2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Distributed Consensus Optimization via ADMM-Tracking Gradient

Guido Carnevale, Nicola Bastianello, Ruggero Carli, Giuseppe Notarstefano

***Abstract*— In this paper, we propose a novel distributed algorithm for consensus optimization over networks. The key idea is to achieve dynamic consensus on the agents' average and on the global descent direction by iteratively solving an online auxiliary optimization problem through the Alternating Direction Method of Multipliers (ADMM). Such a mechanism is suitably interlaced with a local proportional action steering each agent estimate to the solution of the original consensus optimization problem. The analysis uses tools from system theory to prove the linear convergence of the scheme with strongly convex costs. Finally, some numerical simulations confirm our findings and show the robustness of the proposed scheme.**

I. Introduction

Recently, consensus (or cost-coupled) optimization has gained popularity, see the recent surveys [1]–[3] for the related applications and the state-of-the-art distributed algorithms. In this setup, large attention has been received by the so-called *Gradient Tracking* distributed algorithm originally proposed in in [4]–[6] and extended to deal with different problem setups like, e.g., the asynchronous case [7], the nonconvex setting [8]–[10], or the online one [11], [12]. However, as highlighted in [10], [13], [14], Gradient Tracking suffers the presence of a marginally stable dynamics related to the embedded perturbed consensus dynamics (see [15], [16]) devoted to reconstruct the unavailable global gradient of the problem cost function. In ideal setups, a suitable initialization is sufficient to avoid undesired bias due to such a marginally stable part that, however, as shown in [17], makes the whole scheme not robust with respect to the presence of errors (due, e.g., to quantization effects, asynchronous updates, or unreliable communication).

On the contrary, in [18], it is shown that dynamic average consensus can be robustly addressed by suitably formulating an associated *online* optimization problem and using the (static) distributed version of *Alternating Direction Method of Multipliers* (ADMM) as proposed in [19]. The use of ADMM for distributed optimization was popularized by the monograph [20], and has been extensively studied since, see *e.g.* [21]–[23]. Its application for (static) average consensus was also discussed in [24], and for online distributed optimization in [25]–[27].

The main contribution of this work is the design of ADMM-Tracking Gradient, i.e., a novel distributed algorithm for consensus optimization. The idea consists of controlling each solution estimate through a proportional action with a twofold purpose of removing (i) consensus error among agents' estimates, and (ii) the optimality error. Such an action would require unavailable global information in each agent. Thus, inspired by [18], we formulate an auxiliary, online optimization problem whose solution coincides with the unavailable information. We dynamically tackle this online problem through a distributed implementation of the ADMM and, in each agent, we suitably interlace its output with the mentioned local, proportional action. By using tools from system theory, it is possible to show that the overall strategy exponentially steers the agents' estimates to the consensus configuration coinciding with the optimal solution of the original problem. In detail, we interpret our algorithm as a *singularly perturbed* system given by the interconnection between (i) a slow subsystem given by the dynamics of the solution estimates, and (ii) a fast one related to the states of the ADMM. Numerical tests assess that ADMM-Tracking Gradient outperforms Gradient Tracking in terms of convergence rate and robustness.

The paper is organized as follows. Section II introduces the problem setup. In Section III, we design the novel distributed algorithm named ADMM-Tracking Gradient and provide its convergence properties. Section IV reformulate ADMM-Tracking Gradient as a singularly perturbed system Finally, in Section V, we compare ADMM-Tracking Gradient and Gradient Tracking on (i) a quadratic setup and (ii) a noisy, logistic regression scenario.

Notation: A square matrix $M \in \mathbb{R}^{n\times n}$ is said to be Schur if all its eigenvalues lie in the open unit circle. The identity matrix in $\mathbb{R}^{m\times m}$ is I_m. , while 0_m is the all-zero matrix in $\mathbb{R}^{m\times m}$. The vector of N ones is denoted by 1_N, while $\mathbf{1}_{N,n} := 1_N \otimes I_n$ with $\otimes$ being the Kronecker product. Dimensions are omitted whenever they are clear from the context. For a finite set S, we denote by $|S|$ its cardinality. The vertical concatenation of the column vectors $v_1, \dots, v_N$ is $\text{COL}(v_i)_{i\in\{1,\dots,N\}}$. We denote as $\text{blkdiag}(M_1, \dots, M_N) \in \mathbb{R}^{\sum_{i=1}^N n_i}$ the block diagonal matrix whose i-th block is given by $M_i \in \mathbb{R}^{n_i\times n_i}$.

This work was supported in part by the Italian Ministry of Foreign Affairs and International Cooperation, grant number BR22GR01.

Guido Carnevale and Giuseppe Notarstefano are with the Department of Electrical, Electronic and Information Engineering, Alma Mater Studiorum - Universita' di Bologna, Bologna, Italy. e-mail: {guido.carnevale, giuseppe.notarstefano}@unibo.it.

Nicola Bastianello is with the School of Electrical Engineering and Computer Science and Digital Futures, KTH Royal Institute of Technology, Stockholm, Sweden. Email: nicolba@kth.se

Ruggero Carli is with the Department of Information Engineering of the University of Padova, Via G. Gradenigo 6/B, 35131 Padova, Italy. Email: carlirug@dei.unipd.it

II. Problem Description and Preliminaries

We consider a network of N agents that aim to solve

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

where $x \in \mathbb{R}^n$ is the (common) decision variable, while $f_i : \mathbb{R}^n \to \mathbb{R}$ is the objective function of agent $i \in \{1,\dots,N\}$. In the following, we will also use the function $f : \mathbb{R}^n \to \mathbb{R}$ defined as $f(x) := \sum_{i=1}^{N} f_i(x)$. Our goal is to design an algorithm to solve (1) in a distributed way, namely with update laws implementable over a network of agents using only (i) local information and (ii) neighboring communication. Indeed, we consider a network of agents communicating according to an undirected graph $\mathcal{G} = (\{1,\dots,N\}, \mathcal{E})$, with $\mathcal{E} \subset \{1,\dots,N\}\times\{1,\dots,N\}$ such that i and j can exchange information only if $(i,j) \in \mathcal{E}$. The set of neighbors of i is $\mathcal{N}_i := \{j \in \{1,\dots,N\} \mid (i,j) \in \mathcal{E}\}$, while its degree is $d_i := |\mathcal{N}_i|$ and $\mathbf{d} := \sum_{i=1}^{N} d_i$. Notice that it holds $\mathbf{d} = 2|\mathcal{E}|$, where $|\cdot|$ denotes the cardinality operator.

The following assumptions formalize the considered setup.

Assumption II.1. *[Network Connectivity] The graph $\mathcal{G}$ is connected.* □

Assumption II.2. *[Objective functions] The objective function f is* **c***-strongly convex, while the gradients $\nabla f_i(\cdot)$ are L-Lipschitz continuous for all $i \in \{1,\dots,N\}$.* □

We notice that Assumption II.2 ensures that problem (1) has a unique minimizer and we denote it as $x^\star \in \mathbb{R}^n$.

III. ADMM-Tracking Gradient: Algorithm Design and Convergence Properties

Let $x_i^t \in \mathbb{R}^n$ be the estimate of the solution to problem (1) maintained by agent i at iteration $t \in \mathbb{N}$. We follow a control-oriented design for the update of x_i^t. Then, let $u_i^t \in \mathbb{R}^n$ be the i-th control input and consider the single integrator dynamics

$$x_i^{t+1} = x_i^t + u_i^t. \tag{2}$$

The control law determining u_i^t should have the twofold purpose of removing (i) the consensus error with respect to the other agents' estimates, and (ii) the optimality error related to problem (1). Hence, one may design u_i^t as a proportional action with respect to the above errors, namely

$$u_i^t = \gamma\left(\sum_{j=1}^{N} x_j^t/N - x_i^t\right) - \frac{\gamma}{N}\sum_{j=1}^{N}\nabla f_j(x_j^t), \tag{3}$$

where $\gamma > 0$ is a tuning gain. By plugging the control law (3) into (2), we get the closed-loop dynamics

$$x_i^{t+1} = x_i^t + \gamma\left(\sum_{j=1}^{N} x_j^t/N - x_i^t\right) - \frac{\gamma}{N}\sum_{j=1}^{N}\nabla f_j(x_j^t). \tag{4}$$

However, in a distributed setting, agent i cannot access the global terms $\frac{1}{N}\sum_{j=1}^{N} x_j^t$ and $\frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j^t)$. Therefore, we modify the control law (3) by employing two auxiliary variables $y_i^t, s_i^t \in \mathbb{R}^n$ aimed at reconstructing $\frac{1}{N}\sum_{j=1}^{N} x_j^t$ and $\frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j^t)$, respectively. More in detail, the control law (3) is replaced by

$$u_i^t = \gamma(y_i^t - x_i^t) - \gamma s_i^t. \tag{5}$$

We note that, if $y_i^t = \frac{1}{N}\sum_{j=1}^{N} x_j^t$ and $s_i^t = \frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j^t)$, the desired update (3) is recovered. Then, inspired by [18], for each iteration $t \geq 0$, we turn these two consensus-oriented goals into the online optimization problem

$$\begin{aligned}\min_{\substack{(y_1,\dots,y_N)\in\mathbb{R}^{Nn}\\(s_1,\dots,s_N)\in\mathbb{R}^{Nn}}} &\sum_{i=1}^{N} g_i^t(y_i,s_i)\\ \text{s.t.: } &\begin{bmatrix}y_i\\s_i\end{bmatrix} = \begin{bmatrix}y_j\\s_j\end{bmatrix} \text{ if } (i,j)\in\mathcal{E},\end{aligned} \tag{6}$$

where, for all $i \in \{1,\dots,N\}$, $g_i^t : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ reads as

$$g_i^t(y_i,s_i) = \frac{1}{2}\left\|y_i - x_i^t\right\|^2 + \frac{1}{2}\left\|s_i - \nabla f_i(x_i^t)\right\|^2.$$

Indeed, if the graph $\mathcal{G}$ is connected, the (unique) optimal solution of problem (6), say it $(y_\star^t, s_\star^t) \in \mathbb{R}^{2Nn}$, reads as $(y_\star^t, s_\star^t) = (\mathbf{1}_{N,n}\frac{1}{N}\sum_{j=1}^{N} x_j^t, \mathbf{1}_{N,n}\frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j^t))$ [18]. From this observation, we design the updates of y_i^t and s_i^t by resorting to the distributed ADMM proposed in [19]. Hence, each agent i maintains an additional variable $z_{ij}^t \in \mathbb{R}^{2n}$ for each neighbor $j \in \mathcal{N}_i$ and implements

$$\begin{aligned}\begin{bmatrix}y_i^t\\s_i^t\end{bmatrix} = \underset{\substack{y_i\in\mathbb{R}^n\\s_i\in\mathbb{R}^n}}{\arg\min}\Bigg\{ & g_i^t(y_i,s_i) - \begin{bmatrix}y_i^\top & s_i^\top\end{bmatrix}\sum_{j\in\mathcal{N}_i} z_{ij}^t\\ &+ \frac{\rho d_i}{2}\left(\|y_i\|^2 + \|s_i\|^2\right)\Bigg\}\\ z_{ij}^{t+1} = (1-\alpha)z_{ij}^t + \alpha&\left(-z_{ji}^t + 2\rho\begin{bmatrix}y_j^t\\s_j^t\end{bmatrix}\right),\end{aligned}$$

with $\rho > 0$ and $\alpha \in (0,1)$. Being g_i^t quadratic, the above updates are equivalent to the closed form

$$\begin{bmatrix}y_i^t\\s_i^t\end{bmatrix} = \frac{1}{1+\rho d_i}\left(\begin{bmatrix}x_i^t\\\nabla f_i(x_i^t)\end{bmatrix} + \sum_{j\in\mathcal{N}_i} z_{ij}^t\right) \tag{7a}$$

$$z_{ij}^{t+1} = (1-\alpha)z_{ij}^t + \alpha\left(-z_{ji}^t + 2\rho\begin{bmatrix}y_j^t\\s_j^t\end{bmatrix}\right). \tag{7b}$$

Let us introduce $m_{ji}^t \in \mathbb{R}^{2n}$ to denote the message from agent j needed by agent i to perform (7b), namely

$$m_{ji}^t := -z_{ji}^t + 2\rho\begin{bmatrix}y_j^t\\s_j^t\end{bmatrix}. \tag{8}$$

Then, we compactly rewrite (7) as

$$\begin{bmatrix}y_i^t\\s_i^t\end{bmatrix} = \frac{1}{1+\rho d_i}\left(\begin{bmatrix}x_i^t\\\nabla f_i(x_i^t)\end{bmatrix} + \sum_{j\in\mathcal{N}_i} z_{ij}^t\right) \tag{9a}$$

$$z_{ij}^{t+1} = (1-\alpha)z_{ij}^t + \alpha m_{ji}^t. \tag{9b}$$

We report in Algorithm 1 and we name it ADMM-Tracking Gradient the whole distributed protocol arising by plugging (5) into (2) and combining them with (9).

Algorithm 1 ADMM-Tracking Gradient (Agent i)

Initialization: $x_i^0 \in \mathbb{R}^n, z_i^0 \in \mathbb{R}^{2nd_i}$.
for $t = 0, 1, \dots$ **do**
  $\begin{bmatrix} y_i^t \\ s_i^t \end{bmatrix} = \frac{1}{1+\rho d_i}\left(\begin{bmatrix} x_i^t \\ \nabla f_i(x_i^t) \end{bmatrix} + \sum_{j\in\mathcal{N}_i} z_{ij}^t\right)$
  $x_i^{t+1} = x_i^t + \gamma(y_i^t - x_i^t) - \gamma s_i^t$
  for $j \in \mathcal{N}_i$ **do**
    $m_{ij}^t = -z_{ij}^t + 2\rho \begin{bmatrix} y_i^t \\ s_i^t \end{bmatrix}$
    transmit m_{ij}^t to j and receive m_{ji}^t from j
    $z_{ij}^{t+1} = (1-\alpha) z_{ij}^t + \alpha m_{ji}^t$
  end for
end for

Algorithm 1 can be implemented in a fully-distributed fashion since it only requires neighboring communication and local variables. In particular, when running Algorithm 1, each agent needs to exchange variables with $2n$ components with its neighbors. The next theorem states the convergence features of ADMM-Tracking Gradient.

Theorem III.1. *Consider ADMM-Tracking Gradient and let Assumptions II.1 and II.2 hold. Then, there exist $\bar{\gamma}, a_1, a_2 > 0$ such that, for any $\gamma \in (0, \bar{\gamma})$, $(x_i^0, z_i^0) \in \mathbb{R}^n \times \mathbb{R}^{2nd_i}$, it holds*

$$\left\|x_i^t - x^\star\right\| \leq a_1 \exp(-a_2 t),$$

for all $i \in \{1, \dots, N\}$. □

Due to space constraints, the proof of Theorem III.1 is omitted in this paper and will be provided in a forthcoming document. The proof is based on reformulating the aggregate version of ADMM-Tracking Gradient as a *singularly perturbed system*, i.e., the interconnection between a slow subsystem and a fast one having an equilibrium parametrized in the slow state. Based on this interpretation, given some $z^\star \in \mathbb{R}^{2n\mathbf{d}}$ that will be clear later, it is possible to show that $(\mathbf{1}_{N,n} x^\star, z^\star)$ is a globally exponentially stable equilibrium point for the aggregate form of ADMM-Tracking Gradient.

IV. ADMM-Tracking Gradient as a Singularly Perturbed System

First of all, let us rewrite ADMM-Tracking Gradient in a more compact form. To this end, let $z_i^t \in \mathbb{R}^{2d_i n}$ be the vector stacking all the variables z_{ij}^t of the agent i, i.e., $z_i^t := \text{COL}(z_{ij}^t)_{j\in\mathcal{N}_i} \in \mathbb{R}^{2nd_i}$, while let $z_{\mathcal{N}_i}^t := \text{COL}(z_{ji}^t)_{j\in\mathcal{N}_i} \in \mathbb{R}^{2nd_i}$, i.e., the vector stacking all the variables z_{ji}^t with $j \in \mathcal{N}_i$. Moreover, let us introduce the functions $h_i^x : \mathbb{R}^n \times \mathbb{R}^{nd_i} \to \mathbb{R}^n$, $h_i^\nabla : \mathbb{R}^n \times \mathbb{R}^{nd_i} \to \mathbb{R}^n$, and $h_{\mathcal{N}_i} : \mathbb{R}^{nd_i} \times \mathbb{R}^{nd_i} \to \mathbb{R}^{2nd_i}$ defined as

$$h_i^x(x_i, \psi_i) = \frac{x_i + \sum_{j\in\mathcal{N}_i} \psi_{ij}}{1+\rho d_i}$$

$$h_i^\nabla(x_i, \psi_i) = \frac{\nabla f_i(x_i) + \sum_{j\in\mathcal{N}_i} \psi_{ij}}{1+\rho d_i}$$

$$h_{\mathcal{N}_i}(x_{\mathcal{N}_i}, z_{\mathcal{N}_i}) = \text{COL}\left(\frac{\text{COL}(x_j^t, \nabla f_j(x_j^t)) + \sum_{j\in\mathcal{N}_i} z_{ji}^t}{1+\rho d_j}\right)_{j\in\mathcal{N}_i},$$

where $\psi_i \in \mathbb{R}^{nd_i}$ has been decomposed as $\text{COL}(\psi_{ij})_{j\in\mathcal{N}_i}$ with $\psi_{ij} \in \mathbb{R}^n$ for all $j \in \mathcal{N}_i$. With these functions at hand, we are able to rewrite the local update characterizing ADMM-Tracking Gradient (cf. Algorithm 1) in a more compact form described by

$$x_i^{t+1} = x_i^t + \gamma\left(h_i^x(x_i^t, z_i^t) - x_i^t\right) - \gamma h_i^\nabla(x_i^t, z_i^t) \tag{10a}$$

$$z_i^{t+1} = (1-\alpha) z_i^t + \alpha(-z_{\mathcal{N}_i}^t + 2\rho h_{\mathcal{N}_i}(x_{\mathcal{N}_i}^t, z_{\mathcal{N}_i}^t)), \tag{10b}$$

Now, let us provide the aggregate formulation of (10). To this end, let us introduce the permutation matrix $P \in \mathbb{R}^{2n\mathbf{d}\times 2n\mathbf{d}}$ that swaps the ij-th element with the ji-th element, the matrices $A_x \in \mathbb{R}^{2n\mathbf{d}\times Nn}$, $A_z \in \mathbb{R}^{2n\mathbf{d}\times Nn}$, $A \in \mathbb{R}^{2n\mathbf{d}\times 2Nn}$, $H \in \mathbb{R}^{Nn\times Nn}$, $\mathcal{H} \in \mathbb{R}^{2Nn\times 2Nn}$, and $T_{\alpha\rho} \in \mathbb{R}^{2n\mathbf{d}\times 2n\mathbf{d}}$ defined as

$$A_x := \begin{bmatrix} \mathbf{1}_{d_1,n} & & \\ 0_{d_1 n} & & \\ & \ddots & \\ & & \mathbf{1}_{d_N,n} \\ & & 0_{d_N n} \end{bmatrix}$$

$$A_z := \begin{bmatrix} 0_{d_1 n} & & \\ \mathbf{1}_{d_1,n} & & \\ & \ddots & \\ & & 0_{d_N n} \\ & & \mathbf{1}_{d_N,n} \end{bmatrix}$$

$$H := \begin{bmatrix} \frac{1}{1+\rho d_1} I_n & & \\ & \ddots & \\ & & \frac{1}{1+\rho d_N} I_n \end{bmatrix}$$

$$\mathcal{H} := \begin{bmatrix} \frac{1}{1+\rho d_1} I_{2n} & & \\ & \ddots & \\ & & \frac{1}{1+\rho d_N} I_{2n} \end{bmatrix}$$

$$T_{\alpha\rho} := (1-\alpha) I - \alpha P + 2\alpha\rho P A \mathcal{H} A^\top.$$

Then, we introduce the stacking vectors $x^t \in \mathbb{R}^{Nn}$ and $z^t \in \mathbb{R}^{2n\mathbf{d}}$ defined as

$$x^t := \begin{bmatrix} x_1^t \\ \vdots \\ x_N^t \end{bmatrix}, \quad z^t := \begin{bmatrix} z_1^t \\ \vdots \\ z_N^t \end{bmatrix}.$$

The aggregate formulation of (10) reads as

$$x^{t+1} = x^t + \gamma\left(H\left(x^t + A_x^\top z^t\right) - x^t\right) - \gamma H\left(G(x^t) + A_\nabla^\top z^t\right) \tag{11a}$$

$$z^{t+1} = T_{\alpha\rho} z^t + 2\alpha\rho P A \mathcal{H} v(x^t), \tag{11b}$$

where we introduced the operators $G : \mathbb{R}^{Nn} \to \mathbb{R}^{Nn}$ and $v : \mathbb{R}^{Nn} \to \mathbb{R}^{2Nn}$ that, given any $x := \text{COL}(x_1, \dots, x_N) \in \mathbb{R}^{Nn}$ with $x_i \in \mathbb{R}^n$ for all $i \in \{1, \dots, N\}$, are defined as

$$G(x) := \begin{bmatrix} \nabla f_1(x_1) \\ \vdots \\ \nabla f_N(x_N) \end{bmatrix}, \quad v(x) := \begin{bmatrix} x_1 \\ \nabla f_1(x_1) \\ \vdots \\ x_N \\ \nabla f_N(x_N) \end{bmatrix}. \tag{12}$$

Fig. 1 reports a block diagram graphically describing (11).

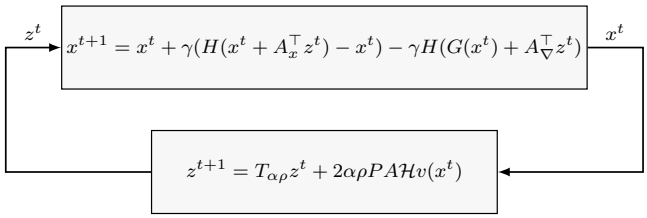

Fig. 1: Block diagram representing (11).

Now, we recall that the eigenvalues of $T_{\alpha\rho}$ are either equal to 1 or strictly inside the unitary circle [19]. Moreover, the eigenvalues in 1 are all semi-simple [19]. Based on these observations, in the next, we introduce a decomposition to isolate the marginally stable part of $T_{\alpha\rho}$. To this end, let $b \in \mathbb{N}$ be the dimension of the subspace $\mathcal{S}$ spanned by the eigenvectors associated to 1, $B \in \mathbb{R}^{2n\mathbf{d}\times b}$ be the matrix whose columns represent an orthonormal basis of $\mathcal{S}$, and $M \in \mathbb{R}^{2n\mathbf{d}\times n_d}$ be the matrix such that $B^\top M = 0$ and $M^\top M = I_{n_d}$, with $n_d := 2n\mathbf{d} - b$. Then, let $\bar{z} \in \mathbb{R}^b$ and $z_\perp \in \mathbb{R}^{n_d}$ be defined as

$$\begin{bmatrix} \bar{z} \\ z_\perp \end{bmatrix} := \begin{bmatrix} B^\top \\ M^\top \end{bmatrix} z. \tag{13}$$

By the construction of B, it holds

$$B^\top T_{\alpha\rho} B = I_b, \tag{14}$$

which allows us to claim that system (11) in the coordinates (22) and (13) reads as

$$\begin{aligned} x^{t+1} &= x^t + \gamma \left(H\left(x^t + A_x^\top M z_\perp^t\right) - x^t\right) \\ &\quad - \gamma H\left(G(x^t) + A_\nabla^\top M z_\perp^t\right) && \text{(15a)} \\ \bar{z}^{t+1} &= \bar{z}^t && \text{(15b)} \\ z_\perp^{t+1} &= \bar{T}_{\alpha\rho} z_\perp^t + 2\alpha\rho M^\top P A \mathcal{H} v(x^t), && \text{(15c)} \end{aligned}$$

where we used $\bar{T}_{\alpha\rho} := M^\top T_{\alpha\rho} M$ and the results

$$A_x^\top B = 0, \quad A_\nabla^\top B = 0, \quad B^\top P A \mathcal{H} A^\top = 0. \tag{16}$$

Notably, the variable $\bar{z}^t$ does not affect the other updates of (15) (and it holds $\bar{z}^t \equiv \bar{z}^0$ for all $t \geq 0$). Thus, we ignore it in the analysis considering the equivalent system

$$\begin{aligned} x^{t+1} &= x^t + \gamma \left(H\left(x^t + A_x^\top M z_\perp^t\right) - x^t\right) \\ &\quad - \gamma H\left(G(x^t) + A_\nabla^\top M z_\perp^t\right) && \text{(17a)} \\ z_\perp^{t+1} &= \bar{T}_{\alpha\rho} z_\perp^t + 2\alpha\rho M^\top P A \mathcal{H} v(x^t). && \text{(17b)} \end{aligned}$$

We interpret (17) as a singularly perturbed system (see, e.g., [28, Ch. 11]), i.e., the interconnection between the slow subsystem (17a) and the fast one (17b). Indeed, we can arbitrarily reduce the variation speed of (17a) through the parameter γ, while the fast scheme (17b) has an equilibrium parametrized in the slow state x through the function $z_\perp^{\text{eq}} : \mathbb{R}^{Nn} \to \mathbb{R}^{n_d}$ defined as

$$z_\perp^{\text{eq}}(x) := 2\alpha\rho(I - \bar{T}_{\alpha\rho})^{-1} M^\top P A \mathcal{H} v(x). \tag{18}$$

Moreover, given any $x \in \mathbb{R}^{Nn}$, it is possible to show that

$$H A_x^\top M z_\perp^{\text{eq}}(x) = \frac{\mathbf{1}_{N,n}\mathbf{1}_{N,n}^\top}{N} x - Hx \tag{19a}$$

$$H A_\nabla^\top M z_\perp^{\text{eq}}(x) = \frac{\mathbf{1}_{N,n}\mathbf{1}_{N,n}^\top}{N} G(x) - HG(x). \tag{19b}$$

With these results at hand, we introduce the error coordinates

$$\tilde{x} := x - \mathbf{1}_{N,n} x^\star, \quad \tilde{z}_\perp := z_\perp - z_\perp^{\text{eq}}(x), \tag{20}$$

which allows us to rewrite (17) as

$$\begin{aligned} \tilde{x}^{t+1} &= \tilde{x}^t - \gamma \tfrac{\mathbf{1}_{N,n}\mathbf{1}_{N,n}^\top}{N} G(\tilde{x}^t + \mathbf{1}_{N,n} x^\star) \\ &\quad + \gamma \left(\tfrac{\mathbf{1}_{N,n}\mathbf{1}_{N,n}^\top}{N} - I\right) \tilde{x}^t + \gamma H (A_x + A_\nabla)^\top M \tilde{z}_\perp^t && \text{(21a)} \\ \tilde{z}_\perp^{t+1} &= \bar{T}_{\alpha\rho} \tilde{z}_\perp^t - z_\perp^{\text{eq}}(\tilde{x}^{t+1} + x^\star) + z_\perp^{\text{eq}}(\tilde{x}^t + x^\star). && \text{(21b)} \end{aligned}$$

Now, let us introduce the matrix $R \in \mathbb{R}^{Nn\times(N-1)n}$ whose columns span the space orthogonal to the one of $\mathbf{1}_{N,n}$ and such that $R^\top R = I$. Then, let us employ the matrices $\mathbf{1}_{N,n}$ and R to decompose $\tilde{x}^t$ into $\mu^t \in \mathbb{R}^n$ and $x_\perp^t \in \mathbb{R}^{(N-1)n}$ according to

$$\mu^t := \frac{\mathbf{1}_{N,n}^\top}{N} \tilde{x}^t, \quad x_\perp^t := R^\top \tilde{x}^t. \tag{22}$$

By using the coordinates (22), $\mathbf{1}_{N,n} \in \ker(I - \frac{\mathbf{1}_{N,n}\mathbf{1}_{N,n}^\top}{N})$, and $R^\top \frac{\mathbf{1}_{N,n}\mathbf{1}_{N,n}^\top}{N} = 0$, we reformulate (21) as

$$\begin{aligned} \mu^{t+1} &= \mu^t - \gamma \frac{\mathbf{1}_{N,n}^\top}{N} G(\mathbf{1}_{N,n}\mu^t + R x_\perp^t + \mathbf{1}_{N,n} x^\star) \\ &\quad + \gamma \frac{\mathbf{1}_{N,n}^\top}{N} H (A_x^\top + A_\nabla^\top) M \tilde{z}_\perp^t && \text{(23a)} \\ x_\perp^{t+1} &= (1-\gamma) x_\perp^t + \gamma R^\top H (A_x^\top + A_\nabla^\top) M \tilde{z}_\perp^t && \text{(23b)} \\ \tilde{z}_\perp^{t+1} &= \bar{T}_{\alpha\rho} \tilde{z}_\perp^t - z_\perp^{\text{eq}}(\mathbf{1}_{N,n}\mu^{t+1} + x_\perp^{t+1} + x^\star) \\ &\quad + z_\perp^{\text{eq}}(\mathbf{1}_{N,n}\mu^t + R^\top x_\perp^t + x^\star). && \text{(23c)} \end{aligned}$$

For the sake of compactness, let us introduce $\Delta G : \mathbb{R}^n \times \mathbb{R}^{(N-1)n} \to \mathbb{R}^n$, $\ell : \mathbb{R}^{n_d} \to \mathbb{R}^{Nn}$, and $\Delta z_\perp^{\text{eq}} : \mathbb{R}^n \times \mathbb{R}^{(N-1)n} \to \mathbb{R}^{n_d}$ defined as

$$\begin{aligned} \Delta G\left(\mu^t, x_\perp^t\right) &:= -\frac{\mathbf{1}_{N,n}^\top}{N} G(\mathbf{1}_{N,n}\mu^t + R x_\perp^t + \mathbf{1}_{N,n} x^\star) \\ &\quad + \frac{\mathbf{1}_{N,n}^\top}{N} G(\mathbf{1}_{N,n}\mu^t + \mathbf{1}_{N,n} x^\star) \\ \ell(\tilde{z}_\perp^t) &:= H(A_x^\top + A_\nabla^\top) M \tilde{z}_\perp^t \\ \Delta z_\perp^{\text{eq}}(\mu^t, x_\perp^t) &:= -z_\perp^{\text{eq}}(\mathbf{1}_{N,n}\mu^{t+1} + R x_\perp^{t+1} + x^\star) \\ &\quad + z_\perp^{\text{eq}}(\mathbf{1}_{N,n}\mu^t + R x_\perp^t + x^\star). \end{aligned}$$

By using this notation and adding and subtracting $\frac{\gamma}{N}\nabla f(\mu^t + x^\star)$ into (23a), we compactly rewrite system (23) as

$$\begin{aligned} \mu^{t+1} &= \mu^t - \frac{\gamma}{N}\nabla f(\mu^t + x^\star) + \gamma \Delta G\left(\mu^t, x_\perp^t\right) \\ &\quad + \gamma \frac{\mathbf{1}_{N,n}^\top}{N} \ell(\tilde{z}_\perp^t) && \text{(24a)} \\ x_\perp^{t+1} &= (1-\gamma) x_\perp^t + \gamma R^\top \ell(\tilde{z}_\perp^t) && \text{(24b)} \\ \tilde{z}_\perp^{t+1} &= \bar{T}_{\alpha\rho} \tilde{z}_\perp^t + \Delta z_\perp^{\text{eq}}(\mu^t, x_\perp^t). && \text{(24c)} \end{aligned}$$

We note that (24a) reads as the gradient method applied to (1) (i) written in error coordinates with respect to $x^\star$ and (ii) perturbed via the vanishing terms $\gamma\Delta G\left(\mu^t, x_\perp^t\right)$ and $\gamma\frac{\mathbf{1}_{N,n}^\top}{N}\ell(\tilde{z}_\perp^t)$. Moreover, the subsystem (24b) can be seen as a linear system with Schur state matrix $(1-\gamma)I_{(N-1)n}$ perturbed by the vanishing term $\gamma R^\top \ell(\tilde{z}_\perp^t)$. Finally, system (24c) is a linear system with Schur state matrix $\bar{T}_{\alpha\rho}$ and a perturbation term $\Delta z_\perp^{\text{eq}}(\mu^t, x_\perp^t)$ given by the variations of the slow states μ^t and $x_\perp^t$ and, thus, that can be made arbitrarily small through the parameter γ. Therefore, based on these observations, we conclude that this system reformulation paves the way to prove Theorem III.1 but, for the sake of space, we will provide this proof in a forthcoming document.

V. Numerical Simulations

In this section, we perform some numerical simulations to compare ADMM-Tracking Gradient with the well-known Gradient Tracking algorithm [4]–[14], which is described by the local update equations

$$x_i^{t+1} = \sum_{j\in\mathcal{N}_i} w_{ij}x_j^t - \gamma s_i^t \tag{25a}$$

$$s_i^{t+1} = \sum_{j\in\mathcal{N}_i} w_{ij}s_j^t + \nabla f_i(x_i^{t+1}) - \nabla f_i(x_i^t), \tag{25b}$$

where $\gamma > 0$ is a parameter called step-size, $x_i^t \in \mathbb{R}^n$ is the local solution estimate, $s_i^t \in \mathbb{R}^n$ is so-called tracker, while each w_{ij} matches the graph $\mathcal{G}$, namely, $w_{ij} > 0$ whenever $(j,i) \in \mathcal{E}$ and $w_{ij} = 0$ otherwise. First, we consider a quadratic scenario and, then, we address a logistic regression problem in the case in which some errors affect the update equations of the algorithms. All the simulations are performed by considering networks of $N = 10$ agents and an underlying randomly generated Erdős-Rényi graph with connectivity parameter 0.1. In all the simulations, we run our schemes by randomly selecting $x_i^0 \in \mathbb{R}^n$ and $z_{ij}^0 \in \mathbb{R}^{2n}$ for all $i \in \{1,\dots,N\}$ and $j \in \mathcal{N}_i$. As for Gradient Tracking, we run it by setting the same x_i^0 used to perform our schemes, while, as prescribed in [4]–[6], we set $s_i^0 = \nabla f_i(x_i^0)$ for all $i \in \{1,\dots,N\}$. The comparisons are done in terms of the convergence of the norm $\|e^t\|$, in which $e^t \in \mathbb{R}^{Nn}$ denotes the optimality error $e^t := x^t - \mathbf{1}_{N,n}x^\star$.

A. Quadratic Scenario

In this section, we consider a quadratic setup described by

$$\min_{x\in\mathbb{R}^n} \sum_{i=1}^N \left(\tfrac{1}{2}x^\top Q_i x + r_i^\top x\right),$$

where $Q_i \in \mathbb{R}^{n\times n}$ and $r_i \in \mathbb{R}^n$. Moreover, it holds $Q_i = Q_i^\top > 0$ for all $i \in \{1,\dots,N\}$ and, thus, the problem is strongly convex. We set $n = 2$ and, for all $i \in \{1,\dots,N\}$, we randomly generate each matrix Q_i so that all its eigenvalues belong to the interval $[1,5]$, while the components of each vector r_i are randomly generated from the interval $[-10,20]$ with a uniform distribution. Since the quadratic scenario gives rise to algorithm updates with linear form, we choose the parameters of the algorithms as the ones minimizing the largest eigenvalue of the matrices describing the algorithms in error coordinates. Specifically, we choose $\gamma = 0.865$, $\rho = 0.528$, $\alpha = 0.8924$ for ADMM-Tracking Gradient, while we set $\gamma = 0.0627$ for Gradient Tracking. Fig. 2 reports the simulation results and shows that ADMM-Tracking Gradient outperforms Gradient Tracking in terms of convergence rate.

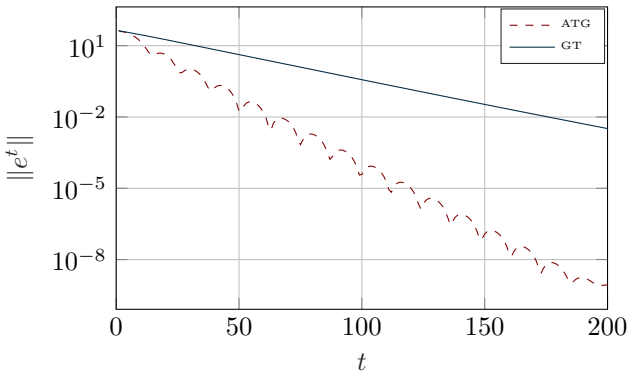

Fig. 2: Quadratic setup: comparison between ADMM-Tracking Gradient (ATG) and Gradient Tracking (GT).

B. Logistic Regression Scenario with Errors

In this section, we consider a logistic regression scenario. A network of agents aims to cooperatively train a linear classifier for a set of points in a given feature space. Each agent i is equipped with $m_i \in \mathbb{N}$ points $p_{i,1},\dots,p_{i,m_i} \in \mathbb{R}^{n-1}$ with binary labels $l_{i,k} \in \{-1,1\}$ for all $k \in \{1,\dots,m_i\}$. The problem consists of building a linear classification model from these points solving the minimization problem described by

$$\min_{w\in\mathbb{R}^{n-1},b\in\mathbb{R}} \sum_{i=1}^N\sum_{k=1}^{m_i} \log\left(1+e^{-l_{i,k}(w^\top p_{i,k}+b)}\right) + \frac{C}{2}\left\|\begin{bmatrix}w\\b\end{bmatrix}\right\|^2,$$

where $w \in R^d$ and $b \in \mathbb{R}$ are the optimization variables and $C > 0$ is the so-called regularization parameter. Notice that the presence of the regularization makes the cost function strongly convex. We set $n = 2$, $m_i = 10$ for all $i \in \{1,\dots,N\}$, and we randomly generate each $p_{i,k}$ and $l_{i,k}$ for all $i \in \{1,\dots,10\}$ and $k \in \{1,\dots,10\}$. We empirically tune the algorithms' parameters by choosing $\alpha = 0.9$ and $\rho = 0.9$ for ADMM-Tracking Gradient, while we set $\gamma = 0.1$ for both ADMM-Tracking Gradient and Gradient Tracking. Moreover, we consider the presence of fixed errors affecting the update equations of the algorithms. In detail, we consider an additive error $v_n = \epsilon 1_n$ affecting the local update equations (10a), (25a), and (25b), and, analogously, an additive error $v_{2nd_i} = \epsilon 1_{2nd_i}$ affecting the update (10b), where $\epsilon > 0$ denotes the amplitudes of these errors. Fig. 3 compares the algorithms' performance for different values of ϵ. Differently from Gradient Tracking, we note that ADMM-Tracking Gradient does not diverge and, hence, is robust with respect to these errors.

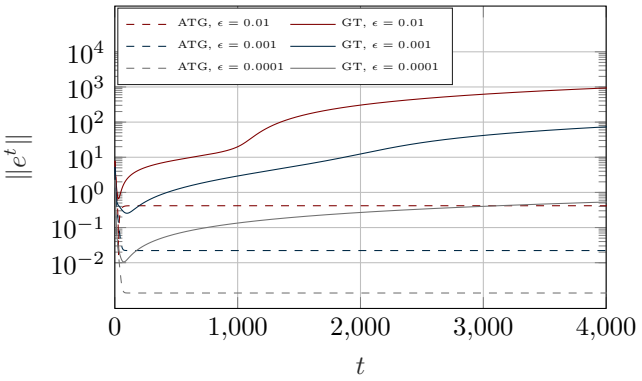

Fig. 3: Logistic regression scenario: comparison between ADMM-Tracking Gradient (ATG) and Gradient Tracking (GT) with fixed errors.

VI. Conclusions

In this paper, we proposed a novel distributed algorithm for consensus optimization. In detail, we designed the algorithm by interpreting the dynamic consensus problem as an additional optimization problem addressed through the ADMM. We interlaced the obtained scheme with suitable local, proportional actions giving rise to the novel distributed algorithm that we named ADMM-Tracking Gradient. In the case of strongly convex problems, we proved the linear convergence of the algorithm through a system theoretical analysis. Finally, we tested our method to show its effectiveness.

References

[1] G. Notarstefano, I. Notarnicola, and A. Camisa, "Distributed optimization for smart cyber-physical networks," *Foundations and Trends® in Systems and Control*, vol. 7, no. 3, pp. 253–383, 2019.

[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] T. Yang, X. Yi, J. Wu, Y. Yuan, D. Wu, Z. Meng, Y. Hong, H. Wang, Z. Lin, and K. H. Johansson, "A survey of distributed optimization," *Annual Reviews in Control*, vol. 47, pp. 278–305, 2019.

[4] 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.

[5] 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.

[6] C. Xi, R. Xin, and U. A. Khan, "ADD-OPT: Accelerated distributed directed optimization," *IEEE Transactions on Automatic Control*, vol. 63, no. 5, pp. 1329–1339, 2017.

[7] G. Carnevale, I. Notarnicola, L. Marconi, and G. Notarstefano, "Triggered gradient tracking for asynchronous distributed optimization," *Automatica*, vol. 147, p. 110726, 2023.

[8] P. Di Lorenzo and G. Scutari, "NEXT: In-network nonconvex optimization," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, no. 2, pp. 120–136, 2016.

[9] A. Daneshmand, G. Scutari, and V. Kungurtsev, "Second-order guarantees of distributed gradient algorithms," *SIAM Journal on Optimization*, vol. 30, no. 4, pp. 3029–3068, 2020.

[10] G. Carnevale and G. Notarstefano, "Nonconvex distributed optimization via lasalle and singular perturbations," *IEEE Control Systems Letters*, vol. 7, pp. 301–306, 2022.

[11] G. Carnevale, F. Farina, I. Notarnicola, and G. Notarstefano, "GTAdam: Gradient tracking with adaptive momentum for distributed online optimization," *IEEE Transactions on Control of Network Systems*, 2022.

[12] K. Yuan, W. Xu, and Q. Ling, "Can Primal Methods Outperform Primal-Dual Methods in Decentralized Dynamic Optimization?" *IEEE Transactions on Signal Processing*, vol. 68, pp. 4466–4480, 2020.

[13] M. Bin, I. Notarnicola, L. Marconi, and G. Notarstefano, "A system theoretical perspective to gradient-tracking algorithms for distributed quadratic optimization," in *2019 IEEE 58th Conference on Decision and Control (CDC)*. IEEE, 2019, pp. 2994–2999.

[14] I. Notarnicola, M. Bin, L. Marconi, and G. Notarstefano, "The gradient tracking is a distributed integral action," *IEEE Transactions on Automatic Control*, 2023.

[15] M. Zhu and S. Martínez, "Discrete-time dynamic average consensus," *Automatica*, vol. 46, no. 2, pp. 322–329, 2010.

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

[17] M. Bin, I. Notarnicola, and T. Parisini, "Stability, linear convergence, and robustness of the wang-elia algorithm for distributed consensus optimization," in *2022 IEEE 61st Conference on Decision and Control (CDC)*. IEEE, 2022, pp. 1610–1615.

[18] N. Bastianello and R. Carli, "Admm for dynamic average consensus over imperfect networks," *IFAC-PapersOnLine*, vol. 55, no. 13, pp. 228–233, 2022.

[19] 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.

[20] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers," *Foundations and TrendsÂ® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2010.

[21] J. F. C. Mota, J. M. F. Xavier, P. M. Q. Aguiar, and M. PÃ¼schel, "D-ADMM: A Communication-Efficient Distributed Algorithm For Separable Optimization," *IEEE Transactions on Signal Processing*, vol. 61, no. 10, pp. 2718–2723, May 2013.

[22] Z. Peng, Y. Xu, M. Yan, and W. Yin, "ARock: an Algorithmic Framework for Asynchronous Parallel Coordinate Updates," *SIAM Journal on Scientific Computing*, vol. 38, no. 5, pp. A2851–A2879, Jan. 2016.

[23] A. Makhdoumi and A. Ozdaglar, "Convergence Rate of Distributed ADMM Over Networks," *IEEE Transactions on Automatic Control*, vol. 62, no. 10, pp. 5082–5095, Oct. 2017.

[24] N. Bof, R. Carli, and L. Schenato, "Is ADMM always faster than Average Consensus?" *Automatica*, vol. 91, pp. 311–315, May 2018.

[25] S. Hosseini, A. Chapman, and M. Mesbahi, "Online distributed ADMM via dual averaging," in *53rd IEEE Conference on Decision and Control*. Los Angeles, CA, USA: IEEE, Dec. 2014, pp. 904–909.

[26] Q. Ling and A. Ribeiro, "Decentralized Dynamic Optimization Through the Alternating Direction Method of Multipliers," *IEEE Transactions on Signal Processing*, vol. 62, no. 5, pp. 1185–1197, Mar. 2014.

[27] N. Bastianello, A. Simonetto, and R. Carli, "Distributed Prediction-Correction ADMM for Time-Varying Convex Optimization," in *2020 54th Asilomar Conference on Signals, Systems, and Computers*, Nov. 2020, pp. 47–52.

[28] H. K. Khalil, "Nonlinear systems," *Upper Saddle River*, 2002.