Estimation Network Design framework for efficient distributed optimization

Bianchi, Mattia; Grammatico, Sergio

DOI
10.1109/CDC56724.2024.10885996

Publication date
2025

Document Version
Final published version

Published in
Proceedings of the IEEE 63rd Conference on Decision and Control, CDC 2024

2024 IEEE 63rd Conference on Decision and Control (CDC)
December 16-19, 2024. MiCo, Milan, Italy

Estimation Network Design framework for efficient distributed optimization

Mattia Bianchi and Sergio Grammatico

2024 IEEE 63rd Conference on Decision and Control (CDC) | 979-8-3503-1633-9/24/$31.00 ©2024 IEEE | DOI: 10.1109/CDC56724.2024.10885996

***Abstract*— Distributed decision problems feature a group of agents that can only communicate over a peer-to-peer network, without a central memory. In applications such as network control and data ranking, each agent is only affected by a small portion of the decision vector: this sparsity is typically ignored in distributed algorithms, while it could be leveraged to improve efficiency and scalability. To address this issue, our recent paper [1] introduces Estimation Network Design (END), a graph theoretical language for analysis and design of distributed iterations. END methods can be tuned to exploit the sparsity of specific problem instances, reducing communication overhead and minimizing redundancy, yet without requiring case-by-case convergence analysis. In this paper, we showcase the flexibility of END in the context of distributed optimization. In particular, we study the sparsity-aware version of many established algorithms, including ADMM, AugDGM and Push-Sum DGD. Simulations on an estimation problem in sensor networks demonstrate that END algorithms can boost convergence speed and greatly reduce the communication cost.**

I. Introduction

Modern big data optimization problems in network control and machine learning are typically *partially separable* [2]: the cost function is the sum of N individual costs, each depending only on a small portion of the overall optimization variable. This structure is widely exploited in *parallel* algorithms [2], [3] – where multiple distinct processors share the computation cost, but having access to a common memory. Yet, this is not the case for *distributed* scenarios – where the processors (or agents) are constrained to communicate uniquely with some neighbors over a communication network. In fact, most distributed optimization methods entail the agents reaching consensus on the entire optimization vector [4], [5], even when each agent eventually only uses a few components of the solution, as in resource allocation and network control [6]. This may result in prohibitive memory and communication requirements, and in poor scalability if the decision vector grows with the network size.

Efficient solutions are known for problems where the local cost functions only directly couple each agent to its communication neighbors [6]–[9]. Notably, this requires that the *communication* graph matches the *interference* graph (describing the coupling among the agents in the cost or constraints). This is possible if the communication graph can be freely chosen [10]; but it is not usually the case for wireless and ad-hoc networks, where remote nodes cannot be connected directly. We are interested in the latter case, where the communication graph is given and cannot be designed.

M. Bianchi is with the Automatic Control Laboratory, ETH Zürich, Switzerland (mbianch@ethz.ch). S. Grammatico is with the Delft Center for Systems and Control, TU Delft, The Netherlands (s.grammatico@tudelft.nl). This work is supported by NWO (OMEGA 613.001.702), by ERC (COSMOS 802348), by ETH Zürich funds.

In this setup, partially-separable problems were addressed via distributed dual methods, by Mota et al. [11] and later by Alghunaim et al. [12], [13]: in this approach, each component of the optimization variable is estimated by a suitably chosen cluster of agents only. Unfortunately, the dual formulation is only effective over undirected communication networks. Other works [14], [15] rely on the concept of *locality*, which result in improved efficiency, but at the cost of accuracy.

To deal with these challenges, in our recent work [1], we introduced END, a graph-theoretic language to describe how the estimates of any variable of interest (e.g., optimization vector, dual multipliers, cost gradient) are allocated and combined among the agents in a generic distributed algorithm. END allows assigning the estimate of each component of the variable of interest to a subset of the agents, according to the sparsity structure of a given problem and given communication graph – without resorting to a case-by-case convergence analysis. Leveraging sparsity is especially convenient in repeated or time-varying problems (e.g., distributed estimation and model predictive control (MPC) [11]), where the one time-cost of efficiently assigning the estimates yields improved iterative online performance.

Contributions: In this paper, we tailor the END framework, developed in [1] for distributed games, to the case of distributed optimization problems. Our analysis unifies and generalizes several recent approaches. For the case of dual algorithms, our setup retrieves the formulation in [11], [13] (see Proposition 1). Here we present a novel sparsity-aware ADMM, but one can obtain the END version of virtually any dual algorithm (Section III-A). Further, compared to [11], [13], our framework has broader applicability:

(i) *it can be used for primal methods.* To illustrate, we present the END version of the ABC method [5], encompassing many established algorithms. As an example, we derive a gradient-tracking iteration where each agent only needs to estimate a fraction of the cost gradient – the END counterpart of AugDGM [16] (Section III-B);
(ii) *it works on directed graphs.* Specifically, we present the sparsity-aware version of the Push-Sum DGD algorithm [17], that is guaranteed to converge over time-varying and column stochastic graphs (Section III-C).

It will be clear that, thanks to our powerful stacked notation, the analysis of END algorithms presents little complication compared to their sparsity-unaware counterparts. Yet, the impact in terms of flexibility and performance is remarkable. We illustrate numerically this point on an estimation problem for wireless sensor network, where END decreases the communication cost by more than 90% (Section IV).

A. Background and notation

1) Basic notation: $\mathbb{N}$ is the set of natural numbers, including 0. $\mathbb{R}$ ($\mathbb{R}_{\geq 0}$) is the set of (nonnegative) real numbers. $\mathbf{0}_q \in \mathbb{R}^q$ ($\mathbf{1}_q \in \mathbb{R}^q$) is a vector with all elements equal to 0 (1); $\mathrm{I}_q \in \mathbb{R}^{q\times q}$ is an identity matrix; the subscripts may be omitted when there is no ambiguity. e_i denotes a vector of appropriate dimension with i-th element equal to 1 and all other elements equal to 0. For a matrix $\mathrm{A} \in \mathbb{R}^{p\times q}$, $[\mathrm{A}]_{i,j}$ is the element on row i and column j; $\mathrm{null}(\mathrm{A}) \coloneqq \{x \in \mathbb{R}^q \mid \mathrm{A}x = \mathbf{0}_n\}$ and $\mathrm{range}(\mathrm{A}) \coloneqq \{v \in \mathbb{R}^p \mid v = \mathrm{A}x, x \in \mathbb{R}^q\}$. If $\mathrm{A} = \mathrm{A}^\top \in \mathbb{R}^{q\times q}$, $\boldsymbol{\lambda}_{\min}(\mathrm{A}) =: \boldsymbol{\lambda}_1(\mathrm{A}) \leq \cdots \leq \boldsymbol{\lambda}_q(\mathrm{A}) =: \boldsymbol{\lambda}_{\max}(\mathrm{A})$ denote its eigenvalues. $\mathrm{diag}(\mathrm{A}_1, \ldots, \mathrm{A}_N)$ is the block diagonal matrix with $\mathrm{A}_1, \ldots, \mathrm{A}_N$ on its diagonal. Given N vectors $x_1, \ldots, x_N$, $\mathrm{col}(x_1, \ldots, x_N) \coloneqq [x_1^\top \ldots x_N^\top]^\top$. $\otimes$ denotes the Kronecker product. Given a positive definite matrix $\mathrm{Q} \in \mathbb{R}^{q\times q} \ni \mathrm{Q} \succ 0$, $\langle x \mid y\rangle_\mathrm{Q} = x^\top \mathrm{Q} y$ os the Q-weighted inner product, $\|\cdot\|_\mathrm{Q}$ is the associated norm; we omit the subscripts if $\mathrm{Q} = \mathrm{I}$. Given a function $\psi : \mathbb{R}^q \to \overline{\mathbb{R}} \coloneqq \mathbb{R} \cup \{\infty\}$, its set-valued subdifferential operator is denoted by $\partial\psi : \mathbb{R}^q \rightrightarrows \mathbb{R}^q : x \mapsto \{v \in \mathbb{R}^q \mid \psi(z) \geq \psi(x) + \langle v \mid z - x\rangle, \forall z \in \mathbb{R}^q\}$.

2) Graph theory: A (directed) graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consists of a nonempty set of vertices (or nodes) $\mathcal{V} = \{1, 2, \ldots, V\}$ and a set of edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. We denote by $\mathcal{N}(v) \coloneqq \{u \mid (u, v) \in \mathcal{E}\}$ and $\overline{\mathcal{N}}(v) \coloneqq \{u \mid (v, u) \in \mathcal{E}\}$ the set of in-neighbors (or simply neighbors) and out-neighbors of vertex $v \in \mathcal{V}$, respectively. A path from $v_1 \in \mathcal{V}$ to $v_T \in \mathcal{V}$ of length T is a sequence of vertices $(v_1, v_2, \ldots, v_T)$ such that $(v_t, v_{t+1}) \in \mathcal{E}$ for all $t = 1, \ldots, T-1$. $\mathcal{G}$ is strongly connected if there exist a path from u to v, for all $u, v \in \mathcal{V}$; in case $\mathcal{G}$ is undirected, namely if $(u, v) \in \mathcal{E}$ whenever $(v, u) \in \mathcal{E}$, we simply say that $\mathcal{G}$ is connected. The restriction of the graph $\mathcal{G}$ to a set of vertices $\mathcal{V}^\mathrm{A} \subseteq \mathcal{V}$ is defined as $\mathcal{G}|_{\mathcal{V}^\mathrm{A}} \coloneqq (\mathcal{V}^\mathrm{A}, \mathcal{E} \cap (\mathcal{V}^\mathrm{A} \times \mathcal{V}^\mathrm{A}))$. We also write $\mathcal{G} = (\mathcal{V}^\mathrm{A}, \mathcal{V}^\mathrm{B}, \mathcal{E})$ to highlight that $\mathcal{G}$ is bipartite, namely $\mathcal{V} = \mathcal{V}^\mathrm{A} \cup \mathcal{V}^\mathrm{B}$ and $\mathcal{E} \subseteq \mathcal{V}^\mathrm{A} \times \mathcal{V}^\mathrm{B}$. We may associate to $\mathcal{G}$ a weight matrix $\mathrm{W} \in \mathbb{R}^{V\times V}$ compliant with $\mathcal{G}$, namely $w_{u,v} \coloneqq [\mathrm{W}]_{u,v} > 0$ if $(v, u) \in \mathcal{E}$, $w_{u,v} = 0$ otherwise. $\mathcal{G}$ is unweighted if $w_{u,v} = 1$ if $(v, u) \in \mathcal{E}$. Given two graphs $\mathcal{G}^\mathrm{A} = (\mathcal{V}^\mathrm{A}, \mathcal{E}^\mathrm{A})$ and $\mathcal{G}^\mathrm{B} = (\mathcal{V}^\mathrm{B}, \mathcal{E}^\mathrm{B})$, we write $\mathcal{G}^\mathrm{A} \subseteq \mathcal{G}^\mathrm{B}$ if $\mathcal{G}^\mathrm{A}$ is a subgraph of $\mathcal{G}^\mathrm{B}$, i.e., if $\mathcal{V}^\mathrm{A} \subseteq \mathcal{V}^\mathrm{B}$ and $\mathcal{E}^\mathrm{A} \subseteq \mathcal{E}^\mathrm{B}$; we define $\mathcal{G}^\mathrm{A} \bigcup \mathcal{G}^\mathrm{B} \coloneqq (\mathcal{V}^\mathrm{A} \cup \mathcal{V}^\mathrm{B}, \mathcal{E}^\mathrm{A} \cup \mathcal{E}^\mathrm{B})$. A time-varying graph $(\mathcal{G}^k)_{k\in\mathbb{N}}$, $\mathcal{G}^k = (\mathcal{V}, \mathcal{E}^k)$ is Q-strongly connected if $\bigcup_{t=kQ}^{(k+1)Q-1} \mathcal{G}^t$ is strongly connected for all $k \in \mathbb{N}$.

II. END for distributed optimization

We first recall the general END framework [1], that describes the information structure in any distributed algorithm. It is characterized by:

1) a **set of agents** $\mathcal{I} \coloneqq \{1, 2, \ldots, N\}$;
2) a given (directed) **communication graph** $\mathcal{G}^\mathrm{C} = (\mathcal{I}, \mathcal{E}^\mathrm{C})$, over which the agents can exchange information: agent i can receive data from agent j if and only if $j \in \mathcal{N}^\mathrm{C}(i)$;
3) a **variable of interest** $y \in \mathbb{R}^{n_y}$, partitioned as $y = \mathrm{col}((y_p)_{p\in\mathcal{P}})$, $\mathcal{P} \coloneqq \{1, \ldots, P\}$, $y_p \in \mathbb{R}^{n_{y_p}}$;
4) a given **interference graph** $\mathcal{G}^\mathrm{I} = (\mathcal{P}, \mathcal{I}, \mathcal{E}^\mathrm{I})$, $\mathcal{E}^\mathrm{I} \subseteq \mathcal{P} \times \mathcal{I}$, specifying which components of y are indispensable for each agent: $p \in \mathcal{N}^\mathrm{I}(i)$ means that agent i needs (an estimate of) y_p to perform some essential local computation;[1]
5) a bipartite **estimate graph** $\mathcal{G}^\mathrm{E} = (\mathcal{P}, \mathcal{I}, \mathcal{E}^\mathrm{E})$, $\mathcal{E}^\mathrm{E} \subseteq \mathcal{P} \times \mathcal{I}$. Since agents might be unable to access y, each agent estimates some of the components y_p's, as specified by the estimate graph: agent i keeps an estimate $\boldsymbol{y}_{i,p} \in \mathbb{R}^{n_{y_p}}$ of y_p if and only if $p \in \mathcal{N}^\mathrm{E}(i)$;
6) P directed **design graphs** $\{\mathcal{G}_p^\mathrm{D}\}_{p\in\mathcal{P}}$, $\mathcal{G}_p^\mathrm{D} = (\overline{\mathcal{N}}^\mathrm{E}(p), \mathcal{E}_p^\mathrm{D})$. $\mathcal{G}_p^\mathrm{D}$ describes how the agents that estimate y_p exchange their estimates: agent i can receive $\boldsymbol{y}_{j,p}$ from agent j if and only if $i \in \mathcal{N}_p^\mathrm{D}(j)$;

Specifically, in this paper, we apply the END framework to the distributed optimization problems

$$\min_{y\in\mathbb{R}^{n_y}} \ \textstyle\sum_{i\in\mathcal{I}} f_i(y), \tag{1}$$

where $f_i : \mathbb{R}^{n_y} \to \overline{\mathbb{R}}$ is a private cost function of agent i. We choose the variable of interest in the END framework to coincide with the optimization variable[2]. Hence, we partition the optimization variable as $y = \mathrm{col}((y_p)_{p\in\mathcal{P}})$.

The common approach [17], [18] to solve (1) over a communication network $\mathcal{G}^\mathrm{C}$ is to assign to each agent $i \in \mathcal{I}$ a copy $\tilde{\boldsymbol{y}}_i \coloneqq \mathrm{col}((\boldsymbol{y}_{i,p})_{p\in\mathcal{P}}) \in \mathbb{R}^{n_y}$ of the whole decision variable and to let the agents exchange their estimates with every neighbor over $\mathcal{G}^\mathrm{C}$; in END notation, we write this as[3]

$$\mathcal{E}^\mathrm{E} = \mathcal{P} \times \mathcal{I}, \qquad \mathcal{G}_p^\mathrm{D} = \mathcal{G}^\mathrm{C} \ (\forall p \in \mathcal{P}). \tag{2}$$

Yet, in several applications, like network control and data ranking [2], each cost function f_i depends only on some of the components of y, as specified by an interference graph $\mathcal{G}^\mathrm{I}$: f_i depends on y_p if and only if $p \in \mathcal{N}^\mathrm{I}(i) \subseteq \mathcal{P}$. With some abuse of notation, we highlight this fact by writing

$$f_i(y) = f_i((y_p)_{p\in\mathcal{N}^\mathrm{I}(i)}). \tag{3}$$

Clearly, the standard choice (2) for the graphs $\mathcal{G}^\mathrm{E}$ and $\{\mathcal{G}_p^\mathrm{D}\}_{p\in\mathcal{P}}$ does not take advantage of the structure in (3). In fact, agent i only needs $(y_p)_{p\in\mathcal{N}^\mathrm{I}(i)}$ to evaluate (the gradient of) its local cost f_i; storing a copy of the whole vector y could be unnecessary and inefficient – especially if $\mathcal{G}^\mathrm{I}$ is sparse and n_y is large.

A. Problem-dependent design, unified analysis

Let us emphasize that the graphs $\mathcal{G}^\mathrm{C}$ and $\mathcal{G}^\mathrm{I}$ are fixed and part of the problem formulation. In contrast, the graphs $\mathcal{G}^\mathrm{E}$ and $\{\mathcal{G}_p^\mathrm{D}\}_{p\in\mathcal{P}}$ are design choices, although with some constraints.

Standing Assumption 1: $\mathcal{G}^\mathrm{E}$ and $\mathcal{G}_p^\mathrm{D}$ are chosen such that $\mathcal{G}^\mathrm{I} \subseteq \mathcal{G}^\mathrm{E}$ and $\mathcal{G}_p^\mathrm{D} \subseteq \mathcal{G}^\mathrm{C}$ for all $p \in \mathcal{P}$. □

In particular, $\mathcal{G}^\mathrm{I} \subseteq \mathcal{G}^\mathrm{E}$ means that each agent estimates at least the components of y which are indispensable for local computation. Moreover, since the estimates are exchanged

[1] For ease of notation, assume $\overline{\mathcal{N}}^\mathrm{I}(p) \neq \varnothing$ for all $p \in \mathcal{P}$.

[2] Except for Section III-D, where we instead select the dual variable as variable of interest; see also [1] for different possible choices (e.g., aggregative values) in variational problems.

[3] In the following, we refer to (2) as the "standard" choice, as it is the most widely studied scenario. With $\mathcal{G}^\mathrm{c} = \mathcal{G}_p^\mathrm{D}$ we also imply $\mathrm{W}^\mathrm{C} = \mathrm{W}_p^\mathrm{D}$.

over $\mathcal{G}_p^{\mathrm{D}}$ and communication can only happen over $\mathcal{G}^{\mathrm{C}}$, it must hold that $\mathcal{G}_p^{\mathrm{D}} \subseteq \mathcal{G}^{\mathrm{C}}$. In addition, we will always need some level of connectedness for each graph $\mathcal{G}_p^{\mathrm{D}}$, to ensure that the agents can reach consensus on the estimates of y_p, as for instance in the following condition.

Assumption 1: For each $p \in \mathcal{P}$, $\mathcal{G}_p^{\mathrm{D}}$ is undirected and connected. □

Designing $\mathcal{G}^{\mathrm{E}}$ and $\{\mathcal{G}_p^{\mathrm{D}}\}_{p\in\mathcal{P}}$ to satisfy Standing Assumption 1 and Assumption 1 is not difficult if $\mathcal{G}^{\mathrm{C}}$ is itself undirected and connected: one trivial choice is (2). Yet, one wishes to also consider *efficiency* specifications (e.g., in terms of memory allocation, communication or bandwidth) by imposing extra (soft) constraints on $\mathcal{G}^{\mathrm{E}}$ and $\{\mathcal{G}_p^{\mathrm{D}}\}_{p\in\mathcal{P}}$. We present a simple design example in Figure 1 and refer to [1, App. A] for a detailed discussion. An optimal design is in general computationally expensive. Nonetheless, the performance advantages in terms of algorithm execution can be well worth the (one-time) cost of an efficient algorithm design, especially in repeated problems [11], [19] (where the same distributed problem is solved multiple times for different values of some parameters/measurements).

Further, while this design procedure is very problem and goal dependent, it does not affect the analysis of END algorithms. By simply postulating some connectedness property, as in Assumption 1, we can unify the convergence analysis of standard algorithms (that use (2)) with that of methods specifically devised for problems with unique sparsity.

B. END Notation

We next introduce the stacked END notation, crucial in our analysis. For all $p \in \mathcal{P}$, let $N_p := \left|\overline{\mathcal{N}}^{\mathrm{E}}(p)\right|$ be the number of copies of y_p. Recalling that that $\boldsymbol{y}_{i,p}$ is the estimate of y_p kept by agent i, we define:

$$\boldsymbol{y}_p := \mathrm{col}((\boldsymbol{y}_{i,p})_{i\in\overline{\mathcal{N}}^{\mathrm{E}}(p)}) \in \mathbb{R}^{N_p n_{y_p}}, \quad \forall p \in \mathcal{P}; \tag{4}$$

$$\boldsymbol{y} := \mathrm{col}((\boldsymbol{y}_p)_{p\in\mathcal{P}}) \in \mathbb{R}^{n_{\boldsymbol{y}}}, \tag{5}$$

where $n_{\boldsymbol{y}} := \sum_{p\in\mathcal{P}} N_p n_{y_p}$. Note that $\boldsymbol{y}_p$ collects all the copies of y_p kept by different agents. We denote

$$\mathbf{W}^{\mathrm{D}} := \mathrm{diag}((\mathrm{W}_p^{\mathrm{D}} \otimes \mathrm{I}_{n_{y_p}})_{p\in\mathcal{P}}), \tag{6}$$

where $\mathrm{W}_p^{\mathrm{D}}$ is the weight matrix of $\mathcal{G}_p^{\mathrm{D}}$. Let

$$\boldsymbol{\mathcal{C}}_p := \{\boldsymbol{y}_p \in \mathbb{R}^{N_p n_{y_p}} \mid \boldsymbol{y}_p = \mathbf{1}_{N_p} \otimes v, v \in \mathbb{R}^{n_{y_p}}\}, \tag{7}$$

be the consensus space for $\boldsymbol{y}_p$ (where all the estimates of y_p are equal); $\boldsymbol{\mathcal{C}} := \prod_{p\in\mathcal{P}} \boldsymbol{\mathcal{C}}_p$ be the overall consensus space;

$$\boldsymbol{\mathcal{C}}(y) := \mathrm{col}((\mathbf{1}_{N_p} \otimes y_p)_{p\in\mathcal{P}}). \tag{8}$$

For each $p \in \mathcal{P}$, for each $i \in \overline{\mathcal{N}}^{\mathrm{E}}(p)$, we denote by

$$i_p := \sum_{j\in\overline{\mathcal{N}}^{\mathrm{E}}(p),\, j\le i} 1 \tag{9}$$

the position of i in the ordered set of nodes $\overline{\mathcal{N}}^{\mathrm{E}}(p)$. For all $i \in \mathcal{N}^{\mathrm{E}}(p)$, we denote by $\mathrm{R}_{i,p} \in \mathbb{R}^{n_{y_p} \times N_p n_{y_p}}$ the matrix that selects $\boldsymbol{y}_{i,p}$ from $\boldsymbol{y}_p$, i.e., $\boldsymbol{y}_{i,p} = \mathrm{R}_{i,p}\boldsymbol{y}_p$.

Sometimes it is useful to define agent-wise quantities, indicated with a tilde. Let

$$\tilde{\boldsymbol{y}}_i := \mathrm{col}((\boldsymbol{y}_{i,p})_{p\in\mathcal{N}^{\mathrm{E}}(i)}), \ \tilde{\boldsymbol{y}} := \mathrm{col}((\tilde{\boldsymbol{y}}_i)_{i\in\mathcal{I}}) \in \mathbb{R}^{n_{\boldsymbol{y}}}, \tag{10}$$

where $\tilde{\boldsymbol{y}}_i$ collects all the estimates kept by agent i.

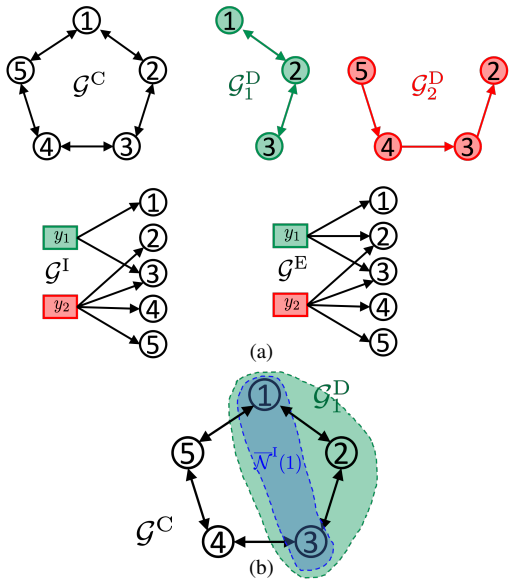

Fig. 1: (a) A simple example of END design from [1]. On the left, the given communication and interference graphs, with $\mathcal{I} = \{1,2,3,4,5\}$ and $\mathcal{P} = \{1,2\}$. On the right a possible choice for the design graphs and the corresponding estimate graphs.
(b) We focus on the design of $\mathcal{G}_1^{\mathrm{D}}$. The given efficiency specification is to minimize the number of copies of y_1 (i.e., the number of nodes in $\mathcal{G}_1^{\mathrm{D}}$), but provided that $\mathcal{G}_1^{\mathrm{D}}$ is connected and Standing Assumption 1 is met. A solution to this design problem can be obtained by solving an Unweighted Steiner Tree problem [20], for which distributed off-the-shelf algorithms are available [20]. Note that agent 2 has to estimate y_1 (i.e., $1 \in \mathcal{N}^{\mathrm{E}}(2)$), even though agent 2 is not directly affected by y_1 (i.e., $1 \notin \mathcal{N}^{\mathrm{I}}(2)$): otherwise, the information could not travel between nodes 1 and 3, which are not communication neighbors.

III. Distributed Optimization Algorithms

In this section, we leverage the END framework to extend several distributed optimization algorithms by exploiting partial coupling. We recall the cost-coupled problem in (1):

$$\min_{y\in\mathbb{R}^{n_y}} \ f(y) := \sum_{i\in\mathcal{I}} f_i((y_p)_{p\in\mathcal{N}^{\mathrm{I}}(i)}), \tag{11}$$

where f_i is a private cost function of agent i, and we choose the optimization variable $y = \mathrm{col}((y_p)_{p\in\mathcal{P}})$ as the variable of interest in the END; with some usual overloading, we write

$$f_i(y) = f_i((y_p)_{p\in\mathcal{N}^{\mathrm{E}}(i)}) = f_i((y_p)_{p\in\mathcal{N}^{\mathrm{I}}(i)}). \tag{12}$$

Let $\mathcal{Y}^\star$ be the solution set of (11), assumed to be nonempty.

A. END dual methods

Under Standing Assumption 1, we can recast (11) by introducing local estimates and consensus constraints.The following reformulation is not novel, and in fact it was employed for the dual methods in [11]–[13].

Proposition 1: Let Assumption 1 hold. Then, problem (11) is equivalent to:

$$\begin{cases} \min\limits_{\tilde{\boldsymbol{y}}\in\mathbb{R}^{n_{\boldsymbol{y}}}} \ \sum_{i\in\mathcal{I}} f_i(\tilde{\boldsymbol{y}}_i) = f_i((\boldsymbol{y}_{i,p})_{p\in\mathcal{N}^{\mathrm{E}}(i)}) \\ \quad \text{s.t. } \boldsymbol{y}_{i,p} = \boldsymbol{y}_{j,p} \quad \forall p \in \mathcal{P}, \forall (i,j) \in \mathcal{E}_p^{\mathrm{D}}. \end{cases} \tag{13}$$

□

If $\mathcal{G}_p^{\mathrm{D}} = \mathcal{G}^{\mathrm{C}}$ for all $p \in \mathcal{P}$, then (13) reverts to the formulation used in standard dual methods [21]–[23]: these algorithms require each agent to store a copy of the whole optimization vector. Instead, choosing a sparse $\mathcal{G}_p^{\mathrm{D}}$ can conveniently reduces the number constraints in (13). Regardless, due to its structure (i.e., separable costs and coupling constraints compliant with $\mathcal{G}_p^{\mathrm{D}}$, hence with the communication graph), the problem in (13) can be immediately solved via several established Lagrangian-based algorithms (provided that the functions f_i's are sufficiently well-behaved). In practice, this allows one to extend most (virtually all) the existing dual methods to the END framework.

Example 1 (END ADMM): Let Assumption 1 hold, and assume that f_i is proper closed convex, for all $i \in \mathcal{I}$. Applying the alternating direction method of multipliers (ADMM) in [22] to (13)[4] results in the iteration

$$\tilde{\boldsymbol{y}}_i^{k+1} = \operatorname*{argmin}_{\tilde{\boldsymbol{y}}_i} \Big\{ f_i(\tilde{\boldsymbol{y}}_i) + \textstyle\sum_{p\in\mathcal{N}^{\mathrm{E}}(i)} \sum_{j\in\mathcal{N}_p^{\mathrm{D}}(i)} \Big(\|\boldsymbol{y}_{i,p}\|^2 - \langle z_{i,j,p}, \boldsymbol{y}_{i,p}\rangle \Big) \Big\} \tag{14a}$$

$$z_{i,j,p}^{k+1} = (1-\alpha) z_{i,j,p}^k - \alpha z_{j,i,p}^k + 2\alpha \boldsymbol{y}_{j,p}^{k+1}, \tag{14b}$$

where $z_{i,j,p}$ is an auxiliary variable kept by agent i, for each $i \in \mathcal{I}$, $p \in \mathcal{N}^{\mathrm{E}}(i)$, $j \in \mathcal{N}_p^{\mathrm{D}}(i)$. Then, for any $\alpha \in (0,1)$, $\boldsymbol{y}_{i,p}$ converges to $y_p^\star$, where $y^\star = \mathrm{col}((y_p^\star)_{p\in\mathcal{P}})$ is a solution of (11), for all $i \in \mathcal{I}$ and $p \in \mathcal{N}^{\mathrm{E}}(i)$. Note that performing the update (14b) requires agent i to receive data from its neighbor $j \in \mathcal{N}_p^{\mathrm{D}}(i)$ (while (14b) requires no communication). If $\mathcal{G}_p^{\mathrm{D}} = \mathcal{G}^{\mathrm{C}}$ for all $p \in \mathcal{P}$, then the method retrieves the standard ADMM for consensus optimization [22, Eq. (13)]. Yet, in general (14) requires the agents to store and exchange less (auxiliary) variables. □

While Proposition 1 would hold even if the graphs $\mathcal{G}_p^{\mathrm{D}}$'s are only strongly connected, distributed algorithms to efficiently solve (13) typically require undirected communication.

B. END ABC algorithm

In this subsection, we propose an END version of the ABC algorithm, recently developed in [5]. For differentiable costs f_i's, let us consider the iteration: $(\forall i \in \mathcal{I})(\forall p \in \mathcal{N}^{\mathrm{E}}(i))$

$$\boldsymbol{y}_{i,p}^{k+1} = -\boldsymbol{z}_{i,p}^k + \sum_{j\in\overline{\mathcal{N}}^{\mathrm{E}}(p)} [\mathrm{A}_p]_{i_p,j_p} \boldsymbol{y}_{j,p}^k - \gamma [\mathrm{B}_p]_{i_p,j_p} \nabla_{y_p} f_j(\tilde{\boldsymbol{y}}_j^k) \tag{15a}$$

$$\boldsymbol{z}_{i,p}^{k+1} = \boldsymbol{z}_{i,p}^k + \sum_{j\in\overline{\mathcal{N}}^{\mathrm{E}}(p)} [\mathrm{C}_p]_{i_p,j_p} \boldsymbol{y}_{j,p}^{k+1}, \tag{15b}$$

where $\boldsymbol{z}_{i,p} \in \mathbb{R}^{n_{y_p}}$ is a local variable kept by agent i; for all $p \in \mathcal{P}$, $\mathrm{A}_p, \mathrm{B}_p, \mathrm{C}_p$ are matrices in $\mathbb{R}^{N_p \times N_p}$; $\gamma > 0$ is a step size; and we recall the notation in (9). Note that if the matrices $\mathrm{A}_p, \mathrm{B}_p, \mathrm{C}_p$'s are compliant with the corresponding graphs $\mathcal{G}_p^{\mathrm{D}}$'s (e.g., $\mathrm{A}_p = \mathrm{B}_p = \mathrm{C}_p = \mathrm{W}_p^{\mathrm{D}}$), then the iteration (15) is distributed. We can rewrite (15) in stacked form as

$$\boldsymbol{y}^{k+1} = \mathrm{A}\boldsymbol{y}^k - \gamma \mathrm{B} \nabla_{\boldsymbol{y}} \boldsymbol{f}(\boldsymbol{y}^k) - \boldsymbol{z}^k \tag{16a}$$

$$\boldsymbol{z}^{k+1} = \boldsymbol{z}^k + \mathrm{C}\boldsymbol{y}^{k+1}, \tag{16b}$$

where $\mathrm{A} := \mathrm{diag}((\mathrm{A}_p \otimes \mathrm{I}_{n_{y_p}})_{p\in\mathcal{P}})$, $\mathrm{B} := \mathrm{diag}((\mathrm{B}_p \otimes \mathrm{I}_{n_{y_p}})_{p\in\mathcal{P}})$, $\mathrm{C} := \mathrm{diag}((\mathrm{C}_p \otimes \mathrm{I}_{n_{y_p}})_{p\in\mathcal{P}})$ belong to $\mathbb{R}^{n_{\boldsymbol{y}} \times n_{\boldsymbol{y}}}$, $\boldsymbol{z} := \mathrm{col}((\boldsymbol{z}_p)_{p\in\mathcal{P}})$ with $\boldsymbol{z}_p := \mathrm{col}((\boldsymbol{z}_{i,p})_{i\in\overline{\mathcal{N}}^{\mathrm{E}}(p)})$, and $\boldsymbol{f}(\boldsymbol{y}) := \sum_{i\in\mathcal{I}} f_i(\tilde{\boldsymbol{y}}_i)$. If $\mathcal{G}_p^{\mathrm{D}} = \mathcal{G}^{\mathrm{C}}$ for all p, and A_p, B_p, C_p are independent of p, then (16) retrieves the ABC algorithm [5, Eq. 3].

We next characterize the asymptotic behavior of (16) for appropriately chosen A, B, C (all the proofs are in appendix). We recall the notation in (7)-(8).

Theorem 1: Let $\mathrm{D} := \mathrm{diag}((\mathrm{D}_p \otimes \mathrm{I}_{n_{y_p}})_{p\in\mathcal{P}})$, for some $\{\mathrm{D}_p \in \mathbb{R}^{N_p \times N_p}\}_{p\in\mathcal{P}}$. Assume that f_i is L-smooth and convex for each $i \in \mathcal{I}$, and that:

(a) $\mathrm{A} = \mathrm{BD}$ and $\mathrm{B} \succcurlyeq 0$, $\mathrm{D} \succ 0$;
(b) $(\forall \boldsymbol{y} \in \boldsymbol{\mathcal{C}})\ \mathrm{D}\boldsymbol{y} = \boldsymbol{y}$, $\mathrm{B}\boldsymbol{y} = \boldsymbol{y}$;
(c) $\mathrm{C} \succcurlyeq 0$, $\mathrm{null}(\mathrm{C}) = \boldsymbol{\mathcal{C}}$;
(d) B and C commute: $\mathrm{BC} = \mathrm{CB}$;
(e) $\mathrm{I} - \frac{1}{2}\mathrm{C} - \sqrt{\mathrm{B}}\mathrm{D}\sqrt{\mathrm{B}} \succcurlyeq 0$.

Let $y^\star \in \mathcal{Y}^\star$, $\boldsymbol{y}^\star := \boldsymbol{\mathcal{C}}(y^\star)$, and consider the merit function

$$\mathfrak{M}(\boldsymbol{y}) := \max\{\|\Pi_\perp \boldsymbol{y}\| \|\nabla_{\boldsymbol{y}} \boldsymbol{f}(\boldsymbol{y}^\star)\|, |\boldsymbol{f}(\boldsymbol{y}) - \boldsymbol{f}(\boldsymbol{y}^\star)|\}. \tag{17}$$

Then, for any $\boldsymbol{y}^0 \in \mathbb{R}^{n_{\boldsymbol{y}}}$, $\boldsymbol{z}^0 = \boldsymbol{0}_{n_{\boldsymbol{y}}}$, $\gamma \in (\frac{\lambda_{\min}(D)}{L})$, the sequence $(\boldsymbol{y}^k)_{k\in\mathbb{N}}$ generated by (16) satisfies

$$\mathfrak{M}\left(\boldsymbol{y}_{\mathrm{avg}}^k\right) \leq \mathcal{O}(\tfrac{1}{k}), \tag{18}$$

for all $k \in \mathbb{N}$, where $\boldsymbol{y}_{\mathrm{avg}}^k := \frac{1}{k}\sum_{t=1}^k \boldsymbol{y}^t$. □

It is shown in [5] that many celebrated schemes for consensus optimization can be retrieved as particular instances of the ABC algorithm, by suitably choosing the matrices A, B, C [5, Tab. 2]: EXTRA [24], NEXT [18], DIGing [4], NIDS [25], and others. Theorem 1 allows the extension of each of these methods to the END framework. We only discuss an example below; for the other schemes, the analysis can be carried out analogously, see also [5, §III.A].

Example 2 (END AugDGM): The following gradient-tracking algorithm is the END version of [16, Alg. 1]: $(\forall i \in \mathcal{I})(p \in \mathcal{N}^{\mathrm{E}}(i))$

$$\boldsymbol{y}_{i,p}^{k+1} = \sum_{j\in\mathcal{N}_p^{\mathrm{D}}(i)} [\mathrm{W}_p^{\mathrm{D}}]_{i_p,j_p} (\boldsymbol{y}_{j,p}^k - \gamma \boldsymbol{v}_{j,p}^k)$$

$$\boldsymbol{v}_{i,p}^{k+1} = \sum_{j\in\mathcal{N}_p^{\mathrm{D}}(i)} [\mathrm{W}_p^{\mathrm{D}}]_{i_p,j_p} (\boldsymbol{v}_{j,p}^k + \nabla_{y_p} f_j(\tilde{\boldsymbol{y}}^{k+1}) - \nabla_{y_p} f_j(\tilde{\boldsymbol{y}}^k)),$$

or, in stacked form,

$$\boldsymbol{y}^{k+1} = \mathbf{W}^{\mathrm{D}}(\boldsymbol{y}^k - \gamma \boldsymbol{v}^k) \tag{20a}$$

$$\boldsymbol{v}^{k+1} = \mathbf{W}^{\mathrm{D}}(\boldsymbol{v}^k + \nabla_{\boldsymbol{y}} \boldsymbol{f}(\boldsymbol{y}^{k+1}) - \nabla_{\boldsymbol{y}} \boldsymbol{f}(\boldsymbol{y}^k)); \tag{20b}$$

we impose $\boldsymbol{y}(0) = \boldsymbol{0}$, $\boldsymbol{v}(0) = \mathbf{W}^{\mathrm{D}} \nabla_{\boldsymbol{y}} \boldsymbol{f}(\boldsymbol{y}^0)$. Here, $\boldsymbol{v}_{i,p}$ represents an estimate of $\nabla_{y_p} \sum_{j\in\mathcal{I}} f_j(y)/N_p$ kept by agent i. Note that agent i only estimates and exchanges the components of the cost gradient (and of the optimization variable) specified by $\mathcal{N}^{\mathrm{E}}(i)$, instead of the whole vector

[4] After decoupling the constraints in (13) by introducing auxiliary bridge variables as $\{\boldsymbol{y}_{i,p} = h_{(i,j),p}, h_{(i,j),p} = h_{(j,i),p}, h_{(j,i),p} = \boldsymbol{y}_{j,p}\}$; the approach is standard and we refer to [22] for a complete derivation.

as in [16, Alg. 1] – the two algorithms coincide only if $\mathrm{W}_p^{\mathrm{D}} = \mathrm{W}^{\mathrm{C}}$ for all $p \in \mathcal{P}$. By eliminating the $\boldsymbol{v}$ variable in (20), we obtain

$$\begin{aligned}\boldsymbol{y}^{k+2} = 2\mathbf{W}^{\mathrm{D}}\boldsymbol{y}^{k+1} - (\mathbf{W}^{\mathrm{D}})^2\boldsymbol{y}^k \\ - \gamma(\mathbf{W}^{\mathrm{D}})^2(\nabla_{\boldsymbol{y}}\boldsymbol{f}(\boldsymbol{y}^{k+1}) - \nabla_{\boldsymbol{y}}\boldsymbol{f}(\boldsymbol{y}^k)).\end{aligned} \tag{21}$$

Instead, eliminating $\boldsymbol{z}$ from (16) we get

$$\begin{aligned}\boldsymbol{y}^{k+2} = (\mathrm{I} - \mathrm{C} + \mathrm{A})\boldsymbol{y}^{k+1} - \mathrm{A}\boldsymbol{y}^k \\ - \gamma\mathrm{B}(\nabla_{\boldsymbol{y}}\boldsymbol{f}(\boldsymbol{y}^{k+1}) - \nabla_{\boldsymbol{y}}\boldsymbol{f}(\boldsymbol{y}^k)).\end{aligned} \tag{22}$$

which retrieves (21) for $\mathrm{A} = \mathrm{B} = (\mathbf{W}^{\mathrm{D}})^2$, $\mathrm{C} = (\mathrm{I} - \mathbf{W}^{\mathrm{D}})^2$.[5] This choice satisfies the conditions in Theorem 1, with $\mathrm{D} = \mathrm{I}$, under Assumption 1 and doubly stochasticity.[6]

Corollary 1: Let Assumption 1 hold; assume that $\mathrm{W}_p^{\mathrm{D}}\mathbf{1}_{N_p} = \mathbf{1}_{N_p}$, $\mathrm{W}_p^{\mathrm{D}} = \mathrm{W}_p^{\mathrm{D}\top}$, for all $p \in \mathcal{P}$, and that f_i is L-smooth and convex, for all $i \in \mathcal{I}$. Then, for any $\gamma \in (0, \frac{1}{L})$ the rate (18) holds for (20). □

Theorem 1 requires a recovery procedure (i.e., (18) holds for the running average only), as e.g. in [26], but pointwise convergence could be shown for several special cases of (16), see e.g. [16]. We note that Theorem 1 enhances customizability with respect to [5, Th. 24], even in the standard scenario (2) (the sparsity-oblivious case), by allowing for non-identical blocks A_p's, B_p's, C_p's – corresponding to integrating different methods for the components of y.

Remark 1 (Linear convergence): For certain design choices, the ABC algorithm achieves linear convergence when each function f_i is strongly convex in y [18, Th. 15]. Interestingly, it can be analogously shown that the END ABC (16) converges linearly under a weaker assumption, namely strong convexity of each f_i with respect to $\tilde{\boldsymbol{y}}_i$ only. Note that this condition requires $\mathcal{G}^{\mathrm{E}} = \mathcal{G}^{\mathrm{I}}$, which is only a viable choice in specific cases [1, App. A]. □

C. END Push-sum DGD

Techniques to solve optimization problems over switching or directed graphs also find their counterpart in the END framework. As an example, here we generalize the push-sum subgradient algorithm in [17, Eq. (1)].

Let the agents communicate over a time-varying network $(\mathcal{G}^{\mathrm{C},k})_{k\in\mathbb{N}}$, $\mathcal{G}^{\mathrm{C},k} = (\mathcal{I}, \mathcal{E}^{\mathrm{C},k})$. Given a fixed estimate graph $\mathcal{G}^{\mathrm{E}} \supseteq \mathcal{G}^{\mathrm{I}}$, for each $p \in \mathcal{P}$ we consider a time-dependent design graph $(\mathcal{G}_p^{\mathrm{D},k})_{k\in\mathbb{N}}$, $\mathcal{G}_p^{\mathrm{D},k} = (\overline{\mathcal{N}}^{\mathrm{E}}(p), \mathcal{E}_p^{\mathrm{D},k}) \subseteq \mathcal{G}^{\mathrm{C},k}$ (note that the set of nodes is fixed in $\mathcal{G}_p^{\mathrm{D},k}$). For all $i \in \mathcal{I}$ and $p \in \mathcal{N}^{\mathrm{E}}(i)$, agent i performs the following updates:

$$q_{i,p}^{k+1} = \textstyle\sum_{j\in\overline{\mathcal{N}}^{\mathrm{E}}(p)} [\mathrm{W}_p^{\mathrm{D},k}]_{i_p,j_p} q_{j,p}^k \tag{23a}$$

$$\boldsymbol{w}_{i,p}^{k+1} := \textstyle\sum_{j\in\overline{\mathcal{N}}^{\mathrm{E}}(p)} [\mathrm{W}_p^{\mathrm{D},k}]_{i_p,j_p} \boldsymbol{z}_{j,p}^k \tag{23b}$$

$$\boldsymbol{g}_{i,p}^{k+1} \in \partial_{y_p} f_i(\tilde{\boldsymbol{y}}_i^{k+1}), \quad \boldsymbol{y}_{i,p}^{k+1} := \frac{\boldsymbol{w}_{i,p}^{k+1}}{q_{i,p}^{k+1}} \tag{23c}$$

$$\boldsymbol{z}_{i,p}^{k+1} = \boldsymbol{w}_{i,p}^{k+1} - \gamma^k \boldsymbol{g}_{i,p}^{k+1}, \tag{23d}$$

initialized at $\boldsymbol{z}_{i,p}^0 \in \mathbb{R}^{n_{y_p}}$, $q_{i,p}^0 = 1$. With respect to [17, Eq. (1)], agent i keeps one scalar $q_{i,p}$ for each $p \in \mathcal{N}^{\mathrm{E}}(i)$ (instead of one overall), but does not store and exchange the variables $\boldsymbol{z}_{i,p} \in \mathbb{R}^{n_{y_p}}$ for $p \notin \mathcal{N}^{\mathrm{E}}(i)$.

Assumption 2: For all $k \in \mathbb{N}$ and $p \in \mathcal{P}$, it holds that:
(i) *Self-loops:* for all $i \in \overline{\mathcal{N}}^{\mathrm{E}}(p)$, $(i,i) \in \mathcal{E}_p^{\mathrm{D},k}$;
(ii) *Column-stochasticity*: $\mathbf{1}_{N_p}^\top \mathrm{W}_p^{\mathrm{D},k} = \mathbf{1}_{N_p}^\top$;
(iii) *Finite weights*: $[\mathrm{W}_p^{\mathrm{D},k}]_{i_p,j_p} \geq \nu > 0$, $\forall (i,j) \in \mathcal{E}_p^{\mathrm{D},k}$. □

Assumption 3: There exists an integer $Q > 0$ such that, for all $p \in \mathcal{P}$, $(\mathcal{G}_p^{\mathrm{D},k})_{k\in\mathbb{N}}$ is Q-strongly connected. □

Example 3 (Choosing time-varying design graphs): Assume $\mathcal{G}^{\mathrm{C}} \subseteq \bigcup_{t=kQ}^{(k+1)Q-1} \mathcal{G}^{\mathrm{C},k}$, for all $k \in \mathbb{N}$ and some strongly connected graph $\mathcal{G}^{\mathrm{C}}$. Choose some graphs $(\mathcal{G}_p^{\mathrm{D}})_{p\in\mathcal{P}}$ that satisfy Standing Assumption 1 and such that each $\mathcal{G}_p^{\mathrm{D}}$ is strongly connected. Then, Assumption 3 holds by settihg $\mathcal{G}_p^{\mathrm{D},k} = \mathcal{G}_p^{\mathrm{D}} \bigcap \mathcal{G}^{\mathrm{C},k}$, for all $p \in \mathcal{P}$ and all $k \in \mathbb{N}$. □

Theorem 2: Let Assumptions 2 and 3 hold. Assume that, for all $i \in \mathcal{I}$, f_i is convex and there is $L > 0$ such that $\|g_i\| \leq L$, for all $y \in \mathbb{R}^{n_y}$ and $g_i \in \partial_y f_i(y)$. Let $(\gamma^k)_{k\in\mathbb{N}}$ be a positive non-increasing sequence such that $\sum_{k=0}^\infty \gamma^k = \infty$, $\sum_{k=0}^\infty (\gamma^k)^2 < \infty$. Then, the sequence $(\boldsymbol{y}^k)_{k\in\mathbb{N}}$ generated by (23) converges to $\boldsymbol{\mathcal{C}}(y^\star)$, for some $y^\star \in \mathcal{Y}^\star$. □

D. Constraint-coupled distributed optimization

Finally, we study a different, constraint-coupled problem:

$$\begin{cases} \min\limits_{x_i\in\mathbb{R}^{n_{x_i}}, i\in\mathcal{I}} \ \sum_{i\in\mathcal{I}} f_i(x_i) & \text{(24a)} \\ \text{s.t.} \quad \sum_{i\in\overline{\mathcal{N}}^{\mathrm{I}}(p)} \mathrm{A}_{p,i}x_i - a_{p,i} = \mathbf{0}, \quad \forall p \in \mathcal{P} & \text{(24b)} \end{cases}$$

for a given interference graph $\mathcal{G}^{\mathrm{I}} = (\mathcal{P}, \mathcal{I}, \mathcal{E}^{\mathrm{I}})$, where f_i and $\{\mathrm{A}_{p,i} \in \mathbb{R}^{n_{y_p}\times n_{x_i}}, a_{p,i} \in \mathbb{R}^{n_{y_p}}\}_{p\in\mathcal{N}^{\mathrm{I}}(i)}$ are private data kept by agent i; and the constraints (24b) are *not* compliant with the communication graph $\mathcal{G}^{\mathrm{C}}$, namely $\overline{\mathcal{N}}^{\mathrm{I}}(p) \not\subseteq \mathcal{N}^{\mathrm{C}}(i)$ for any i. Differently from what we did with the cost-coupled problem in (11), here we choose as the variable of interest the dual variable associated with the constraints in (24b), $y = \mathrm{col}((y_p)_{p\in\mathcal{P}}) \in \mathbb{R}^{n_y}$. Typical distributed methods to solve (24) require each agent to store a copy of the entire dual variable (and possibly of other variables in $\mathbb{R}^{n_y}$, e.g., an estimate of the constraint violation) [27], [28]. END primal-dual or dual methods can improve efficiency by exploiting the sparsity of $\mathcal{G}^{\mathrm{I}}$. For instance, (a simplified version of) the algorithm in [1, Eq. (31)] can be directly used to solve (24). Alternatively, let us consider the dual of (24):

$$\max_{y\in\mathbb{R}^{n_y}} \ \textstyle\sum_{i\in\mathcal{I}} \varphi_i((y_p)_{p\in\mathcal{N}^{\mathrm{I}}(i)}), \tag{25}$$

$\varphi_i(y) := \min_{x_i\in\mathbb{R}^{n_{x_i}}} f_i(x_i) + \sum_{p\in\mathcal{N}^{\mathrm{I}}(i)} \langle y_p, \mathrm{A}_{p,i}x_i - a_{p,i}\rangle$; note that (25) is in the form (11). In fact, (25) was solved in [12] via the reformulation (13); this approach has the disadvantage of requiring undirected communication. Nonetheless, (25) can also be solved over directed (time-varying) networks, e.g., via the iteration in (23).[7]

[5] In fact, the sequence $(\boldsymbol{y}^k)$ generated by (16) coincide with that generated by (20) for the given initialization.

[6] Note that the properties of $\mathrm{W}_p^{\mathrm{D}}$'s easily translate to $\mathbf{W}^{\mathrm{D}}$ due to the block structure in (6). For instance, under the stated conditions, clearly $\mathrm{null}(I - \mathrm{W}_p^{\mathrm{D}})^2 = \mathrm{range}(\mathbf{1}_{N_p})$ and $\mathrm{null}(I - \mathbf{W}^{\mathrm{D}})^2 = \boldsymbol{\mathcal{C}}$.

[7] If each f_i is convex with compact domain, where the subgradients of the local dual function φ_i can be computed as $\boldsymbol{g}_{i,p}^k = \mathrm{A}_{p,i}x_i^\star(\tilde{\boldsymbol{y}}_i^k) - a_{p,i}$, with $x_i^\star(\tilde{\boldsymbol{y}}_i) \in \mathrm{argmin}_{x_i\in\mathbb{R}^{n_{x_i}}} f_i(x_i) + \sum_{p\in\mathcal{N}^{\mathrm{I}}(i)} \langle \boldsymbol{y}_{i,p}, \mathrm{A}_{p,i}x_i - a_{p,i}\rangle$.

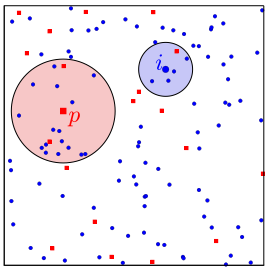

Fig. 2: Distribution of sources (red) and sensors (blue). Sensors in the red circle receive signal from source p. Sensors in the blue circle can receive data by (but not necessarily send to) sensor i.

IV. Illustrative application

In this section we study numerically a regression problem with sparse measurements [2], [13], arising from distributed estimation in wireless and ad-hoc sensor networks. Let us consider some sensors $\{1, 2, \dots, N\} =: \mathcal{I}$ and some sources $\{1, 2, \dots, P\} =: \mathcal{P}$, spatially distributed on a plane in the square $[0, 1] \times [0, 1]$, as illustrated in Figure 2. Each source p emits a signal $\bar{y}_p \in \mathbb{R}$, sensed by all the sensors in a radius $r_{\mathrm{s}} > 0$; in turn, each sensor i measures

$$h_i := \mathrm{H}_i \, \mathrm{col}((\bar{y}_p)_{p \in \mathcal{N}^{\mathrm{I}}(i)}) + w_i, \tag{26}$$

where $h_i \in \mathbb{R}^{n_{h_i}}$, H_i is a known output matrix, w_i is the measurement noise. Sensor i can send information to all the peers in a radius r_{c}^i (e.g., proportional to the sensor specific power); this induces a directed communication network $\mathcal{G}^{\mathrm{C}} = (\mathcal{I}, \mathcal{E}^{\mathrm{C}})$ among the sensors, which we assume to be *strongly connected*.

1) Linear regression: In our first simulation, the sensors' goal is to collaboratively solve the least square problem

$$\min_{y \in \mathbb{R}^P} \sum_{i \in \mathcal{I}} \left\| h_i - \mathrm{H}_i \mathrm{col}((y_p)_{p \in \mathcal{N}^{\mathrm{I}}(i))} \right\|^2, \tag{27}$$

where $\mathcal{N}^{\mathrm{I}}(i)$ is the set of sources positioned less than r_{s} away from sensor i. Problem (27) is in the form (11). We seek a solution via algorithm (23) (with fixed communication graph), comparing the performance for two choices of the design graphs:

- *Standard*: $\mathcal{G}_p^{\mathrm{D}}$'s are chosen as in (2): with this choice, (23) boils down to the standard Push-sum DGD [17].
- *Customized*: each $\mathcal{G}_p^{\mathrm{D}}$ is designed to exploit the sparsity in (27). In particular, we aim at optimizing the memory allocation for the estimates, by minimizing the number of nodes in $\mathcal{G}_p^{\mathrm{D}}$, provided that $\mathcal{G}_p^{\mathrm{D}}$ must be strongly connected (and Standing Assumption 1 must be satisfied). Designing such a $\mathcal{G}_p^{\mathrm{D}}$ corresponds to (approximately) solving a Strongly Connected Steiner Subgraph Problem [29] (where $\mathcal{G}_p^{\mathrm{D}}$ is a subgraph of $\mathcal{G}^{\mathrm{C}}$)[8].

We set $N = 100$, $P = 20$, and randomly generate sensor/sources positions as in Figure 2. We choose $r_{\mathrm{s}} = 0.2$, and draw each r_{c}^i uniformly in $[r_{\mathrm{c}}^{\min}, r_{\mathrm{c}}^{\min} + 0.1]$. For all $i \in \mathcal{I}$, we fix $n_{h_i} = 10$, we generate H_i by first uniformly drawing entries in $[0, 1]$ and then normalizing the rows to unitary norm, we draw each element of w_i from an unbiased normal distribution with variance 0.1; each signal $\bar{y}_p$ is uniformly randomly chosen in $[0, 1]$; the step size is set as $\gamma^k = k^{-0.51}$ in (23).[9] The advancement is evaluated via the merit function $\mathfrak{V}(\boldsymbol{y}) := \max\{\|\mathrm{diag}((\frac{1}{N_p}\mathrm{I})_{p \in \mathcal{P}})\Pi_{\perp}\boldsymbol{y}\|, \|\nabla_{\boldsymbol{y}} \boldsymbol{f}(\boldsymbol{y}^\star)\|, |\boldsymbol{f}(\Pi_{\parallel}\boldsymbol{y}) - \boldsymbol{f}(\boldsymbol{y}^\star)|\}$, where $\boldsymbol{y}^\star = \boldsymbol{\mathcal{C}}(y^\star)$ and $y^\star$ solves (27). Figure 3 shows the results for different values of $r_{\mathrm{c}}^{\min}$. For $r_{\mathrm{c}}^{\min} = 0.1$, the customized method is 15 times faster then the standard one. Increasing $r_{\mathrm{c}}^{\min}$ only marginally reduces the per-iteration communication cost of the customized method. In fact, already for $r_{\mathrm{c}}^{\min} = 0.25$, the graph $\mathcal{G}^{\mathrm{C}}|_{\overline{\mathcal{N}}^{\mathrm{I}}(p)}$ is strongly connected for all $p \in \mathcal{P}$, so $\mathcal{G}^{\mathrm{E}} = \mathcal{G}^{\mathrm{I}}$ can be chosen (in other terms, each agent only estimates and exchanges the components of y that directly affect its local cost, while it also has to estimate other components for smaller $r_{\mathrm{c}}^{\min}$). In this situation, the customized method achieves a reduction of the communication cost (where sending a variable to *all* the neighbors on $\mathcal{G}^{\mathrm{C}}$ has a cost of 1, in a broadcast fashion) of over 99.9%.

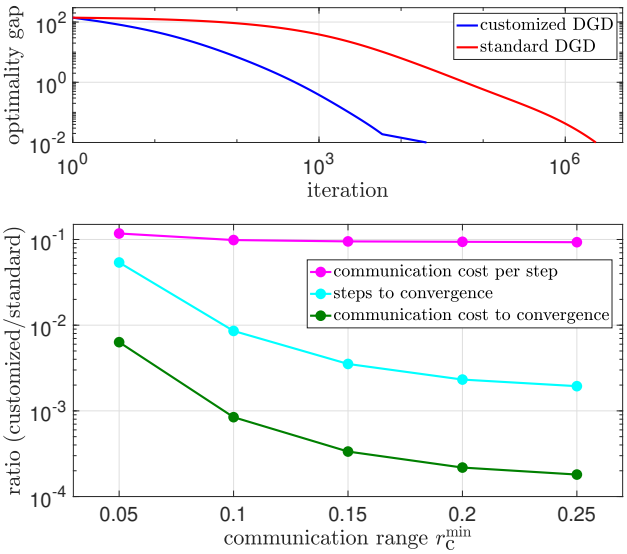

Fig. 3: Linear regression via algorithm (23), for different values of the minimum sensor communication radius $r_{\mathrm{c}}^{\min}$ and stopping criterion $\mathfrak{V}(\boldsymbol{y}) \leq 10^{-2}$ (bottom), and the trajectories obtained with $r_{\mathrm{c}}^{\min} = 0.1$ (top). A larger $r_{\mathrm{c}}^{\min}$ induces a denser graph $\mathcal{G}^{\mathrm{C}}$.

2) LASSO: Next, we assume that only 30% of the sources emits a signal at a given instant (the vector $\bar{y}$ is sparse). The sensors collaboratively solve the following problem, regularized to promote sparsity,

$$\min_{y \in \mathbb{R}^P} \ \|y\|_1 + \sum_{i \in \mathcal{I}} \left\| h_i - \mathrm{H}_i \mathrm{col}(y_p)_{p \in \mathcal{N}^{\mathrm{I}}(i)} \right\|^2,$$

where $\|\cdot\|_1$ is the ℓ_1 norm. By defining $f_i((y_p)_{p \in \mathcal{N}^{\mathrm{I}}(i)}) = \|h_i - \mathrm{H}_i \mathrm{col}((y_p)_{p \in \mathcal{N}^{\mathrm{I}}(i)})\|^2 + \sum_{p \in \mathcal{N}^{\mathrm{I}}(i)} \frac{1}{|\overline{\mathcal{N}}^{\mathrm{I}}(p)|} |y_p|$, we retrieve the form (11). We set $N = 10$, $P = 20$, $r_{\mathrm{c}}^{\min} =$

[8] We use all the available edges, i.e., $\mathcal{G}_p^{\mathrm{D}} = \mathcal{G}^{\mathrm{C}}|_{\overline{\mathcal{N}}^{\mathrm{E}}(p)}$.

[9] Although the bounded subgradient assumption in Theorem 2 fails, boundedness of the sequences generated by (23), and hence convergence, can be established based on coercivity of the cost function.

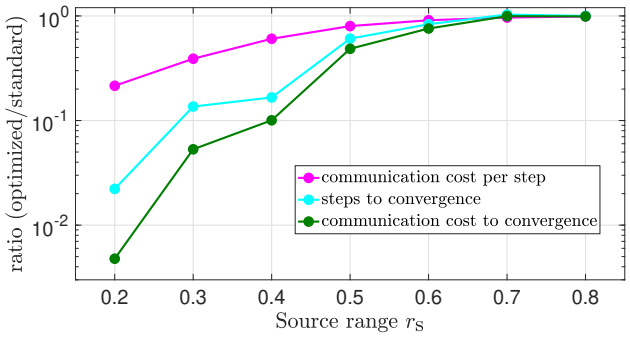

Fig. 4: LASSO via algorithm (23), and different source ranges r_{s}.

0.1, $n_{h_i} = 1$ for all i, generate random positions for the sensors and sources, and choose the other parameters as above, for both the standard and customized methods. Figure 4 compares the results for different values of r_{s}. For larger r_{s}, the interference graph $\mathcal{G}^{\mathrm{I}}$ is denser, and the gap between customized and standard method decreases: in fact, for $r_{\mathrm{s}} = 0.8$ the two algorithms coincide, as $\mathcal{G}^{\mathrm{I}}$ is complete. Nonetheless, when $\mathcal{G}^{\mathrm{I}}$ is sparse, the customized algorithm saves up to 99% of the communication cost.

In conclusion, while requiring some initial computational effort to choose the design graphs $\mathcal{G}_p^{\mathrm{D}}$, the sparsity-aware method results in substantial efficiency improvement – especially if the estimation problem is solved repeatedly, e.g., each time new signals are received from the sources.

V. Conclusion

We have shown that the END framework [1] can be applied to a variety of distributed optimization problems, to enhance efficiency by accounting for the intrinsic sparsity in the agents coupling. Besides revisiting dual methods, we derived the END (i.e., sparsity aware) version of the very general ABC method and of Push-sum DGD; and we showed how to efficiently tackle constraint-coupled problems with sparse constraints, even over directed graphs. Our simulations show that END algorithms can substantially reduce the computational and communication cost, while introducing little complication in the convergence proof with respect to their sparsity-unaware counterparts.

As sparsity-aware END algorithms require some initial design effort for the allocation of the estimates, their use is particularly recommended for problems with special structure [1, App. A.4], or repeated/time-varying problems like distributed estimation and MPC. Future work should focus on computationally efficient and distributed methods to perform the allocation of the estimates online, thus avoiding the need for any a priori design.

A. Proof of Theorem 1

We adapt the proof of [5, Th. 24]. We note that $\boldsymbol{z}^0 = \mathbf{0}_{n_{\boldsymbol{y}}} \in \mathrm{range}(\mathrm{B})$; by the conditions (a) and (d), the update in (16), and an induction argument, we have $\boldsymbol{y}^k, \boldsymbol{z}^k \in \mathrm{range}(\mathrm{B})$, for all $k \geq 1$. Hence, we rewrite (16) as

$$\boldsymbol{y}^k = \mathrm{B}\underline{\boldsymbol{y}}^k, \quad \boldsymbol{z}^k = \gamma \mathrm{B}\underline{\boldsymbol{z}}^k \tag{28a}$$

$$\underline{\boldsymbol{y}}^{k+1} = \mathrm{D}\boldsymbol{y}^k - \gamma(\nabla_{\boldsymbol{y}} \boldsymbol{f}(\boldsymbol{y}^k) + \underline{\boldsymbol{z}}^k) \tag{28b}$$

$$\underline{\boldsymbol{z}}^{k+1} = \underline{\boldsymbol{z}}^k + \tfrac{1}{\gamma}\mathrm{C}\underline{\boldsymbol{y}}^{k+1} \tag{28c}$$

for all $k \geq 1$. Let $\Phi(\boldsymbol{y}, \boldsymbol{z}) \coloneqq \boldsymbol{f}(\boldsymbol{y}) + \langle \boldsymbol{y}, \boldsymbol{z} \rangle$; the form in (28) can be exploited to prove the following lemma.

Lemma 1: Let $(\boldsymbol{y}^k, \underline{\boldsymbol{y}}^k, \underline{\boldsymbol{z}}^k)$ be a sequence generated by (28). Then, for all $\boldsymbol{y} \in \boldsymbol{\mathcal{C}}$, $\boldsymbol{z} \in \boldsymbol{\mathcal{C}}_\perp$, it holds that:

$$\Phi(\boldsymbol{y}_{\mathrm{avg}}^{k+1}, \boldsymbol{z}) - \Phi(\boldsymbol{y}, \boldsymbol{z}) \leq \tfrac{1}{2k} h(\boldsymbol{y}, \boldsymbol{z}),$$

where $h(\boldsymbol{y}, \boldsymbol{z}) \coloneqq \frac{1}{\gamma}\|\boldsymbol{y}^0 - \boldsymbol{y}\|_{\mathrm{D}}^2 + \gamma \frac{\|\mathrm{B} - \Pi_{\|}\|}{\underline{\lambda}} \|\boldsymbol{z}\|^2$ and $\underline{\lambda} \coloneqq \min\{(\lambda_2(\mathrm{C}_p))_{p \in \mathcal{P}}\}$. □

Proof: The proof is analogous to that of [5, Lemma 23], and omitted here. Note that [5] uses a matrix notation (i.e., $\boldsymbol{y} \in \mathbb{R}^{I \times n}$), while we need a stacked notation (as the vectors $(\boldsymbol{y}_i)_{i \in \mathcal{I}}$ are not homogeneous in size). Nonetheless, (28) matches [5, Eq. (33)], which allows us to repeat all the steps in [5, Lem. 23] (with the only precaution of replacing J, $\mathrm{span}(\mathbf{1}_m)$, $\lambda_2(\mathrm{C})$ in [5] with $\Pi_{\|}$, $\boldsymbol{\mathcal{C}}$, $\underline{\lambda}$). ■

For all $\boldsymbol{z} \in \boldsymbol{\mathcal{C}}_\perp$ (so that $\langle \boldsymbol{z}, \boldsymbol{y}^\star \rangle = 0$), setting $\boldsymbol{y} = \boldsymbol{y}^\star$ in Lemma 1, together with the definition of Φ, yields $\boldsymbol{f}(\boldsymbol{y}_{\mathrm{avg}}^k) - \boldsymbol{f}(\boldsymbol{y}^\star) + \langle \boldsymbol{y}_{\mathrm{avg}}^k, \boldsymbol{z} \rangle \leq \frac{1}{2k} h(\boldsymbol{y}^\star, \boldsymbol{z})$. Further choosing $\boldsymbol{z} = 2\frac{\Pi_\perp \boldsymbol{y}_{\mathrm{avg}}^k}{\|\Pi_\perp \boldsymbol{y}_{\mathrm{avg}}^k\|}\|\boldsymbol{z}^\star\|$, with $\boldsymbol{z}^\star \coloneqq -\nabla_{\boldsymbol{y}} \boldsymbol{f}(\boldsymbol{y}^\star)$, leads to

$$\boldsymbol{f}(\boldsymbol{y}_{\mathrm{avg}}^k) - \boldsymbol{f}(\boldsymbol{y}^\star) + 2\|\boldsymbol{z}^\star\| \left\|\Pi_\perp \boldsymbol{y}_{\mathrm{avg}}^k\right\| \leq \tfrac{1}{2k} h(\boldsymbol{y}^\star, 2\boldsymbol{z}^\star). \tag{29}$$

By convexity and since $\boldsymbol{z}^\star \in \boldsymbol{\mathcal{C}}_\perp$ (by optimality conditions), it holds that $f(\boldsymbol{y}_{\mathrm{avg}}^k) - f(\boldsymbol{y}^\star) \geq -\langle \boldsymbol{y}_{\mathrm{avg}}^k - \boldsymbol{y}^\star, \boldsymbol{z}^\star \rangle = -\langle \Pi_\perp \boldsymbol{y}_{\mathrm{avg}}^k, \boldsymbol{z}^\star \rangle \geq -\|\Pi_\perp \boldsymbol{y}_{\mathrm{avg}}^k\| \|\boldsymbol{z}^\star\|$; the latter inequality and (29) imply $\mathfrak{M}(\boldsymbol{y}_{\mathrm{avg}}^k) \leq \frac{1}{2k} h(\boldsymbol{y}^\star, 2\boldsymbol{z}^\star)$. ■

B. Proof of Theorem 2

Note that, for each $p \in \mathcal{P}$, (23) is the standard perturbed push-sum protocol [17, Eq. (4)], with perturbation term $-\gamma^k \boldsymbol{g}_{i,p}^{k+1}$. Therefore, since $\boldsymbol{g}_{i,p}^{k+1}$ is uniformly bounded by assumption and by the choice of $(\gamma^k)_{k \in \mathbb{N}}$, we can apply [17, Lem. 1] to infer that, for all $i \in \mathcal{I}$, $p \in \mathcal{N}^{\mathrm{E}}(i)$

$$\lim_{k \to \infty} \|\boldsymbol{y}_{i,p}^k - \bar{\boldsymbol{z}}_p^k\| = 0, \tag{30}$$

$$\textstyle\sum_{k=0}^{\infty} \gamma^k \|\boldsymbol{y}_{i,p}^k - \bar{\boldsymbol{z}}_p^k\| = 0, \tag{31}$$

where $\bar{\boldsymbol{z}}_p^k \coloneqq \frac{1}{N_p} \sum_{i \in \overline{\mathcal{N}}^{\mathrm{E}}(p)} \boldsymbol{z}_{i,p}^k \in \mathbb{R}^{n_{y_p}}$, for all $k \in \mathbb{N}$. Let us also define $\bar{\boldsymbol{z}}^k \coloneqq \mathrm{col}((\bar{\boldsymbol{z}}_p^k)_{p \in \mathcal{P}}) \in \mathbb{R}^{n_y}$. By (23) and Assumption 2(ii), it follows that

$$\bar{\boldsymbol{z}}_p^{k+1} = \bar{\boldsymbol{z}}_p^k - \gamma^k \tfrac{1}{N_p} \textstyle\sum_{i \in \overline{\mathcal{N}}^{\mathrm{E}}(p)} \boldsymbol{g}_{i,p}^{k+1}. \tag{32}$$

We next show that $\lim_{k \to \infty} \bar{\boldsymbol{z}}^k = y^\star \in \mathcal{Y}^\star$; then, the theorem follows by (30). The main complication with respect to the proof of [17, Th. 1] is that we need a modification of [17, Lem. 8] to cope with the non-homogeneity of the estimates.

Lemma 2: For all $y^\star \in \mathcal{Y}^\star$, for all $k \in \mathbb{N}$, it holds that

$$\begin{aligned}\|\bar{\boldsymbol{z}}^{k+1} - y^\star\|_{\mathrm{D}}^2 \leq \|\bar{\boldsymbol{z}}^k - y^\star\|_{\mathrm{D}}^2 - 2\gamma^k (f(\bar{\boldsymbol{z}}^k) - f(y^\star)) \\ + 4L\gamma^k \sum_{i \in \mathcal{I}} \sum_{p \in \overline{\mathcal{N}}^{\mathrm{E}}(i)} \|\bar{\boldsymbol{z}}_p^k - \tilde{\boldsymbol{y}}_{i,p}^{k+1}\| \\ + (\gamma^k)^2 N L^2,\end{aligned}$$

where $\mathrm{D} := \mathrm{diag}((N_p \mathrm{I}_{n_p})_{p\in\mathcal{P}})$. □

Proof: By (32), we have

$$\|\bar{\boldsymbol{z}}^{k+1} - y^\star\|_{\mathrm{D}}^2 = \|\bar{\boldsymbol{z}}^{k} - y^\star\|_{\mathrm{D}}^2 - 2\gamma^k \sum_{p\in\mathcal{P}} \left\langle \bar{\boldsymbol{z}}_p^k - y_p^\star, \textstyle\sum_{i\in\overline{\mathcal{N}}^{\mathrm{E}}(p)} \boldsymbol{g}_{i,p}^{k+1} \right\rangle + (\gamma^k)^2 \sum_{p\in\mathcal{P}} \tfrac{1}{N_p} \left\| \textstyle\sum_{i\in\overline{\mathcal{N}}^{\mathrm{E}}(p)} \boldsymbol{g}_{i,p}^{k+1} \right\|^2. \tag{33}$$

The third addend on the right-hand side of (33) is bounded above by $(\gamma^k)^2 N L^2$. For the second addend, we have

$$\begin{aligned}
&\sum_{p\in\mathcal{P}} \left\langle \bar{\boldsymbol{z}}_p^k - y_p^\star, \textstyle\sum_{i\in\overline{\mathcal{N}}^{\mathrm{E}}(p)} \boldsymbol{g}_{i,p}^{k+1} \right\rangle \\
&= \sum_{i\in\mathcal{I}} \sum_{p\in\overline{\mathcal{N}}^{\mathrm{E}}(i)} \left\langle (\bar{\boldsymbol{z}}_p^k - \boldsymbol{y}_{i,p}^{k+1}) + (\boldsymbol{y}_{i,p}^{k+1} - y_p^\star), \boldsymbol{g}_{i,p}^{k+1} \right\rangle \\
&\overset{(a)}{\geq} \sum_{i\in\mathcal{I}} -L\|\mathrm{col}((\bar{\boldsymbol{z}}_p^k)_{p\in\overline{\mathcal{N}}^{\mathrm{E}}(i)}) - \tilde{\boldsymbol{y}}_i^{k+1}\| + f_i(\tilde{\boldsymbol{y}}_i^{k+1}) - f_i(y^\star) \\
&\overset{(b)}{\geq} \sum_{i\in\mathcal{I}} -2L\|\mathrm{col}((\bar{\boldsymbol{z}}_p^k)_{p\in\overline{\mathcal{N}}^{\mathrm{E}}(i)}) - \tilde{\boldsymbol{y}}_i^{k+1}\| + f_i(\bar{\boldsymbol{z}}^{k+1}) - f_i(y^\star),
\end{aligned}$$

where in (a) we used that $\boldsymbol{g}_i^{k+1} \in \partial_{\tilde{\boldsymbol{y}}_i} f_i(\tilde{\boldsymbol{y}}_i^{k+1})$ and convexity of f_i, and (b) follows by adding and subtracting (inside the sum) $f_i((\bar{\boldsymbol{z}}_p^{k+1})_{p\in\mathcal{N}^{\mathrm{E}}(i)}) = f_i(\bar{\boldsymbol{z}}^{k+1})$ and by L-Lipschitz continuity of f_i. The result follows by substituting the bound back into (33). ■

We finally note that, due to (31) and the choice of $(\gamma_k)_{k\in\mathbb{N}}$, the inequality in Lemma 2 satisfies all the conditions of [17, Lem. 7], in the norm $\|\cdot\|_{\mathrm{D}}$; hence we can conclude that $\bar{\boldsymbol{z}}^k \to y^\star$, for some $y^\star \in \mathcal{Y}^\star$. ■

REFERENCES

[1] M. Bianchi and S. Grammatico, "The END: Estimation Network Design for games under partial-decision information," *IEEE Transactions on Control of Network Systems*, 2024. [Online]. Available: https://doi.org/10.1109/TCNS.2024.3393668

[2] I. Necoara and D. Clipici, "Parallel random coordinate descent method for composite minimization: Convergence analysis and error bounds," *SIAM Journal on Optimization*, vol. 26, no. 1, pp. 197–226, 2016.

[3] P. Richtárik and M. Takáč, "Distributed coordinate descent method for learning with big data," *Journal of Machine Learning Research*, vol. 17, 2016.

[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.

[5] J. Xu, Y. Tian, Y. Sun, and G. Scutari, "Distributed algorithms for composite optimization: Unified framework and convergence analysis," *IEEE Transactions on Signal Processing*, vol. 69, pp. 3555–3570, 2021.

[6] I. Notarnicola, R. Carli, and G. Notarstefano, "Distributed partitioned big-data optimization via asynchronous dual decomposition," *IEEE Transactions on Control of Network Systems*, vol. 5, no. 4, pp. 1910–1919, 2018.

[7] T. Erseghe, "A distributed and scalable processing method based upon ADMM," *IEEE Signal Processing Letters*, vol. 19, no. 9, pp. 563–566, 2012.

[8] X. Jiang, X. Zeng, J. Sun, and J. Chen, "Distributed synchronous and asynchronous algorithms for semidefinite programming with diagonal constraints," *IEEE Transactions on Automatic Control*, vol. 68, no. 2, pp. 1007–1022, 2023.

[9] P. Giselsson, M. D. Doan, T. Keviczky, B. D. Schutter, and A. Rantzer, "Accelerated gradient methods and dual decomposition in distributed model predictive control," *Automatica*, vol. 49, no. 3, pp. 829–833, 2013.

[10] Z. Song, W. Li, K. Jin, L. Shi, M. Yan, W. Yin, and K. Yuan, "Communication-efficient topologies for decentralized learning with $o(1)$ consensus rate," in *Advances in Neural Information Processing Systems*, vol. 35, 2022.

[11] J. F. Mota, J. M. Xavier, P. M. Aguiar, and M. Puschel, "Distributed optimization with local domains: Applications in MPC and network flows," *IEEE Transactions on Automatic Control*, vol. 60, no. 7, pp. 2004–2009, 2015.

[12] S. A. Alghunaim, K. Yuan, and A. H. Sayed, "A proximal diffusion strategy for multiagent optimization with sparse affine constraints," *IEEE Transactions on Automatic Control*, vol. 65, no. 11, pp. 4554–4567, 2020.

[13] S. A. Alghunaim and A. H. Sayed, "Distributed coupled multiagent stochastic optimization," *IEEE Transactions on Automatic Control*, vol. 65, no. 1, pp. 175–190, 2020.

[14] P. Rebeschini and S. Tatikonda, "Locality in network optimization," *IEEE Transactions on Control of Network Systems*, vol. 6, 2019.

[15] R. Brown, F. Rossi, K. Solovey, M. Tsao, M. T. Wolf, and M. Pavone, "On local computation for network-structured convex optimization in multi-agent systems," *IEEE Transactions on Control of Network Systems*, 2021.

[16] J. Xu, S. Zhu, Y. C. Soh, and L. Xie, "Augmented distributed gradient methods for multi-agent optimization under uncoordinated constant stepsizes," in *2015 54th IEEE Conference on Decision and Control (CDC)*, 2015, pp. 2055–2060.

[17] 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.

[18] P. D. 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.

[19] M. Bianchi and S. Grammatico, "Fully distributed Nash equilibrium seeking over time-varying communication networks with linear convergence rate," *IEEE Control Systems Letters*, vol. 5, no. 2, pp. 499–504, 2021.

[20] J. C. Parinya and Fakcharoenphol, "Simple distributed algorithms for approximating minimum Steiner trees," L. Wang, Ed. Springer Berlin Heidelberg, 2005, pp. 380–389.

[21] 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 Trends in Machine Learning*, vol. 3, 2010.

[22] 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.

[23] C. A. Uribe, S. Lee, A. Gasnikov, and A. Nedić, "A dual approach for optimal algorithms in distributed optimization over networks," *Optimization Methods and Software*, vol. 36, pp. 1–37, 2021.

[24] 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.

[25] 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.

[26] G. Qu and N. Li, "Harnessing smoothness to accelerate distributed optimization," *IEEE Transactions on Control of Network Systems*, vol. 5, pp. 159–166, 2018.

[27] A. Falsone, I. Notarnicola, G. Notarstefano, and M. Prandini, "Tracking-ADMM for distributed constraint-coupled optimization," *Automatica*, vol. 117, p. 108962, 2020.

[28] X. Li, G. Feng, and L. Xie, "Distributed proximal algorithms for multiagent optimization with coupled inequality constraints," *IEEE Transactions on Automatic Control*, vol. 66, no. 3, pp. 1223–1230, 2021.

[29] M. Charikar, C. Chekuri, T. yat Cheung, Z. Dai, A. Goel, S. Guha, and M. Li, "Approximation algorithms for directed Steiner problems," *Journal of Algorithms*, vol. 33, pp. 73–91, 1999.