# Differentially Private Objective Functions in Distributed Cloud-based Optimization

Yu Wang, Matthew Hale, Magnus Egerstedt and Geir E. Dullerud

***Abstract*— In this work, we study the problem of keeping the objective functions of individual agents $\varepsilon$-differentially private in cloud-based distributed optimization, where agents are subject to global constraints and seek to minimize local objective functions. The communication architecture between agents is cloud-based – instead of communicating directly with each other, they coordinate by sharing states through a trusted cloud computer. In this problem, the difficulty is twofold: the objective functions are used repeatedly in every iteration, and the influence of perturbing them extends to other agents and lasts over time. To solve the problem, we analyze the propagation of perturbations on objective functions over time, and derive an upper bound on them. With the upper bound, we design a noise-adding mechanism that randomizes the cloud-based distributed optimization algorithm to keep the individual objective functions $\varepsilon$-differentially private. In addition, we study the trade-off between the privacy of objective functions and the performance of the new cloud-based distributed optimization algorithm with noise. We present simulation results to numerically verify the theoretical results presented.**

## I. Introduction

Distributed optimization problems appear in various applications [1]. Applications that make use of distribution optimization include power systems [2], [3], communications [4], [5], [6], signal processing [7], [8], sensor networks [9], [10], and machine learning [11], [12]. In these applications it is common to have agents in a network directly share information with each other in the process of optimizing.

However, in some distributed systems, when the agents share information with each other, a major concern is the privacy of their personal data. A common framework for privacy is the concept of $\varepsilon$-differential privacy [13], [14]. Generally speaking, the personal information of an agent is $\varepsilon$-differentially private if its change does not cause significant differences in the observables of the system, and hence cannot be detected by outside observation.

Recently, there has been increased interest in applying the concept of differential privacy in the context of distributed optimization [15], [16], [17], [18]. The general goal is to design distributed optimization algorithms that keep the personal data of each agent $\varepsilon$-differentially private, while still giving results reasonably close to the optima. Usually in these studies, higher degrees of privacy result in further deviation from the optima and *vice versa*. This trade-off between privacy and performance is even a common phenomenon in the broader context of privacy independent of optimization [19], [20].

Unlike most of previous works focusing on keeping the states of the agents private, in this work, we study the problem of keeping each agent's objective (or utility) functions $\varepsilon$-differentially private. While an agent's state can reveal an agent's location and perhaps what it is doing, an agent's objective function can reveal what it values and, as a result, what it is thinking or intending to do. For example, knowing that someone is trying to minimize the cost or time required to travel to some destination gives a strong indication of their future location. The challenge here is that adding independent noise at each time can only keep the objective functions differential privacy for a finite time horizon [21], otherwise, these independent noise will cancel out and reveal the objective functions [22]. To circumvent this trouble, a method of perturbing directly the objective functions has been proposed [22]. This method is suitable for a distributed implementation, though it applies only to strongly convex objective functions. Work presented in [21] is likewise well-suited to distributed problems and focuses on privacy for additive terms in affine objective functions.

In this work, we are interested in keeping objective functions private in multi-agent linear programs and present results targeting such problems. We propose a centralized mechanism for protecting the differential privacy of objective functions for long time horizon by adding completely correlated noise, in the framework of a cloud-based architecture. This architecture is inspired by real-world systems, and provides a mix of centralized and decentralized computations that lends itself to a variety of multi-agent problems [23], [24]. In the cloud-based architecture, each agent possesses an objective function to minimize, while subject to global constraints. To coordinate on the constraints, the agents share their current states through the cloud, instead of communicating directly with each other.

The rest of the paper is organized as follows. In Section II, the non-differentially private version of cloud-based distributed optimization algorithm is formally presented. In Section III, we incorporate privacy into the cloud-based distributed optimization algorithm by adding noise and give the definition of $\varepsilon$-differential privacy for the new algorithm. In Section IV, we design a noise-adding mechanism for the cloud-based distributed algorithm that preserves the privacy of objective functions. In Section V, the trade-off between the privacy of the objective functions and the performance of the

Y. Wang and G. E. Dullerud are with the Coordinated Science Laboratory and the Department of Mechanical Science and Engineering, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA. {yuwang8, dullerud}@illinois.edu

M. Hale and M. Egerstedt are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA. {matthale, magnus}@gatech.edu

noisy optimization algorithm is studied for the noise-adding mechanism proposed. Section VI then presents simulation results to demonstrate the bounds on performance we derive. Finally, we summarize the work and point out possible further extensions in Section VII.

## II. Problem Formulation

### A. Notations

In this paper, we denote the set of *natural*, *real*, and *non-negative real* numbers by $\mathbb{N}$, $\mathbb{R}$ and $\mathbb{R}_{\geq 0}$ respectively. The imaginary unit is denoted by $\mathrm{i} = \sqrt{-1}$. For any positive integer $n$, let $[n] = \{1, \ldots, n\}$. The Euclidean norm on $\mathbb{R}^n$ is denoted by $\|\cdot\|$. A finite sequence with index from 1 to $T$ is abbreviated by $v^{(T)} = \{v(1), \ldots, v(T)\}$. An infinite sequence $\{v(t)\}_{t=1}^{\infty} \subseteq \mathbb{R}_{\geq 0}$ is *dominated* by another infinite sequence $\{u(t)\}_{t=1}^{\infty} \subseteq \mathbb{R}_{\geq 0}$ as $t \to \infty$, written as $v \preceq u$, if $\lim_{t\to\infty} v/u \leq 1$. A finite sequence of 0 is also denoted by 0 when there is no ambiguity. Given a set $X \subseteq \mathbb{R}^n$, we denote the projection of a point $a \in \mathbb{R}^n$ onto the set by $\Pi_X(a)$. The set of *second-order continuously differentiable convex* functions is denoted by $\mathcal{C}^2$. The *expectation* and *variance* of a random variable $x$ are denoted by $\mathbb{E}[x]$ and $\mathrm{Var}[x]$, respectively.

### B. Distributed Optimization

The problem studied in this work derives from [24]. Consider a discrete- time distributed system of $N$ agents indexed over $I = \{1, 2, \ldots, N\}$. The state $x_i$ of the $i$th agent takes value in a *non-empty*, *compact*, and *convex* set $\mathcal{X}_i \subseteq \mathbb{R}^{n_i}$. The *ensemble state* of the system is given by

$$x = [x_1^{\mathrm{T}}, x_2^{\mathrm{T}}, \ldots, x_N^{\mathrm{T}}]^{\mathrm{T}} \in \mathcal{X} \subseteq \mathbb{R}^n, \tag{1}$$

where $n = \sum_{i=1}^{N} n_i$ and $\mathcal{X} = \Pi_{i=1}^{N} \mathcal{X}_i$.

Each agent seeks to minimize a local $\mathcal{C}^2$ *objective function*

$$f_i : \mathbb{R}^{n_i} \to \mathbb{R}, \quad i \in [N], \tag{2}$$

while subject to $M$ global $\mathcal{C}^2$ *constraints* $g_j(x) \leq 0$, where

$$g_j : \mathbb{R}^n \to \mathbb{R}, \quad j \in [M]. \tag{3}$$

As with other related works, we make the following assumption on the constraints.

*Assumption 1:* The constraints satisfy Slater's condition: there exists an ensemble state $x \in \mathcal{X}$ such that every constraint is strictly satisfied, namely $g_j(x) < 0$ for $j \in [M]$.

As with other related works in differentially private optimization [25], [26], the constraints should be *Lipschitz continuous*.

*Assumption 2:* For each $j \in [M]$, there exist constants $l_{i,j} \in \mathbb{R}_{\geq 0}$ for $i \in [N]$ such that

$$\left\| \frac{\partial g_j}{\partial x_i} \right\| \leq l_{i,j}. \tag{4}$$

We call $l_{i,j}$ the $i$th *Lipschitz constant*, and $l_j = \max_{i\in[N]} l_{i,j}$ the *overall Lipschitz constant* for $g_j(x)$.

*Cloud-based architecture:* Due to the fact that the objective functions are local while the constraints are global, we employ a cloud-based communication architecture to coordinate the agents while they optimize [24]. Specifically, the agents do not communicate directly with each other; instead, to enforce privacy of the agents' objective functions, they coordinate with each other by communicating through the cloud. At each timestep, the agents collect information from cloud, update their states, and send them back to the cloud. Meanwhile, the central server in the cloud gathers the new states and computes the new information required by the agents to re-update their states in the next round. It will be shown in Section III how these computations in the cloud are made private.

For the whole system, optimizing every $f_i(x_i)$ simultaneously is equivalent to optimizing the *global objective function*

$$f(x) = \sum_{i=1}^{N} f_i(x_i). \tag{5}$$

For this global optimization problem, the Lagrangian is

$$L(x, \mu) = f(x) + \mu^{\mathrm{T}} g(x), \tag{6}$$

where $\mu$ is a vector of Kuhn-Tucker (KT) multipliers that takes values in $\mathcal{M} = \mathbb{R}_{\geq 0}^{M}$. We minimize $f$ subject to $g(x) \leq 0$ by using the gradient-based algorithm of Bakushinskii and Polyak presented in [27], which takes the form

$$x \leftarrow \Pi_{\mathcal{X}} \left[ x - \gamma_t \left( \frac{\partial f}{\partial x}(x) + \frac{\partial g}{\partial x}(x)^T \mu + \alpha_t x \right) \right] \tag{7}$$

$$\mu \leftarrow \Pi_{\mathcal{M}} \left[ \mu + \gamma_t \left( g(x) - \alpha_t \mu \right) \right], \tag{8}$$

where the time-dependent parameters $\alpha_t$ and $\gamma_t$ asymptotically vanish. We choose this algorithm as the basis for the current paper due to the noise-rejecting properties it provides and the ability to naturally distribute it over a network, and these properties will be exploited when noise is added for privacy.

In this problem we will consider separable constraints. A careful inspection of (7) and (8) shows that this algorithm is therefore compatible with the cloud-based architecture. While (8) is computed globally from $x$ and $\mu$ by the central server, (7) can be computed locally by each agent given $\mu$. Therefore, we derive the cloud-based optimization Algorithm 1.

The convergence of Algorithm 1 is guaranteed by the following theorem from [28], Theorem 6.

*Theorem 1:* Algorithm 1 converges to an optimum as the times of iteration $T \to \infty$, if the time-dependent parameters $\alpha_t$ and $\gamma_t$ satisfy

$$\gamma_t = \gamma_1 t^{-c_1}, \quad \alpha_t = \alpha_1 t^{-c_2}, \tag{9}$$

where $\alpha_1, \gamma_1 > 0$, $c_1 > c_2 > 0$, and $c_1 + c_2 < 1$.

Convergence rates for this algorithm can range from linear to geometric [29] and depend upon the specific problem under consideration, as well as the choices of $\alpha_t$ and $\gamma_t$.

**Algorithm 1** Non-differentially Private Cloud-based Optimization

```
Require: $x_0 \in \mathcal{X}$, $\mu_0 \in \mathcal{M}$, parameters $\alpha_t, \gamma_t$, times of iteration $T$.
  $x_i \leftarrow x_i(0)$, $\mu \leftarrow \mu_0$
  for $t = 1$ to $T$ do
    $\mu' \leftarrow \Pi_{\mathcal{M}}\left[\mu + \gamma_t\left(g(x) - \alpha_t \mu\right)\right]$
    for $i = 1$ to $N$ do
      $x_i \leftarrow \Pi_{\mathcal{X}_i}\left[x_i - \gamma_t\left(\frac{\partial f_i}{\partial x_i}(x_i) + \mu\frac{\partial g}{\partial x_i}(x_i) + \alpha_t x_i\right)\right]$
    end for
    $\mu \leftarrow \mu'$
    $t \leftarrow t + 1$
  end for
```

## III. Differential Privacy

When the agents communicate through the cloud, privacy is a major concern. A commonly used means for keeping data private is the concept of $\varepsilon$-differential privacy. But, unlike a number of other efforts to keep the agents' state $x_i$ private, here we try to keep each agent's objective function $f_i$ private using differential privacy.

In the distributed system, the information from the central server $\mu(t)$ is public. The objective function $f_i$ that we want to keep private is chosen correspondingly from a parametrized family of functions

$$\mathcal{F}_i = \{f_i(\cdot; a_i) \mid a_i \in \mathcal{A}_i\}, \quad i \in [N], \tag{10}$$

where each parameter set $\mathcal{A}_i$ is equipped with a *metric*

$$d_i(\cdot,\cdot) : \mathcal{A}_i \times \mathcal{A}_i \to \mathbb{R}_{\geq 0}, \quad i \in [N]. \tag{11}$$

### A. Technical Assumptions

For now on, we make the following two assumptions.

- The parametrized family of objective functions contain only linear objective functions, namely

  $$\mathcal{F}_i = \left\{f_i(x) = a_i^{\mathrm{T}} x_i \mid a_i \in \mathbb{R}^{n_i}\right\}, \quad i \in [N]. \tag{12}$$

  The distance on the parameters inherits from the Euclidean norm on $\mathbb{R}^{n_i}$,

  $$d_i(x, y) = \|x - y\|. \tag{13}$$

- There is only one linear constraint,

  $$g(x) = b^{\mathrm{T}} x = \sum_{i=1}^{N} b_i^{\mathrm{T}} x_i, \tag{14}$$

  where $b = [b_1^{\mathrm{T}}, \dots b_N^{\mathrm{T}}]^{\mathrm{T}}$. The $i$th *Lipschitz constant* for $g(x)$ is $l_i = \|b_i\|$, and the *overall Lipschitz constant* is $l = \max_{i \in [N]} l_i$.

These two assumptions are only to simplify the forthcoming presentation of results – the theoretical results presented in the following sections are still valid when the objective functions $f_i(x_i)$ and the constraint $g(x)$ are $\mathcal{C}^2$ and Lipschitz continuous.

### B. Noise-adding Mechanism

To keep the objective functions $\varepsilon$-differentially private, we consider the mechanism of adding noise to the public multiplier $\mu$, as shown in Algorithm 2. At each iteration $t$, the central server in the cloud adds a zero mean noise to the public multiplier by

$$\mu \leftarrow \mu + v(t), \quad t \in [T] \tag{15}$$

before sending it to the agents. The noise $v(t)$ added over time can be correlated.

**Algorithm 2** Differentially Private Cloud-based Distributed Optimization

```
Require: $x_0 \in \mathcal{X}$, $\mu_0 \in \mathcal{M}$, parameters $\alpha_t, \gamma_t$, number of iterations $T$.
  $x_i \leftarrow x_i(0)$, $\mu \leftarrow \mu_0$
  for $t = 1$ to $T$ do
    $\mu' \leftarrow \Pi_{\mathcal{M}}\left[\mu + \gamma_t\left(b^{\mathrm{T}} x - \alpha_t \mu\right)\right]$
    for $i = 1$ to $N$ do
      $x_i \leftarrow \Pi_{\mathcal{X}_i}\left[x_i - \gamma_t\left(a_i + \mu b_i + \alpha_t x_i\right)\right]$
    end for
    $\mu \leftarrow \mu' + v(t)$
    $t \leftarrow t + 1$
  end for
```

Compared to some other mechanisms that add noise to the states or the objective functions of every agent, this mechanism is much easier to implement in practice because only the cloud needs to add noise to its computations.

### C. Differential Privacy with Metric

The main ingredients for defining $\varepsilon$-differential privacy are *private data* and *observables*. The private data here is the set of objective functions $D = \{f_i(x_i) = a_i^{\mathrm{T}} x_i\}_{i=1}^N \in \Pi_{i=1}^N \mathcal{F}_i$. Two sets of private data are *adjacent* if they differ in at most one entry. In addition, we define a distance between two adjacent private data, which derives naturally from the distance between objective functions.

*Definition 1:* Two private data $D = \{f_i(x_i) = a_i^{\mathrm{T}} x_i\}_{i=1}^N$ and $D' = \{f'_i(x_i) = a_i'^{\mathrm{T}} x_i\}_{i=1}^N$ are adjacent if there exists an index $i \in [N]$ such that $a_j = a'_j$ for each $j \in [N] \backslash \{i\}$. In addition, the distance between the two *adjacent* data is defined by

$$d(D, D') = \|a_i - a'_i\|, \tag{16}$$

where $\|\cdot\|$ is the Euclidean norm on $\mathbb{R}^{n_i}$

The observable in the distributed system is the sequence of public multipliers $\mu^{(T)} = \{\mu(1), \dots, \mu(T)\}$, where $\mu(t) \in \mathcal{M}$ for all $t \in [T]$. It is determined by three factors:

- the choice of initial condition $x_0 \in \mathcal{X}$, $\mu_0 \in \mathcal{M}$,
- the noise-adding mechanism, namely the probability distribution of $v^{(T)} = \{v(1), \dots, v(T)\}$ in Algorithm 2,
- and the private data $D$.

*Remark 2:* Therefore, when fixing the initial condition $x_0 \in \mathcal{X}$ and $\mu_0 \in \mathcal{M}$, the private data $D$, and the noise-adding mechanism $v^{(T)}$, $\mu^{(T)}$ is a sequence in $\mathbb{R}$, which we denote by $(\mu_{D,v^{(T)}}^{x_0,\mu_0})^{(T)}$; when only fixing the initial condition $x_0 \in \mathcal{X}$ and $\mu_0 \in \mathcal{M}$ and the private data $D$, $\mu^{(T)}$ is a random process in $\mathbb{R}$, which we denote by $(\mu_D^{x_0,\mu_0})^{(T)}$.

In this work, we use the metric version of $\varepsilon$-differential privacy from [30], which is stronger than the commonly used non-metric version of $\varepsilon$-differential privacy.

*Definition 2:* The distributed system is $(\varepsilon, \delta)$-differentially private, if for any choice of initial condition $x_0 \in \mathcal{X}$, $\mu_0 \in \mathcal{M}$, for any two sets of *adjacent* private data $D, D'$, and for any possible set of observations $\mathcal{O}$, the inequality

$$\mathbb{P}\left[(\mu_D^{x_0,\mu_0})^{(T)} \in \mathcal{O}\right] \leq e^{\varepsilon d(D,D')}\mathbb{P}\left[(\mu_{D'}^{x_0,\mu_0})^{(T)} \in \mathcal{O}\right] + \delta \tag{17}$$

holds, where $(\mu_D^{x_0,\mu_0})^{(T)}$ and $(\mu_{D'}^{x_0,\mu_0})^{(T)}$ are random processes as explained in Remark 2. When $\delta = 0$, we call the distributed system $\varepsilon$-differentially private.

### D. Influence of Noise on Performance

Generally, adding noise $v^{(T)}$ to Algorithm 2 weakens its ability to converge to optima of the cloud-based distributed optimization problem. We measure this *loss of performance* by the difference of the result derived by Algorithm 2 and the result derived by Algorithm 1.

*Definition 3:* The loss of performance for private data $D$ in Algorithm 2 due to the noise is defined by

$$\Lambda_D = \max_{\substack{x_0 \in \mathcal{X}\\ \mu_0 \in \mathcal{M}}} \mathrm{Var}_{v^{(T)}}\left[\mu_{D,v^{(T)}}^{x_0,\mu_0}(T) - \mu_{D,0}^{x_0,\mu_0}(T)\right], \tag{18}$$

where $\mu_{D,v^{(T)}}^{x_0,\mu_0}(t), \mu_{D',0}^{x_0,\mu_0}(t)$ are the public multipliers at time $T$ generated by Algorithm 2 with initial condition $x_0, \mu_0$, set of objective functions $D$, and noise $v^{(T)}$ and 0, respectively.

## IV. Differentially Private Noise-adding Mechanism

The difficulty of keeping the objective functions $\varepsilon$-differentially private lies in the fact that they are used repeatedly at every iteration. As observations accumulate, the blurring effect of noise can be weakened. In addition, the influence of perturbing an objective function even temporarily extends to other agents and remains over time. Furthermore, the noise added to the multipliers to keep them $\varepsilon$-differentially private also remains over time.

To solve this problem, we analyze how the non-differentially private cloud-based optimization algorithm responds to perturbation on the objective functions, and then design noise to cover this perturbation. The approach is an extension to the commonly used *sensitivity analysis*.

### A. Temporary Perturbation on Objective Functions

To begin with, we study the case of perturbing the parameter $a_i$ of the objective function $f_i(x_i) = a_i^{\mathrm{T}} x_i$ temporarily at time $s$, when no noise is added to the algorithm. By Algorithm 2, this is equivalent to applying the same perturbation directly to the states $x_i$ at the same time, but scaled by $\gamma_t/(1 - \alpha_t\gamma_t)$. Though the perturbation is temporary, its influence remains after time $s$.

Formally, setting $v^{(t)} = 0$, the updating law of Algorithm 2 is written as

$$\begin{bmatrix} x \\ \mu \end{bmatrix} \leftarrow \Pi_{\mathcal{X}\times\mathcal{M}}\left[\begin{bmatrix}(1-\alpha_t\gamma_t)I_N & -\gamma_t b \\ \gamma_t b^{\mathrm{T}} & 1-\alpha_t\gamma_t\end{bmatrix}\begin{bmatrix} x \\ \mu \end{bmatrix} - \begin{bmatrix}\gamma_t a \\ 0\end{bmatrix}\right], \tag{19}$$

where $a = [a_1^{\mathrm{T}}, \ldots, a_N^{\mathrm{T}}]^{\mathrm{T}}$ and $I_N$ is the $N \times N$ identity matrix. Therefore, by the non-expansiveness of the projection $\Pi_{\mathcal{X}\times\mathcal{M}}$, when there is a perturbation $\tau$ on the state $x$ and a perturbation $\delta$ on the multiplier $\mu$ at time $t$, the perturbation propagated to the next iteration is bounded by

$$\begin{bmatrix}(1-\alpha_t\gamma_t)I_N & -\gamma_t b \\ \gamma_t b^{\mathrm{T}} & 1-\alpha_t\gamma_t\end{bmatrix}\begin{bmatrix}\tau \\ \delta\end{bmatrix}. \tag{20}$$

In this case, we start with the perturbation

$$\begin{aligned} \delta &= 0, \\ \tau &= [0, \ldots, 0, 1, 0, \ldots, 0]^{\mathrm{T}}. \end{aligned} \tag{21}$$

where "1" is the $i$th element, and want to design a mechanism that adds noise only to $\mu$ to cover its influence over time.

To cover the persistent influence of perturbing an objective function temporarily at time $s$, we add noise to the public multiplier bite by bite over time. As shown in Figure 1, the perturbation in $x(s)$ propagates to both $x(s+1)$ and $\mu(s+1)$. This influence on $\mu(s+1)$ can be covered by directly adding Laplace noise to it, while the influence on $x(s+1)$ will be covered by adding noise on $\mu(s+2), \mu(s+3), \ldots$ later on. Since all the influences are generated by a single perturbation, we choose to add completely correlated noise over time.

$$\begin{tikzcd}
 & \mu(s+1) & \mu(s+2) & \cdots \\
x(s) \arrow[r] \arrow[ur] & x(s+1) \arrow[r] \arrow[ur] & x(s+2) \arrow[r] \arrow[ur] & \cdots
\end{tikzcd}$$

Fig. 1. To cover the persistent influence of perturbing an objective function temporarily at time $s$, we add noise at each time $t \geq s+1$ that just covers the proportion of the perturbation that propagates through the solid lines.

This idea yields the following $\varepsilon$-differentially private mechanism to cover a temporary perturbation on an arbitrary objective function $f_i(x_i)$.

*Mechanism 1:* In Algorithm 2, add completely correlated Laplacian noise

$$v_s(t) = \begin{cases} 0, & \text{if } 1 \leq t \leq s \\ \gamma_s\gamma_{s+1} l w, & \text{if } t = s+1 \\ \gamma_s\gamma_t\left[\Pi_{k=s+1}^{t-1}(1-\alpha_k\gamma_k)\right] l w, & \text{if } t \geq s+2 \end{cases}. \tag{22}$$

where $\alpha_t, \gamma_t$ are given in (9), $l$ is the *overall Lipschitz constant* for the constraint $g(x)$, and $w \sim \mathrm{Lap}(1/\varepsilon)$ is a Laplacian noise.

The following lemma gives the asymptotic behavior of noise $v_s(t)$ in Mechanism 1.

*Lemma 3:* The noise $v_s(t)$ in Mechanism 1 obeys

$$v_s(t) \leq \gamma_1^2 (st)^{-c_1} \exp\left(-\alpha_1\gamma_1 \frac{t^{1-c_1-c_2} - s^{1-c_1-c_2}}{1-c_1-c_2}\right) l w, \tag{23}$$

when $t \to \infty$.

### B. Constant Perturbation on Objective Functions

When the perturbation on the parameter $a_i$ of the objective function $f_i(x_i) = a_i^{\mathrm{T}} x_i$ at time $s$ is constant, by the setup of Algorithm 2, the influence is no larger than adding the influence of temporary perturbations at all iterations. Therefore, the following noise-adding mechanism keeps each objective function $f_i(x_i)$ $\varepsilon$-differentially private for constant perturbation. Again, since all the influences are generated by a single constant perturbation, we choose to add completely correlated noise over time.

*Mechanism 2:* In Algorithm 2, add completely correlated Laplacian noise

$$v(t) = \sum_{s=1}^{t-1} v_s(t) =$$
$$\begin{cases} 0, & \text{if } t = 1 \\ \gamma_1 \gamma_2 l w, & \text{if } t = 2 \\ \gamma_t \left( \gamma_{t-1} + \sum_{s=1}^{t-1} \gamma_s \Pi_{k=s+1}^{t-1} (1 - \alpha_k \gamma_k) \right) l w, & \text{if } t \geq 3 \end{cases}. \tag{24}$$

where $\alpha_t, \gamma_t$ are given in (9), $l$ is the *overall Lipschitz constant* for the constraint $g(x)$, and $w \sim \mathrm{Lap}(1/\varepsilon)$ is a Laplacian noise.

*Theorem 4:* The noise-adding Mechanism 2 keeps the objective functions $\varepsilon$-differentially private in Algorithm 2.

The asymptotic behavior of $v(t)$ as $t \to \infty$ is given by the following theorem.

*Theorem 5:* In Mechanism 2, the noise added at time $t$ is dominated by

$$v(t) \preceq \frac{\gamma_1}{\alpha_1} t^{-(c_1 - c_2)} l w, \tag{25}$$

where constants $c_1 > c_2$ are given in (9). Thus, $v(t)$ converges to 0 as $t \to \infty$.

## V. Trade-off between Privacy and Performance

To keep the objective functions in the cloud-based distributed optimization algorithm $\varepsilon$-differentially private, we add noise to the sequence of public multipliers $\mu^{(T)}$ according to Mechanism 2. This noise will prevent Algorithm 2 from attaining the optima. In this section, though achieving the exact optima is not always possible, the result derived by Algorithm 2 will stay in the vicinity of the original optima in a probabilistic sense.

### A. Response to Perturbation on Public Multipliers

To begin with, we compute the influence of perturbing the public multiplier $\mu(s)$ temporarily at time $s$ on the state of the $i$th agent at time $t$. Let $(\mu_{D,v^{(T)}}^{x_0,\mu_0})^{(T)}$ be the sequence of public multipliers generated by Algorithm 2 with noise-adding mechanism $v^{(T)}$, set of objective functions $D$ and initial condition $x_0, \mu_0$. The *s-perturbation* of $(\mu_{D,v^{(T)}}^{x_0,\mu_0})^{(T)}$ is derived by giving it an at-most-unit-in-norm perturbation at time $s$.

*Definition 4:* An $s$-perturbation $\nu^{(T)}$ of $(\mu_{D,v^{(T)}}^{x_0,\mu_0})^{(T)}$ is a sequence of public multipliers such that

- for $k \leq s-1$, $\nu(k) = \mu_{D,v^{(T)}}^{x_0,\mu_0}(k)$,
- $\|\nu(s) - \mu_{D,v^{(T)}}^{x_0,\mu_0}(s)\| \leq 1$,
- for $k \geq s+1$, $\nu^{(T)}$ evolves by the same law as $(\mu_{D,v^{(T)}}^{x_0,\mu_0})^{(T)}$, namely by Algorithm 2 with noise-adding mechanism $v^{(T)}$, set of objective functions $D$ and initial condition $x_0, \mu_0$.

The set of $s$-perturbations of $(\mu_{D,v^{(T)}}^{x_0,\mu_0})^{(T)}$ is denoted by $\mathcal{Q}^s(x_0, \mu_0, D, v^{(T)})$.

Using Definition 4 of $s$-perturbations, we can define the *impulse response* of the $i$th agent for perturbation on the public multiplier $\mu(s)$.

*Definition 5:* The impulse response of agent $i \in [N]$ at time $t \in [T]$ for an $s$-perturbation on the public multiplier is defined by

$$\kappa_{i,s\to t} = \max_{\substack{x_0 \in \mathcal{X},\ \mu_0 \in \mathcal{M} \\ v^{(T)} \subseteq \mathbb{R},\ \nu \in \mathcal{Q}}} |\mu_{D,0}^{x_0,\mu_0}(t) - \nu|, \tag{26}$$

where $\mathcal{Q} = \mathcal{Q}^s(x_0, \mu_0, D, v^{(T)})$ is the set of $s$-perturbations of $(\mu_{D,v^{(T)}}^{x_0,\mu_0})^{(T)}$.

Below, Theorem 6 gives an *approximate upper bound* on the impulse response, and ensures that the upper bound converges under the following condition.

*Condition 1:* The parameters $\alpha_1$, $\gamma_1$ in (9) satisfy

$$\frac{\alpha_1}{\gamma_1} \gg \sum_{i=1}^{N} l_i^2, \tag{27}$$

where $l_i$ is the $i$th Lipschitz constant of the constraint $g(x)$.

*Theorem 6:* The impulse response of agent $i \in [N]$ at time $t \in [T]$ for perturbation on the public multiplier $\mu$ at time $s \leq t$ is bounded by

$$\kappa_{i,s\to t} \leq \Pi_{k=s+1}^{t} r_k, \tag{28}$$

with the *rescaling factor*

$$r_k = \sqrt{(1 - \alpha_k \gamma_k)^2 + \gamma_k^2 \sum_{i=1}^{N} l_i^2}, \tag{29}$$

This upper bound is dominated by

$$\Pi_{k=s+1}^{t} r_k \preceq \exp\left( -\alpha_1 \gamma_1 \frac{t^{1-c_1-c_2} - s^{1-c_1-c_2}}{1 - c_1 - c_2} \right), \tag{30}$$

which converges to 0, as $t \to \infty$ under Condition 1.

### B. Trade-off between Privacy and Performance

To derive an upper bound on the influence of adding noise by Mechanism 2 on the final result derived by Algorithm 2, it suffices to collect the influence of noise added at each iteration.

*Theorem 7:* The loss of performance (see Definition 3) for protecting the private data $D$ with Mechanism 2 is bounded by

$$\Lambda_D \leq \sum_{t=1}^{T-1} v(t)\kappa_{i,t\to T}$$
$$\leq \gamma_1\gamma_2 lw\Pi_{k=3}^{T} r_k +$$
$$\sum_{t=3}^{T-1} \gamma_t \Pi_{k=t+1}^{T} r_k \left(\gamma_{t-1} + \sum_{s=1}^{t-1} \gamma_s \Pi_{k=s+1}^{t-1}(1-\alpha_k\gamma_k)\right), \tag{31}$$

where $\alpha_t, \gamma_t$ and $r_k$ are given in (9) and (29).

Combining Theorem 6 and Theorem 5 gives an explicit upper bound on the loss of performance $\Lambda_D$ under Condition 1.

*Theorem 8:* The loss of performance for private date $D$ is bounded by

$$\Lambda_D \leq \frac{2T^{2c_2}l}{\alpha_1^2\varepsilon^2}, \tag{32}$$

for large $T$, under Condition 1.

By Theorem 8, the upper bound on $\Lambda_D$ depends only on $\alpha_1$ not on $\gamma_1$, because $\gamma_t$ decreases faster and is therefore dominated by $\alpha_t$ in the long run. Since this upper bound is proportional to $T^{2c_2}$, it is preferable to choose $c_2 \ll 1$. Finally, noting that the upper bound is proportional to $1/\varepsilon^2$, we see a trade-off between privacy and performance: when $\varepsilon$ decreases, a higher degree of privacy is achieved at the cost of larger deviation from the optima, and *vice versa*. For this reason, Theorem 8 can roughly be understood as demonstrating that a $T^{2c_2}$ penalty is paid in convergence in exchange for a linear improvement in the privacy parameter.

## VI. Simulation Results

We now present simulation results to demonstrate the preceding bound on the loss of performance when keeping the agents' objectives private. The simulation consists of $N = 6$ agents, each with $x_i \in \mathbb{R}^2$. The values of $a_i$ for the agents' objectives and $b_i$ for the constraint are shown in Table I. The privacy parameter was chosen to be $\varepsilon = \ln 3$. In accordance with Condition 1 the values $\alpha_1 = 1$ and $\gamma_1 = 0.01$ were selected, along with $c_1 = 1/2$ and $c_2 = 2/5$. Together, these give the time-varying regularization and stepsize values

$$\alpha_t = t^{-2/5} \text{ and } \gamma_t = 0.01t^{-1/2}. \tag{33}$$

The initial state and dual value were chosen to be $x(0) = 0$ and $\mu(0) = 0$. Using these problem parameters, two simulation runs were conducted for $T = 2,000$ iterations, one with noise and one without, in order to compare the values of $\mu_{D,v(T)}^{x_0,\mu_0}(t)$ and $\mu_{D,0}^{x_0,\mu_0}(t)$ to assess performance loss.

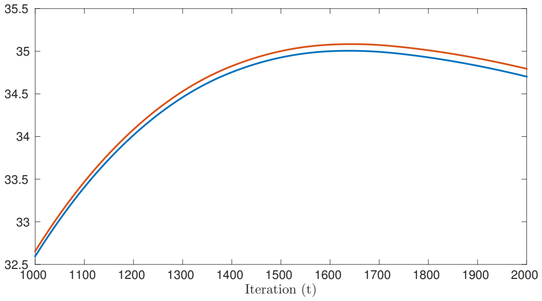

Fig. 2. The dual trajectory $\mu_{D,v(T)}^{x_0,\mu_0}(t)$ from the private optimization run (lower curve), and the dual trajectory $\mu_{D,0}^{x_0,\mu_0}(t)$ from the non-private optimization run (upper curve) for values of $t$ between $1,000$ and $2,000$. The small difference between these two curves indicates that the results of the private optimization algorithm closely match those of the non-private algorithm. Therefore, there is only small performance loss incurred by the addition of privacy to the optimization process.

| $i$ | $a_i$ | $b_i$ |
|---|---|---|
| 1 | $[-50 \quad -50]^T$ | $[1.2 \quad 1.8]^T$ |
| 2 | $[-30 \quad -20]^T$ | $[1.5 \quad 1.3]^T$ |
| 3 | $[-10 \quad -40]^T$ | $[1.0 \quad 0.9]^T$ |
| 4 | $[-10 \quad -60]^T$ | $[1.6 \quad 1.8]^T$ |
| 5 | $[-30 \quad -20]^T$ | $[1.7 \quad 1.4]^T$ |
| 6 | $[-100 \quad -50]^T$ | $[1.8 \quad 1.3]^T$ |

TABLE I

The values of $a_i$ and $b_i$ for $i \in [6]$.

Figure 2 shows the values of $\mu_{D,v(T)}^{x_0,\mu_0}(t)$ (lower curve) and $\mu_{D,0}^{x_0,\mu_0}(t)$ (upper curve) across the range $1000 \leq t \leq 2000$. Here we find that, despite the noise added in accordance with Mechanism 2, the two trajectories are very close, with the distance between them a small fraction of their values. Figure 2 suggests that the level of performance loss between $\mu_{D,v(T)}^{x_0,\mu_0}(t)$ and $\mu_{D,0}^{x_0,\mu_0}(t)$ is quite small, with values differing only slightly across the time horizon shown.

To further this point, Figure 3 shows the value of

$$\left|\frac{\mu_{D,v(T)}^{x_0,\mu_0}(t) - \mu_{D,0}^{x_0,\mu_0}(t)}{\Lambda_D(t)}\right|, \tag{34}$$

with $\Lambda_D(t)$ computed according to Theorem 7 by evaluating $\Lambda_D$ at each point in time during the simulation. In Figure 3 it can be seen that the performance loss incurred by privacy is bounded by approximately $0.05\Lambda_D(t)$ for all time, indicating close agreement between the private run and non-private run. From Figures 2 and 3 we see that privacy of the agents' objective functions is achieved in a way that still allows the optimization to proceed successfully.

## VII. Conclusion

In this work, we studied the problem of randomizing a cloud-based distributed optimization algorithm by adding noise to meet the requirement of $\varepsilon$-differential privacy for individual objective functions. To design such a noise-adding mechanism, we analyzed the impulse responses of the distributed system, derived an upper bound, and then designed a noise-adding mechanism that keeps the objective functions $\varepsilon$-differentially private for the cloud-based distributed optimization algorithm. In addition, we showed that there is a trade-off between the privacy of objective functions and the performance of the randomized cloud-based distributed

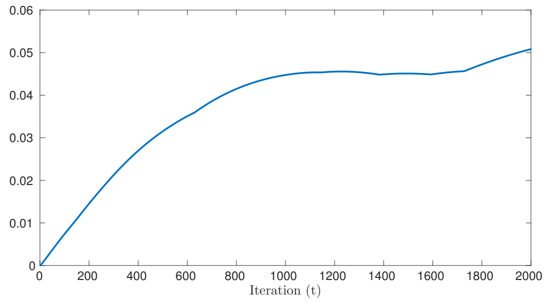

Fig. 3. The value of $\left|\left(\mu^{x_0,\mu_0}_{D,v(T)}(t) - \mu^{x_0,\mu_0}_{D,0}(t)\right)/\Lambda_D(t)\right|$ for times $t$ between $1$ and $2,000$. Here we see that the performance loss is not only bounded above by $\Lambda_D(t)$ as shown in Theorem 7, but that it is almost always bounded above by $0.05\Lambda_D(t)$ in this case. This small value of performance loss indicates that the private optimization algorithm is closely matching the behavior of the non-private algorithm while providing privacy guarantees to the agents' objectives. Therefore, while the introduction of privacy of agents' objectives may impact the results of the optimization, it does so in a way that still produces results that will be within reasonable bounds of error in many applications.

optimization algorithm with noise. Finally, the theoretical results were verified by simulation.

## References

[1] M. Mesbahi and M. Egerstedt, *Graph Theoretic Methods in Multiagent Networks*. Princeton University Press, Jul. 2010.

[2] M. Nazari, Z. Costello, M. Feizollahi, S. Grijalva, and M. Egerstedt, "Distributed frequency control of prosumer-based electric energy systems," *Power Systems, IEEE Transactions on*, vol. 29, no. 6, Nov 2014.

[3] M. Pipattanasomporn, H. Feroze, and S. Rahman, "Multi-agent systems in a distributed smart grid: Design and implementation," in *Power Systems Conference and Exposition, 2009*, March 2009.

[4] P. Xuan, V. Lesser, and S. Zilberstein, "Communication decisions in multi-agent cooperation: Model and experiments," in *Proceedings of the Fifth International Conference on Autonomous Agents*, ser. AGENTS '01. New York, NY, USA: ACM, 2001, pp. 616–623.

[5] R. Becker, A. Carlin, V. Lesser, and S. Zilberstein, "Analyzing myopic approaches for multi-agent communication," *Computational Intelligence*, vol. 25, no. 1, pp. 31–50, 2009.

[6] F. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: shadow prices, proportional fairness and stability," in *Journal of the Operational Research Society*, vol. 49, 1998.

[7] S. White, "Applications of distributed arithmetic to digital signal processing: a tutorial review," *ASSP Magazine, IEEE*, vol. 6, no. 3, pp. 4–19, July 1989.

[8] A. Dimakis, S. Kar, J. Moura, M. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proceedings of the IEEE*, vol. 98, no. 11, pp. 1847–1864, Nov 2010.

[9] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 2. IEEE, 2002, pp. 1327–1332.

[10] N. Trigoni and B. Krishnamachari, "Sensor network algorithms and applications Introduction," *Phil. Trans. of the Royal Scoeity A - Mathematical, Physical, and Engineering Sciences*, 2012.

[11] L. Panait and S. Luke, "Cooperative multi-agent learning: The state of the art," *Autonomous Agents and Multi-Agent Systems*, vol. 11, no. 3, pp. 387–434, Nov. 2005.

[12] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, Jan. 2011.

[13] C. Dwork, "Differential Privacy," in *Automata, Languages and Programming*, ser. Lecture Notes in Computer Science, M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener, Eds. Springer Berlin Heidelberg, Jul. 2006, no. 4052, pp. 1–12.

[14] ——, "Differential privacy: A survey of results," in *Theory and applications of models of computation*. Springer Berlin Heidelberg, 2008, pp. 1–19.

[15] Z. Huang, S. Mitra, and G. Dullerud, "Differentially Private Iterative Synchronous Consensus," in *Proceedings of the 2012 ACM Workshop on Privacy in the Electronic Society*, ser. WPES '12. New York, NY, USA: ACM, 2012, pp. 81–90.

[16] Z. Huang, Y. Wang, S. Mitra, and G. E. Dullerud, "On the Cost of Differential Privacy in Distributed Control Systems," in *Proceedings of the 3rd International Conference on High Confidence Networked Systems*, ser. HiCoNS '14. New York, NY, USA: ACM, 2014, pp. 105–114.

[17] Z. Huang, S. Mitra, and N. Vaidya, "Differentially Private Distributed Optimization," in *Proceedings of the 2015 International Conference on Distributed Computing and Networking*, ser. ICDCN '15. New York, NY, USA: ACM, 2015, pp. 4:1–4:10.

[18] S. Han, U. Topcu, and G. J. Pappas, "Differentially private convex optimization with piecewise affine objectives," in *2014 IEEE 53rd Annual Conference on Decision and Control (CDC)*, Dec. 2014, pp. 2160–2166.

[19] N. Mohammed, R. Chen, B. C. Fung, and P. S. Yu, "Differentially Private Data Release for Data Mining," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '11. New York, NY, USA: ACM, 2011, pp. 493–501.

[20] H. Brenner and K. Nissim, "Impossibility of Differentially Private Universally Optimal Mechanisms," *SIAM Journal on Computing*, vol. 43, no. 5, pp. 1513–1540, Jan. 2014.

[21] S. Han, U. Topcu, and G. J. Pappas, "Differentially private convex optimization with piecewise affine objectives," in *53rd IEEE Conference on Decision and Control*, Dec 2014, pp. 2160–2166.

[22] E. Nozari, P. Tallapragada, and J. Corts, "Differentially private distributed convex optimization via objective perturbation," in *2016 American Control Conference (ACC)*, July 2016, pp. 2061–2066.

[23] M. T. Hale and M. Egerstedt, "Cloud-based optimization: A quasi-decentralized approach to multi-agent coordination," in *2014 IEEE 53rd Annual Conference on Decision and Control (CDC)*, Dec. 2014, pp. 6635–6640.

[24] ——, "Differentially private cloud-based multi-agent optimization with constraints," in *American Control Conference (ACC), 2015*, Jul. 2015, pp. 1235–1240.

[25] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially Private Empirical Risk Minimization," *J. Mach. Learn. Res.*, vol. 12, pp. 1069–1109, Jul. 2011.

[26] R. Bassily, A. Smith, and A. Thakurta, "Private Empirical Risk Minimization: Efficient Algorithms and Tight Error Bounds," in *2014 IEEE 55th Annual Symposium on Foundations of Computer Science (FOCS)*, Oct. 2014, pp. 464–473.

[27] A. Bakushinskii and B. Poljak, "Solution of variational inequalities," *Doklady Akademii Nauk SSSR*, vol. 219, no. 5, pp. 1038–1041, 1974.

[28] B. Poljak, "Nonlinear programming methods in the presence of noise," *Mathematical programming*, vol. 14, no. 1, pp. 87–97, 1978.

[29] B. T. Poljak, *Introduction to optimization*. Optimization Software, 1987.

[30] Y. Wang, Z. Huang, S. Mitra, and G. E. Dullerud, "Entropy-minimizing mechanism for differential privacy of discrete-time linear feedback systems," in *2014 IEEE 53rd Annual Conference on Decision and Control (CDC)*, Dec. 2014, pp. 2130–2135.