Distributed ADMM over directed networks

Kiran Rokade and Rachel Kalpana Kalaimani

Abstract—Distributed optimization over a network of agents is ubiquitous with applications in areas including power system control, robotics and statistical learning. In many settings, the communication network is directed, i.e., the communication links between agents are unidirectional. While several variations of gradient-descent-based primal methods have been proposed for distributed optimization over directed networks, an extension of dual-ascent methods to directed networks remains a less-explored area. In this paper, we propose a distributed version of the Alternating Direction Method of Multipliers (ADMM) for directed networks. ADMM is a dual-ascent method that is known to perform well in practice. We show that if the objective function is smooth and strongly convex, our distributed ADMM algorithm achieves a geometric rate of convergence to the optimal point. Through numerical examples, we observe that the performance of our algorithm is comparable with some state-of-the-art distributed optimization algorithms over directed graphs. Additionally, our algorithm is observed to be robust to changes in its parameters.

Index Terms—ADMM, dual-ascent, directed graphs, distributed computing, parallel computing.

I. INTRODUCTION

In this paper, we focus on solving an unconstrained optimization problem of the form

$$\min_{x \in \mathbb{R}} \sum_{i=1}^{n} f_i(x),$$

(P1)

where $x \in \mathbb{R}$ is the decision variable. The computations involved in solving the problem are performed by a group of $n$ agents which are part of a communication network. Due to the restrictions imposed by the network, an agent can communicate only with its neighboring agents. For each $i \in \{1, \ldots, n\}$, $f_i : \mathbb{R} \rightarrow \mathbb{R}$ is a convex function privately known to agent $i$. The goal of each agent is to solve (P1), without explicitly revealing its function $f_i$. This restriction on the exchange of $f_i$’s could arise due to privacy reasons or due to the communication cost involved in such an exchange. Since each agent minimizes the sum of all functions, it computes a “globally optimal” solution without revealing its own function to other agents. Such distributed optimization problems are encountered in several applications such as achieving average consensus in a distributed manner, distributed power system control, formation control of robots, statistical learning etc [1].

Distributed optimization problems have been widely studied in the literature. Existing methods for solving these problems can be broadly classified into two types: gradient-descent-based primal methods [2]–[5] and Lagrangian-based dual-ascent methods [6]–[9]. While most of the methods in the literature assume that the communication network is undirected [6]–[8], i.e., communication links between agents are bidirectional, in practice, the network is often directed, i.e., the links are unidirectional. For example, two agents can have a different broadcast range or the network can have varying levels of interference. Generalizing distributed optimization methods, which work for undirected graphs, to directed graphs is not straightforward and in the past has required some novel concepts. Gradient-descent methods for directed networks were proposed by introducing new concepts such as balancing weights [5], push-sum [2], push-DIGing (distributed in-exact gradient tracking) [4] etc. On the other hand, there is little work on Lagrangian-based dual-ascent methods for directed graphs. Recently, a dual-ascent method called PANDA (primarily averaged network dual ascent) [8] was proposed which can be extended to directed graphs. However, implementing PANDA requires the knowledge of a doubly-stochastic weight matrix compatible with the underlying directed graph. Finding such a matrix in a distributed manner is a non-trivial task [5].

A special type of Lagrangian-based dual-ascent method is the alternating direction method of multipliers (ADMM). ADMM enjoys the benefit of decomposition given by standard dual-ascent methods, while the presence of an additional penalty term in the Lagrangian gives ADMM superior convergence properties [10]. ADMM has been studied long back, e.g., in the 1980s [11], but there has been a recent surge in its popularity due to its numerous applications in various distributed optimization problems such as low-rank matrix estimation [12], optimal power flow [13], covariance matrix estimation [14] etc. Distributed implementation of ADMM is well-studied for undirected graphs [6], [7]. An extension to directed graphs was non-existent in the literature (see [4]), until recently. A method called DC-DistADMM (Directed Constrained Distributed ADMM) was proposed in [9] which uses the idea of push-sum to deal with the asymmetric nature of a directed graph. Independent of this work, we propose another ADMM algorithm for distributed optimization over directed graphs. The main difference between our algorithm and the one in [9] is that instead of the push-sum method, we use balancing weights which are updated dynamically.
in a distributed manner. Through numerical examples, it is observed that our algorithm performs better than DC-DistADMM in terms of convergence rate and communication cost.

Our main contributions are as follows.

- We propose an ADMM algorithm which solves (P1) in a distributed manner over directed networks. At each iteration, our algorithm runs an “inner loop” where each agent computes the average of the primal variable iterates of all agents in a distributed manner in B communication rounds. The method used to compute this average is based on the ideas of balancing weights [5] and dynamic average consensus [15]. The freedom in choosing B is important to guarantee convergence. We provide an explicit lowerbound on B.

- Under the assumption that the $f_i$’s are strongly convex and smooth, we show that the primal-dual iterates of the algorithm converge to their unique optimal points at a geometric rate, provided some explicit bounds on B and the penalty parameter of the ADMM algorithm are satisfied.

- Through numerical examples, we demonstrate that the performance of our algorithm is comparable with some state-of-the-art algorithms for distributed optimization over directed graphs, while it is better than DC-DistADMM in terms of convergence rate and communication cost.

- We also demonstrate that the algorithm is robust to changes in its parameters.

Rest of the paper is organized as follows. In Section II we formally describe the problem of solving (P1) in a distributed manner over a directed graph. In Section III we present our distributed ADMM algorithm for solving (P1). In Section IV we state our main result which guarantees convergence of the algorithm. We also give an outline of the proof of this result. In Section V we present some numerical examples to support our main result and to compare the performance of our algorithm with some other distributed algorithms over directed graphs. Finally, we give some concluding remarks in Section VI. Proofs of all the results can be found in the appendix.

Notation: Let log denote logarithm with base 10. $1$ is the vector of all ones. For a vector $x$, $x_1$ is 1th element of $x$, $x^T$ is the transpose of $x$ and $\|x\|$ denotes the 2-norm of $x$. We use $\{s^k\}$ for the sequence $\{s^k\}_{k \geq 0}$. For a matrix $M$, let $\|M\|$ be the induced 2-norm of $M$. Let $\text{diag}(x)$ be the diagonal matrix with elements $x_1, \ldots, x_n$ on the diagonal. For a function $f$, $f^*$ denotes the conjugate of $f$, i.e., $f^*(a) = \sup_{x} a^T x - f(x)$ for all $a \in \mathbb{R}^n$ such that the supremum is finite. $\nabla f$ is the gradient of $f$.

II. PROBLEM FORMULATION

Consider a set of $n$ agents denoted by $V = \{1, \ldots, n\}$. The communication pattern between the agents is depicted by a directed graph $G = (V, E)$, where $E$ is the set of all directed edges. We denote $(i,j) \in E$ if there exists a directed edge from agent $j$ to agent $i$. The set $\{j \in V : (i,j) \in E\}$ is called the set of all in-neighbours of agent $i$, while $\{j \in V : (j,i) \in E\}$ is called the set of all out-neighbours of agent $i$. The number of in-neighbours (resp. out-neighbours) of node $i$ is called its in-degree denoted by $d_i^\text{in}$ (resp. out-degree denoted by $d_i^\text{out}$). Following assumption is standard in the literature when agents desire to reach consensus by communicating over a directed graph (see [2], [5], [8]).

Assumption 1. The graph $G$ is strongly connected, i.e., there exists a directed path between each pair of agents.

The agents must cooperatively solve the optimization problem (P1), which can be equivalently written as

$$\begin{align*}
\min_{x_i \in \mathbb{R}} & \quad \sum_{i=1}^{n} f_i(x_i) \\
\text{s.t.} & \quad x_i = x_j \text{ for all } i, j \in V,
\end{align*}$$

(P2)

where $f_i : \mathbb{R} \to \mathbb{R}$ is a convex function known only to agent $i$ and $x_i \in \mathbb{R}$ is the decision variable of agent $i$. The constraint $x_i = x_j$ for all $i, j \in V$ is called the consensus constraint.

To write (P2) in a compact form, let $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$ be the vector of all decision variables. Then, the consensus constraint can be written as

$$x = (11^T/n)x.$$

Further, let $f(x) = \sum_{i=1}^{n} f_i(x_i)$. Then, (P2) can be compactly written as

$$\begin{align*}
\min_{x \in \mathbb{R}^n} & \quad f(x) \\
\text{s.t.} & \quad x = (11^T/n)x.
\end{align*}$$

(P3)

We assume that (P3) is solvable, i.e., there exists an optimal point $x^* \in \mathbb{R}^n$ of (P3). We propose to solve (P3) in a distributed manner using ADMM. We make the following assumptions on the function $f$. These assumptions are standard in the literature whenever a geometric rate of convergence is desired (see [4], [7], [8]).

Assumption 2. Consider the objective function $f$ in (P3).

1) $f$ is differentiable.

2) $f$ is $\mu$-strongly convex, i.e., $\exists \mu > 0$ such that for all $x_1, x_2 \in \mathbb{R}^n$, $f(x_2) \geq f(x_1) + \nabla f(x_1)^T(x_2 - x_1) + (\mu/2)\|x_1 - x_2\|^2$.

3) $f$ is $L$-smooth, i.e., $\exists L \geq 0$ such that for all $x_1, x_2 \in \mathbb{R}^n$, $\|\nabla f(x_1) - \nabla f(x_2)\| \leq L\|x_1 - x_2\|$.

We reformulate (P3) by introducing a new variable $y \in \mathbb{R}^n$. This reformulation enables us to derive a distributed ADMM algorithm in which the effect of the communication network is isolated to a single update step, while the rest of the algorithm follows from the standard ADMM algorithm. The reformulated problem is given by

$$\begin{align*}
\min_{x \in \mathbb{R}^n, y \in \mathbb{R}^n} & \quad f(x) \\
\text{s.t.} & \quad x = y, \ y = (11^T/n)y.
\end{align*}$$

(P4)

Note that this reformulation decouples the decision variables $x_1, \ldots, x_n$ of the agents from each other. Next, we transfer
the constraint \( y = (11T/n)y \) into the objective function using an indicator function as follows. Define
\[
I(y) = \begin{cases} 0 & \text{if } y = (11T/n)y, \\ \infty & \text{otherwise}. \end{cases}
\]
Now, \((P4)\) is equivalent to
\[
\min_{x \in \mathbb{R}^n, y \in \mathbb{R}^n} f(x) + I(y) \\
\text{s.t.} \quad x = y. 
\]
(P5)
Since \( f \) is strongly convex, \((P5)\) has a unique optimal point \((x^*, y^*)\) where \( x^* \) is the minimizer of \( f \) that satisfies the constraint \((11T/n)x^* = x^*\).

We define the augmented Lagrangian of \((P5)\) as
\[
L_\rho(x, y, a) = f(x) + I(y) + a^T(x - y) + \frac{\rho}{2}\|x - y\|^2,
\]
where \( a \in \mathbb{R}^n \) is the dual variable associated with the constraint \( x = y \) and \( \rho > 0 \) is the penalty parameter. The term \((\rho/2)\|x - y\|^2\) is called the penalty term. Let \( a^* \in \mathbb{R}^n \) be a dual optimal point of \((P5)\). In the next section, we propose our distributed ADMM algorithm which generates sequences \(\{a^k\}\) and \(\{x^k\}\) of primal-dual iterates that converge to \(a^*\) and \(x^*\) respectively.

III. ALGORITHM

We begin by writing the standard 2-block ADMM algorithm (see [10]) for \((P1)\) as follows. For simplicity, all iterates are initialized at zero.\(^3\) For each \( k \geq 0 \),
\[
x^{k+1} = \arg\min_{x \in \mathbb{R}^n} L_\rho(x, y^k, a^k), \\
y^{k+1} = \arg\min_{y \in \mathbb{R}^n} L_\rho(x^{k+1}, y, a^k), \\
a^{k+1} = a^k + \rho(x^{k+1} - y^{k+1}). \tag{4}
\]
In the algorithm above, the primal iterates \((x^k, y^k)\) are divided in two blocks which are updated sequentially, before updating the dual iterate \(a^k\). Note that the second block of primal iterates is updated using the latest available value of the first block of primal iterates.

We consider that each agent \( i \) maintains the iterates \(\{x^k_i, y^k_i, a^k_i\}\) for all \( k \geq 0 \). To implement the algorithm above in a distributed manner, each agent must update its iterates using only the information received from its in-neighbours. We analyze each update step above to see if such a distributed implementation is possible.

\( x \) update step: Consider the \( x \) update step given in \((3)\). Substituting the expression for \( L_\rho \) from \((1)\), this step can be written as
\[
x^{k+1} = \arg\min_{x \in \mathbb{R}^n} f(x) + (a^k)^T x + \frac{\rho}{2}\|x - y^k\|^2. \tag{5}
\]
Note that \( f \) can be decomposed as the sum of \( f_i \)'s. Such a decomposition is a feature that ADMM inherits from the dual-ascent method. Due to this decomposition, the update step above is equivalent to
\[
x^{k+1}_i = \arg\min_{x_i \in \mathbb{R}} f_i(x_i) + (a^k_i)^T x_i + \frac{\rho}{2}(x_i - y^k_i)^2. \tag{6}
\]
\(^3\)A more general initialization is provided later in Remark \(8\).

for all \( i \in V \). Each agent \( i \) can independently implement the update step above by solving an unconstrained minimization problem using local information such as the function \( f_i \) and the iterates \(\{a^k_i, y^k_i\}\).

\( y \) update step: Consider now the \( y \) update step given in \((2)\). We can derive an explicit expression for \( y^{k+1} \) as shown by the following result.

Lemma 1. The \( y \) update step given in \((3)\) is equivalent to
\[
y^{k+1}_i = \frac{1}{n} \sum_{j=1}^{n} \left( x^{k+1}_j + \frac{a^k_{ij}}{\rho} \right) \text{ for all } i \in V. \tag{7}
\]
In other words, for each \( i \in V \), \( x^{k+1}_i \) is the average of all \( x^{k+1}_j + a^k_{ij}/\rho, j \in V \). Computing an average of a set of values in a distributed manner over a network is a widely studied problem [16], [17]. The problem is particularly challenging if the underlying network is directed. Several methods have been proposed to solve this problem over directed graphs [5], [18]. Further, note that \((7)\) involves computing the average of a quantity that is “time-varying”. For such problems, a scheme called dynamic average consensus is known to perform well [15]. Compared to the static average consensus rule, a dynamic average consensus rule uses the first-order difference of the iterates to give a better rate of convergence [19]. We use the idea of balancing weights [5] along with a dynamic average consensus type update rule [15] to compute an estimate of \( y^{k+1} \) in a distributed manner.

Our update rule to estimate \( y^{k+1} \) is as follows. Consider an agent \( i \in V \) and an iteration \( k \geq 0 \). Let the agent’s estimate of \( y_i^{k+1} \) be denoted by \( \zeta_i^{k+1}(\cdot) \). This estimate is updated dynamically. Let the estimate be initialized as \( \zeta_i^{k+1}(0) = y_i^k + x_i^{k+1} - x_i^k + (a_i^k - a_i^{k-1})/\rho\).\(^4\) The intuition behind this initialization is to perturb \( y_i^k \), the last-known estimate of this average, by \( x_i^{k+1} - x_i^k + (a_i^k - a_i^{k-1})/\rho \), the change in agent \( i \)'s contribution to the average. Such an initialization is typical of a dynamic average consensus type update rule [19]. Note that this initialization implicitly assumes that agent \( i \) knows the exact value of \( y_i^k \), which may not be possible. This assumption can be relaxed with the following observation. From \((4)\), we have \( x_i^k - y_i^k = (a_i^k - a_i^{k-1})/\rho \). Hence, \( \zeta_i^{k+1}(0) = x_i^{k+1} \). Thus, initializing the estimate \( \zeta_i^{k+1}(\cdot) \) is straightforward. Now, the estimate is updated as follows. Consider an integer \( B \geq 1 \) which satisfies a lower-bound to be specified later. In each iteration \( k \geq 0 \), the agents run an “inner loop” \( B \) number of times where they communicate and update their estimate \( \zeta_i^{k+1}(\cdot) \).

\[\zeta_i^{k+1}(b + 1) = \zeta_i^{k+1}(b)(1 - d_i^{out}(b)) + \sum_{j \in V:\langle i,j \rangle \in E} w_{ij}(b) \zeta_j^{k+1}(b). \tag{8}\]

Recall that \( d_i^{out} \) is the out-degree of agent \( i \). We refer to \( w_{ij}(b) \in \mathbb{R} \) as the weight used by agent \( j \) to scale its outgoing information. To ensure that the agents reach consensus when the underlying graph is directed, the weights must be chosen

\(^4\)We use the notation \( a^{-1} = 0 \).
appropriately. One such set of weights are the “balancing weights” of a graph. We next recall the notion of balancing weights introduced in [5] and describe our update rule used to compute the weights in (8).

Let $D_{\text{out}} := \text{diag}(d_{1\text{out}}, \ldots, d_{n\text{out}})$ be the diagonal matrix of out-degrees of the agents. Let $A \in \mathbb{R}^{n \times n}$ be the adjacency matrix of the graph, i.e., $A_{ij} = 1$ if $(i,j) \in E$ and $A_{ij} = 0$ otherwise. Given a vector $w \in \mathbb{R}^n$ of weights, let $W = I - (D_{\text{out}} - A)\text{diag}(w)$ be the weight matrix associated with $w$.

**Definition 1. (Balancing weights)** Given a graph $G$, a vector $w$ is said to be a set of balancing weights of $G$ if the weight matrix $W = I - (D_{\text{out}} - A)\text{diag}(w)$ associated with $w$ is doubly-stochastic, i.e., $W1 = 1, 1^TW = 1^T$.

For a balanced graph, i.e., a graph where each node has the same in-degree and out-degree, the vector $w = 1$ is trivially the set of balancing weights of the graph. A directed graph is not necessarily balanced and hence computing its set of balancing weights is non-trivial.

We use the same update rule as described in [5] to compute the set of weights used in (8). Let $d^*_i := \max_{j \in V} d_{ij}$ be the maximum out-degree of the graph and $D$ be the diameter of the graph. Following [5], we initialize the weights as $w_i^k(0) \leq (1/d^*_i)^{2D+1}$. Then, for all $k \geq 0$ and $b \in \{0, B - 1\}$,

$$w_i^{k+1}(b + 1) = \frac{1}{2} \left( w_i^{k+1}(b) + \frac{1}{d^*_i} \sum_{j \in V: (i,j) \in E} w_j^{k+1}(b) \right)$$

(9)

and $w_i^{k+2}(0) = w_i^{k+1}(B)$. The update rule can be written compactly as

$$w_i^{k+1} = P w_i^k$$

(10)

and $w^{k+1}(0) = w^k(B)$, where $P := (I + A(D_{\text{out}} - 1)/2)$. It is known that the weights updated according to this rule converge to the set of balancing weights of the graph [5]. Additionally, we require that the weight matrix associated with $w^k(b)$ has a bounded norm. These properties are stated in our next result.

**Lemma 2.** Let $W^k(b) = I - (D_{\text{out}} - A)\text{diag}(w^k(b))$ be the weight matrix associated with the weights $w^k(b)$ updated using (10). The matrix has the following properties.

1) For all $k \geq 0, b \in [0, B - 1]$, $1^TW^k(b) = 1^T$ and $W^k(b)p^k(b) = p^k(b)$ for some vector $p^k(b)$ such that $p^k(b)^T1 = 1$.

2) For all $b \in [0, B - 1]$, $\lim_{k \to \infty} p^k(b) = 1/n$.

3) $\exists k \geq 0, \delta \geq 0$ such that $\|W^k(b) - p^k(b)1^T\| \leq \delta$ for all $k \geq k, b \in [0, B - 1]$.

4) $\exists k \geq 0, M \geq 0$ such that $\|W^k(b)\| \leq M$ for all $k \geq k, b \in [0, B - 1]$.

In words, points 1) and 2) of Lemma 2 state that $W^k(b)$ is left-stochastic for all $b, k$ and it is right-stochastic for each $b$ in the limit $k \to \infty$. Further, points 3) and 4) guarantee additional properties of the weight matrix $W^k(b)$ which will be useful in proving convergence of the algorithm.

With the notation introduced above, the update rule for the estimate $\zeta_i^{k+1}()$ of $y_i^{k+1}$ given in (8) can be written compactly as

$$\zeta_i^k(b + 1) = W^k(b)\zeta_i^k(b)$$

(11)

where $\zeta_i^0(0) = x_i^k$.

Finally, we look at the distributed implementation of the dual update step.

**a update step:** From (4), we have $a_i^{k+1} = a_i^k + \rho(x_i^{k+1} - y_i^{k+1})$. Thus, each agent $i \in V$ can compute $a_i^{k+1}$ in an independently using its set of iterates $\{x_i^{k+1}, y_i^{k+1}\}$.

**Remark 1.** The agents need to communicate with their neighbours ($B$ times per iteration) only during the $y$ update step.

The steps implemented by each agent $i \in V$ are summarized in our main algorithm below.

**Algorithm 1 Distributed ADMM at agent $i \in V$**

Initialize $x_i^0 = y_i^0 = a_i^0 = 0, w_i^1(0) \leq (1/d^*_i)^{2D+1}$

for $k = 0, 1, \ldots$

Compute $x_i^{k+1}$ using (6).

Initialize $\zeta_i^0(0) = x_i^{k+1}$

for $b = 0, \ldots, B - 1$

Send $w_i^{k+1}(b)$ and $\zeta_i^{k+1}(b)$ to out-neighbours.

Compute $w_i^{k+1}(b + 1)$ using (9).

Compute $\zeta_i^{k+1}(b + 1)$ using (8).

end for

Fix $w_i^{k+2}(0) = w_i^{k+1}(B)$.

Fix $y_i^{k+1} = \zeta_i^{k+1}(B)$.

Compute $a_i^{k+1} = a_i^k + \rho(x_i^{k+1} - y_i^{k+1})$.

end for

**Remark 2.** In Algorithm 1 for the ease of notation, we “rename” the final estimate $\zeta_i^{k+1}(B)$ to $y_i^{k+1}$ for all $i \in V$. Henceforth, $y_i^k, k \geq 1$ refers to the estimate $\zeta_i^k(B)$. This is not to be confused with the “original” $y$ as defined by (7) which we will not refer to in the rest of the paper.

**Remark 3.** In proving convergence of Algorithm 1 we use the fact that $1^Ty^k = 1^Tx^k, 1^Ta^k = 0$ for all $k \geq 0$. It is easy to verify that if we initialize the iterates such that $1^Ty^0 = 1^Tx^0$ and $1^Ta^0 = 0$, the condition above is satisfied. This is due to the fact that $1^TW^k(b) = 1^T$ as given in Lemma 2.

Next, we present our main result which states that if the parameters of Algorithm 1 are chosen appropriately, then its primal-dual iterates converge to their respective optimal points with a geometric rate.

**IV. MAIN RESULT**

**Theorem 1.** Given a problem of the form (5), consider Algorithm 1 where the number of communication rounds per iteration satisfy

$$B \geq \max \left\{ 1, \left[ \frac{\log(\gamma_1 \gamma_3)}{\log(1/\delta)} \right] \right\},$$

(12)
the convergence rate $\lambda \in (0, 1)$ satisfies
\[
2L \left( \frac{1}{\lambda^2} - 1 \right) < \left( \frac{\lambda}{\lambda + 1} \right) \min \left\{ \frac{1}{M_\beta}, \frac{c_3}{c_3 + 1} \right\}
\]
and the penalty parameter satisfies
\[
\rho \in \left( 2L \left( \frac{1}{\lambda^2} - 1 \right), \left( \frac{\lambda \mu}{\lambda + 1} \right) \min \left\{ \frac{1}{M_\beta}, \frac{c_3}{c_3 + 1} \right\} \right)
\]
where
\[
\begin{align*}
\gamma_1 &= 1/\mu, \\
\gamma_3 &= (1 + \rho c_1) \sqrt{\rho(L + \beta)/c_2}, \\
c_1 &= (\lambda + 1)/(\lambda \mu - \rho(\lambda + 1)), \\
c_2 &= (1 + \rho/L - 1/\lambda^2 - \rho^3 \lambda^2 (1/\beta + 1/\mu)), \\
c_3 &= 1/\sqrt{2L(1/\beta + 1/\mu)}.
\end{align*}
\]
$\beta > 0$ is arbitrary, while $\delta \in [0, 1)$ and $M \geq 0$ are as defined in Lemma 2. Then, the iterates of Algorithm 2 satisfy
\[
\|a^k - a^*\| \leq c_\alpha \lambda^k, \quad \|x^k - x^*\| \leq c_x \lambda^k
\]
for some non-negative constants $c_\alpha, c_x$, where $(x^*, a^*)$ is the unique primal-dual optimal point of (P5).

Remark 4. It is easy to check that a (large enough) $\lambda \in (0, 1)$ which satisfies (13) always exists. As $\lambda \to 1$, LHS of (13) converges to zero, while the RHS converges to a positive number.

Remark 5. If $f$ is $\mu$-strongly convex and $L$-smooth (as defined in Assumption 2), then $\kappa = L/\mu$ is called the condition number of $f$. We can argue that the convergence of Algorithm 1 is faster if the problem is well-conditioned, i.e., $\kappa \geq 1$ is small. To see this, note that with a decrease in $\kappa$, the LHS of (13) decreases while the RHS increases. Thus, smaller values of $\kappa$ give a wider range of values of the convergence rate $\lambda$ that satisfy (13).

Next, we give an outline of the proof of Theorem 1. The proof uses the idea of the small gain theorem, which was proposed for showing convergence of distributed optimization algorithms in [4], and was later also used in [8].

We briefly recall the small gain theorem after introducing some notation. Given a sequence $\{s^k\}$ of vectors in $\mathbb{R}^n$ and a convergence rate $\lambda \in (0, 1)$, let
\[
\|s\|_{\lambda,K} = \max_{k=0,\ldots,K} \frac{\|s^k\|}{\lambda^k}, \quad \|s\|_{\lambda} = \sup_{k \geq 0} \frac{\|s^k\|}{\lambda^k}
\]
for all $K \geq 0$. Further, given two sequences $\{s^k_1\}, \{s^k_2\}$ and a constant $\gamma \geq 0$, let us use the notation $s_1 \xrightarrow{\gamma} s_2$ to denote
\[
\|s_2\|_{\lambda,K} \leq \gamma \|s_1\|_{\lambda,K} + \omega
\]
for all $K \geq 0$, for some constant $\omega \geq 0$. Now, the small gain theorem can be stated as follows.

**Proposition 1.** [4, Theorem 3.7] (Small gain theorem) Given $m$ sequences $\{s^k_1\}, \ldots, \{s^k_m\}$ of vectors in $\mathbb{R}^n$, suppose there exist non-negative constants $\gamma_1, \ldots, \gamma_m$ and a convergence rate $\lambda \in (0, 1)$ satisfying the cycle of relations
\[
s_1 \xrightarrow{\gamma_1} s_2 \xrightarrow{\gamma_2} s_3 \ldots s_m \xrightarrow{\gamma_m} s_1
\]
such that $\gamma_1 \ldots \gamma_m < 1$. Then, there exist constants $c_j \geq 0$ such that $\|s^j\|_{\lambda} \leq c_j$ for all $j \in \{1, \ldots, m\}$.

Note that the small gain theorem implies $s^j_k \to 0$ at a geometric rate $\lambda$.

We use the small gain theorem to prove Theorem 1 as follows. Let $\tilde{a}^k - a^k$ and $\tilde{x}^k - x^k$. To show that $a^k \to a^*$ and $x^k \to x^*$ with a geometric rate of $\lambda$, it is enough to show that $\|\tilde{a}\|_{\lambda}$ and $\|\tilde{x}\|_{\lambda}$ are bounded. To show this, we first prove that there exist non-negative numbers $\gamma_1, \gamma_2, \gamma_3$ such that the cycle of relations
\[
\tilde{a} + \rho \Delta y \xrightarrow{\gamma_1} x_\perp \xrightarrow{\gamma_2} y_\perp \xrightarrow{\gamma_3} \tilde{a} + \rho \Delta y
\]
holds with $\gamma_1 \gamma_2 \gamma_3 < 1$, where $\Delta y_k = y_k - y_{k-1}, x_k = x_k - (11^T/n)x_k, y_\perp = y_k - (11^T/n)y_k$ for all $k \geq 0$. The proof of each arrow in (17) can be found in Appendix D. In particular, we show that the conditions given in Theorem 1 are sufficient to prove the cycle of relations in (17) such that $\gamma_1 \gamma_2 \gamma_3 < 1$. Then, by Proposition 1, it follows that $\|y_\perp\|_{\lambda}$ and $\|\tilde{a} + \rho \Delta y\|_{\lambda}$ are bounded.

After proving that $\|y_\perp\|_{\lambda}$ and $\|\tilde{a} + \rho \Delta y\|_{\lambda}$ are bounded, it is shown in Appendix D that
\[
\begin{align*}
\|\tilde{a}\|_{\lambda} &\leq \gamma_0 \|y_\perp\|_{\lambda} + \omega_0, \\
\|\tilde{x}\|_{\lambda} &\leq \gamma_0 \|\tilde{a}\|_{\lambda} + \rho \Delta y + \omega_x
\end{align*}
\]
for some non-negative constants $\gamma_0, \omega_0, \omega_x$. Then, it follows that $\|\tilde{a}\|_{\lambda} \leq c_\alpha$ and $\|\tilde{x}\|_{\lambda} \leq c_x$ for some non-negative constants $c_\alpha$ and $c_x$. From the definition of $\|s\|_{\lambda}$ given in (15), this implies $\|\tilde{a}\| \leq c_\alpha \lambda^k$ and $\|\tilde{x}\| \leq c_x \lambda^k$ for all $k \geq 0$, which completes the proof of Theorem 1.

Next, we present numerical examples to illustrate our main result and to compare our ADMM algorithm with other distributed optimization algorithms over directed graphs.

**V. NUMERICAL EXAMPLES**

**Problem setup:** We consider a strongly connected, directed graph of $n = 50$ agents as shown in Fig. 1. The graph is generated as a wireless sensor network (WSN). Each node is placed uniformly at random in the unit square and is assumed to have a broadcast radius of 0.3. We consider local agent functions of the form $f_i(x) = (1/2)\|H_i x - g_i\|^2$, where $H_i \in \mathbb{R}^{50 \times 10}, g_i \in \mathbb{R}^{10}$ are generated randomly.

**Fig. 1.** A strongly connected, directed graph of $n = 50$ agents

5We use the notation $y^{-1} = 0$. 

Algorithms chosen for comparison: We solve problem (12) using various distributed optimization algorithms for a directed graph. Other than the ADMM algorithm proposed in this paper (Algorithm 1), we consider DC-DistADMM [9]. The main difference between DC-DistADMM and our algorithm is the way in which the average of primal variable iterates is computed at each iteration. DC-DistADMM uses an ε-consensus protocol [9] based on the idea of push-sum consensus [2], while our algorithm uses the the idea of balancing weights [5] and dynamic average consensus [15]. While the former are Lagrangian-based dual-ascent algorithms, we also compare the performance of our algorithm with some gradient-based primal algorithms such as subgradient-push [2], DIGing [4] and distributed heavyball [20]. Note that the bounds on the parameters of each of these algorithms, such as the ones on ρ and B given in Theorem 1, are often very conservative. Hence, adopting the same approach as [4], we tune the parameters of each algorithm manually until the observed convergence rate cannot be improved.

Convergence rate: A plot of the normalized primal residual obtained by each of the algorithms mentioned above is shown in Fig. 2. It can be observed that the convergence rate due to ADMM is comparable to these state-of-the-art algorithms. Note that in each iteration of DC-DistADMM, the agents have to communicate with each other until the average of the primal variables is within a pre-specified bound given by the parameter ε [9]. Hence, each iteration possibly requires multiple rounds of communication. For this particular example, the average number of communication rounds per iteration of DC-DistADMM was observed to be 176. On the other hand, ADMM required only one round of communication per iteration in this example (B = 1). The effect of different values of B on the performance of ADMM is analyzed later. It is worth noting that ADMM and DC-DistADMM also generate a sequence \{a^k\} of dual variable iterates which converge to the dual optimal point a^* (not shown here due to space constraints). This is not the case with gradient-based algorithms.

Communication cost: For each algorithm, the number of values sent by an agent per communication round is \(\Omega(n)\), where \(n\) is the number of agents in the network.

Effect of B on the convergence rate of ADMM: We study the effect of changing the number of communication rounds per iteration \(B\) in the inner loop of our ADMM algorithm. As observed in Fig. 3 (left), increasing the number of communication rounds per iteration from \(B = 1\) to \(B = 2\) improves the convergence rate. However, further increasing the value to \(B = 4\) and \(B = 10\) slows down the overall convergence of the algorithm. Thus, choosing an optimal value of \(B\) is a non-trivial task and requires further investigation.

Robustness of ADMM to changes in ρ: Recall that the convergence of our ADMM algorithm is guaranteed if the penalty parameter \(ρ\) is within certain bounds as given in (14). However, in practice, the algorithm is robust to large changes in the value of \(ρ\). This is illustrated in Fig. 3 (right), where it can be observed that the primal residual decreases monotonically for a large range of values of \(ρ\), albeit with different rates. On the other hand, it was observed that the gradient-based algorithms, i.e., subgradient-push, DIGing and distributed heavyball fail to converge if their step-sizes are increased beyond a certain threshold.

VI. CONCLUSION

We proposed an ADMM algorithm to solve distributed optimization problems over directed graphs. Our algorithm uses the ideas of balancing weights and dynamic average consensus. Under the assumption that the objective function is strongly convex and smooth, we showed that the primal-dual iterates of the algorithm converge to their unique optimal points at a geometric rate, provided the parameters of the algorithm are chosen appropriately. Through a numerical example, we demonstrated that the performance of our algorithm is comparable to other state-of-the-art algorithms for distributed optimization over directed graphs. Moreover, its performance was observed to be better than DC-DistADMM [9], which requires significantly more communication rounds to obtain the same level of accuracy as our algorithm. Additionally, the algorithm was observed to be robust to changes in its parameters. In the future, it will be interesting to see if convergence of the algorithm can be guaranteed by relaxing the assumptions of strong convexity and smoothness. Further, it will also be interesting to extend the algorithm to time-varying graphs.
Appendix

A. Proof of Lemma 1

The update step in (3) can be written as

$$y^{k+1} = \arg \min_{y \in (\mathbb{R}^n)^y} ||y - (x^{k+1} + a^k / \rho)||^2.$$  (19)

We parameterize the constraint set of (19), i.e., $\mathcal{C} = \{ y \in \mathbb{R}^n : y = (11T/n)y \}$, using the parameter $\nu \in \mathbb{R}$ as $\mathcal{C} = \{ y(\nu) \in \mathbb{R}^n : y(\nu) = (11T/n)(x^{k+1} + a^k / \rho) + \nu 1 \}$. With this parameterization, let $y(\nu^*)$ be the minimizer of (19). We show that $\nu^* = 0$ as follows. By definition,

$$y^{k+1} = y(\nu^*) = \arg \min_{y(\nu) = (11T/n)(x^{k+1} + a^k / \rho) + \nu 1} ||y(\nu) - (x^{k+1} + a^k / \rho)||^2.$$  

This implies

$$\nu^* = \arg \min_{\nu \in \mathbb{R}} ||(I - 11T/n)(x^{k+1} + a^k / \rho) - \nu 1||^2 = \arg \min_{\nu \in \mathbb{R}} ||(I - 11T/n)(x^{k+1} + a^k / \rho)||^2 + n\nu^2 = 0,$$

where we have used the fact that the vectors $(I - 11T/n)(x^{k+1} + a^k / \rho)$ and $\nu 1$ are orthogonal. Thus, we have $y^{k+1} = y(\nu^*) = y(0) = (11T/n)(x^{k+1} + a^k / \rho)$, i.e., $y^{k+1} = (1/n)\sum_{j=1}^n (x_j^{k+1} + a_j^k / \rho)$ for all $i \in V$.

B. Proof of Lemma 2

1) follows from the fact that, by definition, $W(k)(b) = I - (Dout - A)\text{diag}(w(k))$ is column stochastic. We prove 2) as follows. It is shown in [5, Lemma 1] that $\lim_{k \to \infty} P_k$ exists. Hence, $\lim_{k \to \infty} W(k)(b) = P_{B+1}(I, B^{-1})W_1(0)$ exists and is independent of $b$. This implies $\lim_{k \to \infty} W(k)(b)$ exists and is independent of $b$. Let $W^\infty := \lim_{k \to \infty} W(k)(b)$. Now, we write

$$\|W(k)(b) - p^k(b)1^T\| \leq \|W(k) - W^\infty\| + \|W^\infty - 11T/n\| + \|11T/n - p^k(b)1^T\|. $$  (20)

We analyze each term in (20) one-by-one.

(i) $\|W(k)(b) - W^\infty\|$: By definition of $W^\infty$, $\|W(k) - W^\infty\| \to 0$ as $k \to \infty$.

(ii) $\|W^\infty - 11T/n\|$: It is shown in [5, Lemma 1] that $W^\infty$ is a doubly-stochastic matrix. Further, it is shown in [5, Lemma 2] that $W^\infty$ is a non-negative matrix with $|W^\infty|_{ij} > 0$ for all $i \in V$ and $|W^\infty|_{ij} > 0$ if and only if $(i, j) \in E$. Using these facts, we first show that $W^\infty(W^\infty)^T$ is a primitive matrix. To show this, it is enough to show that $(W^\infty(W^\infty)^T)^D$ is a positive matrix, where $D$ is the diameter of $G$. For any $(i, j) \in V \times V$,

$$[(W^\infty)^D]_{ij} = \sum_{i_1 \in V} \sum_{i_2 \in V} \cdots \sum_{i_D \in V} |W^\infty|_{i_1i_2} \cdots |W^\infty|_{i_Dj}.$$  

By definition, $D$ is the length of the largest path connecting any two nodes. Moreover, $G$ is strongly connected. Hence, any given $(i, j) \in V \times V$, there exists a path $(i, q_1, q_1, q_2, \ldots, (q_D, j)$ from $i$ to $j$. Hence, $[(W^\infty)^D]_{ij} \geq \|W^\infty\|_{q_1q_2} \cdots \|W^\infty\|_{q_Dj} > 0$. Thus, $(W^\infty)^D$ is a positive matrix. Now, $[W^\infty(W^\infty)^T]_{ij} \geq \|W^\infty\|_{ii}[W^\infty]_{jj} > 0$. Thus, $(W^\infty(W^\infty)^T)^D$ is also a positive matrix. It then follows that $W^\infty(W^\infty)^T$ is a primitive matrix. This implies $\rho(W^\infty(W^\infty)^T - 11T/n) < 1$ (see [21, (8.3.16)]). Now,

$$\|W^\infty - 11T/n\|^2 = \rho((W^\infty - 11T/n)(W^\infty - 11T/n)) = \rho(W^\infty(W^\infty)^T - 11T/n) < 1,$$

where we have used the facts $W^\infty 1 = 1, 1^T W^\infty = 1^T$ as shown in [5, Lemma 1]. Thus, $\|W^\infty - 11T/n\| < 1$.

(iii) $\|11T/n - p^k(b)1^T\|$: Consider any $\epsilon > 0$. We show that $\exists k_3 \geq 0$ such that $\|11T/n - p^k(b)1^T\| < \epsilon$ for all $k \geq k_3$. For all $k \geq 0$ and $l \geq 0$,

$$\|11T/n - p^k(b)1^T\| \leq \|11T/n - (W^\infty)^l\| + \|(W^\infty)^l - (W^\infty)\| + \|(W^\infty)\|_p - p^k(b)1^T\|.$$  (21)

We argued above that $W^\infty$ is a primitive matrix with $1^T W^\infty = 1^T$ and $W^\infty 1 = 1$. Hence, $\rho(W^\infty - 11T/n) < 1$. It follows that $\exists k_3 \geq 0$ such that $\|11T/n - (W^\infty)^l\| < \epsilon/3$ for all $l \geq l_1$ (see [21, (8.3.10)]). Following the same arguments as with $W^\infty$ above, we can show that $\rho(W^\infty - p^k(b)1^T) < 1$ for all $k \geq 0$ and $0 \leq b \leq B - 1$. Hence, for all $k \geq 0$, $\exists l_2 \geq 0$ such that $\|(W^\infty)^l - p^k(b)1^T\| < \epsilon/3$ for all $l \geq l_2$. Moreover, since $\lim_{k \to \infty} W^\infty W^\infty = W^\infty$, the eigenvalues of $W^\infty$ converge to those of $W^\infty$. This implies $l_2$ is independent of $k$ (see [21, (8.3.10)], which implies the rate of convergence of $(W^\infty)^l$ to $p^k(b)1^T$ depends only on the second largest eigenvalue $W^\infty$). Now, let $l_0 := \max\{l_1, l_2\}$. Note that given any matrix $C \in \mathbb{R}^{n \times n}$, $C^{l_0}$ is a continuous function of the entries of $C$. Hence, $\lim_{k \to \infty} W^\infty W^\infty = W^\infty$ implies $\exists k_1 \geq 0$ such that $\|W^\infty W^\infty - (W^\infty)1^T\|_p \leq \epsilon/3$ for all $k \geq k_1$. Substituting the aforementioned facts in (21), we have

$$\|11T/n - p^k(b)1^T\| < \epsilon$$

for all $k \geq k_3, k_1$. This implies $\|11T/n - p^k(b)1^T\| \to 0$ as $k \to \infty$.

Using the results derived for each of the three terms above, from (20) we can conclude that $\|W(k)(b) - p^k(b)1^T\| < 1$ for $k$ large-enough. Thus, we have proved the second property in Lemma 2.

Finally, 3) follows from the fact that $\|11T/n - p^k(b)1^T\| \to 0$ as $k \to \infty$, which we have proved above. To prove 4), note that for all $k \geq 1$, $\|W(k)(b)\| \leq \|W^\infty\| - p^k(b)1^T\| + \|p^k(b)1^T\|$. Now, choosing $k \geq \delta \geq k_0$ implies $\|W(k)(b) - p^k(b)1^T\| \leq \delta$ for all $k \geq k_0$. On the other hand, $\|p^k(b)1^T\|$ is bounded since $p^k(b)$ converges to $1/n$. Thus, $\|W(k)(b)\| \leq M$ for all $k \geq k$ for some $M \geq 0$.

C. Some intermediate results

We state some useful intermediate results. Let $f^*(a)$ denote the conjugate of $f$, i.e., $f^*(a) = \sup_{x} a^T x - f(x)$ for all $a \in \mathbb{R}^n$ such that the supremum is finite.
Lemma 3. The dual optimal point $a^*$ of (P5) satisfies $(11T/n)a^* = 0$.
Proof. Given the standard Lagrangian $L_0$ of (P5) (equation (1) with $\rho = 0$), the dual problem of (P5) is given by

$$\begin{align*}
\sup_a \inf \limits_{(x,y)} L_0(x,y,a) &= \sup_a \left( \inf \limits_y (f(x) + a^Ty) + \inf \limits_y (I(y) - a^Ty) \right) \\
&= \sup_a \left( -f^*(-a) + \inf \limits_y (11T/n)y - a^Ty \right) \\
&= \sup \limits_{(11T/n)a=0} -f^*(-a) .
\end{align*}$$

Thus, the dual optimal point satisfies $(11T/n)a^* = 0$. $\square$

Proposition 2. [22, Theorem 6] Under Assumption 2, $f^*$ is $1/L$-strongly convex and $1/\mu$-smooth.

Lemma 4. The primal-dual iterates of Algorithm 1 satisfy $x^{k+1} = \nabla f^*(-a^{k+1} - \rho \Delta y^{k+1})$ for all $k \geq 0$ and the primal-dual optimal points of (P5) satisfy $x^* = \nabla f^*(-a^*)$.
Proof. From the definition of $f^*$, we have

$$f^*(-a) = \sup \limits_x -a^Tx - f(x).$$

This implies

$$\nabla f^*(-a) = \arg \max \limits_x -a^Tx - f(x) = \arg \min \limits_x a^Tx + f(x).$$

(22)

Now, from the update step (5) of Algorithm 1 we have, for all $k \geq 0$,

$$\nabla f(x^{k+1}) + a^k + \rho \Delta y^{k+1} - y^k = 0.$$

Using (4), this implies

$$\nabla f(x^{k+1}) + a^{k+1} + \rho \Delta y^{k+1} = 0.$$

Hence, we have

$$x^{k+1} = \arg \min \limits_x f(x) + (a^{k+1} + \rho \Delta y^{k+1})^T x.$$

Now, using above relation with (22) gives $\nabla f^*(-a^{k+1} - \rho \Delta y^{k+1}) = x^{k+1}$, which is the first part of the lemma. Next, by definition of a primal-dual optimal point, we have

$$(x^*, y^*) = \arg \min \limits_{(x,y)} L_0(x,y,a^*),$$

where $L_0$ is the Lagrangian of (P5) (equation (1) with $\rho = 0$). This implies $x^* = \arg\min \limits_x f(x) + (a^*)^T x$. Using this relation with (22), it follows that $\nabla f^*(-a^*) = x^*$. $\square$

D. Proof of each arrow in (17)

Proof of the first arrow ($\hat{a} + \rho \Delta y \xrightarrow{\gamma_1} x_\perp$): From Proposition 2 we have for all $k \geq 0$,

$$\|\nabla f^*(-a^{k+1} - \rho \Delta y^{k+1}) - \nabla f^*(-a^*)\| \leq \frac{1}{\mu} \|a^{k+1} + \rho \Delta y^{k+1}\| = \frac{1}{\mu} \|\tilde{a}^{k+1} + \rho \Delta y^{k+1}\| .$$

Using Lemma 4 with above relation, we have

$$\|x^{k+1} - x^*\| \leq \frac{1}{\mu} \|\tilde{a}^{k+1} + \rho \Delta y^{k+1}\| .$$

Further, $\|x^{k+1} - x^*\|^2 = \|x^{k+1}\|^2 + \|\nabla f^*(a^*)\|^2$. Substituting this relation in (23), we have

$$\|x^{k+1}\| \leq \frac{1}{\mu} \|\tilde{a}^{k+1} + \rho \Delta y^{k+1}\| .$$

Hence,

$$\|x_\perp\|_{\lambda,K} \leq \frac{1}{\mu} \|\tilde{a} + \rho \Delta y\|_{\lambda,K} + \|\tilde{x}_0\|$$

for all $K \geq 0$. Thus, $\gamma_1 = 1/\mu$.

Proof of the second arrow ($x_\perp \xrightarrow{\gamma_2} y_\perp$): From (11), we have, for all $k \geq 1$ and $b \in (0, B-1]$,

$$(I - p^k(b)1^T)\xi_k^*(b + 1) = (I - p^k(b)1^T)W^k(b)c_k(b) = (W^k(b) - p^k(b)1^T)(I - p^k(b)1^T)\xi_k^*(b),$$

where we have used the first property in Lemma 2. Now,

$$(I - p^k(b)1^T \pm 11T/n)\xi_k^*(b + 1) = (W^k(b) - p^k(b)1^T)(I - p^k(b)1^T \pm 11T/n)\xi_k^*(b).$$

This implies

$$\xi_k^*(b + 1) + (1/n - p^k(b))1^T\xi_k^*(b + 1) \geq (W^k(b) - p^k(b)1^T)\left[\xi_k^*(b) + (1/n - p^k(b))1^T\xi_k^*(b)\right],$$

where $\xi_k^*(b) := (I - 11T/n)\xi_k^*(b + 1)$. By the second property in Lemma 2 for all $k \geq \bar{k}$,

$$\|\xi_k^*(b + 1) + (1/n - p^k(b))1^T\xi_k^*(b)\| \leq \|\xi_k^*(b) + (1/n - p^k(b))1^T\xi_k^*(b)\| ,$$

where we have used the fact that $1^T\xi_k^*(b + 1) = 1^T\xi_k^*(b) = 1^T x_k$ since $\xi_k^*(0) = x_k$ and $1^T W^k(b) = 1^T B$. Now, by the third property in Lemma 2 we know that $\|1/n - p^k(b)\|$ can be made arbitrarily small by choosing a large-enough $k$. Hence, for the inequality above to hold for all $k \geq \bar{k}$, there must exist a $\bar{k} \geq \tilde{k}$ such that for all $k \geq \bar{k}$, $\|\xi_k^*(b + 1)\| \leq \delta \|\xi_k^*(b)\|$. Iterating this inequality over $b = 0$ to $b = B - 1$, we have $\|y_k\| \leq \delta^B \|x_\perp\|$ for all $k \geq \bar{k}$ since $\xi_k^*(B) = y_k$ and $\xi_k^*(0) = x_k$. Hence, for all $k \geq 0$,

$$\|y_k\|_{\lambda,K} \leq \delta^B \|x_\perp\|_{\lambda,K} + \tilde{c}_1$$

where $\tilde{c}_1 := \max_{k=0,...,\bar{k}}(\|y_k\|_{\lambda,K}/\lambda)$. Thus, $\gamma_2 = \delta^B$.

Proof of the third arrow ($y_\perp \xrightarrow{\gamma_3} \hat{a} + \rho \Delta y$): To prove the third arrow, we prove two intermediate arrows. First, we show $\hat{a} \xrightarrow{\gamma_3} \Delta y$ for some $c_1 \geq 0$ and then we show $y_\perp \xrightarrow{\gamma_3} \hat{a}$ for some $\gamma \geq 0$. Using these two, the main arrow will follow.

To prove $\hat{a} \xrightarrow{\gamma_3} \Delta y$, note that for all $k \geq 1$, $y_k = \xi_k^*(B) = \prod_{b=0}^{B-1} W^k(b)\xi_k^*(0) = \prod_{b=0}^{B-1} W^k(b)x_k$. Thus, by the fourth property in Lemma 2 for all $k \geq \bar{k}$, $\|y_k\| \leq \delta^B \|x_\perp\|_{\lambda,K} + \tilde{c}_1$.

8
\[ \prod_{b=0}^{B-1} \|W^k(b)\| \|x^k\| \leq MB \|x^k\|. \]

Using the relation above, we have for all \( k \geq 0 \),
\[ \|\Delta y^{k+1}\| = \|y^{k+1} - y^k\| \leq MB\|x^{k+1} - x^k\| \]
\[ \leq MB(\|x^{k+1} - x^k\| + \|x^k - x^k\|) \]
\[ = MB(\|\nabla f^*(a^{k+1} - \rho \Delta y^{k+1}) - \nabla f^*(-a^*)\| \]
\[ + \|\nabla f^*(-a^k - \rho \Delta y^k) - \nabla f^*(-a^*)\|) \]
\[ \leq \frac{MB}{\mu}(\|a^{k+1} - a^* + \rho \Delta y^{k+1}\| + \|a^k - a^* + \rho \Delta y^k\|) \]
\[ \leq \frac{MB}{\mu}(\|\bar{a}^{k+1}\| + \|\bar{a}^k\| + \rho(\|\Delta y^{k+1}\| + \|\Delta y^k\|)) \]
where we have used Lemma 4 and the fact that \( f^* \) is 1/\( \mu \)-smooth. Dividing above inequality by \( \lambda^{k+1} \) and taking suprenum over \( k = 0, \ldots, K - 1 \), we have
\[ \|\Delta y\|_{\lambda,K} \leq \frac{MB}{\mu}(\|\bar{a}\|_{\lambda,K} + \frac{1}{\lambda} |\bar{a}|_{\lambda,K}) \]
\[ + \rho\left(\|\Delta y\|_{\lambda,K} + \frac{1}{\lambda} \|\Delta y\|_{\lambda,K}\right) + \bar{c}_2, \]
where \( \bar{c}_2 := \max_{k=0,\ldots,k} |\|\Delta y\|_{\lambda,k}/\|/\lambda\| \). This implies
\[ \left(\frac{\mu - MB(\lambda + 1)}{\lambda \mu}\right)\|\Delta y\|_{\lambda,K} \leq MB\left(\frac{\lambda + 1}{\lambda \mu}\right)\|\bar{a}\|_{\lambda,K} + \bar{c}_2. \] 

As given in Theorem 1, \( \rho < (\lambda \mu)/(MB(\lambda + 1)) \). This implies \( (\lambda \mu - MB(\lambda + 1))/(\lambda \mu) > 0 \). Thus,
\[ \|\Delta y\|_{\lambda,K} \leq c_1 |\bar{a}|_{\lambda,K} + \bar{c}_3, \tag{24} \]
where \( c_1 := (\lambda + 1)/(\mu - MB(\lambda + 1)) \) and \( \bar{c}_3 := \bar{c}_2 (\lambda \mu/(\mu - \lambda (\lambda + 1))). \)

To prove \( y_{\perp} \rightarrow \tilde{a} \), let \( \Delta a^{k+1} = a^{k+1} - a^k \). For all \( k \geq 0 \),
\[ \|\bar{a}^{k+1}\|^2 = \|a^k - a^*\|^2 = \|a^k - a^{k+1} + a^{k+1} - a^*\|^2 \]
\[ = \|\Delta a^{k+1}\|^2 + \|\bar{a}^{k+1}\|^2 + 2(a^k - a^{k+1})^T(a^{k+1} - a^*). \]

This implies
\[ \|\bar{a}^{k+1}\|^2 \leq \|a^k - a^*\|^2 + 2(a^k - a^{k+1})^T(a^{k+1} - a^*). \tag{25} \]

We will try to bound the term \( 2(a^k - a^{k+1})^T(a^{k+1} - a^*) \) above. The update step in Algorithm 1 can be written as
\[ a^{k+1} = \arg\min_{a:(11^T/n)a = 0} \rho(y^{k+1} - x^{k+1})^Ta + \frac{1}{2}\|a - a^k\|^2 \]
\[ = \arg\min_{a:(11^T/n)a = 0} \rho(y^{k+1} - x^{k+1})^Ta + \frac{1}{2}\|a - a^k\|^2, \]

From Lemma 3, we know that \( (11^T/n)a^* = 0 \). Thus, we can evaluate the optimality condition for above minimization problem at \( a = a^* \) to obtain
\[ \rho(y^{k+1} - x^k + a^{k+1} - a^k)^T(a^* - a^{k+1}) \geq 0, \]
which implies
\[ (a^{k+1} - a^*)^T(a^{k+1} - a^*) \leq \rho(\nabla f^*(-a^{k+1} - \rho \Delta y^{k+1}) - y^{k+1})^T(a^{k+1} - a^*) \tag{26} \]

using Lemma 4. For the ease of notation, let \( \tilde{a} = -a^{k+1} - \rho \Delta y^{k+1} \). Now, we bound the term \( \rho(\nabla f^*(-\tilde{a}) - y^{k+1})^T(a^{k+1} - a^*) \). Since \( f^* \) is 1/\( L \)-strongly convex, we have, for all \( a_1 \) in the domain of \( f^* \), \( f^*(a_1) \geq f^*(a) + \nabla f^*(a)(a_1 - a) + (1/2L)\|a_1 - a\|^2 \) which implies
\[ \nabla f^*(\tilde{a}) \leq f^*(a_1) - f^*(a) - (y^{k+1})^T(a_1 - a) \leq \frac{\|a_1 - a\|^2}{2L} \]
\[ \leq \frac{\|a_1 - a\|^2}{2\alpha} + \frac{\|a_1 - a\|^2}{2\lambda}, \]
where the last inequality follows by using Peter-Paul inequality on \( (y^{k+1})^T(a_1 - a) \), where \( \alpha > 0 \) is arbitrary. Similarly, using the fact that \( f^* \) is 1/\( \mu \)-smooth, we have, for all \( a_2 \) in the domain of \( f^* \),
\[ \nabla f^*(\tilde{a}) \leq \frac{\|a_1 - a\|^2}{2\alpha} + \frac{\|a_1 - a\|^2}{2\lambda}, \]
where \( \beta > 0 \) is arbitrary. Adding above two inequalities with \( a_1 = -a^*, a_2 = -a^{k+1} \), we have
\[ \nabla f^*(\tilde{a}) \leq \frac{\|a^*\|^2}{2\alpha} + \frac{\|a^{k+1}\|^2}{2\lambda} + \rho(\frac{1}{\alpha} + \frac{1}{L})\|\Delta y^{k+1}\|^2. \tag{27} \]

Choose \( \alpha = L \). Further, since \( f^* \) is 1/\( L \)-strongly convex,
\[ f^*(-a^{k+1}) \geq f^*(-a^*) + \nabla f^*(-a^*)^T(-a^{k+1} + a^*) \]
\[ + \frac{\|a^{k+1} - a^*\|^2}{2L} \]
where \( \nabla f^*(-a^*)^T(-a^{k+1} + a^*) = 0 \), since \( \nabla f^*(-a^*) = x^* = (11^T/n)a^* \) and \( (11^T/n)a^{k+1} = (11^T/n)a^* = 0 \) (using Remark 3, Lemma 5 and Lemma 4). Thus,
\[ f^*(-a^*) - f^*(-a^{k+1}) \leq -\rho \|\tilde{a}^{k+1}\|^2. \]

We substitute above inequality in (27). Then, dividing by \( (\lambda K)^2 \) and taking supremum over \( k = 0, \ldots, K - 1 \), we have
\[ \left(1 + \frac{\rho}{L} - \frac{1}{\lambda K}\right)\|\tilde{a}\|_{\lambda,K}^2 \leq \rho(L + \beta)(\|y\|_{\lambda,K}^2) + \rho^3\left(\frac{1}{\beta} + \frac{1}{\mu}\right)\|\Delta y\|_{\lambda,K}^2 + \|\tilde{a}\| \]
for all \( K \geq 0 \). Now, using (24), we have
\[ \left(1 + \frac{\rho}{L} - \frac{1}{\lambda K} - \rho^3\left(\frac{1}{\beta} + \frac{1}{\mu}\right)\right)\|\tilde{a}\|_{\lambda,K}^2 \]
\[ \leq \rho(L + \beta)(\|y\|_{\lambda,K}^2 + \bar{c}_4), \tag{28} \]
where $c_3 := \|\tilde{a}\|^2 + \rho^3(1/\beta + 1/\mu)\bar{c}_3$. Now, it can be verified that $c_2 > 0$ given

$$2L\left(\frac{1}{\lambda^2} - 1\right) < \rho < \frac{c_3 \lambda \mu}{(c_3 + 1)(\lambda + 1)},$$

(29)
as given in Theorem 1 where $c_3 = 1/\sqrt{2L(1/\beta + 1/\mu)}$. Thus, (28) implies

$$\|\tilde{a}\|^2 \leq \sqrt{\frac{\rho(L + \beta)}{c_2}} \|y\|^2 \|\lambda\|^2 + \bar{c}_5$$

(30)
for some constant $\bar{c}_5 \geq 0$ and for all $K \geq 0$.

Using inequalities (24) and (30), we have

$$\|\tilde{a} + \rho \Delta y\|^2 \leq (1 + \rho c_1) \sqrt{\frac{\rho(L + \beta)}{c_2}} \|y\|^2 \|\lambda\|^2 + \bar{c}_6$$

for some constant $\bar{c}_6 \geq 0$. Thus, we have proved the third arrow in (17) with $\gamma_3 = (1 + \rho c_1) \sqrt{\rho(L + \beta)/c_2}$.

E. Choice of $\lambda$, $\rho$ and $B$ which ensures $\gamma_1 \gamma_2 \gamma_3 < 1$

First, we choose an arbitrary $\beta > 0$ and a desired $\lambda \in (0, 1)$ satisfying (13). Then, we choose a $\rho$ satisfying (14).

Note that $\gamma_2 = \delta^B$ where $\delta < 1$, while $\gamma_1$ and $\gamma_3$ are independent of $B$. Hence, $B \geq \max\{1, [\log(\gamma_1 \gamma_3)/\log(1/\delta)]\}$ ensures $\gamma_1 \gamma_2 \gamma_3 < 1$.

F. Proof of the additional inequalities in (18)

From (30), we have

$$\|\tilde{a}\|^2 \leq \sqrt{\frac{\rho(L + \beta)}{c_2}} \|y\|^2 \|\lambda\|^2 + \bar{c}_6$$

and from (23) we have

$$\|\tilde{x}\|^2 \leq (1/\mu) \|\tilde{a}\|^2 + \rho \Delta y \|y\|^2 + \|\tilde{x}\|^2$$.

### References

[1] A. Nedić and J. Liu, “Distributed optimization for control,” Annual Review of Control, Robotics, and Autonomous Systems, vol. 1, no. 1, pp. 77–103, 2018. [Online]. Available: https://doi.org/10.1146/annurev-control-060117-105131

[2] A. Nedić and A. Olshevsky, “Distributed optimization over time-varying directed graphs,” IEEE Transactions on Automatic Control, vol. 60, no. 3, pp. 601–615, 2015.

[3] S. Pu, W. Shi, J. Xu, and A. Nedic, “Push-pull gradient methods for distributed optimization in networks,” IEEE Transactions on Automatic Control, pp. 1–20, 2020.

[4] A. Nedić, A. Olshevsky, and W. Shi, “Achieving geometric convergence for distributed optimization over time-varying graphs,” SIAM Journal on Optimization, vol. 27, no. 4, pp. 2597–2633, 2017. [Online]. Available: https://doi.org/10.1137/16M1084316

[5] A. Makhdoumi and A. Ozdaglar, “Graph balancing for distributed subgradient methods over directed graphs,” in 2015 54th IEEE Conference on Decision and Control (CDC), 2015, pp. 1364–1371.

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

[7] A. Makhdoumi and A. Ozdaglar, “Convergence rate of distributed ADMM over networks,” IEEE Transactions on Automatic Control, vol. 62, no. 10, pp. 5082–5095, 2017.

[8] M. Maros and J. Jaldén, “PANDA: A dual linearly converging method for distributed optimization over time-varying undirected graphs,” in 2018 IEEE Conference on Decision and Control (CDC), 2018, pp. 6520–6525.

[9] V. Khatana and M. V. Salapaka, “DC-DistADMM: ADMM algorithm for constrained distributed optimization over directed graphs,” 2020.

[10] 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, p. 1–122, Jan. 2011. [Online]. Available: https://doi.org/10.1561/2200000016

[11] D. Gabay and B. Mercier, “A dual algorithm for the solution of nonlinear variational problems via finite element approximation,” Computers & Mathematics with Applications, vol. 2, no. 1, pp. 17 – 40, 1976. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0898122176900341

[12] M. Mardani, G. Mateos, and G. B. Giannakis, “Decentralized sparsity-regularized rank minimization: Algorithms and applications,” IEEE Transactions on Signal Processing, vol. 61, no. 21, pp. 5374–5388, Nov 2013.

[13] T. Erseghe, “Distributed optimal power flow using ADMM,” IEEE Transactions on Power Systems, vol. 29, no. 5, pp. 2370–2380, 2014.

[14] A. Zare, M. R. Jovanović, and T. T. Georgiou, “Alternating direction optimization algorithms for covariance completion problems,” in 2015 American Control Conference (ACC), July 2015, pp. 515–520.

[15] M. Zhu and S. Martínez, “Discrete-time dynamic average consensus,” Automatica, vol. 46, no. 2, pp. 322 – 329, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0005109809004828

[16] K. O. Saber and R. M. Murray, “Consensus protocols for networks of dynamic agents,” in Proceedings of the 2003 American Control Conference, 2003, vol. 2, pp. 951–956.

[17] L. Xiao and S. Boyd, “Fast linear iterations for distributed averaging,” Systems & Control Letters, vol. 53, no. 1, pp. 65 – 78, 2004. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167691104000398

[18] K. I. Tsianos, S. Lawlor, and M. G. Rabbat, “Push-sum distributed dual averaging for convex optimization,” in 2012 IEEE 51st IEEE Conference on Decision and Control (CDC), 2012, pp. 5453–5458.

[19] S. S. Kia, B. Van Scoy, J. Cortes, R. A. Freeman, K. M. Lynch, and S. Martinez, “Tutoral on dynamic average consensus: The problem, its applications, and the algorithms,” IEEE Control Systems Magazine, vol. 39, no. 3, pp. 40–72, 2019.

[20] R. Xin and U. A. Khan, “Distributed heavy-ball: A generalization and acceleration of first-order methods with gradient tracking,” IEEE Transactions on Automatic Control, vol. 65, no. 6, pp. 2627–2633, 2020.

[21] C. D. Meyer, Matrix Analysis and Applied Linear Algebra. USA: Society for Industrial and Applied Mathematics, 2000.

[22] S. Kakade, S. Shalev-Shwartz, and A. Tewari, “On the duality of strong convexity and strong smoothness: Learning applications and matrix regularization,” Unpublished Manuscript, 2009. [Online]. Available: http://ttic.uchicago.edu/shai/papers/KakadeShalevTewari09.pdf