Distributed Big-Data Optimization via Block Communications

Ivano Notarnicola*, Ying Sun*, Gesualdo Scutari, Giuseppe Notarstefano

***Abstract*—We study distributed multi-agent large-scale optimization problems, wherein the cost function is composed of a smooth possibly nonconvex sum-utility plus a DC (Difference-of-Convex) regularizer. We consider the scenario where the dimension of the optimization variables is so large that optimizing and/or transmitting the entire set of variables could cause unaffordable computation and communication overhead. To address this issue, we propose** BLOCK-SONATA**, the first distributed algorithm whereby agents optimize and communicate only a portion of their local variables. The scheme hinges on successive convex approximation (SCA) to handle the nonconvexity of the objective function, coupled with a novel block-signal tracking scheme, aiming at locally estimating the average of the agents' gradients. Asymptotic convergence to stationary solutions of the nonconvex problem is established. Numerical results on a sparse regression problem show the effectiveness of the proposed algorithm and the impact of the block size on its practical convergence speed and communication cost.**

I. INTRODUCTION

We consider a multi-agent system composed of N agents that cooperatively aim at solving the following (possibly nonconvex) optimization problem:

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & U(\mathbf{x}) \triangleq \sum_{i=1}^{N} f_i(\mathbf{x}) + \sum_{\ell=1}^{B} \underbrace{r_\ell^+(\mathbf{x}_\ell) - r_\ell^-(\mathbf{x}_\ell)}_{r_\ell(\mathbf{x}_\ell)} \\ \text{subject to} \quad & \mathbf{x} = [\mathbf{x}_1^\top, \ldots, \mathbf{x}_B^\top]^\top \\ & \mathbf{x}_\ell \in \mathcal{K}_\ell, \quad \forall \ell \in \{1, \ldots, B\}, \end{aligned} \tag{P}$$

where $\mathbf{x} \in \mathbb{R}^{dB}$ is the vector of the optimization variables, partitioned in B blocks, whose ℓ-th block is denoted by $\mathbf{x}_\ell \in \mathbb{R}^d$; $f_i : \mathbb{R}^{dB} \to \mathbb{R}$ is a smooth possibly nonconvex cost function of agent i; $r_\ell : \mathbb{R}^d \to \mathbb{R}$, $\ell \in \{1, \ldots, B\}$, is a difference of convex (DC) function commonly known by all the agents; and $\mathcal{K}_\ell$, $\ell \in \{1, \ldots, B\}$, is a closed convex set. Function r_ℓ usually plays the role of a regularizer, used to promote some favorable structure on the solution $\mathbf{x}$, such as sparsity. The DC structure of r_ℓ is motivated by the need of capturing in a unified formulation both convex and nonconvex regularizers, the latter being shown to achieve superior performance than their convex counterparts [1]. Problem (P) is of broad interest and models a wide range of applications including network resource allocation, target localization, as well as statistical learning problems.

Our goal is to design a distributed algorithm solving large-scale instances of (P). These problems, also referred to as big-data problems, pose the following two challenges: (1) optimizing the objective function, or even just computing the gradient with respect to all the variables, can be too costly; (2) broadcasting over the network at each iteration all agents' local variables would incur in an unaffordable communication overhead. We are not aware of any work in the literature that can address both challenges (1) and (2) for problem (P). In fact, as discussed next, the existing distributed algorithms either call for the optimization and transmission of the entire vector $\mathbf{x}$ per iteration (or auxiliary variables of the same size of $\mathbf{x}$) or impose restrictive structures on the objective function to work.

There is a vast literature of distributed algorithms for both convex [2]–[10] and nonconvex problems [11]–[16]. Although substantially different, these methods are all based on two main steps, namely: a local optimization and then a communication step of the *entire* vector $\mathbf{x}$ (or some related variables of the same size, e.g., multipliers). They thus fail to address challenges (1) and (2). On the other hand, (block) coordinate descent methods [4], [17]–[19] and parallel algorithms [20]–[23] have been shown to be quite effective in handling large-scale problems by optimizing one block of the variables per time. These algorithms, however, are not readily implementable in the aforementioned distributed setting, because they either assume that all agents know the whole sum-utility or that, at each iteration, each agent has access to the current value of the other agents' variables. While these assumptions are naturally satisfied in a share-memory system (e.g., data centric architecture) or complete (graph) networks, if enforced for problem (P), they would call for an heavy message passing among the agents. We are aware of only a few distributed schemes operating on block variables, namely: [24]–[26]. They however require a certain degree of graph separability on the sum-utility function, meaning that each agent's function f_i can depend only on the variables of that agent and its neighbors, which makes them not applicable to problem (P).

In this work, we propose BLOCK-SONATA, the first distributed algorithm for the general class of problem (P) that is able to address both challenges (1) and (2). Leveraging the block separable structure of (P), each agent iteratively optimizes and transmits only one block of its local copy of $\mathbf{x}$. More specifically, BLOCK-SONATA consists of two steps, namely: 1) a local optimization step wherein agents locally solve a covexification of (P), with respect to a chosen block of their local variables; and 2) a *blockwise* consensus step, aiming at forcing an agreement among the agents' local copies. Moreover, it also employs a novel blockwise signal tracking scheme instrumental to dynamically estimate the gradient of the sum-utility function, using only local information. Agents select the blocks to optimize and then transmit in a totally uncoordinated fashion. Asymptotic convergence is established under mild assumptions. Compared to our recent proposal [27], BLOCK-SONATA is computationally more efficient, since it does not require at each iteration the computation of the entire gradient of each function f_i.

II. PROBLEM SET-UP

We study problem (P) under the following assumptions.

Assumption 2.1 (On Problem (P)*):*

(i) $\mathcal{K}_\ell \neq \emptyset$ is closed and convex;

(ii) $f_i : \mathbb{R}^{dB} \to \mathbb{R}$ is C^1 on (an open set containing) $\mathcal{K}$;

(iii) ∇f_i is L_i-Lipschitz continuous and bounded on $\mathcal{K}$;

(iv) $r_\ell^+ : \mathbb{R}^d \to \mathbb{R}$ is convex (possibly nonsmooth) on $\mathcal{K}$, with bounded subgradients on $\mathcal{K}$; and $r_\ell^- : \mathbb{R}^d \to \mathbb{R}$ is convex on $\mathcal{K}$, with Lipschitz continuous bounded gradient;

(v) U is coercive on $\mathcal{K}$, i.e., $\lim_{\mathbf{x}\in\mathcal{K}, \|\mathbf{x}\|\to\infty} U(\mathbf{x}) = \infty$.

*These authors equally contributed and are in alphabetic order.

The work of Notarnicola and Notarstefano has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 638992 - OPT4SMART). The work of Sun and Scutari was supported by the USA NSF Grants CIF 1632599 and CAREER Award 1555850; and in part by the ONR Grant N00014-16-1-2244.

Ivano Notarnicola and Giuseppe Notarstefano are with the Department of Engineering, Università del Salento, Lecce, Italy, name.lastname@unisalento.it.

Ying Sun and Gesualdo Scutari are with the School of Industrial Engineering, Purdue University, West-Lafayette, IN, USA, {sun578,gscutari}@purdue.edu.

Assumption 2.1 is standard and can be easily satisfied in practice [23]. We remark that both the local cost f_i and the common regularizer $\sum_{\ell=1}^B r_\ell$ are allowed to be nonconvex.

On the communication network: The communication among the agents is modeled by a fixed, directed graph $\mathcal{G} = (\{1,\dots,N\}, \mathcal{E})$, where $\mathcal{E} \subseteq \{1,\dots,N\} \times \{1,\dots,N\}$ is the set of edges. There is an edge $(i,j) \in \mathcal{E}$ if agent i can send a message to agent j. We denote by $\mathcal{N}_i$ the set of *in-neighbors* of node i in $\mathcal{G}$, i.e., $\mathcal{N}_i \triangleq \{j \in \{1,\dots,N\} \mid (j,i) \in \mathcal{E}\}$. We assume that $\mathcal{E}$ contains self-loops and, thus, $\mathcal{N}_i$ contains $\{i\}$ itself. To let information propagate over the network, we impose the following assumption on $\mathcal{G}$.

Assumption 2.2 (Network connectivity): The graph $\mathcal{G}$ is strongly connected.

The above setting and problem are quite general and model many applications of interest. An example in the context of signal estimation is given next.

`Sparse regression`**:** Consider the problem of estimating a sparse signal $\mathbf{x}_0$ from linear measurements $\{\mathbf{b}_i\}_{i=1}^N$, where $\mathbf{b}_i = \mathbf{D}_i\mathbf{x}_0 + \mathbf{n}_i$ with $\mathbf{n}_i$ being the measurement noise at agent i's side. The problem can be formulated as

$$\underset{\mathbf{x}\in\mathcal{K}}{\text{minimize}} \quad \sum_{i=1}^N \|\mathbf{b}_i - \mathbf{D}_i\mathbf{x}\|^2 + R(\mathbf{x}), \tag{1}$$

where $R : \mathbb{R}^{dB} \to \mathbb{R}$ is a sparsity-promoting regularizer having the structure $R(\mathbf{x}) \triangleq \sum_{k=1}^{dB} r(x_k)$. The DC structure of R is motivated by the fact that both convex regularizers (e.g., ℓ_1, ℓ_2, and elastic net) and the widely used nonconvex regularizers (e.g., SCAD, Log, Exp, ℓ_p norm for $0 < p < 1$) can be written as

$$r(x) \triangleq \underbrace{\eta(\theta)|x|}_{r^+(x)} - \underbrace{(\eta(\theta)|x| - r(x))}_{r^-(x)},$$

where $r^- : \mathbb{R} \to \mathbb{R}$ is a convex function with Lipschitz continuous derivative. Problem (1) is clearly an instance of Problem (P).

III. Algorithmic Design

Before describing the proposed distributed algorithm, we introduce a block-wise dynamic average consensus scheme, where the agents aim at cooperatively tracking the average of a time-varying signal via block-wise communications.

A. Average signal tracking via block communication

We consider the problem of tracking the average of a signal over a graph $\mathcal{G}$ satisfying Assumption 2.2. Each agent i can evaluate locally a time-varying signal $\{\mathbf{u}_i^t\}_{t\in\mathbb{N}}$, and the agents aim at tracking the average signal $\bar{\mathbf{u}}^t \triangleq \frac{1}{N}\sum_{i=1}^N \mathbf{u}_i^t$ by exchanging information over the network. We assume that the cost of acquiring $\mathbf{u}_i^t$ is non-negligible, e.g., $\mathbf{u}_i^t$ can be the gradient of a function with respect to a large number of variables. Distributed tracking has been studied in [16]. However, such a scheme requires the acquisition and communication at each iteration of the entire signal $\mathbf{u}_i^t$, which is too costly. To cope with the curse of dimensionality, we develop next a signal tracking scheme that operates at the level of the blocks of signals $\mathbf{u}_i^t$.

Each agent i maintains a local variable $\mathbf{x}_{(i)}^t$, whose ℓ-th block is denoted by $\mathbf{x}_{(i,\ell)}^t$, with $\ell \in \{1,\dots,B\}$. At iteration t, each agent i picks a block-index, say ℓ_i^t, and broadcasts the block $\mathbf{x}_{(i,\ell_i^t)}^t$ to its neighbors. Based on the information (blocks) received from its neighbors and the acquired block of the local signal $\mathbf{u}_i^t$, agent i updates block-wise its entire vector $\mathbf{x}_{(i)}^t$ (according to the mechanism that we will introduce shortly). Since there is no coordination among the agents, they will likely transmit blocks associated with different indices. This implies that blocks with different index will "travel" on different communication graphs, which in general do not coincide with $\mathcal{G}$: agent j is an in-neighbor of i if $j \in \mathcal{N}_i$ and agent j sends block ℓ to i at iteration t. This naturally suggests the adoption of block-dependent communication graphs, one per block ℓ. Specifically, $\mathcal{G}_\ell^t \triangleq (\{1,\dots,N\}, \mathcal{E}_\ell^t)$, which is a *time-varying* subgraph of $\mathcal{G}$ associated to block ℓ at iteration t, whose edge set is defined as $\mathcal{E}_\ell^t \triangleq \{(j,i) \in \mathcal{E} \mid j \in \mathcal{N}_{i,\ell}^t, i \in \{1,\dots,N\}\}$, where $\mathcal{N}_{i,\ell}^t$ is the in-neighborhood of agent i associated with the block-index ℓ, $\mathcal{N}_{i,\ell}^t \triangleq \{j \in \mathcal{N}_i \mid \ell_j^t = \ell\} \cup \{i\} \subseteq \mathcal{N}_i$.

By using block-dependent graphs one can solve the tracking problem block-wise. Therefore, in the following, we focus only on block ℓ, without loss of generality. The task reduces to developing a tracking algorithm over the time-varying directed graph $\{\mathcal{G}_\ell^t\}_{t\in\mathbb{N}}$. Building on [16], we propose the following adapt-then-combine scheme:

$$\begin{aligned}
\mathbf{v}_{(i,\ell)}^t &= \mathbf{x}_{(i,\ell)}^t + \frac{1}{\phi_{(i,\ell)}^t}(\mathbf{u}_{i,\ell}^{t+1} - \mathbf{u}_{i,\ell}^t) \\
\phi_{(i,\ell)}^{t+1} &= \sum_{j\in\mathcal{N}_{i,\ell}^t} a_{ij\ell}^t \phi_{(j,\ell)}^t, \quad \phi_{(i,\ell)}^0 = 1, \qquad \forall \ell \in \{1,\dots,B\}, \\
\mathbf{x}_{(i,\ell)}^{t+1} &= \frac{1}{\phi_{(i,\ell)}^{t+1}} \sum_{j\in\mathcal{N}_i} a_{ij\ell}^t \phi_{(j,\ell)}^t \mathbf{v}_{(j,\ell)}^t.
\end{aligned} \tag{2}$$

The above scheme can be interpreted as follows: each agent first moves its local estimate towards the current signal $\mathbf{u}_{i,\ell}^{t+1}$, then averages it with the value of the neighbors. The extra scalar variable $\phi_{(i,\ell)}$ is introduced to balance the digraph and form a convex combination of the received $\mathbf{x}_{(i,\ell)}$'s through equivalent weights.

While the tracking scheme (2) unlocks block communications, it still requires, at each iteration, the acquisition of the entire signal $\mathbf{u}_i^t$. To cope with this issue, we propose to replace $\mathbf{u}_i^t$ with a surrogate local variable, denoted by $\widehat{\mathbf{u}}_i^t$, initialized as $\widehat{\mathbf{u}}_i^0 = \mathbf{u}_i^0$. At iteration t, agent i acquires only a block of signal $\mathbf{u}_i^t$, say block ℓ_i^t for notation simplicity, and updates $\widehat{\mathbf{u}}_i^t$ as

$$\widehat{\mathbf{u}}_{i,\ell}^t = \begin{cases} \mathbf{u}_{i,\ell}^t, & \text{if } \ell = \ell_i^t \\ \widehat{\mathbf{u}}_{i,\ell}^{t-1}, & \text{if } \ell \neq \ell_i^t \end{cases} \tag{3}$$

where $\mathbf{u}_{i,\ell}^t$ [resp. $\widehat{\mathbf{u}}_{i,\ell}^t$] denotes the ℓ-th block of $\mathbf{u}_i^t$ [resp. $\widehat{\mathbf{u}}_i^t$]. That is, vector $\widehat{\mathbf{u}}_i^t$ collects agent i's most recent information on signal $\mathbf{u}_i^t$.

To summarize, the proposed block-tracking scheme reads as (2), where $\mathbf{u}_{i,\ell}^t$ [resp. $\mathbf{u}_{i,\ell}^{t+1}$] is replaced by $\widehat{\mathbf{u}}_{i,\ell}^t$ [resp. $\widehat{\mathbf{u}}_{i,\ell}^{t+1}$], defined in (3). To ensure $\lim_{t\to\infty} \|\mathbf{x}_{(i,\ell)}^t - \widehat{\mathbf{u}}_\ell^t\| = 0$, we need the following standard assumptions for push-sum-like algorithms on the connectivity of $\{\mathcal{G}_\ell^t\}_{t\in\mathbb{N}}$ and weight matrix $\mathbf{A}_\ell^t$.

Assumption 3.1: For all $\ell \in \{1,\dots,B\}$, there exists a finite integer $T > 0$ such that graph sequence $\{\mathcal{G}_\ell^t\}_{t\in\mathbb{N}}$ is T-strongly connected, i.e., the union graph $\cup_{s=t}^{t+T-1}\mathcal{G}_\ell^s$ is strongly connected, for all $t > 0$.

Assumption 3.2 (On the Weighting Matrix $\mathbf{A}_\ell^t$): For all $\ell \in \{1,\dots,B\}$, matrix $\mathbf{A}_\ell^t$ satisfies the following conditions:

(i) $a_{ii\ell}^t \geq \vartheta > 0$, for all $i \in \{1,\dots,N\}$;
(ii) $a_{ij\ell}^t \geq \vartheta > 0$, for all $(j,i) \in \mathcal{E}_\ell^t$;
(iii) $\mathbf{A}_\ell^t$ is column stochastic, i.e., $\mathbf{1}^\top \mathbf{A}_\ell^t = \mathbf{1}^\top$, for all $t > 0$.

Since each digraph $\mathcal{G}_\ell^t$ is induced by the used block selection rule, its connectivity clearly depends on it. Two open question, addressed next, are thus how to design the block selection rule and $\mathbf{A}_\ell^t$ that fulfill Assumption 3.1 and 3.2.

By the definition of $\mathcal{G}_\ell^t$, all the edges in the underlying graph $\mathcal{G}$ leaving node i will be also edges of $\mathcal{G}_\ell^t$ if agent i sends block ℓ at time t. Since $\mathcal{G}$ is strongly connected (cf. Assumption 2.2), $\mathcal{G}_\ell^t$ is T-strongly connected if, starting from any time $t > 0$, all the agents

send block ℓ within T iterations, which translates in the following essentially cyclic block selection rule.

Assumption 3.3 (Block Updating Rule): For each agent $i \in \{1,\dots,N\}$, there exists a (finite) constant $T_i > 0$ such that $\cup_{s=0}^{T_i-1}\{\ell_i^{t+s}\} = \{1,\dots,B\}$, for all $t \geq 0$.

Note that the above rule does not impose any coordination among the agents: at each iteration, different agents may update different blocks. It is not difficult to show that, under Assumption 2.2 and 3.3, there exits a $0 < T \leq \max_{i\in\{1,\dots,N\}} T_i$, such that $\cup_{s=0}^{T-1}\mathcal{G}_\ell^{t+s}$, $\ell \in \{1,\dots,B\}$, is strongly connected, for all $t \geq 0$.

We show next how nodes can *locally* build a matrix $\mathbf{A}_\ell^t$ satisfying Assumption 3.2 for each time-varying, directed graph $\mathcal{G}_\ell^t$. Observe that at iteration t, if agent j selects block ℓ, it sends $\mathbf{v}_{(j,\ell)}^t$ to any agent i that is its out-neighbor; or send it to no one, otherwise. In addition, $a_{jj\ell}^t$ must be nonzero by Assumption 3.2. Consequently, the j-th column of $\mathbf{A}_\ell^t$, denoted by $\mathbf{A}_\ell^t(:,j)$, can only have the following two possible sparsity patterns: (i) all $a_{ij\ell}^t$, with $i \in \{\{1,\dots,B\} : (j,i) \in \mathcal{E}\}$, is nonzero if $\ell_j^t = \ell$; (ii) only $a_{jj\ell}^t$ is nonzero if $\ell_j^t \neq \ell$. To meet the requirement that $\mathbf{A}_\ell^t$ is column stochastic, agent j thus either select a stochastic vector $\mathbf{A}_\ell^t(:,j)$ matching the sparsity pattern described in case (i), if $\ell_j^t = \ell$; or set $\mathbf{A}_\ell^t(:,j)$ to be the j-th vector of the canonical basis, if $\ell_j^t \neq \ell$.

We conclude this section, noting that the proposed block-trackig scheme can be used also to solve the average consensus problem wherein agents aim to estimate the average of their initial value, i.e., $\frac{1}{N}\sum_{i=1}^N \mathbf{x}_{(i)}^0$. Specifically, by reinterpreting the consensus problem as tracking of the average of the constant signal $\mathbf{x}^0 \triangleq [\mathbf{x}_{(1)}^{0\top},\dots,\mathbf{x}_{(N)}^{0\top}]^\top$, it is enough to set in (2) $\mathbf{u}_i^t \equiv \mathbf{x}^0$ and absorb the v-variable, which leads to the following *block-consensus* algorithm:

$$\begin{aligned}\phi_{(i,\ell)}^{t+1} &= \sum_{j\in\mathcal{N}_i} a_{ij\ell}^t \phi_{(j,\ell)}^t,\\ \mathbf{x}_{(i,\ell)}^{t+1} &= \frac{1}{\phi_{(i,\ell)}^{t+1}}\sum_{j\in\mathcal{N}_i} a_{ij\ell}^t \phi_{(j,\ell)}^t \mathbf{x}_{(j,\ell)}^t,\end{aligned} \qquad \forall \ell \in \{1,\dots,B\}. \quad (4)$$

B. BLOCK-SONATA*: A constructive approach*

We are now in the position to introduce our algorithm, BLOCK-SONATA. Observe that in problem (P) it is the common variable $\mathbf{x}$ that couples the cost function f_i. Therefore to decouple the problem we introduce for each agent i a local copy $\mathbf{x}_{(i)}$ of $\mathbf{x}$. Yet, agent i faces the following challenge in (P) w.r.t. its $\mathbf{x}_{(i)}$: (1) the dimension of $\mathbf{x}_{(i)}$ is large; (2) f_i and $-r_\ell$ is nonconvex; and (3) $\sum_{j\neq i} f_j$ is unknown. We address these issues by devising BLOCK-SONATA as an iterative procedure leveraging the SCA optimization techniques, coupled with a parallel blockwise consensus/tracking step based on (2) and (4), as described next.

Local optimization: At iteration t, agent i selects and optimizes a block of $\mathbf{x}_{(i)}^t$, say ℓ_i^t [this addresses challenge (1)]. To deal with the nonconvexity of f_i and $-g_{\ell_i^t}$, we approximate these functions, respectively, by a strongly convex surrogate $\tilde{f}_{i,\ell_i^t} : \mathbb{R}^d \to \mathbb{R}$ and its linearization at point $\mathbf{x}_{(i,\ell_i^t)}^t$ [challenge (2)]. The unknown term $\sum_{j\neq i} f_j$ is replaced by a linear function whose coefficient $\widetilde{\boldsymbol{\pi}}_{(i,\ell_i^t)}^t$ aims to track $\sum_{j\neq i}\nabla f_{j,\ell_i^t}(\mathbf{x}_{(i)}^t)$ [challenges (3)]. The resulting problem (6) is thus a strongly convex approximation of problem (P) and admits a unique solution $\widetilde{\mathbf{x}}_{(i,\ell_i^t)}^t$. Agent i then moves its local copy $\mathbf{x}_{(i,\ell_i^t)}^t$ along direction $\widetilde{\mathbf{x}}_{(i,\ell_i^t)}^t - \mathbf{x}_{(i,\ell_i^t)}^t$ with step-size γ^t as in (7). Note that agent i does not optimize blocks $\ell \neq \ell_i^t$, hence we let $\mathbf{v}_{(i,\ell)}^t = \mathbf{x}_{(i,\ell)}^t$, $\forall \ell \neq \ell_i^t$. Agent i then transmits $\mathbf{v}_{(i,\ell_i^t)}^t$ to its neighbors.

Distributed Algorithm BLOCK-SONATA

Set $t=0$, $\boldsymbol{\phi}_{(i)}^0 = \mathbf{1}$, $\widehat{\mathbf{g}}_i^0 = \mathbf{y}_{(i)}^0 = \nabla f_i(\mathbf{x}_{(i)}^0)$, $\ell_i^0 \in \{1,\dots,B\}$.

Local Optimization:

$$\widetilde{\boldsymbol{\pi}}_{(i,\ell_i^t)}^t = N\cdot\mathbf{y}_{(i,\ell_i^t)}^t - \nabla_{\ell_i^t} f_i\big(\mathbf{x}_{(i)}^t\big) \quad (5)$$

$$\widetilde{\mathbf{x}}_{(i,\ell_i^t)}^t \triangleq \operatorname*{argmin}_{\mathbf{x}_{\ell_i^t}\in\mathcal{K}_{\ell_i^t}} r_{\ell_i^t}^+(\mathbf{x}_{\ell_i^t}) + \widetilde{f}_{i,\ell_i^t}(\mathbf{x}_{\ell_i^t};\mathbf{x}_{(i)}^t) + \big(\widetilde{\boldsymbol{\pi}}_{(i,\ell_i^t)}^t - \nabla r_{\ell_i^t}^-(\mathbf{x}_{(i,\ell_i^t)}^t)\big)^\top(\mathbf{x}_{\ell_i^t} - \mathbf{x}_{(i,\ell_i^t)}^t) \quad (6)$$

$$\mathbf{v}_{(i,\ell_i^t)}^t = \mathbf{x}_{(i,\ell_i^t)}^t + \gamma^t(\widetilde{\mathbf{x}}_{(i,\ell_i^t)}^t - \mathbf{x}_{(i,\ell_i^t)}^t) \quad (7)$$

Broadcast $\mathbf{v}_{(i,\ell_i^t)}^t$, $\phi_{(j,\ell)}^t$, $\mathbf{y}_{(j,\ell_i^t)}^t$ to the out-neighbors

Averaging and Gradient Tracking:

For $\ell\in\{1,\dots,B\}$: receive $\phi_{(j,\ell)}^t$,$\mathbf{v}_{(j,\ell)}^t$ from $j\in\mathcal{N}_{i,\ell}^t$

$$\phi_{(i,\ell)}^{t+1} = \sum_{j\in\mathcal{N}_i} a_{ij\ell}^t\,\phi_{(j,\ell)}^t \quad (8)$$

$$\mathbf{x}_{(i,\ell)}^{t+1} = \frac{1}{\phi_{(i,\ell)}^{t+1}}\sum_{j\in\mathcal{N}_i} a_{ij\ell}^t\,\phi_{(j,\ell)}^t\mathbf{v}_{(j,\ell)}^t \quad (9)$$

Select $\ell_i^{t+1} \in \{1,\dots,B\}$ and update

$$\widehat{\mathbf{g}}_{i,\ell}^{t+1} = \begin{cases}\nabla_{\ell_i^{t+1}} f_i(\mathbf{x}_{(i)}^{t+1}) & \text{if } \ell = \ell_i^{t+1}\\ \widehat{\mathbf{g}}_{i,\ell}^t & \text{otherwise}\end{cases} \quad (10)$$

For $\ell\in\{1,\dots,B\}$: receive $\phi_{(j,\ell)}^t\mathbf{y}_{(j,\ell)}^t + \widehat{\mathbf{g}}_{j,\ell}^{t+1} - \widehat{\mathbf{g}}_{j,\ell}^t$ from $j\in\mathcal{N}_{i,\ell}^t$

$$\mathbf{y}_{(i,\ell)}^{t+1} = \frac{1}{\phi_{(i,\ell)}^{t+1}}\sum_{j\in\mathcal{N}_i} a_{ij\ell}^t\left(\phi_{(j,\ell)}^t\mathbf{y}_{(j,\ell)}^t + \widehat{\mathbf{g}}_{j,\ell}^{t+1} - \widehat{\mathbf{g}}_{j,\ell}^t\right) \quad (11)$$

Blockwise consensus/gradient tracking: To force consensus on $\mathbf{x}_{(i)}^t$, agent i update in parallel $\mathbf{x}_{(i,\ell)}^t$, $\ell \in \{1,\dots,B\}$ based on received variables $\mathbf{v}_{(j,\ell_j^t)}^t$. Naturally, we apply the block consensus scheme (4), which leads to updates (8) and (9).

Finally, we need to introduce the update of $\widetilde{\boldsymbol{\pi}}_{(i,\ell)}^t$ so that $\lim_{t\to\infty}\|\widetilde{\boldsymbol{\pi}}_{(i,\ell)}^t - \sum_{j\neq i}\nabla f_{j,\ell}(\mathbf{x}_{(i)}^t)\| = 0$. To this end, we rewrite $\sum_{j\neq i}\nabla f_{j,\ell}(\mathbf{x}_{(i)}^t)$ as

$$\sum_{j\neq i}\nabla f_{j,\ell}(\mathbf{x}_{(i)}^t) = N\cdot\underbrace{\frac{1}{N}\sum_{j=1}^N\nabla f_{j,\ell}(\mathbf{x}_{(i)}^t)}_{\overline{\nabla f}_\ell(\mathbf{x}_{(i)}^t)} - \nabla f_{i,\ell}(\mathbf{x}_{(i)}^t).$$

Since $\nabla f_{i,\ell}(\mathbf{x}_{(i)}^t)$ can be evaluated locally by agent i, the task boils down to estimate the average gradient $\overline{\nabla f}_\ell(\mathbf{x}_{(i)}^t)$, $\forall \ell \in \{1,\dots,B\}$. We can then readily invoke the blockwise tracking scheme (2) with ℓ_i^{t+1} selected according to the essentially cyclic rule (Assumption 3.3) and $\mathbf{u}_i^t \triangleq \nabla f_i(\mathbf{x}_{(i)}^t)$, leading to the updates (10) and (11).

Remark 3.4: Note that the block selected in the tracking step (10) need not to be the same as the one selected in the optimization step (5). However, in BLOCK-SONATA we let them equal so that the gradient of f_i only need to be evaluated with respect to one block.

Having introduced the algorithm, the remaining question is how to choose surrogate function $\tilde{f}_i$ and step-size γ^t. Convergence of BLOCK-SONATA is guaranteed under the following assumptions.

Assumption 3.5 (On the Surrogate Functions): Given problem (P) under Assumption 2.1, each surrogate function $\tilde{f}_{i,\ell} : \mathcal{K}_\ell \times \mathcal{K} \to \mathbb{R}$ is chosen so that

(i) $\tilde{f}_{i,\ell}(\bullet;\mathbf{x})$ is uniformly strongly convex on $\mathcal{K}_\ell$;
(ii) $\nabla\tilde{f}_{i,\ell}(\mathbf{x}_\ell;\mathbf{x}) = \nabla_\ell f_i(\mathbf{x})$, for all $\mathbf{x}\in\mathcal{K}$;
(iii) $\nabla\tilde{f}_{i,\ell}(\mathbf{x}_\ell;\bullet)$ is uniformly Lipschitz continuous on $\mathcal{K}$;

where $\nabla \tilde{f}_{i,\ell}$ denotes the partial gradient of $\tilde{f}_{i,\ell}$ with respect to its first argument.

Assumption 3.6 (On the step-size): The sequence $\{\gamma^t\}$, with each $0 < \gamma^t \leq 1$, satisfies: (i) $\sum_{t=0}^{\infty} \gamma^t = \infty$ and $\sum_{t=0}^{\infty} (\gamma^t)^2 < \infty$; (ii) $\gamma^t/\eta \leq \gamma^{t+1} \leq \gamma^t$, for all $t \geq 0$ and some $\eta \in (0,1)$.

Assumption 3.5 states that $\tilde{f}_i$ should be regarded as a (simple) strongly convex approximation of f_i that preserves its first order properties. Several choices of constructing $\tilde{f}_i$ are available, see e.g., [15], [23]. Assumption 3.6 is the standard diminishing step-size rule (i) with an extra requirement (ii) that ensures all the blocks contribute equally to the optimization. Condition (ii) can be met easily in practice [28], [29], an example is given in Sec. IV. The convergence of BLOCK-SONATA is given in the following theorem, whose proof is omitted due to space limit.

Theorem 3.7: Let $\{(\mathbf{x}_{(i)}^t)_{i=1}^N\}_{t\in\mathbb{N}}$ be the sequence generated by BLOCK-SONATA, and let $\bar{\mathbf{x}}^t \triangleq (1/N)\sum_{i=1}^N \mathbf{x}_{(i)}^t$. Suppose that Assumption 2.1, and 3.1-3.6 are satisfied; then the following hold: (ii) `consensus`: $\|\mathbf{x}_{(i)}^t - \bar{\mathbf{x}}^t\| \to 0$ as $t \to \infty$, for all $i \in \{1,\ldots,N\}$; (i) `convergence`: $\{\bar{\mathbf{x}}^t\}_{t\in\mathbb{N}}$ is bounded and every of its limit points is a stationary solution of problem (P).

Theorem 3.7 states two results: all local copies $\mathbf{x}_{(i)}^t$ converges to their weighted average $\bar{\mathbf{x}}^t$ asymptotically; every limit point of $\bar{\mathbf{x}}^t$ is a stationary point of problem (P).

BLOCK-SONATA enjoys the property that at each iteration agents not only solve a low-dimensional optimization problem, but also transmit a limited amount of information. Moreover, compared to our previous scheme in [27], in BLOCK-SONATA the gradient of f_i are computed only with respect to one block rather than the whole variable, and this further saves local computation cost.

IV. NUMERICAL SIMULATIONS

In this section we test our distributed optimization algorithm on an instance of the sparse regression problem (1), with $\mathcal{K}$ being a box constraint set $\mathcal{K}_\ell \triangleq [k_L, k_U]^d$.

We build as surrogate $\tilde{f}_{i,\ell}$ of f_i (cf. Assumption 3.5) the linearizatin of f_i at the current iterate, i.e.,

$$\begin{aligned}&\tilde{f}_{i,\ell}(\mathbf{x}_{(i,\ell)}; \mathbf{x}_{(i)}^t)\\ &= \left(2\mathbf{D}_{i,\ell}^\top(\mathbf{D}_i\mathbf{b}_i)\right)^\top(\mathbf{x}_{(i,\ell)} - \mathbf{x}_{(i,\ell)}^t) + \frac{\tau_i}{2}\|\mathbf{x}_{(i,\ell)} - \mathbf{x}_{(i,\ell)}^t\|^2\\ &\quad - \sum_{k=1}^d \left(\frac{dr^-((\mathbf{x}_{(i,\ell)}^t)_k)}{dx}(\mathbf{x}_{(i,\ell)} - \mathbf{x}_{(i,\ell)}^t)_k\right),\end{aligned} \tag{12}$$

where x is a scalar variable and, e.g., $(\mathbf{x}_{(i,\ell)}^t)_k$ denotes the k-th scalar component of $\mathbf{x}_{(i,\ell)}^t$. It is worth noting that (12) admits a unique minimizer given by

$$\mathbf{x}_{(i,\ell)}^{t+1} = \mathcal{P}_{\mathcal{K}_\ell}\left(\mathcal{S}_{\frac{\lambda\eta}{\tau_i}}\left\{\mathbf{x}_{(i,\ell)}^t - \frac{1}{\tau_i}(2\mathbf{D}_{i,\ell}^\top(\mathbf{D}_i\mathbf{b}_i) - \mathbf{r}_{i,\ell}^t)\right\}\right)$$

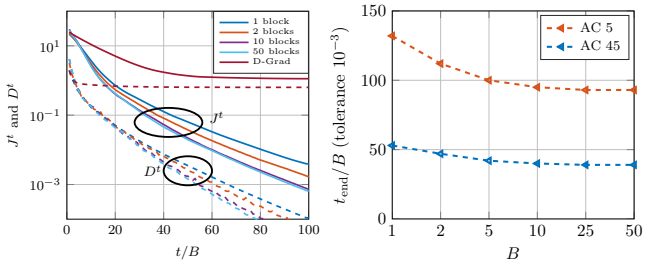
where $\mathbf{r}_{i,\ell}^t \triangleq (\frac{dr^-((\mathbf{x}_{(i,\ell)}^t)_k)}{dx})_{k=1}^d$, $\mathcal{S}_\lambda(\mathbf{x}) \triangleq \text{sign}(\mathbf{x})\cdot\max\{|\mathbf{x}|-\lambda, 0\}$ (operations are performed element-wise), and $\mathcal{P}_{\mathcal{K}_\ell}$ is the Euclidean projection onto $\mathcal{K}_\ell$.

We consider a network of $N = 50$ agents communicating over a fixed undirected graph $\mathcal{G}$ generated using an Erdős-Rényi random model. We compare two extreme topologies: a densely and a poorly-connected one, which have algebraic connectivity equal to 45 and 5, respectively. The variable dimension is large and set to 500, $\mathcal{K}$ is set to be $[-10, 10]^{500}$. Regularizer R is chosen to be the logarithmic function [30] with parameters $\lambda = 0.1$ and $\theta = 10$. The components of the ground-truth signal $\mathbf{x}_0$ are generated independently according to the Gaussian distribution $\mathcal{N}(0,1)$. To impose sparsity on $\mathbf{x}_0$, we set the smallest 80% of the entries of $\mathbf{x}_0$ to zero. Each agent i has a measurement matrix $\mathbf{D}_i \in \mathbb{R}^{50\times 500}$ with i.i.d. $\mathcal{N}(0,1)$ distributed entries (with ℓ_2-normalized rows), and the observation noise $\mathbf{n}_i$ has entries i.i.d. distributed according to $\mathcal{N}(0, 0.5)$. The diminishing step-size γ^t follows the rule $\gamma^t = \gamma^{t-1}(1 - \mu\gamma^{t-1})$ with $\gamma^0 = 0.1$ and $\mu = 10^{-4}$. The proximal parameter is $\tau_i = 5$ for the poorly connected example and $\tau_i = 1$ for the densely connected one. The algorithm parameters have been tuned for each of the topologies to yield the best performance.

To evaluate the algorithmic performance, we use two merit functions. The first one measures the distance from stationarity of the average of the agents' iterates $\bar{\mathbf{x}}^t = \frac{1}{N}\sum_{i=1}^N \mathbf{x}_{(i)}^t$, and is given by $J^t \triangleq \|\bar{\mathbf{x}}^t - \mathcal{P}_{\mathcal{K}}\left(\mathcal{S}_{\eta\lambda}\left(\bar{\mathbf{x}}^t - (\sum_{i=1}^N \nabla f_i(\bar{\mathbf{x}}^t) - r(\bar{\mathbf{x}}^t))\right)\right)\|_\infty$. The second merit function quantifies the consensus disagreement at each iteration, and is defined as $D^t \triangleq \max_{i\in\{1,\ldots,N\}} \|\mathbf{x}_{(i)}^t - \bar{\mathbf{x}}^t\|$.

We compare our algorithm with a non-block-wise distributed gradient algorithm. We basically adapted the gradient-push in [2] to a constrained nonconvex problem according to the protocol proposed in [31].[1] The performance of BLOCK-SONATA for different choices of the block dimension are reported in Fig. 1 (a). Recalling that t is the iteration counter used in the algorithm description, to fairly compare the algorithm runs for different block sizes, we plot J^t and D^t, versus the normalized number of iterations t/B.

The figure shows that for all runs (with different block sizes), both consensus and stationarity have been achieved by BLOCK-SONATA within 100 normalized iterations, while the plain gradient scheme using all the blocks is much slower. Let t_{end} be the completion time up to a tolerance of 10^{-3}, i.e., the iteration counter of the distributed algorithm such that $J^{t_{\text{end}}} < 10^{-3}$. Fig. 1 (b) shows the normalized completion time t_{end}/B versus the number of blocks B. This highlights how the communication cost reduces by increasing the number of blocks.



(a) (b)

Figure 1: (a) optimality measurement J^t (solid) and consensus error D^t (dashed) versus the normalized iteration for several choices of blocks B: Algebraic connectivity equal to 5. (b) Completion time required to obtain $J^t < 10^{-3}$ versus the number of blocks B.

V. CONCLUSION

In this paper we studied non-convex distributed big-data optimization problems and proposed an algorithm, named BLOCK-SONATA, to solve them. The key distinctive feature of the proposed algorithm is a block-wise successive approximation of each local cost function combined with a block-wise scheme to average *both* the local copies of the decision variable *and* the local estimates of the cost-function gradient. Asymptotic convergence to a stationary point of the problem is established, and numerical tests on the sparse regression problem demonstrate the effectiveness of algorithm.

[1] Note that there is no formal proof of convergence for such an algorithm in the nonconvex setting.

REFERENCES

[1] M. Ahn, J.-S. Pang, and J. Xin, "Difference-of-convex learning I: Directional stationarity, optimality, and sparsity," submitted for publication, 2016.

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

[3] A. Nedić, A. Ozdaglar, and P. A. Parrilo, "Constrained consensus and optimization in multi-agent networks," *IEEE Transactions on Automatic Control*, vol. 55, no. 4, pp. 922–938, 2010.

[4] J. Tsitsiklis, D. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE transactions on automatic control*, vol. 31, no. 9, pp. 803–812, 1986.

[5] A. I. Chen and A. Ozdaglar, "A fast distributed proximal-gradient method," in *Proccedings of the 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2012, pp. 601–608.

[6] J. C. Duchi, A. Agarwal, and M. J. Wainwright, "Dual averaging for distributed optimization: Convergence analysis and network scaling," *IEEE Transactions on Automatic control*, vol. 57, no. 3, pp. 592–606, 2012.

[7] 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, no. 1, pp. 1–122, 2011.

[8] J. F. Mota, J. M. Xavier, P. M. Aguiar, and M. Puschel, "D-ADMM: A communication-efficient distributed algorithm for separable optimization," *IEEE Transactions on Signal Processing*, vol. 61, no. 10, pp. 2718–2723, 2013.

[9] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "On the linear convergence of the admm in decentralized consensus optimization." *IEEE Trans. Signal Processing*, vol. 62, no. 7, pp. 1750–1761, 2014.

[10] T.-H. Chang, M. Hong, and X. Wang, "Multi-agent distributed optimization via inexact consensus admm," *IEEE Transactions on Signal Processing*, vol. 63, no. 2, pp. 482–497, 2015.

[11] P. Bianchi and J. Jakubowicz, "Convergence of a multi-agent projected stochastic gradient algorithm for non-convex optimization," *IEEE Transactions on Automatic Control*, vol. 58, no. 2, pp. 391–405, Feb 2013.

[12] T. Tatarenko and B. Touri, "Non-convex distributed optimization," *arXiv preprint arXiv:1512.00895*, 2015.

[13] M. Hong, "Decomposing linearly constrained nonconvex problems by a proximal primal dual approach: Algorithms, convergence, and applications," *arXiv preprint arXiv:1604.00543*, 2016.

[14] P. Bianchi, G. Fort, and W. Hachem, "Performance of a distributed stochastic approximation algorithm," *IEEE Transactions on Information Theory*, vol. 59, no. 11, pp. 7405–7418, 2013.

[15] 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.

[16] Y. Sun and G. Scutari, "Distributed nonconvex optimization for sparse representation," in *Proceedings of the IEEE International Conference on Speech and Signal Processing (ICASSP)*, 2017.

[17] S. J. Wright, "Coordinate descent algorithms," *Mathematical Programming*, vol. 151, no. 1, pp. 3–34, 2015.

[18] Y. Nesterov, "Efficiency of coordinate descent methods on huge-scale optimization problems," *SIAM Journal on Optimization*, vol. 22, no. 2, pp. 341–362, 2012.

[19] A. Mokhtari, A. Koppel, and A. Ribeiro, "Doubly random parallel stochastic methods for large scale learning," in *Proceedings of American Control Conference (ACC)*. IEEE, 2016, pp. 4847–4852.

[20] P. Richtárik and M. Takáč, "Parallel coordinate descent methods for big data optimization," *Mathematical Programming*, pp. 1–52, 2012.

[21] ——, "Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function," *Mathematical Programming*, vol. 144, no. 1-2, pp. 1–38, 2014.

[22] 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.

[23] F. Facchinei, G. Scutari, and S. Sagratella, "Parallel selective algorithms for nonconvex big data optimization," *IEEE Transactions on Signal Processing*, vol. 63, no. 7, pp. 1874–1889, 2015.

[24] I. Notarnicola and G. Notarstefano, "A randomized primal distributed algorithm for partitioned and big-data non-convex optimization," in *IEEE 55th Conference on Decision and Control (CDC)*, 2016, pp. 153–158.

[25] R. Carli and G. Notarstefano, "Distributed partition-based optimization via dual decomposition," in *52nd IEEE Conference on Decision and Control (CDC)*, Dec 2013, pp. 2979–2984.

[26] B. Recht, C. Re, S. Wright, and F. Niu, "Hogwild: A lock-free approach to parallelizing stochastic gradient descent," in *Advances in Neural Information Processing Systems (NIPS)*, 2011, pp. 693–701.

[27] I. Notarnicola, Y. Sun, G. Scutari, and G. Notarstefano, "Distributed big-data optimization via block-iterative convexification and averaging," in *IEEE 56th Conference on Decision and Control (CDC), submitted*, 2017.

[28] L. Cannelli, F. Facchinei, V. Kungurtsev, and G. Scutari, "Asynchronous parallel algorithms for nonconvex big-data optimization: Model and convergence," *arXiv preprint arXiv:1607.04818*, 2016.

[29] ——, "Asynchronous parallel algorithms for nonconvex big-data optimization. part ii: Complexity and numerical results," *arXiv preprint arXiv:1701.04900*, 2017.

[30] J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping, "Use of the zero-norm with linear models and kernel methods," *Journal of machine learning research*, vol. 3, no. Mar, pp. 1439–1461, 2003.

[31] P. Bianchi and J. Jakubowicz, "Convergence of a multi-agent projected stochastic gradient algorithm for non-convex optimization," *IEEE Transactions on Automatic Control*, vol. 58, no. 2, pp. 391–405, 2013.