Individual Regret Bounds for the Distributed Online Alternating Direction Method of Multipliers

Mohammad Akbari, Bahman Gharesifard, and Tamás Linder

***Abstract*—We consider a distributed online optimization problem where, at each time, a group of agents choose their individual states, after which an individual cost function is revealed to each of them. The whole network then faces a regret according to the cumulative sum of costs incurred by the agents' chosen states, and each agent faces an individual regret according to the cumulative sum of costs incurred by the agent's state estimation, perceived as the whole network's chosen state. In order to tackle the minimization of the individual regret using only local information, we assume that the group of agents communicate over a fixed undirected connected graph. We then propose an online version of the alternating direction method of multipliers algorithm, distributed over the communication graph, which allows each agent to drive its individual average regret over time to zero.**

***Index Terms*—Network analysis and control, sensor networks, optimization, optimization algorithms.**

I. Introduction

There is a large body of work in the literature on multiagent systems that can be cast as a so-called distributed optimization problem. In this framework, each agent is given an individual cost function and the objective for the group of agents is to cooperatively minimize the sum of these functions [15], [16], [18]. In some realistic scenarios of distributed optimization, for example, localization in sensor networks, the individual cost functions change over time. The key point here is that the agents are not aware of the way their cost functions evolve over time. This variation of the distributed optimization problem has been investigated recently in [1], [11], [13], and [14], in the context of *online* optimization [4], [6], [10], [17], [23].

In this version of the online optimization problem, at each time, an individual convex cost function is revealed to each agent after the agent has chosen its current state. In this sense, the agents only see their cost functions in hindsight and hence their states do not necessarily correspond to the minimizers of these functions; hence, each agent incurs a so-called *individual regret*, which is the difference between the accumulated collective cost incurred by the agent's state estimation and the cost incurred by the best fixed state, when all functions are known in advance [13]. The objective here is to design algorithms such that the individual regret will be sublinear in the length of the time horizon; in other words, the algorithm drives the average regret over time to zero.

Manuscript received October 20, 2017; revised May 29, 2018; accepted July 5, 2018. Date of publication July 27, 2018; date of current version March 27, 2019. Recommended by Associate Editor L. Wu. *(Corresponding author: Mohammad Akbari.)*

The authors are with the Department of Mathematics and Statistics, Queen's University, Kingston, ON K7L 3N6, Canada (e-mail: 13mav1@queensu.ca; bahman@mast.queensu.ca; linder@mast.queensu.ca).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TAC.2018.2860546

Several algorithms, provably achieving sublinear regret in time, have been designed to solve distributed online optimization problems. In [13] and [14], motivated by consensus-based distributed optimization protocols, at each time step, agents choose their future state using a combination of a subgradient flow step and an averaging step over the states of their neighbors. Unlike this class of algorithms, in this note, we investigate distributed online optimization algorithms that are based on the alternating direction method of multipliers (ADMM). This method, which is suitable for large-scale constrained optimization problems [3], [5], has already been used for distributed optimization [3], [21], [22], by casting the "consensus" step as a linear constraint. In a nutshell, the ADMM splits this (linearly) constrained optimization problem into two subproblems. Interestingly, although each step of the computation is more expensive than the one in the subgradient algorithms, using ADMM, one can achieve a convergence rate of $\mathcal{O}(\frac{1}{T})$ for distributed optimization, in contrast to the $\mathcal{O}(\frac{1}{\sqrt{T}})$ rate of consensus-based methods [7], [21]. The main objective of this note is to investigate the application of this latter class of distributed optimization protocols in online settings, and the regret bound that can be guaranteed using them. In the centralized setting, it is already known that the ADMM achieves an individual regret bound similar to the subgradient algorithms [20]. A distributed version of ADMM is proposed for minimizing a version of *network* regret in [9]. In this note, we instead consider the more challenging problem of minimizing the *individual* regret. We refer the reader to [1], [11], and [13] for more details on the difference between the individual and network regret problems. Our proposed algorithm does not have subgradient step, in contrast to [9], and the regret bounds are explicit in terms of the size of the network.

A. Preliminaries

Here, we introduce some notational conventions used throughout this note. Let $\mathbb{R}, \mathbb{R}_{\geq 0}, \mathbb{R}_{>0}, \mathbb{Z}, \mathbb{Z}_{\geq 1}$ denote the set of real, nonnegative real, positive real, integer, and positive integer numbers, respectively. We denote by $\|\cdot\|_2$ and $\|\cdot\|_1$ the Euclidean norm and 1-norm on $\mathbb{R}^d$, $d \in \mathbb{Z}_{\geq 1}$, respectively. We use the short-hand notation $\mathbf{1}_d = (1,\ldots,1)^\mathsf{T} \in \mathbb{R}^d$. We let I_d denote the identity matrix in $\mathbb{R}^{d\times d}$. For matrices $A \in \mathbb{R}^{d_1\times d_2}$ and $B \in \mathbb{R}^{e_1\times e_2}$, $d_1, d_2, e_1, e_2 \in \mathbb{Z}_{\geq 1}$, we let $A \otimes B$ denote their Kronecker product. For the matrix A, we denote by $[A]^i$, the ith row of the matrix A.

Convex Analysis: An extended real-valued function $f : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ is convex if for all $x, y \in \mathbb{R}^d$ and for all $\lambda \in [0, 1]$, we have $\lambda f(x) + (1-\lambda)f(y) \geq f(\lambda x + (1-\lambda)y)$. The convex function f is closed if for all $\alpha \in \mathbb{R}$, the sublevel set $\{x \in \mathbb{R}^d | f(x) \leq \alpha\}$ is a closed set. Given a convex function f and $x \in \mathbb{R}^d$, we call $g_x \in \mathbb{R}^d$ a subgradient of f at x, if $f(y) - f(x) \geq g_x^\mathsf{T}(y - x)$, for all $y \in \mathbb{R}^d$. It is well known that the set of subgradients of a convex function is nonempty, convex, and compact for all $x \in \mathbb{R}^d$, see [2, Proposition 4.2.1]. We denote by $\partial f(x)$ the set of subgradients of f at x. We say $\partial f(x)$ is L-bounded if there exists $L \in \mathbb{R}_{\geq 0}$ such that $\|g_x\|_1 \leq L$ for all

$g_x \in \partial f(x)$ and $x \in \mathbb{R}^d$. The function $f : \mathbb{R}^d \to \mathbb{R}$ is called Lipschitz, if for all $x, y \in \mathbb{R}^d$, $|f(x) - f(y)| \le C\|x - y\|_2$ for some $C \in \mathbb{R}_{\ge 0}$. Note that a function with L-bounded subgradients is Lipschitz.

Graph Theory: An undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathsf{A})$ consists of a vertex set $\mathcal{V} = \{1, \dots, N\}$, an edge set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ with the assumption that for $i, j \in \mathcal{V}$, if $(i, j) \in \mathcal{E}$, then $(j, i) \in \mathcal{E}$; and an adjacency matrix $\mathsf{A} \in \mathbb{R}^{n \times n}_{\ge 0}$ with $a_{ij} = 1$, iff $(i, j) \in \mathcal{E}$ and $a_{ij} = 0$, otherwise. We also denote by e_{ij} the edge between vertices i and j, with $i < j$. We assume no agent has a self-loop, so $a_{ii} = 0$ for all $i \in \mathcal{V}$. A path is a sequence of distinct vertices connected by edges. The graph $\mathcal{G}$ is connected if there is a path between any pair of distinct vertices. We define the neighbors of node i as $\mathcal{N}(i) = \{j \mid e_{ij} \text{ or } e_{ji} \in \mathcal{E}\}$.

II. Problem Statement

We start by formally describing the distributed online optimization problem [13]. For the centralized online optimization problem, we refer the reader to [6] and [23]. In the distributed approach, we consider a group of agents communicating with each other over a network, modeled by a fixed undirected graph, with properties that will be described shortly. At each time step $t \in \{1, 2, \dots, T\}$, an agent $i \in \mathcal{V} = \{1, \dots, N\}$ chooses its state $z_i(t) \in \mathbb{R}^d$. After this, a convex cost function $f_i^t : \mathbb{R}^d \to \mathbb{R}$ is revealed, and the agent incurs the cost $f_i^t(z_i(t))$. In this scenario, at each time t, the whole network aims to minimize the cost function $F^t(z) = \sum_{i=1}^N f_i^t(z)$, which is distributed among agents and is revealed when the agents have chosen their states. Therefore, each agent estimates its state based on what it thinks the whole network would choose. Moreover, due to lack of access to the cost functions before the decisions are made, the decisions do not necessarily correspond to the minimizers and the agents face a so-called *individual regret*. The individual regret function for agent j computes the difference between the network cost incurred by the agent's state estimation and the cost incurred by the best fixed choice, when all functions are known in advance, see [6], [13], and [23]. Formally, the regret of agent $j \in \mathcal{V}$ can be written as

$$\mathsf{R}^j(T) := \sum_{t=1}^T \sum_{i=1}^N f_i^t(z_j(t)) - \sum_{t=1}^T \sum_{i=1}^N f_i^t(z^\star) \tag{1}$$

where

$$z^\star \in \operatorname*{argmin}_{z \in \mathbb{R}^d} \sum_{t=1}^T \sum_{i=1}^N f_i^t(z). \tag{2}$$

Throughout this note, we assume that the minimizer set is nonempty. The objective here is to design a strategy for the agents so that each of them achieves an individual regret that is sublinear in T, i.e., $\limsup_{T\to\infty} \frac{\mathsf{R}^j(T)}{T} = 0$, which guarantees that the average regret overtime goes to zero.

III. Distributed Online ADMM

Our proposed algorithm relies on a distributed version of the so-called ADMM algorithm. We start by reviewing the ADMM algorithm [3].

A. Alternating Direction Method of Multipliers

Consider the following optimization problem:

$$\begin{aligned} &\min_{z \in Z, \xi \in \Xi} f(z) + g(\xi) \\ &\text{s.t.} \quad Az + B\xi = c \end{aligned} \tag{3}$$

where $z \in Z \subseteq \mathbb{R}^N, \xi \in \Xi \subseteq \mathbb{R}^M, c \in \mathbb{R}^W, A \in \mathbb{R}^{W \times N}, B \in \mathbb{R}^{W \times M}$, $W, M, N \in \mathbb{Z}_{\ge 1}$, and $f : \mathbb{R}^N \to \mathbb{R} \cup \{+\infty\}$ and $g : \mathbb{R}^M \to \mathbb{R} \cup \{+\infty\}$ are (extended)-real-valued functions, assumed to be closed, proper, and convex. The optimal value of (3) is denoted by $p^\star = \inf\{f(z) + g(\xi) \mid Az + B\xi = c\}$. A so-called *augmented Lagrangian function* is defined to find the optimal value of the problem (3), using the ADMM algorithm, which will be described next. For $\beta \in \mathbb{R}_{>0}$, the augmented Lagrangian is the function $L_\beta : Z \times \Xi \times \mathbb{R}^W \to \mathbb{R} \cup \{+\infty\}$, given by

$$\begin{aligned} L_\beta(z, \xi, \lambda) = f(z) + g(\xi) + \lambda^\mathsf{T}(Az + B\xi - c) \\ + \frac{\beta}{2}\|Az + B\xi - c\|_2^2. \end{aligned} \tag{4}$$

The variable λ is called the dual variable and the constant $\beta \in \mathbb{R}_{>0}$ is called the penalty parameter. The ADMM at each time $t \in \mathbb{Z}_{\ge 0}$ updates the triplet of variables (z, ξ, λ) using

$$\begin{aligned} z(t+1) &= \operatorname*{argmin}_{z \in Z} L_\beta(z, \xi(t), \lambda(t)) \\ \xi(t+1) &= \operatorname*{argmin}_{\xi \in \Xi} L_\beta(z(t+1), \xi, \lambda(t)) \\ \lambda(t+1) &= \lambda(t) + \beta(Az(t+1) + B\xi(t+1) - c). \end{aligned}$$

It is shown in [3] that the ADMM algorithm converges to the solution of (3) under the assumption that a saddle point for the Lagrangian function exists. Moreover, it can be shown that the convergence rate is $\mathcal{O}(\frac{1}{T})$ [19], [21].

B. Distributed Online ADMM

In the distributed optimization problem, a function $F(z) = \sum_{i=1}^N f_i(z)$ is distributed among N agents, where each agent $i \in \{1, \dots, N\}$ has information about its own function $f_i : \mathbb{R} \to \mathbb{R}$ and the agents cooperatively try to minimize $F(z)$. To do this, they communicate their states through a graph. We consider an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \dots, N\}$ is the set of agents and $\mathcal{E}$ is the set of edges, through which agents can communicate their states. We label each agent by a number from 1 to N. We denote by $e_{ij} \in \mathcal{E}$ the edge between agent i and j, with $i < j$. We also label each e_{ij} with a number from 1 to M, where $M = |\mathcal{E}|$.

We define $\mathbf{z} = (z_1, \dots, z_N)^\mathsf{T} \in \mathbb{R}^N$. We also assign a state ξ_{ij} to each edge $e_{ij} \in \mathcal{E}$ and let $\boldsymbol{\xi} = (\xi^1, \dots, \xi^M)^\mathsf{T}$ be the vector containing all ξ_{ij}s, where l in $\xi^l = \xi_{ij}$ is the label assigned to e_{ij}. For each pair of agents i and j connected with an edge $e_{ij} \in \mathcal{E}$, we assign the constraints $z_i + \xi_{ij} = 0$ and $z_j + \xi_{ij} = 0$, which means that all agents' states must agree, since the graph is connected. Consider now the optimization problem with linear constraints, given by

$$\min_{\mathbf{z} \in Z, \boldsymbol{\xi} \in \Xi} F(\mathbf{z}) = \sum_{i=1}^N f_i(z_i), \quad \text{s.t.} \quad A\mathbf{z} + B\boldsymbol{\xi} = 0$$

where we define the matrices $A \in \mathbb{R}^{2M \times N}$ and $B \in \mathbb{R}^{2M \times M}$, $M = |\mathcal{E}|, N = |\mathcal{V}|$ as follows. We set $B = \mathsf{I}_M \otimes \mathbf{1}_2$, and A is determined by the requirement that for each $e_{ij} \in \mathcal{E}$, the row $[A]^{2l-1}$ has a 1 in the ith position and 0s in all other positions, and the row $[A]^{2l}$ has a 1 in the jth position and 0s in all other position, where l is the label assigned to e_{ij}. We also assign the dual variables $\lambda_{ij} \in \mathbb{R}$ and $\lambda_{ji} \in \mathbb{R}$ to each $e_{ij} \in \mathcal{E}$ and let $\boldsymbol{\lambda} = (\lambda^1, \dots, \lambda^{2M})^\mathsf{T}$ be the vector containing all λ_{ij}s, where l in $\lambda^{2l-1} = \lambda_{ij}$ and $\lambda^{2l} = \lambda_{ji}$ is the label assigned to e_{ij}.

We partition the neighbors $\mathcal{N}(i)$ of a node i into two sets, denoted by $\mathcal{N}_s(i)$ and $\mathcal{N}_\ell(i)$, where $\mathcal{N}_s(i) = \{j \mid e_{ji} \in \mathcal{E}, j < i\}$ is the set of neighbors of agent i with index smaller than i, and $\mathcal{N}_\ell(i) = \{j \mid e_{ij} \in \mathcal{E}, i < j\}$ is the set of neighbors of agent i with index larger than i.

Example 3.1: Consider a path graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V} = \{1, 2, 3\}$ and $\mathcal{E} = \{e_{12}, e_{23}\}$, we have

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \; B = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}, \; \mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix}, \; \boldsymbol{\xi} = \begin{bmatrix} \xi_{12} \\ \xi_{23} \end{bmatrix}$$

and

$$\boldsymbol{\lambda} = [\lambda_{12}, \lambda_{21}, \lambda_{23}, \lambda_{32}]^{\mathsf{T}}, \qquad \text{and}$$

$$\mathcal{N}_s(1) = \emptyset, \; \mathcal{N}_\ell(1) = \{2\}, \; \mathcal{N}_s(2) = \{1\}$$

$$\mathcal{N}_\ell(2) = \{3\}, \; \mathcal{N}_s(3) = \{2\}, \; \mathcal{N}_\ell(3) = \emptyset.$$

We also have the following constraints: $z_1 = \xi_{12}, \;\; z_2 = \xi_{12}, \;\; z_2 = \xi_{23}, \;\; z_3 = \xi_{23}$, so $z_1 = z_2 = z_3$.

As we have described previously, at each iteration $t \in \mathbb{Z}_{\geq 1}$, each agent $i \in \mathcal{V}$ chooses $z_i(t) \in \mathbb{R}^d$, and after that a convex cost function $f_i^t : \mathbb{R}^d \to \mathbb{R}$ is revealed and agent i incurs cost $f_i^t(z_i(t))$. The *distributed online ADMM algorithm* is defined as follows.

1) *Initialization*: Choose arbitrary $z_i(1)$ for $i \in \mathcal{V}$ and arbitrary $\lambda_{ij}(1) = -\lambda_{ji}(1)$ and $\xi_{ij}(1)$ for all $e_{ij} \in \mathcal{E}$, which are not necessarily all equal.
2) *Updates*: For $t \geq 2$.
   a) Agents, sequentially, update their estimate using

$$\begin{aligned} z_i(t) = \underset{z_i}{\operatorname{argmin}} \; & f_i^{t-1}(z_i) \\ & + \frac{\beta}{2} \sum_{j \in \mathcal{N}_s(i)} \left| z_i + \xi_{ji}(t-1) + \frac{1}{\beta}\lambda_{ij}(t-1) \right|^2 \\ & + \frac{\beta}{2} \sum_{j \in \mathcal{N}_\ell(i)} \left| z_i + \xi_{ij}(t-1) + \frac{1}{\beta}\lambda_{ij}(t-1) \right|^2 . \end{aligned} \tag{5}$$

   b) Each agent i updates ξ_{ji}, for all j in $\mathcal{N}_s(i)$ as

$$\begin{aligned} \xi_{ji}(t) = \underset{\xi_{ji}}{\operatorname{argmin}} \; & \frac{\beta}{2} \left(\left| z_j(t) + \xi_{ji} + \frac{1}{\beta}\lambda_{ji}(t-1) \right|^2 \right. \\ & \left. + \left| z_i(t) + \xi_{ji} + \frac{1}{\beta}\lambda_{ij}(t-1) \right|^2 \right). \end{aligned} \tag{6}$$

   c) Each agent i updates λ_{ji} and λ_{ij}, for all j in $\mathcal{N}_s(i)$

$$\lambda_{ij}(t) = \lambda_{ij}(t-1) + \beta(z_i(t) + \xi_{ji}(t)) \tag{7}$$

$$\lambda_{ji}(t) = \lambda_{ji}(t-1) + \beta(z_j(t) + \xi_{ji}(t)). \tag{8}$$

Note that this algorithm is different from the one presented in [9], in the sense, it does not have a subgradient step.

IV. Regret Bounds for the Distributed Online ADMM

As the main result of this note, we provide an upper bound on each agent's individual regret, given by (1), under the distributed online ADMM algorithm.

Theorem 4.1: Let $\{f_1^t, \ldots, f_N^t\}_{t=1}^T$ be an arbitrary sequence of convex functions where, for each $i \in \mathcal{V}$, f_i^t has an L-bounded subgradient. Let the sequences $\{\mathbf{z}(t) = (z_1(t), \ldots, z_N(t))^{\mathsf{T}}\}_{t=1}^T$, $\{\boldsymbol{\xi}(t) = (\xi^1(t), \ldots, \xi^M(t))^{\mathsf{T}}\}_{t=1}^T$, and $\{\boldsymbol{\lambda}(t) = (\lambda^1(t), \ldots, \lambda^{2M}(t))^{\mathsf{T}}\}_{t=1}^T$ be generated by the distributed online ADMM over an undirected fixed connected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $N = |\mathcal{V}|$ and $M = |\mathcal{E}|$. Suppose that there exist $z^\star \in \operatorname{argmin} \sum_{i=1}^N \sum_{t=1}^T f_i^t(z)$ and choose $\mathbf{z}^\star = (z^\star, \ldots, z^\star)$, $\boldsymbol{\xi}^\star$ satisfying $A\mathbf{z}^\star + B\boldsymbol{\xi}^\star = 0$, and let $\beta = (N+1)\sqrt{T}$. Then, we have

$$\begin{aligned} \mathsf{R}^j(T) \leq & \frac{1}{2(N+1)\sqrt{T}} \|\boldsymbol{\lambda}(1)\|_2^2 + \sqrt{T} M L^2 (N+1) \\ & + \frac{(N+1)\sqrt{T}}{2} \|B\boldsymbol{\xi}^\star - B\boldsymbol{\xi}(1)\|_2^2 \\ & + \frac{N\sqrt{T}}{2} \|A\mathbf{z}(1) + B\boldsymbol{\xi}(1)\|_2^2 . \end{aligned}$$

Before stating the proof, let us mention a few key points about this results, especially in comparison with other available distributed algorithms for online optimization. This algorithm provides a regret bound of order $\mathcal{O}(\sqrt{T})$, which is similar to the ones with subgradient flow protocols. However, this algorithm gives us an *explicit* dependency of the regret bound on a polynomial of the size of the network and it does not depend on the spectral radius of the adjacency matrix of the network, in contrast to the most existing algorithms, see for example [12] and [13]. It is also worth mentioning that the distributed online ADMM algorithm presented in [9] considers the problem of bounding the *network* regret, and not the *individual* regret. As our proof illustrates, the latter problem is more challenging.

We now start the process of proving Theorem 4.1. Our proof relies on a sequence of results, which we establish via Lemmas 4.2–4.7.

Lemma 4.2: Let the sequence $\{\mathbf{z}(t), \boldsymbol{\xi}(t), \boldsymbol{\lambda}(t)\}_{t=1}^T$ be generated by the distributed online ADMM. Then, for any $\tilde{\mathbf{z}} \in \mathbb{R}^N$, we have

$$\begin{aligned} & \sum_{i=1}^N f_i^t(z_i(t+1)) - \sum_{i=1}^N f_i^t(\tilde{z}_i) \\ & \quad \leq -(A(\mathbf{z}(t+1) - \tilde{\mathbf{z}}))^{\mathsf{T}} \left(\boldsymbol{\lambda}(t+1) + \beta B(\boldsymbol{\xi}(t) - \boldsymbol{\xi}(t+1))\right). \end{aligned}$$

Proof: For each $t \in \mathbb{Z}_{\geq 1}$, since $z_i(t+1)$ satisfies (5), we have

$$\begin{aligned} 0 \in \partial f_i^t(z_i(t+1)) + \beta & \left(\sum_{j \in \mathcal{N}_s(i)} \left(z_i(t+1) + \xi_{ji}(t) + \frac{1}{\beta}\lambda_{ij}(t) \right) \right. \\ & \left. + \sum_{j \in \mathcal{N}_\ell(i)} \left(z_i(t+1) + \xi_{ij}(t) + \frac{1}{\beta}\lambda_{ij}(t) \right) \right). \end{aligned}$$

Hence

$$\begin{aligned} -\beta & \left(\sum_{j \in \mathcal{N}_s(i)} \left(z_i(t+1) + \xi_{ji}(t) + \frac{1}{\beta}\lambda_{ij}(t) \right) + \sum_{j \in \mathcal{N}_\ell(i)} \left(z_i(t+1) \right.\right. \\ & \left.\left. + \; \xi_{ij}(t) + \frac{1}{\beta}\lambda_{ij}(t) \right) \right) \in \partial f_i^t(z_i(t+1)). \end{aligned} \tag{9}$$

Since each f_i^t is convex, we have

$$f_i^t(z_i(t+1)) - f_i^t(\tilde{z}_i) \leq (z_i(t+1) - \tilde{z}_i) g_i^t(z_i(t+1))$$

for all $g_i^t(z_i(t+1)) \in \partial f_i^t(z_i(t+1))$. As a result, using (9)

$$\begin{aligned} & f_i^t(z_i(t+1)) - f_i^t(\tilde{z}_i) \\ & \quad \leq -\beta(z_i(t+1) - \tilde{z}_i) \left(\sum_{j \in \mathcal{N}_s(i)} \left(z_i(t+1) + \xi_{ji}(t) + \frac{1}{\beta}\lambda_{ij}(t) \right) \right. \\ & \quad \left. + \sum_{j \in \mathcal{N}_\ell(i)} \left(z_i(t+1) + \xi_{ij}(t) + \frac{1}{\beta}\lambda_{ij}(t) \right) \right). \end{aligned}$$

Using (7) and (8), we have

$$\begin{aligned}
&f_i^t(z_i(t+1)) - f_i^t(\tilde{z}_i)\\
&\quad\le -(z_i(t+1)-\tilde{z}_i)\Bigg(\sum_{j\in\mathcal{N}_s(i)}(\lambda_{ij}(t+1)+\beta(\xi_{ji}(t)\\
&\qquad - \xi_{ji}(t+1))) + \sum_{j\in\mathcal{N}_\ell(i)}(\lambda_{ij}(t+1)+\beta(\xi_{ij}(t)-\xi_{ij}(t+1)))\Bigg)\\
&\quad= -(z_i(t+1)-\tilde{z}_i)\left(([A]_i)^\mathsf{T}\boldsymbol{\lambda}(t+1)\right.\\
&\qquad\left. + \beta\left(\left[A^\mathsf{T}B\right]_i\right)(\boldsymbol{\xi}(t)-\boldsymbol{\xi}(t+1))\right)
\end{aligned}$$

where we have used the definition of A and B in the last equality. Hence

$$\begin{aligned}
&\sum_{i=1}^N f_i^t(z_i(t+1)) - \sum_{i=1}^N f_i^t(\tilde{z}_i)\\
&\le -(\mathbf{z}(t+1)-\tilde{\mathbf{z}})^\mathsf{T}\times\left(A^\mathsf{T}\boldsymbol{\lambda}(t+1)+\beta\left[A^\mathsf{T}B\right](\boldsymbol{\xi}(t)-\boldsymbol{\xi}(t+1))\right)\\
&= -(A(\mathbf{z}(t+1)-\tilde{\mathbf{z}}))^\mathsf{T}\\
&\quad\times(\boldsymbol{\lambda}(t+1)+\beta B(\boldsymbol{\xi}(t)-\boldsymbol{\xi}(t+1)))
\end{aligned}$$

which proves the claim. ■

Lemma 4.3: Let the sequence $\{\mathbf{z}(t),\boldsymbol{\xi}(t),\boldsymbol{\lambda}(t)\}_{t=1}^T$ be generated by the distributed online ADMM. Then, for all $t\in\mathbb{Z}_{\ge1}$, we have

$$B^\mathsf{T}\boldsymbol{\lambda}(t)=0. \tag{10}$$

Proof: Since $\xi_{ji}(t+1)$ is given by (6), we have

$$\begin{aligned}
\beta\bigg(&z_j(t+1)+\xi_{ji}(t+1)+\frac{1}{\beta}\lambda_{ji}(t)+z_i(t+1)\\
&+\xi_{ji}(t+1)+\frac{1}{\beta}\lambda_{ij}(t)\bigg)=0.
\end{aligned}$$

Now, using (7) and (8)

$$(\lambda_{ji}(t+1)+\lambda_{ij}(t+1))=0$$

or equivalently, $B^\mathsf{T}\boldsymbol{\lambda}(t)=0$, for all $t\in\mathbb{Z}_{\ge1}$. ■

Lemma 4.4: Let the sequence $\{\mathbf{z}(t),\boldsymbol{\xi}(t),\boldsymbol{\lambda}(t)\}_{t=1}^T$ be generated by the distributed online ADMM. For any $\mathbf{z}^\star,\boldsymbol{\xi}^\star$, satisfying $A\mathbf{z}^\star+B\boldsymbol{\xi}^\star=0$, we have

$$\begin{aligned}
&\sum_{i=1}^N f_i^t(z_i(t+1)) - \sum_{i=1}^N f_i^t(z^\star)\\
&\le \frac{1}{2\beta}(\|\boldsymbol{\lambda}(t)\|_2^2-\|\boldsymbol{\lambda}(t+1)\|_2^2)-\frac{\beta}{2}\|A\mathbf{z}(t+1)+B\boldsymbol{\xi}(t)\|_2^2\\
&\quad+\frac{\beta}{2}\left(\|B\boldsymbol{\xi}^\star-B\boldsymbol{\xi}(t)\|_2^2-\|B\boldsymbol{\xi}^\star-B\boldsymbol{\xi}(t+1)\|_2^2\right).
\end{aligned}$$

Proof: From Lemma 4.2, for $\tilde{\mathbf{z}}=\mathbf{z}^\star$, we have

$$\begin{aligned}
&\sum_{i=1}^N f_i^t(z_i(t+1)) - \sum_{i=1}^N f_i^t(z^\star)\\
&\le -\left(A\mathbf{z}(t+1)-A\mathbf{z}^\star\right)^\mathsf{T}\left(\boldsymbol{\lambda}(t+1)+\beta B(\boldsymbol{\xi}(t)-\boldsymbol{\xi}(t+1))\right)\\
&= -\left(A\mathbf{z}(t+1)+B\boldsymbol{\xi}^\star\right)^\mathsf{T}\boldsymbol{\lambda}(t+1)\\
&\quad-\beta(A\mathbf{z}(t+1)+B\boldsymbol{\xi}^\star)^\mathsf{T}(B\boldsymbol{\xi}(t)-B\boldsymbol{\xi}(t+1))\\
&= -\left(A\mathbf{z}(t+1)+B\boldsymbol{\xi}^\star\right)^\mathsf{T}\boldsymbol{\lambda}(t+1)\\
&\quad-\frac{\beta}{2}\left(\|B\boldsymbol{\xi}(t+1)-B\boldsymbol{\xi}^\star\|_2^2-\|B\boldsymbol{\xi}(t)-B\boldsymbol{\xi}^\star\|_2^2\right.\\
&\quad\left.+\|A\mathbf{z}(t+1)+B\boldsymbol{\xi}(t)\|_2^2-\|A\mathbf{z}(t+1)+B\boldsymbol{\xi}(t+1)\|_2^2\right)
\end{aligned} \tag{11}$$

where we have used the fact that for all $u_1,u_2,u_3,u_4\in\mathbb{R}^d$

$$\begin{aligned}
(u_1+u_2)^\mathsf{T}(u_3-u_4)=\frac{1}{2}(&\|u_4-u_2\|_2^2-\|u_3-u_2\|_2^2\\
&+\|u_1+u_3\|_2^2-\|u_1+u_4\|_2^2)
\end{aligned} \tag{12}$$

to obtain the last equality. Note that by Lemma 4.3, we have $B^\mathsf{T}\boldsymbol{\lambda}(t+1)=0$; hence, we can rewrite the sum of the first and the last terms in (11) as

$$\begin{aligned}
&\frac{\beta}{2}\|A\mathbf{z}(t+1)+B\boldsymbol{\xi}(t+1)\|_2^2-(A\mathbf{z}(t+1)+B\boldsymbol{\xi}^\star)^\mathsf{T}\boldsymbol{\lambda}(t+1)\\
&=-(A\mathbf{z}(t+1)+B\boldsymbol{\xi}(t+1))^\mathsf{T}\boldsymbol{\lambda}(t+1)\\
&\quad+\frac{\beta}{2}\|A\mathbf{z}(t+1)+B\boldsymbol{\xi}(t+1)\|_2^2\\
&=\frac{1}{2\beta}\left(2(\boldsymbol{\lambda}(t)-\boldsymbol{\lambda}(t+1))^\mathsf{T}\boldsymbol{\lambda}(t+1)+\|\boldsymbol{\lambda}(t+1)-\boldsymbol{\lambda}(t)\|_2^2\right)\\
&=\frac{1}{2\beta}\left(\|\boldsymbol{\lambda}(t)\|_2^2-\|\boldsymbol{\lambda}(t+1)\|_2^2\right)
\end{aligned}$$

which completes the proof. ■

Lemma 4.5: Let the sequence $\{\mathbf{z}(t),\boldsymbol{\xi}(t),\boldsymbol{\lambda}(t)\}_{t=1}^T$ be generated by the distributed online ADMM. For all $t\in\mathbb{Z}_{\ge1}$

$$\begin{aligned}
&\|A\mathbf{z}(t+1)+B\boldsymbol{\xi}(t+1)\|_2^2+\|B\boldsymbol{\xi}(t+1)-B\boldsymbol{\xi}(t)\|_2^2\\
&\quad=\|A\mathbf{z}(t+1)+B\boldsymbol{\xi}(t)\|_2^2.
\end{aligned}$$

Proof: Using Lemma 4.3, we can write

$$\begin{aligned}
0&=(\boldsymbol{\xi}(t+1)-\boldsymbol{\xi}(t))^\mathsf{T}B^\mathsf{T}\boldsymbol{\lambda}(t+1),\\
0&=(\boldsymbol{\xi}(t+1)-\boldsymbol{\xi}(t))^\mathsf{T}B^\mathsf{T}\boldsymbol{\lambda}(t).
\end{aligned}$$

Adding these equation, we can write

$$\begin{aligned}
0&=(\boldsymbol{\xi}(t+1)-\boldsymbol{\xi}(t))^\mathsf{T}B^\mathsf{T}(\boldsymbol{\lambda}(t+1)-\boldsymbol{\lambda}(t))\\
&=\beta(B\boldsymbol{\xi}(t+1)-B\boldsymbol{\xi}(t))^\mathsf{T}(A\mathbf{z}(t+1)+B\boldsymbol{\xi}(t+1))\\
&=\frac{\beta}{2}\left(\|A\mathbf{z}(t+1)+B\boldsymbol{\xi}(t+1)\|_2^2+\|B\boldsymbol{\xi}(t+1)-B\boldsymbol{\xi}(t)\|_2^2\right.\\
&\quad\left.-\|A\mathbf{z}(t+1)+B\boldsymbol{\xi}(t)\|_2^2\right)
\end{aligned}$$

where we have used $(u_1-u_2)^\mathsf{T}(u_3+u_1)=\frac{1}{2}(\|u_3+u_1\|_2^2+\|u_1-u_2\|_2^2-\|u_3+u_2\|_2^2)$, for all $u_1,u_2,u_3\in\mathbb{R}^d$. This proves the claim. ■

Lemma 4.6: Let $\{f_1^t,\ldots,f_N^t\}_{t=1}^T$ be a sequence of convex functions, where for each $i\in\mathcal{V}$, f_i^t has an L-bounded subgradient. Let the sequence $\{\mathbf{z}(t),\boldsymbol{\xi}(t),\boldsymbol{\lambda}(t)\}_{t=1}^T$ be generated by the distributed online ADMM. Then, for all $t\in\mathbb{Z}_{\ge1}$ and some $\alpha\in\mathbb{R}_{>0}$, we have

$$\begin{aligned}
&\sum_{i=1}^N f_i^t(z_i(t))-\sum_{i=1}^N f_i^t(z_i(t+1))\\
&\le\frac{2ML^2}{\alpha}+\frac{\alpha}{2}\left(\|A\mathbf{z}(t+1)+B\boldsymbol{\xi}(t+1)\|_2^2\right.\\
&\quad\left.+\|A\mathbf{z}(t)+B\boldsymbol{\xi}(t)\|_2^2+\|B\boldsymbol{\xi}(t+1)-B\boldsymbol{\xi}(t)\|_2^2\right).
\end{aligned}$$

Proof: Since f_i^t is a convex function, we can write

$$\begin{aligned} f_i^t(z_i(t)) - f_i^t(z_i(t+1)) &\le g_i^t(z_i(t))(z_i(t) - z_i(t+1)) \\ &\le |g_i^t(z_i(t))||z_i(t) - z_i(t+1)| \\ &\le L|z_i(t) - z_i(t+1)| \end{aligned}$$

for all $g_i^t(z_i(t)) \in \partial f(z_i(t))$, where we have used that $|g_i^t(z)| \le L$. Now, we have

$$\sum_{i=1}^N f_i^t(z_i(t)) - \sum_{i=1}^N f_i^t(z_i(t+1)) \le L \sum_{i=1}^N |z_i(t) - z_i(t+1)|.$$

Hence, we can write

$$\begin{aligned} \sum_{i=1}^N |z_i(t) - z_i(t+1)| &\le \|A\mathbf{z}(t) - A\mathbf{z}(t+1)\|_1 \\ &\le \sqrt{2M}\|A\mathbf{z}(t) - A\mathbf{z}(t+1)\|_2 \\ &\le \sqrt{2M}(\|A\mathbf{z}(t) + B\boldsymbol{\xi}(t)\|_2 \\ &\quad + \|A\mathbf{z}(t+1) + B\boldsymbol{\xi}(t)\|_2) \end{aligned}$$

where we have used the fact that since the graph is connected, for all $i \in \mathcal{V}$, $z_i(t)$ has appeared in some elements of the vector $A\mathbf{z}(t)$. Using this, we have

$$\begin{aligned} &\sum_{i=1}^N f_i^t(z_i(t)) - \sum_{i=1}^N f_i^t(z_i(t+1)) \\ &\le L\sqrt{2M}\|A\mathbf{z}(t) + B\boldsymbol{\xi}(t)\|_2 + L\sqrt{2M}\|A\mathbf{z}(t+1) + B\boldsymbol{\xi}(t)\|_2 \\ &\le \frac{2ML^2}{2\alpha} + \frac{\alpha}{2}\|A\mathbf{z}(t) + B\boldsymbol{\xi}(t)\|_2^2 \\ &\quad + \frac{2ML^2}{2\alpha} + \frac{\alpha}{2}\|A\mathbf{z}(t+1) + B\boldsymbol{\xi}(t)\|_2^2 \\ &\le \frac{2ML^2}{\alpha} + \frac{\alpha}{2}\left(\|A\mathbf{z}(t) + B\boldsymbol{\xi}(t)\|_2^2\right. \\ &\quad \left. + \|A\mathbf{z}(t+1) + B\boldsymbol{\xi}(t+1)\|_2^2 + \|B\boldsymbol{\xi}(t) - B\boldsymbol{\xi}(t+1)\|_2^2\right) \end{aligned}$$

where in the second inequality, we have used Young's inequality: $2u_1^\mathsf{T} u_2 \le \|u_1\|_2^2 + \|u_2\|_2^2$ for all $u_1, u_2 \in \mathbb{R}^d$. ■

The next lemma will be instrumental in bounding the *individual regret function* for one particular agent's states, say the jth agent. It is the final step in proving our main result.

Lemma 4.7: Let $\{f_1^t, \dots, f_N^t\}_{t=1}^T$ be a sequence of convex functions, where for each $i \in \mathcal{V}$, f_i^t has an L-bounded subgradient. Let the sequence $\{\mathbf{z}(t), \boldsymbol{\xi}(t), \boldsymbol{\lambda}(t)\}_{t=1}^T$ be generated by the distributed online ADMM. Then, for all $t \in \mathbb{Z}_{\ge 1}$, all $j \in \mathcal{V}$, and all $\alpha \in \mathbb{R}_{>0}$

$$\begin{aligned} \sum_{i=1}^N f_i^t(z_j(t)) - \sum_{i=1}^N f_i^t(z_i(t)) &\le \frac{2ML^2(N-1)}{2\alpha} \\ &\quad + \frac{(N-1)\alpha}{2}\|A\mathbf{z}(t) + B\boldsymbol{\xi}(t)\|_2^2. \end{aligned}$$

Proof: Since f_i^t is convex with L-bounded subgradients, we have that $f_i^t(z_j(t)) - f_i^t(z_i(t)) \le L|z_j(t) - z_i(t)|$. Hence

$$\sum_{i=1}^N f_i^t(z_j(t)) - \sum_{i=1}^N f_i^t(z_i(t)) \le L\sum_{i=1}^N |z_j(t) - z_i(t)|.$$

Note that, since the graph is connected, for agent i and j that are not neighbors, there exists a path connecting them, and we have $|z_j(t) - z_i(t)| \le \|A\mathbf{z}(t) + B\boldsymbol{\xi}(t)\|_1$. As a result

$$\begin{aligned} \sum_{i=1}^N |z_j(t) - z_i(t)| &\le (N-1)\|A\mathbf{z}(t) + B\boldsymbol{\xi}(t)\|_1 \\ &\le (N-1)\sqrt{2M}\|A\mathbf{z}(t) + B\boldsymbol{\xi}(t)\|_2. \end{aligned}$$

Now, for any $\alpha > 0$

$$\begin{aligned} \sum_{i=1}^N f_i^t(z_j(t)) - \sum_{i=1}^N f_i^t(z_i(t)) &\le L(N-1)\sqrt{2M}\|A\mathbf{z}(t) + B\boldsymbol{\xi}(t)\|_2 \\ &\le \frac{2ML^2(N-1)}{2\alpha} \\ &\quad + \frac{(N-1)\alpha}{2}\|A\mathbf{z}(t) + B\boldsymbol{\xi}(t)\|_2^2 \end{aligned}$$

where we have used Young's inequality. ■

We are now in a position to prove Theorem 4.1.

Proof. (Theorem 4.1): Using Lemmas 4.4, 4.6, and 4.7, we can write

$$\begin{aligned} &\sum_{i=1}^N f_i^t(z_j(t)) - \sum_{i=1}^N f_i^t(z^\star) \\ &\le \frac{1}{2\beta}(\|\boldsymbol{\lambda}(t)\|_2^2 - \|\boldsymbol{\lambda}(t+1)\|_2^2) + \frac{ML^2(N+1)}{\alpha} \\ &\quad - \frac{\beta}{2}\left(\|A\mathbf{z}(t+1) + B\boldsymbol{\xi}(t+1)\|_2^2 + \|B\boldsymbol{\xi}(t) - B\boldsymbol{\xi}(t+1)\|_2^2\right) \\ &\quad + \frac{\beta}{2}\left(\|B\boldsymbol{\xi}^\star - B\boldsymbol{\xi}(t)\|_2^2 - \|B\boldsymbol{\xi}^\star - B\boldsymbol{\xi}(t+1)\|_2^2\right) \\ &\quad + \frac{\alpha}{2}\left(\|A\mathbf{z}(t+1) + B\boldsymbol{\xi}(t+1)\|_2^2 + \|A\mathbf{z}(t) + B\boldsymbol{\xi}(t)\|_2^2\right) \\ &\quad + \frac{\alpha}{2}\|B\boldsymbol{\xi}(t+1) - B\boldsymbol{\xi}(t)\|_2^2 + \frac{(N-1)\alpha}{2}\|A\mathbf{z}(t) + B\boldsymbol{\xi}(t)\|_2^2. \end{aligned} \tag{13}$$

Note that

$$\begin{aligned} &-\frac{\beta}{2}\|A\mathbf{z}(t+1) + B\boldsymbol{\xi}(t+1)\|_2^2 + \frac{\alpha}{2}\left(\|A\mathbf{z}(t+1) + B\boldsymbol{\xi}(t+1)\|_2^2\right. \\ &\quad \left. + \|A\mathbf{z}(t) + B\boldsymbol{\xi}(t)\|_2^2\right) + \frac{(N-1)\alpha}{2}\|A\mathbf{z}(t) + B\boldsymbol{\xi}(t)\|_2^2 \\ &= \frac{\alpha - \beta}{2}\|A\mathbf{z}(t+1) + B\boldsymbol{\xi}(t+1)\|_2^2 + \frac{N\alpha}{2}\|A\mathbf{z}(t) + B\boldsymbol{\xi}(t)\|_2^2. \end{aligned} \tag{14}$$

Let

$$\begin{aligned} K = &-\sum_{t=1}^T \frac{\beta - \alpha}{2}\|A\mathbf{z}(t+1) + B\boldsymbol{\xi}(t+1)\|_2^2 \\ &+ \sum_{t=1}^T \frac{N\alpha}{2}\|A\mathbf{z}(t) + B\boldsymbol{\xi}(t)\|_2^2. \end{aligned}$$

For $\beta \ge (N+1)\alpha$, using (14), we can write

$$\begin{aligned} K &= \sum_{t=1}^{T-1} -\frac{\beta - \alpha - N\alpha}{2}\|A\mathbf{z}(t+1) + B\boldsymbol{\xi}(t+1)\|_2^2 \\ &\quad - \frac{\beta - \alpha}{2}\|A\mathbf{z}(T+1) + B\boldsymbol{\xi}(T+1)\|_2^2 \\ &\quad + \frac{N\alpha}{2}\|A\mathbf{z}(1) + B\boldsymbol{\xi}(1)\|_2^2 \\ &\le \frac{N\alpha}{2}\|A\mathbf{z}(1) + B\boldsymbol{\xi}(1)\|_2^2. \end{aligned} \tag{15}$$

We also have

$$-\frac{\beta}{2}\|B\boldsymbol{\xi}(t) - B\boldsymbol{\xi}(t+1)\|_2^2 + \frac{\alpha}{2}\|B\boldsymbol{\xi}(t+1) - B\boldsymbol{\xi}(t)\|_2^2 \le 0 \quad (16)$$

for $\beta \ge \alpha$. Using (15) and (16) along with (13), we have

$$\begin{aligned}\mathsf{R}^j(T) &= \sum_{t=1}^{T}\sum_{i=1}^{N} f_i^t(z_j(t)) - \sum_{t=1}^{T}\sum_{i=1}^{N} f_i^t(z^\star) \\ &\le \sum_{t=1}^{T}\frac{1}{2\beta}\left(\|\boldsymbol{\lambda}(t)\|_2^2 - \|\boldsymbol{\lambda}(t+1)\|_2^2\right) + \sum_{t=1}^{T}\frac{ML^2(N+1)}{\alpha} \\ &\quad + \sum_{t=1}^{T}\frac{\beta}{2}\left(\|B\boldsymbol{\xi}^\star - B\boldsymbol{\xi}(t)\|_2^2 - \|B\boldsymbol{\xi}^\star - B\boldsymbol{\xi}(t+1)\|_2^2\right) \\ &\quad + \frac{N\alpha}{2}\|A\mathbf{z}(1) + B\boldsymbol{\xi}(1)\|_2^2 \\ &\le \frac{1}{2\beta}\|\boldsymbol{\lambda}(1)\|_2^2 + \frac{T}{\alpha}ML^2(N+1) + \frac{\beta}{2}\|B\boldsymbol{\xi}^\star - B\boldsymbol{\xi}(1)\|_2^2 \\ &\quad + \frac{N\alpha}{2}\|A\mathbf{z}(1) + B\boldsymbol{\xi}(1)\|_2^2\end{aligned}$$

where we have used the telescopic series to establish the last inequality. By choosing $\alpha = \sqrt{T}$ and $\beta = (N+1)\sqrt{T}$, we have

$$\begin{aligned}\mathsf{R}^j(T) &\le \frac{1}{2(N+1)\sqrt{T}}\|\boldsymbol{\lambda}(1)\|_2^2 + \sqrt{T}ML^2(N+1) \\ &\quad + \frac{(N+1)\sqrt{T}}{2}\|B\boldsymbol{\xi}^\star - B\boldsymbol{\xi}(1)\|_2^2 \\ &\quad + \frac{N\sqrt{T}}{2}\|A\mathbf{z}(1) + B\boldsymbol{\xi}(1)\|_2^2.\end{aligned}$$

■

Remark 4.8 (Computation Load): In order to execute the above-mentioned algorithm, each agent i needs to store and update the variables $z_i, \xi_{ji}, \lambda_{ji}$, and λ_{ij} where $j \in \mathcal{N}_s(i)$. Hence, the overall computational load of the network is $|\mathcal{V}| + 3|\mathcal{E}|$. As we were finalizing this note, a new ADMM algorithm for distributed optimization has appeared in the literature [24], which has a lower computation load at each step of time. Particularly, the number of updated variables in each time step is $3|\mathcal{V}|$, which is less than $|\mathcal{V}| + 3|\mathcal{E}|$ for connected graphs in our case. The setting in [24] is, however, not online. It is, hence, an interesting future problem to investigate if this new algorithm can be incorporated into the online setting to reduce the required computational load.

V. Simulation

We illustrate our results by an example on localization in sensor networks, motivated by [8]. Consider a network of $N = 8$ sensors that are used to estimate a variable $s \in \mathbb{R}$. The communication graph is given in Fig. 1. Each sensor $i \in \mathcal{V}$, at each time $t \in \{1, \ldots, T\}$, observes $q_i^t \in \mathbb{R}$, a noisy observation of variable s. In this example, we assume that this noisy observation is given by $q_i^t = a_i^t \tilde{s} + b_i^t$, where $a_i^t \in [0, 2]$ and $b_i^t \in [-0.5, 0.5]$ are chosen at random and independently from a uniform distribution and $\tilde{s}$ is a fixed real value. We use the squared observation error as the cost function, i.e., $f_i^t(s) = \frac{1}{2}(q_i^t - s)^2$. The best estimation of s is the minimizer of the function

$$\hat{s} = \operatorname*{argmin}_s F(s) = \operatorname*{argmin}_s \sum_{t=1}^{T}\sum_{i=1}^{N}\frac{1}{2}\left(q_i^t - s\right)^2.$$

Let us assume the actual value of $\tilde{s} = \frac{1}{4}$.

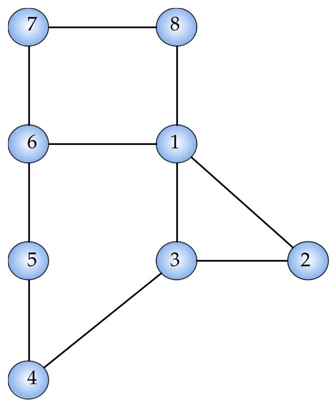

Fig. 1. $N = 8$ sensors are communicating through the graph.

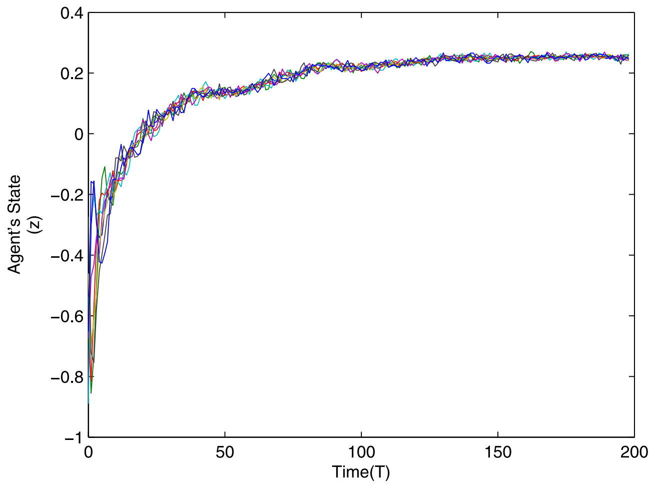

Fig. 2. Sensors' state estimations versus time for eight sensors are shown. The network consists of $N = 8$ sensors communicating over an undirected graph. The ith sensor observes $q_i^t = a_i^t s + b_i^t$, where a_i^t and b_i^t are chosen at random from a uniform distribution on $[0, 2]$ and $[-1/2, 1/2]$, respectively. We use the distributed online ADMM algorithm to estimate $\hat{s}$.

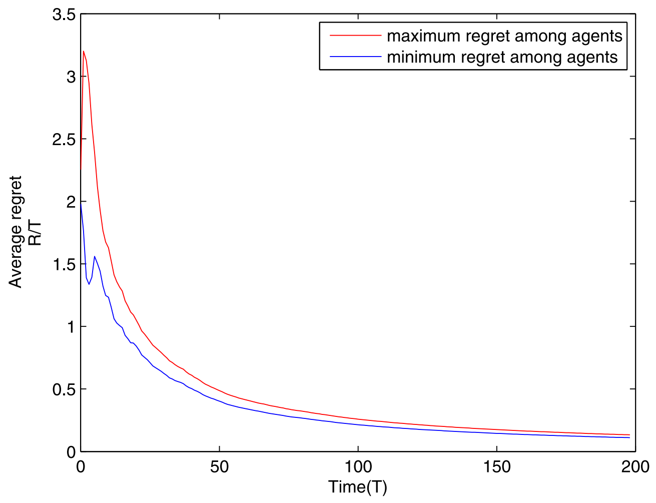

Fig. 3. Average regrets over time $\mathsf{R}^j(T)/T$ versus T for two sensors with the maximum and minimum average regrets are shown, under the same assumptions as in Fig. 2.

We use the distributed online ADMM to estimate the variable s. Here, the noise characteristics are not known to the agents.

Fig. 2 shows the states of four sensors over 200 time iterations. By using the distributed online ADMM algorithm, the sensors reach a consensus near the expected value of $\hat{s}$, which is one-fourth, as shown in Fig. 2. Fig. 3 shows the average individual regret of the two sensors with the maximum and minimum average regrets over time.

VI. Conclusion and Future Work

In this note, we presented a distributed online version of ADMM to solve a distributed online optimization problem. We showed that by choosing proper parameters, we can bound the individual regret function by $\mathcal{O}(\sqrt{T})$. This bound is similar to the one for consensus-based subgradient flow protocols, however, our algorithm exhibits an explicit dependency of the regret bound on the size of the network and is gradient free. Extending the results to scenarios with additional internal linear constraints and time-varying graphs and directed graphs are among the avenues of future work. It is a particularly important avenue of research to investigate the possibility of extending our results to directed and time-varying settings. This relies on relaxing the assumptions on the communication graph in [22], where the undirected nature of the graph topology plays a major role in assuring that the dual variables are updated consistently throughout the network.

References

[1] 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.

[2] D. P. Bertsekas, A. Nedić, and A. E. Ozdaglarx. *Convex Analysis and Optimization*, 1st ed. Belmont, MA, USA: Athena Scientific, 2003.

[3] 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, pp. 1–122, 2011.

[4] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge, U.K.: Cambridge Univ. Press, 2006.

[5] R. Glowinski and A. Marrocco, "Sur l'approximation, par elements finis d'ordre un, et la resolution, par penalisation-dualité, d'une classe de problems de dirichlet non lineares," *Revue Francaise d'Automatique, Informatique, et Recherche Opérationelle*, vol. 9, pp. 41–76, 1975.

[6] E. Hazan, A. Agarwal, and S. Kale, "Logarithmic regret algorithms for online convex optimization," *Mach. Learn.*, vol. 69, no. 2/3, pp. 169–192, 2007.

[7] B. He and X. Yuan, "On the $\mathcal{O}(1/n)$ convergence rate of the Douglas-Rachford alternating direction method," *SIAM J. Numer. Anal.*, vol. 50, no. 2, pp. 700–709, 2012.

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

[9] S. Hosseini, A. Chapman, and M. Mesbahi, "Online distributed ADMM on networks," arXiv:1412.7116, 2014.

[10] A. Koppel, F. 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.

[11] S. Lee, A. Nedich, and M. Raginsky, "Coordinate dual averaging for decentralized online optimization with nonseparable global objectives," *IEEE Trans. Control Netw. Syst.*, vol. 5, no. 1, pp. 34–44, Mar. 2018.

[12] S. Lee and M. M. Zavlanos, "Distributed primal-dual methods for online constrained optimization," in *Proc. IEEE Amer. Control Conf.*, 2016, pp. 7171–7176.

[13] D. Mateos-Nunez and J. Cortes, "Distributed online convex optimization over jointly connected digraphs," *IEEE Trans. Netw. Sci. Eng.*, vol. 1, no. 1, pp. 23–37, Jan. 2014.

[14] D. Mateos-Núñez and J. Cortés, "Distributed online second-order dynamics for convex optimization over switching connected graphs," in *Proc. 21st Int. Symp. Math. Theory Netw. Syst.*, Groningen, The Netherlands, 2014, pp. 15–22.

[15] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Trans. Autom. Control*, vol. 54, no. 1, pp. 48–61, Jan. 2009.

[16] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *Proc. Symp. Inf. Process. Sensor Netw.*, Berkeley, CA, USA, Apr. 2004, pp. 20–27.

[17] S. Shalev-Shwartz, "Online learning and online convex optimization," *Found. Trends Mach. Learn.*, vol. 4, no. 2, pp. 107–194, 2011.

[18] J. N. Tsitsiklis, D. P. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE Trans. Autom. Control*, vol. AC-31, no. 9, pp. 803–812, Sep. 1986.

[19] H. Wang and A. Banerjee, "Online alternating direction method," in *Proc. Int. Conf. Mach. Learn.*, Edinburgh, U.K., Jul. 2012, pp. 1119–1126.

[20] H. Wang and A. Banerjee, "Online alternating direction method (longer version)," arXiv:1306.3721, 2013.

[21] E. Wei and A. Ozdaglar, "Distributed alternating direction method of multipliers," in *Proc. IEEE Conf. Decis. Control*, Maui, HI, USA, 2012, pp. 5445–5450.

[22] E. Wei and A. Ozdaglar, "On the $\mathcal{O}(1/k)$ convergence of asynchronous distributed alternating direction method of multipliers," arXiv:1307.8254, 2013.

[23] M. Zinkevich, "Online convex programming and generalized infinitesimal gradient ascent," in *Proc. 20th Int. Conf. Mach. Learn.*, Washighton, DC, USA, 2003, pp. 928–936.

[24] A. Makhdoumi and A. Ozdaglar, "Convergence rate of distributed ADMM over networks," *IEEE Trans. Autom. Control*, vol. 62, no. 10, pp. 5082–5095, Oct. 2017.