Accelerated Alternating Direction Method of Multipliers Gradient Tracking for Distributed Optimization

Eduardo Sebastián, Mauro Franceschelli, Andrea Gasparri, Eduardo Montijano and Carlos Sagüés

Abstract— **This paper presents a novel accelerated distributed algorithm for unconstrained consensus optimization over static undirected networks. The proposed algorithm combines the benefits of acceleration from momentum, the robustness of the alternating direction method of multipliers, and the computational efficiency of gradient tracking to surpass existing state-of-the-art methods in convergence speed, while preserving their computational and communication cost. First, we prove that, by applying momentum on the average dynamic consensus protocol over the estimates and gradient, we can study the algorithm as an interconnection of two singularly perturbed systems: the outer system connects the consensus variables and the optimization variables, and the inner system connects the estimates of the optimum and the auxiliary optimization variables. Next, we prove that, by adding momentum to the auxiliary dynamics, our algorithm always achieves faster convergence than the achievable linear convergence rate for the non-accelerated alternating direction method of multipliers gradient tracking algorithm case. Through simulations, we numerically show that our accelerated algorithm surpasses the existing accelerated and non-accelerated distributed consensus first-order optimization protocols in convergence speed.**

Index Terms— **Optimization algorithms, distributed control, cooperative control**

I. INTRODUCTION

DISTRIBUTED optimization refers to the problem of finding the global optimum of an optimization problem where the cost function, the constraints or the available information is distributed over a network [1]. Of particular interest is the case of unconstrained consensus optimization [2], where the global cost function is the sum of local cost functions. Practical instances of such a setting are federated learning [3], networked games [4] or multi-robot control [5]. In all

E. Sebastián, E. Montijano and C. Sagüés are with the Instituto de Investigación en Ingeniería de Aragón, Universidad de Zaragoza, Spain (email:esebastian@unizar.es, emonti@unizar.es, csagues@unizar.es). M. Franceschelli is with the Department of Electrical and Electronic Engineering, University of Cagliari, Italy (email: mauro.franceschelli@unica.it). A. Gasparri is with the Department of Engineering, Roma Tre University, Italy (email:andrea.gasparri@uniroma3.it).

these applications, it is of key importance to reconstruct the global optimum as fast as possible to reduce the number of gradient computations, the communication efforts and adapt to evolving environments. To cope with this, we propose a novel accelerated distributed optimization algorithm that exploits momentum to outperform existing first order approaches in convergence speed, while leveraging the robustness of the alternating direction method of multipliers (ADMM) and the computational simplicity of gradient tracking (GT).

The evolution of distributed optimization algorithms can be divided in three stages. The first approaches departed from subgradient [6], gradient [7] and proximal methods [8] to derive distributed versions that achieve convergence to the optimum if and only if the step size diminishes with time. This property poses a fundamental trade-off between speed and accuracy because fixed step sizes lead to convergence within a given distance from the optimum.

To overcome the speed-accuracy trade-off, several works capitalized from different points of view the properties of dynamic average consensus [9], [10] and splitting methods to develop GT [11]–[14] and ADMM or primal/dual methods [15]–[17] for distributed optimization. In both cases, linear convergence is achieved for constant step sizes; however, these approaches suffer from their respective caveats. GT is simple to compute but is inherently non-robust against non-ideal affections such as initialization errors, gradient noise or communication disturbances [18]. ADMM is robust against those undesirable effects but, as a proximal method, it has a non-negligible computational cost at each iteration [16]. In both cases, due to the importance of convergence speed, different works have exploited acceleration methods to improve convergence rate such as momentum [19], [20], Nesterov acceleration [21], Anderson acceleration and adaptive preconditioning for proximal-based optimization such as ADMM and Douglas-Rachford Splitting [22], [23] or smooth/non-smooth regularizers [24]. Many of the aforementioned methods work for time-varying and/or directed graphs, whereas in this paper we focus on the static undirected case.

Recently, a distributed optimization protocol that combines ADMM and GT [25], [26] was proposed as an alternative to combine the benefits of both approaches. The key idea is that GT can be reformulated in ADMM form, where the ADMM cost function is quadratic in the optimization variables. Therefore, it is possible to obtain a closed analytical solution that is cheap to compute and robust. Nevertheless,

despite proving linear convergence for strongly convex cost functions, the algorithm needs time-scale decoupling between the optimum estimate and auxiliary dynamics, limiting the achievable convergence speed.

To overcome such limitation, **our main contribution** (Section III) is a novel accelerated distributed optimization method for static undirected networks based on ADMM and GT which exploits momentum (for details about the concept of momentum, see [27]). Momentum allows to accelerate the convergence of tracking average consensus variables (e.g., [10]) over the estimates and gradients, and speed up the convergence speed of the auxiliary variables' dynamics. As a consequence, we can study the convergence properties of our protocol and show that it always achieves linear convergence to the global optimum. More importantly, we also prove that, by adding momentum, acceleration is guaranteed compared to the same algorithm without momentum. We empirically evaluate our proposed algorithm (Section IV) against existing state-of-the-art algorithms, showing that ours surpasses all of them in convergence speed, using the same computation-communication resources while increasing the memory burden.

II. PRELIMINARIES

A. Problem Formulation

A network of $\mathsf{N} > 0$ nodes[1] cooperates to find the global optimum $\mathbf{x}^* \in \mathbb{R}^n$ of the consensus optimization problem

$$\mathbf{x}^* = \arg \min_{\mathbf{x} \in \mathbb{R}^n} \sum_{i=1}^{\mathsf{N}} f_i(\mathbf{x}), \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the $n-$dimensional decision variable. Each node $i \in \mathcal{V} = \{1, \ldots, \mathsf{N}\}$ only has access to its local objective function $f_i(\bullet)$ and the local estimate of global optimum $\mathbf{x}_i^t$, with $t \in \mathbb{N}$ denoting the discrete time steps. Problem (1) is reformulated in distributed form by including the consensus constraint as follows

$$\mathbf{x}^* = \arg \min_{\mathbf{x}_i \in \mathbb{R}^n} \sum_{i=1}^{\mathsf{N}} f_i(\mathbf{x}_i) \quad \text{s.t. } \mathbf{x}_i = \mathbf{x}_j \quad \forall i, j \in \mathcal{V}. \tag{2}$$

We assume that $f_i(\bullet)$ is $\mathbf{c}$-strongly convex and the gradients $\nabla f_i(\mathbf{x})$ are L-Lipschitz continuous $\forall i \in \mathcal{V}$. This assumption guarantees that $\mathbf{x}^*$ is unique.

The network of nodes is described by an undirected connected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges of the graph, such that $(i, j) \in \mathcal{E}$ means that nodes i and j can communicate with each other. The set of neighbors of node i is $\mathcal{N}_i = \{j | (i,j) \in \mathcal{E}\}$. Note that $i \notin \mathcal{N}_i$. The degree of node i is $d_i = \mathsf{car}(\mathcal{N}_i)$ and $d = \sum_{i=1}^{\mathsf{N}} d_i$.

The goal of this work is to develop a distributed algorithm such that each local estimate $\mathbf{x}_i^t$ linearly converges to the global optimum $\mathbf{x}^*$, respecting the topology structure of $\mathcal{G}$.

[1]**Notation.** Let $\mathbb{N}$ and $\mathbb{R}$ be the natural and real number sets. We use car$(\bullet)$ to denote the cardinality of a set. We define as $\mathbf{I}_p, \mathbf{0}_{p \times k}, \mathbf{1}_{p \times k}$ the identity matrix of dimension p, the $p \times k$ matrix of zeros and the $p \times k$ matrix of ones respectively. Besides, $\mathbf{1}_{N,p} = \mathbf{1}_N \otimes \mathbf{I}_p$ where $\otimes$ denotes the Kronecker product. Let $\sigma_i(\bullet)$ be the i-th eigenvalue of a matrix and $\det(\bullet)$ its determinant. Let $|| \bullet ||$ be the L2-norm. We denote by diag$(\{\mathbf{E}_i\}_{i=1}^N)$ the block diagonal matrix whose $i-$th block element matrix is $\mathbf{E}_i$.

B. Singularly Perturbed Systems

Some of the mathematical derivations exposed in this work are based on singularly perturbed systems theory [28]. Singularly perturbed systems are dynamical systems composed by two interconnected dynamics, each of them evolving at different time-scales. Typically, the slow dynamics corresponds to the desired states to be controlled (e.g., the estimates $\mathbf{x}_i^t$) while the fast dynamics corresponds to auxiliary variables. To ensure stability, the speed of the slow system is constrained by the speed of the fast dynamics. For clarity, we reproduce a theorem that establishes that, for a sufficiently slow dynamics, the slow system (3a) is decoupled from the fast system (3b) and they are both globally exponentially stable. We will use this fact to prove convergence properties of our algorithm.

Theorem 1 (Theorem II.3, [26]): Let

$$\mathbf{p}^{t+1} = \mathbf{p}^t + \gamma f(\mathbf{p}^t, \mathbf{q}^t, t) \tag{3a}$$
$$\mathbf{q}^{t+1} = g(\mathbf{q}^t, \mathbf{p}^t, t), \tag{3b}$$

with $\mathbf{p} \in \mathbb{R}^n$, $\mathbf{q} \in \mathbb{R}^m$, $\gamma > 0$ and $f(\bullet), g(\bullet)$ Lipschitz continuous uniformly over t. Let assume that there exists a function $\mathbf{q}_{\mathsf{eq}}(\mathbf{p})$ that is Lipschitz continuous such that

$$\mathbf{0}_n = \gamma f(\mathbf{0}_n, \mathbf{q}_{\mathsf{eq}}(\mathbf{0}_n), t) \tag{4a}$$
$$\mathbf{q}_{\mathsf{eq}}(\mathbf{p}) = g(\mathbf{q}_{\mathsf{eq}}(\mathbf{p}), \mathbf{p}, t). \tag{4b}$$

We call reduced system the system given by

$$\mathbf{p}^{t+1} = \mathbf{p}^t + \gamma f(\mathbf{p}^t, \mathbf{q}_{\mathsf{eq}}(\mathbf{p}^t), t) \tag{5}$$

and boundary system layer with $\tilde{\mathbf{q}} \in \mathbb{R}^m$ the system

$$\tilde{\mathbf{q}}^{t+1} = g(\tilde{\mathbf{q}}^t + \mathbf{q}_{\mathsf{eq}}(\mathbf{p}^t), \mathbf{p}^t, t) - \mathbf{q}_{\mathsf{eq}}(\mathbf{p}^t). \tag{6}$$

Assume that there exist continuous function $u(\bullet)$ and gain $\bar{\gamma}_1 > 0$ such that, for any $\gamma \in (0, \bar{\gamma}_1)$, there exist $b_1, b_2, b_3, b_4 > 0$ and values $\tilde{\mathbf{q}}, \tilde{\mathbf{q}}_1, \tilde{\mathbf{q}}_2 \in \mathbb{R}^m$ such that

$$b_1 ||\tilde{\mathbf{q}}||^2 \le u(\tilde{\mathbf{q}}, t) \le b_2 ||\tilde{\mathbf{q}}||^2 \tag{7a}$$
$$u(\tilde{\mathbf{q}}^{t+1}, t+1) - u(\tilde{\mathbf{q}}^t, t) \le -b_3 ||\tilde{\mathbf{q}}||^2 \tag{7b}$$
$$|u(\tilde{\mathbf{q}}_1, t) - u(\tilde{\mathbf{q}}_2, t)| \le b_4 ||\tilde{\mathbf{q}}_1 - \tilde{\mathbf{q}}_2||(||\tilde{\mathbf{q}}_1|| + ||\tilde{\mathbf{q}}_2||). \tag{7c}$$

Also, assume that there exists a continuous function $w(\bullet)$, and gain $\bar{\gamma}_2 > 0$ such that, for any $\gamma \in (0, \bar{\gamma}_2)$, there exist $c_1, c_2, c_3, c_4 > 0$ and values $\mathbf{p}, \mathbf{p}_1, \mathbf{p}_2 \in \mathbb{R}^n$ such that

$$c_1 ||\mathbf{p}||^2 \le w(\mathbf{p}, t) \le c_2 ||\mathbf{p}||^2 \tag{8a}$$
$$w(\mathbf{p}^{t+1}, t+1) - w(\mathbf{p}^t, t) \le -c_3 ||\mathbf{p}||^2 \tag{8b}$$
$$|w(\mathbf{p}_1, t) - w(\mathbf{p}_2, t)| \le c_4 ||\mathbf{p}_1 - \mathbf{p}_2||(||\mathbf{p}_1|| + ||\mathbf{p}_2||). \tag{8c}$$

Then, there exists $\bar{\gamma} \in (0, \min(\bar{\gamma}_1, \bar{\gamma}_2))$ and gains $\kappa_1, \kappa_2 > 0$ such that, for any $\gamma \in (0, \bar{\gamma})$ and $(\mathbf{p}^0, \mathbf{q}^0)$, it holds that

$$\left\| \begin{pmatrix} \mathbf{p}^t - \mathbf{p}^* \\ \mathbf{q}^t - \mathbf{q}_{\mathsf{eq}}(\mathbf{p}^t) \end{pmatrix} \right\| \le \kappa_1 \left\| \begin{pmatrix} \mathbf{p}^0 \\ \mathbf{q}^0 - \mathbf{q}_{\mathsf{eq}}(\mathbf{p}^0) \end{pmatrix} \right\| e^{-\kappa_2 t}, \tag{9}$$

where $\mathbf{p}^* \in \mathbb{R}^n$ is the desired equilibrium of $\mathbf{p}$.
Conditions (7a), (8a), (7b), (8b) come from standard Lyapunov stability arguments, whereas (7c) and (8c) bound the changes in the Lyapunov functions $w(\mathbf{p}, t)$ and $u(\tilde{\mathbf{q}}, t)$ to apply Lyapunov exponential stability arguments.

C. ADMM Gradient Tracking

As introduced in Section I, a recent paper [26] proposes a first order distributed optimization algorithm that combines ADMM and GT to obtain a robust linearly convergent algorithm. Algorithm 1 presents such scheme. Each node i updates

---

Algorithm 1 Original [26] ADMM-GT at node i

1: **State of agent**: $\mathbf{x}_i^0 \in \mathbb{R}^n$, $z_{ij}^0 \in \mathbb{R}^{2n}$ $\quad \forall j \in \mathcal{N}_i$
2: **Parameters**: $\gamma > 0$, $\rho > 0$, $\alpha \in (0,1)$
3: **for** $t = 1, \dots$ **do**
4: $\quad \mathbf{y}_i^{t+1} = \frac{1}{1+\rho d_i}(\mathbf{x}_i^t + \begin{pmatrix} \mathbf{I}_n & \mathbf{0}_{n \times n} \end{pmatrix} \sum_{j \in \mathcal{N}_i} \mathbf{z}_{ij}^t)$
5: $\quad \mathbf{s}_i^{t+1} = \frac{1}{1+\rho d_i}(\nabla f_i(\mathbf{x}_i^t) + \begin{pmatrix} \mathbf{0}_{n \times n} & \mathbf{I}_n \end{pmatrix} \sum_{j \in \mathcal{N}_i} \mathbf{z}_{ij}^t)$
6: $\quad \mathbf{z}_{ij}^{t+1} = (1-\alpha)\mathbf{z}_{ij}^t - \alpha(\mathbf{z}_{ji}^t - 2\rho\begin{pmatrix} \mathbf{y}_i^{t+1} \\ \mathbf{s}_i^{t+1} \end{pmatrix})$ $\quad \forall j \in \mathcal{N}_i$
7: $\quad \mathbf{x}_i^{t+1} = \mathbf{x}_i^t + \gamma(\mathbf{y}_i^{t+1} - \mathbf{x}_i^t) - \gamma\mathbf{s}_i^{t+1}$
8: **end for**

---

four variables. First, $\mathbf{x}_i^t$ is the variable that estimates the global optimum $\mathbf{x}^*$. The variable $\mathbf{y}_i^t$ reconstructs the average of the estimates over the nodes of the network $\frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i^t$ and aims at eliminating the consensus error. The variable $\mathbf{s}_i^t$ reconstructs the average of the gradients $\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{x}_i^t)$ and aims at reducing the error $\mathbf{e}_i^t = \|\mathbf{x}_i^t - \mathbf{x}^*\|$. Finally, $\mathbf{z}_{ij}^t \in \mathbb{R}^{2n}$ are auxiliary variables that enforce consensus over $\mathbf{y}_i^t$ and $\mathbf{s}_i^t$ [16].

One fundamental aspect of Algorithm 1 is that, by conveniently rewriting the dynamics of $\mathbf{x}^t = [(\mathbf{x}_1^t)^\top, \dots, (\mathbf{x}_N^t)^\top]^\top$ and $\mathbf{z}^t = [(\mathbf{z}_1^t)^\top, \dots, (\mathbf{z}_N^t)^\top]^\top$ (with $\mathbf{z}_i^t = [(\mathbf{z}_{i1}^t)^\top, \dots, (\mathbf{z}_{id_i}^t)^\top]^\top$), it can be shown that the overall dynamics of the scheme have the structure of a time-varying singularly perturbed system [25]. Therefore, relying on Theorem 1, for a sufficiently small γ, Algorithm 1 converges linearly to the global optimum of (1).

In the following section, we present an algorithm that, by adding momentum to the dynamics of $\mathbf{y}_i^t$, $\mathbf{s}_i^t$ and $\mathbf{z}_{ij}^t$, speeds up the convergence while preserving the computation and communication properties of Algorithm 1.

III. ACCELERATED ADMM GRADIENT TRACKING

The proposed accelerated distributed optimization scheme is presented in Algorithm 2. Algorithm 2 adds momentum to the dynamics of $\mathbf{y}_i^t$ and $\mathbf{s}_i^t$ (lines 6 and 7) and the dynamics of $\mathbf{z}_{ij}^t$ (line 9). The former allows to write Algorithm 2 as a chain of two singularly perturbed systems. The outer singularly perturbed system interconnects the dynamics given by $\mathbf{y}_i^t$ and $\mathbf{s}_i^t$ with the dynamics given by $\mathbf{x}_i^t$ and $\mathbf{z}_i^t$. The inner singularly perturbed system interconnects the dynamics given by $\mathbf{z}_i^t$ with the dynamics given by $\mathbf{x}_i^t$. In terms of convergence speed, there are three time-scales, from the fastest to the slowest: (1) $\mathbf{y}_i^t, \bar{\mathbf{y}}_i^t, \mathbf{s}_i^t, \bar{\mathbf{s}}_i^t$; (2) $\mathbf{z}_i^t, \bar{\mathbf{z}}_i^t$; and (3) $\mathbf{x}_i^t$. This time-scale separation leads to a novel convergence analysis of the ADMM-GT algorithm, discovering a limitation in the maximum achievable convergence rate of the dynamics of $\mathbf{z}_i^t$. By adding momentum to $\mathbf{z}_i^t$ this bound is overcome, enhancing the overall convergence rate when compared to Algorithm 1. Additionally, by respecting the ADMM gradient tracking structure of Algorithm 1, Algorithm 2 preserves the robustness

---

Algorithm 2 A2DMM-GT at node i

1: **State of agent**: $\mathbf{x}_i^0 \in \mathbb{R}^n$, $\mathbf{z}_{ij}^0 \in \mathbb{R}^{2n}$ $\quad \forall j \in \mathcal{N}_i$, $\mathbf{z}_{ij}^{-1} \in \mathbb{R}^{2n}$ $\quad \forall j \in \mathcal{N}_i$, $\mathbf{y}_i^0 \in \mathbb{R}^n$, $\mathbf{s}_i^0 \in \mathbb{R}^n$
2: **Parameters**: $\gamma > 0$, $\rho > 0$, $\alpha \in (0,1)$, $\lambda \in (1,2)$, $\mu \in (1,2)$, $\epsilon \in (0,1)$
3: **for** $t = 1, \dots$ **do**
4: $\quad \bar{\mathbf{y}}_i^{t+1} = \frac{1}{1+\rho d_i}(\mathbf{x}_i^t + \begin{pmatrix} \mathbf{I}_n & \mathbf{0}_{n \times n} \end{pmatrix} \sum_{j \in \mathcal{N}_i} \mathbf{z}_{ij}^t)$
5: $\quad \bar{\mathbf{s}}_i^{t+1} = \frac{1}{1+\rho d_i}(\nabla f_i(\mathbf{x}_i^t) + \begin{pmatrix} \mathbf{0}_{n \times n} & \mathbf{I}_n \end{pmatrix} \sum_{j \in \mathcal{N}_i} \mathbf{z}_{ij}^t)$
6: $\quad \mathbf{y}_i^{t+1} = \mathbf{y}_i^t + \lambda(\bar{\mathbf{y}}_i^{t+1} - \mathbf{y}_i^t)$
7: $\quad \mathbf{s}_i^{t+1} = \mathbf{s}_i^t + \lambda(\bar{\mathbf{s}}_i^{t+1} - \mathbf{s}_i^t)$
8: $\quad \bar{\mathbf{z}}_{ij}^{t+1} = (1-\alpha)\mathbf{z}_{ij}^t - \alpha(\mathbf{z}_{ji}^t - 2\rho\begin{pmatrix} \mathbf{y}_i^{t+1} \\ \mathbf{s}_i^{t+1} \end{pmatrix})$ $\quad \forall j \in \mathcal{N}_i$
9: $\quad \mathbf{z}_{ij}^{t+1} = \mu(\mathbf{z}_{ij}^t + \epsilon(\bar{\mathbf{z}}_{ij}^{t+1} - \mathbf{z}_{ij}^t)) + (1-\mu)\mathbf{z}_{ij}^{t-1}$ $\quad \forall j \in \mathcal{N}_i$
10: $\quad \mathbf{x}_i^{t+1} = \mathbf{x}_i^t + \gamma(\mathbf{y}_i^{t+1} - \mathbf{x}_i^t) - \gamma\mathbf{s}_i^{t+1}$
11: **end for**

---

from ADMM proved in [26] and the computational simplicity of gradient tracking methods (the momentum operations are inexpensive to compute). Comparing Algorithms 1 and 2, their communication cost is the same since only the $\mathbf{z}_{ij}^t$ are exchanged. In terms of memory, each node has to additionally store $\mathbf{z}_{ij}^{t-1}$, $\mathbf{y}_i^t$, $\mathbf{s}_i^t$.

To analyze the convergence properties of Algorithm 2, we define the following matrices:

$$\mathbf{A}_x = \text{diag}\left(\left\{\begin{pmatrix} \mathbf{1}_{d_i,n} \\ \mathbf{0}_{d_i,n} \end{pmatrix}\right\}_{i=1}^{N}\right), \mathbf{A}_z = \text{diag}\left(\left\{\begin{pmatrix} \mathbf{0}_{d_i,n} \\ \mathbf{1}_{d_i,n} \end{pmatrix}\right\}_{i=1}^{N}\right)$$

$$\mathbf{A} = \text{diag}\left(\{\mathbf{1}_{d_i,2n}\}_{i=1}^{N}\right), \mathbf{H} = \text{diag}\left(\left\{\frac{\mathbf{I}_n}{1+\rho d_i}\right\}_{i=1}^{N}\right).$$

Besides, $\mathbf{P} \in \{0,1\}^{2nd \times 2nd}$ is a permutation matrix that exchanges auxiliary variable $\mathbf{z}_{ij}^t$ with auxiliary variable $\mathbf{z}_{ji}^t$ and implements the pairwise message passing found in line 8 of Algorithm 2, $\mathbf{x}^t = [(\mathbf{x}_1^t)^\top, \dots, (\mathbf{x}_N^t)^\top]^\top$, $\mathbf{z}^t = [(\mathbf{z}_1^t)^\top, \dots, (\mathbf{z}_N^t)^\top]^\top$, $\mathbf{z}_i^t = [(\mathbf{z}_{i1}^t)^\top, \dots, (\mathbf{z}_{id_i}^t)^\top]^\top$, $\mathbf{y}^t = [(\mathbf{y}_1^t)^\top, \dots, (\mathbf{y}_N^t)^\top]^\top$ and $\mathbf{s}^t = [(\mathbf{s}_1^t)^\top, \dots, (\mathbf{s}_N^t)^\top]^\top$. Finally, let $\mathbf{g}(\mathbf{x}^t) = [(\nabla f_1(\mathbf{x}_1^t))^\top, \dots, (\nabla f_N(\mathbf{x}_N^t))^\top]^\top$ and $\mathbf{v}(\mathbf{y}^t, \mathbf{s}^t) = [(\mathbf{y}_1^t)^\top, (\mathbf{s}_1^t)^\top, \dots, (\mathbf{y}_N^t)^\top, (\mathbf{s}_N^t)^\top]^\top$.

Exploiting these definitions, Algorithm 2 for the whole network can be written as

$$\mathbf{y}^{t+1} = (1-\lambda)\mathbf{y}^t + \lambda\mathbf{H}(\mathbf{x}^t + \mathbf{A}_x^\top \mathbf{z}^t), \tag{10a}$$

$$\mathbf{s}^{t+1} = (1-\lambda)\mathbf{s}^t + \lambda\mathbf{H}(\mathbf{g}(\mathbf{x}^t) + \mathbf{A}_z^\top \mathbf{z}^t), \tag{10b}$$

$$\mathbf{z}^{t+1} = \mu\mathbf{F}\mathbf{z}^t + (1-\mu)\mathbf{z}^{t-1} + 2\mu\epsilon\alpha\rho\mathbf{P}\mathbf{A}\mathbf{v}(\mathbf{y}^{t+1}, \mathbf{s}^{t+1}), \tag{10c}$$

$$\mathbf{x}^{t+1} = \mathbf{x}^t + \gamma(\mathbf{y}^{t+1} - \mathbf{x}^t) - \gamma\mathbf{s}^{t+1} \tag{10d}$$

where $\mathbf{F} = (1-\epsilon\alpha)\mathbf{I}_{2nd} - \epsilon\alpha\mathbf{P}$. The compact expression in (10) can be reformulated as an interconnection of two singularly perturbed systems. In Fig. 1 we represent the interconnections between the dynamics of $\mathbf{x}^t$, $\mathbf{z}^t$, $\mathbf{y}^t$ and $\mathbf{s}^t$. The inner system (green) connects the dynamics of $\mathbf{x}^t$ (slow) and the dynamics of $\mathbf{z}^t$ (fast). Indeed, if $\mu = \epsilon = 1$, the inner system corresponds to the singularly perturbed system studied in [26]. On the other hand, the outer system is also a singularly perturbed system given by $\mathbf{r}^t = [(\mathbf{y}^t)^\top, (\mathbf{s}^t)^\top]^\top$,

$\mathbf{w}^t = [(\mathbf{x}^t)^\top, (\mathbf{z}^t)^\top, (\mathbf{z}^{t-1})^\top]^\top$, $\mathbf{v}(\mathbf{r}^t) = \mathbf{v}(\mathbf{y}^t, \mathbf{s}^t)$, and

$$\mathbf{r}^{t+1} = (1-\lambda)\mathbf{r}^t + \lambda \begin{pmatrix} \mathbf{H}(\mathbf{x}^t + \mathbf{A}_x\mathbf{z}^t) \\ \mathbf{H}(\mathbf{g}(\mathbf{x}^t) + \mathbf{A}_z\mathbf{z}^t) \end{pmatrix} = \\ (1-\lambda)\mathbf{r}^t + \lambda\mathbf{h}(\mathbf{w}^t), \qquad (11)$$

$$\mathbf{w}^{t+1} = \begin{pmatrix} (1-\gamma)\mathbf{I}_{Nn} & \mathbf{0}_{Nn\times 2nd} & \mathbf{0}_{Nn\times 2nd} \\ \mathbf{0}_{2nd\times 2nd} & \mu\mathbf{F} & (1-\mu)\mathbf{I}_{2nd} \\ \mathbf{0}_{2nd\times 2nd} & \mathbf{I}_{2nd} & \mathbf{0}_{2nd\times 2nd} \end{pmatrix}\mathbf{w}^t + \\ \begin{pmatrix} \gamma\left(\mathbf{I}_{Nn} \quad -\mathbf{I}_{Nn}\right)\mathbf{r}^t \\ 2\mu\epsilon\alpha\rho\mathbf{PAv}(\mathbf{r}^t) \\ \mathbf{0}_{2nd} \end{pmatrix} = \mathbf{Q}\mathbf{w}^t + \mathbf{B}(\mathbf{r}^t). \qquad (12)$$

Given an arbitrary constant $\mathbf{w}^t = \mathbf{w}$ for all t, let function $\mathbf{r}_{eq}(\mathbf{w})$ from (11) be

$$\mathbf{r}_{eq}(\mathbf{w}) = (1-\lambda)\mathbf{r}_{eq}(\mathbf{w}) + \lambda\mathbf{h}(\mathbf{w}) \Rightarrow \mathbf{r}_{eq}(\mathbf{w}) = \mathbf{h}(\mathbf{w}). \quad (13)$$

Then, we define the error coordinates $\tilde{\mathbf{r}}^t = \mathbf{r}^t - \mathbf{r}_{eq}(\mathbf{w})$. Regarding the slow system dynamics (12), we define the desired equilibrium $\mathbf{w}^*$ as

$$\mathbf{w}^* = \begin{pmatrix} \mathbf{x}^* \\ 2(\mathbf{I}_{2nd} + \mathbf{P})^{-1}\rho\mathbf{PAv}(\mathbf{r}_{eq}(\mathbf{w}^t)) \\ 2(\mathbf{I}_{2nd} + \mathbf{P})^{-1}\rho\mathbf{PAv}(\mathbf{r}_{eq}(\mathbf{w}^t)) \end{pmatrix}, \qquad (14)$$

where we have exploited the fact that $\mathbf{z}^*$ can be arbitrarily chosen, making it coincide with the value of $\mathbf{z}^t$ when $\mathbf{z}^t = \mathbf{z}^{t-1}$, which is the steady-state value of $\mathbf{z}^t$ in Eq. (10c). Finally, we define the error dynamics $\tilde{\mathbf{w}}^t = \mathbf{w}^t - \mathbf{w}^*$.

We aim at exploiting Theorem 1 to prove the convergence properties of Algorithm 2. To do so, we first study the convergence properties of the boundary system layer error dynamics $\tilde{\mathbf{r}}^{t+1}$ and the reduced system error dynamics $\tilde{\mathbf{w}}^{t+1}$ by drawing their equivalence with the fast (3b) and slow (3a) dynamics respectively, proving that conditions (7a)-(7c) and (8a)-(8c) hold. In Lemmas 1 and 2 we prove that $\tilde{\mathbf{r}}^{t+1}$ can be written in the form of Eq. (3b) and $\tilde{\mathbf{w}}^{t+1}$ in the form of Eq. (3a). After that, Proposition 1 resorts to Theorem 1 to prove the global exponential stability of the dynamics of $\tilde{\mathbf{r}}^{t+1}$ and $\tilde{\mathbf{w}}^{t+1}$ and, consequently, the global exponential stability of Algorithm 2. Finally, we use Lemma 2 to demonstrate that Algorithm 2 leads to acceleration with respect to Algorithm 1, formalized through Theorem 2, where we substitute the slow state from $\mathbf{p}$ to $\tilde{\mathbf{w}}$ and the fast state from $\mathbf{q}$ to $\tilde{\mathbf{r}}$.

Lemma 1: Let $\mathbf{w}^t = \mathbf{w}$. There exists a continuous function $u(\tilde{\mathbf{r}}^t, t)$ such that, for $\lambda \in (1, 2)$, (7a), (7b) and (7c) hold.
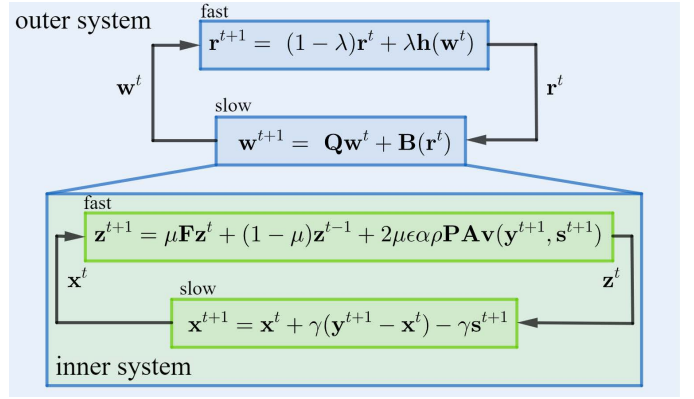
Proof: The boundary system layer dynamics of (11) is

$$\tilde{\mathbf{r}}^{t+1} = \mathbf{r}^{t+1} - \mathbf{r}_{eq}(\mathbf{w}) = (1-\lambda)(\tilde{\mathbf{r}}^t + \mathbf{r}_{eq}(\mathbf{w})) + \lambda\mathbf{h}(\mathbf{w}^t) - \mathbf{r}_{eq}(\mathbf{w}), \quad (15)$$

from which, using (13), it is derived that $\tilde{\mathbf{r}}^{t+1} = (1-\lambda)\tilde{\mathbf{r}}^t$. Thus, if $\lambda \in (1, 2)$, then the error dynamics of $\tilde{\mathbf{r}}^{t+1}$ is globally exponentially stable and the conditions (7a), (7b) and (7c) are satisfied by choosing as Lyapunov function $u(\tilde{\mathbf{r}}^t, t) = \frac{1}{2}(\tilde{\mathbf{r}}^t)^\top\tilde{\mathbf{r}}^t$. ∎

Lemma 2: Let $\mathbf{r}^t = \mathbf{r}_{eq}(\mathbf{w}^t)$, $\lambda, \mu \in (1, 2)$, $\gamma > 0$, $\epsilon, \alpha \in (0, 1)$ and $\rho > 0$. There exists a continuous function $w(\tilde{\mathbf{w}}^t, t)$ such that (8a), (8b) and (8c) hold.

Proof: First, the error dynamics $\tilde{\mathbf{w}}^{t+1}$ are

$$\tilde{\mathbf{w}}^{t+1} = \mathbf{Q}(\tilde{\mathbf{w}}^t + \mathbf{w}^*) + \mathbf{B}(\mathbf{r}_{eq}(\mathbf{w}^t)) - \mathbf{w}^* \quad \Rightarrow \qquad (16)$$

Fig. 1: Representation of Algorithm 2 as an interconnection of two singularly perturbed systems. The outer singularly perturbed system connects the dynamics of the consensus variables $\mathbf{r}^t$ and the estimates/auxiliary variables $\mathbf{w}^t$. The inner singularly perturbed system connects the dynamics of the estimates $\mathbf{x}^t$ and the auxiliary variables $\mathbf{z}^t$.

$$\tilde{\mathbf{w}}^{t+1} = \mathbf{Q}\tilde{\mathbf{w}}^t + \begin{pmatrix} -\gamma\mathbf{x}^* \\ \mu\mathbf{F}\mathbf{z}^* - \mu\mathbf{z}^* \\ \mathbf{z}^* - \mathbf{z}^* \end{pmatrix} + \mathbf{B}(\mathbf{r}_{eq}(\mathbf{w}^t)). \qquad (17)$$

By substituting the definition of $\mathbf{z}^*$ from Eq. (14) in matrix $\mathbf{B}(\mathbf{r}_{eq}(\mathbf{w}^t))$ of Eq. (17), we obtain that

$$\tilde{\mathbf{w}}^{t+1} = \mathbf{Q}\tilde{\mathbf{w}}^t + \begin{pmatrix} -\gamma\mathbf{x}^* + \gamma\left(\mathbf{I}_{Nn} \quad -\mathbf{I}_{Nn}\right)\mathbf{h}(\mathbf{w}^t) \\ \mathbf{0}_{2nd} \\ \mathbf{0}_{2nd} \end{pmatrix}. \qquad (18)$$

According to (18), the error dynamics associated to $\mathbf{z}^{t+1}$ and $\mathbf{z}^t$ do not depend on $\mathbf{x}^t$

$$\begin{pmatrix} \tilde{\mathbf{z}}^{t+1} \\ \tilde{\mathbf{z}}^t \end{pmatrix} = \begin{pmatrix} \mu\mathbf{F} & (1-\mu)\mathbf{I}_{2nd} \\ \mathbf{I}_{2nd} & \mathbf{0}_{2nd\times 2nd} \end{pmatrix}\begin{pmatrix} \tilde{\mathbf{z}}^t \\ \tilde{\mathbf{z}}^{t-1} \end{pmatrix} = \mathbf{L}\begin{pmatrix} \tilde{\mathbf{z}}^t \\ \tilde{\mathbf{z}}^{t-1} \end{pmatrix}, \quad (19)$$

so they can be independently analyzed. Moreover, (19) can be rewritten in the form of (3a) by adding and subtracting the identity matrix from $\mathbf{L}$. On the other hand, the error dynamics associated to $\mathbf{x}^t$

$$\tilde{\mathbf{x}}^{t+1} = \tilde{\mathbf{x}}^t + \gamma(\mathbf{H}(\mathbf{x}^t + \mathbf{A}_x\mathbf{z}^t) - \mathbf{x}^t) - \gamma\mathbf{H}(\mathbf{g}(\mathbf{x}^t) + \mathbf{A}_z\mathbf{z}^t) \quad (20)$$

are the error dynamics of the reduced system of the inner singularly perturbed system, which are the same to those analyzed in [26]. In particular, Theorem III.1 in [26] proves that Algorithm 1 is globally exponentially stable when $\rho > 0$, $\alpha \in (0, 1)$ and $\gamma > 0$ sufficiently small. Conveniently, the proof of Theorem III.1 in [26] refers to the same dynamics in (20), and therefore the result can be applied. However, this result holds only if the dynamics in (19) are also globally exponentially stable. Otherwise, Theorem 1 applied to the dynamics of $\mathbf{x}^t$ and $\mathbf{z}^t$ does not hold. Therefore, the next step is to prove the global exponential stability of (19). The error dynamics in (19) are globally exponentially stable if and only if $\{|\sigma_l(\mathbf{L})| < 1\}_{l=1}^{4nd}$. Taking into account that $\det(\mathbf{L} - \sigma\mathbf{I}_{4nd}) = 0$ and that $\mathbf{F} = (1 - \epsilon\alpha)\mathbf{I}_{2nd} - \epsilon\alpha\mathbf{P}$, the eigenvalues of $\mathbf{L}$ are

$$-\sigma^2 + \sigma(\mu - \mu\alpha\epsilon \pm \mu\alpha\epsilon) + (1-\mu) = 0. \qquad (21)$$

The $\pm$ in (21) comes from the fact that $\det(\mathbf{P}) = (-1)^p$, where p is the number of permutations. If p is even,

$$-\sigma^2 + \mu\sigma + (1-\mu) = 0 \Rightarrow \sigma = \frac{\mu}{2} \pm \sqrt{\frac{\mu^2}{4} + (1-\mu)}. \quad (22)$$

From (22), $-1 < \sigma < 1$. If $\mu \in (1,2)$. If p is odd, then

$$-\sigma^2 + \mu(1-2\alpha\epsilon)\sigma + (1-\mu) = 0. \quad (23)$$

Equation (23) implies that

$$-1 < \frac{\mu(1-2\epsilon\alpha)}{2} \pm \sqrt{\frac{\mu^2}{4}(1-2\epsilon\alpha)^2 + (1-\mu)} < 1. \quad (24)$$

By solving the four inequality equations in (24) and taking the more restrictive conditions, we obtain that $\epsilon\alpha \in (0,1)$, and since $\alpha \in (0,1)$, then $\epsilon \in (0,1)$. Therefore, if $\epsilon \in (0,1)$ and $\mu \in (1,2)$, then the error dynamics in (19) are globally exponentially stable. This implies that the error dynamics in (20) are globally exponentially stable as well and the conditions for the inner singularly perturbed system in Theorem 1 hold. Subsequently, the error dynamics in (18) are globally exponentially stable and there exist a Lyapunov function $w(\tilde{\mathbf{w}}^t, t)$ such that (8a), (8b) and (8c) are satisfied. ∎

Proposition 1: Consider Algorithm 2 with $\rho > 0$, $\alpha, \epsilon \in (0,1)$ and $\lambda, \mu \in (1,2)$. Then, there exists $\bar{\gamma}, c_1, c_2 > 0$ such that, for any $\gamma \in (0, \bar{\gamma})$, $(\mathbf{x}^0, \mathbf{z}^0, \mathbf{z}^{-1}, \mathbf{y}^0, \mathbf{s}^0)$, it holds that $||\mathbf{x}^t - \mathbf{1}_{N,n}^\top \mathbf{x}^*|| \le c_1 e^{-c_2 t}$.

Proof: Let $\tilde{\mathbf{w}}^t$ correspond to the error state of the slow dynamics in (3a) and $\tilde{\mathbf{r}}^t$ the error state of the fast error dynamics in (3b). According to Lemmas 1 and 2, the conditions (7a)-(8c) hold for the error dynamics of $\tilde{\mathbf{w}}^t$ and $\tilde{\mathbf{r}}^t$ respectively. Besides, from the definition of $\mathbf{r}_{\text{eq}}(\mathbf{w}^t)$ it can be verified that condition (4) holds. Then, Theorem 1 can be invoked to prove the global exponential stability of $\tilde{\mathbf{w}}^t$ and $\tilde{\mathbf{r}}^t$. By changing the coordinates from $\tilde{\mathbf{w}}^t$ and $\tilde{\mathbf{r}}^t$ to $\mathbf{w}^t$ and $\mathbf{r}^t$, the proof ends. ∎

Proposition 1 establishes that Algorithm 2 linearly converges to the global optimum $\mathbf{x}^*$. Nevertheless, acceleration is not proved yet. This is proved in the next theorem.

Theorem 2: Consider $\rho > 0$, $\alpha \in (0,1)$ and $\lambda \in (1,2)$, and $\gamma > 0$ sufficiently small. Also, let β_1 and β_2 be the convergence rate of the dynamics of $\mathbf{z}^t$ for Algorithm 1 and 2 respectively. Then, Algorithm 2 always converges faster than Algorithm 1 ($\beta_2 < \beta_1$) for any $\epsilon \in (0,1)$ and $\mu \in (1,2)$.

Proof: In Lemma 2 Eq. (19) it is proven that the dynamics of $\tilde{\mathbf{z}}^t$ are determined by $\mathbf{L}$. Therefore, the convergence rate of $\tilde{\mathbf{z}}^t$ and subsequently $\mathbf{z}^t$ is given by $\max(\{|\sigma_l(\mathbf{L})|\}_{l=1}^{4nd})$. Therefore, we have that

$$\max(\{|\sigma_l(\mathbf{L})|\}_{l=1}^{4nd}) = \max\left(\left| \frac{\mu}{2} + \sqrt{\frac{\mu^2}{4} + (1-\mu)} \right|, \right.$$
$$\left. \left| \frac{\mu(1-2\epsilon\alpha)}{2} + \sqrt{\frac{\mu^2}{4}(1-2\epsilon\alpha)^2 + (1-\mu)} \right| \right).$$

Moreover, notice that Algorithm 1 is Algorithm 2 with $\mu = \epsilon = 1$. Thus, $\beta_1 = \max(1, |1-2\alpha|)$ for Algorithm 1. Under the assumption that no integral term is present, $\beta_1 = |1-2\alpha|$; otherwise, $\beta_1 = 1$. On the other hand, under the constraints on μ, α, ϵ in Proposition 1, we have that $\left| \frac{\mu}{2} + \sqrt{\frac{\mu^2}{4} + (1-\mu)} \right| < 1$, and also $\left| \frac{\mu(1-2\epsilon\alpha)}{2} + \sqrt{\frac{\mu^2}{4}(1-\epsilon\alpha)^2 + (1-\mu)} \right| < |1-2\alpha|$. Henceforth, $\beta_2 < \beta_1$. ∎

Proposition 1 establishes that the slow and fast dynamics of Algorithm 2 must evolve at different time-scales to ensure convergence. Therefore, the convergence rate of the auxiliary variables determines the convergence speed of the optimum estimates. Thanks to adding momentum, γ, ρ and α can be increased without violating the conditions of time-scale decoupling of singularly perturbed systems, improving the convergence speed.

IV. ILLUSTRATIVE EXAMPLE

We compare our algorithm (A2DMM-GT) with the one in [26] (ADMM-GT), the accelerated gradient tracking approaches in [14] (AGT) and [19] (A2GT), and the older distributed gradient tracking protocol in [13] (DIGing). We consider two settings. First, a quadratic programming setup, that arises in robotic, smart grid or feature selection applications [29]. We set $N = 200$, $n = 2$ and $f_i(\mathbf{x}) = \mathbf{x}^\top \mathbf{B}_i \mathbf{x} + \mathbf{a}_i \mathbf{x}$, where $\mathbf{B}_i$ is a positive definite matrix with uniformly random eigenvalues in the range $[1,5]$ and $\mathbf{a}_i$ is a random vector with elements uniformly drawn in the range $[-10, 20]$. Note that $f_i(\mathbf{x})$ fulfills the assumptions in terms of strong convexity and Lipschitzness of the gradients. Graph $\mathcal{G}$ is generated as a sparse undirected connected proximity graph, where the nodes are randomly placed in a 2×2 square if and only if the new node is within $[0.1, 0.17]$ from an already placed node. The parameters for the all the approaches are hand-tuned to achieve fastest convergence possible: $\gamma = 1.6$, $\alpha = 0.9924$, $\rho = 3.028$, $\lambda = 0.2$, $\mu = 1.6$, $\epsilon = 0.6$ for A2DMM-GT; $\gamma = 0.4865$, $\alpha = 0.8924$, $\rho = 0.528$ for ADMM-GT; $\alpha = 0.001$, $\beta = 0.9$ for AGT (we use the notation in [14]); $\alpha = 0.0008$, $\beta = 0.7$, $\gamma = 0.2$ for A2GT (we use the notation in [19]); $\gamma = 0.0127$ for DIGing. We simulate 500 steps. Convergence is assessed using the evolution of the error $\mathbf{e}^t = ||\mathbf{x}^t - \mathbf{1}_{N,n}\mathbf{x}^*||$.

The second setup is a logistic regression scenario where nodes cooperate to train a linear classifier. In this case, $N = 50$, where each node has a local dataset of $m = 10$ one-dimensional points p_i^j drawn from a normal distribution of mean 0 and standard deviation 1. The labels are also randomly drawn, such that $l_i^j \in \{-1, 1\}$. We define $\mathbf{x} = [q_1, q_2]$ and $f_i(\mathbf{x}) = \sum_{j=1}^m \log(1 + \exp^{-l_i^j(q_1 p_i^j + q_2)}) + C||\mathbf{x}||^2$, with $C = 1$ to ensure strong-convexity. The parameters for all methods are the same that in quadratic programming setup.

The evolution of the error (Fig. 2) shows that A2DMM-GT is the fastest method among all the approaches for both numerical settings, with different network configurations and local costs. Importantly, this is achieved without increasing the number of gradient computations per time step nor the communication burden, leading to a significant reduction in power consumption and total bandwidth usage in practical applications. This comes at the cost of increasing the memory burden compared to the non-accelerated methods. However, given current advances in MEMS technology, it is generally better to leverage memory against communication burden.
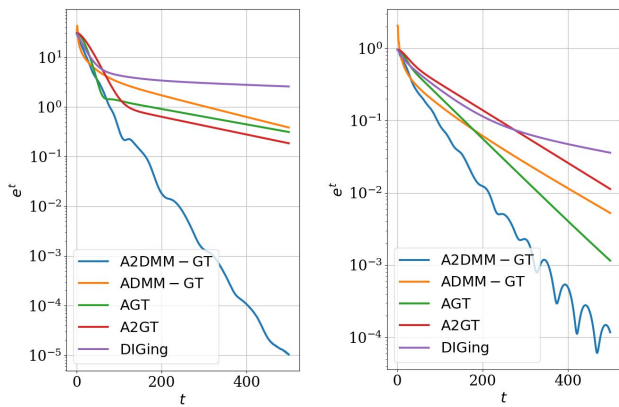
Fig. 2: Evolution of the error with time for the five protocols under comparison. Left is the quadratic programming setup, right is the linear classification setup.

V. CONCLUSIONS

This letter presents a novel accelerated distributed optimization method for the problem of consensus optimization. The method combines the benefits of momentum to those from gradient tracking and ADMM algorithms, namely: fast convergence, simplicity of computation and robustness. From the theoretical point of view, we prove that, by adding momentum, we overcome the limitation on the convergence rate of the auxiliary variables in current ADMM-GT algorithms. From a practical perspective, we found that our algorithm is faster than the existing distributed first-order algorithms, while preserving their lightweight properties in terms of computation and communication burden. As future work, we aim at exploring the effect of adding momentum in the step size γ, the extension of the algorithm to time-varying directed graphs, and the extension to constrained settings.

REFERENCES

[1] A. Nedić, A. Olshevsky, and M. G. Rabbat, "Network topology and communication-computation tradeoffs in decentralized optimization," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 953–976, 2018.

[2] A. Nedić and J. Liu, "Distributed optimization for control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 77–103, 2018.

[3] A. Nedic, "Distributed gradient methods for convex machine learning problems in networks: Distributed optimization," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 92–101, 2020.

[4] F. Parise and A. Ozdaglar, "A variational inequality framework for network games: Existence, uniqueness, convergence and sensitivity analysis," *Games and Economic Behavior*, vol. 114, pp. 47–82, 2019.

[5] O. Shorinwa, T. Halsted, J. Yu, and M. Schwager, "Distributed Optimization Methods for Multi-robot Systems: Part 2-A Survey," *IEEE Robotics & Automation Magazine*, 2024.

[6] S. Sundhar Ram, A. Nedić, and V. V. Veeravalli, "Distributed stochastic subgradient projection algorithms for convex optimization," *Journal of optimization theory and applications*, vol. 147, pp. 516–545, 2010.

[7] D. Jakovetić, J. Xavier, and J. M. Moura, "Fast distributed gradient methods," *IEEE Transactions on Automatic Control*, vol. 59, no. 5, pp. 1131–1146, 2014.

[8] D. P. Bertsekas, "Incremental proximal methods for large scale convex optimization," *Mathematical programming*, vol. 129, no. 2, pp. 163–195, 2011.

[9] M. Franceschelli and A. Gasparri, "Multi-stage discrete time and randomized dynamic average consensus," *Automatica*, vol. 99, pp. 69–81, 2019.

[10] E. Sebastián, E. Montijano, C. Sagüés, M. Franceschelli, and A. Gasparri, "Accelerated multi-stage discrete time dynamic average consensus," *IEEE Control Systems Letters*, vol. 7, pp. 2731–2736, 2023.

[11] W. Shi, Q. Ling, G. Wu, and W. Yin, "Extra: An exact first-order algorithm for decentralized consensus optimization," *SIAM Journal on Optimization*, vol. 25, no. 2, pp. 944–966, 2015.

[12] G. Qu and N. Li, "Harnessing smoothness to accelerate distributed optimization," *IEEE Transactions on Control of Network Systems*, vol. 5, no. 3, pp. 1245–1260, 2017.

[13] A. Nedic, 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.

[14] 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, 2019.

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

[16] N. Bastianello, R. Carli, L. Schenato, and M. Todescato, "Asynchronous distributed optimization over lossy networks via relaxed ADMM: Stability and linear convergence," *IEEE Transactions on Automatic Control*, vol. 66, no. 6, pp. 2620–2635, 2020.

[17] I. Notarnicola, M. Franceschelli, and G. Notarstefano, "A duality-based approach for distributed min–max optimization," *IEEE Transactions on Automatic Control*, vol. 64, no. 6, pp. 2559–2566, 2019.

[18] M. Bin, I. Notarnicola, and T. Parisini, "Stability, linear convergence, and robustness of the wang-elia algorithm for distributed consensus optimization," in *IEEE Conference on Decision and Control*, 2022, pp. 1610–1615.

[19] D. T. A. Nguyen, D. T. Nguyen, and A. Nedić, "Accelerated AB/push-pull methods for distributed optimization over time-varying directed networks," *IEEE Transactions on Control of Network Systems*, 2023.

[20] K. Huang, S. Pu, and A. Nedić, "An accelerated distributed stochastic gradient method with momentum," *arXiv:2402.09714*, 2024.

[21] G. Qu and N. Li, "Accelerated distributed nesterov gradient descent," *IEEE Transactions on Automatic Control*, vol. 65, no. 6, pp. 2566–2581, 2019.

[22] A. Fu, J. Zhang, and S. Boyd, "Anderson accelerated douglas–rachford splitting," *SIAM Journal on Scientific Computing*, vol. 42, no. 6, pp. A3560–A3583, 2020.

[23] M. Adil, S. Tavakkol, and R. Madani, "Rapid convergence of first-order numerical algorithms via adaptive conditioning," *arXiv preprint arXiv:2103.00736*, 2021.

[24] Z. Li, W. Shi, and M. Yan, "A decentralized proximal-gradient method with network independent step-sizes and separated convergence rates," *IEEE Transactions on Signal Processing*, vol. 67, no. 17, pp. 4494–4506, 2019.

[25] G. Carnevale, N. Bastianello, R. Carli, and G. Notarstefano, "Distributed consensus optimization via ADMM-tracking gradient," in *IEEE Conference on Decision and Control*, 2023, pp. 290–295.

[26] G. Carnevale, N. Bastianello, G. Notarstefano, and R. Carli, "ADMM-Tracking Gradient for Distributed Optimization over Asynchronous and Unreliable Networks," *arXiv:2309.14142*, 2023.

[27] B. Ghosh, S. Muthukrishnan, and M. H. Schultz, "First and second order diffusive methods for rapid, coarse, distributed load balancing," in *ACM symposium on Parallel algorithms and architectures*, 1996, pp. 72–81.

[28] G. Carnevale, F. Fabiani, F. Fele, K. Margellos, and G. Notarstefano, "Tracking-based distributed equilibrium seeking for aggregative games," *IEEE Transactions on Automatic Control*, 2024.

[29] M. Ubl and M. T. Hale, "Totally asynchronous distributed quadratic programming with independent stepsizes and regularizations," in *IEEE Conference on Decision and Control*, 2019, pp. 7423–7428.