Ordered Gradient Approach for Communication-Efficient Distributed Learning

Yicheng Chen
Lehigh University
Bethlehem, PA, USA
yic917@lehigh.edu

Brian M. Sadler
Army Research Laboratory
Adelphi, MD, USA
brian.m.sadler6.civ@mail.mil

Rick S. Blum
Lehigh University
Bethlehem, PA, USA
rblum@lehigh.edu

***Abstract*—The topic of training machine learning models by employing multiple gradient-computing workers is attracting great interest recently. Communication efficiency in such distributed learning settings is an important consideration, especially for the case where the needed communications are expensive in terms of power usage. We develop a new approach which is efficient in terms of communication transmissions. In this scheme, only the most informative worker results are transmitted to reduce the total number of transmissions. Our ordered gradient approach provably achieves the same order of convergence rate as gradient descent for nonconvex smooth loss functions while gradient descent always requires more communications. Experiments show significant communication savings compared to the best existing approaches in some cases.**

***Index Terms*—communication efficiency, distributed learning, ordered transmissions.**

I. Introduction

In this paper, we propose an ordering-based communication-efficient algorithm to solve the following problem

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^d} \mathcal{L}(\boldsymbol{\theta}) \quad \text{with} \quad \mathcal{L}(\boldsymbol{\theta}) \triangleq \sum_{m\in\mathcal{M}} \mathcal{L}_m(\boldsymbol{\theta}) \tag{1}$$

where $\boldsymbol{\theta} \in \mathbb{R}^d$ is the parameter vector to be optimized, $\mathcal{L}(\boldsymbol{\theta})$ is the objective function to be minimized, $\mathcal{L}_m(\boldsymbol{\theta})$ is the local objective function for worker m with $m \in \mathcal{M}$, and $\mathcal{M} = \{1, 2, ..., M\}$ is a set to collect the indices of all workers. The problem in (1) has been successfully applied to model multi-agent systems [1], distributed learning [2] [3], and distributed processing in sensor networks [4]. In a distributed learning scenario, $\mathcal{L}_m(\boldsymbol{\theta}) \triangleq \sum_{n=1}^{N_m} \ell(\boldsymbol{\theta}; \boldsymbol{x}_n, y_n)$ at worker m is a sum of the loss functions $\ell(\boldsymbol{\theta}; \boldsymbol{x}_n, y_n)$ for $n = 1, 2, ..., N_m$ where $\boldsymbol{x}_n$ is the n-th feature vector and y_n is the corresponding label. In order to reduce privacy and security risk, each worker usually does not transmit its local dataset to the server [5]. Instead, each worker calculates a gradient of its local function, and only this gradient is communicated to the server. The server aggregates all gradients and updates the globally shared parameter vector $\boldsymbol{\theta}$ by running gradient-based algorithms. The learning task is solved after many communications between workers and the server.

High communication cost in distributed learning can be a serious bottleneck [5] [6]. Hence, communication efficiency is an important consideration in distributed learning. Methods for improving communication efficiency include reducing the number of communications and compressing gradients at each worker, see [3] [5], [7] and references therein. It is worth mentioning that the recent algorithm called LAG-WK in [3] employs the censoring idea in distributed learning where workers transmit only highly informative updates. The communications are skipped when a worker does not have a sufficiently different gradient from its previously transmitted one, and the server reuses previously sent but still accurate gradients, which is very reasonable. Censoring in distributed learning reduces communications but loses some information which might be useful. The ordering approach we discuss next can reduce the loss and provide better performance with fewer communications in some cases.

Focusing on a different problem, [8]–[10] employ the idea of ordered transmissions for the hypothesis testing problem where workers with the most informative observations transmit first. Transmissions can be halted when sufficient information is accumulated for the sever to decide which hypothesis is true. In this paper, we employ ordering in distributed learning to eliminate some worker-to-server uplink communications normally needed in the gradient descent (GD) approach. The resultant gradient-based approach is called the ordered gradient (OG) approach. OG is guaranteed to achieve the same order convergence rate as GD for nonconvex smooth loss functions. We provide numerical results that show that OG can eliminate communications required by GD and LAG-WK in some cases.

The paper is organized as follows. We introduce the OG algorithm in Section II and present the convergence analysis in Section III. Section IV contains several numerical examples. We conclude the paper in Section V. Throughout this paper, we use bold lower case letters to denote column vectors. We use $\|\mathbf{x}\|$ to denote the ℓ_2-norm of $\mathbf{x}$. The notation for transpose is $(\cdot)^\top$. The set of workers that do and do not transmit at iteration k are denoted by $\mathcal{M}^k$ and $\mathcal{M}_c^k$, respectively. We use $|\mathcal{A}|$ to denote the cardinality of the set $\mathcal{A}$.

The work is supported by the U. S. Army Research Laboratory and the U. S. Army Research Office under grant number W911NF-17-1-0331, the National Science Foundation under Grant ECCS-1744129, and a grant from the Commonwealth of Pennsylvania, Department of Community and Economic Development, through the Pennsylvania Infrastructure Technology Alliance (PITA).

978-1-7281-5478-7/20/$31.00 ©2020 IEEE

II. Ordered Gradient Approach

In this section, we describe our OG approach to implement GD with a smaller number of transmissions. In our approach, worker m ($m = 1, 2, ..., M$) maintains 2 vectors at each iteration k. The first is a parameter vector $\boldsymbol{\theta}^k$ which is received from the server at the start of iteration k, and the other vector is the last gradient that was transmitted from the worker to the server prior to the start of iteration k, called $\nabla\mathcal{L}_m(\hat{\boldsymbol{\theta}}_m^{k-1})$. An important feature of OG is that worker m is not allowed to transmit the gradient $\nabla\mathcal{L}_m(\boldsymbol{\theta}^k)$ to the server if the gradient $\nabla\mathcal{L}_m(\boldsymbol{\theta}^k)$ is not sufficiently different from the last gradient it transmitted $\nabla\mathcal{L}_m(\hat{\boldsymbol{\theta}}_m^{k-1})$. Define the difference as $\delta\nabla_m^k \triangleq \nabla\mathcal{L}_m(\boldsymbol{\theta}^k) - \nabla\mathcal{L}_m(\hat{\boldsymbol{\theta}}_m^{k-1})$. To implement the OG algorithm, at the beginning of iteration k (called time t_k) the server broadcasts $\boldsymbol{\theta}^k$ to all workers and initializes the set of workers who have transmitted as $\mathcal{M}^k = \emptyset$. Then worker m determines a time $\tau/\|\delta\nabla_m^k\|$ to transmit $\delta\nabla_m^k$ to the server, where the positive number τ can be made as small as the system will allow. Thus the first worker to transmit will transmit $\delta\nabla_{(1)}^k$ and the second worker to transmit will transmit $\delta\nabla_{(2)}^k$ if it does transmit and this process continues such that

$$\|\delta\nabla_{(1)}^k\| > \|\delta\nabla_{(2)}^k\| > ... > \|\delta\nabla_{(m)}^k\| > ... > \|\delta\nabla_{(M)}^k\| \quad (2)$$

where (m) is the index of the worker who has the m-th largest term $\|\delta\nabla_{(m)}^k\|$, so that the most informative data is transmitted first. Immediately after transmitting, each worker will update their transmitted gradient $\nabla\mathcal{L}_m(\hat{\boldsymbol{\theta}}_m^k) = \nabla\mathcal{L}_m(\boldsymbol{\theta}^k)$ while others keep their previous values $\nabla\mathcal{L}_m(\hat{\boldsymbol{\theta}}_m^k) = \nabla\mathcal{L}_m(\hat{\boldsymbol{\theta}}_m^{k-1})$. If the server receives a transmission from worker m, the server will add worker m into $\mathcal{M}^k$. After the OG stopping condition (described later) is satisfied, the server updates the parameter using

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^k - \alpha\nabla^k \quad \text{with} \quad \nabla^k = \nabla^{k-1} + \sum_{m=1}^{|\mathcal{M}^k|} \delta\nabla_{(m)}^k \quad (3)$$

where ∇^k is equivalent to

$$\nabla^k = \nabla\mathcal{L}\left(\boldsymbol{\theta}^k\right) - \sum_{m=|\mathcal{M}^k|+1}^{M} \delta\nabla_{(m)}^k \quad (4)$$

$$= \nabla\mathcal{L}\left(\boldsymbol{\theta}^k\right) - \sum_{m\in\mathcal{M}_c^k} \delta\nabla_m^k. \quad (5)$$

This approach reduces the number of communications at iteration k from M in GD to $|\mathcal{M}^k| < M$ in OG. Note that (5) is obtained from (4) since the corresponding workers belong to $\mathcal{M}_c^k$.

Note that if we want to increase $|\mathcal{M}_c^k|$ to save communications, then more iterations might be required. The key to balance this tradeoff between communications and iterations is to expect OG to have a larger reduction in the objective function value $\mathcal{L}(\boldsymbol{\theta}^k)$ per communication for the k-th update shown in (3) when compared to GD. Before comparing the descent amount, we first review the basic descent lemma for GD [11].

Lemma 1: If $\mathcal{L}(\boldsymbol{\theta})$ is L-smooth[1] and the step size is $\alpha = 1/L$, the GD update satisfies

$$\mathcal{L}\left(\boldsymbol{\theta}^{k+1}\right) - \mathcal{L}\left(\boldsymbol{\theta}^k\right) \le \Delta_{\mathrm{GD}}^k(\boldsymbol{\theta}^k), \quad (6)$$

where $\Delta_{\mathrm{GD}}^k(\boldsymbol{\theta}^k) \triangleq -\frac{1}{2L}\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}^k\right)\right\|^2$.
The descent lemma of OG is different from that of GD due to skipping some communications, as seen in *Lemma 2*.

Lemma 2: If $\mathcal{L}(\boldsymbol{\theta})$ is L-smooth and the step size is $\alpha = 1/L$, the OG update yields

$$\mathcal{L}\left(\boldsymbol{\theta}^{k+1}\right) - \mathcal{L}\left(\boldsymbol{\theta}^k\right) \le \Delta_{\mathrm{OG}}^k(\boldsymbol{\theta}^k) \quad (7)$$

where $\Delta_{\mathrm{OG}}^k(\boldsymbol{\theta}^k) \triangleq -\frac{1}{2L}\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}^k\right)\right\|^2 + \frac{|\mathcal{M}_c^k|^2}{2L}\left\|\delta\nabla_{(|\mathcal{M}^k|)}^k\right\|^2$.

Proof: Using *Lemma 2* in [3] (which does not study OG), we find

$$\mathcal{L}\left(\boldsymbol{\theta}^{k+1}\right) - \mathcal{L}\left(\boldsymbol{\theta}^k\right) \le -\frac{\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}^k\right)\right\|^2}{2L} + \frac{\left\|\sum_{m\in\mathcal{M}_c^k}\delta\nabla_m^k\right\|^2}{2L} \quad (8)$$

when we choose $\alpha = 1/L$ in [3]. Ordered transmissions imply

$$\left\|\sum_{m\in\mathcal{M}_c^k}\delta\nabla_m^k\right\|^2 \le |\mathcal{M}_c^k|^2 \cdot \left\|\delta\nabla_{(|\mathcal{M}^k+1|)}^k\right\|^2 \le |\mathcal{M}_c^k|^2 \cdot \left\|\delta\nabla_{(|\mathcal{M}^k|)}^k\right\|^2. \quad (9)$$

Plugging (9) into (8) leads to (7). ∎

It is worth mentioning that $\frac{|\mathcal{M}_c^k|^2}{2L}\left\|\delta\nabla_{(|\mathcal{M}^k|)}^k\right\|^2$ is the cost of skipping communications and this cost can be ignored when the most recent transmission $\delta\nabla_{(|\mathcal{M}^k|)}^k$ has a very small magnitude. Note that $|\mathcal{M}_c^k|$ in (9) is known at the server since the server can count the number of transmissions that have already been received at any given time. In the censoring approach in [3], the number of transmissions is fixed prior to observing any data. A desirable criterion for OG to select $\mathcal{M}^k$ is to ensure that OG results in a greater objective function descent per uplink communication than GD, that is

$$\frac{\Delta_{\mathrm{OG}}^k(\boldsymbol{\theta}^k)}{|\mathcal{M}^k|} \le \frac{\Delta_{\mathrm{GD}}^k(\boldsymbol{\theta}^k)}{M} \quad (10)$$

which is equivalent to

$$\left\|\delta\nabla_{(|\mathcal{M}^k|)}^k\right\|^2 \le \frac{\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}^k\right)\right\|^2}{M\cdot|\mathcal{M}_c^k|}. \quad (11)$$

Thus when (11) is true, it would be desirable for the worker transmissions to be halted to end the iteration. However, the server is unable to check (11) since calculating $\nabla\mathcal{L}(\boldsymbol{\theta}^k)$ would require every worker to transmit its information to the server but the server only receives the first $|\mathcal{M}^k|$ worker

[1]A function $\mathcal{L}(\boldsymbol{\theta})$ is L-smooth if there exists a constant $L \ge 0$ such that $\|\nabla\mathcal{L}(\boldsymbol{\theta}_1) - \nabla\mathcal{L}(\boldsymbol{\theta}_2)\| \le L\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|,\ \forall\ \boldsymbol{\theta}_1, \boldsymbol{\theta}_2$ [11].

transmissions. Instead, by employing Young's inequality[2] to (5), we obtain

$$\left\|\nabla^k\right\|^2 \leq (1+\rho)\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}^k\right)\right\|^2 + (1+\rho^{-1})\left\|\sum_{m\in\mathcal{M}_c^k}\delta\nabla_m^k\right\|^2 \quad (12)$$

where ρ is any positive number. The result in (12) implies

$$\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}^k\right)\right\|^2 \geq \frac{\left\|\nabla^k\right\|^2}{1+\rho} - \frac{1+\rho^{-1}}{1+\rho}\left\|\sum_{m\in\mathcal{M}_c^k}\delta\nabla_m^k\right\|^2 \quad (13)$$

$$\geq \frac{\left\|\nabla^k\right\|^2}{1+\rho} - \frac{1+\rho^{-1}}{1+\rho}\cdot|\mathcal{M}_c^k|^2\left\|\delta\nabla^k_{(|\mathcal{M}^k|)}\right\|^2 \quad (14)$$

where (14) is obtained since $\|\delta\nabla^k_{(|\mathcal{M}^k|)}\| > \|\delta\nabla_m^k\|$ for all $m\in\mathcal{M}_c^k$. We can obtain a sufficient condition for (11) if we replace $\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}^k\right)\right\|^2$ in (11) with the terms on the right-hand side of the inequality in (14), which yields,

$$\left\|\delta\nabla^k_{(|\mathcal{M}^k|)}\right\|^2 \leq \frac{\left\|\nabla^k\right\|^2}{|\mathcal{M}_c^k|\left((1+\rho^{-1})|\mathcal{M}_c^k| + (1+\rho)M\right)}. \quad (15)$$

In order to save more transmissions, the positive parameter ρ can be chosen based on

$$\min_{\rho}|\mathcal{M}_c^k|\left((1+\rho^{-1})|\mathcal{M}_c^k| + (1+\rho)M\right). \quad (16)$$

The parameter ρ that minimizes (16) is $\rho = \sqrt{|\mathcal{M}_c^k|/M}$. Finally, we obtain our OG stopping condition which is

$$\left\|\delta\nabla^k_{(|\mathcal{M}^k|)}\right\|^2 \leq \frac{\left\|\nabla^k\right\|^2}{M|\mathcal{M}_c^k|}\frac{1}{\left(1+\sqrt{\frac{|\mathcal{M}_c^k|}{M}}\right)^2}. \quad (17)$$

Note that we do not use any approximation in our derivation from (10) to (17) which implies the original criterion in (10) still holds when we employ (17) to select $\mathcal{M}^k$. At a given iteration, the right-hand side of (17), that we refer to as the OG threshold, is updated according to the information the server has received so far, which is different from censoring [3] where all workers have the same threshold for a given iteration.

Worker m does not transmit in censoring if [3],

$$\|\delta\nabla_m^k\|^2 \leq \frac{\sum_{d=1}^D \xi_d\left\|\boldsymbol{\theta}^{k+1-d} - \boldsymbol{\theta}^{k-d}\right\|^2}{\alpha^2M^2} \quad (18)$$

and we will call the right-hand side of (18) the censoring threshold. We point out that our OG threshold contains the newest approximated gradient ∇^k which can not be obtained at iteration k in the censoring algorithm in (18) but it can be computed using (3) in OG. This newest information can help OG outperform censoring when the gradient of the objective function changes significantly. Compared with the censoring threshold in (18), the server in OG does not need to tune the constants $\{\xi_d\}_{d=1}^D$ to decide a proper OG stopping condition in (17).

[2]Young's inequality is $\|\boldsymbol{a}+\boldsymbol{b}\|^2 \leq (1+\rho)\|\boldsymbol{a}\|^2 + \left(1+\rho^{-1}\right)\|\boldsymbol{b}\|^2$ where ρ is any positive number

To evaluate (17), the server needs to broadcast the current parameter $\boldsymbol{\theta}^k$ at the start of iteration k. Worker m for all $m = 1, 2, ..., M$ computes $\nabla\mathcal{L}_m(\boldsymbol{\theta}^k)$ and transmits $\delta\nabla_m^k$ (if allowed by (17)) after a time equal to $\tau/\|\delta\nabla_m^k\|$ where the positive number τ can be made as small as the system will allow. The server checks the OG stopping condition in (17) after receiving each transmission. If (17) is satisfied, then the server immediately updates $\boldsymbol{\theta}^k$ and starts the next iteration by transmitting $\boldsymbol{\theta}^{k+1}$. Any workers who did not yet transmit will not transmit during iteration k. The OG algorithm is summarized as Algorithm 1. If all transmission propagation delays are known and timing is synchronized, one can schedule all transmissions back to the server so they arrive in the correct order. However, even with inaccurate estimates of propagation delays or imperfect synchronization, even with some small ordering errors, the server can put them back in order correctly as long as the server waits a short period related to the uncertainty; see [12].

Algorithm 1 OG.

Input: The step size is $\alpha = 1/L$, and a positive number τ.
Initialize: $\boldsymbol{\theta}^1$, ∇^0, $\{\nabla\mathcal{L}_m(\hat{\boldsymbol{\theta}}_m^0), \forall m\}$

for $k = 1, 2, ..., K$ **do**
  Server broadcasts $\boldsymbol{\theta}^k$ at time t_k which is denoted as the starting time of iteration k.
  Server sets $m = 1$ and initializes $\mathcal{M}^k = \emptyset$.
  while $m \leq M$ and the stopping condition (17) is not satisfied **do**
    $\tau/\|\delta\nabla^k_{(m)}\|$ seconds after t_k, $\delta\nabla^k_{(m)}$ is transmitted to server where (m) is the index of the worker who has the m-th largest term in (2).
    Worker (m) updates $\nabla\mathcal{L}_{(m)}(\hat{\boldsymbol{\theta}}^k_{(m)}) = \nabla\mathcal{L}_{(m)}(\boldsymbol{\theta}^k)$.
    Server adds worker (m) into $\mathcal{M}^k$.
    $m = m + 1$.
  end while
  Server updates via (3).
  $k = k + 1$.
end for

III. CONVERGENCE AND COMMUNICATION ANALYSIS

In this section we provide convergence analysis of OG under the following assumption.

Assumption 1: Objective function $\mathcal{L}(\boldsymbol{\theta})$ is L-smooth, continuously differentiable and bounded below by a constant function (in $\boldsymbol{\theta}$) $\mathcal{L}^* \in \mathbb{R}$.

Theorem 1: If *Assumption 1* holds and the step size $\alpha = 1/L$, then the iterates $\{\boldsymbol{\theta}^k\}$ of OG satisfy

$$\min_{1\leq k\leq K}\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}^k\right)\right\|^2 = o(1/K). \quad (19)$$

Proof: Recall that (17) is a sufficient condition to guarantee (10) which implies that

$$\mathcal{L}\left(\boldsymbol{\theta}^{k+1}\right) - \mathcal{L}\left(\boldsymbol{\theta}^k\right) \leq -\frac{1}{2L}\cdot\frac{\left|\mathcal{M}^k\right|}{M}\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}^k\right)\right\|^2. \quad (20)$$

Thus, summing over $k \in \{0, 1, 2, ..., K\}$, we have

$$\frac{1}{2L}\sum_{k=0}^{K}\frac{\left|\mathcal{M}^k\right|}{M}\left\|\nabla\mathcal{L}(\boldsymbol{\theta}^k)\right\|^2 \leq \mathcal{L}\left(\boldsymbol{\theta}^0\right) - \mathcal{L}\left(\boldsymbol{\theta}^{K+1}\right). \quad (21)$$

Since $\mathcal{L}(\boldsymbol{\theta})$ is bounded below by a constant $\mathcal{L}^*$, we know that

$$\mathcal{L}\left(\boldsymbol{\theta}^0\right) - \mathcal{L}\left(\boldsymbol{\theta}^{K+1}\right) \leq \mathcal{L}\left(\boldsymbol{\theta}^0\right) - \mathcal{L}^* < \infty. \quad (22)$$

Combining (21) with (22), it follows that

$$\lim_{K\to\infty}\sum_{k=0}^{K}\frac{\left|\mathcal{M}^k\right|}{M}\left\|\nabla\mathcal{L}(\boldsymbol{\theta}^k)\right\|^2 < \infty \quad (23)$$

which implies

$$\lim_{k\to\infty}\frac{\left|\mathcal{M}^k\right|}{M}\left\|\nabla\mathcal{L}(\boldsymbol{\theta}^k)\right\|^2 \to 0. \quad (24)$$

We note that the OG algorithm transmits at least once during each iteration which implies that $\left|\mathcal{M}^k\right| \geq 1$ for all k. Thus, we finally obtain

$$\lim_{k\to\infty}\left\|\nabla\mathcal{L}(\boldsymbol{\theta}^k)\right\|^2 \to 0. \quad (25)$$

Employing the result of summable sequences in [13] (*Lemma 3*), the result in (19) follows. ∎

The result in (25) indicates that the OG algorithm must converge to a stationary point without requiring a convex objective function $\mathcal{L}(\boldsymbol{\theta})$. Note that *Theorem* 1 shows that OG can achieve the same order of convergence rate as GD under nonconvex smooth cases [11].

From numerical results given in next section, we observe that OG is powerful for a system with a sufficiently large number of workers when the data samples are similar among different workers. With censoring, sufficiently similar data samples among different workers will cause all workers to transmit or none of the workers to transmit. If the system has a large number of workers, many communications will occur in censoring when all workers violate the censoring condition whereas OG can reduce the number of transmissions because of its adaptive stopping rule for different workers at each iteration.

IV. Numerical Results

In order to illustrate the convergence analysis and demonstrate communication savings, here we present a few numerical results for linear regression with the objective function at worker m being

$$\mathcal{L}_m(\boldsymbol{\theta}) \triangleq \sum_{n=1}^{N_m}\left(y_n - \mathbf{x}_n^\top\boldsymbol{\theta}\right)^2 \quad (26)$$

where $\boldsymbol{\theta} \in \mathbb{R}^d$ is the parameter vector, $\mathbf{x}_n$ is the n-th feature vector and y_n is the corresponding label. To benchmark OG, we compare it with two methods, the GD method and the censoring-based GD method (called LAG-WK in [3]). For the censoring-based GD method, we use (18) with $D = 1$ and $\xi_1 = 0.25$ as the censoring threshold. The step size α for GD, LAG-WK and OG is chosen as $1/L$.

We first consider a scenario with one server and nine workers. For worker m, we set $y_n = 1$ for $n = 1, 2, ..., N_m$ and fix $d = N_m = 50$. We set $\mathbf{X}$ as a diagonal matrix with its diagonal elements being $200(j-1)/49$ for $j = 1, 2, ..., 50$. We set $\mathbf{X}_m = \mathbf{X} + \mathbf{N}_m = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{N_m}]^\top$ where $\mathbf{N}_m$ is a random noise matrix with its elements being independent and standard normally distributed. Fig. 1 indicates that OG and LAG-WK require nearly the same number of iterations as GD to achieve the same objective error. Results in Fig. 1 also indicate that OG needs fewer number of uplink communications than GD and LAG-WK and ordering can save transmissions by employing its adaptive stopping rule. Note that in this case all the workers have similar data samples.

Next we test performance using a real dataset called winequality-red [14] in a scenario with a server and 120 workers. We collect the feature vectors in the first 11 columns of the dataset as a feature matrix $\mathbf{X}$ and set the data in the 12th column of the dataset as the labels y_n. We set $\mathbf{X}_m = \mathbf{X} + \mathbf{N}_m$ with the elements of $\mathbf{N}_m$ being independent and standard normally distributed. Fig. 2 shows that OG and LAG-WK have nearly the same convergence rate as GD. Results in Fig. 2 also illustrate that the number of uplink communications required by OG is significantly smaller than that of LAG-WK for this case where the system has a large number of workers and each worker has similar datasets $\mathbf{X}_m$ for $m = 1, 2, ..., M$.

Now we consider a different synthetic data test where we scale the data at different workers very differently. We assume a scenario with one server and nine workers where for each worker m, we use the same method as [3] to randomly generate feature vectors $\mathbf{x}_n \in \mathbb{R}^{50}$ and labels y_n and rescale the data to mimic the increasing smoothness constants $L_m = (1.3^{m-1} + 4)^2$. Fig. 3 shows that OG outperforms the alternatives in terms of the number of communications saved while employing a smaller number of iterations than GD. Additionally, Fig. 4 is obtained by rescaling the data to mimic the increasing smoothness constants $L_m = (1.3^{m-1})^2$. Fig. 3 and Fig. 4 together indicate that the communication saving gains of OG over LAG-WK are problem specific for the case where different workers have very different data. Further investigation will be pursued in future work.

V. Conclusion

A new class of communication-efficient distributed learning algorithms called OG has been described that attempt to decrease the number of transmissions needed. In OG, each worker sometimes transmits the difference between its own current gradient and its last transmitted gradient. OG can provably achieve the same order convergence rate as GD for a nonconvex smooth objective function. Numerical results employing linear regression models have shown that our new approach can reduce the total number of transmissions to achieve a targeted objective error when the system has a large number of workers and data samples are similar among different workers.

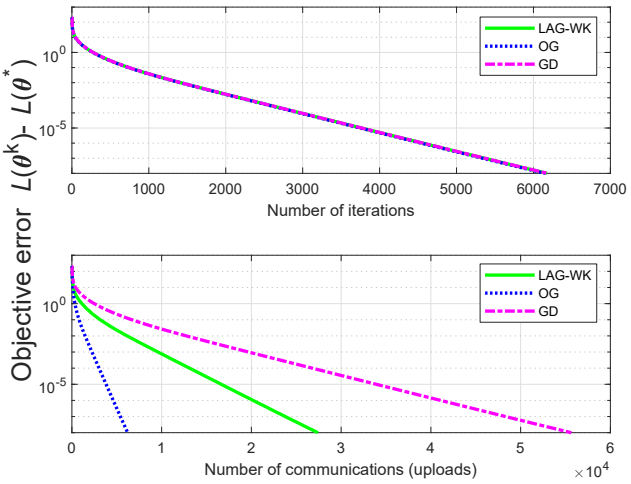

Fig. 1. Objective error versus number of iterations and communications in the scenario with 9 workers and uniform smoothness constants between workers.

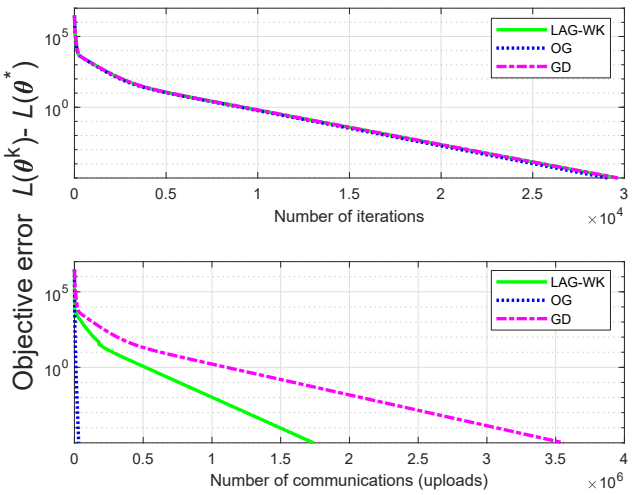

Fig. 2. Objective error versus number of iterations and communications in winequality-red dataset with 120 workers.

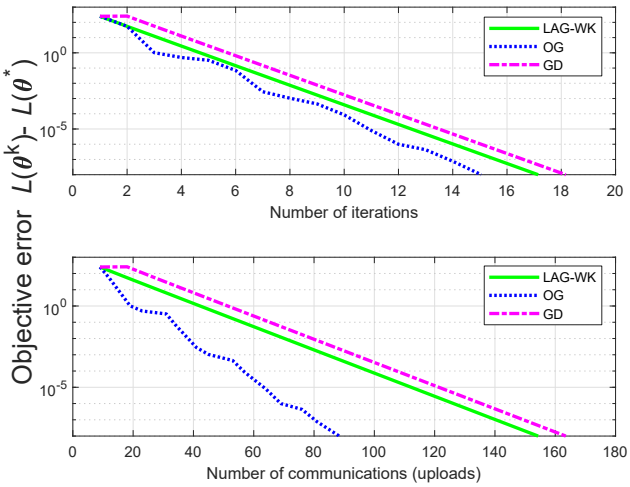

Fig. 3. Objective error versus number of iterations and communications in the scenario with 9 workers and $L_m = (1.3^{m-1} + 4)^2$ between workers.

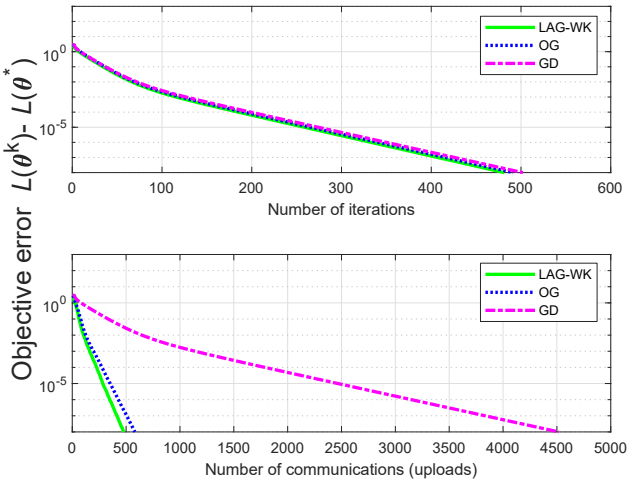

Fig. 4. Objective error versus number of iterations and communications in the scenario with 9 workers and $L_m = (1.3^{m-1})^2$ between workers.

REFERENCES

[1] C. Sun, M. Ye, and G. Hu, "Distributed time-varying quadratic optimization for multiple agents under undirected graphs," *IEEE Transactions on Automatic Control*, vol. 62, no. 7, pp. 3687–3694, 2017.

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

[3] T. Chen, G. Giannakis, T. Sun, and W. Yin, "Lag: Lazily aggregated gradient for communication-efficient distributed learning," in *Advances in Neural Information Processing Systems*, 2018, pp. 5050–5060.

[4] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *Proceedings of the 3rd international symposium on Information processing in sensor networks*. ACM, 2004, pp. 20–27.

[5] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *AISTATS*, 2016.

[6] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 4424–4434.

[7] F. Seide, H. Fu, J. Droppo, G. Li, and D. Yu, "1-bit stochastic gradient descent and its application to data-parallel distributed training of speech dnns," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[8] R. S. Blum and B. M. Sadler, "Energy efficient signal detection in sensor networks using ordered transmissions," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 3229–3235, 2008.

[9] Y. Chen, B. M. Sadler, and R. S. Blum, "Ordered transmission for efficient wireless autonomy," in *2018 52nd Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2018, pp. 1299–1303.

[10] Y. Chen, R. S. Blum, B. M. Sadler, and J. Zhang, "Testing the structure of a gaussian graphical model with reduced transmissions in a distributed setting," *IEEE Transactions on Signal Processing*, vol. 67, no. 20, pp. 5391–5401, Oct 2019.

[11] Y. Nesterov, *Lectures on convex optimization*. Springer, 2018, vol. 137.

[12] R. S. Blum, "Ordering for estimation and optimization in energy efficient sensor networks," *IEEE Transactions on Signal Processing*, vol. 59, no. 6, pp. 2847–2856, 2011.

[13] D. Davis and W. Yin, "Convergence rate analysis of several splitting schemes," in *Splitting methods in communication, imaging, science, and engineering*. Springer, 2016, pp. 115–163.

[14] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis, "Modeling wine preferences by data mining from physicochemical properties," *Decision Support Systems*, vol. 47, no. 4, pp. 547–553, 2009. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Wine+Quality