Distributed Online Learning Algorithm With Differential Privacy Strategy for Convex Nondecomposable Global Objectives

Huqiang Cheng, Xiaofeng Liao, Fellow, IEEE, and Huaqing Li, Senior Fellow, IEEE

Abstract—In the machine learning domain, datasets share several new features, including distributed storage, high velocity, and privacy concerns, which naturally requires the development of distributed privacy-preserving algorithms. Moreover, nodes (e.g., learners, sensors, GPUs, mobiles, etc.) in the real networks usually process tasks in real time, which inevitably requires nodes to have online learning capabilities. Therefore, we deal with a general distributed constrained online learning problem with privacy over time-varying networks, where a class of nondecomposable objective functions are considered. Under this setting, each node only controls a part of the global decision variable, and the goal of all nodes is to collaboratively minimize the global objective over a time horizon $T$ while guarantees the security of the transmitted information. For such problems, we first design a novel generic algorithm framework, named as DPSDA, of differentially private distributed online learning using the Laplace mechanism and the stochastic variants of dual averaging method. Note that in the dual updates, all nodes of DPSDA employ the noise-corrupted gradients for more generality. Then, we propose two algorithms, named as DPSDA-C and DPSDA-PS, under this framework. In DPSDA-C, the nodes implement a circulation-based communication in the primal updates so as to alleviate the disagreements over time-varying undirected networks. In addition, for the extension to time-varying directed ones, the nodes implement the broadcast-based push-sum dynamics in DPSDA-PS, which can achieve average consensus over arbitrary directed networks. Theoretical results show that both algorithms attain an expected regret upper bound in $O(\sqrt{T})$ when the objective function is convex, which matches the best utility achievable by cutting-edge algorithms. Finally, numerical experiment results on both real-world and randomly generated datasets verify the effectiveness of our algorithms.

Index Terms—Differential privacy, nondecomposable objectives, distributed online learning, time-varying networks.

I. INTRODUCTION

MORE recently, there has been an increasing interest in distributed learning problems arising from its extensive use in areas like machine learning [1], sensor network [2], smart grids [3], and so on. A distinctive feature of this class of problems is that all nodes collaboratively solve a learning problem without knowledge of the global gradient information. In the setting, nodes transmit local estimates to each other with its immediate neighbors, which in turn makes nodes converge asymptotically to the optimal point.

The study on distributed learning method has developed rapidly and achieved some remarkable results. Depending on the environment in which distributed learning occurs, the methods are categorized into two types: offline and online. Distributed offline learning occurs in a static environment, where the objective function remains the same over time. Correspondingly, distributed online learning can be applied to situations in which the sequence of objective functions may change due to the uncertainty of the network environment.

A. Related Works

**Distributed offline learning:** The study of distributed offline algorithm is relatively mature, such as subgradient-push [4], primal-dual [5], [6], gradient tracking [7], [8], [9], subgradient rescaling [10], asynchronous optimization [11], [12], etc. Such algorithms often use residual error [1] as the performance measurement. Generally, the algorithm is said to be elegant if it achieves a linear convergence rate $O(a^t)$, where $t$ denotes the update counter and $0 < a < 1$ is a constant, is established for strongly convex and smooth objective functions while it reach a sublinear rate $O(1/t^2)$ for convex objective functions.

**Distributed online learning:** In practice, a host of application scenarios are dynamic, and data often needs online processing to respond quickly to the real-time needs of users. For example, many people are always keen on online activities, such as watching videos, reading news, shopping, and so on. In order to increase advertising revenue, IT companies have to provide quality ad push services for each user based on their browsing data. As users’ online activities are dynamic and uncertain over time, the task of sampling data on all users needs to be performed repeatedly. In consequence, handling nearly petabytes of data every day is their daily routine. Thus, this naturally calls for the study on online learning problem. In recent years, various types of distributed online algorithms have been developed, such as ADMM-based [13], [14], primal-dual [15], [16], [17], dual-averaging [18], [19], weight-balancing [20], subgradient-push [21], mirror descent [22], [23] and so on.

Unlike the offline algorithms, to measure the progress of online algorithms, the authors in [24] provided a standard
metric called as regret, which well captures the real-time performance of online algorithms. Note that an online learning algorithm could be claimed to be good if its regret is sublinear. It is well known that the optimal upper regret bounds is an order $O(\sqrt{T})$ (resp. $O(\log T)$) for convex (resp. strongly convex) objectives.

Nevertheless, a large amount of data for online learning may include some serious personal information, e.g., salary or medical records. Due to the distributed network topology, the information is transmitted and processed through mutual communication between neighboring nodes, and the information may be eavesdropped during the transmission, which may lead to the leakage of sensitive information. To address the privacy concern, this paper mainly focuses on the differential privacy mechanism, which scrambles the public information by adding a certain amount of noise, thus making it impossible for an attacker to learn the users’ private data. Differential privacy has become widely popular among researchers, and it has developed extremely rich mathematical formulation and provable privacy properties. The basic idea of differential privacy-based algorithms is to inject random noise or bias in the nodes’ communication or computation.

**Differential privacy:** There have been a number of research results on differential privacy. Zhu et al. [20] developed a distributed private online algorithm using a weight-balancing technique over time-varying undirected networks. Employing the Laplacian mechanism, a differential privacy version of the online subgradient-push algorithm [21] is presented in [27] for time-varying directed networks. In [26, 27], only the unconstrained optimization problems are considered. For constrained problems, by adopting the Laplace noise to the online version of the projected subgradient algorithm [10], Xiong et al. [28] proposed a subgradient preserving algorithm. Moreover, Han et al. [30] considered a privacy version of the work [18] by adding perturbation to the data flow in communication channels. In [31], a differential privacy by functional perturbation is achieved, but it is limited due to the requirement of squared integrability on the objective functions.

All of the above literature on differential privacy requires a decomposable system objective function $f_t(x)$, i.e., $f_t(x) = \sum_{i=1}^n f_{i,t}(x)$, in which any local objective $f_{i,t}$ is just revealed to node $i$ at time $t$. This case is described as overlap [33], in which each node $i$ holds and updates an estimate $x_i \in \mathbb{R}^d$ that converges asymptotically to the best decision $x^* \in \mathbb{R}^d$ of the system problem. Beware that the dimension $d$ is possibly consistent with the number of nodes $n$ or not. However, some objective functions $f_i(x)$ may not allow decomposition operations, and the decision variable $x = (x_1, \ldots, x_n)^\top \in \mathbb{R}^n$ is stored fragmentarily in all the distributed nodes, where each node $i$ only controls a part $x_i$ of $x$. This case is described as specialization [33].

**Motivations:** Such distributed problems on specialization have been considered in [19, 32, 33, 34, 35, 36, 37]. References [32, 33, 34, 35, 36, 37] are oriented towards distributed offline problems. In [36], the authors proposed an online algorithm by adopting a local dynamic programming relaxation strategy, while Lee et al. [19] designed two distributed online dual averaging algorithms with the help of the idea of [38] and the broadcast-based subgradient push [4], respectively. Regrettably, the privacy concerns are ignored in the existing works on specialization. Compared to the well-studied differential privacy on overlap, how to do differential privacy on specialization remains largely unexplored. It is of interest, therefore, to apply privacy mechanisms to the distributed problems on specialization.

**B. Contributions**

In this study, we are committed to a structured investigation of differentially private distributed online problems on specialization over a dynamic environment. To begin with, we formulate a differentially private stochastic dual-averaging distributed online algorithm in a generic form, named as DPSDA. We first inject the Laplace noise to perturb the dual variables so that the algorithm has the same output for the execution of two adjacent datasets (cf. Definition 1), which in turn makes it infeasible for the attacker to infer the true data. Then the Nesterov’s dual-averaging subgradient method [39] is adopted in DPSDA as a learning subroutine. As far as we know, DPSDA is the first formal framework of differentially private distributed online learning on specialization.

Guided by the rules of DPSDA, we design two differentially private distributed online learning algorithms. One employs the circulation-based protocol [38] for undirected communication (named as DPSDA-C), while the other utilizes the push-sum method [4] for the directed communication (named as DPSDA-PS). To be more general, the dual averaging steps of both algorithms use the noise-corrupted gradients, i.e., the gradients are evaluated with some random error, instead of the exact gradients. This arises from numerous applications as well, including distributed learning over the network and recursive regression. It is important, but not trivial, to extend this since the stochastic gradient error of each node is spread to other nodes in real time through the communication, making the dynamics statistically dependent on time and nodes.

We further conduct a rigorous expected regret analysis on the DPSDA-C and DPSDA-PS. The results show that both algorithms attain an expected regret upper bound in $O(\sqrt{T})$ for convex objective functions, which matches the best utility achievable by cutting-edge algorithms. Moreover, the derived expected regret bounds capture explicitly the effect of privacy level, vector dimension, network size, and network topology. Besides, our results concerning differential privacy reveal an inevitable trade-off between privacy levels and algorithms accuracy.

**C. Organization and Notations**

The remaining parts of this study are scheduled as follows. Section II introduces several fundamentals about the problem of interest. In Section III, we present DPSDA and a basic regret bound. DPSDA-C and its analysis are given in Section IV, while DPSDA-PS and its analysis are presented in Section V. Then, Section VI conducts numerical experiments to verify
the correctness of the established results. Finally, Section VII briefly provides some remarks on this study.

The notations of this paper are listed below.

| Notation | Definition |
|----------|------------|
| $z, z, Z$ | scalar, column vector, and matrix |
| $\mathbb{R}^d, \mathbb{R}, \mathbb{Z}_n$, and $\mathbb{Z}_+$ | the set of $d$-dimensional real vector, real numbers, nonnegative integers, and positive integers |
| $1_d$ and $I_n$ | the $d$-dimensional all-one vector and the $n \times n$ identity matrix |
| $[A]_{ij}$ and $(A)^\top$ | the $(i, j)$-th entry and the transpose of matrix $A$ |
| $\langle \cdot, \cdot \rangle$ | the inner product |
| $\|\cdot\|$ and $\|\cdot\|_1$ | $2$-norm and the $1$-norm |
| $\delta_i^k$ | the canonical vector with $i$-th entry being $1$ and the others being $0$ |
| $\Delta_k^i$ | the Kronecker delta symbol, i.e., $\delta_i^k = 1$ if $i = k$ and $\delta_i^k = 0$ otherwise |
| $\text{Lap}(\sigma)$ with $\sigma > 0$ | the Laplace distribution with probability density function $p_\sigma(x) = \frac{\sigma}{\exp\left(-\frac{|x|}{\sigma}\right)}$ |
| $\mathbb{E}(\cdot)$ and $\mathbb{P}(\cdot)$ | the expectation and probability distribution |
| $A(t:s)$ with $t \geq s \geq 0$ | the product of the time-varying matrix sequence $\{A(k)\}_{k=s}^{t}$, i.e., $A(t)\cdots A(s)$. In particular, $A(t-1:t) \equiv I_n$ with $t \geq 1$. |

II. Preliminaries

Now, we provide several preliminary materials, including graph theory, differential privacy, and problems of interest.

A. Graph Theory

Consider a general network with $n$ nodes, indicated by elements of the set $\mathcal{V} = \{1, \ldots, n\}$. The network topology specifies the local communication between nodes, which is usually modeled by one of the two graphs:

i) The one is the time-varying undirected graphs $G_1(t) = (\mathcal{V}, \mathcal{E}(t))$. Here, $\mathcal{E}(t)$ is an undirected edge set at time $t$. That is, if $(i, j) \in \mathcal{E}(t)$, then nodes $i$ and $j$ can send messages to each other at time $t$. Let $\mathcal{N}_i(t) = \{j \in \mathcal{V} | (i, j) \in \mathcal{E}(t)\} \cup \{i\}$ denote the neighbor set of node $i$ at time $t$. Also, define the degree of node $i$ at any time $t$ as $\text{deg}_i(t) = |\mathcal{N}_i(t)|$.

ii) The other is the $B$-strongly connected time-varying digraphs $G_2(t) = (\mathcal{V}, \mathcal{E}(t))$. Concretely, $\exists B > 0$ such that the union of $B$ consecutive time links $\mathcal{E}_B(t) = \bigcup_{k=(t-1)B+1}^{kB} \mathcal{E}(k)$ is strongly connected for any $t \in \mathbb{Z}_+$, where $\mathcal{E}(t)$ is a directed edge set at time $t$. That is, if $(i, j) \in \mathcal{E}(t)$, then node $i$ can send messages to node $j$ at time $t$. Let $\mathcal{N}_i^{\text{out}}(t) = \{j \in \mathcal{V} | (i, j) \in \mathcal{E}(t)\} \cup \{i\}$ and $\mathcal{N}_i^{\text{in}}(t) = \{j \in \mathcal{V} | (j, i) \in \mathcal{E}(t)\} \cup \{i\}$ denote the out- and in-neighbors of node $i$ at time $t$, respectively. Also, define the out- and in-degrees of node $i$ at time $t$ as $\text{deg}_i^{\text{out}}(t) = |\mathcal{N}_i^{\text{out}}(t)|$ and $\text{deg}_i^{\text{in}}(t) = |\mathcal{N}_i^{\text{in}}(t)|$.

B. Differential Privacy

The main role of differential privacy is to ensure that participating nodes in the network can safely broadcast their information. In the sequel, we review some basic concepts of differential privacy.

**Definition 1.** Given two datasets $\mathcal{D} = \{d_i\}_{i \in \mathcal{V}}$ and $\mathcal{D}' = \{d_i'\}_{i \in \mathcal{V}}$. The datasets $\mathcal{D}$ and $\mathcal{D}'$ are claimed to be adjacent if $d_i \neq d_i'$ and $d_j = d_j'$ for $\forall j \neq i$.

In brief, the datasets $\mathcal{D}$ and $\mathcal{D}'$ are adjacent to each other with at most one participant having different data. For the sake of brevity, let us use $\text{Adj}(\mathcal{D}, \mathcal{D}')$ to denote this relationship.

For differential privacy, we consider the worst scenario. Suppose there exists an adversary in the network who has adequate background knowledge to access all shared information by eavesdropping on the communication channel and also to capture any auxiliary information to derive sensitive information. The following definition gives the differential privacy of the algorithm in such a scenario.

**Definition 2.** Given an algorithm $\mathcal{A}$ and its set of outputs $\Psi$, i.e., $\Psi \subseteq \text{Range}(\mathcal{A})$, where $\text{Range}(\mathcal{A})$ represents the outputs of the randomized algorithm $\mathcal{A}$. For any adjacent datasets $\mathcal{D}$ and $\mathcal{D}'$ and a constant $\epsilon > 0$, if

$$\mathbb{P}[\mathcal{A}(\mathcal{D}) \in \Psi] \leq \exp(\epsilon) \cdot \mathbb{P}[\mathcal{A}(\mathcal{D}') \in \Psi],$$

is satisfied, then the algorithm $\mathcal{A}$ can be claimed to be $\epsilon$-differential privacy.

Definition 2 indicates that the output of algorithm $\mathcal{A}$ does not produce large fluctuations regardless of the addition or deletion of entries to the dataset. Thus, the adversary cannot infer sensitive information. Note that the magnitude of this fluctuation depends on the constant $\epsilon$, so the smaller the constant $\epsilon$, the higher the privacy level. Yet, an argument cannot be made that the algorithm has greater performance as long as $\epsilon$ is kept small enough. Since $\epsilon$ is related to the amount of noise added, a smaller $\epsilon$ can result in poorer accuracy of algorithm. That is to say, $\epsilon$ is a trade-off between the privacy level and the accuracy of $\mathcal{A}$.

A natural concern is how much noise amount is appropriate to add. Appropriate here means that after adding noise, the algorithm provides excellent accuracy while keeping the sensitive information secure. Next, we introduce a concept called sensitivity, which is an important factor in determining the amount of noise to be added.

**Definition 3.** Given an algorithm $\mathcal{A}$, its sensitivity is defined as, for any $t \in \mathbb{Z}_+$,

$$\Delta(t) = \sup_{\text{Adj}(\mathcal{D}_t, \mathcal{D}'_t)} \|\mathcal{A}(\mathcal{D}_t) - \mathcal{A}(\mathcal{D}'_t)\|,$$

where $\mathcal{D}_t$ and $\mathcal{D}'_t$ represent adjacent datasets at time $t$.

Sensitivity indicates the maximum impact that changing any single coordinate of data in the dataset will have on the query results, and serves a critical role in identifying how much noise needs to be added to ensure a certain privacy level.
C. Problem Formulation

Consider a constrained multi-node system in an online setting. Each node $i \in \mathcal{V}$ first comes to a decision $x_i(t)$ taken from the constrained set $\chi \subset \mathbb{R}$ at each time $t \in \mathbb{Z}_+$. Define a stack variable $\mathbf{x}(t) = [x_1(t), \ldots, x_n(t)]^\top \in \chi^n$ as the global decision at time $t$. After committing to the decision, an uncertain objective function $f_i$ is revealed as well as a cost $f_t(x(t))$ is generated by the network system. Suppose that $f_i \in \mathcal{F}$, where $\mathcal{F}$ denote a generalized class of convex functions $f_i(x(t))$. Note that $f_i$ is not known in advance by any node and is unpredictable and uncertain. The determination of the function $f_i$ may be done through the adaptive nature of the network environment, i.e., all generated information up to time $t$ may be of some significance for the selection of the $f_i$.

Let $|T| = \{1, \cdots, T\}$ wherein $T \in \mathbb{Z}_+$ is a time horizon. In this study, we investigate the following distributed constrained online learning problem:

$$\min_{t=1}^{T} \sum_{i=1}^{T} f_i(x(t)) \text{ s.t. } x \in \chi^n. \quad (1)$$

Note that since nodes do not know the objective function before making a decision, the decision is inevitably different from the best one. The performance measure of algorithm caused by this difference is called regret as follows:

$$\mathcal{R}(x(t), T) = \sum_{t=1}^{T} f_i(x(t)) - \inf_{\mathbf{v} \in \chi^n} \sum_{t=1}^{T} f_i(\mathbf{v}), \quad (2)$$

where $\mathbf{v}$ is a single action.

However, we consider a noise-bearing environment in this paper, the sequence $\{f_i\}_{i \in [T]}$ chosen according to the environment is random. Furthermore, since the gradients are evaluated with random errors, the sequence $\{x(t)\}_{t \in [T]}$ is also random variable. Hence, the conventional regret form (2), which is commonly used in the analysis of online learning, has to be corrected in some way. Here, we consider the expected form of (2) below.

$$\widetilde{\mathcal{R}}(x(t), T) = \mathbb{E} \left[ \sum_{t=1}^{T} f_i(x(t)) \right] - \inf_{\mathbf{v} \in \chi^n} \mathbb{E} \left[ \sum_{t=1}^{T} f_i(\mathbf{v}) \right]. \quad (3)$$

Following the terminology in [40], we describe $\mathcal{R}(x(t), T)$, $t \in [T]$, $T \in \mathbb{Z}_+$ as the pseudo-regret about decision $x(t)$ at time horizon $T$. Clearly, the pseudo-regret $\mathcal{R}(x(t), T)$ is the difference between the expectation of the total cost generated by the algorithm at time $T$ and the expected total cost generated by the best single decision in $\chi^n$ in hindsight.

To solve the problem (1), our objective is to design the differentially private distributed online learning algorithms that enable the pseudo-regret $\mathcal{R}(x(t), T)$ to be sublinear scaling with respect to $T$. That is to say, the algorithm performs good if $\lim_{T \to \infty} \mathcal{R}(x(t), T) / T = 0$.

**Remark 1.** Note that the problem (1) is a specialization type and not an overlap type. Recalling the definition of $x(t)$, each node $i \in \mathcal{V}$ just controls a coordinate $x_i(t)$ of it at time $t$, but each node maintains an estimate of $x(t)$ in the case of overlap. Assume that $x(t) \in \mathbb{R}^d$ and $x_i(t) \in \mathbb{R}^{d_i}$ with $d, d_i \in \mathbb{Z}_+$. In the case of specialization, it holds $d = \sum_{i=1}^{n} d_i$ while $d = d_i$ in the case of overlap. Since all nodes do not grasp the global information, the pseudo-regret in (3) reflects the impact of decentralization.

III. THE BASIC FRAMEWORK AND REGRET BOUND

In this section, we propose an universal algorithmic framework (i.e., DPSDA) of differentially private distributed online learning for problem (1), then we provide a general regret bound that allows for any algorithm deduced from DPSDA.

A. The Basic Framework—DPSDA

The algorithm employs the Laplace mechanism and uses Nesterov’s dual averaging method as the optimization subroutine. Every node $i \in \mathcal{V}$ maintains the tuple $(y_i(t), z_i(t))$ for $t \in [T]$, in which the primal variable

$$y_i(t) = \left[ y_i^1(k), \cdots, y_i^n(k) \right]^\top \in \chi^n,$$

and the dual variable

$$z_i(t) = \left[ z_i^1(k), \cdots, z_i^n(k) \right]^\top \in \mathbb{R}^n,$$

are updated by a generic framework shown in Protocol 1.

**Protocol 1** DPSDA

1: **Input:** A network graph $\mathcal{G}(t)$, constrained set $\chi$, and function class $\mathcal{F}$; randomly initialize $z_i(0) = 0$ for $i \in \mathcal{V}$; step-size $\alpha(t)$ for $\forall t \in [T]$.

2: for $t = 1, \cdots, T - 1, i \in \mathcal{V}$ do

3: Generate noise $\eta_i(t) \sim \text{Lap}(\alpha(t))$;

4: Use $\eta_i(t)$ to distort $z_i(t)$ to acquire the noisy messages $m_i(t)$ via

$$m_i(t) = z_i(t) + \eta_i(t). \quad (4)$$

5: Update the dual variable via, for $k \in \mathcal{V}$,

$$z_i^k(t + 1) = z_i^k(t) + \frac{1}{r_i(t)} \delta_i^k u_i(t) + A_{i,t}^k (m_i(t)). \quad (5)$$

6: Update the primal variable via

$$y_i(t + 1) = \Pi_{\chi_i}^e (G_{i,t} (z_i(t + 1), \alpha(t))). \quad (6)$$

7: Compute the decision variable: $x_i(t + 1) = y_i^k(t + 1)$.

8: end for

9: **Output:** $\{x_i(t)\}, i \in \mathcal{V}$.

As shown in Protocol 1, to hide the real variable $z_i(t)$, each node $i$ first perturbs $z_i(t)$ using Laplace noise $\eta_i(t)$, 3

3It can be directly extended to multidimensional space $\mathbb{R}^d$ with the help of augmented matrix. Here, we consider the scalar case, i.e., $d = 1$, for simplicity.
see [4]. In the dual update [5], \( r_i(t) > 0 \) represents a time-varying weight associated with the network topology at time \( t \), \( \delta_i^k \) is the Kronecker delta symbol, and \( u_i(t) \in \mathbb{R} \) is a random signal involving a local computation of node \( i \). Note that \( \delta_i^k \) is introduced to ensure that each node \( i \) uses \( u_i(t) \) only in its own coordinate. The main task of this step is to perform a local averaging operation \( A_i^k(\cdot) \) on the received noise information \( m_i(t) \), where the distributed features (e.g., distributed communication) are reflected in \( A_i^k(\cdot) \).

Then, the dynamic [6] of \( y_i(t+1) \) is essentially an appropriation of the dual-averaging method. Here, \( C_{i,t}(\cdot) : \mathbb{R}^n \to \mathbb{R}^n \) is a mapping operation on the dual variable \( z_i(t+1) \), \( \alpha(t) \) is a positive decay step-size, and \( \Pi_{\chi}^n \) is a mapping to ensure that the optimal point is in the feasible region, which is given by

\[
\Pi_{\chi}^n (z, \alpha) \triangleq \arg \min_{x \in \chi^n} \left\{ (z, x) + \frac{1}{\alpha} \psi(x) \right\},
\]

with a proximal function \( \psi : \chi^n \to [0, \infty) \). Suppose \( \psi \) is 1-strongly convex, i.e., it holds, for all \( a, b \in \chi^n \),

\[
\psi(b) \geq \psi(a) + \langle \partial \psi(a), b - a \rangle + \frac{1}{2} \| b - a \|^2,
\]

where \( \partial \psi(a) \) is a subgradient of \( \psi \). Note that the function \( \psi \) and step-size \( \alpha(t) \) are used to prevent excessive oscillations of the primal variable \( y_i(t) \). Once the primal and dual updates are completed, each node makes its decision via \( x_i(t+1) = y_i^t(t+1) \), where \( y_i^t \) denotes the \( i \)-th coordinate of the vector \( y_i(t+1) \).

Later on, we develop two algorithms using circulation-based method and push-sum method, respectively, which obey the rules of DPSDA.

**B. The Generic Regret Bound**

Before providing a generic regret bound, we need the standard assumption below.

**Assumption 1.** The following conditions are hold:

a) Each cost function \( f_i \in \mathcal{F} \) is convex for all \( i \in \mathbb{Z}_0 \);

b) \( \exists c > 0 \) such that \( \psi(y) \leq c \) for all \( y \in \chi^n \);

c) The constraint set \( \chi^n \) is nonempty, convex, and closed.

Moreover, the diameter of \( \chi \) is bounded by \( D_{\chi} \), i.e.,

\[
D_{\chi} \triangleq \sup_{x, y \in \chi} \| x - y \|;
\]

d) \( \exists L > 0 \) such that \( \| \nabla f_i(y) \| \leq L \) for all \( i \in \mathbb{Z}_0 \) and \( y \in \chi^n \);

e) All cost functions \( f_i \in \mathcal{F} \) are \( G \)-smooth with \( G > 0 \), i.e., it holds \( \| \nabla f_i(x) - \nabla f_i(x) \| \leq G \| x - y \| \) for all \( x, y \in \chi^n \).

We next introduce some properties about the projection operator \( \Pi \) defined in (7).

**Proposition 1.** ([4]) Given a sequence of random vectors \( \{ \varphi(t) \}_{t \in \mathbb{Z}_+} \subseteq \mathbb{R}^n \), we define a network-level summation of random variables by

\[
\bar{x}(t+1) = \Pi_{\chi}^n \left( \sum_{s=1}^{t} \varphi(s), \alpha(t) \right).
\]

Then, it follows that, for \( T \in \mathbb{Z}_+ \) and \( y \in \chi^n \),

\[
\sum_{t=1}^{T} \langle \varphi(t), \bar{x}(t) - y \rangle \leq \frac{1}{2} \sum_{t=1}^{T} \alpha(t-1) \| \varphi(t) \|^2 + \frac{\psi(y)}{\alpha(T)} \tag{9}
\]

In addition, for any \( z_1, z_2 \in \mathbb{R}^n \), it holds

\[
\| \Pi_{\chi}^n (z_1, \alpha) - \Pi_{\chi}^n (z_2, \alpha) \| \leq \alpha \| z_1 - z_2 \|. \tag{10}
\]

Using Assumption 1 and Proposition 1, we establish a generic upper bound for (4).

**Theorem 1.** Consider the problem [11] under Assumption 1. Then, for all \( T \in \mathbb{Z}_+ \), we can bound the pseudo-regret \( \bar{R}(\bar{x}, T) \) as follows:

\[
\begin{aligned}
\bar{R}(\bar{x}, T) &\leq \sum_{t=1}^{T} \alpha(t-1) \left\{ \frac{1}{2} \sum_{i=1}^{T} \alpha(T) \mathbb{E} \left[ \| \varphi(t) \|^2 \right] + \frac{C}{\alpha(T)} \right. \\
&\quad + \sum_{i=1}^{T} \mathbb{E} \left[ \| x(t) - \bar{x}(t) \|^2 \right] \\
&\quad + \sqrt{n} D_{\chi} \sum_{t=1}^{T} \mathbb{E} \left[ \| \nabla f_i(\bar{x}(t)) - g(t) \|^2 \right] \\
&\quad + \sup_{\nu \in \chi} \sum_{t=1}^{T} \mathbb{E} \left[ \| g(t) - \varphi(t), \bar{x}(t) - \nu \| \right].
\end{aligned}
\]

where \( g(t), t \in \mathbb{Z}_+ \), is an arbitrary vector.

**Proof.** See Appendix A.

Theorem 1 states that for a tight upper bound on the pseudo-regret, the following conditions are required:

E1) \( \mathbb{E} \left[ \| \varphi(t) \|^2 \right] \) remains bounded.

E2) \( x(t) \) is not too far away from \( \bar{x}(t) \).

E3) \( g(t) \) stays near the gradient \( \nabla f_i(\bar{x}(t)) \).

Once the vectors \( g(t) \) and \( \varphi(t) \) are determined, the term (E4) can be derived directly. The theorem has a significant utility for the convergence analysis of this class of algorithms, and also serves as a guide for the designing of \( u_i(t) \), \( A_i^k(\cdot) \), and \( C_{i,t}(\cdot) \).

**IV. DPSDA-C and Its Analysis**

In this section, we show an algorithm developed from DPSDA, which uses the circulation-based framework over the network \( G(t) \), hence the name DPSDA-C. Then, on the basis of Theorem 1 we give its differential privacy and pseudo-regret analysis.
A. DPSDA-C

The update rules of DPSDA-C are summarized in Algorithm 1, which essentially uses the circulation-based framework in the update of dual variable \( \mathbf{5} \) of Protocol 1. In DPSDA-C, we consider a row-stochastic weighted matrix \( W(t) \), \( t \in \mathbb{Z}_0 \), associated with the topology of \( G_1(t) \), i.e., \( W(t) 1_n = 1_n \) for all \( t \in \mathbb{Z}_+ \). Besides, \( \exists \phi > 0 \) such that \( [W(t)]_{ii} \geq \phi \) for \( i \in \mathcal{V} \), and \( [W(t)]_{ij} \geq \phi \) for \( (i, j) \in \mathcal{E}(t) \). Thus, the matrix \( W(t) \) can be given by:

\[
W(t) = \begin{cases} 
[W(t)]_{ij} (\geq \phi), & \text{if } (i, j) \in \mathcal{E}(t); \\
1 - \sum_{t \in \mathcal{N}(t) \setminus \{t\}} [W(t)]_{ii} (\geq \phi), & \text{if } i = j; \\
0, & \text{otherwise.}
\end{cases}
\]

Note that \( \phi \) is only used in the regret analysis of DPSDA-C and its exact value is not important and does not need to be known by the nodes.

Algorithm 1 DPSDA-C

1: **Input**: A network graph \( G_1(t) \), constrained set \( \chi \), and function class \( \mathbb{F} \); randomly initialize \( \mathbf{z}_i(0) = 0 \) for \( i \in \mathcal{V} \); step-size \( \alpha(t) \) for \( \forall t \in [T] \).
2: for \( t = 1, \ldots, T - 1, i \in \mathcal{V} \) do
3: Generate noise \( \eta_i(t) \sim \text{Lap}(\sigma(t)) \);
4: Use \( \eta_i(t) \) to distort \( \mathbf{z}_i(t) \) to acquire \( \mathbf{h}_i(t) \) via
   \[
   \mathbf{h}_i(t) = \mathbf{z}_i(t) + \eta_i(t). \tag{11}
   \]
5: Update the dual variable via, for \( k \in \mathcal{V} \),
   \[
   \mathbf{z}_i^k(t + 1) = n \delta^k_i u_i(t) + h_i^k(t) + \sum_{j=1}^{n} [W(t)]_{ij} (h_j^k(t) - h_i^k(t)). \tag{12}
   \]
6: Update the primal variable via
   \[
   \mathbf{y}_i(t + 1) = \Pi^h_{\mathcal{N}} \left( \mathbf{z}_i(t + 1), \alpha(t) \right). \tag{13}
   \]
7: Compute the decision variable: \( x_i(t + 1) = y_i(t + 1) \).
8: end for
9: **Output**: \( \{x_i(T)\}, i \in \mathcal{V} \).

In Algorithm 1, the perturbation mechanism regarding the model estimation \( \{11\} \) is directly come from Protocol 1. The dual update rule \( \{12\} \) draws inspiration from the local control laws \( \{38\} \), while the dual process \( \{13\} \) is essentially a Nesterov’s scheme \( \{39\} \). Recalling the challenge of algorithm design, i.e., determining the local update \( u_i(t) \) and the mappings \( A^h_{i,t}(\cdot) \) and \( C_{i,t}(\cdot) \), Algorithm 1 has given the specific form of \( A^h_{i,t}(\cdot) \) and \( C_{i,t}(\cdot) \), so next we make \( u_i(t) \) in \( \{12\} \) explicit.

After the objective function \( f_i \) is revealed at time \( t \), each node \( i \in \mathcal{V} \) needs to compute the \( i \)-th coordinate of global gradient at \( \mathbf{y}_i(t) \), i.e., \( \langle \nabla f_i(\mathbf{y}_i(t)), \mathbf{e}_i \rangle \) with some error \( \xi_i(t) \) caused by the noise in the communication network. Let \( \hat{\mathbf{g}}_i(t) \triangleq \langle \nabla f_i(\mathbf{y}_i(t)), \mathbf{e}_i \rangle + \xi_i(t) \) be the noise gradient. Thus, for the random signal \( u_i(t) \) in \( \{12\} \), we directly define

\[
u_i(t) = \hat{\mathbf{g}}_i(t), \quad t \in \mathbb{Z}_0. \tag{14}
\]

Defined as \( \mathcal{F}_t \), all generated messages of entire history by the proposed algorithm up to time \( t \in \mathbb{Z}_+ \). Then, we make an assumption about the stochastic gradient signals.

**Assumption 2.** The noise gradient \( \hat{\mathbf{g}}_i(t) \) is unbiased for \( i \in \mathcal{V} \) and \( t \in \mathbb{Z}_0 \), i.e., \( \mathbb{E}[\hat{\mathbf{g}}_i(t)|\mathcal{F}_{t-1}] = \langle \nabla f_i(\mathbf{y}_i(t)), \mathbf{e}_i \rangle \), and its second moment is finite, i.e., \( \mathbb{E}[|\hat{\mathbf{g}}_i(t)|^2 |\mathcal{F}_{t-1}| \leq L^2 \) with \( \bar{L} > 0 \).

This assumption is standard \( \{19\}, \{41\} \) and also implies \( \mathbb{E}[\xi_i(t)|\mathcal{F}_{t-1}] = 0 \). Note that if \( \mathbb{E}[|\xi_i(t)|^2 |\mathcal{F}_{t-1}| \leq \nu^2 \) holds for \( i \in \mathcal{V} \) and \( t \geq 0 \), combined with Assumption 1(d), then Assumption 2 holds with \( \bar{L} = L^2 + \nu^2 \).

B. Differential Privacy Analysis of DPSDA-C

To explore the differential privacy of DPSDA-C, a key metric is the sensitivity, which determines how much noise needs to be added to ensure that DPSDA-C achieves \( \epsilon \)-differential privacy. Therefore, we first need to derive an upper bound on the sensitivity to determine the amount of noise under \( \epsilon \)-differential privacy. The following lemma shows an upper bound of the sensitivity \( \Delta(t) \), \( t \in \mathbb{Z}_0 \), for DPSDA-C.

**Lemma 1.** Under the networks \( \{G_1(t)\}_{t \in \mathbb{Z}_0} \) and Assumptions I-2, the sensitivity of DPSDA-C yields

\[
\Delta(t) \leq 2n \bar{L}, \tag{15}
\]

**Proof.** See Appendix B.

**Remark 2.** Note that the last inequality can also factor out the coefficient \( \sqrt{d} \), when each component of the \( x \)-variable is \( d \)-dimensional vector (in this paper we consider \( d = 1 \) for ease of analysis). Thus, one should not ignore the effect of dimension \( d \) in the later discussions.

It can be learned from Lemma 1 that the upper bound of sensitivity is related to the network size \( n \), the bound of noise gradient \( \bar{L} \), and the dimension \( d \) of the vectors \( x_i(t) \) (cf. Remark 2). Then, we provide a sufficient condition to guarantee that DPSDA-C achieves \( \epsilon \)-differential privacy.

**Theorem 2.** Under the network \( \{G_1(t)\}_{t \in \mathbb{Z}_0} \), suppose that Assumptions I-2 hold. Given that the dual variables \( \{\mathbf{z}_i(t)\}_{i \in \mathcal{V}} \) are perturbed by the Laplace noises with parameter \( \sigma(t) \), such that \( \sigma(t) = \Delta(t)/\epsilon \) for all \( t \in [T] \) and \( \epsilon > 0 \), then any \( \mathbf{z}_i(t) \) of DPSDA-C for responding to queries is \( \epsilon \)-differential privacy at \( t \)-th iteration.

**Proof.** See Appendix C.

From Theorem 2, it is clearly stated that for a certain \( \epsilon \) at time \( t \in [T] \), the amount of noise added \( \sigma(t) \) is inversely proportional to the sensitivity \( \Delta(t) \).
C. Network Error of DPSDA-C

The following lemma demonstrates DPSDA-C has the traits of DPCDA, and shows the recursive forms of the dual variable $z_{i}(t)$. Note that although the computation of $\bar{z}(t)$ involves the global information, it is just available in the analysis of the algorithm.

Lemma 2. Let $\{z_{i}(t)\}_{i \in V}$ and $\{u_{i}(t)\}_{i \in V}$, $t \in \mathbb{Z}_{+}$, be the sequence generated by the dual iterates \((12)\). Define a stacking vector $u(t) = (u_{1}(t), \ldots, u_{n}(t))^{\top}$.

(a) For the weighted sum $\bar{z}(t)$, we have
\[ \bar{z}(t+1) = \bar{z}(t) + \frac{1}{n} \sum_{i=1}^{n} \eta_{i}(t) + u(t). \]
(b) For any $i, k \in V$, it follows that, from \((12)\),
\[ z_{i}^{k}(t) = n \sum_{s=0}^{t-1} [W(t-1:s+1)]_{ik} u_{k}(s) + \sum_{s=0}^{t-1} \sum_{j=1}^{n} [W(t-1:s)]_{ij} \eta_{j}^{k}(s), \]
where $\eta_{j}^{k}(s)$ represents the $k$-th entry of $\eta_{j}(s)$.

Proof. See Appendix D.

The recursive forms of $z_{i}(t)$ and $\bar{z}(t)$ in Lemma 1 help in the analysis of the network-wide disagreement. The lemma below shows a fixed upper bound of the network error.

Lemma 3. Suppose that Assumptions 1-2 hold. For the iterates \((12)\) over the networks $\{G_{1}(t)\}_{t \in \mathbb{Z}_{+}}$, it holds, for all $t \in \mathbb{Z}_{+}$,
\[ \sum_{i=1}^{n} \mathbb{E} \left[ \|z_{i}(t) - \bar{z}(t)\|^{2} \right] \leq \frac{3n^{4} \hat{L}^{2}}{\theta^{2} (1 - \theta)^{2}} + 3n^{4} \hat{L}^{2} + \frac{24n^{6} \hat{L}^{2}}{\theta^{2} (1 - \theta)^{2} \epsilon^{2}}, \]
where $0 < \theta < 1$ is a network parameter.

Proof. See Appendix E.

Actually, the result of Lemma 3 helps to deal with (E3) and (E4) in Theorem 1.

D. Pseudo-Regret of DPSDA-C

Using the result in Lemma 3, we provide the pseudo-regret bound of DPSDA-C, which is mainly for the result in Theorem 1 in the corresponding analysis of DPSDA-C by using the specific update rules \((11) - (13)\).

Theorem 3. Suppose that Assumptions 1-2 hold. Via choosing $\alpha(t) = 1/\sqrt{T}$, the pseudo-regret \((5)\) of DPSDA-C over the networks $\{G_{1}(t)\}_{t \in \mathbb{Z}_{+}}$ satisfies, for $T \in \mathbb{Z}_{+}$,
\[ \bar{R}(\bar{x}(t), T) \leq M_{1}\sqrt{T}, \]
where
\[ M_{1} = \frac{16n^{2} \hat{L}^{2}}{\epsilon^{2}} + 2n \hat{L}^{2} + C + 2 \left( L + \sqrt{nD_{x}G} \right) \times \sqrt{\frac{3n^{5} \hat{L}^{2}}{\theta^{2} (1 - \theta)^{2}} + 3n^{6} \hat{L}^{2} + \frac{24n^{7} \hat{L}^{2}}{\theta^{2} (1 - \theta)^{2} \epsilon^{2}}}. \]

Proof. See Appendix F.

Theorem 3 indicates good performance of the DPSDA-C through sublinear regret $O(\sqrt{T})$. Moreover, it explicitly emphasizes the significance of the underlying network via the parameters $n$ and $\theta$, the vector dimension $d$ (cf. Remark 2), the privacy level $\epsilon$, the constraint set $D_{x}$, and the objective function through the parameters $L$, $G$, and $\hat{L}$. To further demonstrate the performance of DPSDA-C, another variation of the pseudo-regret based on the temporal running average of the decision will be considered. That is
\[ \bar{R}(\bar{x}(t), T) = \mathbb{E} \left[ \sum_{t=1}^{T} f_{t}(\bar{x}(t)) \right] - \inf_{v} \mathbb{E} \left[ \sum_{t=1}^{T} f_{t}(v) \right], \]
where $\bar{x}(t) = \frac{1}{T} \sum_{s=1}^{T} x(s)$. The regret analysis for \((19)\) exhibits the same dependencies as the above mentioned parameters.

Corollary 1. Suppose that Assumptions 1-2 hold. Via choosing $\alpha(t) = 1/\sqrt{T}$, the pseudo-regret \((19)\) of DPSDA-C over the networks $\{G_{1}(t)\}_{t \in \mathbb{Z}_{+}}$, satisfies, for $T \in \mathbb{Z}_{+}$,
\[ \bar{R}(\bar{x}(t), T) \leq 2M_{1}\sqrt{T}, \]
where $M_{1}$ is defined in Theorem 3.

Proof. See Appendix G.

V. DPSDA-PS AND ITS ANALYSIS

We now introduce another instantiation of DPSDA, which uses a push-sum based framework over the network $G_{2}$, hence the name DPSDA-PS. Then, we use the analysis similar to that of DPSDA-C to give its differential privacy and pseudo-regret analysis.

A. DPSDA-PS

The update rules of DPSDA-PS are reported in Algorithm 2. Here, each node $i$ at time $t \in [T]$ maintains three variables: $z_{i}(t) \in \mathbb{R}^{n}$, $y_{i}(t) \in \mathbb{R}^{n}$, and $w_{i}(t) \in \mathbb{R}$. At time $t$, each node $i$ first performs a Laplace mechanism, where a Laplace noise $\eta_{i}(t)$ is injected to perturb the dual variable $z_{i}(t)$. Then, we consider an asymmetric broadcast communication, which is reflected by the column-stochastic weight matrix $A(t)$. Usually, the uniform weighting strategy is adopted to generate $A(t)$, i.e., $[A(t)]_{ij} = 1/\text{deg}_{y}^{\text{out}}(t)$ if $j \in N_{i}(t)$ and $[A(t)]_{ij} = 0$ otherwise.

In \((20) - (21)\), each node $i$ pushes its noise information $[A]_{ji}(t) \hat{h}_{i}^{k}(t)$ and auxiliary information $[A(t)]_{ij} w_{j}(t)$ to its out-neighbors. After pushing their information, nodes subsequently perform a local update by summing the information they receive. As $k \to \infty$, a bias $\pi_{c}$, which is caused by the column-stochastic matrix $[A]_{ji}(t)$ and is also the right Perron eigenvector of $[A(t)]_{ij}(t)$, arises in the consensus process among the all coordinates of $z_{i}(t)$. Hence, the scalar variable $w_{i}(t)$ is introduced in \((21)\) to eliminate the bias. The principle of this way is that the bias built up in $z_{i}(t)$ is also built up in $w_{i}(t)$. Then, dividing $z_{i}(t)$ by $w_{i}(t)$ makes the consensus process unbiased. This is why we use $z_{i}(t+1)/w_{i}(t+1)$ instead of $z_{i}(t+1)$ in the projection operation of \((22)\). In contrast to
Algorithm 2 DPSDA-PS

1: **Input:** A network graph \( G_2 = (\mathcal{V}, \mathcal{E}) \), constrained set \( \chi \), and function class \( \mathcal{F} \); randomly initialize \( z_i(0) = 0 \) and \( w_i(0) = 1 \) for \( i \in \mathcal{V} \); step-size \( \alpha(t) \) for \( \forall t \in [T] \).

2: for \( t = 1, \ldots, T \), \( i \in \mathcal{V} \) do

3: Generate noise \( \eta_i(t) \sim \text{Lap}(\sigma(t)) \);

4: Use \( \eta_i(t) \) to distort \( z_i(t) \) to acquire \( h_i(t) \) via (11);

5: Update the dual variable, for \( k \in \mathcal{V} \),

\[
z^k(t+1) = n\delta^k u_i(t) + \sum_{j=1}^{n} [A(t)]_{ij} h^j(t). \quad (20)
\]

6: Update the auxiliary variable via

\[
w_i(t+1) = \sum_{j=1}^{n} [A(t)]_{ij} w_j(t). \quad (21)
\]

7: Update the primal variable via

\[
y_i(t+1) = \Pi^\alpha_{\chi} \left( \frac{z_i(t+1)}{w_i(t+1)}, \alpha(t) \right). \quad (22)
\]

8: Compute the decision variable: \( x_i(t+1) = y^i(t+1) \).

9: end for

10: **Output:** \( \{x_i(T)\} \), \( i \in \mathcal{V} \).

DPSDA-C, the main benefit of DPSDA-PS is that it can be applied to asymmetric communication. The statements about other operations are the same as in Algorithm 1.

B. Differential Privacy Analysis of DPSDA-PS

We next provide the differential privacy of DPSDA-PS, and its proof follows the same path of DPSDA-C and thus is omitted.

**Lemma 4.** Under the network \( \{G_2(t)\}_{t \in \mathbb{Z}_0} \) and Assumptions 1-2, the sensitivity of DPSDA-PS yields (15).

**Theorem 4.** Under the network \( \{G_2(t)\}_{t \in \mathbb{Z}_0} \), suppose that Assumptions 1-2 hold. Given that the dual variables \( \{z_i(t)\}_{i \in \mathcal{V}} \) are perturbed by the Laplace noises with parameter \( \sigma(t) \), such that \( \sigma(t) = \Delta(t)/\epsilon \) for all \( t \in [T] \) and \( \epsilon > 0 \), then any \( z_i(t) \) of DPSDA-PS for responding to queries is \( \epsilon \)-differential privacy at \( t \)-th iteration.

For the statements of Lemma 4 and Theorem 4, please refer to Section IV (B), which will not be repeated here.

C. Network Error of DPSDA-PS

For DPSDA-PS, Lemma 5 below shows the results similar to Lemma 2, and its proof is omitted.

**Lemma 5.** Let \( \{z_i(t)\}_{i \in \mathcal{V}} \) and \( \{u_i(t)\}_{i \in \mathcal{V}}, t \in \mathbb{Z}_0 \) be the sequence generated by the dual iterates (20). Then, the following statements hold.

(a) The weighted sum \( \bar{z}(t) \) still satisfies (16)

(b) For any \( i, k \in \mathcal{V} \), it follows that, from (20),

\[
z^k_i(t) = n \sum_{s=0}^{t-1} [A(t-1:s+1)]_{ik} u_k(s) + \sum_{s=0}^{t-1} \sum_{j=1}^{n} [A(t-1:s)]_{ij} \eta^j_k(s). \quad (23)
\]

We provides a fixed upper bound of the network-wide disagreement term below as well as using Lemma 5.

**Lemma 6.** Suppose that Assumptions 1-2 hold. For the iterates \( \{G_2(t)\}_{t \in \mathbb{Z}_0} \), it holds, for \( t \in \mathbb{Z}_+ \),

\[
\sum_{i=1}^{n} \mathbb{E} \left[ \left\| \frac{z_i(t)}{w_i(t)} - \bar{z}(t) \right\| \right] \leq \frac{8n^2 \beta^2 \bar{L}^2}{\gamma^2 \lambda^2 (1 - \lambda)^2} + \frac{64n^4 \beta^2 \bar{L}^2}{\gamma^2 (1 - \lambda)^2} \epsilon^2,
\]

where

\[
\gamma = \inf_{t \geq 0} \left( \min_{1 \leq i \leq n} [A(t : 0) 1]_i \right), \beta = 2, \text{ and } \lambda = (1 - 1/n)^{1/2}.
\]

**Proof.** See Appendix H.

D. Pseudo-Regret of DPSDA-PS

We now present the main result of DPSDA-PS.

**Theorem 5.** Suppose that Assumptions 1-2 hold. Via choosing \( \alpha(t) = 1/\sqrt{t} \), the pseudo-regret (3) of DPSDA-PS over the networks \( \{G_2(t)\}_{t \in \mathbb{Z}_0} \) satisfies, for \( T \in \mathbb{Z}_+ \),

\[
\mathcal{R}(\bar{x}(t), T) \leq M_2 \sqrt{T},
\]

where

\[
M_2 = \frac{16n^2 \bar{L}^2}{\epsilon^2} + 2n \bar{L}^2 + C + 2(L + \sqrt{n}D_4G) \times \sqrt{\frac{8n^2 \beta^2 \bar{L}^2}{\gamma^2 \lambda^2 (1 - \lambda)^2} + \frac{64n^4 \beta^2 \bar{L}^2}{\gamma^2 (1 - \lambda)^2} \epsilon^2}.
\]

**Proof.** See Appendix I.

Comparing the results of Theorems 3 and 5, we can see that the asymmetry of the network affects the performance of DPSDA-PS through the parameters \( \gamma \) and \( \lambda \). In addition, the performance of DPSDA-PS concerned with \( \mathcal{R}(\bar{x}(t), T) \) is presented in Corollary 2 below.

**Corollary 2.** Suppose that Assumptions 1-2 hold. Via choosing \( \alpha(t) = 1/\sqrt{t} \), the pseudo-regret (19) of DPSDA-PS over the networks \( \{G_2(t)\}_{t \in \mathbb{Z}_0} \) satisfies, for \( T \in \mathbb{Z}_+ \),

\[
\mathcal{R}(\bar{x}(t), T) \leq 2M_2 \sqrt{T},
\]

where \( M_2 \) is defined in Theorem 5.

VI. Numerical Experiments

In the real world, massive data streams containing sensitive information spread in social interactions. Motivated by this, we use synthetic and real-world (see supplementary material) datasets in numerical experiments to confirm our theoretical results. Moreover, the performances of DPSDA-C and DPSDA-PS are based on averaging over 10 Monte-Carlo experiments.
We examine DPSDA-C and DPSDA-PS on a distributed constrained online linear regression problem, which is stated as follows:

$$\min_{\mathbf{x}} \frac{1}{T} \sum_{t=1}^{T} \left( \mathbf{a}(t)^\top \mathbf{x} - b(t) \right)^2 \quad \text{s.t. } \mathbf{x} \in \chi^d,$$

where the data \((\mathbf{a}_t, b_t) \in \mathbb{R}^d \times \mathbb{R}\) is revealed only at time \(t\), and the decision is limited to a scope \(\chi^d\).

A. Numerical Results on Synthetic Data

We set \(\mathbf{a}(t)\) with its entries being generated randomly drawn from the interval \([-1, 1]\) as well as \(b(t) = \mathbf{a}(t)^\top \tilde{\mathbf{x}} + \epsilon(t)\), where \(\tilde{\mathbf{x}}_i\) is set as 1 if \(1 \leq i \leq \lfloor d/2 \rfloor\) and 0 otherwise, and the noise \(\epsilon(t)\) is generated with i.i.d. \(\mathcal{N}(0, 1)\). For DPSDA-PS, we consider a simple 4-strongly connected time-varying digraphs with \(n = 7\) nodes, see Fig. 1. Also, we delete the directions of the network topology in Fig. 1 for DPSDA-C. Then, the weight matrices \(M(t)\) and \(A(t)\) in DPSDA-C and DPSDA-PS are separately generated by the uniform weighting strategy, i.e., \([M(t)]_{ij} = (\deg_i(t))^{-1}\) and \([A(t)]_{ij} = (\deg_i^\text{out}(t))^{-1}\). We set the parameters as follows: \(n = 7, d = 21, \chi = [-5, 5],\) and \(\alpha(t) = 1/\sqrt{T}\).

We show the empirical performances of the rescaled pseudo-regret \(\bar{\mathcal{R}}(\mathbf{x}(t), T) / T\) and the pseudo-regrets concerning the running average of the decisions \(\bar{\mathbf{x}}(t), T\) as two functions of the time horizon \(T\), and run Algorithms 1 and 2 with \(\epsilon = 0, 0.1, 1, 5\), respectively. The numerical experiments show that: i) the empirical results of DPSDA-C and DPSDA-PS shown in Fig. 2(Top) are accordant with the theoretical results provided by Theorems 1-2 that the regret \(\bar{\mathcal{R}}(\mathbf{x}(t), T) / T\) tends to zero as \(T\) tends to infinity; ii) the empirical results of DPSDA-C and DPSDA-PS shown in Fig. 2(Bottom) validate the Corollaries 1-2; and iii) the constant \(\epsilon\) determines a trade-off between the privacy level and the algorithm accuracy. In addition, as \(\epsilon\) gradually increases, the performance of the algorithm gradually approaches or even slightly exceeds that of the non-private version.

VII. Conclusion

This paper has focused on a distributed online constrained learning problem on specialization with differential privacy. For such problems, we proposed DPSDA, a framework of differentially private stochastic dual-averaging distributed online algorithm. Further, we developed two algorithms, DPSDA-C and DPSDA-PS, based on DPSDA. Using the convexity of the objective functions, we derived sublinear expected regrets \(O(\sqrt{T})\) for both algorithms, which is the best theoretical regret of state-of-the-art algorithms, and also revealed that a choice of \(\epsilon\) is a trade-off between the privacy level and the algorithm performance. Finally, the numerical results further verified the correctness of all theoretical results. In the future, we will consider inequality constraints and time-delay communication.

ACKNOWLEDGMENTS

This work was supported in part by the National Key R&D Program of China under Grant 2018AAA0100101, in part by the National Natural Science Foundation of China under Grant 61932006 and 61772434, and in part by the Chongqing technology innovation and application development project under Grant cstc2020jscx-msxmX0156.

REFERENCES

[1] X. Ran, S. Kar, and U. A. Khan, “Decentralized stochastic optimization and machine learning: A unified variance-reduction framework for robust performance and fast convergence,” IEEE Signal Process. Mag., vol. 37, no. 3, pp. 102–113, May, 2020.
[2] G. Foderaro, P. Zhu, H. Wei, T. A. Wettergren, and S. Ferrari, “Distributed optimal control of sensor networks for dynamic target tracking,” IEEE Trans. Control Netw. Syst., vol. 5, no. 1, pp. 142–153, Mar. 2018.
[3] M. Ubl and M. Hale, “Totally asynchronous large-scale quadratic programming: Regularization, convergence rates, and parameter selection,” IEEE Trans. Control Netw. Syst., vol. 8, no. 3, pp. 1465–1476, Sep. 2021.
[4] A. Nedić and A. Ozolshevsky, “Distributed optimization over time-varying directed graphs,” IEEE Trans. Autom. Control, vol. 60, no. 3, pp. 601–615, Mar. 2015.
[5] H. Li, E. Su, C. Wang, J. Liu, Z. Zheng, Z. Wang, and D. Xia, “A primal-dual forward-backward splitting algorithm for distributed convex optimization,” IEEE Trans. Emerg. Topics Comput. Intell., 2021, doi: 10.1109/TETCI.2021.3098831.
[6] W. Shi, Q. Ling, G. Wu, and W. Yin, “EXTRA: An exact first-order algorithm for decentralized consensus optimization,” SIAM J. Optim., vol. 25, no. 2, pp. 944–966, 2015.

[7] S. Pu, W. Shi, J. Xu, and A. Nedic, “Push-pull gradient methods for distributed optimization in networks,” IEEE Trans. Autom. Control, vol. 66, no. 1, pp. 1–16, Jan. 2021.

[8] G. Qu and N. Li, “Harnessing smoothness to accelerate distributed optimization,” IEEE Trans. Control Netw. Syst., vol. 5, no. 3, pp. 1245–1260, Sep. 2018.

[9] A. Nedić, A. Olshesky, and W. Shi, “Achieving geometric convergence for distributed optimization over time-varying graphs,” SIAM J. Optim., vol. 27, pp. 2597–2633, Dec. 2017.

[10] V. S. Mai and E. H. Abed, “Distributed optimization over directed graphs with row stochasticity and constraint regularity,” Automatica, vol. 102, pp. 94–104, 2019.

[11] J. Zhang and K. You, “AsySPA: An exact asynchronous algorithm for convex optimization over digraphs,” IEEE Trans. Autom. Control, vol. 65, no. 6, pp. 2494–2509, Jun. 2020.

[12] Y. Tian, Y. Sun, and G. Scutari, “Achieving linear convergence in distributed asynchronous multiagent optimization,” IEEE Trans. Autom. Control, vol. 65, no. 12, pp. 5264–5279, Dec. 2020.

[13] M. Akbari, B. Gharesifard, and T. Linder, “Individual regret bounds for the distributed online alternating direction method of multipliers,” IEEE Trans. Autom. Control, vol. 64, no. 4, pp. 1746–1752, Apr. 2019.

[14] A. Koppel, F. Y. Jakubiec, and A. Ribeiro, “Online learning over a decentralized network through ADMM,” J. Oper. Res. Soc., vol. 3, no. 4, pp. 537–562, 2015.

[15] Y. Yuan, D. W. Ho, and G. P. Jiang, “An adaptive primal-dual subgradient algorithm for online distributed constrained optimization,” IEEE Trans. Cybern., vol. 48, no. 11, pp. 3045–3055, Nov. 2018.

[16] A. Koppel, E. Y. Jakubiec, and A. Ribeiro, “A saddle point algorithm for networked online convex optimization,” IEEE Trans. Signal Process., vol. 63, no. 19, pp. 5149–5164, Oct. 2015.

[17] A. Koppel, S. Paternain, C. Richard, and A. Ribeiro, “Decentralized online learning with kernels,” IEEE Trans. Signal Process., vol. 66, no. 12, pp. 3240–3255, Jun. 2018.

[18] S. Hosseini, A. Chapman, and M. Mesbahi, “Online distributed optimization on dynamic networks,” IEEE Trans. Autom. Control, vol. 61, no. 11, pp. 3545–3550, Nov. 2016.

[19] S. Lee, A. Nedić, and M. Raginsky, “Stochastic dual averaging for decentralized online optimization on time-varying communication graphs,” IEEE Trans. Autom. Control, vol. 62, no. 12, pp. 6407–6414, Dec. 2017.

[20] C. Xu, J. Zhu, and D. O. Wu, “Decentralized online learning methods based on weight-balancing over time-varying digraphs,” IEEE Trans. Emerg. Topics Comput. Intell., vol. 5, no. 3, pp. 394–406, Jun. 2021.

[21] M. Akbari, B. Gharesifard, and T. Linder, “Distributed online convex optimization on time-varying directed graphs,” IEEE Trans. Control Netw. Syst., vol. 4, no. 3, pp. 417–428, Sep. 2017.

[22] X. Yi, X. Li, L. Xie, and K. H. Johanssen, “Distributed online convex optimization with time-varying coupled inequality constraints,” IEEE Trans. Signal Process., vol. 68, pp. 731–746, 2020.

[23] S. Shahrampour and A. Jadbabaie, “Distributed online optimization in dynamic environments using mirror descent,” IEEE Trans. Autom. Control, vol. 63, no. 3, pp. 714–725, Mar. 2018.

[24] D. Mateos-Núñez and J. Cortés, “Distributed online convex optimization over jointly connected digraphs,” IEEE Trans. Netw. Sci. Eng., vol. 1, no. 1, pp. 23–37, Jan.-Jun. 2014.

[25] C. Dwork, F. McSherry, K. Nissim, and A. Smith, “Calibrating noise to sensitivity in private data analysis,” in Proc. 3rd Theory Cryptography Conf., 2006, pp. 265–284.

[26] Y. Xu, C. Xu, J. Guan, and D. O. Wu, “Differentially private distributed online algorithms over time-varying directed networks,” IEEE Trans. Signal Inform. Process. Over Netw., vol. 7, no. 1, pp. 4–17, Mar. 2018.

[27] Q. Lu, X. Liao, T. Xiang, H. Li, and T. Huang, “Privacy masking stochastic subgradient-push algorithm for distributed online optimization,” IEEE Trans. Cybern., vol. 51, no. 6, pp. 3224–3237, Jun. 2021.

[28] Y. Xiong, J. Xu, K. You, J. Liu, and L. Wu, “Privacy preserving online optimization over unbalanced digraphs via subgradient rescaling,” IEEE Trans. Control Netw. Syst., vol. 7, no. 1, pp. 1366–1379, Sep. 2020.

[29] D. Kempe, A. Dobra, and J. Gehrke, “Gossip-based computation of aggregate information,” in Proc. 44th Annu. IEEE Symp. Foundations of Computer Science, Cambridge, MA, USA, 2003, pp. 482–491, DOI: 10.1109/FOCS.2003.1238221.

[30] D. Han, K. Liu, Y. Lin, and Y. Xia, “Differentially private distributed online learning over time-varying digraphs via dual averaging,” Int. J. Robust Nonlin. Control, pp. 1–15, 2021, DOI: 10.1002/rnc.6535.
What follows includes additional theoretical and numerical results, as well as all the proofs of the lemmas and theorems presented in the main paper.

VIII. DISCUSSION ABOUT DPSDA

Although we have given such a framework, it is still a challenge to determine the local update $u_i(t)$ and the mappings $A^k_{i,t}(\cdot)$ and $C_{i,t}(\cdot)$. Our framework is based on the Laplace mechanism and Nesterov’s dual-averaging method, so the specific design of the algorithm must follow similar rules as in the following differentially private centralized dual-averaging (DPDCDA) method, which is based on the centralized dual averaging method [39] and the laplace mechanism [25]:

$$z(t+1) = z(t) + \eta(t) + g(t),$$  \hspace{1cm} (24)

$$y(t+1) = \Pi^c_{V_n}(z(t+1), \alpha(t)),$$  \hspace{1cm} (25)

where $z(t)$, $\eta(t)$, $g(t)$, $y(t)$, and $\alpha(t)$ are the dual variable, the Laplace noise, the gradient, the primal variable, and the step-size at time $t$, respectively. Define a weighted sum of the dual variables $\bar{z}(t) = \frac{1}{n} \sum_{i=1}^n z_i(t)$. For simplicity, we assume that symmetric or asymmetric communication requires that the positive weights are uniform for all nodes. Therefore, we directly take $r_i(t) = 1/n$ for $i \in V$ and $t \in [T]$.

Adding $z^k_i(t+1)$ in (5) over $i$ yields

$$\bar{z}^k(t+1) = u_k(t) + \frac{1}{n} \sum_{i=1}^n A^k_{i,t}(z_i(t) + \eta_i(t)).$$  \hspace{1cm} (26)

Comparing (24) and (26), it is clear that $u_i$ is related to the gradient computation as well as $A^k_{i,t}(\cdot)$ satisfies:

$$\frac{1}{n} \sum_{i=1}^n A^k_{i,t}(z_i(t) + \eta_i(t)) = \bar{z}^k(t) + \bar{\eta}(t).$$

After obtaining $\bar{z}^k(t+1)$, (5) is transformed into

$$\bar{y}_i(t+1) = \Pi^c_{V_n}(C_{i,t}(\bar{z}_i(t+1)), \alpha(t)).$$  \hspace{1cm} (27)

Due to $C_{i,t}(\cdot) : \mathbb{R}^n \rightarrow \mathbb{R}^n$, it is easy to determine the mapping $C_{i,t}(\cdot)$ from $C_{i,t}(\bar{z}_i(t+1)) = \bar{z}_i(t+1)$.

IX. ADDITIONAL NUMERICAL RESULTS

If not specifically highlighted, we use the same settings as in Section VI of the main paper.

A. Effects of vector dimension

We check the effect of vector dimensions on the performance of DPSDA-C and DPSDA-PS. We choose $d = 7, 21, 35, 49, 63$ to observe the differences in the empirical curves. In this setting, we fix $\epsilon = 1$ and use the network in Fig. 1. As shown in Fig. 3, the performance is really degraded as the dimensionality of the synthetic data increases.

Fig. 3: Algorithm performance under different vector dimensions $d$. (Left): DPSDA-C; (Right): DPSDA-PS.

B. Results on Real-world Datasets

In the following, our numerical experiments are built on real-world datasets. One of them is the Cleveland Heart Disease (CHD) dataset from UCI repository\footnote{Available at \url{http://archive.ics.uci.edu/ml}} with 14 features. We preprocess the CHD by removing observations with missing values, which has little effect on the distribution of the samples. Then, the values of all feature elements are shrunk to $0-1$. The remaining datasets are from the LIBSVM\footnote{Available at \url{http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/}} library. The following table summarizes the details of the datasets.

| dataset   | # of features | # of instances |
|-----------|---------------|----------------|
| bodyfat   | 14            | 252            |
| CHD       | 14            | 297            |
| australian| 14            | 690            |
| mushroom  | 112           | 8124           |

1) Performances on different datasets: We fix $\epsilon = 1$ and also use the network in the above experiments. For each dataset, we plot the pseudo-regret $\mathcal{R}(x(t), T)$ of DPSDA-C and DPSDA-PS. It is decipted from Fig. 4 that i) the numerical results on real-world datasets are consistent with the theoretical results, and ii) algorithms perform differently on different datasets. The larger the sample size and the higher the dimension, the worse the performance of the algorithms.
Here, we use CHD dataset and choose finally we observe the coordinates’ estimation of the nodes. We have confirmed the utilities of DPSDA-C and DPSDA-PS, and with DPPS and DPSR.

DPSDA-C and DPSDA-PS have better global pseudo-regrets compared towards decomposable objective functions. Therefore, we need to modify the linear regression problem as follows:

\[
\min_{\mathbf{x} \in \mathbb{R}^d} \sum_{t=1}^{T} \sum_{i=1}^{n} \left( a_i^t (t) \mathbf{x} - y_i(t) \right)^2 \text{ s.t. } \mathbf{x} \in \chi^n.
\]

We use the networks in Fig. 1 for DPSDA-PS and DPPS, its undirected version for DPSDA-C, and its fixed version for DPSR. Note that each node has its individual pseudo-regret for decomposable problem. We denote the maximum and minimum individual pseudo-regrets as AReg and IReg, i.e.,

\[
\text{AReg} \triangleq \max_{j \in V} \mathcal{R} (\mathbf{x}_j (t), T) \quad \text{and} \quad \text{IReg} \triangleq \min_{j \in V} \mathcal{R} (\mathbf{x}_j (t), T).
\]

In addition, to better and fairly convey the all nodes’ performance in DPPS and DPSR, we define VReg as a global metric for them. We set \( \epsilon = 1 \) for all algorithms. The performance comparisons based on different datasets are shown in Fig. 5. Based on these plots it can be seen that DPSDA-C and DPSDA-PS match the best pseudo-regret performance (i.e., IReg) of DPPS and DPSR. Moreover, DPSDA-C and DPSDA-PS have better global pseudo-regrets compared with DPPS and DPSR.

3) Hiding the decision-making: All the above experiments have confirmed the utilities of DPSDA-C and DPSDA-PS, and finally we observe the coordinates’ estimation of the nodes. Here, we use CHD dataset and choose \( \epsilon = 0, 0.1, 1, 5 \), respectively. Let the results under \( \epsilon = 0 \) be a public information. In this setting, we have \( \mathbf{x}_i \in \mathbb{R}^2 \) for \( i \in \{1, \cdots, 7\} \), where \( x_i \) is the \( i \)-th coordinate of decision vector \( \mathbf{x} \). We plot the estimates \( x_{4,1} \), the first entry of \( \mathbf{x}_4 \), of DPSDA-C and DPSDA-PS shown by Fig. 6. The results show that the hiding of the primal variable is enabled by perturbing the dual variable. The more noise is added, the more the hiding effect is.

![Fig. 4: The performance of the proposed algorithms under different datasets. (Left): DPSDA-C; (Right): DPSDA-PS.](image)

![Fig. 6: The Trajectories of \( x_{4,1} \) in the proposed algorithms. (Left): DPSDA-C; (Right): DPSDA-PS.](image)
where the second inequality uses Cauchy-Schwarz inequality, and the last inequality follows from the fact that the constraint set \( \chi \) is upper bounded by \( D \chi \). Combining the above results and taking the total expectation, we can arrive at the claimed result. 

\[ \square \]

**APPENDIX B**

**PROOF OF LEMMA 1**

**Proof.** Recall the adjacent relation between \( D \) and \( D' \) at time \( t \) in Definition 3. Let \( z_i(t) \) and \( z'_i(t) \) be the outputs by running \( A(D) \) and \( A(D') \), respectively. Hence, following from (12), we can obtain that, for \( t \in \mathbb{Z}_0 \) and \( k \in \mathcal{V} \),

\[
\| z_i(t + 1) - z'_i(t + 1) \|_1 \\
\leq \sum_{k=1}^{n} | z_k^i (t + 1) - z_k^{i'}(t + 1) | \\
= \sum_{k=1}^{n} n \delta_i^k u_{i}(t) - n \delta^{i'}_k u_{i}'(t) \\
= \| nu_k(t) - nu'_k(t) \|,
\]

where the first inequality and the third equality use the definitions of 1-norm and Kronecker delta symbol, respectively. From Assumption 2, it holds \( \mathbb{E} [ \| u_i(t) \| | F_{t-1} ] \leq \tilde{L} \). Then, since the selection of adjacent dataset pairs \((D_t, D'_t)\) is random, we have

\[
\Delta(t) \leq \mathbb{E} [ \| z_i(t + 1) - z'_i(t + 1) \|_1 | F_t ] \leq 2n \tilde{L},
\]

which is the desired result. 

\[ \square \]

**APPENDIX C**

**PROOF OF THEOREM 2**

**Proof.** Note that the information queried by the attacker is equal, i.e., \( h_i(t) = h_i(t), i \in \mathcal{V}, t \in [T] \), for any adjacent datasets \( D_t \) and \( D'_t \). Following the path of [42], we can obtain that

\[
\prod_{i=1}^{n} \prod_{k=1}^{n} \mathbb{P} \left[ h_i^k(t) - z_i^k(t) \right] \\
\leq \prod_{i=1}^{n} \prod_{k=1}^{n} \exp \left( \frac{h_i^k(t) - z_i^k(t) - h_i^k(t) + z_i^k(t)}{\sigma(t)} \right) \\
= \exp \left( \frac{\Delta(t)}{\sigma(t)} \right) = \exp(e),
\]

where \( h_i^k(t) \) and \( h_i^k(t) \) are the \( k \)-th component of \( h_i(t) \) and \( h_i(t) \), respectively, the first inequality uses the triangle inequality, and the second inequality is obtained from the definition of sensitivity. The desired result is obtained. 

\[ \square \]

**APPENDIX D**

**PROOF OF LEMMA 2**

**Proof.** (a) Define a matrix \( P^k(t) \in \mathbb{R}^{n \times n} \) with entries \( [P_k(t)]_{ij} = z^i_j(t) - z^i_j(t) \). So, it holds that \( P^k(t) \) is a skew-symmetric matrix. Using the definition of \( \mathbb{E}(t) \) and the dual dynamic (12) gives

\[
\tilde{z}^k(t + 1) \\
= \frac{1}{n} \sum_{i=1}^{n} \left( h_i^k(t) + n \delta^k u_{i}(t) + \sum_{j=1}^{n} [W(t)]_{ij} [P^k(t)]_{ij} \right) \\
= \tilde{z}^k(t) + \frac{1}{n} \sum_{i=1}^{n} [W(t)]_{ij} u_{i}(t) + tr(W(t)P^k(t)),
\]

where each entry of \( W(t) \in \mathbb{R}^{n \times n} \) is given by \( [W(t)]_{ij} = \frac{1}{n} [W(t)]_{ij} \cdot W(t) + tr(W(t)P^k(t)), \)
we can learn that $\tilde{W}(t)$ is symmetric and $P^k(t)$ is skew-symmetric. Thus, it can be derived that $tr[\tilde{W}(t) P^k(t)] = 0$.

The desired result (16) is obtained directly.

(b) Let $r_k(t)$ and $\vartheta_k(t)$ be given by stacking up the $k$-th coordinates of $z_i(t)$ and $\eta_i(t)$, respectively. That is, $r_k(t) = [z^1_k(t), \ldots, z^n_k(t)]^T$ and $\vartheta_k(t) = [\eta^1_k(t), \ldots, \eta^n_k(t)]^T$.

Then, stacking up the $h^k_i(t)$ in (11) over $i$ gives

$$r_k(t + 1) = W(t) (r_k(t) + \vartheta_k(t)) + n u_k(t) e_k,$$  \hspace{1cm} (33)

Since $z_i(0) = 0$ for $i \in V$, it holds that $r_k(0) = 0$ for $k \in V$.

By computing (33) recursively, we have

$$r_k(t) = W(t - 1 : 0) r_k(0) + \sum_{s=0}^{t-1} W(t - 1 : s) \vartheta_k(s) + n \sum_{s=0}^{t-1} u_k(s) (t - 1 + s) e_k,$$

$$= W(t - 1 : s) \vartheta_k(s) + n \sum_{s=0}^{t-1} u_k(s) W(t - 1 + s) e_k,$$

where the first step uses the fact that $W(t - 1) = I_n$. Then, separating the $i$-th component of $r_k(t)$ gives the claimed result.

APPENDIX E
PROOF OF LEMMA 3

Proof. Recalling the definitions of $z_i(t)$ and $\bar{z}_i(t)$, it follows that, for $t \in \mathbb{Z}_0$,

$$\sum_{i=1}^{n} \| z_i(t) - \bar{z}_i(t) \|^2 = \sum_{i=1}^{n} \sum_{k=1}^{n} | z^k_i(t) - \bar{z}^k_i(t) |^2.$$  \hspace{1cm} (34)

Then, computing (16) recursively yields

$$\bar{z}^k(t) = \left\{ \begin{array}{ll} \frac{1}{n} \sum_{i=1}^{n} \eta^k_i(s) + \sum_{s=0}^{t-1} u_k(s), & \text{for } \sum_{i=1}^{n} \frac{\eta^k_i(s)}{n} \\ \frac{1}{n} \sum_{i=1}^{n} \eta^k_i(s), & \text{for } \sum_{i=1}^{n} \frac{\eta^k_i(s)}{n} \end{array} \right.$$  \hspace{1cm} (35)

Subtracting (35) from (17), we obtain

$$z^k_i(t) - \bar{z}^k_i(t) = n \sum_{s=0}^{t-1} ([W(t - 1 : s + 1)]_{ik} - \frac{1}{n} u_k(s))$$

$$+ \sum_{j=1}^{t-1} \sum_{s=0}^{t-1} ([W(t - 1 : s)]_{ij} - \frac{1}{n} \eta^k_j(s))$$

$$= n \sum_{s=1}^{t} ([W(t - 1 : s)]_{ik} - \frac{1}{n} u_k(s - 1))$$

$$+ n ([W(t - 1 : t)]_{ik} - \frac{1}{n} u_k(t - 1))$$

$$+ \sum_{s=0}^{t-1} \sum_{j=1}^{t-1} ([W(t - 1 : s)]_{ij} - \frac{1}{n} \eta^k_j(s)).$$

According to (43), it holds, for $\forall i, k \in V$ and $t \geq s \in \mathbb{Z}_0$,

$$\| W(t : s) \|_{ik} \leq \theta^{t-s-1},$$

where $\theta = 1 - \frac{\sigma}{\pi n^2} < 1$. Then, we can bound $| z^k_i(t) - \bar{z}^k_i(t) |$ as

$$| z^k_i(t) - \bar{z}^k_i(t) | \leq n \sum_{s=1}^{t} \theta^{t-s} | u_k(s - 1)| + n | u_k(t - 1) |$$

$$+ \sum_{s=0}^{t-1} \sum_{j=1}^{t-1} \theta^{t-s} | \eta^k_j(s) |.$$

Using the inequalities $\left( \sum_{j=1}^{n} a_j \right)^2 \leq q \sum_{j=1}^{n} (a_j)^2$ with $q \in \mathbb{Z}_+$ and $a_j \geq 0$ for $j \in [q]$ yields

$$| z^k_i(t) - \bar{z}^k_i(t) |^2 \leq \frac{3n^2}{\theta^2 (1 - \theta)^2} \max_{s=1, \ldots, t-1} | u_k(s - 1) |^2 + 3n^2 | u_k(t - 1) |^2$$

$$+ \frac{3n}{\theta^2 (1 - \theta)^2} \max_{s=1, \ldots, t-1} \left( \sum_{j=1}^{n} | \eta^k_j(s) |^2 \right).$$

Since $\bar{z}_i(t) \sim \text{Lap}(\sigma(t))$, and each $\eta_i(t) \in \mathbb{R}$ is independent, it gives $E[| \eta^k_i(t) |^2] = 2 \sigma(t)^2$. Recalling $\sigma(t) = \Delta(t)/\epsilon$ in Theorem 1, for $\forall \epsilon \in \mathbb{Z}_0$ and $\epsilon > 0$, one obtains that

$$E \left( \sum_{j=1}^{n} | \eta^k_j(s) |^2 \right) = 2 n (\sigma(t))^2 \leq \frac{8n^3 \tilde{L}^2}{\epsilon^2},$$  \hspace{1cm} (36)

where the inequality uses the fact that $\Delta(t) \leq 2 n \tilde{L}$. According to (14) and (36), taking the expectation, we arrive at

$$E \left[ | z^k_i(t) - \bar{z}^k_i(t) |^2 \right] \leq \frac{3n^2 \tilde{L}^2}{\theta^2 (1 - \theta)^2} + 3n^2 \tilde{L}^2 + \frac{24n^4 \tilde{L}^2}{\theta^2 (1 - \theta)^2 \epsilon^2}.$$

Combining this and relation (34), we complete the proof.

APPENDIX F
PROOF OF THEOREM 3

Proof. Recalling the results of Theorem 1, we define $\bar{x}(t) \triangleq \Pi^\psi_{\epsilon} (\bar{z}(t), \alpha (t-1))$. Moreover, let the dual update rule in (12) have the form of (8) as follows:

$$\bar{x}(t) \triangleq \Pi^\psi_{\epsilon} (\bar{z}(t), \alpha (t-1))$$

$$= \Pi^\psi_{\epsilon} \left( \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{t-1} u_i(s) \right).$$  \hspace{1cm} (37)

Here, (37) takes the recursive form of (16). According to our analysis, the $\varphi$-variable in (8) can be defined as $\varphi(t) \triangleq \frac{1}{n} \sum_{i=1}^{n} \vartheta_i(t) + u(t)$. Then, recalling (E1) in Theorem 1, we have

$$E \left[ \| \varphi(t) \| \right] \leq \frac{2}{n} E \left[ \sum_{i=1}^{n} \| \eta_i(s) \| \right] + 2n \tilde{L}^2$$

$$\leq \frac{16n^2 \tilde{L}^2}{\epsilon^2} + 2n \tilde{L}^2,$$

where the last inequality uses

$$E \left[ \sum_{i=1}^{n} \| \eta_i(s) \| \right] = \sum_{i=1}^{n} E \left[ \| \eta_i(s) \| \right] = 2n \sigma^2(t)$$

$$= \frac{2n \Delta^2(t)}{\epsilon^2} \leq \frac{8n^3 \tilde{L}^2}{\epsilon^2}.$$  \hspace{1cm} (38)
Let \( \bar{x}(t) = (\bar{x}^1(t), \cdots, \bar{x}^n(t)) \). For (E2) in Theorem 1, it holds that, from the definition of the global decision \( x(t) \),
\[
\|x(t) - \bar{x}(t)\| = \left\| \sum_{i=1}^{n} (x_i(t) - \bar{x}^i(t)) e_i \right\| 
\leq \sum_{i=1}^{n} \|y_i(t) - \bar{x}(t)\|, \tag{39}
\]
where the inequality uses the fact that \( x_i(t) = y_i^t(t) \).

We next consider (E3) in Theorem 1. Let \( g(t) = (g_1(t), \cdots, g_n(t)) \) be a stacking vector with each component \( g_i(t), i \in \mathcal{V} \), satisfying
\[
g_i(t) = E[g_i(t) | \mathcal{F}_{t-1}] = (\nabla f_t(y_i(t)), e_i).
\]
Then, it follows that
\[
\|\nabla f_t(\bar{x}(t)) - g(t)\|
= \left\| \sum_{i=1}^{n} (\nabla f_t(\bar{x}(t)) - \nabla f_t(y_i(t)), e_i) e_i \right\|
\leq \sum_{i=1}^{n} \|\nabla f_t(\bar{x}(t)) - \nabla f_t(y_i(t))\|
\leq G \sum_{i=1}^{n} \|\bar{x}(t) - y_i(t)\|,
\]
where the last inequality is derived from the fact that all functions \( f_t \in \mathcal{F} \) are \( G \)-smooth. For (E2) and (E3), using (10) in Proposition 1, we further have
\[
\|\bar{x}(t) - y_i(t)\|
= \|\Pi_{\mathcal{X}_0}^{\alpha}(\bar{x}(t), \alpha(t-1)) - \Pi_{\mathcal{X}_0}^{\alpha}(z_i(t), \alpha(t-1))\|
\leq \alpha(t-1) \|\bar{z}(t) - z_i(t)\|.
\]

For (E4) in Theorem 1, due to \( E[\eta_i(s)] = 0 \) for \( i \in \mathcal{V} \) and \( s \in \mathcal{Z}_0 \), it holds that \( E[\varphi(t)] = E[u(t)] = g(s) \). Consequently, we obtain \( E[(g(t) - \varphi(t), \bar{x}(t) - v)] = 0 \).

Substituting the analytical results of (E1), (E2), (E3) and (E4) in DPSDA-C to Theorem 1 yields
\[
\hat{R}(x(t), T)
\leq \left( \frac{8n^2 \ell^2}{c^2} + n \ell^2 \right) \sum_{i=1}^{T} \alpha(t-1) + \frac{C}{\alpha(T)}
+ (L + \sqrt{n} D_\chi G) \sum_{i=1}^{T} \alpha(t-1) \sum_{i=1}^{n} E[\|\bar{x}(t) - z_i(t)\|]. \tag{40}
\]
Applying Jensen’s inequality to the last term in (40), combined with the result of Lemma 3, we obtain
\[
\sum_{i=1}^{n} E[\|z_i(t) - \bar{x}(t)\|]
\leq \sqrt{n} \sqrt{\sum_{i=1}^{n} E[\|z_i(t) - \bar{x}(t)\|^2]} 
\leq \sqrt{n} \sqrt{\sum_{i=1}^{n} \sum_{i=1}^{T} E[\|z_i(t) - \bar{x}(t)\|^2]}
\leq \sqrt{\frac{3n^5 \ell^2}{\theta^2 (1-\theta)^2} + \frac{24n^7 \ell^2}{\theta^2 (1-\theta)^2} c^2}.
\]
Then, since \( \alpha(t) = 1/\sqrt{t} \), it holds \( \sum_{i=1}^{T} 1/\sqrt{t} \leq 2\sqrt{T} \).

Lastly, substituting them into (40) gives the desired result. \( \square \)

**APPENDIX G**

**PROOF OF COROLLARY 1**

**Proof.** By using the convexity of \( f_t \), it holds that \( f_t(\bar{x}(t)) \leq \frac{1}{T} \sum_{s=1}^{T} f_t(x(s)) \). Then, we obtain
\[
E \left[ \sum_{i=1}^{T} f_t(\bar{x}(t)) \right] = \inf_{v \in \mathcal{X}^n} E \left[ \sum_{i=1}^{T} f_t(v) \right]
= \sum_{i=1}^{T} \left( E \left[ \sum_{j=1}^{t} f_t(x(s)) \right] - \inf_{v \in \mathcal{X}^n} E \left[ \sum_{s=1}^{t} f_t(v) \right] \right)
\leq \sum_{i=1}^{T} \frac{1}{T} \hat{R}(x(s), t).
\]
The desired result is derived from the result in Theorem 3 in conjunction with the relation \( \sum_{i=1}^{T} 1/\sqrt{i} \leq 2\sqrt{T} \). \( \square \)

**APPENDIX H**

**PROOF OF LEMMA 6**

**Proof.** Recalling the definitions of \( z_i(t) \) and \( \bar{z}(t) \), it holds that, for \( t \in \mathcal{Z}_0 \),
\[
\sum_{i=1}^{n} \left\| \frac{z_i(t)}{w_i(t)} - \bar{z}(t) \right\|^2 = \sum_{i=1}^{n} \sum_{k=1}^{n} \left\| \frac{z_i^k(t)}{w_i(t)} - \bar{z}^k(t) \right\|^2. \tag{41}
\]
Then, recalling the update of \( w \)-variable in (19), it holds that, for \( i \in \mathcal{V} \) and \( t \in \mathcal{Z}_+ \),
\[
w_i(t) = \sum_{j=1}^{n} [A(t - 1 : 0)]_{ij} w_j(0) = \sum_{j=1}^{n} [A(t - 1 : 0)]_{ij}. \tag{42}
\]
Using (19, 23) and (42) gives
\[
\frac{z_i^k(t)}{w_i(t)} - \bar{z}^k(t)
= \frac{\sum_{s=0}^{t-1} [A(t - 1 : s+1)]_{ik} u_k(s) - \sum_{j=1}^{n} [A(t - 1 : 0)]_{ij}}{\sum_{j=1}^{n} [A(t - 1 : 0)]_{ij}} - \frac{\sum_{j=1}^{n} [A(t - 1 : 0)]_{ij}}{\sum_{j=1}^{n} [A(t - 1 : 0)]_{ij}}
+ \frac{\sum_{s=0}^{t-1} [A(t - 1 : s+1)]_{ik} \eta_k(s) - \sum_{j=1}^{n} [A(t - 1 : 0)]_{ij}}{\sum_{j=1}^{n} [A(t - 1 : 0)]_{ij}} - \frac{\sum_{j=1}^{n} [A(t - 1 : 0)]_{ij}}{\sum_{j=1}^{n} [A(t - 1 : 0)]_{ij}}
= \sum_{s=0}^{t-1} \sum_{j=1}^{n} [A(t - 1 : s+1)]_{ik} - \sum_{s=0}^{t-1} \sum_{j=1}^{n} [A(t - 1 : 0)]_{ij}
+ \sum_{s=0}^{t-1} \sum_{j=1}^{n} [A(t - 1 : s+1)]_{ik} - \sum_{s=0}^{t-1} \sum_{j=1}^{n} [A(t - 1 : 0)]_{ij},
\]
where the last equality follows from \( n [A(t - 1 : s+1)]_{ik} = \sum_{s=1}^{n} [A(t - 1 : s+1)]_{ik} \). Before proceeding, we introduce a property of \( A(t) \) from (4) as follows:
\[
[A(t : s)]_{ij} - \phi_i(t) \leq \beta \lambda^{t-s}, \tag{43}
\]
where \( \phi_i(t) \) is from a sequence of stochastic vectors \( \{\phi(t)\} \). Moreover, we define \( \gamma \triangleq \inf_{t \in \mathcal{Z}_0} \min_{i \in \mathcal{V}} [A(t : 0)]_{ij} \). Then, we can bound \( \frac{z_i^k(t)}{w_i(t)} - \bar{z}^k(t) \) as shown in (44) at the top of the next page, where the last inequality is due to
\[ \left| \frac{z^k(t)}{w(t)} - z^k(t) \right| \leq \sum_{s=0}^{t-1} \left| u_k(s) \right| \left( \sum_{i=1}^{n} \left| A(t : 1 : s + 1) \right| + \sum_{j=1}^{m} \left| A(t : 1 : 0) \right| \right) \]

\[ \leq \sum_{s=0}^{t-1} \left| u_k(s) \right| \left( \frac{2\lambda^{t-s-2}}{\gamma} + \frac{\beta \lambda^{t-1}}{\gamma} \right) + \frac{\beta \lambda^{t-1}}{\gamma} + \frac{\beta \lambda^{t-1}}{\gamma} \]

\[ \leq \sum_{s=0}^{t-1} \left| u_k(s) \right| \left( \frac{2\beta \lambda^{t-s-2}}{\gamma} + \frac{\beta \lambda^{t-1}}{\gamma} \right) + \frac{\beta \lambda^{t-1}}{\gamma} \]

\[ \leq \sum_{s=0}^{t-1} \left| u_k(s) \right| \frac{2\beta \lambda^{t-s-2}}{\gamma} + \frac{1}{n} \sum_{s=0}^{t-1} \left| u_k(s) \right| \frac{2\beta \lambda^{t-1}}{\gamma}. \]

(44)

\[ \lambda^{t-s-a} \geq \lambda^{t-1} \] with \( a = 1, 2 \) for all \( s = 0, \ldots, t-1 \). Further, using \( (a + b)^2 \leq 2a^2 + 2b^2 \), \( a, b \in \mathbb{R} \), we have

\[ \left| \frac{z^k(t)}{w(t)} - z^k(t) \right|^2 \leq 2 \sum_{s=0}^{t-1} \frac{2\beta \lambda^{t-s-2}}{\gamma} \max_s \left| u_k(s) \right|^2 \]

\[ + 2 \sum_{s=0}^{t-1} \frac{\beta \lambda^{t-1}}{\gamma} \left( \frac{1}{n} \sum_{i=1}^{n} \left| \eta_i^k(s) \right|^2 \right) \]

\[ \leq \frac{8\beta^2 \bar{L}^2}{\gamma^2 \lambda^2 (1 - \lambda)^2} + \frac{8\beta^2}{\gamma^2 (1 - \lambda)^2} \max_s \left( \sum_{i=1}^{n} \left| \eta_i^k(s) \right|^2 \right). \]

Applying the relation (41) gives

\[ \sum_{i=1}^{n} \left| \frac{z_i(t)}{w_i(t)} - \bar{z}(t) \right|^2 \leq \frac{8n^2 \beta^2 \bar{L}^2}{\gamma^2 \lambda^2 (1 - \lambda)^2} + \frac{8n^2 \beta^2}{\gamma^2 (1 - \lambda)^2} \max_s \left( \sum_{i=1}^{n} \left| \eta_i^k(s) \right|^2 \right). \]

(45)

The desired result follows by taking the expectation on (45) and using (46).

\[ \text{APPENDIX I} \]
\[ \text{PROOF OF THEOREM 5} \]

\[ \text{Proof.} \] Note that (E1), (E2), and (E4) in Theorem 1 apply to DPSDA-PS as well. So our main task is to solve (E3). From Lema 5(a), we have same statements as follows:

\[ x(t) \triangleq \Pi_k^\psi \left( \bar{z}(t), \alpha(t - 1) \right) = \Pi_k^\psi \left( \frac{1}{n} \sum_{s=0}^{t-1} \sum_{i=1}^{n} \eta_i^k(s) + \sum_{s=0}^{t-1} u(s), \alpha(t - 1) \right). \]

Recalling Proposition 1, we obtain

\[ \| x(t) - y_i(t) \|

\[ = \| \Pi_k^\psi \left( \bar{z}(t), \alpha(t - 1) \right) - \Pi_k^\psi \left( \frac{z_i(t)}{w_i(t)}, \alpha(t - 1) \right) \|

\[ \leq \alpha(t - 1) \left\| \frac{z_i(t)}{w_i(t)} - \bar{z}(t) \right\|. \]

(46)