GRAPH TOPOLOGY INVARIANT GRADIENT AND SAMPLING COMPLEXITY FOR DECENTRALIZED AND STOCHASTIC OPTIMIZATION*

GUANGHUI LAN†, YUYUAN OUYANG‡, AND YI ZHOU§

Abstract. One fundamental problem in decentralized multi-agent optimization is the trade-off between gradient/sampling complexity and communication complexity. In this paper we propose new algorithms whose gradient and sampling complexities are graph topology invariant, while their communication complexities remain optimal. Specifically, for convex smooth deterministic problems, we propose a primal dual sliding (PDS) algorithm that is able to compute an ε-solution with $\mathcal{O}((\tilde{L}/\varepsilon)^{1/2})$ gradient complexity and $\mathcal{O}((\tilde{L}/\varepsilon)^{1/2}+\|\mathcal{A}\|/\varepsilon)$ communication complexity, where $\tilde{L}$ is the smoothness parameter of the objective function and $\mathcal{A}$ is related to either the graph Laplacian or the transpose of the oriented incidence matrix of the communication network. The complexities can be further improved to $\mathcal{O}((\tilde{L}/\mu)^{1/2}\log(1/\varepsilon))$ and $\mathcal{O}((\tilde{L}/\mu)^{1/2}\log(1/\varepsilon)+\|\mathcal{A}\|/\varepsilon^{1/2})$ respectively with the additional assumption of strong convexity modulus μ. We also propose a stochastic variant, namely the primal dual sliding (SPDS) algorithm for convex smooth problems with stochastic gradients. The SPDS algorithm utilizes the mini-batch technique and enables the agents to perform sampling and communication simultaneously. It computes a stochastic ε-solution with $\mathcal{O}((\tilde{L}/\varepsilon)^{1/2} + (\sigma/\varepsilon)^2)$ sampling complexity, which can be further improved to $\mathcal{O}((\tilde{L}/\mu)^{1/2}\log(1/\varepsilon) + \sigma^2/\varepsilon)$ in the strong convexity case. Here σ^2 is the variance of the stochastic gradient. The communication complexities of SPDS remain the same as that of the deterministic case. All the aforementioned gradient and sampling complexities match the lower complexity bounds for centralized convex smooth optimization and are independent of the network structure. To the best of our knowledge, these gradient and sampling complexities have not been obtained before in the literature of decentralized optimization.

Key words. Multi-agent optimization, decentralized optimization, saddle point problems, gradient complexity, sampling complexity, communication complexity, gradient sliding

AMS subject classifications. 90C25, 90C06, 49M37, 93A14, 90C15

1. Introduction. The problem of interest in this paper is the following decentralized multi-agent optimization problem in which the agents will collaboratively minimize an overall objective as the sum of all local objective f_i's.:

$$(1.1) \qquad \min_{x \in X} \sum_{i=1}^m f_i(x).$$

Here each $f_i : X^{(i)} \to \mathbb{R}$ is a convex smooth function defined over a closed convex set $X^{(i)} \in \mathbb{R}^d$ and $X := \cap_{i=1}^m X^{(i)}$. Under decentralized settings, each agent is expected to collect data, perform numerical operations using local data, and pass information to the neighboring agents in a communication network; no agent has full knowledge about other agents' local objectives or the communication network. This type of decentralized problems has many applications in signal processing, control, robust statistical inference, machine learning among others (see, e.g., [6, 11, 25, 26, 21, 7]).

In this paper, we assume that the communication network between the agents are defined by a connected undirected graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N} = \{1, \dots, m\}$

1

is the set of indices of agents and $\mathcal{E} \subseteq \mathcal{N} \times \mathcal{N}$ is the set of communication edges between them. Each agent $i \in \mathcal{N}$ could directly communicate with the agents in its neighborhood $N_i := \{ j \in \mathcal{N} \mid (i,j) \in \mathcal{E} \} \cup \{i\}$. For convenience we assume that there exists a loop (i,i) for all agents $i \in \mathcal{N}$. In addition, we assume that the local objectives are all large-dimensional functions and the i-th agent may only be able to access the information of its local objective $f_i(x)$ through a sampling process of its first-order information, e.g., the agent has objective $f_i(x) = \mathbb{E}_{\xi_i}[F_i(x, \xi_i)]$, where the expectation is taken with respect to the random variable ξ_i and the distribution of ξ_i is not known in advance. In order to solve problem (1.1) collaboratively, the agents have to sample the first-order information of their own local objective f_i and also communicate with the neighbors in the communication network in order to reach consensus. Our goal in this paper is to design an algorithm that is efficient, in terms of both the sampling and communication complexity, on solving the multi-agent decentralized problem (1.1).

1.1. Problem formulation and assumptions. The aforementioned multi-agent problem can also be formulated as the following linearly constrained problem:

$$(1.2) \qquad \min_{x \in \mathcal{X}} f(x) := \sum_{i=1}^{m} f_i(x^{(i)}) \text{ s.t. } \mathcal{A}x = 0,$$

where the overall objective function $f : \mathcal{X} \to \mathbb{R}$ is defined on the Cartesian product $\mathcal{X} := X^{(1)} \times \cdots \times X^{(m)} \subseteq \mathbb{R}^{md}$ and we use the notation $x = (x^{(1)}, \ldots, x^{(m)}) \in \mathcal{X}$ to summarize a collection of local solutions $x^{(i)} \in X^{(i)}$. The matrix $\mathcal{A}$ in the linear constraint $\mathcal{A}x = 0$ is designed to enforce the conditions $x^{(i)} = x^{(j)}$ for all agents i and j that are connected by a communication edge. One possible choice of $\mathcal{A}$ is $\mathcal{A} = \mathcal{L} \otimes I_d \in \mathbb{R}^{md \times md}$ where $I_d \in \mathbb{R}^{d \times d}$ is an identity matrix and $\mathcal{L} \in \mathbb{R}^{m \times m}$ is the graph Laplacian matrix whose (i,j)-entry is $|N_i|-1$ if $i = j$, -1 if $i \neq j$ and $(i,j) \in \mathcal{E}$, and 0 otherwise. Here $|N_i|$ is the degree of node i. One other choice of $\mathcal{A}$ is $\mathcal{A} = \mathcal{B}^\top \otimes I_d \in \mathbb{R}^{|\mathcal{E}|d \times md}$ where $\mathcal{B} \in \mathbb{R}^{m \times |\mathcal{E}|}$ is an oriented incidence matrix of graph $\mathcal{G}$. Here $|\mathcal{E}|$ is the number of edges in $\mathcal{G}$. For both cases, it can be shown that $\mathcal{A}x = 0$ if and only if $x^{(i)} = x^{(j)}$ for all agents i and j that are connected via a communication edge. Note that problem (1.2) is equivalent to the saddle point problem

$$(1.3) \qquad \min_{x \in \mathcal{X}} \max_{z \in \mathbb{R}^{md}} f(x) + \langle \mathcal{A}x, z \rangle.$$

We assume that there exists a saddle point (x^*, z^*) for problem (1.3).

In this paper, our goal is to solve the multi-agent problem (1.1) by obtaining an ε-approximate solution $\overline{x}$ to the linearly constrained problem (1.2) such that $f(\overline{x}) - f(x^*) \leq \varepsilon$ and $\|\mathcal{A}\overline{x}\|_2 \leq \varepsilon$. We make the following assumptions on the information and sampling process of the local objective functions. For each i, we assume that

$$(1.4) \qquad f_i(x^{(i)}) := \tilde{f}_i(x^{(i)}) + \mu\nu_i(x^{(i)})$$

where $\mu \geq 0$, $\tilde{f}_i$ is a convex smooth function, and ν_i is a strongly convex function with strong convexity modulus 1 with respect to a norm $\|\cdot\|$. Here the gradient $\nabla\tilde{f}_i$ is Lipschitz continuous with constant $\tilde{L}$ with respect to the norm $\|\cdot\|$. The i-th agent can access first-order information of $\tilde{f}_i$ through a stochastic oracle which returns an unbiased gradient estimator $G_i(x^{(i)}, \xi^{(i)})$ for any inquiry point $x^{(i)} \in X^{(i)}$, where $\xi^{(i)}$ is a sample of an underlying random vector with unknown distribution. The unbiased gradient estimator satisfies $\mathbb{E}[G_i(x^{(i)}, \xi^{(i)})] = \nabla f_i(x^{(i)})$ and

$$(1.5) \qquad \mathbb{E}[\|G_i(x^{(i)}, \xi^{(i)}) - \nabla f_i(x^{(i)})\|_2^2] \leq \sigma^2, \ \forall x^{(i)} \in X^{(i)},$$

where $\|\cdot\|_2$ is the Euclidean norm. Note that the above assumption also covers the deterministic case, i.e., when $\sigma = 0$. For simplicity, in this paper we assume that the strong convexity constant μ, the Lipschitz constant $\tilde{L}$, and the variance σ^2 are the same among all local objective functions. With the introduction of $\tilde{f}_i$'s in (1.4), we can study the cases of both strongly convex and general convex objective functions for the multi-agent optimization problem (1.1). For convenience, we introduce the following notation for describing the overall objective function:

$$(1.6) \quad f(x) := \tilde{f}(x) + \mu\nu(x) \text{ where } \tilde{f}(x) := \textstyle\sum_{i=1}^{m} \tilde{f}_i(x^{(i)}) \text{ and } \nu(x) := \textstyle\sum_{i=1}^{m} \nu_i(x^{(i)}).$$

1.2. Related works and contributions of this paper. In many real-world applications, the topology of the communication network may constantly change, due to possible connectivity issues especially for agents that are Internet-of-Things devices like cellphones or car sensors. Therefore, it is important to design decentralized algorithms with sampling or computation complexity independent of the graph topology [20]. Ideally, our goal is to develop decentralized methods whose sampling (resp. gradient computation) complexity bounds are graph invariant and in the same order as those of centralized methods for solving stochastic (resp. deterministic) problems. Although there are fruitful research results in which the sampling or computational complexities and communication complexities are separated for decentralized (stochastic) optimization, for instance, (stochastic) (sub)gradient based algorithms [30, 22, 5, 29, 19, 10], dual type or ADMM based decentralized methods [28, 2, 27, 1, 31, 3], communication efficient methods [14, 4, 12, 16], and second-order methods [18, 17], none of the existing decentralized methods can achieve the aforementioned goal. In this paper, we pursue the goal of developing graph topology invariant decentralized optimization algorithms based on the gradient sliding methods [13, 15]. Our contributions in this paper can be summarized as follows.

First, for the general convex deterministic problem (1.1), we propose a novel decentralized algorithm, namely the primal dual sliding (PDS) algorithm, that is able to compute an ε-solution with $\mathcal{O}(1)((\tilde{L}/\varepsilon)^{1/2})$ gradient complexity. This complexity is invariant with respect to the topology of the communication network. To the best of our knowledge, this is the first decentralized algorithm for problem (1.1) that achieves the same order of gradient complexity bound as those for centralized methods. Such gradient complexity can be improved to $\mathcal{O}(1)((\tilde{L}/\mu)^{1/2} \log(1/\varepsilon))$ for the strongly convex case of problem (1.1).

Second, for the general convex stochastic problem (1.1) in which the gradients of the objective functions can only be estimated through stochastic first-order oracle, we propose a stochastic primal dual sliding (SPDS) algorithm that is able to compute an ε-solution with $\mathcal{O}(1)((\tilde{L}/\varepsilon)^{1/2} + (\sigma/\varepsilon)^2)$ sampling complexity. The established complexity is also invariant with respect to the topology of the communication network. Such result can be improved to $\mathcal{O}(1)((\tilde{L}/\mu)^{1/2} \log(1/\varepsilon) + \sigma^2/\varepsilon)$ for the strongly convex case.

Third, as a byproduct, we show that a simple extension of the PDS algorithm can be applied to solve certain convex-concave saddle point problems with bilinear coupling. For general convex smooth case, the number of gradient evaluations of $\nabla\tilde{f}$ and matrix operations (involving $\mathcal{A}$ and $\mathcal{A}^\top$) are bounded by $\mathcal{O}(1)((\tilde{L}/\varepsilon)^{1/2})$ and $\mathcal{O}(1)((\tilde{L}/\varepsilon)^{1/2} + \|\mathcal{A}\|/\varepsilon)$ respectively. As a special case, our proposed algorithm is also able to solve convex smooth optimization problems with linear constraints. This is the first time such complexity results are achieved for linearly constrained convex smooth optimization. We also extend our results to strongly convex problems.

1.3. Organization of the paper. This paper is organized as follows. In section 2 we present the PDS algorithm for decentralized optimization. In section 3 we present the SPDS algorithm for problems that require sampling of stochastic gradients. The byproduct results on general bilinearly coupled saddle point problems are presented in Section 4. To facilitate reading, we postpone all the major proofs to Section 5. In Section 6 we present some preliminary numerical experiment results. Finally the concluding remarks are presented in Section 7.

2. The primal dual sliding algorithm for decentralized optimization. In this section, we propose a primal dual sliding (PDS) algorithm for solving the linearly constrained formulation (1.2) for decentralized multi-agent optimization in which each agent i has access to the deterministic first-order information of its local objective f_i (i.e., $\sigma = 0$ in the assumption (1.5)). Inspired by the saddle point formulation (1.3) and the decoupling of the strongly convex term in (1.4), we propose to study the following saddle point problem:

$$(2.1) \qquad \min_{x \in \mathcal{X}} \max_{y \in \mathbb{R}^{md}, z \in \mathbb{R}^{md}} \mu\nu(x) + \langle x, y + \mathcal{A}^\top z \rangle - \tilde{f}^*(y).$$

Here $\tilde{f}^*$ is the convex conjugate of the function $\tilde{f}$ defined in (1.6). Specifically, $\tilde{f}^*$ can be described as the sum of the convex conjugates $\tilde{f}_i^*$ of functions $\tilde{f}_i$, i.e., $\tilde{f}^*(y) = \sum_{i=1}^m \tilde{f}_i^*(y^{(i)})$, where $y := (y^{(1)}, \ldots, y^{(m)}) \in \mathbb{R}^{md}$ and $y^{(1)}, \ldots, y^{(m)} \in \mathbb{R}^d$. Our proposed PDS algorithm is a primal-dual algorithm that constantly maintains and updates primal variables x and dual variables y and z. Most importantly, its algorithmic scheme allows the skipping of computations of gradients $\nabla \tilde{f}$ from time to time, to which we refer as a "sliding" feature. We describe the proposed PDS algorithm in Algorithms 2.1 and 2.2, where Algorithm 2.1 describes the implementation from agent i's perspective and Algorithm 2.2 focuses on implementation over the whole communication network. Here in Algorithm 2.1 we assume that $\mathcal{A} = \mathcal{L} \otimes I_d$ is defined by the graph Laplacian $\mathcal{L}$ of the communication network; similar implementation can be described if $\mathcal{A} = \mathcal{B}^\top \otimes I_d$ is defined by an oriented incidence matrix $\mathcal{B}$.

A few remarks are in place for Algorithms 2.1 and 2.2. First, in Algorithm 2.1, V_i and W_i are prox-functions utilized by the i-th agent. Specifically, they are defined based on strongly convex functions ν and $\tilde{f}_i^*$ respectively:

$$V_i(\hat{x}^{(i)}, x^{(i)}) := \nu(x^{(i)}) - \nu(\hat{x}^{(i)}) - \langle \nu'(\hat{x}^{(i)}), x^{(i)} - \hat{x}^{(i)} \rangle, \ \forall \hat{x}^{(i)}, x^{(i)} \in X^{(i)},$$

$$(2.8) \quad W_i(\hat{y}^{(i)}, y^{(i)}) := \tilde{f}_i^*(y^{(i)}) - \tilde{f}_i^*(\hat{y}^{(i)}) - \langle (\tilde{f}_i^*)'(\hat{y}^{(i)}), y^{(i)} - \hat{y}^{(i)} \rangle, \ \forall \hat{y}^{(i)}, y^{(i)} \in \mathbb{R}^d,$$

where $(\tilde{f}_i^*)'$ denotes the subgradient of $\tilde{f}_i^*$. We will also use the notation

$$(2.9) \qquad V(\hat{x}, x) := \sum_{i=1}^m V_i(\hat{x}^{(i)}, x^{(i)}) \text{ and } W(\hat{y}, y) := \sum_{i=1}^m W_i(\hat{y}^{(i)}, y^{(i)})$$

in the description of Algorithm 2.2. Second, the major difference between Algorithms 2.1 and 2.2 is the update step for y_k. However, it can be shown that the two update steps for y_k, i.e., (2.3) and (2.5), are equivalent and hence Algorithms 2.1 and 2.2 are equivalent. Indeed, in view of the definition of W in (2.8) and (2.9) and the optimality condition of (4.10), we have $-\tilde{x}_k + (1 + \tau_k)(\tilde{f}^*)'(y_k) - \tau_k(\tilde{f}^*)'(y_{k-1}) = 0$ for certain subgradients $(\tilde{f}^*)'(y_k)$ and $(\tilde{f}^*)'(y_{k-1})$ of $\tilde{f}^*$. Consequently, if $\underline{x}_{k-1} = (\tilde{f}^*)'(y_{k-1})$ or equivalently $y_{k-1} = \nabla \tilde{f}(\underline{x}_{k-1})$, and $\underline{x}_k$ is defined recursively as in (2.2), then we have $y_k = \nabla \tilde{f}(\underline{x}_k)$. Noting in the description of Algorithms 2.1 and 2.2 that

---

Algorithm 2.1 The PDS algorithm for solving (2.1), from agent i's perspective

---

Choose $x_0^{(i)}, \underline{x}_0^{(i)} \in X^{(i)}$, and set $\hat{x}_0^{(i)} = x_{-1}^{(i)} = x_0^{(i)}$, $y_0^{(i)} = \nabla \tilde{f}(\underline{x}_0^{(i)})$, and $z_0^{(i)} = 0$. .
for $k = 1, \ldots, N$ **do**
    Compute

$$\tilde{x}_k^{(i)} = x_{k-1}^{(i)} + \lambda_k(\hat{x}_{k-1}^{(i)} - x_{k-2}^{(i)})$$

(2.2)
$$\underline{x}_k^{(i)} = (\tilde{x}_k^{(i)} + \tau_k \underline{x}_{k-1}^{(i)})/(1 + \tau_k)$$

(2.3)
$$y_k^{(i)} = \nabla \tilde{f}_i(\underline{x}_k^{(i)})$$

and set $x_k^{0,(i)} = x_{k-1}^{(i)}$, $z_k^{0,(i)} = z_{k-1}^{(i)}$, and $x_k^{-1,(i)} = x_{k-1}^{T_{k-1}-1,(i)}$ (set $x_1^{-1,(i)} = x_0^{(i)}$).
    for $t = 1, \ldots, T_k$ **do**
        Compute

$$\tilde{u}_k^{t,(i)} = x_k^{t-1,(i)} + \alpha_k^t(x_k^{t-1,(i)} - x_k^{t-2,(i)})$$

$$z_k^{t,(i)} = \operatorname*{argmin}_{z^{(i)} \in \mathbb{R}^d} \left\langle -\sum_{j \in N_i} \mathcal{L}^{(i,j)} \tilde{u}_k^{t,(j)}, z^{(i)} \right\rangle + \frac{q_k^t}{2} \|z^{(i)} - z_k^{t-1,(i)}\|_2^2$$

(2.4)
$$x_k^{t,(i)} = \operatorname*{argmin}_{x^{(i)} \in X^{(i)}} \mu \nu_i(x) + \left\langle y_k^{(i)} + \sum_{j \in N_i} \mathcal{L}^{(i,j)} z_k^{t,(j)}, x^{(i)} \right\rangle$$
$$+ \eta_k^t V_i(x_k^{t-1,(i)}, x^{(i)}) + p_k V_i(x_{k-1}^{(i)}, x^{(i)})$$

    end for
    Set $x_k^{(i)} = x_k^{T_k,(i)}$, $z_k^{(i)} = z_k^{T_k,(i)}$, $\hat{x}_k^{(i)} = \sum_{t=1}^{T_k} x_k^{t,(i)}/T_k$, and $\hat{z}_k^{(i)} = \sum_{t=1}^{T_k} z_k^{t,(i)}/T_k$.
end for
Output $\overline{x}_N^{(i)} := \left(\sum_{k=1}^N \beta_k\right)^{-1} \sum_{k=1}^N \beta_k \hat{x}_k^{(i)}$.

Algorithm 2.2 The PDS algorithm for solving (2.1), whole network perspective

---

Choose $x_0, \underline{x}_0 \in X^{(i)}$, and set $\hat{x}_0 = x_{-1} = x_0$, $y_0 = \nabla \tilde{f}(\underline{x}_0)$, and $z_0 = 0$.
for $k = 1, \ldots, N$ **do**
    Compute

$$\tilde{x}_k = x_{k-1} + \lambda_k(\hat{x}_{k-1} - x_{k-2})$$

(2.5)
$$y_k = \operatorname*{argmin}_{y \in \mathbb{R}^{md}} \langle -\tilde{x}_k, y \rangle + \tilde{f}^*(y) + \tau_k W(y_{k-1}, y).$$

Set $x_k^0 = x_{k-1}$, $z_k^0 = z_{k-1}$, and $x_k^{-1} = x_{k-1}^{T_{k-1}-1}$ (when $k = 1$, set $x_1^{-1} = x_0$).
    for $t = 1, \ldots, T_k$ **do**
(2.6)
$$\tilde{u}_k^t = x_k^{t-1} + \alpha_k^t(x_k^{t-1} - x_k^{t-2})$$

$$z_k^t = \operatorname*{argmin}_{z \in \mathbb{R}^{md}} \langle -\mathcal{A}\tilde{u}_k^t, z \rangle + \frac{q_k^t}{2} \|z - z_k^{t-1}\|_2^2$$

(2.7)
$$x_k^t = \operatorname*{argmin}_{x \in \mathcal{X}} \mu \nu(x) + \langle y_k + \mathcal{A}^\top z_k^t, x \rangle + \eta_k^t V(x_k^{t-1}, x) + p_k V(x_{k-1}, x)$$

    end for
    Set $x_k = x_k^{T_k}$, $z_k = z_k^{T_k}$, $\hat{x}_k = \sum_{t=1}^{T_k} x_k^t/T_k$, and $\hat{z}_k = \sum_{t=1}^{T_k} z_k^t/T_k$.
end for
Output $\overline{x}_N := \left(\sum_{k=1}^N \beta_k\right)^{-1} \sum_{k=1}^N \beta_k \hat{x}_k$.

$y_0 = \nabla \tilde{f}(\underline{x}_0)$, by induction we have that $y_k = \nabla \tilde{f}(\underline{x}_k)$ for all k. The converse of the above derivation is also true, hence we can conclude that the descriptions of y_k in (2.3) and (2.5) are equivalent. It is then immediate to observe that Algorithms 2.1 and 2.2 are equivalent descriptions of the proposed PDS algorithm. Third, observe that there are two loops, namely the inner and outer loops, in the PDS algorithm. On one hand, communications are performed only in the inner loops (for $t = 1, \ldots, T_k$) and hence the PDS algorithm has in total $2 \sum_{k=1}^{N} T_k$ rounds of communications. On the other hand, gradient computations are only performed in the outer loop (for $k = 1, \ldots, N$) and therefore the PDS algorithm has in total N gradient evaluations. Here, note that gradient computations and communications are not performed in a one-to-one fashion; in fact the PDS algorithm skips the computations of gradients $\nabla \tilde{f}$ frequently. The skipping of the gradient computations is the "sliding" feature of PDS that allows the gradient complexity to be independent of the graph topology. Finally, observe that our output is a weighted average of $\hat{x}_k$ from $k = 1$ to N, in which $\hat{x}_k$ is the average of inner loop computation results $\{x_k^t\}_{t=1}^{T_k}$. Note that although we are using equal weights, $\hat{x}_k$ may also be chosen as a weighted average of x_k^t's. The analysis and complexity results will remain the same.

Our main convergence result for the PDS algorithm is described in Proposition 2.1 below. We also provide example parameter choices of Algorithm 2.1 in Theorem 2.2 that satisfy the conditions in Proposition 2.1. To facilitate reading, we delay the proof of Proposition 2.1 to Section 5.

PROPOSITION 2.1. *Suppose that the parameters of Algorithm 2.1 satisfy the following conditions:*

- *For any $k \geq 2$,*

$$\beta_k \tau_k \leq \beta_{k-1}(\tau_{k-1} + 1), \ \beta_{k-1} = \beta_k \lambda_k, \ \tilde{L} \lambda_k \leq p_{k-1} \tau_k,$$

(2.10) $\quad \beta_k T_{k-1} \alpha_k^1 = \beta_{k-1} T_k, \ \alpha_k^1 \|\mathcal{A}\|^2 \leq \eta_{k-1}^{T_{k-1}} q_k^1, \ \beta_k T_{k-1} q_k^1 \leq \beta_{k-1} T_k q_{k-1}^{T_{k-1}},$

$$\beta_k T_{k-1}(\eta_k^1 + p_k T_k) \leq \beta_{k-1} T_k (\mu + \eta_{k-1}^{T_{k-1}} + p_{k-1});$$

- *For any $t \geq 2$ and $k \geq 1$,*

(2.11) $\qquad \alpha_k^t = 1, \ \|\mathcal{A}\|^2 \leq \eta_k^{t-1} q_k^t, \ q_k^t \leq q_k^{t-1}, \ \eta_k^t \leq \mu + \eta_k^{t-1} + p_k;$

- *In the first and last outer iterations,*

(2.12) $\qquad \tau_1 = 0, \ p_N(\tau_N + 1) \geq \tilde{L}, \ and \ \eta_N^{T_N} q_N^{T_N} \geq \|\mathcal{A}\|^2.$

Then we have

$$f(\overline{x}_N) - f(x^*) \leq \left(\sum_{k=1}^{N} \beta_k \right)^{-1} \beta_1 \left(\frac{\eta_1^1}{T_1} + p_1 \right) V(x_0, x^*) \ and$$

$$\|\mathcal{A}\overline{x}_N\|_2 \leq \left(\sum_{k=1}^{N} \beta_k \right)^{-1} \beta_1 \left[\frac{q_1^1}{2T_1}(\|z^*\|_2 + 1)^2 + \left(\frac{\eta_1^1}{T_1} + p_1 \right) V(x_0, x^*) \right].$$

THEOREM 2.2. *Denote $\tau := \sqrt{2\tilde{L}/\mu}$ and $\Delta := \lceil 2\tau + 1 \rceil$ if $\mu > 0$, and $\Delta := +\infty$ if $\mu = 0$. Suppose that the parameters in Algorithm 2.1 are set to the following: for all $k \leq \Delta$,*

(2.13) $\qquad \tau_k = \frac{k-1}{2}, \ \lambda_k = \frac{k-1}{k}, \ \beta_k = k, \ p_k = \frac{2\tilde{L}}{k}, \ T_k = \left\lceil \frac{kR\|\mathcal{A}\|}{\tilde{L}} \right\rceil,$

for all $k > \Delta$,

(2.14) $\quad \tau_k = \tau, \ \lambda_k = \lambda := \frac{\tau}{1+\tau}, \ \beta_k = \Delta\lambda^{-(k-\Delta)}, \ p_k = \frac{\tilde{L}}{1+\tau}, \ T_k = \left\lceil \frac{2(1+\tau)R\|\mathcal{A}\|}{\tilde{L}\lambda^{\frac{k-\Delta}{2}}} \right\rceil,$

and for all k and t,

$$\eta_k^t = (p_k + \mu)(t-1) + p_k T_k, q_k^t = \frac{\tilde{L}T_k}{2\beta_k R^2}, \alpha_k^t = \begin{cases} \frac{\beta_{k-1}T_k}{\beta_k T_{k-1}}, & \text{if } k \geq 2 \text{ and } t = 1; \\ 1, & \text{otherwise.} \end{cases}$$

Applying Algorithm 2.1 to solve problem (1.2) we have

(2.15) $\qquad f(\overline{x}_N) - f(x^*) \leq \min\left\{\frac{2}{N^2}, \lambda^{N-\Delta}\right\} \cdot 4\tilde{L}V(x_0, x^*)$

(2.16) $\qquad \|\mathcal{A}\overline{x}_N\|_2 \leq \min\left\{\frac{2}{N^2}, \lambda^{N-\Delta}\right\}\left[\frac{\tilde{L}}{4R^2}(\|z^*\|_2 + 1)^2 + 4\tilde{L}V(x_0, x^*)\right].$

Specially, if we set

(2.17) $$R = \frac{\|z^*\|_2 + 1}{4\sqrt{V(x_0, x^*)}},$$

then we have the following gradient and communication complexity results for Algorithm 2.1 (here $\mathcal{O}(1)$ is a constant that is independent of N, $\|\mathcal{A}\|$ and ϵ):

    a) If $\mu = 0$, i.e., problem (1.2) is smooth and convex, then we can obtain an ε-solution after at most $N := \left\lceil 4\sqrt{\tilde{L}V(x_0, x^*)/\varepsilon} \right\rceil$ gradient evaluations and $2N + 96\|\mathcal{A}\|/\varepsilon$ communications.

    b) If $\mu > 0$, i.e., problem (1.2) is smooth and strongly convex, then we can obtain an ε-solution after at most $N := \mathcal{O}(1)\left(1 + \sqrt{\tilde{L}/\mu}\log(\tilde{L}V(x_0, x^*)/\varepsilon)\right)$ gradient evaluations and $2N + \mathcal{O}(1)\|\mathcal{A}\|(1 + 1/\sqrt{\varepsilon})$ communications.

    Proof. In view of the selection of η_k^t and q_k^t, we can observe that

$$\eta_k^{t-1}q_k^t \geq \frac{\tilde{L}p_k}{2\beta_k}\left(\frac{T_k}{R}\right)^2 = \begin{cases} \|\mathcal{A}\|^2\left(\frac{\tilde{L}T_k}{kR\|\mathcal{A}\|}\right)^2, & k \leq \Delta; \\ \frac{(2\tau+2)\|\mathcal{A}\|^2}{\Delta}\left(\frac{\tilde{L}\lambda^{\frac{k-\Delta}{2}}T_k}{2(1+\tau)R\|\mathcal{A}\|}\right)^2, & k > \Delta, \end{cases}$$

and together with the definitions of Δ and T_k we have $\eta_k^{t-1}q_k^t \geq \|\mathcal{A}\|^2$ for all $k \geq 1$ and $t \geq 2$. Using this observation, it is easy to verify that (2.11) holds. Noting that the above observation also implies that $\eta_N^{T_N}q_N^{T_N} \geq \|\mathcal{A}\|^2$, and also noting that $\tau_1 = 0$ and that $p_N(\tau_N + 1) = \tilde{L}(N+1)/N$ when $N \leq \Delta$ and $p_N(\tau_N + 1) = \tilde{L}$ when $N > \Delta$, we have (2.12) holds. It remains to verify (2.10). Note that for all k we have $\beta_k T_{k-1}\alpha_k^1 = \beta_{k-1}T_k$ and $\beta_k T_{k-1}q_k^1 = \tilde{L}T_k T_{k-1}/(2R^2) = \beta_{k-1}T_k q_{k-1}^{T_{k-1}}$. We consider three cases: $2 \leq k \leq \Delta$, $k = \Delta + 1$, and $k > \Delta + 1$. When $2 \leq k \leq \Delta$, we have

$$\beta_{k-1}(\tau_{k-1} + 1) = (k-1)k/2 = \beta_k\tau_k,$$

$$\beta_{k-1} = k - 1 = \beta_k\lambda_k,$$

$$\tilde{L}\lambda_k = \tilde{L}(k-1)/k < \tilde{L} = p_{k-1}\tau_k,$$

$$\frac{\eta_{k-1}^{T_{k-1}}q_k^1}{\alpha_k^1\|\mathcal{A}\|^2} \geq \frac{\eta_{k-1}^1 q_k^1}{\alpha_k^1\|\mathcal{A}\|^2} \geq \left(\frac{\tilde{L}T_{k-1}}{(k-1)R\|\mathcal{A}\|}\right)^2 \geq 1,$$

$$\beta_k T_{k-1}(\eta_k^1 + p_k T_k) - \beta_{k-1}T_k(\mu + \eta_{k-1}^{T_{k-1}} + p_{k-1}) = -(k-1)\mu T_k T_{k-1} \leq 0.$$

When $k = \Delta + 1$, from the definition of Δ we have $2\tau + 1 \leq \Delta \leq 2\tau + 2$. Therefore

$$\beta_k \tau_k - \beta_{k-1}(\tau_{k-1} + 1) = \Delta\lambda^{-1}\tau - \Delta(\Delta + 1)/2 = (\Delta/2)(2(1 + \tau) - (\Delta + 1)) \leq 0,$$

$$\beta_{k-1} - \beta_k\lambda_k = \Delta - \Delta\lambda^{-1}\lambda = 0,$$

$$\tilde{L}\lambda_k - p_{k-1}\tau_k = \tilde{L}\lambda - (2\tilde{L}/\Delta)\tau \leq \tilde{L}\lambda - (\tilde{L}/(1 + \tau))\tau = 0,$$

$$\frac{\eta_{k-1}^{T_{k-1}} q_k^1}{\alpha_k^1 \|\mathcal{A}\|^2} \geq \frac{\eta_{k-1}^1 q_k^1}{\alpha_k^1 \|\mathcal{A}\|^2} \geq \left(\frac{\tilde{L}T_\Delta}{\Delta R\|\mathcal{A}\|}\right)^2 \geq 1,$$

$$\beta_k T_{k-1}(\eta_k^1 + p_k T_k) - \beta_{k-1}T_k(\mu + \eta_{k-1}^{T_{k-1}} + p_{k-1}) = \mu\left[\Delta(\tau - 1) - 2\tau^2\right]T_k T_{k-1} \leq 0.$$

When $k > \Delta + 1$, we have

$$\beta_{k-1}(\tau_{k-1} + 1) = \Delta\lambda^{-(k-1-\Delta)}(\tau + 1) = \Delta\lambda^{-(k-\Delta)}\tau = \beta_k \tau_k,$$

$$\beta_{k-1} = \Delta\lambda^{-(k-1-\Delta)} = \beta_k\lambda_k,$$

$$\tilde{L}\lambda_k = \tilde{L}\tau/(1 + \tau) = p_{k-1}\tau_k,$$

$$\frac{\eta_{k-1}^{T_{k-1}} q_k^1}{\alpha_k^1 \|\mathcal{A}\|^2} \geq \frac{\eta_{k-1}^1 q_k^1}{\alpha_k^1 \|\mathcal{A}\|^2} \geq \frac{2\tau+2}{\Delta} \cdot \left(\frac{\tilde{L}\lambda^{(k-1-\Delta)/2} T_{k-1}}{2(1+\tau)R\|\mathcal{A}\|}\right)^2 \geq 1,$$

$$\frac{\beta_k T_{k-1}(\eta_k^1 + p_k T_k)}{\beta_{k-1}T_k(\mu + \eta_{k-1}^{T_{k-1}} + p_{k-1})} = \frac{\tau^2(1+\tau)}{\tau(\tau^2 + 1 + \tau)} \leq 1.$$

We now can conclude that (2.10) holds. Note that the summation $\sum_{k=1}^N \beta_k$ can be lower bounded as follows: when $N \leq \Delta$, we have $\sum_{k=1}^N \beta_k = N(N + 1)/2 > N^2/2$; when $N > \Delta$, we have $\sum_{k=1}^N \beta_k > \beta_N \geq \lambda^{-(N-\Delta)}$.

Applying Proposition 2.1 and using the above note on the summation $\sum_{k=1}^N \beta_k$ and the parameter selection in (2.13), we obtain results (2.15) and (2.16). Moreover, setting R to (2.17) we conclude that

$$\max\{f(\overline{x}_N) - f(x^*), \|\mathcal{A}\overline{x}_N\|_2\} \leq \min\left\{\frac{2}{N^2}, \lambda^{N-\Delta}\right\} 8\tilde{L}V(x_0, x^*).$$

We conclude immediately that $\overline{x}_N$ will be an ε-solution if $N = \left\lceil 4\sqrt{\tilde{L}V(x_0, x^*)/\varepsilon}\right\rceil$ or $N = \left\lceil \Delta + \log_{\lambda^{-1}}(8\tilde{L}V(x_0, x^*)/\varepsilon)\right\rceil$. As a consequence, when the problem is smooth and convex, i.e., $\mu = 0$, by setting $N = \left\lceil 4\sqrt{\tilde{L}V(x_0, x^*)/\varepsilon}\right\rceil$ we can obtain an ε-solution $\overline{x}_N$. Using the definitions of T_k and R in (2.13) and (2.17) respectively, the total number of communication rounds required by the PDS algorithm can be bounded by $2\sum_{k=1}^N T_k \leq 2N + N(N+1)R\|\mathcal{A}\|/\tilde{L} \leq 2N + 96\|\mathcal{A}\|/\varepsilon$. When the problem is smooth and strongly convex, i.e., $\mu > 0$, we may set $N := \left\lceil \Delta + \log_{\lambda^{-1}}(8\tilde{L}V(x_0, x^*)/\varepsilon)\right\rceil$ to obtain an ε-solution $\overline{x}_N$. In view of the fact that $-\log\lambda = \log(1 + 1/\tau) \geq 1/(1 + \tau)$, the total number of gradient computations is bounded by

$$N \leq 1 + \Delta + \frac{1}{-\log\lambda}\log\frac{8\tilde{L}V(x_0, x^*)}{\varepsilon} \leq 1 + \Delta + (\tau + 1)\left(\log\frac{8\tilde{L}V(x_0, x^*)}{\varepsilon}\right)$$

$$= \mathcal{O}(1)\left(1 + \sqrt{\tilde{L}/\mu}\log(\tilde{L}V(x_0, x^*)/\varepsilon)\right).$$

Also, observing that $\lambda^{-N} \leq 8\lambda^{-\Delta-1}\tilde{L}V(x_0, x^*)/\varepsilon$, using the definitions of λ, T_k and R in (2.13), (2.14) and (2.17), the total number of communication rounds required by

290 the PDS algorithm is bounded by

$$2\sum_{k=1}^{N}T_k \leq 2\sum_{k=1}^{\Delta}\left(\frac{kR\|\mathcal{A}\|}{\tilde{L}}+1\right)+2\sum_{k=\Delta+1}^{N}\left(\frac{2(1+\tau)\|\mathcal{A}\|R}{\tilde{L}\lambda^{\frac{k-\Delta}{2}}}+1\right)$$

$$=2N+\frac{\Delta(\Delta+1)R\|\mathcal{A}\|}{\tilde{L}}+\frac{4(1+\tau)R\lambda^{\frac{\Delta}{2}}}{\tilde{L}(1-\lambda^{\frac{1}{2}})}\cdot\|\mathcal{A}\|(\lambda^{-\frac{N}{2}}-\lambda^{-\frac{\Delta}{2}})$$

$$\leq 2N+\mathcal{O}(1)\|\mathcal{A}\|+\mathcal{O}(1)\|\mathcal{A}\|/\sqrt{\varepsilon}. \qquad\qquad \square$$

295 In view of the above theorem, we can conclude that the number of gradient eval-
296 uations required by the PDS algorithm to obtain an ε-solution to the decentralized
297 optimization problem (1.2) does not depend on the communication network topology;
298 it is only dependent on the Lipschitz constant $\tilde{L}$, strong convexity parameter μ, and
299 proximity to an optimal solution $V(x_0, x^*)$. Moreover, the number of gradient eval-
300 uations required by the PDS algorithm is in the same order of magnitude as those
301 optimal centralized algorithms (see, e.g., [24]). To the best of our knowledge, this
302 is the first time such complexity bounds for gradient evaluations are established for
303 decentralized algorithms. Note that the communications complexity for computing
304 an ε-solution still depends on the graph topology of the communication network.

305 **3. The stochastic primal-dual sliding algorithm for stochastic decen-**
306 **tralized optimization.** In this section, we follow the assumption of the unbiased gra-
307 dient estimators stated in (1.5) and propose a stochastic primal-dual sliding (SPDS)
308 algorithm for solving the decentralized multi-agent optimization problem (1.2) under
309 the stochastic first-order oracle. Our proposed SPDS algorithm is a stochastic primal-
310 dual algorithm that possesses a similar "sliding" scheme as the PDS algorithm. In
311 particular, SPDS integrates the mini-batch technique and uses increasing batch size,
312 while skipping stochastic gradient computation from time to time. We describe the
313 SPDS algorithm in Algorithms 3.1 and 3.2.

Algorithm 3.1 The SPDS algorithm for solving (2.1), from agent i's perspective

Modify (2.3) and (2.4) in Algorithm 2.1 to

(3.1)
$$v_k^{(i)} = \frac{1}{c_k}\sum_{j=1}^{c_k}G_i(\underline{x}_k^{(i)}, \xi_k^{(i),j}) \text{ and}$$

$$x_k^{t,(i)} = \underset{x^{(i)}\in X^{(i)}}{\operatorname{argmin}} \ \mu\nu_i(x) + \left\langle v_k^{(i)}+\sum_{j\in N_i}\mathcal{L}^{(i,j)}z_k^{t,(j)}, x^{(i)}\right\rangle$$

$$+\eta_k^t V_i(x_k^{t-1,(i)}, x^{(i)}) + \frac{p_k}{2}\|x_{k-1}^{(i)}-x^{(i)}\|_2^2$$

Algorithm 3.2 The SPDS algorithm for solving (2.1), whole network perspective

Modify (2.7) in Algorithm 2.2 to

$$x_k^t = \underset{x\in\mathcal{X}}{\operatorname{argmin}} \ \mu\nu(x) + \langle v_k+\mathcal{A}^\top z_k^t, x\rangle + \eta_k^t V(x_k^{t-1}, x) + \frac{p_k}{2}\|x_{k-1}-x\|_2^2,$$

in which v_k satisfies $\mathbb{E}[v_k] = y_k$ and

(3.2)
$$\mathbb{E}[\|v_k-y_k\|_2^2] \leq \sigma_k^2, \text{ where } \sigma_k := \frac{\sigma}{\sqrt{c_k}}$$

---

314 The difference between the SPDS and PDS algorithms is that the SPDS algorithm
315 uses a mini-batch of c_k stochastic samples of gradients while PDS requires the access of
316 exact gradients, i.e., $y_k = \nabla\tilde{f}(\underline{x}_k)$. As a result, the update of y_k is replaced by a mini-
317 batch update step and equations (2.4) and (2.7) of the PDS algorithm are replaced

by their stochastic version accordingly. In order to use the mini-batch technique, we need to replace the prox-function $V(\cdot, \cdot)$ in the PDS algorithm in Algorithm 2.2 by Euclidean distances. Recalling the assumption on the variance of stochastic samples in (1.5) and noting that the batch size at the k-th outer iteration is c_k, it is easy to obtain that the variance of the gradient estimator v_k is $\sigma_k^2 := \sigma^2 / c_k$, as stated in (3.2). It should also be noted that in many applications the i-th agent may be able to collect samples $\{\xi_k^{(i),j}\}_{j=1}^{c_k}$ of the underlying random variable in (3.1) for computing the gradient estimator $v_k^{(i)}$. For example, in online supervised machine learning the i-th agent can receive samples $\xi_k^{(i),j}$'s from online data streams of the training dataset. In such case, the SPDS algorithm allows all agents to collect stochastic samples and perform communications simultaneously. This is because that as soon as the mini-batch computation of the gradient estimator $v_k^{(i)}$ in (3.1) is finished in the k-th outer iteration, the i-th agent can start collecting future stochastic samples $\{\xi_{k+1}^{(i),j}\}_{j=1}^{c_{k+1}}$ for the next outer iteration. Consequently, the collection of future samples and the communication with neighboring agents in the current k-th outer iteration can be performed simultaneously.

The convergence of the SPDS algorithm is described in the following proposition, whose proof is presented in Section 5. Followed by the proposition we also describe example parameter choices in Theorem 3.2.

PROPOSITION 3.1. *Suppose that the parameters of Algorithm 3.1 satisfy conditions described in* (2.10)–(2.12), *among which two conditions are modified to*

$$(3.3) \qquad 2\tilde{L}\lambda_k \leq p_{k-1}\tau_k, \ \forall k \geq 2, \ and \ p_N(\tau_N + 1) \geq 2\tilde{L},$$

in which the Lipschitz constant $\tilde{L}$ is with respect to the Euclidean norm. We have

$$\mathbb{E}\left[f(\overline{x}_N) - f(x^*)\right] \leq \left(\sum_{k=1}^N \beta_k\right)^{-1} \left[\frac{\beta_1}{2}\left(\frac{\eta_1^1}{T_1} + p_1\right)\|x_0 - x^*\|_2^2 + \sum_{k=1}^N \frac{\beta_k \sigma^2}{p_k c_k}\right] \ and$$

$$\mathbb{E}\left[\|\mathcal{A}\overline{x}_N\|_2\right] \leq \left(\sum_{k=1}^N \beta_k\right)^{-1}\left[\frac{\beta_1 q_1^1}{2T_1}(\|z^*\|_2 + 1)^2\right.$$
$$\left. + \frac{\beta_1}{2}\left(\frac{\eta_1^1}{T_1} + p_1\right)\|x_0 - x^*\|_2^2 + \sum_{k=1}^N \frac{\beta_k \sigma^2}{p_k c_k}\right].$$

THEOREM 3.2. *Denote $\tau := 2\sqrt{\tilde{L}/\mu}$ and $\Delta := \lceil 2\tau + 1\rceil$ if $\mu > 0$, and $\Delta := +\infty$ if $\mu = 0$. Suppose that the maximum number of outer iterations N is specified, and that the parameters in Algorithm 3.1 are set to the following: for all $k \leq \Delta$,*

$$(3.4) \quad \tau_k = \frac{k-1}{2}, \ \lambda_k = \frac{k-1}{k}, \ \beta_k = k, \ p_k = \frac{4\tilde{L}}{k}, T_k = \left\lceil\frac{kR\|\mathcal{A}\|}{\tilde{L}}\right\rceil, c_k = \left\lceil\frac{\min\{N,\Delta\}\beta_k c}{p_k \tilde{L}}\right\rceil,$$

for all $k \geq \Delta + 1$,

$$\tau_k = \tau, \ \lambda_k = \lambda := \frac{\tau}{1+\tau}, \ \beta_k = \Delta\lambda^{-(k-\Delta)}, \ p_k = \frac{2\tilde{L}}{1+\tau},$$

$$(3.5)$$
$$T_k = \left\lceil\frac{2(1+\tau)R\|\mathcal{A}\|}{\tilde{L}\lambda^{\frac{k-\Delta}{2}}}\right\rceil, c_k = \left\lceil\frac{(1+\tau)^2\Delta c}{\tilde{L}^2\lambda^{\frac{k+N-2\Delta}{2}}}\right\rceil,$$

and for all k and t,

$$(3.6) \quad \eta_k^t = (p_k + \mu)(t-1) + p_k T_k, \ q_k^t = \frac{\tilde{L}T_k}{4\beta_k R^2}, \ \alpha_k^t = \begin{cases} \frac{\beta_{k-1}T_k}{\beta_k T_{k-1}} & k \geq 2 \ and \ t = 1 \\ 1 & otherwise. \end{cases}$$

356 *Applying Algorithm 3.1 to solve problem (1.2) we have*

357 (3.7) $f(\overline{x}_N) - f(x^*) \le \min\left\{\frac{2}{N^2}, \lambda^{N-\Delta}\right\}\left[4\tilde{L}\|x_0 - x^*\|_2^2 + \frac{\tilde{L}\sigma^2}{c}\right]$ *and*

358 (3.8) $\|\mathcal{A}\overline{x}_N\|_2 \le \min\left\{\frac{2}{N^2}, \lambda^{N-\Delta}\right\}\left[\frac{\tilde{L}}{8R^2}(\|z^*\|_2 + 1)^2 + 4\tilde{L}\|x_0 - x^*\|_2^2 + \frac{\tilde{L}\sigma^2}{c}\right].$
359

360 *Specially, if we set*

361 (3.9) $$R = \frac{\|z^*\|_2 + 1}{4\|x_0 - x^*\|_2} \ \text{and} \ c = \frac{2\sigma^2}{\|x_0 - x^*\|_2^2},$$
362

363 *then we have the following sampling and communication complexity results for Algo-*
364 *rithm 3.1 (here $\mathcal{O}(1)$ is a constant that is independent of σ, N, $\|\mathcal{A}\|$ and ϵ):*

365   a) *If $\mu = 0$, by setting $N = \left\lceil\sqrt{10\tilde{L}\|x_0 - x^*\|_2^2/\varepsilon}\right\rceil$ we can obtain a stochastic*
366   *ε-solution with $N + 800\sigma^2\|x_0 - x^*\|_2^2/\varepsilon$ gradient samples and $N + 600\|\mathcal{A}\|/\varepsilon$*
367   *rounds of communication.*
368   b) *If $\mu > 0$, by setting $N = \left\lceil\Delta + \log_{\lambda^{-1}}(5\tilde{L}\|x_0 - x^*\|_2^2/\varepsilon)\right\rceil$ we can obtain a sto-*
369   *chastic ε-solution with $\mathcal{O}(1)\left(1 + \sqrt{\tilde{L}/\mu}\log(\tilde{L}\|x_0 - x^*\|_2^2/\varepsilon) + \sigma^2(1 + 1/\varepsilon)\right)$*
370   *samples and $\mathcal{O}(1)\left(1 + \sqrt{\tilde{L}/\mu}\log(\tilde{L}\|x_0 - x^*\|_2^2/\varepsilon) + \|\mathcal{A}\|(1 + 1/\sqrt{\varepsilon})\right)$ rounds*
371   *of communication.*

372   *Proof.* Following a similar argument to the proof of Theorem 2.2, it is easy to
373 verify that the parameter selections in (3.4) - (3.6) satisfy the conditions stated in
374 Proposition 3.1. The results in (3.7) and (3.8) follows immediately from the propo-
375 sition and our parameter selections. We now consider the case when $\mu = 0$ (hence
376 $\Delta = +\infty$). By the choice of N, the selections of β_k, p_k, R, and c in (3.4) and (3.9),
377 and noting that $\sum_{k=1}^{N}k^2 \le N^3$ we can bound the total number of gradient samples
378 by

379    $$\sum_{k=1}^{N}c_k \le \sum_{k=1}^{N}\left(1 + \frac{cNk^2}{4\tilde{L}^2}\right) = N + \frac{\sigma^2 N}{2\tilde{L}^2\|x_0 - x^*\|_2^2}\sum_{k=1}^{N}k^2$$

380    $$\le N + \frac{\sigma^2 N^4}{2\tilde{L}^2\|x_0 - x^*\|_2^2} = N + \frac{800\sigma^2\|x_0 - x^*\|_2^2}{\varepsilon^2}.$$
381

382 In the case when $\mu > 0$, similar to the proof of Theorem 2.2 we can observe two
383 bounds $N \le \mathcal{O}(1)\left(1 + \sqrt{\tilde{L}/\mu}\log(\tilde{L}\|x_0 - x^*\|_2^2/\varepsilon)\right)$ and $\lambda^{-N} \le (1/\varepsilon)\mathcal{O}(1)$. In view
384 of these bounds, the descriptions of c_k and c in (3.4), (3.5), and (3.9), we can bound
385 the total number of gradient samples by

386    $$\sum_{k=1}^{N}c_k \le \sum_{k=1}^{\Delta}\left(1 + \frac{\Delta\sigma^2 k^2}{2\tilde{L}^2\|x_0 - x^*\|_2^2}\right) + \sum_{k=\Delta+1}^{N}\left(1 + \frac{(1+\tau)^2\Delta c}{2\tilde{L}^2\lambda^{\frac{N-\Delta}{2}}}\lambda^{-\frac{k-\Delta}{2}}\right)$$

387    $$= N + \mathcal{O}(1)\sigma^2 + \mathcal{O}(1)\frac{1 - \lambda^{\frac{N-\Delta}{2}}}{1 - \lambda^{\frac{1}{2}}}\sigma^2\lambda^{-(N-\Delta)}$$

388    $$\le \mathcal{O}(1)\left(1 + \sqrt{\tilde{L}/\mu}\log(\tilde{L}\|x_0 - x^*\|_2^2/\varepsilon)\right) + \mathcal{O}(1)\sigma^2 + \mathcal{O}(1)\sigma^2/\varepsilon. \qquad \square$$
389

390   In view of Theorem 3.2, we can conclude that the sampling complexity achieved
391 by the SPDS algorithm does not depend on the communication network topology;

it is only dependent on the Lipschitz constant $\tilde{L}$, strong convexity parameter μ, the variance of the gradient estimators σ^2, and proximity to an optimal solution $\|x_0 - x^*\|_2$. Moreover, the achieved sampling complexity is in the same order of magnitude as those optimal centralized algorithms, such as accelerated stochastic approximation method [8, 9]. To the best of our knowledge, this is the first time such sampling complexity bounds are established for stochastic decentralized algorithms. Indeed, the total number of communications required to compute an ε-solution still depends on the graph topology. It should also be noted that the constant factors in part a) of the above theorem (800 and 600) can be further reduced, e.g., by assuming that $N \geq 5$. However, our focus is on the development of order-optimal complexity results and hence we skip these refinements.

4. A byproduct on solving bilinearly coupled saddle point problems. In this section, we present a byproduct of our analysis and show that the PDS algorithm described in Algorithm 2.2 is in fact an efficient algorithm for solving certain convex-concave saddle point problem with linear coupling. Specifically, in this section we consider the following saddle point problem:

$$(4.1) \qquad \min_{x \in \mathcal{X}} \max_{z \in \mathcal{Z}} f(x) + \langle \mathcal{A}x, z \rangle - h(z)$$

where $\mathcal{X}$ and $\mathcal{Z}$ are closed convex sets, $\mathcal{A}$ is a linear operator, and f and h are convex functions. Here, we assume that $h(z)$ is relatively simple, and that $f(x) = \tilde{f}(x) + \mu\nu(x)$, where $\mu \geq 0$, $\nu : \mathcal{X} \to \mathbb{R}$ is a strongly convex function with strong convexity modulus 1 with respect to a norm $\|\cdot\|$, and $\tilde{f} : X \to \mathbb{R}$ is a convex smooth function whose gradient is Lipschitz with constant $\tilde{L}$ with respect to the norm $\|\cdot\|$. Note that problem (4.1) is equivalent to

$$(4.2) \qquad \min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}, z \in \mathcal{Z}} \mu\nu(x) + \langle x, y + \mathcal{A}^\top z \rangle - \tilde{f}^*(y) - h(z)$$

where $\tilde{f}^*$ is the convex conjugate of $\tilde{f}$ and $\mathcal{Y} := \operatorname{dom} \tilde{f}^*$. Also, observe that when $\mathcal{Z}$ is a vector space and $h(z) = \langle b, z \rangle$, the saddle point problem (4.1) becomes

$$(4.3) \qquad \min_{x \in \mathcal{X}} f(x) \ \text{s.t.} \ \mathcal{A}x = b.$$

If in addition $b = 0$, $\mathcal{Z} = \mathbb{R}^{md}$, and $\mathcal{X}$, f, and $\mathcal{A}$ are defined in (1.2), then we obtain the linearly constrained reformulation of the multi-agent optimization problem (1.1).

We describe an extension of the PDS algorithm in Algorithm 4.1. In the algorithm description, U, V, and W are prox-functions defined by

$$(4.4) \qquad U(\hat{z}, z) := \zeta(z) - \zeta(\hat{z}) - \langle \zeta'(\hat{z}), z - \hat{z} \rangle,$$

$$(4.5) \qquad V(\hat{x}, x) := \nu(x) - \nu(\hat{x}) - \langle \nu'(\hat{x}), x - \hat{x} \rangle, \ \text{and}$$

$$(4.6) \qquad W(v, y) := \tilde{f}^*(y) - \tilde{f}^*(v) - \langle (\tilde{f}^*)'(v), y - v \rangle.$$

respectively, where ν and $\tilde{f}^*$ are defined in (4.2) and $\zeta : \mathcal{Z} \to \mathbb{R}$ is a strongly convex function with strong convexity modulus 1 with respect a norm $|\cdot|$. Noting from the strong convexity of ζ and ν we have for any $\hat{z}, z \in \mathcal{Z}$ and $\hat{x}, x \in \mathcal{X}$ that

$$(4.7) \qquad U(\hat{z}, z) \geq \tfrac{1}{2}|\hat{z} - z|^2 \ \text{and} \ V(\hat{x}, x) \geq \tfrac{1}{2}\|\hat{x} - x\|^2.$$

Moreover, since $\tilde{f}$ is convex and $\nabla\tilde{f}$ is Lipschitz with constant $\tilde{L}$ with respect to norm $\|\cdot\|$, $\tilde{f}^*$ should be strongly convex with strong convexity parameter $1/\tilde{L}$ with respect

437 to the dual norm $\|\cdot\|_*$. Consequently, noting the definition of W in (4.6) we have

438
439   (4.8) $$W(v,y) \geq \tfrac{1}{2\tilde{L}}\|v-y\|_*^2, \ \forall v,y \in \mathcal{Y}.$$

Algorithm 4.1 An equivalent alternative description of the PDS algorithm

---

Choose $x_0 \in \mathcal{X}$, $y_0 \in \mathcal{Y}$, and $z_0 \in \mathcal{Z}$. Set $\hat{x}_0 = x_{-1} = x_0$.
for $k = 1, \ldots, N$ **do**
   Compute

(4.9) $$\tilde{x}_k = x_{k-1} + \lambda_k(\hat{x}_{k-1} - x_{k-2}),$$

(4.10) $$y_k := \operatorname*{argmin}_{y \in \mathcal{Y}} \langle -\tilde{x}_k, y \rangle + \tilde{f}^*(y) + \tau_k W(y_{k-1}, y).$$

   Set $x_k^0 = x_{k-1}$, $z_k^0 = z_{k-1}$, and $x_k^{-1} = x_{k-1}^{T_{k-1}-1}$ (when $k=1$, set $x_1^{-1} = x_0$).
   for $t = 1, \ldots, T_k$ **do**

$$\tilde{u}_k^t = x_k^{t-1} + \alpha_k^t(x_k^{t-1} - x_k^{t-2})$$
$$z_k^t = \operatorname*{argmin}_{z \in \mathcal{Z}} h(z) + \langle -\mathcal{A}\tilde{u}_k^t, z \rangle + q_k^t U(z_k^{t-1}, z)$$
$$x_k^t = \operatorname*{argmin}_{x \in \mathcal{X}} \mu\nu(x) + \langle y_k + \mathcal{A}^\top z_k^t, x \rangle + \eta_k^t V(x_k^{t-1}, x) + p_k V(x_{k-1}, x)$$

   end for
   Set $x_k = x_k^{T_k}$, $z_k = z_k^{T_k}$, $\hat{x}_k = \sum_{t=1}^{T_k} x_k^t/T_k$, and $\hat{z}_k = \sum_{t=1}^{T_k} z_k^t/T_k$.
end for
Output $\overline{w}_N := (\overline{x}_N, \overline{y}_N, \overline{z}_N) := \left(\sum_{k=1}^N \beta_k\right)^{-1} \left(\sum_{k=1}^N \beta_k(\hat{x}_k, y_k, \hat{z}_k)\right).$

---

440   Extending the analysis of Algorithm 2.2, it is straightforward to observe that the
441 equivalent description of the PDS algorithm described in Algorithm 4.1 can be applied
442 to solve the more general saddle point problem (4.2). We will evaluate the accuracy
443 through the gap function defined by

444 (4.11)
$$Q(\hat{w}_k, w) := \left[\mu\nu(\hat{x}_k) + \langle \hat{x}_k, y + \mathcal{A}^\top z \rangle - \tilde{f}^*(y) - h(z)\right]$$
445
$$- \left[\mu\nu(x) + \langle x, y_k + \mathcal{A}^\top \hat{z}_k \rangle - \tilde{f}^*(y_k) - h(\hat{z}_k)\right],$$

446 in which $\hat{w}_k := (\hat{x}_k, y_k, \hat{z}_k)$ and $w := (x, y, z)$. We have the following proposition
447 describing the convergence result, whose proof will be presented in Section 5.

448   PROPOSITION 4.1. *Suppose that the conditions* (2.10)–(2.12) *(where $\|\mathcal{A}\|$ is the*
449 *norm induced by $\|\cdot\|$ and $|\cdot|$) described in Proposition 2.1 hold for Algorithm 4.1.*
450 *For all $w \in \mathcal{X} \times \mathcal{Y} \times \mathcal{Z}$ we have the following convergence result when Applying*
451 *Algorithm 4.1 to solve problem* (4.2):

452
453 (4.12) $$Q(\overline{w}_N, w) \leq \left(\sum_{k=1}^N \beta_k\right)^{-1} \beta_1 \left[\tfrac{q_1^1}{T_1} U(z_0, z) + \left(\tfrac{\eta_1^1}{T_1} + p_1\right) V(x_0, x)\right].$$

454 *Moreover, in the special case of problem* (4.3) *(i.e., if $\mathcal{Z}$ is a vector space with Euclid-*
455 *ean norm $|\cdot| = \|\cdot\|_2$, the prox-function $U(\cdot, \cdot) = \|\cdot - \cdot\|_2^2/2$, and $h(z) = \langle b, z \rangle$), then*

we can set the initial value $z_0 = 0$ in Algorithm 4.1 and obtain

(4.13) $\quad f(\overline{x}_N) - f(x^*) \leq \left(\sum_{k=1}^{N} \beta_k \right)^{-1} \beta_1 \left(\frac{\eta_1^1}{T_1} + p_1 \right) V(x_0, x^*)$ and

(4.14) $\quad \|\mathcal{A}\overline{x}_N - b\|_2 \leq \left(\sum_{k=1}^{N} \beta_k \right)^{-1} \beta_1 \left[\frac{q_1^1}{2T_1} (\|z^*\|_2 + 1)^2 + \left(\frac{\eta_1^1}{T_1} + p_1 \right) V(x_0, x^*) \right].$

We provide a set of example parameters for Algorithm 4.1 in the theorem below. The proof of the theorem below is similar to that of Theorem 2.2 and thus is skipped.

THEOREM 4.2. *Suppose that the parameters of Algorithm 4.1 are set to the ones described in Theorem 2.2. Applying Algorithm 4.1 to solve problem (4.1) we have*

$$Q(\overline{w}_N, w) \leq \min \left\{ \frac{2}{N^2}, \lambda^{N-\Delta} \right\} \left[\frac{\tilde{L}}{2R^2} U(z_0, z^*) + 4\tilde{L}V(x_0, x^*) \right].$$

Specially, if R is set to (2.17), then we have the following complexity results for Algorithm 4.1 to obtain an ϵ-solution of problem (4.1) (here $\mathcal{O}(1)$ is a constant that is independent of N, $\|\mathcal{A}\|$ and ϵ):

a) If $\mu = 0$, then the PDS algorithm can obtain an ϵ-solution of problem (4.1) after at most $\mathcal{O}(1) \left(\sqrt{\tilde{L}V(x_0, x^*)/\varepsilon} \right)$ number of gradient evaluations and $\mathcal{O}(1) \left(\sqrt{\tilde{L}V(x_0, x^*)/\varepsilon} + \|\mathcal{A}\|/\varepsilon \right)$ number of operator computations (involving $\mathcal{A}$ and $\mathcal{A}^\top$).

b) If $\mu > 0$, then the PDS algorithm can obtain an ϵ-solution of problem (4.1) after at most $\mathcal{O}(1) \left(1 + \sqrt{2\tilde{L}/\mu} \log(\tilde{L}V(x_0, x^*)/\varepsilon) \right)$ number of gradient evaluations and $\mathcal{O}(1) \left(1 + \sqrt{2\tilde{L}/\mu} \log(\tilde{L}V(x_0, x^*)/\varepsilon) + \|\mathcal{A}\|(1 + 1/\sqrt{\varepsilon}) \right)$ number of operator computations (involving $\mathcal{A}$ and $\mathcal{A}^\top$).

To the best of our knowledge, this is the first time that a primal-dual type algorithm achieves the above complexity results for solving bilinearly coupled saddle point problems. The only other method that achieves such complexity results is presented in [15], which requires to utilize a smoothing technique developed in [23]. However, the smoothing technique requires that either set $\mathcal{X}$ or $\mathcal{Z}$ is compact and hence is not applicable to the constrained optimization problem (4.3). Our proposed algorithm is the first algorithm in the literature that is able to solve the linearly constrained optimization problem (4.3) with the above complexities.

5. Convergence analysis. Our goal in this section is to complete the remaining proofs in Sections 2–4, including the proofs of Propositions 2.1, 3.1, and 4.1. We first need the following proposition that describes an estimate of the gap function in (4.11):

PROPOSITION 5.1. *Suppose that $\hat{x}_k = \sum_{t=1}^{T_k} x_k^t / T_k$ and $\hat{z}_k = \sum_{t=1}^{T_k} z_k^t / T_k$, where the iterates $\{x_k^t\}_{t=1}^{T_k}$ and $\{z_k^t\}_{t=1}^{T_k}$ are defined by*

(5.1) $\quad z_k^t = \underset{z \in \mathcal{Z}}{\operatorname{argmin}} \, h(z) + \langle -\mathcal{A}\tilde{u}_k^t, z \rangle + q_k^t U(z_k^{t-1}, z)$ and

(5.2) $\quad x_k^t = \underset{x \in \mathcal{X}}{\operatorname{argmin}} \, \mu\nu(x) + \langle v_k + \mathcal{A}^\top z_k^t, x \rangle + \eta_k^t V(x_k^{t-1}, x) + p_k V(x_{k-1}, x)$

493   *respectively. Letting $\hat{w}_k := (\hat{x}_k, y_k, \hat{z}_k)$ we have*

494
495   (5.3) $\sum_{k=1}^{N} \beta_k Q(\hat{w}_k, w) + A + B \leq C + D, \ \forall w := (x, y, z) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{Z}$

496   *where*

497   (5.4)
$$A := \sum_{k=1}^{N} \beta_k \left[-\langle \hat{x}_k, y \rangle + \langle x, y_k \rangle + \langle v_k, \hat{x}_k - x \rangle - \tilde{f}^*(v_k) + \tilde{f}^*(y) \right.$$
$$\left. + \frac{p_k}{T_k} \sum_{t=1}^{T_k} V(x_{k-1}, x_k^t) \right],$$

498   (5.5)
$$B := \sum_{k=1}^{N} \frac{\beta_k}{T_k} \sum_{t=1}^{T_k} \left[\langle \mathcal{A}^\top z_k^t - \mathcal{A}^\top z, x_k^t - \tilde{u}_k^t \rangle \right.$$
$$\left. + q_k^t U(z_k^{t-1}, z_k^t) + \eta_k^t V(x_k^{t-1}, x_k^t) \right],$$

499   (5.6) $C := \sum_{k=1}^{N} \frac{\beta_k}{T_k} \sum_{t=1}^{T_k} \left[q_k^t U(z_k^{t-1}, z) - q_k^t U(z_k^t, z) \right],$

500
501   $D := \sum_{k=1}^{N} \frac{\beta_k}{T_k} \sum_{t=1}^{T_k} \left[\eta_k^t V(x_k^{t-1}, x) - (\mu + \eta_k^t + p_k) V(x_k^t, x) + p_k V(x_{k-1}, x) \right].$

502   *Proof.* By the optimality conditions of (5.1) and (5.2) and definitions of U and
503   V in (4.4) and (4.5) respectively, we have

504   $\langle h'(z_k^t) - \mathcal{A}\tilde{u}_k^t + q_k^t \zeta'(z_k^t) - q_k^t \zeta'(z_k^{t-1}), z_k^t - z \rangle \leq 0, \ \forall z \in \mathcal{Z}, \ \text{and}$

505
506   $\langle (\mu + \eta_k^t + p_k) \nu'(x_k^t) + v_k + \mathcal{A}^\top z_k^t - \eta_k^t \nu'(x_k^{t-1}) - p_k \nu'(x_{k-1}), x_k^t - x \rangle \leq 0, \ \forall x \in \mathcal{X}.$

507   In view of the above results, the convexity of h and ν, and the definitions of U and
508   V we obtain the following two relations: for any $z \in \mathcal{Z}$ and $x \in \mathcal{X}$,

509   $h(z_k^t) - h(z) + \langle -\mathcal{A}\tilde{u}_k^t, z_k^t - z \rangle + q_k^t U(z_k^{t-1}, z_k^t) + q_k^t U(z_k^t, z) \leq q_k^t U(z_k^{t-1}, z), \ \text{and}$

510   $\langle v_k + \mathcal{A}^\top z_k^t, x_k^t - x \rangle + \mu \nu(x_k^t) - \mu \nu(x)$
$+ \eta_k^t V(x_k^{t-1}, x_k^t) + (\mu + \eta_k^t + p_k) V(x_k^t, x) + p_k V(x_{k-1}, x_k^t)$
511   $\leq \eta_k^t V(x_k^{t-1}, x) + p_k V(x_{k-1}, x).$

512   Summing up the two relations above, while noting that

513   $\langle -\mathcal{A}\tilde{u}_k^t, z_k^t - z \rangle + \langle v_k + \mathcal{A}^\top z_k^t, x_k^t - x \rangle$
514
515   $= \langle \mathcal{A}^\top z_k^t - \mathcal{A}^\top z, x_k^t - \tilde{u}_k^t \rangle + \langle \mathcal{A}^\top z, x_k^t \rangle - \langle v_k + \mathcal{A}^\top z_k^t, x \rangle + \langle v_k, x_k^t \rangle,$

516   we have

517   $\langle \mathcal{A}^\top z_k^t - \mathcal{A}^\top z, x_k^t - \tilde{u}_k^t \rangle + \langle \mathcal{A}^\top z, x_k^t \rangle - \langle v_k + \mathcal{A}^\top z_k^t, x \rangle + \langle v_k, x_k^t \rangle$
518   $+ h(z_k^t) - h(z) + q_k^t U(z_k^{t-1}, z_k^t) + q_k^t U(z_k^t, z)$
519   $+ \mu \nu(x_k^t) - \mu \nu(x) + \eta_k^t V(x_k^{t-1}, x_k^t) + (\mu + \eta_k^t + p_k) V(x_k^t, x) + p_k V(x_{k-1}, x_k^t)$

520
521   $\leq q_k^t U(z_k^{t-1}, z) + \eta_k^t V(x_k^{t-1}, x) + p_k V(x_{k-1}, x).$

522   Summing from $t = 1, \ldots, T_k$ and noting the definitions of $\hat{x}_k$ and $\hat{z}_k$ and the convexity

of functions h and ν we have

$$T_k \left[\langle \mathcal{A}^\top z, \hat{x}_k \rangle - \langle v_k + \mathcal{A}^\top \hat{z}_k, x \rangle + \langle v_k, \hat{x}_k \rangle + h(\hat{z}_k) - h(z) + \mu\nu(\hat{x}_k) - \mu\nu(x) \right]$$

$$+ \sum_{t=1}^{T_k} p_k \left[V(x_{k-1}, x_k^t) + \langle \mathcal{A}^\top z_k^t - \mathcal{A}^\top z, x_k^t - \tilde{u}_k^t \rangle + q_k^t U(z_k^{t-1}, z_k^t) + \eta_k^t V(x_k^{t-1}, x_k^t) \right]$$

$$\leq \sum_{t=1}^{T_k} \left[q_k^t U(z_k^{t-1}, z) - q_k^t U(z_k^t, z) \right.$$

$$\left. + \eta_k^t V(x_k^{t-1}, x) - (\mu + \eta_k^t + p_k) V(x_k^t, x) + p_k V(x_{k-1}, x) \right].$$

We conclude (5.3) by multiplying the above relation by β_k/T_k, noting the definition of the gap function $Q(\hat{w}_k, w)$ in (4.11) and summing the resulting relation from $k = 1, \ldots, N$. $\qquad\square$

It should be noted that the above result is valid for any choices of v_k, $\tilde{u}_k^t$ and x_{k-1} used in the descriptions of z_k^t and x_k^t in (5.1) and (5.2) respectively. Now we are ready to establish tight bounds of A, B, C, and D in (5.3) through a series of technical lemmas. To provide a tight estimate of A, we start with the case when v_k is the exact solution to the dual prox-mapping subproblem (4.10).

LEMMA 5.2. *Suppose that $v_k = y_k$ where y_k is defined by (4.9) and (4.10). If*

$$(5.7) \qquad \tau_1 = 0, \beta_k \tau_k \leq \beta_{k-1}(\tau_{k-1} + 1), \beta_{k-1} = \beta_k \lambda_k, \ \text{and} \ \tilde{L}\lambda_k \leq p_{k-1}\tau_k, \ \forall k \geq 2,$$

then we have

$$A \geq -\beta_N \langle x_{N-1} - \hat{x}_N, y_N - y \rangle + \tfrac{\beta_N p_N}{2} \|x_{N-1} - \hat{x}_N\|^2 + \beta_N(\tau_N + 1) W(y_N, y).$$

Proof. By the optimality condition of (4.10) and the definition of W in (4.6)

$$\langle -\tilde{x}_k + (1 + \tau_k)(\tilde{f}^*)'(y_k) - \tau_k(\tilde{f}^*)'(y_{k-1}), y_k - y \rangle \leq 0, \ \forall y \in \mathcal{Y},$$

which together with the convexity of $\tilde{f}^*$ and the definition of W in (4.6) implies

$$(5.8) \qquad \begin{aligned} &\langle -\tilde{x}_k, y_k - y \rangle + \tilde{f}^*(y_k) - \tilde{f}^*(y) + \tau_k W(y_{k-1}, y_k) + (\tau_k + 1) W(y_k, y) \\ &\leq \tau_k W(y_{k-1}, y), \ \forall y \in \mathcal{Y}. \end{aligned}$$

Combining the above relation with A in (5.4) and noting that $v_k = y_k$ we have

$$A \geq \sum_{k=1}^N \beta_k \left[(\tau_k + 1) W(y_k, y) - \tau_k W(y_{k-1}, y) + \tau_k W(y_{k-1}, y_k) \right.$$

$$\left. - \langle \hat{x}_k, y \rangle + \langle y_k, \hat{x}_k \rangle + \langle -\tilde{x}_k, y_k - y \rangle + \tfrac{p_k}{T_k} \sum_{t=1}^{T_k} V(x_{k-1}, x_k^t) \right]$$

From the definition of $\tilde{x}_k$ we can observe that

$$- \langle \hat{x}_k, y \rangle + \langle y_k, \hat{x}_k \rangle + \langle -\tilde{x}_k, y_k - y \rangle = -\langle x_{k-1} + \lambda_k(\hat{x}_{k-1} - x_{k-2}) - \hat{x}_k, y_k - y \rangle$$

$$= \lambda_k \langle x_{k-2} - \hat{x}_{k-1}, y_{k-1} - y \rangle - \langle x_{k-1} - \hat{x}_k, y_k - y \rangle + \lambda_k \langle x_{k-2} - \hat{x}_{k-1}, y_k - y_{k-1} \rangle.$$

Also, by the definition of $\hat{x}_k$ and the fact that V is lower bounded in (4.7) we have

$$\tfrac{1}{T_k} \sum_{t=1}^{T_k} p_k V(x_{k-1}, x_k^t) \geq p_k V(x_{k-1}, \hat{x}_k) \geq \tfrac{p_k}{2} \|x_{k-1} - \hat{x}_k\|^2.$$

Applying the above two observations, the bound of W in (4.8), the parameter assumption (5.7), and recalling that $x_{-1} = \hat{x}_0$ in Algorithm 4.1 we have

$$A \geq \sum_{k=1}^{N} \left[\beta_k(\tau_k + 1)W(y_k, y) - \beta_k \tau_k W(y_{k-1}, y) \right.$$

$$+ \beta_k \lambda_k \langle x_{k-2} - \hat{x}_{k-1}, y_{k-1} - y \rangle - \beta_k \langle x_{k-1} - \hat{x}_k, y_k - y \rangle$$

$$\left. + \beta_k \lambda_k \langle x_{k-2} - \hat{x}_{k-1}, y_k - y_{k-1} \rangle + \tfrac{\beta_k p_k}{2} \|x_{k-1} - \hat{x}_k\|^2 + \tfrac{\beta_k \tau_k}{2\tilde{L}} \|y_{k-1} - y_k\|_*^2 \right]$$

$$\geq \beta_N(\tau_N + 1)W(y_N, y) - \beta_N \langle x_{N-1} - \hat{x}_N, y_N - y \rangle + \tfrac{\beta_N p_N}{2} \|x_{N-1} - \hat{x}_N\|^2$$

$$+ \sum_{k=2}^{N} \left[\beta_k \lambda_k \|x_{k-2} - \hat{x}_{k-1}\| \|y_k - y_{k-1}\|_* \right.$$

$$\left. + \tfrac{\beta_{k-1} p_{k-1}}{2} \|x_{k-2} - \hat{x}_{k-1}\|^2 + \tfrac{\beta_k \tau_k}{2\tilde{L}} \|y_{k-1} - y_k\|_*^2 \right]$$

$$\geq \beta_N(\tau_N + 1)W(y_N, y) - \beta_N \langle x_{N-1} - \hat{x}_N, y_N - y \rangle + \tfrac{\beta_N p_N}{2} \|x_{N-1} - \hat{x}_N\|^2,$$

where the last inequality follows from the simple relation $a^2 + b^2 \geq 2ab$ and

$$(\beta_k \lambda_k)^2 - \beta_{k-1} p_{k-1}(\beta_k \tau_k/\tilde{L}) = (\beta_k^2 \lambda_k/\tilde{L})(\tilde{L}\lambda_k - p_{k-1}\tau_k) \leq 0. \qquad \square$$

There are cases when v_k is not equal to y_k, for example, in Algorithm 3.2. The following lemma provide a tight bound for A under this case.

LEMMA 5.3. *Suppose that y_k is defined by (4.9) and (4.10) and $v_k = v_k(y_k, \xi_k)$ is an unbiased estimator of y_k with respect to random variable ξ_k such that $\mathbb{E}[\|v_k - y_k\|_*] = 0$ and $\mathbb{E}[\|v_k - y_k\|_*^2] \leq \sigma_k^2$, where $\|\cdot\|_*$ is the dual norm of $\|\cdot\|$. If the parameters satisfy*

$$\tau_1 = 0, \beta_k \tau_k \leq \beta_{k-1}(\tau_{k-1} + 1), \beta_{k-1} = \beta_k \lambda_k, \text{ and } 2\tilde{L}\lambda_k \leq p_{k-1}\tau_k, \ \forall k \geq 2,$$

then we have

$$\mathbb{E}[A] \geq \mathbb{E}\left[-\beta_N \langle x_{N-1} - \hat{x}_N, y_N - y \rangle + \tfrac{\beta_N p_N}{4} \|x_{N-1} - \hat{x}_N\|^2 - \sum_{k=1}^{N} \tfrac{\beta_k \sigma_k^2}{p_k} \right.$$

$$\left. + \beta_N(\tau_N + 1)W(y_N, y) \right].$$

Proof. In view of the optimality condition described previously in (5.8) and the definition of A in (5.4) we have $A \geq A_1 + A_2$ where

$$A_1 := \sum_{k=1}^{N} \beta_k \left[-\langle \hat{x}_k, y \rangle + \langle y_k, \hat{x}_k \rangle + \langle -\tilde{x}_k, y_k - y \rangle + \tfrac{p_k}{2T_k} \sum_{t=1}^{T_k} V(x_{k-1}, x_k^t) \right.$$

$$\left. + \tau_k W(y_{k-1}, y_k) + (\tau_k + 1)W(y_k, y) - \tau_k W(y_{k-1}, y) \right] \text{ and}$$

$$A_2 := \sum_{k=1}^{N} \tfrac{\beta_k}{T_k} \sum_{t=1}^{T_k} \left[\tfrac{p_k}{2} V(x_{k-1}, x_k^t) + \langle v_k - y_k, x_k^t - x_{k-1} \rangle + \langle v_k - y_k, x_{k-1} - x \rangle \right].$$

Applying the same argument as in Lemma 5.2 to A_1, we have

$$A_1 \geq -\beta_N \langle x_{N-1} - \hat{x}_N, y_N - y \rangle + \tfrac{\beta_N p_N}{4} \|x_{N-1} - \hat{x}_N\|^2 + \beta_N(\tau_N + 1)W(y_N, y).$$

To finish the proof it suffices to show that $\mathbb{E}[A_2] \geq -\sum_{k=1}^{N} \beta_k \sigma_k^2/p_k$. Noting from the definition of v_k that $\mathbb{E}_{\xi_1,\ldots,\xi_{k-1}}[\langle v_k - y_k, x_{k-1} - x \rangle] = 0$ and applying the bound of V in (4.7) and Cauchy-Schwartz inequality to the description of A_2 we have

$$\mathbb{E}[A_2] \geq \mathbb{E}\left[\sum_{k=1}^{N} \tfrac{\beta_k}{T_k} \sum_{t=1}^{T_k} \left(\tfrac{p_k}{4} \|x_{k-1} - x_k^t\|^2 + \|v_k - y_k\|_* \|x_k^t - x_{k-1}\| \right) \right]$$

$$\geq \mathbb{E}\left[\sum_{k=1}^{N} \tfrac{\beta_k}{T_k} \sum_{t=1}^{T_k} -\tfrac{1}{p_k} \|v_k - y_k\|_*^2 \right] = -\sum_{k=1}^{N} \tfrac{\beta_k \sigma_k^2}{p_k}. \qquad \square$$

600   We continue with the estimates of B, C, and D in the following two lemmas.

601   LEMMA 5.4. *Suppose that*

602   (5.9) $\alpha_k^t = 1$, $\|\mathcal{A}\|^2 \leq \eta_k^{t-1} q_k^t$, $\forall t \geq 2, k \geq 1$, *and*

603   (5.10) $\beta_k T_{k-1} \alpha_k^1 = \beta_{k-1} T_k$, $\alpha_k^1 \|\mathcal{A}\|^2 \leq \eta_{k-1}^{T_{k-1}} q_k^1$, $\forall k \geq 2$,
604

605   *where $\|\mathcal{A}\|$ is the norm induced by norms $\|\cdot\|$ and $|\cdot|$ introduced in* (4.7). *Then*

606     $$B \geq \tfrac{\beta_N}{T_N} \|\mathcal{A}\| \|z_N - z\| \|x_N - x_N^{T_N - 1}\| + \tfrac{\beta_N \eta_N^{T_N}}{2 T_N} \|x_N^{T_N - 1} - x_N^{T_N}\|^2.$$
607

608   *Proof.* By the definition of $\tilde{u}_k^t$ in (2.6) and bounds of U and V in (4.7) we have

609     $$\sum_{t=1}^{T_k} \left[\langle \mathcal{A}^\top z_k^t - \mathcal{A}^\top z, x_k^t - \tilde{u}_k^t \rangle + q_k^t U(z_k^{t-1}, z_k^t) + \eta_k^t V(x_k^{t-1}, x_k^t) \right]$$

610     $$\geq \sum_{t=1}^{T_k} \left[-\alpha_k^t \langle \mathcal{A}^\top z_k^{t-1} - \mathcal{A}^\top z, x_k^{t-1} - x_k^{t-2} \rangle + \langle \mathcal{A}^\top z_k^t - \mathcal{A}^\top z, x_k^t - x_k^{t-1} \rangle \right.$$

611     $$\left. + \tfrac{q_k^t}{2} \|z_k^{t-1} - z_k^t\|^2 + \tfrac{\eta_k^t}{2} \|x_k^{t-1} - x_k^t\|^2 - \alpha_k^t \langle \mathcal{A}^\top z_k^t - \mathcal{A}^\top z_k^{t-1}, x_k^{t-1} - x_k^{t-2} \rangle \right].$$
612

613   Also observe that $\alpha_k^t = 1$ for all $t \geq 2$ we have

614     $$\sum_{t=1}^{T_k} - \alpha_k^t \langle \mathcal{A}^\top z_k^{t-1} - \mathcal{A}^\top z, x_k^{t-1} - x_k^{t-2} \rangle + \langle \mathcal{A}^\top z_k^t - \mathcal{A}^\top z, x_k^t - x_k^{t-1} \rangle$$

615     $$= - \alpha_k^1 \langle \mathcal{A}^\top z_k^0 - \mathcal{A}^\top z, x_k^0 - x_k^{-1} \rangle + \langle \mathcal{A}^\top z_k^{T_k} - \mathcal{A}^\top z, x_k^{T_k} - x_k^{T_k - 1} \rangle.$$
616

617   In addition, in view of the assumptions (5.9) and (5.10) we have

618     $$\sum_{t=1}^{T_k} \left[\tfrac{q_k^t}{2} \|z_k^{t-1} - z_k^t\|^2 + \tfrac{\eta_k^t}{2} \|x_k^{t-1} - x_k^t\|^2 - \alpha_k^t \langle \mathcal{A}^\top z_k^t - \mathcal{A}^\top z_k^{t-1}, x_k^{t-1} - x_k^{t-2} \rangle \right]$$

619     $$= \tfrac{q_k^1}{2} \|z_k^0 - z_k^1\|^2 + \tfrac{\eta_k^{T_k}}{2} \|x_k^{T_k - 1} - x_k^{T_k}\|^2 - \alpha_k^1 \langle z_k^1 - z_k^0, \mathcal{A}(x_k^0 - x_k^{-1}) \rangle$$

620     $$+ \sum_{t=2}^{T_k} \left[\tfrac{q_k^t}{2} \|z_k^{t-1} - z_k^t\|^2 + \tfrac{\eta_k^{t-1}}{2} \|x_k^{t-2} - x_k^{t-1}\|^2 - \langle z_k^t - z_k^{t-1}, \mathcal{A}(x_k^{t-1} - x_k^{t-2}) \rangle \right]$$

621     $$\geq \tfrac{q_k^1}{2} \|z_k^0 - z_k^1\|^2 + \tfrac{\eta_k^{T_k}}{2} \|x_k^{T_k - 1} - x_k^{T_k}\|^2 - \alpha_k^1 \|\mathcal{A}\| \|z_k^1 - z_k^0\| \|x_k^0 - x_k^{-1}\|.$$
622

623   Applying the above three relations to the definition of B in (5.5), recalling the defi-
624   nitions of x_k^0, z_k^0, x_k^{-1} in Algorithm 4.1, and using the assumption (5.10) we have

625     $$B \geq \sum_{k=1}^{N} \tfrac{\beta_k}{T_k} \left[- \alpha_k^1 \langle \mathcal{A}^\top z_k^0 - \mathcal{A}^\top z, x_k^0 - x_k^{-1} \rangle + \langle \mathcal{A}^\top z_{k+1}^0 - \mathcal{A}^\top z, x_{k+1}^0 - x_{k+1}^{-1} \rangle \right.$$

626     $$\left. + \tfrac{q_k^1}{2} \|z_k^0 - z_k^1\|^2 + \tfrac{\eta_k^{T_k}}{2} \|x_k^{T_k - 1} - x_k^{T_k}\|^2 - \alpha_k^1 \|\mathcal{A}\| \|z_k^1 - z_k^0\| \|x_k^0 - x_k^{-1}\| \right]$$

627     $$= \tfrac{\beta_N}{T_N} \langle \mathcal{A}^\top z_N - \mathcal{A}^\top z, x_N - x_N^{T_N - 1} \rangle$$

628     $$+ \tfrac{\beta_1 q_1^1}{2 T_k} \|z_1^0 - z_1^1\|^2 + \tfrac{\beta_N \eta_N^{T_N}}{2 T_N} \|x_N^{T_N - 1} - x_N^{T_N}\|^2$$

629     $$+ \sum_{k=2}^{N} \left[\tfrac{\beta_k q_k^1}{2 T_k} \|z_k^0 - z_k^1\|^2 + \tfrac{\beta_{k-1} \eta_{k-1}^{T_{k-1}}}{2 T_{k-1}} \|x_k^{-1} - x_k^0\|^2 - \tfrac{\beta_k \alpha_k^1}{T_k} \|\mathcal{A}\| \|z_k^1 - z_k^0\| \|x_k^0 - x_k^{-1}\| \right]$$

630     $$\geq \tfrac{\beta_N}{T_N} \|\mathcal{A}\| \|z_N - z\| \|x_N - x_N^{T_N - 1}\| + \tfrac{\beta_N \eta_N^{T_N}}{2 T_N} \|x_N^{T_N - 1} - x_N^{T_N}\|^2.$$
631

Here in the last inequality we use the following result from the assumption (5.10):

$$(\beta_k \alpha_k^1 \|\mathcal{A}\|/T_k)^2 - (\beta_k q_k^1/T_k)(\beta_{k-1}\eta_{k-1}^{T_{k-1}}/T_{k-1})$$

$$=(\beta_k/T_k)^2 \alpha_k^1(\alpha_k^1\|\mathcal{A}\|^2 - q_k^1\eta_{k-1}^{T_{k-1}}) \geq 0. \qquad \square$$

LEMMA 5.5. *If*

(5.11) $\qquad q_k^t \leq q_k^{t-1}, \ \forall t \geq 2, k \geq 1 \ and \ \beta_k T_{k-1} q_k^1 \leq \beta_{k-1} T_k q_{k-1}^{T_{k-1}}, \ \forall k \geq 2,$

then C in (5.6) can be bounded by

$$C \leq \frac{\beta_1 q_1^1}{T_1}U(z_0, z) - \frac{\beta_N q_N^{T_N}}{T_N}U(z_N, z).$$

Also, if

(5.12) $\qquad \begin{aligned}&\eta_k^t \leq \mu + \eta_k^{t-1} + p_k, \ \forall t \geq 2, k \geq 1 \ and \\ &\beta_k T_{k-1}(\eta_k^1 + p_k T_k) \leq \beta_{k-1} T_k(\mu + \eta_{k-1}^{T_{k-1}} + p_{k-1}), \ \forall k \geq 2,\end{aligned}$

then D in (5.6) can be bounded by

$$D \leq \frac{\beta_1}{T_1}(\eta_1^1 + p_1 T_1)V(x_0, x) - \frac{\beta_N}{T_N}(\mu + \eta_N^{T_N} + p_N)V(x_N, x).$$

Proof. In view of (5.11) and the fact that $z_k^0 = z_{k-1}$ and $z_k = z_k^{T_k}$, we have:

$$C \leq \sum_{k=1}^{N}\left[\frac{\beta_k q_k^1}{T_k}U(z_{k-1}, z) - \frac{\beta_k q_k^{T_k}}{T_k}U(z_k, z)\right] \leq \frac{\beta_1 q_1^1}{T_1}U(z_0, z) - \frac{\beta_N q_N^{T_N}}{T_N}U(z_N, z).$$

Similarly, by (5.12) and noting that $x_k^0 = x_{k-1}$ and $x_k = x_k^{T_k}$ we have

$$D \leq \sum_{k=1}^{N}\frac{\beta_k}{T_k}\left[\eta_k^1 V(x_k^0, x) - (\mu + \eta_k^{T_k} + p_k)V(x_k^{T_k}, x) + p_k T_k V(x_{k-1}, x)\right]$$

$$=\sum_{k=1}^{N}\left[\frac{\beta_k}{T_k}(\eta_k^1 + p_k T_k)V(x_{k-1}, x) - \frac{\beta_k}{T_k}(\mu + \eta_k^{T_k} + p_k)V(x_k, x)\right]$$

$$\leq \frac{\beta_1}{T_1}(\eta_1^1 + p_1 T_1)V(x_0, x) - \frac{\beta_N}{T_N}(\mu + \eta_N^{T_N} + p_N)V(x_N, x). \qquad \square$$

With Lemmas 5.2, 5.4, and 5.5 and Proposition 5.1, we are ready to prove Proposition 4.1.

Proof of Proposition 4.1. Note that the conditions (2.10) and (2.11) for parameters in Algorithm 4.1 are exactly the parameter assumptions (5.7), (5.9) – (5.12), in Lemmas 5.2, 5.4, and 5.5. In view of the definition of $\overline{w}_N$, the convexity of Q, Proposition 5.1 and Lemmas 5.2, 5.4, and 5.5, we have

$$\left(\sum_{k=1}^{N}\beta_k\right)Q(\overline{w}_N, w) \leq \sum_{k=1}^{N}\beta_k Q(\hat{w}_k, w)$$

$$\leq \beta_N \langle x_{N-1} - \hat{x}_N, y_N - y\rangle - \frac{\beta_N p_N}{2}\|x_{N-1} - \hat{x}_N\|^2 - \beta_N(\tau_N + 1)W(y_N, y)$$

$$- \frac{\beta_N}{T_N}\|\mathcal{A}\|\|z_N - z\|\|x_N - x_N^{T_N-1}\| - \frac{\beta_N \eta_N^{T_N}}{2T_N}\|x_N^{T_N-1} - x_N^{T_N}\|^2$$

$$+ \frac{\beta_1 q_1^1}{T_1}U(z_0, z) - \frac{\beta_N q_N^{T_N}}{T_N}U(z_N, z)$$

$$+ \frac{\beta_1}{T_1}(\eta_1^1 + p_1 T_1)V(x_0, x) - \frac{\beta_N}{T_N}(\mu + \eta_N^{T_N} + p_N)V(x_N, x).$$

We conclude (4.12) from the above inequality by noting from (2.12), (4.7), and (4.8) that

$$
\beta_N \langle x_{N-1} - \hat{x}_N, y_N - y \rangle - \tfrac{\beta_N p_N}{2} \|x_{N-1} - \hat{x}_N\|^2 - \beta_N(\tau_N + 1) W(y_N, y)
$$
$$
\leq \tfrac{2\beta_N}{p_N} \|y_N - y\|^2 - \tfrac{\beta_N(\tau_N+1)}{2\tilde{L}} \|y_N - y\|^2 \leq 0, \text{ and}
$$

$$
- \tfrac{\beta_N}{T_N} \|\mathcal{A}\| |z_N - z| \|x_N - x_N^{T_N-1}\| - \tfrac{\beta_N \eta_N^{T_N}}{2T_N} \|x_N^{T_N-1} - x_N^{T_N}\|^2 - \tfrac{\beta_N q_N^{T_N}}{T_N} U(z_N, z)
$$
$$
\leq \tfrac{\beta_N q_N^{T_N}}{2T_N} |z_N - z|^2 - \tfrac{\beta_N q_N^{T_N}}{2T_N} |z_N - z|^2 = 0.
$$

In the special case when $\mathcal{Z}$ is a vector space with Euclidean norm $|\cdot| = \|\cdot\|_2$, the prox-function $U(\cdot, \cdot) = \|\cdot - \cdot\|_2^2/2$, and $h(z) = \langle b, z \rangle$, let us choose any saddle point $w^* = (x^*, y^*, z^*)$ that solves the SPP (4.2) and denote

$$
(5.13) \qquad \overline{w}_N^* := (x^*, y^*, \overline{z}_N^*) \text{ where } \overline{z}_N^* := \tfrac{\|z^*\|_2 + 1}{\|A\overline{x}_N - b\|_2}(A\overline{x}_N - b).
$$

Note that in such special case the SPP (4.2) reduces to the linear constrained optimization (4.3), in which x^* is an optimal solution, $y^* = \nabla \tilde{f}(x^*)$, and z^* is the optimal Lagrange multiplier associated with the equality constraints. Choosing $w = (x^*, \nabla \tilde{f}(\overline{x}_N), 0)$ and noting that $Ax^* = b$, $\tilde{f}(\overline{x}_N) = \langle \overline{x}_N, \nabla \tilde{f}(\overline{x}_N) \rangle - \tilde{f}^*(\nabla \tilde{f}(\overline{x}_N))$, and $\tilde{f}(x^*) \geq \langle x^*, \overline{y}_N \rangle - \tilde{f}^*(\overline{y}_N)$, by the definition of gap function Q in (4.11) we have

$$
Q(\overline{w}_N, w) = \mu\nu(\overline{x}_N) + \langle \overline{x}_N, \nabla \tilde{f}(\overline{x}_N) \rangle - \tilde{f}^*(\nabla \tilde{f}(\overline{x}_N)) - \mu\nu(x^*) - \langle x^*, \overline{y}_N \rangle + \tilde{f}^*(\overline{y}_N)
$$
$$
\geq \mu\nu(\overline{x}_N) + \tilde{f}(\overline{x}_N) - \mu\nu(x^*) - \tilde{f}(x^*) = f(\overline{x}_N) - f(x^*).
$$

Combing the above result with (4.12) we obtain (4.13) immediately. To prove (4.14) let us study the gap functions involving $\overline{w}_N$, w^*, and $\overline{w}_N^*$. Observe that

$$
Q(\overline{w}_N, w^*) = \left[\mu\nu(\overline{x}_N) + \langle \overline{x}_N, y^* + \mathcal{A}^\top z^* \rangle - \tilde{f}^*(y^*) - \langle b, z^* \rangle \right]
$$
$$
- \left[\mu\nu(x^*) + \langle x^*, \overline{y}_N + \mathcal{A}^\top \overline{z}_N \rangle - \tilde{f}^*(\overline{y}_N) - \langle b, \overline{z}_N \rangle \right]
$$
$$
\leq \mu\nu(\overline{x}_N) + \langle \overline{x}_N, y^* \rangle - \tilde{f}^*(y^*) - \mu\nu(x^*) - \langle x^*, \overline{y}_N \rangle + \tilde{f}^*(\overline{y}_N)
$$
$$
+ \|A\overline{x}_N - b\|_2 \|z^*\|_2.
$$

Here also observe that $Q(\overline{w}_N, w^*) \geq 0$ since w^* is a saddle point. By these observations and the definition of $\overline{z}_N^*$ in (5.13) we have

$$
Q(\overline{w}_N, \overline{w}_N^*) = \left[\mu\nu(\overline{x}_N) + \langle \overline{x}_N, y^* + \mathcal{A}^\top \overline{z}_N^* \rangle - \tilde{f}^*(y^*) - \langle b, \overline{z}_N^* \rangle \right]
$$
$$
- \left[\mu\nu(x^*) + \langle x^*, \overline{y}_N + \mathcal{A}^\top \overline{z}_N \rangle - \tilde{f}^*(\overline{y}_N) - \langle b, \overline{z}_N \rangle \right]
$$
$$
= \mu\nu(\overline{x}_N) + \langle \overline{x}_N, y^* \rangle - \tilde{f}^*(y^*) - \mu\nu(x^*) - \langle x^*, \overline{y}_N \rangle + \tilde{f}^*(\overline{y}_N)
$$
$$
+ (\|z^*\|_2 + 1)\|A\overline{x}_N - b\|_2
$$
$$
\geq Q(\overline{w}_N, w^*) + \|A\overline{x}_N - b\|_2 \geq \|A\overline{x}_N - b\|_2.
$$

The above result, together with the bound of Q in (4.12), and the facts that $y^* = \nabla \tilde{f}(x^*)$, $z_0 = 0$, and $\|z_0 - \overline{z}_N^*\|_2^2 = \|\overline{z}_N^*\|_2^2 = (\|z^*\|_2 + 1)^2$, yield the estimate (4.14). $\square$

Note that Proposition 2.1 is in fact a special case of Proposition 4.1. Indeed, since Algorithm 2.1 is the agent view of Algorithm 2.2, which is a special case of Algorithm

4.1 (when $h(z) \equiv 0$, $\mathcal{Z} = \mathbb{R}^{md}$, $z_0 = 0$, $\mathcal{X}$, f, and $\mathcal{A}$ are defined in (1.2), and all norms are Euclidean). Therefore, the results in Proposition 2.1 are immediately implied from Proposition 4.1.

Applying the results in Lemmas 5.3, 5.4, and 5.5 to Proposition 5.1 we can also prove Proposition 3.1. The proof is similar to that of Proposition 4.1 above.

Proof of Proposition 3.1. Denoting $\overline{w}_N := \left(\sum_{k=1}^{N} \beta_k\right)^{-1} \left(\sum_{k=1}^{N} \beta_k \hat{w}_k\right)$, let us study the gap function Q defined in (4.11) in which $h(\cdot) \equiv 0$. Noting the convexity of Q, applying Proposition 5.1 (in which $U(\cdot, \cdot) = \| \cdot - \cdot \|_2^2/2$, $V(\cdot, \cdot) = \| \cdot - \cdot \|_2^2/2$, and all norms are Euclidean) and the bounds of A through D in Lemmas 5.3 (with $\sigma_k = \sigma/\sqrt{c_k}$ as specified in Algorithm 3.2), 5.4, and 5.5 (in which $z_0 = 0$ from the description of Algorithm 3.1), we have

$$\left(\sum_{k=1}^{N} \beta_k\right) \mathbb{E}[Q(\overline{w}_N, w)] \leq \sum_{k=1}^{N} \beta_k \mathbb{E}[Q(\hat{w}_k, w)]$$

$$\leq \mathbb{E}\left[\beta_N \langle x_{N-1} - \hat{x}_N, y_N - y \rangle - \frac{\beta_N p_N}{4} \|x_{N-1} - \hat{x}_N\|_2^2 + \sum_{k=1}^{N} \frac{\beta_k \sigma^2}{p_k c_k}\right.$$

$$- \beta_N (\tau_N + 1) W(y_N, y)$$

$$- \frac{\beta_N}{T_N} \|\mathcal{A}\| \|z_N - z\|_2 \|x_N - x_N^{T_N - 1}\|_2 - \frac{\beta_N \eta_N^{T_N}}{2 T_N} \|x_N^{T_N - 1} - x_N^{T_N}\|_2^2$$

$$+ \frac{\beta_1 q_1^1}{2 T_1} \|z\|_2^2 - \frac{\beta_N q_N^{T_N}}{2 T_N} \|z_N - z\|_2^2$$

$$\left. + \frac{\beta_1}{2 T_1} (\eta_1^1 + p_1 T_1) \|x_0 - x\|_2^2 - \frac{\beta_N}{2 T_N} (\mu + \eta_N^{T_N} + p_N) \|x_N - x\|_2^2 \right].$$

Here noting from (2.12), (3.3), and (4.8) we can simplify the above to

$$\left(\sum_{k=1}^{N} \beta_k\right) \mathbb{E}[Q(\overline{w}_N, w)] \leq \frac{\beta_1 q_1^1}{2 T_1} \|z\|_2^2 + \frac{\beta_1}{2 T_1} (\eta_1^1 + p_1 T_1) \|x_0 - x\|_2^2 + \sum_{k=1}^{N} \frac{\beta_k \sigma^2}{p_k c_k}.$$

The remainder of the proof is similar to that of Proposition 4.1 and hence is skipped. □

6. Numerical experiments. In this section, we demonstrate the advantages of our proposed PDS method through some preliminary numerical experiments and compare it with the state-of-the-art communication-efficient decentralized method, namely the decentralized communication sliding (DCS) method proposed in [14]. We consider a decentralized convex smooth optimization problem of unregularized logistic regression model over a dataset that is not linearly separable. In order to guarantee a fair comparison, all the implementation details described below are the same as suggested in [14]. In the linear constrained problem formulation (1.2) of the decentralized problem we set $\mathcal{A} = \mathcal{L} \times I_d$ from the graph Laplacian. For the underlying communication network, we use the Erhos-Renyi algorithm to generate three connected graphs with $m = 100$ nodes as shown in Figure 1. Note that nodes with different degrees are drawn in different colors, in particular, $G1$ has a maximum degree of $d_{max} = 4$, $G2$ has $d_{max} = 9$ and $G3$ has $d_{max} = 20$. We also use the same dataset as in [14], a real dataset "ijcnn1" from LIBSVM[1] and choose $20,000$ samples from this dataset as our problem instance data to train the decentralized logistic regression model. Since we have $m = 100$ nodes (or agents) in the decentralized network, we evenly split these $20,000$ samples over 100 nodes, and hence each network node has 200 samples.

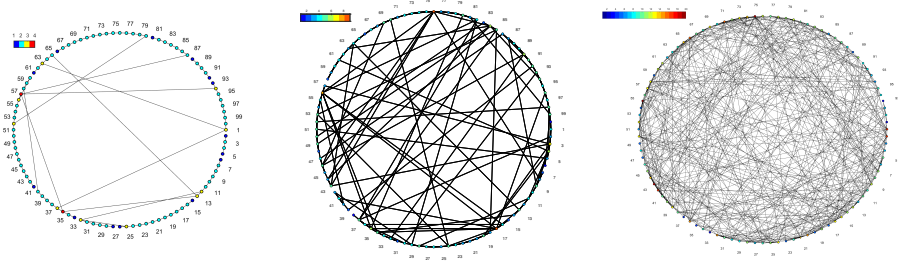[1]This real dataset can be downloaded from https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/.

FIG. 1. *The underlying decentralized networks G1, G2 and G3, from left to right. Nodes with different degrees are drawn in different colors.*

744   All agents in the network start with the same initial point $x_0 = 0$ and $y_0 = z_0 = 0$.
745 We then compare the performances of the proposed PDS/SPDS algorithms with DCS
746 (SDCS for stochastic case) proposed in [14] for solving (1.2) with f_i being the logistic
747 loss function. We show the required number of communication rounds and (stochastic)
748 gradient evaluations for the algorithms in comparison to obtain the same target loss.
749 In all problem instances, we use $\|\cdot\|_2$ norm in both primal and dual spaces. All
750 the experiments are programmed in Matlab 2020a and run on Clemson University's
751 Palmetto high-performance computing clusters (with 6 Intel Xeon Gold CLX 6248R
752 CPUs for a total of 168 cores).
753   In the deterministic case, for the DCS algorithm we use the parameter setting as
754 suggested in [14] including using a dynamic inner iteration limit as $\min\{10k, 5,000\}$
755 for possible performance improvement. For the PDS algorithm we use the parameter
756 setting as suggested in Theorem 2.2 for solving smooth and convex problems ($\mu = 0$).
757 In particular, when setting the inner iteration limit T_k we choose $R = 1/(2\sqrt{2})$ in
758 (2.13) . Note that we also try tuning the estimation of Lipschitz constant $\tilde{L}$ for the
759 best performance of the PDS algorithm.

TABLE 1
Comparison of the DCS and PDS algorithms in terms of reaching the same target loss

Algorithm	Graph	Target Loss	Achieved $\|\mathcal{A}x\|$	Com. rounds	Gradient evaluations
DCS	G1($d_{max} = 4$)	70	$4.94e - 04$	3,110	6,527,500
PDS	G1($d_{max} = 4$)	70	$8.95e - 02$	154	24
DCS	G2($d_{max} = 9$)	70	$3.22e - 14$	5,624	12,812,500
PDS	G2($d_{max} = 9$)	70	$2.08e - 01$	274	25
DCS	G3($d_{max} = 20$)	70	NA[3]	NA	NA
PDS	G3($d_{max} = 20$)	70	$1.84e - 02$	468	24
DCS	G1($d_{max} = 4$)	60	NA[3]	NA	NA
PDS	G1($d_{max} = 4$)	60	$3.60e - 01$	236	60
DCS	G2($d_{max} = 9$)	60	NA[3]	NA	NA
PDS	G2($d_{max} = 9$)	60	$9.15e - 01$	340	58
DCS	G3($d_{max} = 20$)	60	NA[3]	NA	NA
PDS	G3($d_{max} = 20$)	60	$2.09e - 01$	564	54

---

[1]We use "NA" for DCS experiments running more than 8,000 communication rounds but not achieving the target losses.

Table 1 show the results we obtained from the experiments of solving deterministic logistic regression problems. From the table, to reach the same target loss our proposing PDS algorithm requires less rounds of communication and much less gradient evaluations than the DCS algorithm in [14]. In particular, the number of gradient evaluations barely changes as the maximum degree of the graph increases, which matches our theoretic results on graph topology invariant gradient complexity bounds. Moreover, we can see that both algorithms achieves reasonable feasibility residue, which is measured by $\|\mathcal{A}x\|_2$. It needs to be pointed out that the DCS algorithm achieves a smaller feasibility residue but a much higher loss, while the PDS algorithm spends more efforts in reducing the loss value and maintaining an acceptable feasibility residue. Therefore, we can conclude that the PDS algorithm maintains a better trade-off between loss and feasibility residue than the DCS algorithm.

We also consider the stochastic problem in which the algorithms can only access the unbiased stochastic samples of the gradients. For the SDCS algorithm, we use the parameter setting as suggested in [14] (see Theorem 4 within) except using a dynamic inner iteration limit as $\min\{10k, 5,000\}$ for possible performance improvement. For the proposed SPDS algorithm, we use the parameter setting as suggested in Theorem 3.2 for solving smooth and convex stochastic problems with $R = 1$ and $c = 1/4$ in (3.4). We set N to be the smallest iteration limit where $N \mod 10 = 0$, and hence $N = 30$ for the cases when we set target loss as 70 and $N = 100$ when target loss is 60.

Table 2

Comparison of the SDCS and SPDS algorithms in terms of reaching the same target loss

Algorithm	Graph	Target Loss	Achieved $\|\mathcal{A}x\|$	Com. rounds	Stochastic grads.
SDCS	G1($d_{max} = 4$)	70	NA[4]	NA	NA
SPDS	G1($d_{max} = 4$)	70	$6.10e-04$	428	431
SDCS	G2($d_{max} = 9$)	70	NA[4]	NA	NA
SPDS	G2($d_{max} = 9$)	70	$3.77e-04$	740	431
SDCS	G3($d_{max} = 20$)	70	NA[4]	NA	NA
SPDS	G3($d_{max} = 20$)	70	$6.12e-05$	1,418	431
SDCS	G1($d_{max} = 4$)	60	NA [4]	NA	NA
SPDS	G1($d_{max} = 4$)	60	$9.68e-04$	506	1,434
SDCS	G2($d_{max} = 9$)	60	NA [4]	NA	NA
SPDS	G2($d_{max} = 9$)	60	$8.87e-04$	816	1,433
SDCS	G3($d_{max} = 20$)	60	NA [4]	NA	NA
SPDS	G3($d_{max} = 20$)	60	$1.18e-04$	1,496	1,433

Table 2 show the results we obtained from the experiments of solving logistic regression problems using stochastic samples of gradients. Observe from the table that the number of required stochastic gradient samples for the SPDS algorithm do not change as the maximum degree of the graph increases, which matches our theoretic result that the sampling complexity of the SPDS algorithm is graph topology invariant for stochastic problems. The SPDS algorithm can achieve target losses within reasonable amount of CPU time and algorithm iterations, while the SDCS algorithm can not achieve any of the target loss within the required communication rounds. Therefore,

---

[1] We use "NA" for SDCS experiments running more than $4,000$ communication rounds but not achieving the target losses.

we can conclude that the proposed SPDS algorithm outperforms the SDCS algorithm in [14] in terms of both sampling and communication complexities.

7. Concluding remarks. In this paper, we present a new class of algorithms for solving a class of decentralized multi-agent optimization problem. Our proposed primal dual sliding (PDS) algorithm is able to compute an approximate solution to the general convex smooth deterministic problem with $\mathcal{O}\left(\sqrt{\tilde{L}/\varepsilon} + \|\mathcal{A}\|/\varepsilon\right)$ communication rounds, which matches the current state-of-the-art. However, the number of gradient evaluations required by the PDS algorithm is improved to $\mathcal{O}\left(\sqrt{\tilde{L}/\varepsilon}\right)$ and is invariant with respect to the graph topology. To the best of our knowledge, this is the only decentralized algorithm whose gradient complexity is graph topology invariant. We also propose a stochastic primal dual sliding (SPDS) algorithm that is able to compute an approximate solution to the general convex smooth stochastic problem with $\mathcal{O}\left(\sqrt{\tilde{L}/\varepsilon} + \sigma^2/\varepsilon^2\right)$ sampling complexity, which is also the only algorithm in the literature that has graph topology invariant sampling complexity. Similar convergence results of the PDS and SPDS algorithms are also developed for strongly convex smooth problems.

REFERENCES

[1] N. S. Aybat, Z. Wang, T. Lin, and S. Ma, *Distributed linearized alternating direction method of multipliers for composite convex consensus optimization*, IEEE Transactions on Automatic Control, 63 (2017), pp. 5–20.

[2] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Foundations and Trends® in Machine Learning, 3 (2011), pp. 1–122.

[3] T.-H. Chang, M. Hong, and X. Wang, *Multi-agent distributed optimization via inexact consensus admm*, IEEE Transactions on Signal Processing, 63 (2014), pp. 482–497.

[4] 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.

[5] 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 (2011), pp. 592–606.

[6] J. W. Durham, A. Franchi, and F. Bullo, *Distributed pursuit-evasion without mapping or global localization via local frontiers*, Autonomous Robots, 32 (2012), pp. 81–95.

[7] P. A. Forero, A. Cano, and G. B. Giannakis, *Consensus-based distributed support vector machines.*, Journal of Machine Learning Research, 11 (2010).

[8] S. Ghadimi and G. Lan, *Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization i: A generic algorithmic framework*, SIAM Journal on Optimization, 22 (2012), pp. 1469–1492.

[9] S. Ghadimi and G. Lan, *Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization, II: shrinking procedures and optimal algorithms*, SIAM Journal on Optimization, 23 (2013), pp. 2061–2089.

[10] M. Hong and T.-H. Chang, *Stochastic proximal gradient consensus over random networks*, IEEE Transactions on Signal Processing, 65 (2017), pp. 2933–2948.

[11] A. Jadbabaie, J. Lin, and A. S. Morse, *Coordination of groups of mobile autonomous agents using nearest neighbor rules*, IEEE Transactions on automatic control, 48 (2003), pp. 988–1001.

[12] A. Koloskova, S. Stich, and M. Jaggi, *Decentralized stochastic optimization and gossip algorithms with compressed communication*, in International Conference on Machine Learning, 2019, pp. 3478–3487.

[13] G. Lan, *Gradient sliding for composite optimization*, Mathematical Programming, 159 (2016), pp. 201–235.

[14] G. Lan, S. Lee, and Y. Zhou, *Communication-efficient algorithms for decentralized and stochastic optimization*, Mathematical Programming, 180 (2020), pp. 237–284.

[15] G. Lan and Y. Ouyang, *Accelerated gradient sliding for structured convex optimization*, arXiv preprint arXiv:1609.04905, (2016).

[16] Y. Liu, W. Xu, G. Wu, Z. Tian, and Q. Ling, *Communication-censored admm for decentralized consensus optimization*, IEEE Transactions on Signal Processing, 67 (2019), pp. 2565–2579.

[17] A. Mokhtari, W. Shi, Q. Ling, and A. Ribeiro, *A decentralized second-order method with exact linear convergence rate for consensus optimization*, IEEE Transactions on Signal and Information Processing over Networks, 2 (2016), pp. 507–522.

[18] A. Mokhtari, W. Shi, Q. Ling, and A. Ribeiro, *Dqm: Decentralized quadratically approximated alternating direction method of multipliers*, IEEE Transactions on Signal Processing, 64 (2016), pp. 5158–5173.

[19] A. Nedić and A. Olshevsky, *Distributed optimization over time-varying directed graphs*, IEEE Transactions on Automatic Control, 60 (2014), pp. 601–615.

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

[21] A. Nedić, A. Olshevsky, and C. A. Uribe, *Fast convergence rates for distributed non-bayesian learning*, IEEE Transactions on Automatic Control, 62 (2017), pp. 5538–5553.

[22] A. Nedic and A. Ozdaglar, *Distributed subgradient methods for multi-agent optimization*, IEEE Transactions on Automatic Control, 54 (2009), pp. 48–61.

[23] Y. Nesterov, *Smooth minimization of non-smooth functions*, Mathematical programming, 103 (2005), pp. 127–152.

[24] Y. E. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*, Kluwer Academic Publishers, Massachusetts, 2004.

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

[26] S. S. Ram, V. Veeravalli, and A. Nedić, *Distributed non-autonomous power control through distributed convex optimization*, in 28th Conference on Computer Communications, IEEE INFOCOM 2009, 2009, pp. 3001–3005.

[27] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, *On the linear convergence of the admm in decentralized consensus optimization*, IEEE Transactions on Signal Processing, 62 (2014), pp. 1750–1761.

[28] H. Terelius, U. Topcu, and R. M. Murray, *Decentralized multi-agent optimization via dual decomposition*, IFAC proceedings volumes, 44 (2011), pp. 11245–11251.

[29] K. I. Tsianos, S. Lawlor, and M. G. Rabbat, *Consensus-based distributed optimization: Practical issues and applications in large-scale machine learning*, in 2012 50th annual allerton conference on communication, control, and computing (allerton), IEEE, 2012, pp. 1543–1550.

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

[31] E. Wei and A. Ozdaglar, *On the O(1/k) convergence of asynchronous distributed alternating direction method of multipliers*, in 2013 IEEE Global Conference on Signal and Information Processing, IEEE, 2013, pp. 551–554.