DISTRIBUTED NONCONVEX OPTIMIZATION OVER TIME-VARYING NETWORKS

Paolo Di Lorenzo[1] *and Gesualdo Scutari*[2]

[1] Dept. of Engineering, University of Perugia, Via G. Duranti 93, 06125, Perugia, Italy.
[2] Dept. of Industrial Engineering and Cyber Center, Purdue University, West Lafayette, IN 47907, USA.
E-mail: paolo.dilorenzo@unipg.it, gscutari@purdue.edu

ABSTRACT

In this paper we introduce a novel algorithmic framework for *nonconvex distributed* optimization in multi-agent networks with *time-varying (nonsymmetric)* topology. The proposed method hinges on successive convex approximation (SCA) techniques while leveraging dynamic consensus as a mechanism to diffuse information: each agent first solves (possibly *inexactly*) a local convex approximation of the nonconvex original problem, and then performs local averaging operations. Asymptotic convergence to (stationary) solutions of the nonconvex problem is established. Finally, the framework is applied to a distributed nonlinear regression problem.

Index Terms— Distributed optimization, nonconvex optimization, time-varying directed graphs.

1. INTRODUCTION

Recent years have witnessed a surge of interest in distributed optimization methods for multi-agent systems. Many such problems can be mathematically formulated as:

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}} \;\; U(\mathbf{x}) \triangleq F(\mathbf{x}) + G(\mathbf{x}) \qquad (1)\\ &\text{subject to} \;\; \mathbf{x} \in \mathcal{K}, \end{aligned}$$

where

$$F(\mathbf{x}) \triangleq \sum_{i=1}^{I} f_i(\mathbf{x}), \qquad (2)$$

with each $f_i : \mathbb{R}^L \to \mathbb{R}$ being the smooth (possibly *nonconvex*, nonseparable) cost function of agent $i \in \{1, \ldots, I\}$; G is a convex (possibly nonsmooth, nonseparable) function; and $\mathcal{K} \subseteq \mathbb{R}^L$ is a closed and convex set. Usually the nonsmooth term is used to promote some extra structure in the solution, e.g. sparsity. Network-structured optimization problems in the form (1) are found widely in several engineering areas, including sensor networks information processing, communication networks, multi-agent control and coordination, and distributed machine learning, just to name a few. Common to these problems is the necessity of performing a completely decentralized computation, due to the large size of the network and volume of data, time-varying network topology, energy constraints, and/or privacy issues. Motivated by these observations, this paper aims to develop a distributed solution method for the general class of *nonconvex* problems (1), in the presence of time-varying non-symmetric topologies and (possibly) inexact updates.

`Related Work:` Distributed solution methods for *convex* instances of Problem (1) have been widely studied in the literature; they are usually either primal (sub)gradient-based methods [1, 2, 3, 4, 5, 6, 7, 8], or primal-dual schemes [9]. All the above prior art focuses *only* on *convex* problems; algorithms developed therein along with their convergence analysis are not applicable to nonconvex problems. We are aware of only two works dealing with distributed algorithms for (special cases of) Problem (1), namely: [10, 11]. In [10], a consensus-based distributed dual-subgradient algorithm was studied. However, the method calls for the solution of possibly difficult nonconvex subproblems, and it does not find (stationary) solutions of the original problem but those of an auxiliary problem, which are not necessarily stationary for the original problem. In [11], the authors studied convergence of a distributed stochastic projection algorithm involving random gossip between agents. However, the scheme is not applicable to Problem (1) when $G \neq 0$. The nonconvex optimization problem in the general form (1) was tackled for the first time in [12]. The framework proposed in [12], termed as in-Network succEssive conveX approximaTion algorithm (NEXT), hinges on SCA methods [13, 14] while leveraging dynamic consensus [15] as a mechanism to propagate the needed information over the network. The contribution of this paper with respect to [12] is twofold, namely: i) we consider time-varying, nonsymmetric topologies; and ii) we allow (possible) inexact updates of the algorithm. Asymptotic convergence of this extended framework to (stationary) solutions of problem (1) is then established. Finally, we customize the proposed framework to solve a distributed nonlinear regression problem over networks.

2. PROBLEM FORMULATION

Consider a network composed of I autonomous agents aiming to cooperatively and distributively solve Problem (1).

Assumption A [On Problem (1)]:

- **(A1)** The set $\mathcal{K}$ is (nonempty) closed and convex;
- **(A2)** Each f_i is C^1 (possibly nonconvex) on $\mathcal{K}$;
- **(A3)** Each ∇f_i is Lipschitz continuous on $\mathcal{K}$;
- **(A4)** Each ∇f_i is bounded on $\mathcal{K}$;
- **(A5)** G is a convex function (possibly nondifferentiable) with bounded subgradients on $\mathcal{K}$;
- **(A6)** U is coercive on $\mathcal{K}$, i.e., $\lim_{\mathbf{x}\in\mathcal{K}, \|\mathbf{x}\|\to\infty} U(\mathbf{x}) = +\infty$.

Assumption A is standard and satisfied by many practical problems. For instance, A3-A4 hold automatically if $\mathcal{K}$ is bounded, whereas A6 guarantees the existence of a solution. Note that f_i's need not be convex. We also make the blanket assumption that each agent i knows only its own cost function f_i, the common G, and the set $\mathcal{K}$.

On network topology: Time is slotted, and at any time-slot n, the network is modeled as a digraph $\mathcal{G}[n] = (\mathcal{V}, \mathcal{E}[n])$, where $\mathcal{V} = \{1, \ldots, I\}$ is the vertex set (i.e., the set of agents), and $\mathcal{E}[n]$ is the set of (possibly) time-varying directed edges. The in-neighborhood of agent i at time n (including node i) is defined as $\mathcal{N}_i^{\text{in}}[n] =$

This work has been supported by the USA National Science Foundation under Grants CMS 1218717, CIF 1564044, and CAREER Award No. 1254739.

$\{j|(j,i) \in \mathcal{E}[n]\} \cup \{i\}$. Associated with each graph $\mathcal{G}[n]$, we introduce (possibly) time-varying weights $w_{ij}[n]$ matching $\mathcal{G}[n]$:

$$w_{ij}[n] = \begin{cases} \theta \in [\vartheta, 1] & \text{if } j \in \mathcal{N}_i^{\text{in}}[n]; \\ = 0 & \text{otherwise,} \end{cases} \quad (3)$$

for some $\vartheta \in (0,1)$, and define $\boldsymbol{W}[n] \triangleq (w_{ij}[n])_{i,j=1}^{I}$. These weights will be used later on in the proposed algorithm.

Assumption B (On the network topology):

(B1) The sequence of graphs $\mathcal{G}[n]$ is B-strongly connected, i.e., there exists an integer $B > 0$ such that the graph $\mathcal{G}[k] = (\mathcal{V}, \mathcal{E}_B[k])$, with $\mathcal{E}_B[k] = \bigcup_{n=kB}^{(k+1)B-1} \mathcal{E}[n]$ is strongly connected, for all $k \geq 0$;

(B2) Every weight matrix $\boldsymbol{W}[n]$ in (3) satisfies

$$\boldsymbol{W}[n]\,\mathbf{1} = \mathbf{1} \quad \text{and} \quad \mathbf{1}^T \boldsymbol{W}[n] = \mathbf{1}^T \quad \forall n. \quad (4)$$

Assumption B1 allows strong connectivity to occur over a long time period and in arbitrary order. Assumption B2 is instrumental to guarantee convergence of the algorithm we are going to introduce. Note also that $\boldsymbol{W}[n]$ can be time-varying and need not be symmetric.

Our goal is to develop an algorithm that converges to stationary solutions of Problem (1) while being implementable in the above distributed setting (Assumptions A and B).

3. INEXACT NEXT ALGORITHM

Devising distributed solution methods for Problem (1) faces two main challenges, namely: the nonconvexity of the objective function and the lack of global information at local side. Following the approach proposed in [12], we combine SCA techniques (Step 1 below) with consensus mechanisms (Step 2), as described next.

`Step 1 (local SCA optimization)`**:** Each agent i maintains a local estimate $\mathbf{x}_i$ of the optimization variable $\mathbf{x}$ that is iteratively updated. Solving directly Problem (1) may be too costly (due to the nonconvexity of f_i's) and is not even doable in a distributed setting (because of the lack of global knowledge). One may then prefer to approximate Problem (1), in some suitable sense, in order to permit each agent to compute *locally* and *efficiently* the new iteration. Since node i has knowledge only of f_i, writing $F(\mathbf{x}_i) = f_i(\mathbf{x}_i) + \sum_{j\neq i} f_j(\mathbf{x}_i)$, leads naturally to a convexification of F having the following form: i) at every iteration n, the (possibly) nonconvex $f_i(\mathbf{x}_i)$ is replaced by a strongly convex surrogate, say $\widetilde{f}_i(\bullet; \mathbf{x}_i[n]) : \mathcal{K} \to \mathbb{R}$, which may depend on the current iterate $\mathbf{x}_i[n]$; ii) and $\sum_{j\neq i} f_j(\mathbf{x}_i)$ is linearized around $\mathbf{x}_i[n]$ (because it is not available at node i). More formally, the proposed updating scheme reads: at every n, given the local estimate $\mathbf{x}_i[n]$, each agent i solves the following *strongly convex* optimization problem [12]:

$$\widehat{\mathbf{x}}_i(\mathbf{x}_i[n]) \quad (5)$$
$$\triangleq \operatorname*{argmin}_{\mathbf{x}_i \in \mathcal{K}} \widetilde{f}_i(\mathbf{x}_i; \mathbf{x}_i[n]) + \boldsymbol{\pi}_i(\mathbf{x}_i[n])^T(\mathbf{x}_i - \mathbf{x}_i[n]) + G(\mathbf{x}_i),$$

where $\boldsymbol{\pi}_i(\mathbf{x}_i[n])$ is the linearization of $\sum_{j\neq i} f_j(\mathbf{x}_i)$ at $\mathbf{x}_i[n]$, given by

$$\boldsymbol{\pi}_i(\mathbf{x}_i[n]) \triangleq \sum_{j\neq i} \nabla_{\mathbf{x}} f_j(\mathbf{x}_i[n]). \quad (6)$$

Note that $\widehat{\mathbf{x}}_i(\mathbf{x}_i[n])$ is well-defined, because (5) has a unique solution. The idea behind the iterate (5) is to compute stationary solutions of Problem (1) as fixed-points of the mappings $\widehat{\mathbf{x}}_i(\bullet)$. Postponing the convergence analysis, a first natural question is how to choose the surrogate function $\widetilde{f}_i(\bullet; \mathbf{x}_i[n])$. The next proposition addresses this issue and establishes the connection between the fixed-points of $\widehat{\mathbf{x}}_i(\bullet)$ and the stationary solutions of Problem (1) [14].

Proposition 1 *Given Problem (1) under A1-A6, suppose that each $\widetilde{f}_i$ satisfies the following conditions:*

(F1) *$\widetilde{f}_i(\bullet; \mathbf{y})$ is uniformly strongly convex on $\mathcal{K}$;*

(F2) *$\nabla \widetilde{f}_i(\mathbf{x}; \mathbf{x}) = \nabla f_i(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{K}$;*

(F3) *$\nabla \widetilde{f}_i(\mathbf{x}; \bullet)$ is uniformly Lipschitz continuous on $\mathcal{K}$.*

Then, the set of fixed-point of $\widehat{\mathbf{x}}_i(\bullet)$ coincides with that of the stationary solutions of (1). Therefore, $\widehat{\mathbf{x}}_i(\bullet)$ has a fixed-point.

Conditions F1-F3 are quite natural: $\widetilde{f}_i$ should be regarded as a (simple) convex, local, approximation of f_i at the point $\mathbf{x}$ that preserves the first order properties of f_i. Several choices are possible for a given f_i, the appropriate one depends on the problem at hand and computational requirements; some examples are discussed at the end of this section. `Step 2 (consensus update)`**:** To force the asymptotic agreement among the $\mathbf{x}_i$'s, a consensus-based step is employed on $\widehat{\mathbf{x}}_i(\mathbf{x}_i[n])$'s. Each agent i updates its $\mathbf{x}_i$ as

$$\mathbf{x}_i[n+1] = \sum_{j \in \mathcal{N}_i^{\text{in}}[n]} w_{ij}[n]\, \widehat{\mathbf{x}}_j(\mathbf{x}_j[n]), \quad (7)$$

where $(w_{ij}[n])_{ij}$ is any set of weights satisfying Assumption B2. Since the weights are constrained by the network topology, (7) can be implemented via local message exchanges.

`Toward a fully distributed implementation`**:** The computation of $\widehat{\mathbf{x}}_i(\mathbf{x}_i[n])$ is still not fully distributed, because the evaluation of $\boldsymbol{\pi}_i(\mathbf{x}_i[n])$ in (5) would require the knowledge of all $\nabla f_j(\mathbf{x}_i[n])$, which is not available locally at node i. To cope with this issue, the proposed approach consists in replacing $\boldsymbol{\pi}_i(\mathbf{x}_i[n])$ in (5) with a local estimate $\widetilde{\boldsymbol{\pi}}_i[n]$ that asymptotically converges to $\boldsymbol{\pi}_i(\mathbf{x}_i[n])$, and solve instead [12]:

$$\widetilde{\mathbf{x}}_i(\mathbf{x}_i[n], \widetilde{\boldsymbol{\pi}}_i[n]) \quad (8)$$
$$\triangleq \operatorname*{argmin}_{\mathbf{x}_i \in \mathcal{K}} \underbrace{\widetilde{f}_i(\mathbf{x}_i; \mathbf{x}_i[n]) + \widetilde{\boldsymbol{\pi}}_i[n]^T(\mathbf{x}_i - \mathbf{x}_i[n]) + G(\mathbf{x}_i)}_{\triangleq \widetilde{U}_i(\mathbf{x}_i; \mathbf{x}_i[n], \widetilde{\boldsymbol{\pi}}_i[n])}.$$

Rewriting $\boldsymbol{\pi}_i(\mathbf{x}_i[n])$ as

$$\boldsymbol{\pi}_i(\mathbf{x}_i[n]) = I \cdot \underbrace{\left(\frac{1}{I}\sum_{j=1}^{I} \nabla f_j(\mathbf{x}_i[n])\right)}_{\triangleq \overline{\nabla f}(\mathbf{x}_i[n])} - \nabla f_i(\mathbf{x}_i[n]), \quad (9)$$

we propose to update $\widetilde{\boldsymbol{\pi}}_i[n]$ mimicking (9):

$$\widetilde{\boldsymbol{\pi}}_i[n] \triangleq I \cdot \boldsymbol{y}_i[n] - \nabla f_i(\mathbf{x}_i[n]), \quad (10)$$

where $\boldsymbol{y}_i[n]$ aims to asymptotically track $\overline{\nabla f}(\mathbf{x}_i[n])$. Leveraging *dynamic* average consensus methods [15], this can be done updating $\boldsymbol{y}_i[n]$ according to the following recursion:

$$\boldsymbol{y}_i[n+1] \triangleq \sum_{j \in \mathcal{N}_i^{\text{in}}[n]} w_{ij}[n]\boldsymbol{y}_j[n] + \Delta r_i(\mathbf{x}_i[n+1], \mathbf{x}_i[n]) \quad (11)$$

with $\boldsymbol{y}_i[0] \triangleq \nabla f_i(\mathbf{x}_i[0])$, and

$$\Delta r_i(\mathbf{x}_i[n+1], \mathbf{x}_i[n]) = \nabla f_i(\mathbf{x}_i[n+1]) - \nabla f_i(\mathbf{x}_i[n]).$$

In fact, if the sequences $\{\mathbf{x}_i[n]\}_n$ are convergent and consensual, it holds $\left\|\boldsymbol{y}_i[n] - \overline{\nabla f}(\mathbf{x}_i[n])\right\| \underset{n\to\infty}{\longrightarrow} 0$ [16] and thus

$$\|\widetilde{\boldsymbol{\pi}}_i[n] - \boldsymbol{\pi}_i(\mathbf{x}_i[n])\| \underset{n\to\infty}{\longrightarrow} 0.$$

Note that the update of $\boldsymbol{y}_i[n]$, and thus $\widetilde{\boldsymbol{\pi}}_i[n]$, can be now performed locally with message exchanges in the neighborhood $\mathcal{N}_i[n]$.

Algorithm 1: Inexact NEXT

Data : $\mathbf{x}_i[0] \in \mathcal{K}$, $\boldsymbol{y}_i[0] = \nabla f_i[0]$, and $\widetilde{\boldsymbol{\pi}}_i[0] = I\boldsymbol{y}_i[0] - \nabla f_i[0]$, $\forall i = 1, \ldots, I$. Set $n = 0$.

(S.1) If $\mathbf{x}[n]$ satisfies a termination criterion: STOP;

(S.2) Local inexact SCA: Each agent i

(a) solves (8) with accuracy $\varepsilon_i[n]$: Find a $\mathbf{x}_i^{\text{inx}}[n] \in \mathcal{K}$

s.t. $\quad \|\widetilde{\mathbf{x}}_i[n] - \mathbf{x}_i^{\text{inx}}[n]\| \leq \varepsilon_i[n]$;

(b) updates its local variable $\boldsymbol{z}_i[n]$:

$$\boldsymbol{z}_i[n] = \mathbf{x}_i[n] + \alpha[n]\left(\mathbf{x}_i^{\text{inx}}[n] - \mathbf{x}_i[n]\right)$$

(S.3) Consensus update: Each agent i collects data from its neighbors and updates $\mathbf{x}_i[n]$, $\boldsymbol{y}_i[n]$, and $\widetilde{\boldsymbol{\pi}}_i[n]$ as:

(a) $\mathbf{x}_i[n+1] = \sum_{j \in \mathcal{N}_i^{\text{in}}[n]} w_{ij}[n]\, \boldsymbol{z}_j[n]$

(b) $\boldsymbol{y}_i[n+1] = \sum_{j \in \mathcal{N}_i^{\text{in}}[n]} w_{ij}[n]\, \boldsymbol{y}_j[n] + \Delta r_i(\mathbf{x}_i[n+1], \mathbf{x}_i[n])$

(c) $\widetilde{\boldsymbol{\pi}}_i[n+1] = I \cdot \boldsymbol{y}_i[n+1] - \nabla f_i[n+1]$

(S.4) $n \leftarrow n+1$, and go to (S.1).

Inexact updates: In many situations (e.g., in the case of large-scale problems), it can be useful to further reduce the computational effort to solve the subproblems in (8) by allowing inexact computations $\mathbf{x}_i^{\text{inx}}[n]$ of $\widetilde{\mathbf{x}}_i[n]$ in (8), i.e.,

$$\|\mathbf{x}_i^{\text{inx}}[n] - \widetilde{\mathbf{x}}_i[n]\| \leq \varepsilon_i[n], \tag{12}$$

where $\varepsilon_i[n]$ measures the accuracy in computing the solution. This is a noticeable feature of the proposed algorithm that allows to control the cost per iteration without affecting too much, experience shows, the empirical convergence speed.

We are now in the position to formally introduce the Inexact NEXT algorithm, as given in Algorithm 1. The algorithm builds on the iterates (8), (7) (wherein each $\widehat{\mathbf{x}}_j$ is replaced by $\widetilde{\mathbf{x}}_j$), the inexact computation (12), and (10)-(11), introduced in the previous section. Note that in S.2, in addition to solving (inexactly) the strongly convex optimization problem (8), we also introduced a step-size in the iterate. The convergence properties of Algorithm 1 are stated next.

Theorem 1 ([16]) *Let $\{\mathbf{x}[n]\}_n \triangleq \{(\mathbf{x}_i[n])_{i=1}^I\}_n$ be the sequence generated by Algorithm 1, and let $\{\overline{\mathbf{x}}[n]\}_n \triangleq \{1/I \sum_{i=1}^I \mathbf{x}_i[n]\}_n$ be its average. Suppose that i) Assumptions A and B hold; and ii) the step-size sequence $\{\alpha[n]\}_n$ is such that $\alpha[n] \in (0,1]$, for all n,*

$$\textstyle\sum_{n=0}^{\infty} \alpha[n] = \infty \quad \textit{and} \quad \sum_{n=0}^{\infty} \alpha[n]^2 < \infty, \tag{13}$$

and the following holds

$$\textstyle\sum_{n=0}^{\infty} \alpha[n]\, \varepsilon_i[n] < \infty, \quad \forall i = 1, \ldots, I. \tag{14}$$

Then, (a) [convergence]: the sequence $\{\overline{\mathbf{x}}[n]\}_n$ is bounded and all its limit points are stationary solutions of Problem (1);
(b) [consensus]: all the sequences $\{\mathbf{x}_i[n]\}_n$ asymptotically agree, i.e., $\|\mathbf{x}_i[n] - \overline{\mathbf{x}}[n]\| \underset{n\to\infty}{\longrightarrow} 0$, for all $i = 1, \ldots, I$.

As expected, in the presence of errors, convergence of Algorithm 1 is guaranteed if the sequence of approximated problems in S.2(a) is solved with increasing accuracy. Note that, in addition to require $\varepsilon_i[n] \to 0$, condition (14) imposes also a constraint on the rate by which $\varepsilon_i[n]$ goes to zero, which depends on the rate of decrease of $\alpha[n]$. An example of error sequence satisfying the above condition is $\varepsilon_i[n] \leq c_i \alpha[n]$, where c_i is any finite positive constant [14]. Interesting, such a condition can be forced in Algorithm 1 in a distributed way, using classical error bound results in convex analysis; see, e.g., [17, Ch.6, Prop. 6.3.7].

On the choice of the surrogates $\widetilde{f}_i$: Adapting to our setting the approximation functions introduced in [13, 14], the following examples are instances of $\widetilde{f}_i$ satisfying F1-F3.

• When f_i has no special structure to exploit, the most obvious choice for $\widetilde{f}_i$ is the linearization of f_i at $\mathbf{x}_i[n]$:

$$\begin{aligned}\widetilde{f}_i(\mathbf{x}_i; \mathbf{x}_i[n]) = \; & f_i(\mathbf{x}_i[n]) + \nabla f_i(\mathbf{x}_i[n])^T(\mathbf{x}_i - \mathbf{x}_i[n]) \\ & + \frac{\tau_i}{2}\|\mathbf{x}_i - \mathbf{x}_i[n]\|^2,\end{aligned} \tag{15}$$

where τ_i is any positive constant. The proximal regularization guarantees that $\widetilde{f}_i$ is strongly convex. The above surrogate is essentially a reminiscence of the approximation of the objective function used in proximal-gradient algorithms. Note however that standard proximal-gradient algorithms are not directly applicable to Problem (1), as they are not distributed.

• At another extreme, if f_i is convex, one could just take

$$\widetilde{f}_i(\mathbf{x}_i; \mathbf{x}_i[n]) = f_i(\mathbf{x}_i) + \frac{\tau_i}{2}\|\mathbf{x}_i - \mathbf{x}_i[n]\|^2, \tag{16}$$

with any $\tau_i \geq 0$ (τ_i can be set to zero if f_i is strongly convex). This choice gives rise to the *first distributed* nonlinear Jacobi type method for the constrained minimization of U.

• Between the two "extreme" solutions proposed above, one can consider "intermediate" choices. For example, if f_i is convex, mimicking Newton schemes, one can take $\widetilde{f}_i$ as a second order approximation of f_i, i.e.,

$$\begin{aligned}\widetilde{f}_i(\mathbf{x}_i; \mathbf{x}_i[n]) = \; & f_i(\mathbf{x}_i[n]) + \nabla f_i(\mathbf{x}_i[n])^T(\mathbf{x}_i - \mathbf{x}_i[n]) \\ & + \frac{1}{2}(\mathbf{x}_i - \mathbf{x}_i[n])^T \nabla^2 f_i(\mathbf{x}_i[n])(\mathbf{x}_i - \mathbf{x}_i[n]).\end{aligned} \tag{17}$$

• Another "intermediate" choice, relying on a specific structure of each f_i that has important applications is the following. Suppose that f_i is convex only in some components of $\mathbf{x}_i$; let us split $\mathbf{x}_i \triangleq (\mathbf{x}_{i,1}, \mathbf{x}_{i,2})$ so that $f_i(\mathbf{x}_{i,1}, \mathbf{x}_{i,2})$ is convex in $\mathbf{x}_{i,1}$ for every $\mathbf{x}_{i,2}$ such that $(\mathbf{x}_{i,1}, \mathbf{x}_{i,2}) \in \mathcal{K}$, but not in $\mathbf{x}_{i,2}$. A natural choice for $\widetilde{f}_i$ is then: given $\mathbf{x}[n] \triangleq (\mathbf{x}_{i,1}[n], \mathbf{x}_{i,2}[n])$,

$$\begin{aligned}\widetilde{f}_i(\mathbf{x}_i; \mathbf{x}_i[n]) = \; & \widetilde{f}_i^{(1)}(\mathbf{x}_{i,1}; \mathbf{x}_{i,2}[n]) + \frac{\tau_i}{2}\|\mathbf{x}_{i,2} - \mathbf{x}_{i,2}[n]\|^2 \\ & + \nabla_{x_{i,2}} f_i(\mathbf{x}_i[n])^T(\mathbf{x}_{i,2} - \mathbf{x}_{i,2}[n])\end{aligned} \tag{18}$$

where $\widetilde{f}_i^{(1)}(\bullet; \mathbf{x}_{i,2}[n])$ is any function still satisfying F1-F3 (written now in terms of $\mathbf{x}_{i,1}$ for given $\mathbf{x}_{i,2}$). Any of the choices in (15)-(17) are valid for $\widetilde{f}_i^{(1)}(\bullet; \mathbf{x}_{i,2}[n])$. The rationale behind (18) is to preserve the favorable convex part of f_i with respect to $\mathbf{x}_{i,1}$ while linearizing the nonconvex part.

• Consider the case in which $\widetilde{f}_i$ is block-wise convex but not convex on $\mathbf{x}_i$. Without loss of generality, let us assume that $\widetilde{f}_i$ is convex in the two block-variables $\mathbf{x}_{i,1}$ and $\mathbf{x}_{i,2}$ partitioning $\mathbf{x}_i = (\mathbf{x}_{i,1}, \mathbf{x}_{i,2})$, but not jointly (the case of more than two blocks can be similarly considered). Then a natural choice for $\widetilde{f}_i$ is

$$\begin{aligned}\widetilde{f}_i(\mathbf{x}_i; \mathbf{x}_i[n]) = \; & f_i(\mathbf{x}_{i,1}, \mathbf{x}_{i,2}[n]) + f_i(\mathbf{x}_{i,1}[n], \mathbf{x}_{i,2}) \\ & + \frac{\tau_i}{2}\|\mathbf{x}_i - \mathbf{x}_i[n]\|^2.\end{aligned} \tag{19}$$

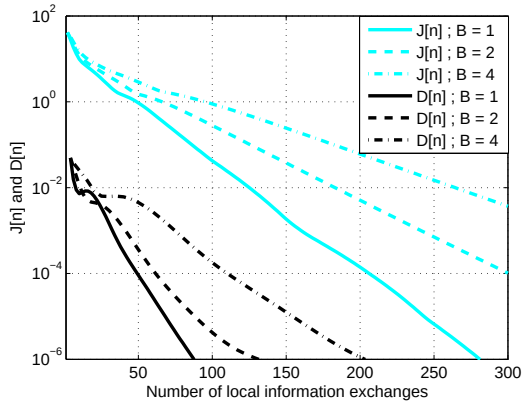

Fig. 1: Distance from stationarity ($J[n]$) and disagreement ($D[n]$) versus number of local information exchanges.

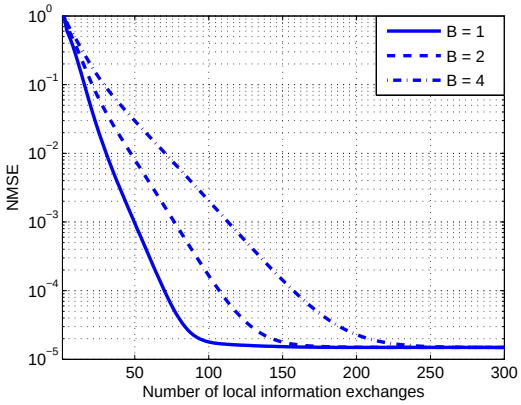

Fig. 2: NMSE versus number of local information exchanges.

Note that, in the same spirit of the previous example, instead of $f_i(\bullet, \mathbf{x}_{i,2}[n])$ and $f_i(\mathbf{x}_{i,1}[n], \bullet)$ one can use any surrogate satisfying F1-F3 in the intended variables.

• As last example, consider the case in which $f(\mathbf{x}_i)$ is the composition of two functions, i.e., $f(\mathbf{x}_i) = q(h(\mathbf{x}_i))$, where q is convex. In this case, a possible choice for $\tilde{f}_i$ is to preserve the convexity of q, while linearizing h. It is easy to see that the resulting surrogate, i.e.

$$\widetilde{f}_i(\mathbf{x}_i; \mathbf{x}_i[n]) = q\big(h(\mathbf{x}_i[n]) + \nabla_{\mathbf{x}_i} h(\mathbf{x}_i[n])^T(\mathbf{x}_i - \mathbf{x}_i[n])\big) + \frac{\tau_i}{2}\|\mathbf{x}_i - \mathbf{x}_i[n]\|^2, \quad (20)$$

satisfies properties F1-F3 in Proposition 1.

4. DISTRIBUTED CONSTRAINED NONLINEAR REGRESSION OVER NETWORKS

We now customize the proposed framework to solve a nonlinear regression problem. Each agent i collects a set of M_i measurements $y_{i,m}$, which are related to some parameter $\mathbf{x} \in \mathbb{R}^L$ of interest through the following nonlinear observation model:

$$y_{i,m} = h_i(\mathbf{x}; \boldsymbol{z}_{i,m}) + v_{i,m}, \quad (21)$$

for all $i = 1, \ldots, I$, $m = 1, \ldots, M_i$, where $h_i : \mathbb{R}^L \to \mathbb{R}$, $\boldsymbol{z}_{i,m} \in \mathbb{R}^P$ are known regression vectors, and $v_{i,m}$ represents additive observation nose. The goal of the network is to solve the following nonlinear least squares problem:

$$\underset{\mathbf{x}\in\mathcal{K}}{\text{minimize}} \quad \sum_{i=1}^{I}\sum_{m=1}^{M_i}\big(y_{i,m} - h_i(\mathbf{x}; \boldsymbol{z}_{i,m})\big)^2 + \frac{\lambda}{2}\|\mathbf{x}\|_2^2 \quad (22)$$

where $\lambda > 0$ and $\mathcal{K}$ is a bounded set, which the parameter $\mathbf{x}$ belongs to. Problem (22) is clearly an instance of Problem (1), with

$$f_i(\mathbf{x}) = \sum_{m=1}^{M_i}\big(y_{i,m} - h_i(\mathbf{x}; \boldsymbol{z}_{i,m})\big)^2 \text{ and } G(\mathbf{x}) = \frac{\lambda}{2}\|\mathbf{x}\|_2^2. \quad (23)$$

To apply Algorithm 1, we need to specify a valid surrogate $\widetilde{f}_i(\mathbf{x}; \mathbf{x}[n])$, to be used in (8) (i.e., step S.2(a) of Algorithm 1). Since f_i in (23) is a composition of functions, we can exploit the approximation in (20). Thus, a possible choice for the surrogate function is:

$$\widetilde{f}_i(\mathbf{x}, \mathbf{x}[n]) = \sum_{m=1}^{M_i}\big(y_{i,m} - \widetilde{h}_i(\mathbf{x}; \mathbf{x}[n], \boldsymbol{z}_{i,m})\big)^2, \quad (24)$$

where $\widetilde{h}_i(\mathbf{x}; \mathbf{x}[n], \boldsymbol{z}_{i,m}) = \nabla_{\mathbf{x}} h_i(\mathbf{x}[n]; \boldsymbol{z}_{i,m})^T(\mathbf{x} - \mathbf{x}[n]) + h_i(\mathbf{x}[n]; \boldsymbol{z}_{i,m})$ is the linearization of $h_i(\mathbf{x}; \boldsymbol{z}_{i,m})$ around $\mathbf{x}[n]$.

Numerical example: We consider a directed network composed of $I = 30$ nodes with B-strongly connected graph topology. The nonlinear functions $h_i(\mathbf{x}; \boldsymbol{z}_{i,m})$ in (21) have a sigmoidal shape, i.e. $h_i(\mathbf{x}; \boldsymbol{z}_{i,m}) = 1/(1 + \exp(-\boldsymbol{z}_{i,m}^T\mathbf{x}))$. The observations at each node are generated from (21), using a random vector $\mathbf{x}^0 \in \mathcal{K} = [0, 1]^{10}$, Gaussian random input vectors $\boldsymbol{z}_{i,m}$ with identity covariance matrix, and Gaussian observation noise with variance $\sigma_v^2 = 10^{-6}$. We consider $M_i = 10$, for all i, and $\lambda = 0.05$. We measure the progress of Algorithm 1 using the following two merit functions: $J[n] \triangleq \|\bar{\mathbf{x}}[n] - \Pi_{\mathcal{K}}(\bar{\mathbf{x}}[n] - \nabla U(\bar{\mathbf{x}}[n]))\|_\infty$ and $D[n] \triangleq \frac{1}{I}\sum_i \|\mathbf{x}_i[n] - \bar{\mathbf{x}}[n]\|^2$, where $\Pi_{\mathcal{K}}(\boldsymbol{y})$ denotes the orthogonal projection onto the convex and closed set $\mathcal{K}$. Note that $J[n] = 0$ if and only if $\bar{\mathbf{x}}[n]$ is a stationary solution of Problem (22); therefore $J[n]$ measures the progress of the algorithm toward stationarity. The sequence $D[n]$ measures the disagreement among the agents' variables. We also report the normalized mean squared error (NMSE), evaluated at $\bar{\mathbf{x}}[n]$, i.e. $\text{NMSE}[n] = \frac{\|\bar{\mathbf{x}}[n] - \mathbf{x}_0\|^2}{\|\mathbf{x}_0\|^2}$. In Fig. 1 we plot $J[n]$ and $D[n]$ versus the number of communication exchanges per node. Fig. 2 shows $\text{NMSE}[n]$ versus the number of exchanged messages. The figures are reported for different values of the uniform graph connectivity coefficient B (the case $B = 1$ corresponds to a fixed graph, and is reported as a benchmark), and are averaged over 100 independent noise realizations. The algorithm is initialized at random, and uses the diminishing step-size rule $\alpha[n] = \alpha[n-1](1 - \mu\alpha[n-1])$, $n \geq 1$, with $\alpha[0] = 0.01$ and $\mu = 0.01$. The optimization in (8) (Step S2(a) of Algorithm 1) leads to a box-constrained least-square problem that is solved using the quadprog Matlab function, reducing the error on the solution obtained at each iteration with a rate proportional to $\alpha[n]$. Both figures illustrate the convergence of the algorithm to a stationary solution of (22), thus confirming the theoretical results of Theorem 1. Furthermore, as expected, larger values of B lead to a lower practical convergence speed, due to the slower diffusion of information over the network.

5. CONCLUSIONS

We introduced a novel algorithmic framework for nonconvex distributed optimization in multi-agent networks with time-varying (nonsymmetric) topology. The proposed method exploits SCA techniques while leveraging dynamic consensus as a mechanism to diffuse the needed information over the network. Asymptotic convergence of the proposed method was established, in the presence of time-varying directed topologies and inexact updates. Finally, the proposed framework was applied to solve a distributed constrained nonlinear least squares problem over networks.

6. REFERENCES

[1] A. Nedić and A. Ozdaglar, "Distributed subgradient methods for multiagent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, Jan. 2009.

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

[3] A. Chen and A. Ozdaglar, "A fast distributed proximal gradient method," in *50th Allerton Conference on Communication, Control and Computing*, Monticello, IL, USA, Oct. 1-5 2012.

[4] J. Chen and A. H. Sayed, "Diffusion adaptation strategies for distributed optimization and learning over networks," *IEEE Transactions on Signal Processing*, vol. 60, no. 8, pp. 4289–4305, August 2012.

[5] P. Di Lorenzo and A. H. Sayed, "Sparse distributed learning based on diffusion adaptation," *IEEE Transactions on Signal Processing*, vol. 61, no. 6, pp. 1419–1433, March 2013.

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

[7] K. I. Tsianos, S. Lawlor, and M. G. Rabbat, "Push-sum distributed dual averaging for convex optimization," in *IEEE Conference on Decision and Control*, Maui, Hawaii, USA, Dec. 10–13 2012.

[8] 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, March 2012 2012.

[9] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, *Distributed optimization and statistical learning via the alternating direction method of multipliers*, ser. Foundations and Trends in Machine Learning. Boston-Delft: NOW Publishers, 2011, vol. 3, no. 1.

[10] M. Zhu and S. Martínez, "An approximate dual subgradient algorithm for distributed non-convex constrained optimization," *IEEE Transactions on Automatic Control*, vol. 58, no. 6, pp. 1534–1539, June 2013.

[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] P. Di Lorenzo and G. Scutari, "Distributed nonconvex optimization over networks," in *IEEE Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, Dec. 13–16, 2015.

[13] G. Scutari, F. Facchinei, P. Song, D. P. Palomar, and J.-S. Pang, "Decomposition by partial linearization: Parallel optimization of multiuser systems," *IEEE Transactions on Signal Processing*, vol. 63, no. 3, pp. 641–656, Feb. 2014.

[14] 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, April 2015.

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

[16] P. Di Lorenzo and G. Scutari, "Next: In-network nonconvex optimization," *to appear on IEEE Transactions on Signal and Information Processing over Networks*, 2016.

[17] F. Facchinei and J.-S. Pang, *Finite-Dimensional Variational Inequalities and Complementarity Problem*. New York: Springer-Verlag, 2003.