A Computation and Communication Efficient Method for Distributed Nonconvex Problems in the Partial Participation Setting

Anonymous Author
Anonymous Institution

Abstract

We present a new method that includes three key components of distributed optimization and federated learning: variance reduction of stochastic gradients, compressed communication, and partial participation. We prove that the new method has optimal oracle complexity and state-of-the-art communication complexity in the partial participation setting. Moreover, we observe that "1 + 1 + 1 is not 3": by mixing variance reduction of stochastic gradients with compressed communication and partial participation, we do not obtain a fully synergetic effect. We explain the nature of this phenomenon, argue that this is to be expected, and propose possible workarounds.

1 Introduction

Federated and distributed learning have become very popular in recent years (Konečný et al., 2016; McMahan et al., 2017). The current optimization tasks require much computational resources and machines. Such requirements emerge in machine learning, where massive datasets and computations are distributed between cluster nodes (Lin et al., 2017; Ramesh et al., 2021). In federated learning, nodes, represented by mobile phones, laptops, and desktops, do not send their data to a server due to privacy and their huge number (Ramaswamy et al., 2019), and the server remotely orchestrates the nodes and communicates with them to solve an optimization problem.

As in classical optimization tasks, one of the main current challenges is to find **computationally efficient** optimization algorithms. However, the nature of distributed problems induces many other (Kairouz et al., 2021), including i) **partial participation** of nodes in algorithm steps: due to stragglers (Li et al., 2020) or communication delays (Vogels et al., 2021), ii) **communication bottleneck**: even if a node participates, it can be costly to transmit information to a server or other nodes (Alistarh et al., 2017; Ramesh et al., 2021; Kairouz et al., 2021; Sapio et al., 2019; Narayanan et al., 2019). It is necessary to develop a method that considers these problems.

2 Optimization Problem

Let us consider the nonconvex distributed optimization problem

$$\min_{x \in \mathbb{R}^d} \left\{ f(x) := \frac{1}{n} \sum_{i=1}^{n} f_i(x) \right\}, \tag{1}$$

where $f_i : \mathbb{R}^d \rightarrow \mathbb{R}$ is a smooth nonconvex function for all $i \in [n] := \{1, \dots, n\}$. The full information about function f_i is stored on i^{th} node. The communication between nodes is maintained in the parameters server fashion (Kairouz et al., 2021): we have a server that receives compressed information from nodes, updates a state, and broadcasts an updated model.[1] Since we work in the nonconvex world, our goal is to find an ε-solution (ε-stationary point) of (1): a (possibly random) point $\widehat{x} \in \mathbb{R}^d$, such that $\mathrm{E}\left[\, \|\nabla f(\widehat{x})\|^2 \,\right] \le \varepsilon$.

We consider three settings:

1. **Gradient Setting.** The i^{th} node has only access to the gradient $\nabla f_i : \mathbb{R}^d \rightarrow \mathbb{R}^d$ of function f_i. Moreover, the following assumptions for the functions f_i hold.

   Assumption 1. *There exists $f^* \in \mathbb{R}$ such that $f(x) \ge f^*$ for all $x \in \mathbb{R}$.*

   Assumption 2. *The function f is L–smooth, i.e., $\|\nabla f(x) - \nabla f(y)\| \le L \|x - y\|$ for all $x, y \in \mathbb{R}^d$.*

   Assumption 3. *The functions f_i are L_i–smooth for all $i \in [n]$. Let us define $\widehat{L}^2 := \frac{1}{n}\sum_{i=1}^{n} L_i^2$.*[2]

[1] Note that this strategy can be used in peer-to-peer communication, assuming that the server is an abstraction and all its algorithmic steps are performed on each node.

[2] Note that $L \le \widehat{L}$, $\widehat{L} \le L_{\max}$, and $\widehat{L} \le L_{\sigma}$.

Preliminary work. Under review by AISTATS 2023. Do not distribute.

2. **Finite-Sum Setting.** The functions $\{f_i\}_{i=1}^n$ have the finite-sum form

$$f_i(x) = \frac{1}{n}\sum_{j=1}^{m} f_{ij}(x), \qquad \forall i \in [n], \tag{2}$$

where $f_{ij} : \mathbb{R}^d \to \mathbb{R}$ is a smooth nonconvex function for all $j \in [m]$. We assume that Assumptions 1, 2 and 3 hold and the following assumption.

Assumption 4. *The function f_{ij} is L_{ij}-smooth for all $i \in [n], j \in [m]$. Let $L_{\max} := \max_{i\in[n],j\in[m]} L_{ij}$.*

3. **Stochastic Setting.** The function f_i is an expectation of a stochastic function,

$$f_i(x) = \mathrm{E}_{\xi}\left[f_i(x;\xi)\right], \qquad \forall i \in [n], \tag{3}$$

where $f_i : \mathbb{R}^d \times \Omega_{\xi} \to \mathbb{R}$. For a fixed $x \in \mathbb{R}$, $f_i(x;\xi)$ is a random variable over some distribution $\mathcal{D}_i$, and, for a fixed $\xi \in \Omega_{\xi}$, $f_i(x;\xi)$ is a smooth nonconvex function. The i^{th} node has only access to a stochastic gradients $\nabla f_i(\cdot;\xi_{ij})$ of the function f_i through the distribution $\mathcal{D}_i$, where ξ_{ij} is a sample from $\mathcal{D}_i$. We assume that Assumptions 1, 2 and 3 hold and the following assumptions.

Assumption 5. *For all $i \in [n]$ and for all $x \in \mathbb{R}^d$, the stochastic gradient $\nabla f_i(x;\xi)$ is unbiased and has bounded variance, i.e., $\mathrm{E}_{\xi}\left[\nabla f_i(x;\xi)\right] = \nabla f_i(x)$, and $\mathrm{E}_{\xi}\left[\left\|\nabla f_i(x;\xi) - \nabla f_i(x)\right\|^2\right] \le \sigma^2$, where $\sigma^2 \ge 0$.*

Assumption 6. *For all $i \in [n]$ and for all $x, y \in \mathbb{R}$, the stochastic gradient $\nabla f_i(x;\xi)$ satisfies the mean-squared smoothness property, i.e., $\mathrm{E}_{\xi}\left[\left\|\nabla f_i(x;\xi) - \nabla f_i(y;\xi)\right\|^2\right] \le L_{\sigma}^2 \left\|x - y\right\|^2$.*

We compare algorithms using *the oracle complexity*, i.e., the number of (stochastic) gradients that each node has to calculate to get ε-solution, and *the communication complexity*, i.e., the number of bits that each node has to send to the server to get ε-solution.

2.1 Unbiased Compressors

We use the concept of unbiased compressors to alleviate the communication bottleneck. The unbiased compressors quantize and/or sparsify vectors that the nodes send to the server.

Definition 1. A stochastic mapping $\mathcal{C} : \mathbb{R}^d \to \mathbb{R}^d$ is an *unbiased compressor* if there exists $\omega \in \mathbb{R}$ such that

$$\mathrm{E}\left[\mathcal{C}(x)\right] = x, \qquad \mathrm{E}\left[\left\|\mathcal{C}(x) - x\right\|^2\right] \le \omega \left\|x\right\|^2, \tag{4}$$

for all $x \in \mathbb{R}^d$.

We denote a set of stochastic mappings that satisfy Definition 1 as $\mathbb{U}(\omega)$. In our methods, the nodes make use of unbiased compressors $\{\mathcal{C}_i\}_{i=1}^n$. The community developed a large number of unbiassed compressors, including Rand$K$ (see Definition **??**) (Beznosikov et al., 2020; Stich et al., 2018), Adaptive sparsification (Wangni et al., 2018) and Natural compression and dithering (Horváth et al., 2019a). We are aware of correlated compressors by Szlendak et al. (2021) and quantizers by Suresh et al. (2022) that help in the homogeneous regimes, but in this work, we are mainly concentrated on generic heterogeneous regimes, though, for simplicity, assume the independence of the compressors.

Assumption 7. *$\mathcal{C}_i \in \mathbb{U}(\omega)$ for all $i \in [n]$, and the compressors are independent.*

2.2 Nodes Partial Participation Assumptions

We now try to formalize the notion of partial participation. Let us assume that we have n events $\{i^{\text{th}}$ node is *participating*$\}$ with the following properties.

Assumption 8. *The partial participation of nodes has the following distribution: exists constants $p_{\text{a}} \in (0, 1]$ and $p_{\text{aa}} \in [0, 1]$, such that*

1. $$\mathbf{Prob}\left(i^{\text{th}} \text{ node is } \textit{participating}\right) = p_{\text{a}}, \quad \forall i \in [n],$$

2. $$\mathbf{Prob}\Big(i^{\text{th}} \text{ node is } \textit{participating} \text{ AND } j^{\text{th}} \text{ node is } \textit{participating}\Big) = p_{\text{aa}},$$

   for all $i \neq j \in [n]$.

3. $$p_{\text{aa}} \le p_{\text{a}}^2, \tag{5}$$

and these events from different communication rounds are independent.

We are not fighting for the full generality and believe that more complex sampling strategies can be considered in the analysis. For simplicity, we settle upon Assumption 8. Standard partial participation strategies, including s–nice sampling, where the server chooses uniformly s nodes without replacement ($p_{\text{a}} = {}^{s}/_{n}$ and $p_{\text{aa}} = {}^{s(s-1)}/_{n(n-1)}$), and independent participation, where each node independently participates with probability p_{a} (due to independence, we have $p_{\text{aa}} = p_{\text{a}}^2$), satisfy Assumption 8. In the literature, s–nice sampling is one of the most popular strategies (Zhao et al., 2021a; Richtárik et al., 2021; Reddi et al., 2020; Konečný et al., 2016).

3 Motivation and Related Work

The main goal of our paper is to develop a method for the nonconvex distributed optimization that will include three key features: variance reduction of stochastic gradients, compressed communication, and partial participation.

1. Variance reduction of stochastic gradients
It is important to consider finite-sum (2) and stochastic (3) settings because, in machine learning tasks, either the number of local functions m is huge or the functions f_i is an expectation of a stochastic function due to the batch normalization (Ioffe and Szegedy, 2015) or random augmentation (Goodfellow et al., 2016), and it is infeasible to calculate the full gradients analytically. Let us recall the results from the nondistributed optimization. In the gradient setting, the optimal oracle complexity is $\mathcal{O}\left({}^{1}/_{\varepsilon}\right)$, achieved by the vanilla gradient descent (GD) (Carmon et al., 2020; Nesterov, 2018). In the finite-sum setting and stochastic settings, the optimal oracle complexities are $\mathcal{O}\left(m + \frac{\sqrt{m}}{\varepsilon}\right)$ and $\mathcal{O}\left(\frac{\sigma^2}{\varepsilon} + \frac{\sigma}{\varepsilon^{3/2}}\right)$ (Fang et al., 2018; Li et al., 2021a; Arjevani et al., 2019), accordingly, achieved by methods from (Fang et al., 2018; Nguyen et al., 2017; Li et al., 2021a).

2. Compressed communication
In distributed optimization (Ramesh et al., 2021; Xu et al., 2021), lossy communication compression can be a powerful tool to increase the communication speed between the nodes and the server. Different types of compressors are considered in the literature, including unbiased compressors (Alistarh et al., 2017; Beznosikov et al., 2020; Szlendak et al., 2021), contractive (biased) compressors (Richtárik et al., 2021), 3PC compressors (Richtárik et al., 2022). We will focus on unbiased compressors because methods (Tyurin and Richtárik, 2022; Szlendak et al., 2021; Gorbunov et al., 2021) that employ unbiased compressors provide the current theoretical state-of-the-art (SOTA) communication complexities.

Many methods analyzed optimization methods with the unbiased compressors (Alistarh et al., 2017; Mishchenko et al., 2019; Horváth et al., 2019b; Gorbunov et al., 2021; Tyurin and Richtárik, 2022). In the gradient setting, the methods by Gorbunov et al. (2021) and Tyurin and Richtárik (2022) establish the current SOTA communication complexity, each method needs $\frac{1+\omega/\sqrt{n}}{\varepsilon}$ communication rounds to get an ε–solution. In the finite-sum and stochastic settings, the current SOTA communication complexity is attained by methods from (Tyurin and Richtárik, 2022), while maintaining the optimal oracle complexities $\mathcal{O}\left(m + \frac{\sqrt{m}}{\varepsilon\sqrt{n}}\right)$ and $\mathcal{O}\left(\frac{\sigma^2}{\varepsilon n} + \frac{\sigma}{\varepsilon^{3/2} n}\right)$ per node.

3. Partial participation
From the beginning of federated learning era, the partial participation has been considered to be the essential feature of distributed optimization methods (McMahan et al., 2017; Konečný et al., 2016; Kairouz et al., 2021). However, previously proposed methods have limitations: i) methods from (Gorbunov et al., 2021; Zhao et al., 2021b) still require synchronization of all nodes with a small probability. ii) in the stochastic settings, proposed methods with the partial participation mechanism (Tyurin and Richtárik, 2022; Zhao et al., 2021a; Karimireddy et al., 2020b; McMahan et al., 2017) provide results without variance reduction techniques from (Fang et al., 2018; Li et al., 2021a; Cutkosky and Orabona, 2019) and, therefore, get suboptimal oracle complexities. Note that the papers by Tyurin and Richtárik (2022) and Zhao et al. (2021a) provide algorithms that reduce the variance *only from compressors in the partial participation and stochastic setting*. iii) in the finite-sum setting, the work by Li et al. (2021b) focuses on the homogeneous regime only (the functions f_i are equal). iv) The paper by Karimireddy et al. (2020a) considers the online version of the problem (1). Therefore, Karimireddy et al. (2020a) require stricter assumptions, including the bounded inter-client gradient variance assumption. Also, their method calculates the full gradient in every communication round.

4 Contributions

We propose a *new family of methods* DASHA-PP for the nonconvex distributed optimization. This is the first method that includes three key ingredients of federated learning methods: *variance reduction of stochastic gradients, compressed communication, and partial participation*. We prove convergence rates and show that these methods have *the optimal oracle complexity and the state-of-the-art communication complexity in the partial participation setting.* Moreover, in our work, we observe a nontrivial side-effect from mixing the variance reduction of stochastic gradients and partial participation. It is a general problem not related to our methods or analysis that we discuss in Section 7.

5 Algorithm Description

We now present DASHA-PP (see Algorithm 1), a family of methods to solve the optimization problem (1). DASHA-PP is based on DASHA by Tyurin and Richtárik (2022). One can easily show that DASHA-PP reduces to DASHA when $p_{\text{a}} = 1$. The refinement of DASHA is not an exercise, let us point out the main differences:

i) The theoretical analysis of DASHA-PP is more complicated: while in DASHA, the randomness from compressors is independent of the randomness from stochastic gradients, in DASHA-PP, these two randomnesses are coupled by the randomness from the partial participation. Moreover, the new methods have to reduce the variance from partial participation.

ii) In the gradient setting, comparing the structure of algorithms DASHA-PP and DASHA, one can see that in DASHA-PP we added at least two crucial things: the momentum b, which helps to reduce the variance of partial participation randomness, and the proper scaling by ${}^{1}/_{p_{\text{a}}}$. Note that in finite-sum and stochastic settings, in DASHA-PP-FINITE-

Algorithm 1 DASHA-PP

```
1: Input: starting point $x^0 \in \mathbb{R}^d$, stepsize $\gamma > 0$, momentum $a \in (0, 1]$, momentum $b \in (0, 1]$, probability $p_{\text{page}} \in (0, 1]$ (only in DASHA-PP-PAGE), batch size $B$ (only in DASHA-PP-PAGE, DASHA-PP-FINITE-MVR and DASHA-PP-MVR), probability $p_{\text{a}} \in (0, 1]$ that a node is participating(a), number of iterations $T \geq 1$
2: Initialize $g_i^0 \in \mathbb{R}^d$, $h_i^0 \in \mathbb{R}^d$ on the nodes and $g^0 = \frac{1}{n}\sum_{i=1}^n g_i^0$ on the server
3: Initialize $h_{ij}^0 \in \mathbb{R}^d$ on the nodes and take $h_i^0 = \frac{1}{m}\sum_{j=1}^m h_{ij}^0$ (only in DASHA-PP-FINITE-MVR)
4: for $t = 0, 1, \ldots, T - 1$ do
5:    $x^{t+1} = x^t - \gamma g^t$
6:    Broadcast $x^{t+1}, x^t$ to all participating(a) nodes
7:    for $i = 1, \ldots, n$ in parallel do
8:       if $i^{\text{th}}$ node is participating(a) then
9:          Calculate $k_i^{t+1}$ using Algorithm 2, 3, 4 or 5
10:         $h_i^{t+1} = h_i^t + \frac{1}{p_{\text{a}}} k_i^{t+1}$
11:         $m_i^{t+1} = \mathcal{C}_i\left(\frac{1}{p_{\text{a}}} k_i^{t+1} - \frac{a}{p_{\text{a}}}\left(g_i^t - h_i^t\right)\right)$
12:         $g_i^{t+1} = g_i^t + m_i^{t+1}$
13:         Send $m_i^{t+1}$ to the server
14:      else
15:         $h_{ij}^{t+1} = h_{ij}^t$ (only in DASHA-PP-FINITE-MVR)
16:         $h_i^{t+1} = h_i^t, \quad g_i^{t+1} = g_i^t, \quad m_i^{t+1} = 0$
17:      end if
18:   end for
19:   $g^{t+1} = g^t + \frac{1}{n}\sum_{i=1}^n m_i^{t+1}$
20: end for
21: Output: $\hat{x}^T$ chosen uniformly at random from $\{x^t\}_{k=0}^{T-1}$
    (a): For the formal description see Section 2.2.
```

Algorithm 2 Calculate k_i^{t+1} for DASHA-PP in the gradient setting. See line 9 in Alg. 1

```
$k_i^{t+1} = \nabla f_i(x^{t+1}) - \nabla f_i(x^t) - b\left(h_i^t - \nabla f_i(x^t)\right)$
```

Algorithm 3 Calculate k_i^{t+1} for DASHA-PP-PAGE in the finite-sum setting. See line 9 in Alg. 1

```
Generate a random set $I_i^t$ of size $B$ from $[m]$ with replacement
$k_i^{t+1} = \begin{cases} \nabla f_i(x^{t+1}) - \nabla f_i(x^t) - \frac{b}{p_{\text{page}}}\left(h_i^t - \nabla f_i(x^t)\right), \\ \qquad \text{with probability } p_{\text{page}} \text{ on all participating nodes,} \\ \frac{1}{B}\sum_{j \in I_i^t}\left(\nabla f_{ij}(x^{t+1}) - \nabla f_{ij}(x^t)\right), \\ \qquad \text{with probability } 1 - p_{\text{page}} \text{ on all participating nodes} \end{cases}$
```

Algorithm 4 Calc. k_i^{t+1} for DASHA-PP-FINITE-MVR in the finite-sum setting. See line 9 in Alg. 1

```
Generate a random set $I_i^t$ of size $B$ from $[m]$ without replacement
$k_{ij}^{t+1} = \begin{cases} \frac{m}{B}\left(\nabla f_{ij}(x^{t+1}) - \nabla f_{ij}(x^t) - b\left(h_{ij}^t - \nabla f_{ij}(x^t)\right)\right), & j \in I_i^t, \\ 0, & j \notin I_i^t \end{cases}$
$h_{ij}^{t+1} = h_{ij}^t + \frac{1}{p_{\text{a}}} k_{ij}^{t+1}$
$k_i^{t+1} = \frac{1}{m}\sum_{j=1}^m k_{ij}^{t+1}$
```

Algorithm 5 Calculate k_i^{t+1} for DASHA-PP-MVR in the stochastic setting. See line 9 in Alg. 1

```
Generate i.i.d. samples $\{\xi_{ij}^{t+1}\}_{j=1}^B$ of size $B$ from $\mathcal{D}_i$.
$k_i^{t+1} = \frac{1}{B}\sum_{j=1}^B \nabla f_i(x^{t+1}; \xi_{ij}^{t+1}) - \frac{1}{B}\sum_{j=1}^B \nabla f_i(x^t; \xi_{ij}^{t+1}) - b\left(h_i^t - \frac{1}{B}\sum_{j=1}^B \nabla f_i(x^t; \xi_{ij}^{t+1})\right)$
```

MVR and DASHA-PP-MVR, accordingly, the momentum b plays the dual role; it also helps to reduce the variance of stochastic gradients.

iii) In the finite-sum setting, we present two methods: DASHA-PP-PAGE and DASHA-PP-FINITE-MVR. The former is based on PAGE (Li et al., 2021a) and with small probability p_{page} calculates the full gradients of the functions f_i. The latter always calculates mini-batches, but it needs extra memory $\mathcal{O}\left(dm\right)$ per node to store vectors h_{ij}^t.

6 Theorems

We now present the convergence rates theorems of DASHA-PP in different settings. We will compare the theorems with the results of DASHA. For any setting, suppose that DASHA converges to ε-solution after T communication rounds. Then, ideally, we would expect the convergence of the new algorithms to ε-solution after up to T/p_{a} communication rounds due to the partial participation. The detailed analysis of the algorithms under Polyak-Łojasiewicz condition we provide in Section **??**. Let us define $\Delta_0 := f(x^0) - f^*$.

6.1 Gradient Setting

Theorem 2. *Suppose that Assumptions 1, 2, 3, 7 and 8 hold. Let us take $a = \frac{p_{\text{a}}}{2\omega+1}, b = \frac{p_{\text{a}}}{2-p_{\text{a}}}$,*

$$\gamma \leq \left(L + \left[\frac{48\omega\left(2\omega+1\right)}{np_{\text{a}}^2} + \frac{16}{np_{\text{a}}^2}\left(1 - \frac{p_{\text{aa}}}{p_{\text{a}}}\right)\right]^{1/2}\widehat{L}\right)^{-1},$$

and $g_i^0 = h_i^0 = \nabla f_i(x^0)$ for all $i \in [n]$ in Algorithm 1 (DASHA-PP), then $\mathrm{E}\left[\left\|\nabla f(\widehat{x}^T)\right\|^2\right] \leq \frac{2\Delta_0}{\gamma T}$.

Let us recall the convergence rate of DASHA or MARINA (Gorbunov et al., 2021), the number of communication rounds to get ε-solution equals $\mathcal{O}\left(\frac{\Delta_0}{\varepsilon}\left[L + \frac{\omega}{\sqrt{n}}\widehat{L}\right]\right)$, while the rate of DASHA-PP equals $\mathcal{O}\left(\frac{\Delta_0}{\varepsilon}\left[L + \frac{\omega+1}{p_{\text{a}}\sqrt{n}}\widehat{L}\right]\right)$. Up to Lipschitz constants factors, we get the degeneration up to $1/p_{\text{a}}$ factor due to the partial participation.

6.2 Finite-Sum Setting

Theorem 3. *Suppose that Assumptions 1, 2, 3, 4, 7, and 8 hold. Let us take $a = \frac{p_{\text{a}}}{2\omega+1}$, $b = \frac{p_{page}p_{\text{a}}}{2-p_{\text{a}}}$, probability $p_{page} \in (0, 1]$,*

$$\gamma \leq \left(L + \left[\frac{48\omega(2\omega+1)}{np_{\text{a}}^2}\left(\widehat{L}^2 + \frac{(1-p_{page})L_{\max}^2}{B}\right)\right.\right.$$
$$\left.\left. + \frac{16}{np_{\text{a}}^2 p_{page}}\left(\left(1 - \frac{p_{\text{aa}}}{p_{\text{a}}}\right)\widehat{L}^2 + \frac{(1-p_{page})L_{\max}^2}{B}\right)\right]^{1/2}\right)^{-1}$$

and $g_i^0 = h_i^0 = \nabla f_i(x^0)$ for all $i \in [n]$ in Algorithm 1 (DASHA-PP-PAGE) then $\mathrm{E}\left[\left\|\nabla f(\widehat{x}^T)\right\|^2\right] \leq \frac{2\Delta_0}{\gamma T}$.

We now choose p_{page} to balance heavy full gradient and light mini-batch calculations. Let us define $\mathbb{1}_{p_{\text{a}}} := \sqrt{1 - \frac{p_{\text{aa}}}{p_{\text{a}}}} \in [0, 1]$. Note that if $p_{\text{a}} = 1$ then $p_{\text{aa}} = 1$ and $\mathbb{1}_{p_{\text{a}}} = 0$.

Corollary 1. *Let the assumptions from Theorem 3 hold and $p_{page} = B/(m+B)$. Then DASHA-PP-PAGE needs*

$$T := \mathcal{O}\left(\frac{\Delta_0}{\varepsilon}\left[L + \frac{\omega}{p_{\text{a}}\sqrt{n}}\left(\widehat{L} + \frac{L_{\max}}{\sqrt{B}}\right)\right.\right. \tag{6}$$
$$\left.\left. + \frac{1}{p_{\text{a}}}\sqrt{\frac{m}{n}}\left(\frac{\mathbb{1}_{p_{\text{a}}}\widehat{L}}{\sqrt{B}} + \frac{L_{\max}}{B}\right)\right]\right)$$

communication rounds to get an ε-solution and the expected number of gradient calculations per node equals $\mathcal{O}\left(m + BT\right)$.

The convergence rate the rate of the current state-of-the-art method DASHA-PAGE without partial participation equals $\mathcal{O}\left(\frac{\Delta_0}{\varepsilon}\left[L + \frac{\omega}{\sqrt{n}}\left(\widehat{L} + \frac{L_{\max}}{\sqrt{B}}\right) + \sqrt{\frac{m}{n}}\frac{L_{\max}}{B}\right]\right)$. Let us closer compare it with (6). As expected, we see that the second term w.r.t. ω degenerates up to $1/p_{\text{a}}$. Surprisingly, the third term w.r.t. $\sqrt{m/n}$ can degenerate up to $\sqrt{B}/p_{\text{a}}$ when $\widehat{L} \approx L_{\max}$. Hence, in order to keep degeneration up to $1/p_{\text{a}}$, one should take the batch size $B = \mathcal{O}\left(L_{\max}^2/\widehat{L}^2\right)$. This interesting effect we analyze separately in Section 7.

In the following corollary, we consider Rand$K$ compressors (see Definition **??**) and show that with the particular choice of parameters, up to the Lipschitz constants and probability p_{a} factor, DASHA-PP-PAGE gets the optimal oracle complexity and SOTA communication complexity. The choice of the compressor is driven by simplicity, and the following analysis can be used for other unbiased compressors.

Corollary 2. *Suppose that assumptions of Corollary 1 hold, $B \leq \min\left\{\frac{1}{p_{\text{a}}}\sqrt{\frac{m}{n}}, \frac{L_{\max}^2}{\mathbb{1}_{p_{\text{a}}}^2\widehat{L}^2}\right\}$[3], and we use the unbiased compressor RandK with $K = \Theta\left(Bd/\sqrt{m}\right)$. Then the communication complexity of Algorithm 1 is*

$$\mathcal{O}\left(d + \frac{L_{\max}\Delta_0 d}{p_{\text{a}}\varepsilon\sqrt{n}}\right), \tag{7}$$

and the expected number of gradient calculations per node equals

$$\mathcal{O}\left(m + \frac{L_{\max}\Delta_0\sqrt{m}}{p_{\text{a}}\varepsilon\sqrt{n}}\right). \tag{8}$$

The convergence rate of DASHA-PP-FINITE-MVR is provided in Section **??**. The conclusions are the same for the method.

[3] If $\mathbb{1}_{p_{\text{a}}} = 0$, then $\frac{L_\sigma^2}{\mathbb{1}_{p_{\text{a}}}^2\widehat{L}^2} = +\infty$

6.3 Stochastic Setting

We define $h^t := \frac{1}{n}\sum_{i=1}^n h_i^t$.

Theorem 4. *Suppose that Assumptions 1, 2, 3, 5, 6, 7 and 8 hold. Let us take* $a = \frac{p_{\mathrm{a}}}{2\omega+1}$, $b \in \left(0, \frac{p_{\mathrm{a}}}{2-p_{\mathrm{a}}}\right]$,

$$\gamma \leq \left(L + \left[\frac{48\omega(2\omega+1)}{np_{\mathrm{a}}^2}\left(\widehat{L}^2 + \frac{(1-b)^2L_\sigma^2}{B}\right) + \frac{12}{np_{\mathrm{a}}b}\left(\left(1 - \frac{p_{\mathrm{aa}}}{p_{\mathrm{a}}}\right)\widehat{L}^2 + \frac{(1-b)^2L_\sigma^2}{B}\right)\right]^{1/2}\right)^{-1},$$

and $g_i^0 = h_i^0$ *for all* $i \in [n]$ *in Algorithm 1* (DASHA-PP-MVR). *Then*

$$\mathrm{E}\left[\left\|\nabla f(\widehat{x}^T)\right\|^2\right] \leq \frac{1}{T}\left[\frac{2\Delta_0}{\gamma} + \frac{2}{b}\left\|h^0 - \nabla f(x^0)\right\|^2 + \left(\frac{32b\omega(2\omega+1)}{np_{\mathrm{a}}^2} + \frac{4\left(1 - \frac{p_{\mathrm{aa}}}{p_{\mathrm{a}}}\right)}{np_{\mathrm{a}}}\right)\left(\frac{1}{n}\sum_{i=1}^n\left\|h_i^0 - \nabla f_i(x^0)\right\|^2\right)\right] + \left(\frac{48b^2\omega(2\omega+1)}{p_{\mathrm{a}}^2} + \frac{12b}{p_{\mathrm{a}}}\right)\frac{\sigma^2}{nB}.$$

In the next corollary, we choose momentum b and initialize vectors h_i^0 to get ε-solution. Let us define $\mathbb{1}_{p_{\mathrm{a}}} := \sqrt{1 - \frac{p_{\mathrm{aa}}}{p_{\mathrm{a}}}} \in [0,1]$.

Corollary 3. *Suppose that assumptions from Theorem 4 hold, momentum* $b = \Theta\left(\min\left\{\frac{p_{\mathrm{a}}}{\omega}\sqrt{\frac{n\varepsilon B}{\sigma^2}}, \frac{p_{\mathrm{a}}n\varepsilon B}{\sigma^2}\right\}\right)$, $\frac{\sigma^2}{n\varepsilon B} \geq 1$, *and* $h_i^0 = \frac{1}{B_{\text{init}}}\sum_{k=1}^{B_{\text{init}}}\nabla f_i(x^0;\xi_{ik}^0)$ *for all* $i \in [n]$, *and batch size* $B_{\text{init}} = \Theta\left(\frac{\sqrt{p_{\mathrm{a}}}B}{b}\right)$, *then Algorithm 1* (DASHA-PP-MVR) *needs*

$$T := \mathcal{O}\left(\frac{\Delta_0}{\varepsilon}\left[L + \frac{\omega}{p_{\mathrm{a}}\sqrt{n}}\left(\widehat{L} + \frac{L_\sigma}{\sqrt{B}}\right) + \frac{\sigma}{p_{\mathrm{a}}\sqrt{\varepsilon n}}\left(\frac{\mathbb{1}_{p_{\mathrm{a}}}\widehat{L}}{\sqrt{B}} + \frac{L_\sigma}{B}\right)\right] + \frac{\sigma^2}{\sqrt{p_{\mathrm{a}}}n\varepsilon B}\right)$$

communication rounds to get an ε*-solution and the number of stochastic gradient calculations per node equals* $\mathcal{O}(B_{\text{init}} + BT)$.

The convergence rate of the DASHA-SYNC-MVR, the state-of-the-art method without partial participation, equals $\mathcal{O}\left(\frac{\Delta_0}{\varepsilon}\left[L + \frac{\omega}{\sqrt{n}}\left(\widehat{L} + \frac{L_\sigma}{\sqrt{B}}\right) + \frac{\sigma}{\sqrt{\varepsilon n}}\frac{L_\sigma}{B}\right] + \frac{\sigma^2}{n\varepsilon B}\right)$. Similar to Section 6.2, we see that in the regimes when $\widehat{L} \approx L_\sigma$ the third term w.r.t. ${}^1/_{\varepsilon^{3/2}}$ can degenerate up to ${}^{\sqrt{B}}/_{p_{\mathrm{a}}}$. However, if we take $B = \mathcal{O}\left({}^{L_\sigma^2}/_{\widehat{L}^2}\right)$, then the degeneration of the third term will be up to ${}^1/_{p_{\mathrm{a}}}$. This effect we analyze in Section 7.

In the following corollary, we consider Rand$K$ compressors (see Definition **??**) and show that with the particular choice of parameters, up to the Lipschitz constants and probability p_{a} factor, DASHA-PP-MVR gets the optimal oracle complexity and SOTA communication complexity of DASHA-SYNC-MVR method.

Corollary 4. *Suppose that assumptions of Corollary 3 hold, batch size* $B \leq \min\left\{\frac{\sigma}{p_{\mathrm{a}}\sqrt{\varepsilon n}}, \frac{L_\sigma^2}{\mathbb{1}_{p_{\mathrm{a}}}^2\widehat{L}^2}\right\}$, *we take Rand*$K$ *compressors with* $K = \Theta\left(\frac{Bd\sqrt{\varepsilon n}}{\sigma}\right)$. *Then the communication complexity equals*

$$\mathcal{O}\left(\frac{d\sigma}{\sqrt{p_{\mathrm{a}}}\sqrt{n\varepsilon}} + \frac{L_\sigma\Delta_0 d}{p_{\mathrm{a}}\sqrt{n\varepsilon}}\right), \qquad (9)$$

and the expected number of stochastic gradient calculations per node equals

$$\mathcal{O}\left(\frac{\sigma^2}{\sqrt{p_{\mathrm{a}}}n\varepsilon} + \frac{L_\sigma\Delta_0\sigma}{p_{\mathrm{a}}\varepsilon^{3/2}n}\right). \qquad (10)$$

We are aware that the initial batch size B_{init} can be suboptimal w.r.t. ω in DASHA-PP-MVR in some regimes (see also (Tyurin and Richtárik, 2022)). This is a side effect of mixing the variance reduction of stochastic gradients and compression. However, Corollary 4 reveals that we can escape these regimes by choosing the parameter K of Rand$K$ compressors in a particular way. To get the complete picture, we analyze the same phenomenon under PŁ condition (see Section **??**) and provide a new method DASHA-PP-SYNC-MVR (see Section **??**).

7 The Problem of Estimating the Mean in the Partial Participation Setting

We now provide the example to explain why the only choice of $B = \mathcal{O}\left(\min\left\{\frac{1}{p_{\mathrm{a}}}\sqrt{\frac{m}{n}}, \frac{L_{\max}^2}{\mathbb{1}_{p_{\mathrm{a}}}^2\widehat{L}^2}\right\}\right)$ and $B = \mathcal{O}\left(\min\left\{\frac{\sigma}{p_{\mathrm{a}}\sqrt{\varepsilon n}}, \frac{L_\sigma^2}{\mathbb{1}_{p_{\mathrm{a}}}^2\widehat{L}^2}\right\}\right)$ in DASHA-PP-PAGE and DASHA-PP-MVR, accordingly, guarantees the degeneration up to ${}^1/_{p_{\mathrm{a}}}$. This is surprising, because in methods with the variance reduction of stochastic gradients (Tyurin and Richtárik, 2022; Li et al., 2021a) we can take the size of batch size $B = \mathcal{O}\left(\sqrt{\frac{m}{n}}\right)$ and $B = \mathcal{O}\left(\frac{\sigma}{\sqrt{\varepsilon n}}\right)$ and guarantee the optimality. Note that the smaller the batch size B, the more the server and the nodes have to communicate to get ε-solution.

Let us consider the task of estimating the mean of vectors in the distributed setting. Suppose that we have n nodes, and each of them contains m vectors $\{x_{ij}\}_{j=1}^m$, where $x_{ij} \in \mathbb{R}^d$ for all $i \in [n], j \in [m]$. First, let us consider that each node samples a mini-batch I^i of size B with replacement and sends it to the server. Then the server calculates the mean of the mini-batches from nodes. One can easily show that the variance of the estimator is

$$\mathrm{E}\left[\left\|\frac{1}{nB}\sum_{i=1}^{n}\sum_{j\in I^i}x_{ij}-\frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}x_{ij}\right\|^2\right] \tag{11}$$

$$=\frac{1}{nB}\frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}\left\|x_{ij}-\frac{1}{m}\sum_{j=1}^{m}x_{ij}\right\|^2.$$

Next, we consider the same task in the partial participation setting with s–nice sampling, i.e., we sample a random set $S\subset[n]$ of $s\in[n]$ nodes without replacement and receive the mini-batches only from the sampled nodes. Such sampling of nodes satisfy Assumption 8 with $p_{\mathrm{a}}={}^{s}\!/_{n}$ and $p_{\mathrm{a}}={}^{s(s-1)}\!/_{n(n-1)}$. In this case, the variance of the estimator (See Lemma **??** with $r_i=0$ and $s_i=\sum_{j\in I^i}x_{ij}$) is

$$\mathrm{E}\left[\left\|\frac{1}{sB}\sum_{i\in S}\sum_{j\in I^i}x_{ij}-\frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}x_{ij}\right\|^2\right] \tag{12}$$

$$=\frac{1}{sB}\underbrace{\frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}\left\|x_{ij}-\frac{1}{m}\sum_{j=1}^{m}x_{ij}\right\|^2}_{\mathcal{L}^2_{\max}}$$

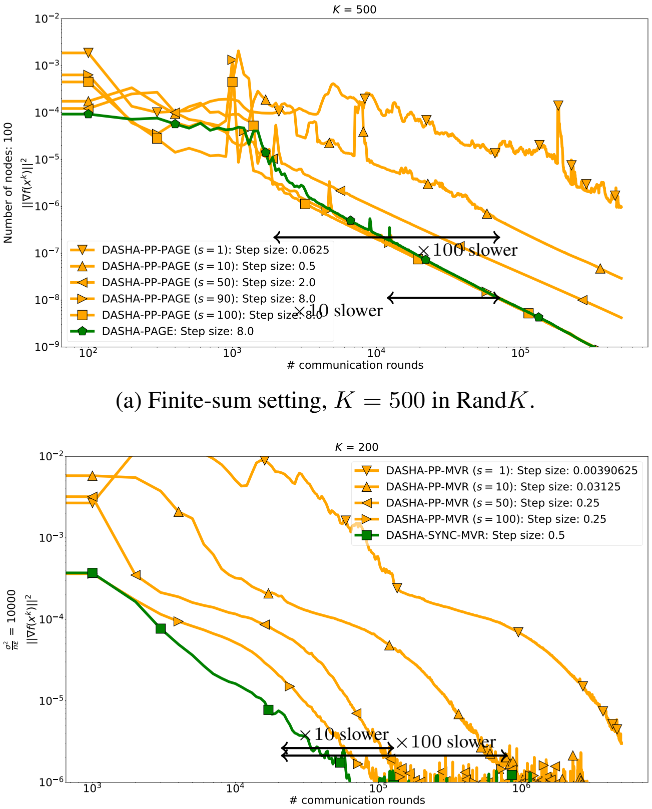
$$+\frac{n-s}{s(n-1)}\underbrace{\frac{1}{n}\sum_{i=1}^{n}\left\|\frac{1}{m}\sum_{j=1}^{m}x_{ij}-\frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}x_{ij}\right\|^2}_{\widehat{\mathcal{L}}^2}.$$

Let us assume that $s\le{}^{n}\!/_{2}$. Note that (11) scales with any $B\ge1$, while (12) only scales when $B=\mathcal{O}\left({}^{\mathcal{L}^2_{\max}}\!/_{\widehat{\mathcal{L}}^2}\right)$. In other words, for large enough B, the variance in (12) does not significantly improves with the growth of B due to the term $\widehat{\mathcal{L}}^2$. In our proof, due to partial participation, the variance from (12) naturally appears, and we get the same effect. As was mentioned in Sections 6.2 and 6.3, it can be seen in our convergence rate bounds.

8 Experiments

In experiments[4], our main goal is to compare DASHA-PP with DASHA. Clearly, DASHA-PP can not generally perform better than DASHA. In different settings, we verify that the bigger p_{a}, the closer DASHA-PP is to DASHA, i.e., DASHA-PP converges no slower than ${}^{1}\!/_{p_{\mathrm{a}}}$ times. We use the standard setting in experiments where all parameters except step sizes are taken as suggested in theory. Step sizes are finetuned from a set $\{2^i\,|\,i\in[-10,10]\}$. We emulate the partial participation setting using s-nice sampling with the number of nodes $n=100$. We consider the Rand$K$ compressor and take the batch size $B=1$. All algorithms are tested on machine learning classification tasks with nonconvex loss functions (see details in Section **??**). We plot the relation between communication rounds and values of the norm of gradients at each communication round.

In the finite-sum (Figure 1a) and in the stochastic setting (Figure 1b), we see that the bigger probability $p_{\mathrm{a}}={}^{s}\!/_{n}$ to 1, the closer DASHA-PP to DASHA. Moreover, DASHA-PP with $s=10$ and $s=1$ converges approximately $\times10$ $(={}^{1}\!/_{p_{\mathrm{a}}})$ and $\times100$ $(={}^{1}\!/_{p_{\mathrm{a}}})$ times slower, accordingly. Our theory predicts such behavior.



(a) Finite-sum setting, $K=500$ in Rand$K$.

(b) Stochastic setting, ${}^{\sigma^2}\!/_{n\varepsilon B}=10000$, and $K=200$ in Rand$K$.

Figure 1: Classification task with the *real-sim* dataset.

References

Alistarh, D., Grubic, D., Li, J., Tomioka, R., and Vojnovic, M. (2017). QSGD: Communication-efficient SGD via gradient quantization and encoding. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1709–1720.

Arjevani, Y., Carmon, Y., Duchi, J. C., Foster, D. J., Srebro, N., and Woodworth, B. (2019). Lower bounds for non-convex stochastic optimization. *arXiv preprint arXiv:1912.02365*.

Beznosikov, A., Horváth, S., Richtárik, P., and Safaryan, M.

[4] Code: https://github.com/mysteryresearcher/dasha-partial-participation

(2020). On biased compression for distributed learning. *arXiv preprint arXiv:2002.12410*.

Carmon, Y., Duchi, J. C., Hinder, O., and Sidford, A. (2020). Lower bounds for finding stationary points i. *Mathematical Programming*, 184(1):71–120.

Cutkosky, A. and Orabona, F. (2019). Momentum-based variance reduction in non-convex SGD. *arXiv preprint arXiv:1905.10018*.

Fang, C., Li, C. J., Lin, Z., and Zhang, T. (2018). SPIDER: Near-optimal non-convex optimization via stochastic path integrated differential estimator. In *NeurIPS Information Processing Systems*.

Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. (2016). *Deep learning*, volume 1. MIT Press.

Gorbunov, E., Burlachenko, K., Li, Z., and Richtárik, P. (2021). MARINA: Faster non-convex distributed learning with compression. In *38th International Conference on Machine Learning*.

Horváth, S., Ho, C.-Y., Horvath, L., Sahu, A. N., Canini, M., and Richtárik, P. (2019a). Natural compression for distributed deep learning. *arXiv preprint arXiv:1905.10988*.

Horváth, S., Kovalev, D., Mishchenko, K., Stich, S., and Richtárik, P. (2019b). Stochastic distributed learning with gradient quantization and variance reduction. *arXiv preprint arXiv:1904.05115*.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456. PMLR.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al. (2021). Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210.

Karimireddy, S. P., Jaggi, M., Kale, S., Mohri, M., Reddi, S. J., Stich, S. U., and Suresh, A. T. (2020a). Mime: Mimicking centralized stochastic algorithms in federated learning. *arXiv preprint arXiv:2008.03606*.

Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S., Stich, S., and Suresh, A. T. (2020b). Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pages 5132–5143. PMLR.

Konečný, J., McMahan, H. B., Yu, F. X., Richtárik, P., Suresh, A. T., and Bacon, D. (2016). Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. (2020). Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2:429–450.

Li, Z., Bao, H., Zhang, X., and Richtárik, P. (2021a). PAGE: A simple and optimal probabilistic gradient estimator for nonconvex optimization. In *International Conference on Machine Learning*, pages 6286–6295. PMLR.

Li, Z., Hanzely, S., and Richtárik, P. (2021b). ZeroSARAH: Efficient nonconvex finite-sum optimization with zero full gradient computation. *arXiv preprint arXiv:2103.01447*.

Lin, Y., Han, S., Mao, H., Wang, Y., and Dally, W. J. (2017). Deep gradient compression: Reducing the communication bandwidth for distributed training. *arXiv preprint arXiv:1712.01887*.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR.

Mishchenko, K., Gorbunov, E., Takáč, M., and Richtárik, P. (2019). Distributed learning with compressed gradient differences. *arXiv preprint arXiv:1901.09269*.

Narayanan, D., Harlap, A., Phanishayee, A., Seshadri, V., Devanur, N. R., Ganger, G. R., Gibbons, P. B., and Zaharia, M. (2019). PipeDream: generalized pipeline parallelism for dnn training. In *Proceedings of the 27th ACM Symposium on Operating Systems Principles*, pages 1–15.

Nesterov, Y. (2018). *Lectures on convex optimization*, volume 137. Springer.

Nguyen, L., Liu, J., Scheinberg, K., and Takáč, M. (2017). SARAH: A novel method for machine learning problems using stochastic recursive gradient. In *The 34th International Conference on Machine Learning*.

Ramaswamy, S., Mathews, R., Rao, K., and Beaufays, F. (2019). Federated learning for emoji prediction in a mobile keyboard. *arXiv preprint arXiv:1906.04329*.

Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I. (2021). Zero-shot text-to-image generation. *arXiv preprint arXiv:2102.12092*.

Reddi, S., Charles, Z., Zaheer, M., Garrett, Z., Rush, K., Konečnỳ, J., Kumar, S., and McMahan, H. B. (2020). Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*.

Richtárik, P., Sokolov, I., and Fatkhullin, I. (2021). EF21: A new, simpler, theoretically better, and practically faster error feedback. *In Neural Information Processing Systems, 2021*.

Richtárik, P., Sokolov, I., Fatkhullin, I., Gasanov, E., Li, Z., and Gorbunov, E. (2022). 3PC: Three point compressors for communication-efficient distributed training and a better theory for lazy aggregation. *arXiv preprint arXiv:2202.00998*.

Sapio, A., Canini, M., Ho, C.-Y., Nelson, J., Kalnis, P., Kim, C., Krishnamurthy, A., Moshref, M., Ports, D. R., and Richtárik, P. (2019). Scaling distributed machine learning with in-network aggregation. *arXiv preprint arXiv:1903.06701*.

Stich, S. U., Cordonnier, J.-B., and Jaggi, M. (2018). Sparsified SGD with memory. *Advances in Neural Information Processing Systems*, 31.

Suresh, A. T., Sun, Z., Ro, J. H., and Yu, F. (2022). Correlated quantization for distributed mean estimation and optimization. *arXiv preprint arXiv:2203.04925*.

Szlendak, R., Tyurin, A., and Richtárik, P. (2021). Permutation compressors for provably faster distributed nonconvex optimization. *arXiv preprint arXiv:2110.03300*.

Tyurin, A. and Richtárik, P. (2022). DASHA: Distributed nonconvex optimization with communication compression, optimal oracle complexity, and no client synchronization. *arXiv preprint arXiv:2202.01268*.

Vogels, T., He, L., Koloskova, A., Karimireddy, S. P., Lin, T., Stich, S. U., and Jaggi, M. (2021). RelaySum for decentralized deep learning on heterogeneous data. *Advances in Neural Information Processing Systems*, 34.

Wangni, J., Wang, J., Liu, J., and Zhang, T. (2018). Gradient sparsification for communication-efficient distributed optimization. *Advances in Neural Information Processing Systems*, 31.

Xu, H., Ho, C.-Y., Abdelmoniem, A. M., Dutta, A., Bergou, E. H., Karatsenidis, K., Canini, M., and Kalnis, P. (2021). Grace: A compressed communication framework for distributed machine learning. In *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS)*, pages 561–572. IEEE.

Zhao, H., Burlachenko, K., Li, Z., and Richtárik, P. (2021a). Faster rates for compressed federated learning with client-variance reduction. *arXiv preprint arXiv:2112.13097*.

Zhao, H., Li, Z., and Richtárik, P. (2021b). FedPAGE: A fast local stochastic gradient method for communication-efficient federated learning. *arXiv preprint arXiv:2108.04755*.