An ADMM-based Algorithm for Zeroth-order Stochastic Optimization over Distributed Processing Networks

***Abstract*—In this paper, we address the problem of stochastic optimization over distributed processing networks, which is motivated by machine learning applications performed in data centers. In this problem, each of a total n nodes in a network receives stochastic realizations of a private function $f_i(x)$ and aims to reach a common value that minimizes $\sum_{i=1}^n f_i(x)$ via local updates and communication with its neighbors. We focus on zeroth-order methods where only function values of stochastic realizations can be used. Such kind of methods, which are also called derivative-free, are especially important in solving real-world problems where either the (sub)gradients of loss functions are inaccessible or inefficient to be evaluated. To this end, we propose a method called Distributed Stochastic Alternating Direction Method of Multipliers (DS-ADMM) which can choose to use two kinds of gradient estimators for different assumptions. The convergence rates of DS-ADMM are $O(n\sqrt{k\log(2k)/T})$ for general convex loss functions and $O(nk\log(2kT)/T)$ for strongly convex functions, where k is the dimension of domain and T is the time horizon of the algorithm. The rates can be improved to $O(n/\sqrt{T})$ and $O(n\log T/T)$ if objective functions have Lipschitz gradients. All these results are better than previous distributed zeroth-order methods. Lastly, we demonstrate the performance of DS-ADMM via experiments of two examples called distributed online least square and distributed support vector machine arising in estimation and classification tasks.**

I. Introduction

Today many applications need to deal with big data, which makes them difficult to be processed by a single machine. Therefore, a common practice to break a big task into small ones and process them separately in a network, such as the MapReduce framework [7]. Machine learning practice, especially deep learning [11], involve both big data and intensive optimization tasks, which further enhances the necessity of distributed optimization methods.

Distributed optimization aims to optimize a loss function $\sum_{i=1}^n f_i(x)$, where f_i is only accessible by a unique node in a network consisting of n nodes. The nodes communicate with their neighbors to reach the consensus of the global optimizer. This topic dates back to Tsitsiklis et al.'s work [30] and has seen renewed interest and advances with techniques such as dual averaging [8], ADMM [20] and Nesterov's acceleration [25]. The readers may refer to [22] for more comprehensive details.

In this paper, we consider a stochastic version of distributed optimization where $f_i(x) = \mathbb{E}_{\xi_i\sim\mathcal{D}_i}[F_i(x;\xi_i)]$ that arises in numerous machine learning applications. During the optimization process, we only observe or use stochastic realizations of $f_i(x)$, i.e., $F_i(x;\xi_i(t))$ at time t. This setup can be applied to expected risk minimization and empirical risk minimization in machine learning [4]. For expected risk minimization, $\mathcal{D}_i$ is the data generating distribution in Node i, and for empirical risk minimization, $\mathcal{D}_i$ is a uniform distribution over the data points stored in Node i. In both cases, F_i is an associated loss function parameterized by x.

Function Type	Convergence Rate
Convex	$O(n\sqrt{k\log(2k)/T})$
Convex & SC	$O(nk\log(2k)\log(T)/T)$
Convex & LG	$O(n/\sqrt{T})$
Convex & LG & SC	$O(n\log T/T)$

TABLE I: Convergence rates of Zeroth-order DS-ADMM under different properties of F_i. "LG" stands for Lipschitz Gradient and "SC" stands for Strongly Convex.

Previous works mainly explore first-order methods using (sub)gradients of $F_i(x;\xi_i)$ (See Section I-A). In some cases, (sub)gradients are inaccessible or hard to be evaluated. Such cases include simulation-based optimization [12], bandit optimization [14] and objectives without simple gradient expressions. For these applications, we need zeroth-order (also called derivative-free or non-derivative) methods where we only utilize function values of $F_i(x;\xi_i)$.

In this paper, we propose an algorithm called Distributed Stochastic Alternating Direction Method of Multipliers (DS-ADMM) for distributed stochastic optimization using zeroth-order information, which can choose to use two kinds of gradient estimators for different assumptions. The advantages of DS-ADMM are summarized as follows:

- DS-ADMM is in fact a unified method which can seamlessly switch between a first-order method and a zeroth-order method naturally. It is done by choosing to use the (sub)gradients or the (sub)gradient estimators in one step of DS-ADMM. In this paper we focus on the zeroth-order version of DS-ADMM, which is directly referred as DS-ADMM for simplicity if not specified.
- Convergence rates of DS-ADMM are shown in Table I, where k is the dimension of domain and T is the time horizon of the algorithm. These results are the main contributions of our paper, because to the best of our knowledge, they are the fastest rates compared with previous methods.
- Through two applications in distributed estimation and classification problems, DS-ADMM is shown to have a faster rate than previous methods using zeroth-order

information.

The organization of the paper is as follows: In Section II, we describe our problem setup and its applications in distributed processing networks. To solve the problem, we propose DS-ADMM and explain its procedure in Section III. The convergence results of DS-ADMM are analyzed in Section IV and V together. In Section VI we use two applications to demonstrate the performance of DS-ADMM and in Section VII we give our conclusion.

A. Related Works

For brevity, we only list the works which give the convergence rates of their algorithms for distributed stochastic optimization.

In Duchi et al.'s work [8], a distributed dual averaging algorithm was proposed to achieve a $O(n\log T/\sqrt{T})$ rate for general convex functions. For time-varying directed graphs, Nedic and Olshevsky [21] investigated a subgradient-push method with a convergence rate of $O(\log T/T)$ for strongly convex functions. The authors in the work [17] presented a class of decentralized primal-dual type algorithms to achieve a $O(1/T)$ (respectively, $O(1/\sqrt{T})$) convergence rate for general convex (respectively, strongly convex) functions. For nonconvex functions, the algorithm in the work [18] had a rate of $O(1/T+1/\sqrt{nT})$ to obtain a first-order stationary solution. By utilizing a time-dependent weighted mixing of history values, the work [27] achieved a convergence rate of $O(n\sqrt{n}/T)$ for strongly convex functions. For delayed gradient information, the authors in the work [28] showed a method with a convergence rate of $O(1/\sqrt{T})$ for general convex functions. Another work [16] presented a method with a $O(1/T)$ rate for strongly convex functions and random networks. The authors in the work [24] introduced two methods, DSGT and GSGT, which enjoy a linear rate converging to a neighborhood of the optimality and a $O(1/T)$ rate converging to the exact solution for smooth and strongly convex functions.

So if we ignore n[1], the fastest convergence rate of existing first-order methods is $O(1/\sqrt{T})$ for general convex functions and $O(1/T)$ for strongly convex functions.

To change a first-order method to a zero-order method needs to be carefully designed because existing gradient estimators [2], [10] used in zeroth-order methods are biased estimators. The work [26] developed a Kiefer-Wolfowitz type method by using the deterministic gradient estimator [2], which achieved a $O(1/\sqrt{T})$ mean square convergence rate for smooth and strongly convex functions. Nonconvex functions were dealt with in [13], but their convergence metric is related to magnitude of gradients.

B. Notation

We denote $\partial F(x;\xi)$ as the subgradient set of F at x for sample ξ. If p is a number, $||x||_p$ denotes the l_p-norm for a vector x and $||A||_p$ is the matrix norm of A, induced by the l_p norm. For a matrix A, $||x||_A^2 = x^TAx$. x^T and A^T are defined as the transpose of x and A, respectively. $\langle\cdot,\cdot\rangle$ is the standard inner product of two vectors. Meanwhile, I_k is a $k\times k$ identity matrix. $\mathbf{0}$ and $\mathbf{1}$ are vectors with all entries equal to 0 and 1, respectively. $A\otimes B$ is the Kronecker product of A and B. dom F is the domain of the function F. diag$(a_1,...,a_n)$ is a $n\times n$ diagonal matrix when entries $a_i\in\mathbb{R}$ and diag$(A_1,...,A_n)$ is a $nk\times nk$ block diagonal matrix with diagonal $k\times k$ matrix A_i. $\pi_\chi\{\centerdot\}$ is the projection operator with regard to a set $\mathcal{X}$, which is defined as

$$\pi_\chi\{x\} = \arg\min_{y\in\mathcal{X}} ||y-x||_2^2$$

II. Problem Setup and Applications

We consider a stochastic optimization problem formulated as:

$$\min_{x\in\mathcal{X}} \sum_{i=1}^{n} f_i(x) \tag{1}$$

where $f_i(x) = \mathbb{E}_{\xi_i\sim\mathcal{D}_i}[F_i(x;\xi_i)]$ and $\mathcal{X}\subset\mathbb{R}^k$ is some constraint set. We assume that the above problem is processed by a network represented by an undirected, connected and weighed graph $G=(V,E)$ where $V=\{1,...,n\}$ is the set of nodes and E is the set of edges. The communication cost of a link is represented by the weight of the corresponding edge. In this paper, we focus on the algorithms where Node i only observes or uses stochastic realizations of $f_i(x)$, i.e., $F_i(x,\xi_i(t))$ at time t. For a first-order algorithm, (sub)gradients of $F_i(x,\xi_i(t))$ can be used, and for a zeroth-order algorithm, we can only use the function values of $F_i(x,\xi_i(t))$ queried at any point x. Each node of the network can only share its information with its neighbors in order to reach the consensus.

Distributed stochastic optimization has many applications in practice. Here we will present two examples, and explain why zeroth methods are needed in these two examples.

The first example stems from distributed online least squares estimation (OLS) in sensor networks. Consider a sensor network which aims to estimate a signal $\hat{x}$. Each sensor receives an observation $\xi_i(t)$ at time t, which is modeled as $\xi_i(t) = H_i^T\hat{x} + w_i(t)$. $H_i^T\hat{x}$ is a linear response of the signal in Sensor i and $w_i(t)$ is white noise sampled from independent and identically Gaussian distributions at time t. The sensors use the observations to cooperatively solve the least square problem $\min_{x\in\mathcal{X}}\sum_{i=1}^n \mathbb{E}_{\xi_i\sim\mathcal{D}_i}||H_i^Tx-\xi_i||_2^2$ where $\mathcal{X}$ is some constraint set. Now we can see that in this setup, $f_i(x)=\mathbb{E}_{\xi_i\sim\mathcal{D}_i}||H_i^Tx-\xi_i||_2^2$, and we can observe its stochastic realizations $F_i(x;\xi_i(t)) = ||H_i^Tx-\xi_i(t)||_2^2$ sequentially. If H_i is time-varying because of environmental factors, H_i needs to re-evaluated every certain time. So it may be not efficient to use first-order methods when H_i has a high dimension (we need to know H_i to use gradients). In this case, each sensor can obtain H_i^Tx by directly giving an input x produced by itself, where the white noise can be neglected. Then values of $||H_i^Tx-\xi_i(t)||_2^2$ at any x can be obtained to use a zeroth-order method.

The second example is distributed support vector machine (SVM) formulation for optimal classification. Consider a bunch

[1]Because the dependence on n is only explored in some of the above works.

of servers connected by a network dealing with a classification task. Server i stores a partition of the total training set, whose s^{th} item is $(\gamma_i(s), \varphi_i(s))$. Here $\varphi_i(s) \in \{1, ..., m\}$ is the class of $\gamma_i(s)$. Suppose data points have $m > 2$ classes. For linear SVM, we choose a vector x^j (j is just a superscript) for class j and use a linear model $x^j\gamma_i(s)$ to represent $\gamma_i(s)$'s *score* of class j. Then we classify $\gamma_i(s)$ into the class with the highest score. A suggested loss function [6] of $x = (x^1, .., x^m)$ we choose for data point $(\gamma_i(s), \varphi_i(s))$ is

$$\begin{aligned} &F_i(x; \gamma_i(s), \varphi_i(s)) \\ &= \max(0, 1 + \max_{j \neq \varphi_i(s)} x^j\gamma_i(s) - x^{\varphi_i(s)}\gamma_i(s)) \end{aligned} \tag{2}$$

If we want to minimize the training loss, then $f_i(x) = \frac{1}{S_i}\sum_{s=1}^{S_i} F_i(x; \gamma_i(s), \varphi_i(s))$, where S_i is the number of data points in Server i. When a data point $(\gamma_i(s), \varphi_i(s))$ is uniformly chosen from the training set in Server i, $F_i(x; \gamma_i(s), \varphi_i(s))$ is a stochastic realization of $f_i(x)$ and can be used in stochastic optimization methods like stochastic gradient descent [4] to reduce computation cost. Meanwhile, (sub)gradients of F_i are dependent on values of F_i, and become more and more complex as m increases. Because of this, it is more convenient to directly use function values. So zeroth-order methods are preferred in this case when m is large.

III. Algorithm Design: DS-ADMM

In the previous section we presented two of many possible scenarios whereby only stochastic realizations are accessible to or used by the distributed servers that aim to find a common parameter that minimizes a total cost, with the need of zeroth-order methods. In this section, we introduce and explain our algorithm called Distributed Stochastic Alternating Direction Method of Multipliers (DS-ADMM), which can operate both with first-order and with zeroth-order information, as necessary.

The algorithm is shown in Algorithm 1. In the algorithm, $x_i(t), y_i(t), p_i(t)$ are the values of node i at time t and η_t is a time-varying stepsize. For the ease of discussion, we define $G' = (V', E')$ as the minimum spanning tree of G if Step 1 is executed, and $G' = G$ otherwise. In this algorithm, $N(i) = \{j|(i, j) \in E'\} \cup \{i\}$, and d_i is the number of neighbors of node i in G'. $A_{ij} \in \mathbb{R}$ is the (i, j)th entry of A and $A \in \mathbb{R}^{n\times n}$ is a communication matrix shown later in Assumption 1 of Section IV. Meanwhile,

$$\begin{aligned} h_t(x_i) &= g_i(t)^T(x_i - x_i(t)) + \sum_{j\in N(i)} (A_{ji}p_j(t)^T x_i \\ &+ \frac{c}{2}||y_j(t) + A_{ji}(x_i - x_i(t))||_2^2) + \frac{1}{2\eta_t}||x_i - x_i(t)||_{G_i(t)}^2 \end{aligned} \tag{3}$$

$$\begin{aligned} \tilde{h}_t(x_i) &= \tilde{g}_i(t)^T(x_i - x_i(t)) + \sum_{j\in N(i)} (A_{ji}p_j(t)^T x_i \\ &+ \frac{c}{2}||y_j(t) + A_{ji}(x_i - x_i(t))||_2^2) + \frac{1}{2\eta_t}||x_i - x_i(t)||_{G_i(t)}^2 \end{aligned} \tag{4}$$

where $G_i(t) \in \mathbb{R}^{k\times k}$ is a positive definite matrix specified later in Assumption 1 of Section IV, η_t is a stepsize,

Algorithm 1 Distributed Stochastic ADMM

1: (**Optional**) Find the minimum spanning tree G' of the network according to the weights and deactivate other links not in the tree.
2: Initialize $c \in \mathbb{R}$, $x_i(1) \in \mathcal{X}$, $p_i(0) = \mathbf{0} \in \mathbb{R}^k$ for each i.
3: **for** $t = 1, ..., T$ **do**
4: $\quad y_i(t) = \frac{1}{d_i+1}\sum_{j\in N(i)} A_{ij}x_j(t)$, $\forall i \in \{1, ..., n\}$.
5: $\quad p_i(t) = p_i(t-1) + cy_i(t)$, $\forall i \in \{1, ..., n\}$.
6: $\quad$ **if** first-order information can be used **then**
7: $\quad\quad x_i(t+1) = \arg\min_{x_i\in\mathcal{X}} h_t(x_i)$, $\forall i \in \{1, ..., n\}$, where $h_t(x_i)$ is shown in (3).
8: $\quad$ **else if** only zeroth-order information can be used **then**
9: $\quad\quad x_i(t+1) = \arg\min_{x_i\in\mathcal{X}} \tilde{h}_t(x_i)$, $\forall i \in \{1, ..., n\}$, where $\tilde{h}_t(x_i)$ is shown in (4).
10: $\quad$ **end if**
11: **end for**
12: **Output:** $\bar{x}_i(T) = \frac{1}{T}\sum_{t=1}^{T} x_i(t)$

$g_i(t) \in \partial F_i(x_i(t); \xi_i(t))$ and $\tilde{g}_i(t)$ is an estimation of $g_i(t)$ using function values of $F_i(x; \xi_i(t))$. The detailed estimator will be presented in Section V.

DS-ADMM is similar to Multi-agent Distributed ADMM (MD-ADMM) proposed in the work [20]. We first formulates the original problem as:

$$\begin{aligned} &\min_{x\in\mathcal{X}} \sum_{i=1}^{n} \mathbb{E}_{\xi_i\sim\mathcal{D}_i}[f_i(x_i; \xi_i)] \\ &s.t \quad (A \otimes I_k)\tilde{x} = 0 \end{aligned}$$

where $\tilde{x} = (x_1^T, ..., x_n^T)^T \in \mathbb{R}^{nk}$. $(A \otimes I_k)\tilde{x} = 0$ represents the consensus requirement in distributed optimization. By introducing an auxiliary variable, we can decouple the variable in each node and use an ADMM-type method, where p_i's are multipliers. The reader may refer to the work [20] for the transformation detail.

Even though inspired by MD-ADMM, DS-ADMM differs from it non-negligibly at Step 7 for the first-order and Step 9 for the zeroth-order version. In particular, our design makes the algorithm unnecessary to know $f_i(x)$, since only stochastic realizations are known. It results that Steps 7 and 9 become constrained minimizations of second-order approximations instead of unconstrained minimizations of the original functions used in MD-ADMM, which poses difficulties for analysis. In implementation perspective, if we let

$$G_i(t) = (\alpha - c\eta_t \sum_{j\in N(i)} A_{ji}^2)I_k \tag{5}$$

where α is a parameter to make $G_i(t)$ positive definite, then Step 7 of DS-ADMM becomes

$$x_i(t+1) = \Pi_{\mathcal{X}}\{x_i(t) - \frac{\eta_t}{\alpha}(g_i(t) + \sum_{j\in N(i)} A_{ji}(cy_j(t) + p_j(t)))\},$$

which is an operation like projected gradient descent. This operation is much more simplified compared with Step b) of Algorithm 1 in [20].

The above differences are not trivial for convergence rate analysis of our algorithm. First, the optimality condition of constrained optimization is different from the unconstrained case [20]. Second, since MD-ADMM is used in the setting of deterministic optimization, its convergence results cannot be modified to the ones in our setting directly. Therefore, we need a new proof for our algorithm.

Step 1 of DS-ADMM can help reduce the communication cost of the algorithm. Moreover, in Section IV, we will show that if Step 1 is executed, a more direct measure of convergence rates can be obtained. On the other hand, this operation may have a significant cost when the graph size is large. Therefore, when the communication costs are negligible compared with computation costs, Step 1 can be neglected.

IV. Convergence Rates of DS-ADMM: A Preliminary Result

In this section, we give the theoretical performance guarantees of First-order DS-ADMM in terms of convergence rates, because the proofs of the first-order results are the basis of the zeroth-order results. The expectations in the theorems and corollaries are taken with regard to all the random variables of the algorithm.

First we need some assumptions for our convergence results.

Assumption 1. *$A = D^{1/2}\mathcal{L}$, where $\mathcal{L}$ is the Laplacian of G' and $D = diag(d_1 + 1, ..., d_n + 1) \in \mathbb{R}^{n\times n}$. Meanwhile, $G_i(t)$ is defined by* (5) *such that $I_k \preceq G_i(t) \preceq \alpha I_k$ for some $\alpha > 1$.*

Remark 1. *$\mathcal{L} \in \mathbb{R}^{n\times n}$ is defined as $\mathcal{L}_{ij} = -1$ for $j \in N(i)\backslash\{i\}$, $\mathcal{L}_{ii} = d_i$ and zero otherwise.*

The communication matrix in Assumption 1 is used for the consensus of the network. Define $Q = \mathcal{L} \otimes I_k \in \mathbb{R}^{nk}$. As null($\mathcal{L}$)=span($\mathbf{1}$) for connected graphs [5], we have $Q\tilde{x} = 0 \Leftrightarrow x_1 = ... = x_n$, where $\tilde{x} = (x_1^T, ..., x_n^T)^T \in \mathbb{R}^{nk}$. Since D is a positive definite matrix, we have $(A \otimes I_k)\tilde{x} = 0 \Leftrightarrow x_1 = ... = x_n$. Meanwhile, due to the property of Q mentioned above, we can use $||Q\tilde{x}||_2^2$ to measure the node disagreement over the network.

Assumption 2. *The constraint set $\mathcal{X}$ is convex and compact, i.e., there exists some constant R such that $||x - x'||_2 < R$ for any $x, x' \in \mathcal{X}$*

Assumption 3. *$F_i(\cdot; \xi_i)$ is convex and β-Lipschitz continuous on $\mathcal{X}$ for any i and ξ_i, i.e., $|F_i(x; \xi_i) - F_i(x'; \xi_i)| \leq \beta||x - x'||_2$ for any $x, x' \in \mathcal{X}$.*

Assumption 4. *$F_i(\cdot; \xi_i)$ is σ-strongly convex ($\sigma > 0$) on $\mathcal{X}$ for any i and ξ_i, i.e., $F_i(x; \xi_i) \geq F_i(x'; \xi_i) + g_i^T(x - x') + \frac{\sigma}{2}||x - x'||_2^2$ for any $x, x' \in \mathcal{X}$ where $g_i \in \partial F_i(x'; \xi_i)$.*

Remark 2. *Assumption 4 can be generalized to the general convex case by allowing $\sigma = 0$. We will use this generalization in the proofs.*

Assumption 2, 3 and 4 are common assumptions in optimization literature [3]. For Assumption 2, if the original problem is unconstrained, we can first estimate the range of the solution and then add this range as a constraint to the problem.

Now we can give a theorem for the convergence rates of DS-ADMM using first-order information. We define $x^* = \arg\min_{x\in\mathcal{X}} \sum_{i=1}^n f_i(x)$ and $\bar{x}(T) = (\bar{x}_1(T)^T, ..., \bar{x}_n(T)^T)^T$

Theorem 1. *Under Assumption 1, 2 and 3, First-order DS-ADMM with $\eta_t = 1/\sqrt{t}$ can achieve $\mathbb{E}[\sum_{i=1}^n f_i(\bar{x}_i(T)) - \sum_{i=1}^n f_i(x^*) + ||Q\bar{x}(T)||_2^2] = O(n/\sqrt{T})$. Additionally, if Assumption 4 is satisfied, First-order DS-ADMM with $\eta_t = \alpha/(\sigma t)$ can achieve $\mathbb{E}[\sum_{i=1}^n f_i(\bar{x}_i(T)) - \sum_{i=1}^n f_i(x^*) + ||Q\bar{x}(T)||_2^2] = O(n \log T/T)$.*

Proof. See Appendix A. □

The above convergence metric, even though not direct, has been widely used in optimization literature such as [23], [29], [31]. Interestingly, if Step 1 of DS-ADMM is included, we can get a more direct convergence metric, which will be shown in Corollary 1.

Corollary 1. *Under Assumption 1, 2 and 3, First-order DS-ADMM **including Step 1** with $\eta_t = 1/\sqrt{t}$ can achieve $\mathbb{E}[\sum_{i=1}^n f_i(\bar{x}_j(T)) - \sum_{i=1}^n f_i(x^*)] = O(n/\sqrt{T})$ for any $j \in \{1, ..., n\}$. Additionally, if Assumption 4 is satisfied, First-order DS-ADMM **including Step 1** with $\eta_t = \alpha/(\sigma t)$ can achieve $\mathbb{E}[\sum_{i=1}^n f_i(\bar{x}_j(T)) - \sum_{i=1}^n f_i(x^*)] = O(n \log T/T)$ for any $j \in \{1, ..., n\}$.*

Proof. See Appendix B □

Compared with the best convergence rates of previous works mentioned in Section I, First-order DS-ADMM matches their order (respectively, up to a $\log T$ factor) in terms of T for general convex (respectively, strongly convex) functions. Therefore, First-order DS-ADMM itself is also a satisfying method. On the other hand, the advantages of DS-ADMM are highlighted in its zeroth-order version, which will be analyzed in the following section based on the above results.

V. Convergence Rates of DS-ADMM: Results for Two Gradient Estimators

In this section, we focus on DS-ADMM using zeroth-order information of stochastic realizations. We will discuss two gradient estimators applied for different function types.

A. Estimator for Lipschitz Continuous Gradients

For the estimator discussed in this part, the loss functions need to satisfy the following assumption.

Assumption 5. *$F_i(\cdot; \xi_i)$ has $L(\xi_i)$-Lipschitz gradients on $\mathcal{X}$ for any i and ξ_i, i.e., $||\nabla F_i(x; \xi_i) - \nabla F_i(x'; \xi_i)||_2 \leq L(\xi_i)||x - x'||_2$ for any x, $x' \in \mathcal{X}$. $L := \max_i\{\sqrt{\mathbb{E}[L(\xi_i)^2]}\}$ is finite.*

With this assumption, we can use the following deterministic estimator [2] for $\tilde{g}_i(t)$ in (4), which approximates each coordinate of the gradient and then sums them up:

$$\sum_{l=1}^{k} \frac{F_i\left(x_i(t) + u_t e_l; \xi_i(t)\right) - F_i\left(x_i(t) - u_t e_l; \xi_i(t)\right)}{2u_t} e_l \quad (6)$$

Here u_t is a scalar and e_l is a standard basis vector with 1 at its l^{th} coordinate. For this estimator, we have the following convergence rates.

Theorem 2. *Under Assumption 1, 2, 3 and 5, if using the deterministic estimator* (6)*, then DS-ADMM with* $\eta_t = 1/\sqrt{t}$ *and* $u_t = 1/(nkt)$ *can achieve* $\mathbb{E}[\sum_{i=1}^n f_i(\bar{x}_i(T)) - \sum_{i=1}^n f_i(x^*) + ||Q\bar{x}(t)||_2^2] = O(n/\sqrt{T})$*. Additionally, if Assumption 4 is satisfied, then DS-ADMM with* $\eta_t = \alpha/(\sigma t)$ *and* $u_t = 1/(nkt)$ *can achieve* $\mathbb{E}[\sum_{i=1}^n f_i(\bar{x}_i(T)) - \sum_{i=1}^n f_i(x^*) + ||Q\bar{x}(T)||_2^2] = O(n \log T/T)$.

Proof. See Appendix C □

Similarly, if Step 1 of DS-ADMM is run, we can have a more direct measure of convergence rates.

Corollary 2. *Under Assumption 1, 2, 3 and 5, if using the deterministic estimator* (6)*, then DS-ADMM* ***including Step 1*** *with* $\eta_t = 1/\sqrt{t}$ *and* $u_t = 1/(nkt)$ *can achieve* $\mathbb{E}[\sum_{i=1}^n f_i(\bar{x}_j(T)) - \sum_{i=1}^n f_i(x^*)] = O(n/\sqrt{T})$ *for any* $j \in \{1, ..., n\}$*. . Additionally, if Assumption 4 is satisfied, then DS-ADMM* ***including Step 1*** *with* $\eta_t = \alpha/(\sigma t)$ *and* $u_t = 1/(nkt)$ *can achieve* $\mathbb{E}[\sum_{i=1}^n f_i(\bar{x}_j(T)) - \sum_{i=1}^n f_i(x^*)] = O(n \log T/T)$ *for any* $j \in \{1, ..., n\}$.

Proof. Similar to Corollary 1 and omitted here. □

Compared with DS-ADMM, the method in [26] used the deterministic gradient (6) and achieved $O(1/T^{1/2})$ mean square convergence rate for functions with Lipschitz gradients and strong convexity, which is worse than our method. Meanwhile, the convergence rates in this case are in the same order with its first-order counterpart at the cost of making k queries to function values in each iteration.

B. Estimator for General Gradients

Now we consider the case where Assumption 5 is not necessarily satisfied. In this case, we apply the general gradient estimator used in [10] for (4):

$$\tilde{g}_i(t) = \frac{\Delta(x_i(t), \xi_i(t), \theta_i(t), z_i(t), u_{1t}, u_{2t})}{u_{2t}} z_i(t) \tag{7}$$

where

$$\Delta(\cdot) = F_i\left(x_i(t) + u_{1t}\theta_i(t) + u_{2t}z_i(t); \xi_i(t)\right) - F_i\left(x_i(t) + u_{1t}\theta_i(t); \xi_i(t)\right)$$

Here u_{1t} and u_{2t} are two scalars. $\theta_i(t) \in \mathbb{R}^k$ and $z_i(t) \in \mathbb{R}^k$ are two random variables sampled from two distributions μ_1 and μ_2, respectively. Here $\theta_i(t)$ is used to smoothen F_i by convolution:

$$F_i^u(x; \xi_i) = \mathbb{E}_\theta[F(x + u\theta; \xi_i)]$$

because convolution is a smoothing operation [9]. For this estimator, we need the following two assumptions to get the convergence rates:

Assumption 6. *dom* $F_i \supset \mathcal{X} + u_{11} \cdot$ *supp* $\mu_1 + u_{21} \cdot$ *supp* μ_2 *for any* i*, where supp* μ *is the support of distribution* μ*.*

Assumption 7. μ_1 *and* μ_2 *are one of the following pairs: (1) both* μ_1 *and* μ_2 *are standard normal with identity covariance; (2) both* μ_1 *and* μ_2 *are uniform on the surface of the Euclidean-ball of radius* $\sqrt{k+2}$

Now we can present the convergence rates of DS-ADMM with a general gradient estimator (7).

Theorem 3. *Under Assumption 1, 2, 3, 6 and 7, if using the gradient estimator* (7)*, then DS-ADMM with* $\eta_t = 1/\sqrt{k \log(2k) t}$*,* $u_{1t} = 1/t$ *and* $u_{2t} = 1/(k^2 n^2 t^2)$ *can achieve* $\mathbb{E}[\sum_{i=1}^n f_i(\bar{x}_i(T)) - \sum_{i=1}^n f_i(x^*) + ||Q\bar{x}(T)||_2^2] = O(n\sqrt{k \log(2k)/T})$*. Additionally, if Assumption 4 is satisfied, then DS-ADMM with* $\eta_t = \alpha/(\sigma t)$*,* $u_{1t} = 1/t$ *and* $u_{2t} = 1/(k^2 n^2 t^2)$ *can achieve* $\mathbb{E}[\sum_{i=1}^n f_i(\bar{x}_i(T)) - \sum_{i=1}^n f_i(x^*) + ||Q\bar{x}(T)||_2^2] = O(nk \log(2k) \log(T)/T)$.

Proof. See Appendix D. □

Still, if Step 1 is included, we have convergence rates of $\mathbb{E}[\sum_{i=1}^n f_i(\bar{x}_j(t)) - \sum_{i=1}^n f_i(x^*)]$ for any $j \in \{1, ..., n\}$.

Corollary 3. *Under Assumption 1, 2, 3, 6 and 7, if* ***including Step 1*** *and using the gradient estimator* (7)*, then DS-ADMM with* $\eta_t = 1/\sqrt{k \log(2k) t}$*,* $u_{1t} = 1/t$ *and* $u_{2t} = 1/(k^2 n^2 t^2)$ *can achieve* $\mathbb{E}[\sum_{i=1}^n f_i(\bar{x}_j(T)) - \sum_{i=1}^n f_i(x^*)] = O(n\sqrt{k \log(2k)/T})$ *for any* $j \in \{1, ..., n\}$*. . Additionally, if Assumption 4 is satisfied, then DS-ADMM* ***including Step 1*** *with* $\eta_t = \alpha/(\sigma t)$*,* $u_{1t} = 1/t$ *and* $u_{2t} = 1/(k^2 n^2 t^2)$ *can achieve* $\mathbb{E}[\sum_{i=1}^n f_i(\bar{x}_j(T)) - \sum_{i=1}^n f_i(x^*)] = O(nk \log(2k) \log(T)/T)$ *for any* $j \in \{1, ..., n\}$*. .*

Proof. Similar to Corollary 1 and omitted here. □

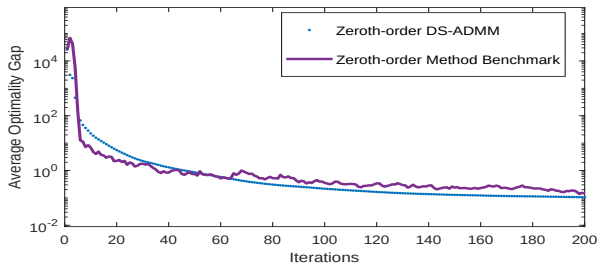
To the best of our knowledge, this is the first convergence result given to a distributed stochastic optimization method for general convex functions. The $O(n\sqrt{k \log(2k)/T})$ matches the optimal $O(\sqrt{k/T})$ bound [10] of zeroth-order methods up to a $\sqrt{\log 2k}$ factor.

VI. Numerical Results

In this section we use the two examples mentioned in Section II to show the performance of DS-ADMM using zeroth-order information. We assume that the communication costs are negligible, so Step 1 of DS-ADMM is excluded. The simulations are applied to one network and the network topology is a connected Erdős-Renyi graph $ER(100, 0.2)$, meaning that 100 nodes connect with each other with probability 0.2.

A. Distributed Online Least Square

Recall that in the setting of distributed online least square (OLS), each sensor in the network aims to estimate a signal $\hat{x}$ and receives a loss function $F_i(x; \xi_i(t)) = ||H_i^T x - \xi_i(t)||_2^2$ at time t, where $\xi_i(t) = H_i^T \hat{x} + w_i(t)$ and $\{w_i(t)\}_t$ are i.i.d Gaussian noise. We want to minimize $\sum_{i=1}^n \mathbb{E}_{\xi_i \sim \mathcal{D}_i} ||H_i^T x - \xi_i||_2^2$ and it is easy to know that $\mathcal{D}_i$ is a Gaussian distribution with unknown mean. In this experiment, we assume $H_i \in \mathbb{R}^{10 \times 10}$ is different for each sensor. Meanwhile, $H_i H_i^T$ is positive definite so that F_i is strongly convex. $w_i(t) \sim \mathcal{N}(0, \text{Var}_i \cdot I_{10})$, and



(a) Comparison of average optimality gaps

(b) Comparison of node disagreements

Fig. 1: Experiments for distributed OLS

$\text{Var}_i \in \mathbb{R}$ is also different for each sensor. The true signal is $\hat{x} = \mathbf{1} \in \mathbb{R}^{10}$ and unknown to sensors. The constraint set is a box constraint where each coordinate of x_i is between -10 and 10.

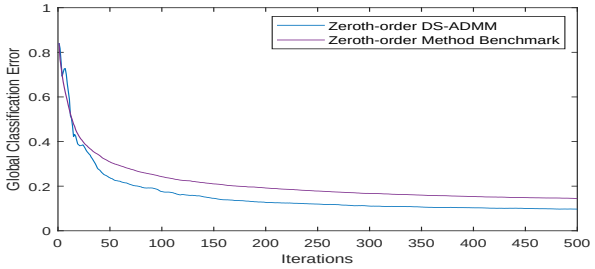
In Figure 1a, we compare average optimality gaps of DS-ADMM and the benchmark method in [26]. The average optimality gap is defined as: $\frac{1}{n}\sum_{j=1}^{n}[\sum_{i=1}^{n} f_i(\bar{x}_j(t)) - \sum_{i=1}^{n} f_i(x^*)]$, where we also use $\bar{x}_j(t) \in \mathbb{R}^k$ to represent the output of the benchmark method. The initial $x_i(1)$ are set to $\mathbf{0}$ for the two methods and other parameters are chosen to achieve their good performance. From the figure, we can see that DS-ADMM has a faster convergence rate than the zeroth-order method in [26] after 60 iterations. Meanwhile, Figure 1b shows the comparison of $||Q\bar{x}(t)||_2^2$ of the two methods. As mentioned in Section IV, $||Q\bar{x}(t)||_2^2$ can measure the node disagreement of the algorithm. So Figure 1b actually shows the consensus rates of the methods. We can see that DS-ADMM has a better consensus rate than the method in [26]. So DS-ADMM is the best option as a zeroth-order method in this experiment.

B. Distributed Support Vector Machine

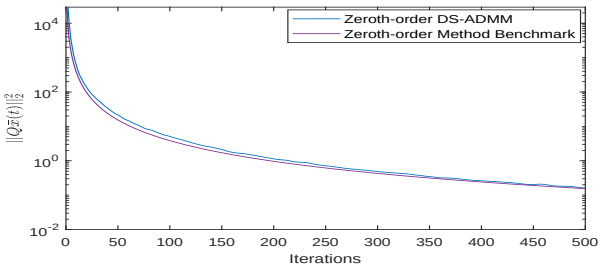
For Distributed Support Vector Machine (SVM), F_i is in the form of (2), where $(\gamma_i(s), \varphi_i(s))$ is the s^{th} data point in Server i. In this experiment, each server has 100 data points and randomly choose one data point to use at each iteration of the optimization process to reduce computation cost. The objective is to minimize the training loss $\frac{1}{10000}\sum_{i=1}^{100}\sum_{s=1}^{100} F_i(x_i; \gamma_i(s), \varphi_i(s))$. Obviously F_i is a function without strong convexity and Lipschitz gradients. So the gradient estimator (7) is used for DS-ADMM. In this experiment, we assume $\gamma_i(s) \in \mathbb{R}^8$ and $\varphi_i(s) \in \{1, 2, 3, 4\}$.

First Element of $\check{\gamma}_i(s)$	+	+	-	-
Fourth Element of $\check{\gamma}_i(s)$	+	-	+	-
Class $\varphi_i(s)$	1	2	3	4

TABLE II: Relation between $\check{\gamma}_i(s)$ and $\varphi_i(s)$. Other elements of $\check{\gamma}_i(s)$ have no impact on $\varphi_i(s)$



(a) Comparison of classification errors



(b) Comparison of node disagreements

Fig. 2: Experiments for distributed SVM

Meanwhile, $\gamma_i(s) = \check{\gamma}_i(s) + w_i(s)$ where $w_i(t)$ is the noise modeled as $\mathcal{N}(0, I_8)$ for all i. The class $\varphi_i(t)$ is related to $\check{\gamma}_i(t)$ in the way shown as Table II, where $+$ means nonnegative and $-$ means negative. Still, the constraint set is a box constraint where each coordinate of x_i is between -10 and 10. Since there is no previous work on distributed stochastic optimization using zeroth-order information for general convex functions, we replace the stochastic gradient used in the first-order method of [28] with the estimator (7) and regard this modified algorithm as the benchmark. Note that the method in [28] has the optimal convergence rate for general convex functions among existing first-order methods.

In Figure 2a, we compare the global classification error of DS-ADMM and the benchmark given by the output of one node in each iteration. The node is randomly chosen from the total nodes before the start of the experiment and kept tracked afterwards for both methods. The global classification error is defined as the classification error when we test the whole training set consisting of 10000 (100×100) data points using the output parameter. In Figure 2a, we can see that DS-ADMM has a faster convergence rate and converges to a lower classification error than the benchmark. In Figure 2b, we still compare the consensus rates of the two methods like Figure 1b. We can see that their consensus rates are close to each other. Again, DS-ADMM has a better performance.

VII. Conclusion

This work is motivated by the need to develop fast converging zeroth-order (a.k.a., non-derivative or derivative-free) methods for large-scale machine learning problems in distributed processing networks. We tackled this challenge by developing Distributed Stochastic Alternating Direction Method of Multipliers (DS-ADMM) that extends a recently proposed Distributed ADMM method to the zeroth-order design through a sequence of nontrivial modifications both to the design and the analysis. We achieved this by first proposing a novel first-order DS-ADMM method that not only yields desirable convergence rate characteristics, but is also amenable to zeroth-order implementation. Then, we investigated the zeroth-order version of the DS-ADMM algorithm to derive its convergence rate for convex, strongly convex, and Lipschitz-gradient functions. In all case, we showed that our zeroth-order design has the fastest convergence rate guarantee of all prior works. We also demonstrated these gains in numerical studies for two machine learning application, related to an estimation and a classification problem.

Appendix

A. Proof of Theorem 1

Define $x(t) = (x_1(t)^T, ..., x_n(t)^T)^T$, $y(t) = (y_1(t)^T, ..., y_n(t)^T)^T$, $p(t) = (p_1(t)^T, ..., p_n(t)^T)^T$, $B = D^{-1} \otimes I_k$, $P = A \otimes I_k$.

Under Assumption 1, we can write Step 4 of Algorithm 1 as

$$y(t) = BPx(t) \tag{8}$$

because $A_{ij} = 0$ when $(i,j) \notin E'$ and $i \neq j$. Since $p(0) = \mathbf{0}$, we have

$$p(t) = c\sum_{s=1}^{t} y(s) = cBP\sum_{s=1}^{t} x(s) \tag{9}$$

by the iteration of Step 5.

Meanwhile, since $\mathcal{X}$ is convex, the optimality condition [3] for Step 6 is

$$\langle \sum_{j\in N(i)} (A_{ji}p_j(t) + cA_{ji}y_j(t) + cA_{ji}^2(x_i(t+1) - x_i(t))) + g_i(t) + \frac{G_i(t)(x_i(t+1) - x_i(t))}{\eta_t}, \tilde{x}_i - x_i(t+1)\rangle \geq 0 \tag{10}$$

for any $\tilde{x}_i \in \mathcal{X}$, $\forall i \in \{1, ..., n\}$.

Now define $M = \text{diag}(\sum_{j\in N(1)} A_{j1}^2 I_k, ..., \sum_{j\in N(n)} A_{jn}^2 I_k)$, $G(t) = \text{diag}(G_1(t), ..., G_n(t))$ and $F(x(t);\xi(t)) = \sum_{i=1}^n F_i(x_i(t);\xi_i(t)) : \mathbb{R}^{nk} \to \mathbb{R}$ and $g(t) \in \partial F(x(t);\xi(t))$. Based on (8) and (9), we can write (10) in a compact form:

$$\langle g(t) + cP^TBP\sum_{s=1}^{t} x(s) + cM(x(t+1) - x(t)) + cP^TBPx(t) + \frac{G(t)}{\eta_t}(x(t+1) - x(t)), \tilde{x} - x(t+1)\rangle \geq 0 \tag{11}$$

where $\tilde{x} := (\tilde{x}_1^T, ..., \tilde{x}_n^T)^T$.

Using a similar proof to Lemma 9 in [20], we can prove that P^TBP is positive semidefinite. Define $Q = (P^TBP)^{\frac{1}{2}} = \mathcal{L} \otimes I_k$, $r(t) = \sum_{s=1}^t Qx(s)$ and $\tilde{x}^* = ((x^*)^T, ..., (x^*)^T)^T \in \mathbb{R}^{nk}$. Since null($\mathcal{L}$)=span($\mathbf{1}$) for connected graphs [5], $Q\tilde{x}^* = \mathbf{0}$. Now suppose that Assumption 4 is satisfied with $\sigma \geq 0$. Note that F_i is just convex if $\sigma = 0$. For any $r \in \mathbb{R}^{nk}$, we have

$$\begin{aligned}
&F(x(t);\xi(t)) - F(\tilde{x}^*;\xi(t)) + cr^TQx(t+1) \\
&\leq g(t)^T(x(t) - \tilde{x}^*) + cr^TQx(t+1) - \frac{\sigma}{2}||x(t) - \tilde{x}^*||_2^2 \qquad (12)\\
&= g(t)^T(x(t+1) - \tilde{x}^*) + cr^TQx(t+1) \\
&\quad + g(t)^T(x(t) - x(t+1)) - \frac{\sigma}{2}||x(t) - \tilde{x}^*||_2^2 \\
&\leq \langle cP^TBP\sum_{s=1}^{t} x(s) + (cM + \frac{G(t)}{\eta_t})(x(t+1) - x(t)) \\
&\quad + cP^TBPx(t), \tilde{x}^* - x(t+1)\rangle + g(t)^T(x(t) - x(t+1)) \\
&\quad + cr^TQx(t+1) - \frac{\sigma}{2}||x(t) - \tilde{x}^*||_2^2 \qquad (13)\\
&= c(\tilde{x}^* - x(t+1))^TQ^Tr(t+1) + cr^TQx(t+1) \\
&\quad + (\tilde{x}^* - x(t+1))^T(cM + \frac{G(t)}{\eta_t} - cP^TBP)(x(t+1) - x(t)) \\
&\quad + g(t)^T(x(t) - x(t+1)) - \frac{\sigma}{2}||x(t) - \tilde{x}^*||_2^2 \qquad (14)\\
&= c(r(t+1) - r(t))^T(-r(t+1) + r) \\
&\quad + (\tilde{x}^* - x(t+1))^T(cM + \frac{G(t)}{\eta_t} - cP^TBP)(x(t+1) - x(t)) \\
&\quad + g(t)^T(x(t) - x(t+1)) - \frac{\sigma}{2}||x(t) - \tilde{x}^*||_2^2 \qquad (15)
\end{aligned}$$

where (12) is from Assumption 4 (including the case when $\sigma = 0$), (13) is from (11), (14) is from $cP^TBP(\sum_{s=1}^t x(s) + x(t)) = cQ^Tr(t+1) - cP^TBP(x(t+1) - x(t))$ and (15) is from $r^TQx(t+1) = (Qx(t+1))^Tr$, $Qx(t+1) = r(t+1) - r(t)$ and $Q\tilde{x}^* = \mathbf{0}$

Define $\Lambda(t) = cM + \frac{G(t)}{\eta_t} - cP^TBP$. From Lemma 10 of [29], we have

$$\begin{aligned}
&(\tilde{x}^* - x(t+1))^T(cM + \frac{G(t)}{\eta_t} - cP^TBP)(x(t+1) - x(t)) \\
&= \frac{1}{2}||x(t) - \tilde{x}^*||_{\Lambda(t)}^2 - \frac{1}{2}||x(t+1) - \tilde{x}^*||_{\Lambda(t)}^2 \\
&\quad - \frac{1}{2}||x(t+1) - x(t)||_{\Lambda(t)}^2 \\
&\leq \frac{1}{2}||x(t) - \tilde{x}^*||_{\Lambda(t)}^2 - \frac{1}{2}||x(t+1) - \tilde{x}^*||_{\Lambda(t)}^2 \\
&\quad - \frac{1}{2}||x(t+1) - x(t)||_{\frac{G(t)}{\eta_t}}^2 \qquad (16)
\end{aligned}$$

$$\begin{aligned}
&c(r(t+1) - r(t))^T(-r(t+1) + r) \\
&= \frac{c}{2}||r(t) - r||_2^2 - \frac{c}{2}||r(t+1) - r||_2^2 - \frac{c}{2}||r(t+1) - r(t)||_2^2 \\
&\leq \frac{c}{2}||r(t) - r||_2^2 - \frac{c}{2}||r(t+1) - r||_2^2
\end{aligned}$$

$$
\begin{aligned}
&g(t)^T(x(t)-x(t+1))\\
&\le ||g(t)||_{G(t)^{-1}}||x(t)-x(t+1)||_{G(t)} \qquad (17)\\
&\le \frac{1}{2\eta_t}||x(t+1)-x(t)||^2_{G(t)}+\frac{\eta_t}{2}||g(t)||^2_{G(t)^{-1}} \qquad (18)
\end{aligned}
$$

where (16) is because $M-P^TBP$ is positive semidefinite by a similar proof to Lemma 9 in [20], (17) is from Cauchy–Schwarz inequality and (18) is from $2ab \le a^2+b^2$.

Apply the above three inequalities to (15), telescope from $t=1$ to T and take expectations to all random variables. Then we have

$$
\begin{aligned}
&\sum_{t=1}^{T}\{\mathbb{E}[F(x(t);\xi(t))-F(\tilde{x}^*;\xi(t))]+\mathbb{E}[cr^TQx(t+1)]\}\\
&\le \frac{c}{2}||r(1)-r||_2^2+\frac{1}{2}||x(1)-\tilde{x}^*||^2_{\Lambda(1)}\\
&\quad+\sum_{t=2}^{T}(\frac{1}{2\eta_t}\mathbb{E}||x(t)-\tilde{x}^*||^2_{G(t)}-\frac{1}{2\eta_{t-1}}\mathbb{E}||x(t)-\tilde{x}^*||^2_{G(t-1)}\\
&\quad-\frac{\sigma}{2}\mathbb{E}||x(t)-\tilde{x}^*||_2^2)+\sum_{t=1}^{T}\frac{\eta_t}{2}\mathbb{E}||g(t)||^2_{G(t)^{-1}}
\end{aligned}
$$

By the form of $G_i(t)$ used in Assumption 1, we have

$$
\begin{aligned}
&\frac{1}{2\eta_t}\mathbb{E}||x(t)-\tilde{x}^*||^2_{G(t)}-\frac{1}{2\eta_{t-1}}\mathbb{E}||x(t)-\tilde{x}^*||^2_{G(t-1)}\\
&=(\frac{\alpha}{2\eta_t}-\frac{\alpha}{2\eta_{t-1}})\mathbb{E}||x(t)-\tilde{x}^*||_2
\end{aligned}
$$

So by Assumption 1, 2 and 3

$$
\begin{aligned}
&\sum_{t=1}^{T}\{\mathbb{E}[F(x(t);\xi(t))-F(\tilde{x}^*;\xi(t))]+\mathbb{E}[cr^TQx(t)]\}\\
&\le \frac{c}{2}||r(1)-r||_2^2+\frac{1}{2}||x(1)-\tilde{x}^*||^2_{\Lambda(1)}\\
&\quad+\sum_{t=2}^{T}\max(\frac{n\alpha}{2\eta_t}-\frac{n\alpha}{2\eta_{t-1}}-\frac{n\sigma}{2},0)R^2+\sum_{t=1}^{T}\frac{\eta_t}{2}n\beta^2\\
&\quad+2c||r^TQ||_2^2R' \qquad (19)
\end{aligned}
$$

where $R' < \infty$ is the bound of $||x(T+1)||_2$ since $\mathcal{X}$ is bounded.

Meanwhile, for any $j \in \{1,...,n\}$ we have

$$
\begin{aligned}
&\mathbb{E}_{\{\xi(1),...,\xi(T)\}}[\sum_{i=1}^{n}f_i(\bar{x}_i(T))-\sum_{i=1}^{n}f_i(x^*)+||Q\bar{x}(T)||^2]\\
&\le \mathbb{E}_{\{\xi(1),...,\xi(T)\}}[\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}_{\xi_i}[\sum_{i=1}^{n}F_i(x_i(t);\xi_i)-\sum_{i=1}^{n}F_i(x^*;\xi_i)]\\
&\quad+||Q\bar{x}(T)||^2] \qquad (20)\\
&=\mathbb{E}[\frac{1}{T}\sum_{t=1}^{T}[F(x(t);\xi(t))-F(\tilde{x}^*;\xi(t))]+\mathbb{E}[(Q\bar{x}(T))^TQ\bar{x}(T)]
\end{aligned}
$$

where (20) is from the convexity of F_i. Setting $r=\frac{Q\bar{x}(T)}{c}$ in (19), we can see that $||r||_2^2$ is bounded since $\mathcal{X}$ is bounded. Define

$$
C=c||r^TQ||_2^2R'+\frac{c}{2}||r(1)-r||_2^2+\frac{1}{2}||x(1)-\tilde{x}^*||^2_{\Lambda(1)}
$$

when $r=\frac{Q\bar{x}(t)}{c}$. Then

$$
\begin{aligned}
&\mathbb{E}_{\{\xi(1),...,\xi(T)\}}[\sum_{i=1}^{n}f_i(\bar{x}_i(T))-\sum_{i=1}^{n}f_i(x^*)+||Q\bar{x}(T)||_2^2]\\
&\le \frac{C}{T}+\frac{1}{T}\sum_{t=2}^{T}\max(\frac{n\alpha}{2\eta_t}-\frac{n\alpha}{2\eta_{t-1}}-\frac{n\sigma}{2},0)R^2+\sum_{t=1}^{T}\frac{\eta_t}{2T}n\beta^2 \qquad (21)
\end{aligned}
$$

Meanwhile, $\sum_{t=1}^{T}\frac{1}{\sqrt{t}}=O(\sqrt{T})$ and $\sum_{t=1}^{T}\frac{1}{T}=O(\log T)$. Therefore, when $\sigma \ge 0$ (first case of Theorem 1), we can get the convergence rate of $O(n/\sqrt{T})$ if we choose $\eta_t=1/\sqrt{t}$; When $\sigma > 0$ (second case of Theorem 1), we can get the $O(n\log(T)/T)$ rate if we choose $\eta_t=\alpha/(\sigma t)$.

B. Proof of Corollary 1

From graph theory [5], $Q=\mathcal{L}\otimes I_k=(W\otimes I_k)(W\otimes I_k)^T$, where $W\in\mathbb{R}^{n\times m}$ is the oriented incidence matrix of G' and m is the number of edges in G'. When G' is a tree, W has linearly independent columns [1] and then has a Moore-Penrose inverse C^+ such that $W^+W=I_{n-1}$ [15]. So $(W\otimes I_k)^T=W^+\otimes I_kQ$.

By the definition of oriented incidence matrix and triangle inequality, we have

$$
\begin{aligned}
\max_i||\bar{x}_i(T)-\bar{x}_j(T)||_2 &\le (n-1)||(W\otimes I_k)^T\bar{x}(T)||_2\\
&\le (n-1)||W^+\otimes I_k||_2||Q\bar{x}(T)||_2
\end{aligned}
$$

for any j. Meanwhile,

$$
\begin{aligned}
&\mathbb{E}[\sum_{i}^{n}f_i(\bar{x}_j(T))-\sum_{i}^{n}f_i(x^*)]\\
&=\mathbb{E}[\sum_{i}^{n}f_i(\bar{x}_i(T))-\sum_{i}^{n}f_i(x^*)]+\mathbb{E}[\sum_{i}^{n}f_i(\bar{x}_j(T))-\sum_{i}^{n}f_i(\bar{x}_i(T))]\\
&\le \frac{1}{T}\sum_{t=1}^{T}\{\mathbb{E}[F(x(t);\xi(t))-F(\tilde{x}^*;\xi(t))]+\mathbb{E}[cr^TQx(t)]\}\\
&\quad+n\beta\mathbb{E}[\max_i||\bar{x}_i(T)-\bar{x}_j(T)||_2]-\mathbb{E}[cr^TQ\bar{x}(T)] \qquad (22)
\end{aligned}
$$

where (22) is from Assumption 3. Now

$$
\begin{aligned}
&n\beta\mathbb{E}[\max_i||\bar{x}_i(T)-\bar{x}_j(T)||_2]-\mathbb{E}[cr^TQ\bar{x}(T)]\\
&\le \mathbb{E}[(n\beta(n-1)||W^+\otimes I_k||_2\frac{Q\bar{x}(T)}{||Q\bar{x}(T)||_2}-cr)^TQ\bar{x}(T)]
\end{aligned}
$$

Letting $cr=n\beta(n-1)||W^+\otimes I_k||_2\frac{Q\bar{x}(T)}{||Q\bar{x}(T)||_2}$, we have

$$
c||r||_2 \le n\beta(n-1)||W^+\otimes I_k||_2 \qquad (23)
$$

Now we have

$$
\mathbb{E}[n\beta\max_i||\bar{x}_i(t)-\bar{x}_j(t)||_2-cr^TQ\bar{x}(T)]\le 0 \qquad (24)
$$

Combine (19) with (24) for (22) using the above r and define C' as the terms not related to η_t in these two inequalities. $C'<\infty$ because of (23). Now we have

$$
\begin{aligned}
&\mathbb{E}[\sum_{i}^{n}f_i(\bar{x}_j(T))-\sum_{i}^{n}f_i(x^*)]\\
&\le \frac{C'}{T}+\frac{1}{T}\sum_{t=2}^{T}\max(\frac{n\alpha}{2\eta_t}-\frac{n\alpha}{2\eta_{t-1}}-\frac{n\sigma}{2},0)R^2+\frac{1}{T}\sum_{t=1}^{T}\frac{\eta_t}{2}n\beta^2
\end{aligned}
$$

Similar to Proof of Theorem 1, we can get the convergence rates in Corollary 1 with appropriate stepsizes.

C. Proof of Theorem 2

Define $\tilde{g}(t) = (\tilde{g}_1(t)^T, ..., \tilde{g}_n(t)^T)^T$, $e(t) = g(t) - \tilde{g}(t)$. We have

$$\begin{aligned}
&F(x(t);\xi(t)) - F(\tilde{x}^*;\xi(t)) + cr^TQx(t+1) \\
&\leq \tilde{g}(t)^T(x(t) - \tilde{x}^*) + cr^TQx(t+1) - \frac{\sigma}{2}||x(t) - \tilde{x}^*||_2^2 \\
&\quad + e(t)^T(x(t) - \tilde{x}^*)
\end{aligned}$$

Compared with (12), the above inequality changes $g(t)$ to $\tilde{g}(t)$ and has one more term $e(t)^T(x(t) - \tilde{x}^*)$. Meanwhile, since Zeroth-order DS-ADMM run Step 9 instead of Step 7, (11) is changed to

$$\begin{aligned}
&\langle \tilde{g}(t) + cP^TBP\sum_{s=0}^{t} x(s) \\
&+ c(P^TBPx(t) + M(x(t+1) - x(t))) \\
&\quad + \frac{G(t)}{\eta_t}(x(t+1) - x(t)), \tilde{x} - x(t+1)\rangle \geq 0
\end{aligned} \tag{25}$$

where $\tilde{g}(t)$ replaces $g(t)$. Following the similar proof in Appendix A and taking expectations to both sides with regard to all random variables in the algorithm, we have

$$\begin{aligned}
&\mathbb{E}[\sum_{i=1}^{n} f_i(\bar{x}_i(T)) - \sum_{i=1}^{n} f_i(x^*) + ||Q\bar{x}(T)||^2] \\
&\leq \frac{C''}{T} + \frac{1}{T}\sum_{t=2}^{T}\max(\frac{n\alpha}{2\eta_t} - \frac{n\alpha}{2\eta_{t-1}} - \frac{n\sigma}{2}, 0)R^2 \\
&\quad + \frac{1}{T}\sum_{t=1}^{T}\frac{\eta_t}{2}\mathbb{E}||\tilde{g}(t)||^2_{G_t^{-1}} + \frac{1}{T}\mathbb{E}[\sum_{t=1}^{T} e(t)^T(x(t) - \tilde{x}^*)]
\end{aligned} \tag{26}$$

where C'' represents terms not related to η_t. Here we need a lemma to bound the moments of the estimator:

Lemma 1. *(Lemma 3 of [19]) Under Assumption 2, 3, 5, 6 and 7, the deterministic estimator satisfies*

$$\mathbb{E}[||\tilde{g}_i(t) - g_i(t)||_2^2] \leq \frac{L^2k^2u_t^2}{4}$$

From Lemma 1, we have

$$\begin{aligned}
\mathbb{E}[||\tilde{g}(t)||_2^2] &= \sum_{i=1}^{n}\mathbb{E}[||\tilde{g}_i(t)||_2^2] \\
&\leq \sum_{i=1}^{n}[2\mathbb{E}[||\tilde{g}_i(t) - g_i(t)||_2^2] + 2\mathbb{E}[||g_i(t)||_2^2]] \\
&\leq \frac{nL^2k^2u_t^2}{2} + 2n\beta^2
\end{aligned}$$

$$\mathbb{E}[e(t)^T(x(t) - \tilde{x}^*)] \leq ||e(t)||_2 \cdot ||x(t) - \tilde{x}^*||_2 \quad \leq \frac{n\sqrt{n}Lku_tR}{2}$$

where $e(t) = g(t) - \tilde{g}(t)$. Apply the above inequalities to (26) and then we can get the final result by choosing corresponding parameters.

D. Proof of Theorem 3

First we need the following lemma for the gradient estimator:

Lemma 2. *(Lemma 2 of [10]) Under Assumption 1, 2, 3, 8 and 9, the general gradient estimator satisfies*

$$\begin{aligned}
\mathbb{E}[\tilde{g}_i(t)] &= g_i^{u_{1t}}(t) + \frac{u_{2t}}{u_{1t}}\beta v_i(t) \\
\mathbb{E}[||\tilde{g}_i(t)||_2^2] &\leq b\beta^2k(\sqrt{\frac{u_{2t}}{u_{1t}}}k + 1 + \log k)
\end{aligned}$$

where $g_i^{u_{1t}}(t) \in \partial F_i^{u_{1t}}(x_i(t);\xi_i(t))$, $F_i^{u_{1t}}(x_i(t);\xi_i(t)) := \mathbb{E}_{\theta_i(t)}[F(x_i(t) + u_{1t}\theta_i(t);\xi_i(t))]$ *is the smoothened function by convolution operation,* $v_i(t)$ *is a term with* $||v_i(t)||_2 \leq \frac{1}{2}\mathbb{E}[||z_i(t)||_2^3]$*, and* b *is a constant.*

Meanwhile, from Lemma E.2, E.3 of [9] and $\sqrt{k+2} \leq \sqrt{3k}$, we have $F_i(x_i(t);\xi_i(t)) \leq F_i^{u_{1t}}(x_i(t);\xi_i(t)) \leq F_i(x_i(t);\xi_i(t)) + u_{1t}\beta\sqrt{3k}$. Now defining $F^{u_{1t}}(x(t);\xi(t)) = \sum_{i=1}^{n} F_i^{u_{1t}}(x_i(t);\xi_i(t))$ and $e'(T) = g^{u_{1t}}(t) - \tilde{g}(t)$, we have

$$\begin{aligned}
&F(x(t);\xi(t)) - F(\tilde{x}^*;\xi(t)) + cr^TQx(t+1) \\
&\leq F^{u_{1t}}(x(t);\xi(t)) - F^{u_{1t}}(\tilde{x}^*;\xi(t)) + nu_{1t}\beta\sqrt{3k} \\
&+ cr^TQx(t+1) \\
&\leq g^{u_{1t}}(t)^T(x(t) - \tilde{x}^*) + cr^TQx(t+1) - \frac{\sigma}{2}||x(t) - \tilde{x}^*||_2^2 \\
&\quad + nu_{1t}\beta\sqrt{3k} \\
&\leq \tilde{g}(t)^T(x(t) - \tilde{x}^*) + cr^TQx(t+1) - \frac{\sigma}{2}||x(t) - \tilde{x}^*||_2^2 \\
&\quad + e'(t)^T(x(t) - \tilde{x}^*) + nu_{1t}\beta\sqrt{3k}
\end{aligned}$$

Based on Lemma 2 and following a similar proof to Appendix C, we have

$$\begin{aligned}
&\mathbb{E}[\sum_{i=1}^{n} f_i(\bar{x}_i(T)) - \sum_{i=1}^{n} f_i(x^*) + ||Q\bar{x}(T)||_2^2] \\
&\leq \frac{C'''}{T} + \frac{1}{T}\sum_{t=2}^{T}\max(\frac{\alpha}{2\eta_t} - \frac{\alpha}{2\eta_{t-1}} - \frac{\sigma}{2}, 0)nR^2 \\
&+ \frac{1}{T}\sum_{t=1}^{T}\frac{\eta_t}{2}\left(b\rho\beta^2kn(\sqrt{\frac{u_{2t}}{u_{1t}}}k + \log(2k))\right) \\
&+ \frac{\sqrt{3k}kn\sqrt{n}}{2T}R\beta\sum_{t=1}^{T}\frac{u_{2t}}{u_{1t}} + \frac{1}{T}\sum_{t=1}^{T}\sqrt{3}n\beta\sqrt{k}u_{1t}
\end{aligned}$$

where C''' is the terms not related to η_t. For any $\sigma \geq 0$, we can get the $O(n\sqrt{k\log(2k)}/\sqrt{T})$ rate if we choose $\eta_t = 1/\sqrt{k\log(2k)t}$ and $u_{1t} = 1/t, u_{2t} = 1/(k^2n^2t^2)$. When $\sigma > 0$, then we can get $O(nk\log(2k)\log(T)/T)$ if we choose $\eta_t = \alpha/(\sigma t)$ and $u_{1t} = 1/t, u_{2t} = 1/(k^2n^2t^2)$.

REFERENCES

[1] Graph matrices. http://compalg.inf.elte.hu/~tony/Oktatas/TDK/FINAL/Chap%2010.PDF. Accessed: 2018-12-12.

[2] A. Agarwal, O. Dekel, and L. Xiao. Optimal algorithms for online convex optimization with multi-point bandit feedback. In *COLT*, pages 28–40. Citeseer, 2010.

[3] D. P. Bertsekas. *Nonlinear programming*. Athena scientific Belmont, 1999.

[4] L. Bottou, F. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018.

[5] F. R. Chung and F. C. Graham. *Spectral graph theory*. Number 92. American Mathematical Soc., 1997.

[6] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of machine learning research*, 2(Dec):265–292, 2001.

[7] J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

[8] J. C. Duchi, A. Agarwal, and M. J. Wainwright. Dual averaging for distributed optimization: Convergence analysis and network scaling. *IEEE Transactions on Automatic control*, 57(3):592–606, 2012.

[9] J. C. Duchi, P. L. Bartlett, and M. J. Wainwright. Randomized smoothing for stochastic optimization. *SIAM Journal on Optimization*, 22(2):674–701, 2012.

[10] J. C. Duchi, M. I. Jordan, M. J. Wainwright, and A. Wibisono. Optimal rates for zero-order convex optimization: The power of two function evaluations. *IEEE Transactions on Information Theory*, 61(5):2788–2806, 2015.

[11] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*, volume 1. 2016.

[12] A. Gosavi et al. Simulation-based optimization. *parametric optimization techniques and reinforcement learning*, 2003.

[13] D. Hajinezhad, M. Hong, and A. Garcia. Zeroth order nonconvex multi-agent optimization over networks. *arXiv preprint arXiv:1710.09997*, 2017.

[14] E. Hazan et al. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

[15] R. A. Horn, R. A. Horn, and C. R. Johnson. *Matrix analysis*. Cambridge university press, 1990.

[16] D. Jakovetic, D. Bajovic, A. K. Sahu, and S. Kar. Convergence rates for distributed stochastic optimization over random networks. *arXiv preprint arXiv:1803.07836*, 2018.

[17] G. Lan, S. Lee, and Y. Zhou. Communication-efficient algorithms for decentralized and stochastic optimization. *arXiv preprint arXiv:1701.03961*, 2017.

[18] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pages 5330–5340, 2017.

[19] S. Liu, B. Kailkhura, P.-Y. Chen, P. Ting, S. Chang, and L. Amini. Zeroth-order stochastic variance reduction for nonconvex optimization, 2018.

[20] A. Makhdoumi and A. Ozdaglar. Convergence rate of distributed admm over networks. *IEEE Transactions on Automatic Control*, 62(10):5082–5095, 2017.

[21] A. Nedić and A. Olshevsky. Stochastic gradient-push for strongly convex functions on time-varying directed graphs. *IEEE Transactions on Automatic Control*, 61(12):3936–3947, 2016.

[22] A. Nedić, A. Olshevsky, and M. G. Rabbat. Network topology and communication-computation tradeoffs in decentralized optimization. *Proceedings of the IEEE*, 106(5):953–976, 2018.

[23] H. Ouyang, N. He, L. Tran, and A. Gray. Stochastic alternating direction method of multipliers. In *International Conference on Machine Learning*, pages 80–88, 2013.

[24] S. Pu and A. Nedic. A distributed stochastic gradient tracking method. 2018.

[25] G. Qu and N. Li. Accelerated distributed nesterov gradient descent. *arXiv preprint arXiv:1705.07176*, 2017.

[26] A. K. Sahu, D. Jakovetic, D. Bajovic, and S. Kar. Distributed zeroth order optimization over random networks: A kiefer-wolfowitz stochastic approximation approach. *arXiv preprint arXiv:1803.07844*, 2018.

[27] M. O. Sayin, N. D. Vanli, S. S. Kozat, and T. Ba?ar. Stochastic subgradient algorithms for strongly convex optimization over distributed networks. *IEEE Transactions on Network Science and Engineering*, 4(4):248–260, Oct 2017.

[28] B. Sirb and X. Ye. Decentralized consensus algorithm with delayed and stochastic gradients. *SIAM Journal on Optimization*, 28(2):1232–1254, 2018.

[29] T. Suzuki. Dual averaging and proximal gradient descent for online alternating direction multiplier method. In *International Conference on Machine Learning*, pages 392–400, 2013.

[30] J. Tsitsiklis, D. Bertsekas, and M. Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE transactions on automatic control*, 31(9):803–812, 1986.

[31] W. Zhong and J. Kwok. Fast stochastic alternating direction method of multipliers. In *International Conference on Machine Learning*, pages 46–54, 2014.