2), we can further modify Algorithm 1 to allow the agents to use different quantizers. Indeed, Lemma 1 would apply equally, but replacing Δ with the maximum of the local quantization level Δ_i.

B. Algorithm

The proposed Algorithm 2 is based on the projected gradient descent (3), where the projection is approximated with the finite-time, quantized coordination scheme discussed in section III-A above.

In particular, the agents apply a local gradient step in steps 2-3. They then apply Algorithm 1 on the result of steps 2-3 ($\boldsymbol{y}_k$), and update their local states $\boldsymbol{x}_k$ with the result. Notice that the algorithm allows for asynchronous operations: steps 2-3 are performed only by active agents, while inactive ones do not update their $y_{i,k}$'s. Additionally, the agents may use stochastic gradients $\hat{\nabla} f_i$ instead of the full gradients.

Algorithm 2 Proposed algorithm

Input: For each agent $i \in \mathcal{V}$ initialize $x_{i,0}$; choose the step-size $\alpha < 2/\bar{\lambda}$.

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

// local update

if agent i active

apply the local (possibly inexact) gradient step

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

else $y_{i,k} = y_{i,k-1}$

end if

// coordination

apply finite-time, quantized coordination

$$\boldsymbol{x}_{k+1} = \text{Algorithm 1}(\boldsymbol{y}_k)$$

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

end for

C. Zooming-in quantization

By the discussion in Remark 1, the quantization level Δ mediates the trade-off between the speed of convergence of Algorithm 1 and its consensus error. The idea then is to exploit this trade-off to improve the performance of Algorithm 2 by changing Δ over time.

By Remark 3 we know that the agents can have *uncoordinated* quantizers, *i.e.* $q_i(x) = \Delta_i\lfloor x/\Delta_i\rceil$. Each agent then is allowed to modify its quantizer whenever necessary. Algorithm 3 reports a prototype of how this can be implemented. Specifically, each agent checks periodically if its

local solution $x_{i,k}$ has stopped improving, and selects $r\Delta_i$, $r \in (0,1)$, if this is the case.

An alternative algorithm with zooming-in quantization was proposed in [24]. However, in [24] the agents reduce their quantization in a synchronized fashion via voting, while in Algorithm 3 the agents can set their quantization levels independently.

Algorithm 3 Proposed algorithm (zooming-in quantization)

Input: For each agent $i \in \mathcal{V}$ initialize $x_{i,0}$; choose the step-size α. Choose the local quantization level Δ_i, and let $r \in (0,1)$ and $T \geq 1$.

1: **for** $k = 0, 1, \ldots$ each agent i **do**
// local update
2: **if** agent i active
3: apply the local (possibly inexact) gradient step

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

4: **else** $y_{i,k} = y_{i,k-1}$
5: **end if**
// coordination
6: $\boldsymbol{x}_{k+1} =$ Algorithm 1$(\boldsymbol{y}_k)$, with local quantization levels Δ_i
// zooming-in quantization
7: **if** agent i activated T times & $\|x_{i,k+1} - x_{i,k}\| \leq \Delta_i$
8: $\Delta_i \leftarrow r\Delta_i$
9: **end if**
10: **end for**

IV. Convergence Analysis

In this section we analyze the convergence of Algorithms 2 and 3 when the agents operate asynchronously and apply stochastic gradients. Before presenting our analysis we make the following assumption.

Assumption 3 (Set-up): Each agent $i \in \mathcal{V}$ activates at iteration k to perform a local gradient step with probability $p_i \in (0,1]$. In particular, active agents use a (possibly inexact) gradient $\hat{\nabla} f_i$, for which there exists $\tau \geq 0$ such that

$$\mathbb{E}\left[\left\|\hat{\nabla} f_i(x) - \nabla f_i(x)\right\|\right] \leq \tau.$$

Under this assumption, we derive the following convergence result, proved in Appendix II.

Proposition 1 (Convergence of Algorithm 2): Let $\{\boldsymbol{x}_k\}_{k\in\mathbb{N}}$ be the trajectory generated by Algorithm 2. Let Assumptions 1, 2, and 3 hold. Then for all $k > 0$ it holds that

$$\mathbb{E}\left[\|\boldsymbol{x}_k - \boldsymbol{x}^*\|\right] \leq \sqrt{\frac{\max_i p_i}{\min_i p_i}}\left(\chi^k \|\boldsymbol{x}_0 - \boldsymbol{x}^*\| + \left(\gamma + \alpha\tau\sqrt{N}\right)\frac{1-\chi^k}{1-\chi}\right)$$

where $\chi = \sqrt{1-(1-\zeta^2)\min_i p_i} \in (0,1)$ with $\zeta = \max\{|1-\alpha\underline{\lambda}|, |1-\alpha\bar{\lambda}|\}$, and $\gamma = O(\Delta)$ (as characterized in Lemma 1).

As a consequence of Proposition 1, we see that

$$\lim_{k\to\infty} \mathbb{E}\left[\|\boldsymbol{x}_k - \boldsymbol{x}^*\|\right] \leq \sqrt{\frac{\max_i p_i}{\min_i p_i}}\,\frac{\gamma + \alpha\tau\sqrt{N}}{1-\chi}$$

which highlights how the different challenges of quantization, stochastic gradients, and asynchrony impact the asymptotic error.

We can similarly characterize the asymptotic error when we employ the zooming-in quantization of Algorithm 3. The proof is reported in Appendix III.

Corollary 1 (Convergence of Algorithm 3): Let $\{\boldsymbol{x}_k\}_{k\in\mathbb{N}}$ be the trajectory generated by Algorithm 3. Let Assumptions 1, 2, and 3 hold. Then it holds that

$$\lim_{k\to\infty} \mathbb{E}\left[\|\boldsymbol{x}_k - \boldsymbol{x}^*\|\right] \leq \sqrt{\frac{\max_i p_i}{\min_i p_i}}\,\frac{\alpha\tau\sqrt{N}}{1-\chi}.$$

Clearly, using zooming-in quantization implies that quantization will not impact the asymptotic error, and only the effects of asynchrony and stochastic gradients are present.

V. Numerical Results

In this section we evaluate the performance of the proposed algorithms on a classification task, and compare it with algorithms from the literature. We consider problem (1) with local costs

$$f_i(x) = \sum_{h=1}^{m_i} \log\left(1 + \exp(-b_i^h a_i^h x)\right) + \frac{\epsilon}{2}\|x\|^2 \qquad (7)$$

defined by the local dataset $\{d_i^h = (a_i^h, b_i^h) \in \mathbb{R}^{1\times n} \times \{-1,1\}\}_{h=1}^{m_i}$. In our experiments we have $N = 10$ agents with $m_i = 150$ data-points each, and the problem size is $n = 10$. The regularization weight is set to $\epsilon = 0.075$. Moreover, unless otherwise stated we use the symmetric quantizer (6). Finally, the data for the problem are randomly generated using the `make_classification` utility of `sklearn` [25], and all algorithms are implemented in `tvopt` [26].

A. Performance of Finite-Time, Quantized Coordination schemes

We start by comparing the performance of the proposed Finite-Time, Quantized Coordination scheme Algorithm 1 with the scheme proposed in [22] and employed in [9], [10]. In Table I we compare the two FTQC schemes in terms of consensus error and number of iterations, for different quantization levels. The algorithms are applied to average randomly generated vectors in $\mathbb{R}^{10}$, the size of x in (7). We can see that Algorithm 1 consistently outperforms [22] in terms of consensus error, since it reaches a smaller neighborhood of the consensus, and in terms of the number of iterations it requires.

Turning exclusively to Algorithm 1, we know that it is characterized by two parameters, the penalty ρ and the quantizer $q(\cdot)$. In the following we provide results to guide the tuning of these parameters. First of all, Figure 1 reports the consensus error and number of iterations for different values of the penalty and quantization levels. Interestingly, both metrics are minimized for a value of $\rho \approx 0.3$. Moreover,

TABLE I

CONSENSUS ERROR AND ITERATION NUMBER FOR DIFFERENT QUANTIZATION LEVELS.

Δ	[22]		Algorithm 1	
	Cons. err.	Num. iter.	Cons. err.	Num. iter.
10^{-8}	5.31×10^{-8}	115	2.85×10^{-8}	105
10^{-7}	5.13×10^{-7}	106	2.88×10^{-7}	95
10^{-6}	5.32×10^{-6}	96	2.89×10^{-6}	86
10^{-5}	5.30×10^{-5}	86	2.86×10^{-5}	80
10^{-4}	5.31×10^{-4}	75	2.89×10^{-4}	66
10^{-3}	5.27×10^{-3}	67	2.87×10^{-3}	57
10^{-2}	5.36×10^{-2}	55	2.91×10^{-2}	48
10^{-1}	5.18×10^{-1}	49	2.87×10^{-1}	43
1	5.17	38	2.89	29

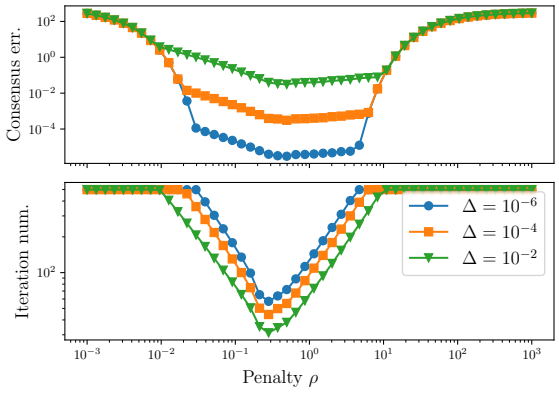

Fig. 1. Consensus error and iteration number for Algorithm 1 with different penalties and quantization levels.

as predicted by Remark 1, the smaller the quantization level is, the smaller the consensus error is, to the detriment of the number of iterations needed to converge.

Finally, Table II reports the performance of the consensus scheme with different quantizers besides the symmetric (6), namely: floor $q(x) = \Delta\lfloor x/\Delta \rfloor$, ceil $q(x) = \Delta\lceil x/\Delta \rceil$, sparsifier, which sets to zero all components of x with absolute value below $\theta = 0.1$. We notice that the floor

TABLE II

PERFORMANCE OF ALGORITHM 1 WITH DIFFERENT QUANTIZERS.

Quantizer	Cons. err.	Num. iter.
Symmetric	3.37×10^{-3}	52
Floor	8.70×10^{-3}	52
Ceil	8.69×10^{-3}	52
Sparsifier	3.79×10^{-3}	50

(employed in [22], [9], [10]) and ceiling quantizers attain a more than double the consensus error of the symmetric one. The sparsifier instead achieves similar performance. Future work will explore the use of the sparsifier from a theoretical perspective.

B. Comparison of gradient descent schemes

The previous section evaluated the performance of the Finite-Time, Quantized Coordination scheme Algorithm 1 which is used as a building block of Algorithm 2. In this section we discuss the performance of Algorithm 2 itself, and compare it with alternative methods.

We start by comparing Algorithm 2 to FTQC-DGD [10], Near-DGD [21], and the distributed gradient tracking (DGT) method [12]. The latter two do not employ a finite-time coordination scheme, but they are modified to use multiple rounds of communications to match the budget of FTQC-DGD and Algorithm 2. Figure 2 reports the error trajectories of all methods. We can see that Algorithm 2 achieves a smaller

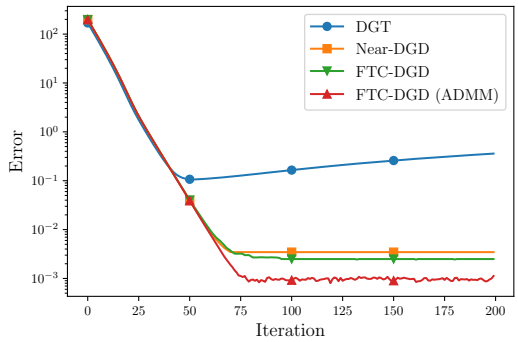

Fig. 2. Comparison of different distributed optimization methods with quantized communications.

asymptotic error than Near-DGD and FTQC-DGD, owing to the improved coordination performance of Algorithm 1 (cf. section V-A). Moreover, DGT appears to diverge, which is known to happen with some gradient tracking schemes perturbed by (quantization) noise [27].

Table III further compares Near-DGD, FTQC-DGD and Algorithm 2 for different quantization levels. The proposed

TABLE III

COMPARISON OF NEAR-DGD [21], FTQC-DGD [10], AND ALGORITHM 2 FOR DIFFERENT QUANTIZATION LEVELS.

Δ	Near-DGD [21]	FTQC-DGD [10]	Algorithm 2
10^{-10}	3.75×10^{-8}	4.76×10^{-8}	2.13×10^{-8}
10^{-8}	3.00×10^{-7}	7.35×10^{-7}	1.04×10^{-7}
10^{-6}	3.29×10^{-5}	2.84×10^{-5}	7.79×10^{-6}
10^{-4}	3.46×10^{-3}	2.49×10^{-2}	1.12×10^{-3}
10^{-2}	1.88×10^{-1}	2.56×10^{-1}	7.86×10^{-2}
1	24.02	20.15	3.46

Algorithm 2 outperforms both alternatives, again owing to the improved coordination precision.

C. Variations of Algorithm 2

In this section we discuss the performance of Algorithm 2 in challenging scenarios, and compare it to that of Algorithm 3.

As discussed in section II, in learning problems the use of full gradients may be prohibitive, and the agents need to resort to stochastic gradients. In Figure 3 we report the asymptotic error achieved by Algorithm 2 when stochastic gradients computed on different batch sizes B are used. Clearly, the larger the batch size, the better the performance. However, due to the use of quantization, even with $B = m_i$

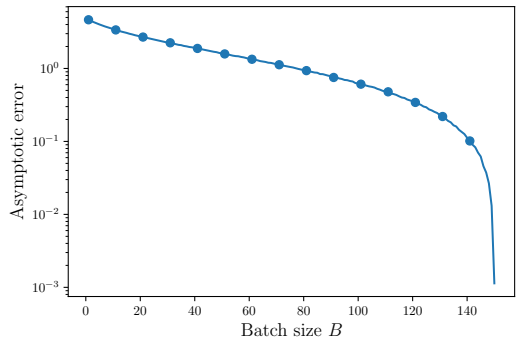

Fig. 3. Asymptotic error of Algorithm 2 when the agents employ stochastic gradients of different batch sizes.

the algorithm can only reach a neighborhood of the optimal solution.

Another one of the challenges discussed in section II is the asynchronous operation of the agents. In Figure 4 we report the performance of Algorithm 2 in this scenario, when agents activate to perform a gradient descent step with probability $p \in (0, 1]$. As predicted by the theory [28], the smaller p is

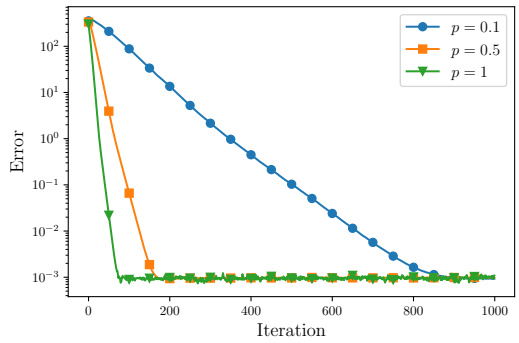

Fig. 4. Error trajectory of Algorithm 2 with different agent activation probabilities.

the fewer updates are performed, and hence the slower the convergence is.

We conclude this section by comparing the performance of Algorithm 2 with the variation Algorithm 3 that employs zooming-in quantization ($T = 25$, $r = 0.1$). In particular, Figure 5 depicts the error trajectory of the latter against the error trajectory of the former with different quantization levels. The x-axis marks the cumulative number of communication rounds. We can thus deduce that using zooming-in quantization can achieve very good performance (in terms of asymptotic error) with a smaller number of communication rounds.

VI. Conclusions

In this paper we addressed distributed learning problems over peer-to-peer networks, with a particular focus on the challenges of quantized communications, asynchrony, and stochastic gradients that arise in this set-up. We first discussed how to turn the presence of quantized communications into an advantage, by resorting to a *finite-time, quantized coordination* scheme. This scheme is combined with a distributed gradient descent method to derive the proposed algorithm. Secondly, we showed how this algorithm can be adapted to allow asynchronous operations of the agents, as well as the use of stochastic gradients. Finally, we proposed a variant of the algorithm which employs zooming-in quantization. We analyzed the convergence of the proposed methods and compared them to state-of-the-art alternatives. The performance of the proposed methods compares very favorably with the alternatives from the literature.

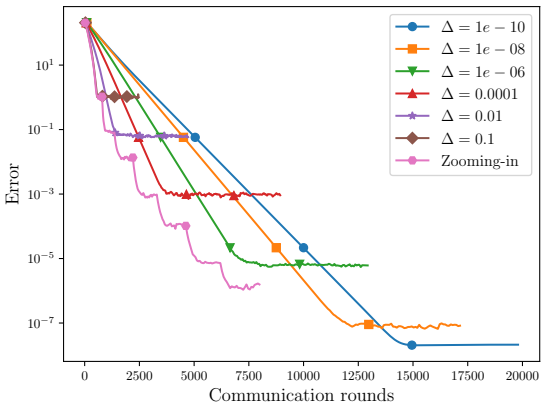

Fig. 5. Comparison of Algorithm 2 (fixed quantization level) with Algorithm 3 (zooming-in quantization).

Appendix I
Proof of Lemma 1

We start by observing that Algorithm 1 consists of an affine update in $\boldsymbol{z} = [z_{ij}]_{i\in\mathcal{V},\ j\in\mathcal{N}_i}$; in particular, for appropriate matrices and vectors we can write $\boldsymbol{z}^{\ell+1} = \boldsymbol{T}\boldsymbol{z}^{\ell} + \boldsymbol{u} + \boldsymbol{e}^{\ell}$, $\boldsymbol{w}^{\ell} = \boldsymbol{H}\boldsymbol{z}^{\ell}$. The vector $\boldsymbol{e}^{\ell}$ represents the noise caused by quantization, that is $e_{ij}^{\ell} = q(-z_{ji}^{\ell} + 2\rho w_j^{\ell}) - (-z_{ji}^{\ell} + 2\rho w_j^{\ell})$. Since Algorithm 1 is an affine operator (plus additive noise) then it is μ-metric subregular for a given $\mu \in (0, 1)$ [29]. Therefore the assumptions of [23, Theorem 3] are verified, and we have

$$d(\boldsymbol{z}^{\ell}) \leq \mu^{\ell} d(\boldsymbol{z}^{0}) + \sum_{h=0}^{\ell-1} \mu^{\ell-h-1} \left\|\boldsymbol{e}^{h}\right\| \tag{8}$$

$$\left\|\boldsymbol{w}^{\ell} - \frac{1}{N}\sum_{i\in\mathcal{V}} y_i \otimes \mathbf{1}_N\right\| \leq C d(\boldsymbol{z}^{\ell}), \tag{9}$$

where $d(\boldsymbol{z})$ measures the distance of $\boldsymbol{z}$ from the set of fixed points $\{\bar{\boldsymbol{z}} \mid \bar{\boldsymbol{z}} = \boldsymbol{T}\bar{\boldsymbol{z}} + \boldsymbol{u}\}$.

Now, since $\boldsymbol{e}^{\ell}$ represents the quantization noise, we can upper bound its norm as follows:

$$\begin{aligned}\left\|\boldsymbol{e}^{\ell}\right\|^2 &= \sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{N}_i} \left\|q(-z_{ji}^{\ell} + 2\rho w_j^{\ell}) - (-z_{ji}^{\ell} + 2\rho w_j^{\ell})\right\|^2 \\ &\leq \sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{N}_i} n(\Delta/2)^2 = n(\Delta/2)^2 \sum_{i\in\mathcal{V}} |\mathcal{N}_i|\end{aligned}$$

where the inequality holds because the quantization commits an error of at most $\Delta/2$. Using this bound and combining (8) with (9) then yields the first thesis.

The goal now is to show that Algorithm 1 achieves finite-time convergence. By (8), we know that $\lim_{\ell\to\infty} d(\boldsymbol{z}^\ell) = \frac{1}{1-\mu}\frac{\Delta}{2}\sqrt{n\sum_{i\in\mathcal{V}}|\mathcal{N}_i|}$. Then to bound the time of convergence we impose that the first term on the right-hand side of (8) be smaller than $\lim_{\ell\to\infty} d(\boldsymbol{z}^\ell)$. By rearranging, taking the logarithm and the absolute value, the thesis follows. □

Appendix II
Proof of Proposition 1

Algorithm 2 was derived in section III as an inexact version of the projected gradient method, where Algorithm 1 replaces the projection onto the consensus set. Additionally, by Assumption 3, the agents apply inexact gradients during local computations. Accounting for both these sources of errors, we can characterize Algorithm 2 as

$$\boldsymbol{x}_{k+1} = \mathrm{proj}_{\mathcal{C}}\left(\boldsymbol{x}_k - \alpha\nabla f_k(\boldsymbol{x}_k)\right) + \boldsymbol{e}_k^q + \boldsymbol{e}_k^g, \tag{10}$$

where $\boldsymbol{e}_k^q$ is the error due to Algorithm 1 (cf. Lemma 1), and $\boldsymbol{e}_k^g$ is the error due to inexact gradients:

$$\begin{aligned}\boldsymbol{e}_k^p &= \text{Algorithm 1}(\boldsymbol{y}_k) - \mathrm{proj}_{\mathcal{C}}(\boldsymbol{y}_k)\\ \boldsymbol{e}_k^g &= \mathrm{proj}_{\mathcal{C}}(\boldsymbol{x}_k - \alpha\hat{\nabla} f_k(\boldsymbol{x}_k)) - \mathrm{proj}_{\mathcal{C}}(\boldsymbol{x}_k - \alpha\nabla f_k(\boldsymbol{x}_k)).\end{aligned}$$

Moreover, Assumption 3 allows the agents to activate asynchronously, each with its probability $p_i \in (0,1]$. This means that the i-th coordinate of $\boldsymbol{x}_k$ is updated with probability p_i.

Finally, we notice that by the choice $\alpha < 2/\bar{\lambda}$, the projected gradient method (without errors and asynchrony) is $\zeta = \max\{|1-\alpha\underline{\lambda}|, |1-\alpha\bar{\lambda}|\}$-contractive [20]. This implies that Algorithm 2 can be interpreted as a projected gradient method with bounded additive noise and random coordinate updates. Thus it verifies the assumptions of [28, Proposition 1], which implies

$$\begin{aligned}\mathbb{E}\left[\|\boldsymbol{x}_k - \boldsymbol{x}^*\|\right] \le \sqrt{\frac{\max_i p_i}{\min_i p_i}}\Big(&\chi^k\|\boldsymbol{x}_0 - \boldsymbol{x}^*\|\\ &+ \sum_{h=0}^{k}\chi^{k-h}\|\boldsymbol{e}_h^p + \boldsymbol{e}_h^g\|\Big).\end{aligned}$$

Now, by Assumption 3 we know that $\mathbb{E}\left[\|\boldsymbol{e}_k^g\|\right] \le \tau\sqrt{N}$, and by Lemma 1 we can bound $\|\boldsymbol{e}_k^p\| \le \sqrt{N}C\frac{\Delta}{2}\sqrt{n\sum_{i\in\mathcal{V}}|\mathcal{N}_i|}\frac{1}{1-\mu} = O(\Delta)$, and the thesis follows. □

Appendix III
Proof of Corollary 1

Following the same derivation as Appendix II yields

$$\begin{aligned}\mathbb{E}\left[\|\boldsymbol{x}_k - \boldsymbol{x}^*\|\right] \le \sqrt{\frac{\max_i p_i}{\min_i p_i}}\Big(&\chi^k\|\boldsymbol{x}_0 - \boldsymbol{x}^*\|\\ &+ \sum_{h=0}^{k}\chi^{k-h}\|\boldsymbol{e}_h^p\| + \|\boldsymbol{e}_h^g\|\Big).\end{aligned}$$

By Assumption 3 we know that $\mathbb{E}\left[\|\boldsymbol{e}_k^g\|\right] \le \tau\sqrt{N}$. On the other hand, by the use of zooming-in quantization, and by Lemma 1, we have

$$\|\boldsymbol{e}_k^p\| \le \sqrt{N}C\frac{\Delta_k}{2}\sqrt{n\sum_{i\in\mathcal{V}}|\mathcal{N}_i|}\frac{1}{1-\mu}$$

with $\Delta_k = \max_i \Delta_{i,k}$ being the largest quantization level among all agents at time k. We know that Δ_k is monotonically non-increasing, and that in particular it decreases at finite intervals, when all agents have stopped seeing an improvement in their local solution $x_{i,k}$ (cf. lines 7-8 in Algorithm 3). Thus $\lim_{k\to\infty}\|\boldsymbol{e}_k^p\| = 0$, and by [30, Lemma 3.1(a)] $\lim_{k\to\infty}\sum_{h=0}^{k}\chi^{k-h}\|\boldsymbol{e}_h^p\| = 0$ since $\chi \in (0,1)$, and the thesis follows. □

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] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] N. Bastianello, R. Carli, L. Schenato, and M. Todescato, "Asynchronous Distributed Optimization Over Lossy Networks via Relaxed ADMM: Stability and Linear Convergence," *IEEE Transactions on Automatic Control*, vol. 66, no. 6, pp. 2620–2635, Jun. 2021.

[9] A. I. Rikos, W. Jiang, T. Charalambous, and K. H. Johansson, "Distributed optimization with gradient descent and quantized communication," *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 5900–5906, 2023.

[10] N. Bastianello, A. I. Rikos, and K. H. Johansson, "Online Distributed Learning with Quantized Finite-Time Coordination," in *2023 62nd IEEE Conference on Decision and Control (CDC)*. Singapore, Singapore: IEEE, Dec. 2023, pp. 5026–5032.

[11] B. M. Assran, A. Aytekin, H. R. Feyzmahdavian, M. Johansson, and M. G. Rabbat, "Advances in Asynchronous Parallel and Distributed Optimization," *Proceedings of the IEEE*, vol. 108, no. 11, pp. 2013–2031, Nov. 2020.

[12] 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.

[13] X. Zhao and A. H. Sayed, "Asynchronous Adaptation and Learning Over Networks—Part I: Modeling and Stability Analysis," *IEEE Transactions on Signal Processing*, vol. 63, no. 4, pp. 811–826, Feb. 2015.

[14] J. Xu, S. Zhu, Y. C. Soh, and L. Xie, "Convergence of Asynchronous Distributed Gradient Methods Over Stochastic Networks," *IEEE Transactions on Automatic Control*, vol. 63, no. 2, pp. 434–448, Feb. 2018.

[15] Y. Tian, Y. Sun, and G. Scutari, "Achieving Linear Convergence in Distributed Asynchronous Multiagent Optimization," *IEEE Transactions on Automatic Control*, vol. 65, no. 12, pp. 5264–5279, Dec. 2020.

[16] X. Yi, S. Zhang, T. Yang, T. Chai, and K. H. Johansson, "A Primal-Dual SGD Algorithm for Distributed Nonconvex Optimization," *IEEE/CAA Journal of Automatica Sinica*, vol. 9, no. 5, pp. 812–833, May 2022.

[17] 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.

[18] 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.

[19] Z. Zhao, Y. Mao, Y. Liu, L. Song, Y. Ouyang, X. Chen, and W. Ding, "Towards efficient communications in federated learning: A contemporary survey," *Journal of the Franklin Institute*, p. S0016003222009346, Jan. 2023.

[20] 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.

[21] 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.

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

[23] N. Bastianello, D. Deplano, M. Franceschelli, and K. H. Johansson, "Robust Online Learning over Networks," *IEEE Transactions on Automatic Control*, 2024.

[24] A. I. Rikos, W. Jiang, T. Charalambous, and K. H. Johansson, "Distributed Optimization via Gradient Descent with Event-Triggered Zooming Over Quantized Communication," in *2023 62nd IEEE Conference on Decision and Control (CDC)*, 2023, pp. 6321–6327.

[25] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[26] N. Bastianello, "tvopt: A Python Framework for Time-Varying Optimization," in *2021 60th IEEE Conference on Decision and Control (CDC)*, 2021, pp. 227–232.

[27] M. Bin, I. Notarnicola, and T. Parisini, "Stability, Linear Convergence, and Robustness of the Wang-Elia Algorithm for Distributed Consensus Optimization," in *2022 IEEE 61st Conference on Decision and Control (CDC)*. Cancun, Mexico: IEEE, Dec. 2022, pp. 1610–1615.

[28] N. Bastianello, L. Madden, R. Carli, and E. Dall'Anese, "A Stochastic Operator Framework for Optimization and Learning With Sub-Weibull Errors," *IEEE Transactions on Automatic Control*, 2024.

[29] A. Themelis and P. Patrinos, "SuperMann: A Superlinearly Convergent Algorithm for Finding Fixed Points of Nonexpansive Operators," *IEEE Transactions on Automatic Control*, vol. 64, no. 12, pp. 4875–4890, Dec. 2019.

[30] S. Sundhar Ram, A. Nedić, and V. V. Veeravalli, "Distributed Stochastic Subgradient Projection Algorithms for Convex Optimization," *Journal of Optimization Theory and Applications*, vol. 147, no. 3, pp. 516–545, Dec. 2010.