Abstract—The paper studies distributed Dictionary Learning (DL) problems where the learning task is distributed over a multi-agent network with time-varying (nonsymmetric) connectivity. This formulation is relevant, for instance, in Big Data scenarios where massive amounts of data are collected/stored in different spatial locations and it is unfeasible to aggregate and/or process all data in a fusion center, due to resource limitations, communication overhead or privacy considerations. We develop a general distributed algorithmic framework for the (nonconvex) DL problem and establish its asymptotic convergence. The new method hinges on Successive Convex Approximation (SCA) techniques coupled with i) a gradient tracking mechanism instrumental to locally estimate the missing global information; and ii) a consensus step, as a mechanism to distribute the computations among the agents. To the best of our knowledge, this is the first distributed algorithm with provable convergence for the DL problem and, in more general, bi-convex optimization problems over (time-varying) directed graphs.

I. INTRODUCTION

The dictionary learning problem \[1\] consists in finding an overcomplete basis (a.k.a the dictionary) by which a given set of data can be sparsely represented. This technique can be leveraged to solve many machine learning and inference tasks, including image denoising/deblurring/inpainting, super-resolution \[2\], \[3\], dimensionality reduction \[4\], bi-clustering \[5\], feature-extraction and classification \[6\], and prediction \[7\].

Denoting by \(S \in \mathbb{R}^{M \times N}\) the data (observation) matrix, by \(D \in \mathbb{R}^{M \times K}\) the overcomplete dictionary matrix, and by \(X \in \mathbb{R}^{K \times N}\) the sparse representation matrix of the signal, the DL problem for sparse representation reads (see, e.g., \[8\])

\[
\min_{D,X} \frac{1}{2} \|S - DX\|_F^2 + \lambda \|X\|_{1,1} + \mu \|X\|_2^2
\]

s.t. \(D \in \mathbb{R}^{M \times K} \ni \{D : \|De_k\|_2 \leq \alpha, k = 1, 2, \ldots, K\}\),

where \(\|X\|_{1,1} \triangleq \sum_{i,j} |X_{i,j}|\), and \(D\) is constrained to belong to the set \(\mathcal{D}\) to avoid unbound solutions, with \(\alpha > 0\) and \(e_k\) denoting the \(k\)-th canonical vector. In \((P1)\), sparsity is imposed on \(X\) using elastic net regularization \[8\] with parameters \(\mu, \lambda > 0\); the elastic net regularization tends to be preferred to the plain \(\ell_1\) regularization (a.k.a. LASSO), since it better preserves group patterns in the variables, especially when there are highly correlated variables. Note that Problem \((P1)\) is a nonconvex optimization problem due to the bi-convex structure of the objective functions. The lack of convexity has motivated a lot of interest in pursuing approximate solutions that approach optimal performance at moderate complexity; some recent efforts are documented in \[8\], \[9\]–\[16\]. All the algorithms therein are centralized, i.e., they require the data matrix \(S\) to be centrally available.

In many large-scale signal processing and machine learning problems, data are not necessarily centrally available, but collected/stored in multiple locations; for example, consider sensor, cloud, or cluster-computer networks. In these scenarios, sharing local information with a central processor might be either unfeasible or not economical/efficient, owing to the large size of the network/data, dynamicity of network topology, energy constraints, and/or privacy issues. Common to all the aforementioned problems is the necessity of performing a completely decentralized computation/optimization.

Motivated by these observations, we aim at developing a solution method for \((P1)\) in a distributed setting wherein the data matrix \(S\) is spread over a network with (possibly) time-varying topology, and each node has access only to some portions of \(S\). We are not aware of any distributed, convergent scheme for such class of problems; some attempts have been documented recently in \[17\]–\[21\]. Although there are substantial differences between these methods, they can be generically abstracted as combinations of local (primal or dual ascent) descent steps followed by variable exchanges and averaging of information among neighboring nodes. However, theoretical convergence of these methods remains an open question. This is mainly due to the fact that these schemes exploit decomposition techniques suitable for (strongly) convex problems, whereas Problem \((P1)\) is nonconvex (and may lack zero-duality gap). Furthermore, numerical results therein are contradictory; for instance, some of the aforementioned schemes are observed not to converge while some others fail to reach asymptotic agreement among the local copies of the dictionary variables. Other relevant papers are \[22\]–\[24\] wherein multiaagent nonconvex optimization over networks is studied. While interesting, the approaches in \[22\]–\[24\] are not able to handle nonconvex problems in the form of \((P1)\) (see Sec. III-B for more details).

The major contribution of this work is to propose the first provably convergent algorithmic framework for the distributed DL problem over (possibly) time-varying network topologies. The crux of the framework is a novel convexification-decomposition technique that hinges on our recent SCA methods \[15\], \[16\] while leveraging i) a gradient tracking mechanism to locally estimate the missing global information; and ii) a consensus step to distribute the computation as well as propagate the needed information over the network. Asymptotic convergence of the proposed algorithm to stationary solutions of \((P1)\) is proved. Preliminary numerical results show that the proposed scheme compare favorably with other decentralized state-of-the-art algorithms.

The rest of the paper is organized as follows. Sec. \[II\]
describes the distributed DL problem along with the network setting. The design of the new algorithm is addressed in Sec. III. Numerical results are presented in Sec. IV and some conclusions are drawn in Sec. V.

II. PROBLEM DESCRIPTION

Consider Problem (P1) defined over a network composed of \( I \) autonomous agents (nodes). Each agent \( i \) owns a subset of columns of the data matrix \( S \triangleq [S_1, \ldots, S_I] \), say \( S_i \in \mathbb{R}^{M \times n_i} \), and controls the corresponding part of the sparse representation matrix \( X \triangleq [X_1, \ldots, X_I] \), where \( X_i \in \mathbb{R}^{n_i \times n} \), and \( \sum_i n_i = N \). Then, Problem (P1) can be rewritten as

\[
\min_{D \{X_i\}_{i=1}^I} \sum_{i=1}^I \left[ \frac{1}{2} \|S_i - DX_i\|_F^2 + \lambda \|X_i\|_{1,1} + \mu \|X_i\|_F^2 \right] \tag{P2}
\]

s.t. \( D \in \mathcal{D} \).

Note that each agent \( i \) knows only its “local” function \( f_i \) (along with \( \lambda, \mu \), and the set \( \mathcal{D} \)). We remark that, even though in (P2) we assumed that the data matrix \( S \) is partitioned by columns, the proposed algorithm can be readily applied also to scenarios wherein \( S \) is partitioned by rows (and thus the dictionary matrix is partitioned accordingly); see the journal version of this work.

Network Topology. Time is slotted and, at each time-slot \( \nu \), the network of the \( I \) agents is modeled as a digraph \( \mathcal{G}^\nu = (\mathcal{V}, \mathcal{E}^\nu) \), where \( \mathcal{V} = \{1, \ldots, I\} \) is the vertex set (i.e., the set of agents) and \( \mathcal{E}^\nu \) is the set of (possibly) time-varying directed edges: \( (i, j) \in \mathcal{E}^\nu \) if there is a link from \( j \) to \( i \) (i.e., agent \( j \) can send information to agent \( i \) at time \( \nu \)). The set of in-neighborhood of agent \( i \) at time \( \nu \) is defined as \( \mathcal{N}^\nu_i = \{j : (i, j) \in \mathcal{E}^\nu \cup \{i\}\} \); it is the set of agents that can communicate with node \( i \) at time \( \nu \). To let information propagate over the network, we make the following standard assumption on the network connectivity.

Assumption A (on the network connectivity). The sequence of graphs \( \mathcal{G}^\nu \) is \( B \)-strongly connected, i.e., there exists a finite integer \( B > 0 \) such that the graph \( \mathcal{G}^\nu = (\mathcal{V}, \mathcal{E}^\nu_B) \), with \( \mathcal{E}^\nu_B = \bigcup_{\nu=kB}^{(k+1)B-1} \mathcal{E}^\nu \), is strongly connected for all \( k \geq 0 \).

To the best of our knowledge, this is the weakest condition under which one can prove convergence of distributed algorithms on time-varying directed networks.

III. ALGORITHMIC DESIGN OF D²L

The design of a distributed scheme for (P2) faces two main challenges, namely: the nonconvexity of each \( f_i \) and the lack of global information on all \( f_i \). To cope with these issues, we propose to combine alternating optimization (Step 1 below) with consensus mechanisms (Step 2), as described next.

Step 1: Local Optimization. Each agent \( i \) maintains a “local copy” \( D_i \) of the common dictionary \( D \) and controls its private variables \( X_i \). We denote by \( (D_i^\nu, X_i^\nu) \) the value of such a pair at iteration \( \nu \). The goal is that each agent updates \( (D_i^\nu, X_i^\nu) \) to \( (D_i^{\nu+1}, X_i^{\nu+1}) \) towards a solution of Problem (P2). However, (directly) solving (P2) is too costly, due to the nonconvexity of \( f_i(D_i, X_i) \), and it is not even feasible (because of the lack of global knowledge of problem, i.e., agent \( i \) does not have access to \( f_j, \forall j \neq i \)). Therefore, the idea is to somehow approximate (P2) so that each agent computes the new iteration locally and efficiently. Since each \( f_i \) is not jointly convex in \( (D_i, X_i) \) but bi-convex (i.e., convex in \( D_i \) and \( X_i \), separately), a natural approach is to update \( D_i \) and \( X_i \) in an alternating fashion. Specifically, fixing \( X_i = X_i^\nu \), \( D_i \) is updated solving the following strongly convex approximation of (P2):

\[
\tilde{D}_i^\nu \triangleq \arg\min_{D \in \mathcal{D}(i)} \tilde{f}_i(D_i; D_i^\nu, X_i^\nu) + \left( \Pi_i^\nu, D_i - D_i^\nu \right), \tag{1}
\]

where \( \langle A, B \rangle \triangleq \tr(A^TB) \) and \( \tilde{f}_i(D_i; D_i^\nu, X_i^\nu) \) is defined as

\[
\tilde{f}_i(D_i; D_i^\nu, X_i^\nu) \triangleq f_i(D_i, X_i^\nu) + \frac{\tau_{D,i}^\nu}{2} \|D_i - D_i^\nu\|_F^2, \tag{2}
\]

with \( \tau_{D,i}^\nu > 0 \) and \( \Pi_i^\nu \triangleq \sum_{j \neq i} \nabla_D f_j(D_i^\nu, X_j^\nu) \). The linearization of the unknown term \( \sum_{j \neq i} \nabla_D f_j(D_i^\nu, X_j^\nu) \) is the quadratic term \( \frac{\tau_{D,i}^\nu}{2} \|D_i - D_i^\nu\|_F^2 \) in (2) serves to make \( \tilde{f}_i \) strongly convex, so that \( \Pi_i^\nu \) has a unique solution, denoted by \( \tilde{D}^\nu_i \). The direct use of \( \tilde{D}^\nu_i \) as the new local estimate \( \hat{D}^\nu_i \) would not help to establish convergence for two reasons: (i) \( \hat{D}^\nu_i \) might be too “aggressive” update and (ii) we have not introduced any mechanism yet to ensure that the local copies \( \hat{D}^\nu_i \) eventually agree among all agents. To cope with these two issues, we introduce a step-size in the update of the dictionary:

\[
U_i^\nu \triangleq \hat{D}^\nu_i + \gamma^\nu (\hat{D}^\nu_i - D_i^\nu), \tag{3}
\]

where \( \gamma^\nu \) is a positive scalar to be properly chosen, see Assumption C in Sec. III-B.

We now consider the update of private variables \( X_i \). Fixing \( D_i = U_i^\nu \), agent \( i \) computes the new update \( X_i^{\nu+1} \) solving the following strongly convex optimization problem

\[
X_i^{\nu+1} \triangleq \arg\min_{X \in \mathcal{X}_i} f_i(X_i; U_i^\nu, X_i^\nu) + \lambda \|X_i\|_{1,1} + \mu \|X_i\|_F^2, \tag{4}
\]

where

\[
\tilde{h}_i(X_i; U_i^\nu, X_i^\nu) \triangleq f_i(U_i^\nu, X_i^\nu) + \frac{\tau_{X,i}^\nu}{2} \|X_i - X_i^\nu\|_F^2, \tag{5}
\]

and \( \tau_{X,i}^\nu \) is a positive scalar (to be properly chosen together with \( \tau_{D,i}^\nu \), see Assumption C in Sec. III-B).

Step 2: Consensus. To force the asymptotic agreement among the \( D^\nu_i \)’s, a consensus-based step is employed on \( U_i^\nu \)’s. Each agent \( i \) computes the new update \( D_i^{\nu+1} \) as

\[
D_i^{\nu+1} \triangleq \sum_{j \in \mathcal{N}^\nu_i} w_{ij}^\nu U_j^\nu \tag{6}
\]

where \( (w_{ij})_{i,j=1}^I \) is a set of weights matching the network topology \( \mathcal{G}^\nu \) at time slot \( \nu \), in the sense defined next.
Assumption B (on the weight matrix). The weights $W^\nu \triangleq (w^\nu_{ij})_{i,j=1}^I$ satisfy the following conditions: i) for every $\nu \geq 0$, 
\[ w^\nu_{ij} = \begin{cases} \theta \in [0, 1] & \text{if } j \in N^\nu_i , \\ 0 & \text{otherwise}, \end{cases} \tag{7} \]
for some $\theta \in (0, 1)$; ii) $W^\nu = 1$; and iii) $1^T W^\nu = 1^T$.

Toward a fully distributed implementation. The computation of $\tilde{D}^-_{i}^{\nu}$ and consequently, the update of $D^{{\nu}+1}_i$ in (8) have a severe drawback: the evaluation of $\Pi^\nu_i$ in (9) would require the knowledge of $\nabla D f_i(D^\nu_i, X^\nu_i)$ for all $j \neq i$, which is not available to agent $i$. To cope with this issue, we replace $\Pi^\nu_i$ in (9) with a “local estimate”, denoted by $\Pi^\nu_i$, and solve instead,
\[ \tilde{D}^-_{i}^{\nu} \triangleq \argmin_{D^-_{i}^{\nu}} f_i(D^\nu_i, D^-_{i}^{\nu}, X^\nu_i) + \langle \Pi^\nu_i, D^\nu_i - D^-_{i}^{\nu} \rangle. \tag{8} \]

The question now becomes how to update $\tilde{D}^-_{i}^{\nu}$ using only local information so that $\Pi^\nu_i$ will track the right $\Pi^\nu_i$. Rewriting $\Pi^\nu_i$ as
\[ \Pi^\nu_i = I \cdot \left( \frac{1}{I} \sum_{j=1}^I \nabla D f_j(D^\nu_i, X^\nu_i) \right) - \nabla D f_i(D^\nu_i, X^\nu_i), \tag{9} \]
we propose to compute $\tilde{\Pi}^\nu_i$ mimicking (9):
\[ \tilde{\Pi}^\nu_i = I \cdot \Theta^\nu_i - \nabla D f_i(D^\nu_i, X^\nu_i), \tag{10} \]
where $\Theta^\nu_i$ is a local auxiliary variable (controlled by user $i$) whose task is to asymptotically track $\Pi^\nu_i$. This can be done by leveraging the tracking mechanism first introduced in [24]:
\[ \Theta^\nu_{i+1} = \sum_{j \in N^\nu_i} w^\nu_{ij} \Theta^\nu_j + \nabla D f_i(D^\nu_{i+1}, X^\nu_i) - \nabla D f_i(D^\nu_i, X^\nu_i) \tag{11} \]
with $\Theta^0 \triangleq \nabla D f_i(D^0_i, X^0_i)$ for all $i = 1, 2, \ldots, I$.

Because of above modifications towards a distributed implementation, the update rule (3) is impacted and needs to be properly modified by replacing $D^-_{i}^{\nu}$ with $\tilde{D}^-_{i}^{\nu}$, which reads
\[ U_{i}^{\nu} + \nabla \nu(D^\nu_i - D^-_{i}^{\nu}), \tag{12} \]
We can now formally introduce the proposed Distributed Dictionary Learning (D$^2$L) algorithm, as given in Algorithm 1. Its convergence properties are stated in Theorem 1.

Algorithm 1: Distributed Dictionary Learning (D$^2$L)

Initialize: $X^0_i = 0$; $D^0_i \in D$; $\Theta^0_i = \nabla D f_i(D^0_i, X^0_i)$, 
\[ \Pi^\nu_i = I \cdot \Theta^0_i - \nabla D f_i(D^0_i, X^0_i), \forall i; \text{ set } \nu = 0; \]

S1. If $(D^\nu_i, X^\nu_i)$ satisfies stopping criterion for all $i$’s: STOP;

S2. Local Updates: Each agent $i$ computes:
(a) $D^\nu_i$ and $U_{i}^{\nu}$ according to (3) and (12);
(b) $X^\nu_{i+1}$ according to (4).

S3. Broadcasting: Each agent $i$ collects data from its current neighbors and updates:
(a) $D^\nu_{i+1}$ according to (6);
(b) $\Theta^\nu_{i+1}$ and $\Pi^\nu_{i+1}$ according to (11) and (10);

S4. Set $\nu + 1 \to \nu$, and go to S1.

(C2) The sequences $(\tau_{X,i}^\nu)$ and $(\tau_{D,i}^\nu)$ are bounded and uniformly positive, for all $i = 1, \ldots, I$. Additionally, $\tau_{X,i}^\nu = \max(\epsilon, \sigma_{\max}(U_{i}^{\nu})^2)$, where $\epsilon > 0$ is arbitrary, and $\sigma_{\max}(U_{i}^{\nu})$ is the maximum singular value of $U_{i}^{\nu}$.

We can now provide the main convergence results, as stated in Theorem 1 below. The proof of the theorem is quite involved and omitted here for lack of space; see the journal version of this work.

**Theorem 1:** Let $(\{D^\nu_i, X^\nu_i\}_{i=1}^I)^{\nu}$ be the sequence generated by Algorithm 1 and let $\bar{D}^\nu \triangleq \frac{1}{I} \sum_{i=1}^I D^\nu_i$. Suppose that Assumptions A-C are satisfied, then the following holds:
(i) $(\bar{D}^\nu, X^\nu)^{\nu}$ is bounded and any of its limit points is a stationary solution of Problem (P2) [and thus (P1)]; and
(ii) all $(D^\nu_i)^{\nu}$ asymptotically reach consensus, i.e., 
\[ \lim_{\nu \to \infty} ||D^\nu_i - \bar{D}^\nu|| = 0 \text{ for all } i = 1, 2, \ldots, I. \]

Roughly speaking, Theorem 1 states two main results: 1) (subsequence) convergence of $(\bar{D}^\nu, X^\nu)^{\nu}$ to a stationary solution of (P1), and 2) asymptotic agreement of all $D^\nu_i$ on the limit point of $(\bar{D}^\nu)^{\nu}$.

B. Discussion

Convergence: To the best of our knowledge, Algorithm 1 is the first distributed algorithm for the DL problem (P1), with convergence guarantees. Our results can be contrasted with [17]-[21] wherein gradient schemes tailored with consensus/diffusion updates are employed for some instance of (P1). The aforementioned schemes do not have any convergence guarantees: it is postulated that the sequence generated by the algorithms is convergent (see, e.g., [20], [21]), and then concluded that any limit point is a stationary solution of the problem. Furthermore, some of these schemes do not even achieve consensus among the local variables. Finally, we remark that our previous results [24] are not applicable to (P1), since: i) [24] cannot handle private variables, i.e., $X_i$; and ii) convergence of [24] (and of most of distributed gradient schemes in the literature [17]-[21]) require some technical properties that are not satisfied by the functions in (P2) [e.g., boundedness and Lipschitzianity of the gradient of $f_i$].
Solutions of the subproblems: The update of the variables $(D_{(i)}^{\nu}, X_{(i)}^{\nu})$ calls for the solutions of strongly convex problems [cf. (4) and (8)]. One can of course rely on standard solvers for convex problems. To alleviate the computational burden, one can alternatively choose different surrogates $f_i$s in (4) and (8) that deliver closed form solutions for $D_{(i)}^{\nu}$ and $X_{(i)}^{\nu+1}$. More specifically, considering (8), one can linearize $f_i$, that is,

$$
\tilde{f}_i(D_{(i)}; D_{(i)}^{\nu}, X_{(i)}^{\nu}) = \left( \nabla_D f_i(D_{(i)}; X_{(i)}^{\nu}), D_{(i)} - D_{(i)}^{\nu} \right) + \frac{\tau_{D,i}^\nu}{2} \|D_{(i)} - D_{(i)}^{\nu}\|^2,
$$

which leads to the following closed form solution for $D_{(i)}^{\nu}$:

$$
\hat{D}_i^{\nu} = P_D \left[ D_{(i)}^{\nu} - \frac{1}{\tau_{D,i}^\nu} \left( \nabla_D f_i(D_{(i)}; X_{(i)}^{\nu}) + \tilde{\nabla}_D f_i(D_{(i)}^{\nu}, X_{(i)}^{\nu}) \right) \right].
$$

Consider now the sparse coding subproblem (4). If $\tilde{h}_i$ is chosen as in (5), the update of the local variables $X_{(i)}^{\nu+1}$ reduces to solving a LASSO problem (see, e.g., [15], [16] for recent efficient algorithms for large-scale LASSO problems). To avoid solving a LASSO problem, we can alternatively use the linearization of $f_i$ as $\tilde{h}_i$, that is,

$$
\tilde{h}_i(X_{(i); U_{(i)}^{\nu}}, X_{(i)}^{\nu}) = \left( \nabla X, f_i(U_{(i)}; X_{(i)}^{\nu}), X_{(i)} - X_{(i)}^{\nu} \right) + \frac{\tau_{X,i}^\nu}{2} \|X_{(i)} - X_{(i)}^{\nu}\|^2,
$$

which leads to the following closed form solution for $X_{(i)}^{\nu+1}$:

$$
X_{(i)}^{\nu+1} = \frac{\tau_{X,i}^\nu}{2\mu + \tau_{X,i}^\nu} \cdot \tilde{T}_X \left( X_{(i)}^{\nu} - \frac{1}{\tau_{X,i}^\nu} \nabla X_i, f_i(U_{(i)}; X_{(i)}^{\nu}) \right),
$$

where $\tilde{T}_X(x) \triangleq \max(|x| - \theta, 0) \cdot \text{sign}(x)$ is the soft-thresholding operator and is applied element-wise in (16). In Sec. IV we present some numerical results comparing the two versions of Algorithm 1, based on the choices of $\tilde{h}_i$ in (5) and (15), respectively. We remark that the convergence results stated in Theorem I remain valid for the aforementioned new choices of surrogate functions; see the journal version of this work.

Tuning of free parameters: There are three set of parameters to tune in Algorithm 1, namely: i) the step-size $\gamma^\nu$; ii) the proximal coefficients $(\tau_{X,i}^\nu)_{i=1}^\nu$ and $(\tau_{D,i}^\nu)_{i=1}^\nu$; and iii) the weights $(w_{ij})_{i,j=1}^{I}$. Theorem 1 offers some flexibility in the choice of these parameters (cf. Assumptions B and C). For instance, the condition on the step-size—Assumption C1—ensures that the sequence decays to zero, but not too fast. There are many diminishing step-size rules in the literature satisfying this condition; see, e.g., [25]. An effective step-size rule that we used in our experiments (see [15], [16]) is $\gamma^\nu = \gamma^\nu - 1(1 - \epsilon e^{\nu - 1})$ with $\gamma^0 \in (1, 2]$ and $\epsilon \in (0, 1/\gamma^0)$. The proximal coefficients can be set as $\tau_{X,i}^\nu = \tau_i$, for all $i$ and $\nu$, where $\tau_i > 0$; and the explicit expression for $\tau_{D,i}^\nu$ is given in Assumption C2. Note that the above choices of step-size and proximal coefficients do not require any form of centralized coordination among the agents, which is a key feature in our distributed environment. Finally, referring to the weights $(w_{ij})_{i,j=1}^{I}$, several choices satisfying Assumption B are available, see [24] and references therein for some examples.

IV. Numerical Results

In this section we present some numerical results comparing Algorithm 1 with the (distributed) ATC algorithm [21]. For Algorithm 1 we simulated two instances, namely: i) one based on the surrogates (13) and (5), which we will refer to as “Plain D^2L”; and ii) one using the surrogates (13) and (15), which will be termed “Linearized D^2L”.

Setting and tuning: We consider denosing a $512 \times 512$ pixels corrupted boat image in a distributed setting. The data set $S$ is composed of the stacked $8 \times 8$ sliding patches of the image. The size of the dictionary and the sparse representation matrices $X_i$ are $64 \times 64$ and $64 \times 255,150$, respectively (overall, the number of variables is around 16 million), and the parameters in (12) are set to $2\mu = \lambda = 1/8$ and $\alpha = 1$. We simulated a time-invariant undirected connected network composed of 150 agents. For all the algorithms, the local copies $D_{(i)}^{\nu}$’s are initialized to random patches of the local data and $X_{(i)}^{\nu}$’s are initialized to zero. In both versions of our algorithms, the diminishing step size sequence $\gamma^\nu$ is generated according to $\gamma^\nu = \gamma^{\nu - 1}(1 - \epsilon e^{\nu - 1})$ with $\gamma^0 = 0.5$ and $\epsilon = 0.1$. The weights $(w_{ij})_{i,j=1}^{I}$ in the consensus steps are chosen according to the Metropolis rule [26].

Merit Function: We introduce the following functions to measure progresses of the algorithms towards stationarity of (P1) and attainment of consensus. Using (15), it is not difficult to check that $\Delta^\nu = \|\text{vec}(\Delta_D^\nu, \Delta_X^\nu)\|_\infty$ is a valid distance from stationarity, where

$$
\Delta_D^\nu = \tilde{D}^\nu - \hat{D}^\nu,
\Delta_X^\nu = X^\nu - \hat{X}^\nu
$$

and

$$
\tilde{D}^\nu \triangleq \arg\min_{D \in D} \sum_{i=1}^I \tilde{f}_i(D; \tilde{D}^\nu, X_{(i)}^{\nu}),
\hat{X}^\nu \triangleq \arg\min_{X \in \mathcal{X}} \sum_{i=1}^I \tilde{h}_i(X_{(i); U_{(i)}^{\nu}}, X_{(i)}^{\nu}) + \lambda \|X\|_{1,1} + \mu \|X\|_F^2.
$$

In Fig. 1 we plot, for each algorithm, the above merit functions [and the objective function of (P1) evaluated at $(\tilde{D}^\nu, X^\nu)$] versus the number of agents’ message exchanges. For ATC, the number of message exchanges coincides with the iterations $\nu$, while for our schemes, it is $2 \cdot \nu$. The figures clearly show that both versions of Algorithm 1 are much faster than ATC while being also guaranteed to converge (or, equivalently, they require less information exchanges than ATC). Note that the computational cost per iteration of Plain
D²L is comparable with that of ATC (both require to solve a LASSO), whereas that of Linearized D²L is cheaper. Note also that ATC does not seem to reach a consensus on the local copies of the dictionary, whereas for our D²L schemes consensus is reached quite early and then maintained.

To measure the quality of the reconstruction, we report in Table I the PSNR and the MSE values of the reconstructed images achieved by the three algorithms, after 200 and 1000 message exchanges.

| Linearized D²L | Plain D²L | ATC |
|-----------------|----------|-----|
| 200 message exchanges | PSNR=27.28db, MSE=121.4 | PSNR=27.32db, MSE=120.2 | PSNR=26.48db, MSE=146.2 |
| 1000 message exchanges | PSNR=27.53db, MSE=114.6 | PSNR=27.65db, MSE=111.69 | PSNR=27.29db, MSE=121.23 |

TABLE I: Reconstructed image quality from a noisy image, with PSNR=20.34db and MSE=601.1.

The comparisons above shows that both D²L schemes attain good quality solutions already after 200 message exchanges, while ATC lags significantly behind. The superior performance of the D²L schemes seems mainly due to the employed gradient tracking mechanism—instead of neglecting \( \sum_{j \neq i} \hat{f}_j \) in (P2) (like in ATC), each agent \( i \) tracks the gradient \( \nabla \hat{f}_j \).

V. CONCLUSIONS

The paper studied the distributed DL problem over (possibly) time-varying networks. We proposed the first distributed algorithmic framework with provable convergence for this class of problems. Preliminary numerical results show promising performance for the proposed scheme.

REFERENCES

[1] M. Elad, Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing. Springer, 1st ed., 2010.
[2] M. Elad and M. Aharon, “Image denoising via sparse and redundant representations over learned dictionaries,” IEEE Transactions on Image Processing, vol. 15, pp. 3736–3745, December 2006.
[3] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, “Online learning for matrix factorization and sparse coding,” Journal of Machine Learning Research, vol. 11, pp. 19–60, January 2010.
[4] H. Shen and J. Z. Huang, “Sparse principal component analysis via regularized low rank matrix approximation,” Journal of Multivariate Analysis, vol. 6, pp. 1015–1034, July 2008.
[5] M. Lee, H. Shen, J. Z. Huang, and J. S. Marron, “Biclustering via sparse singular value decomposition,” Biometrics, vol. 66, pp. 1087–1095, December 2010.
[6] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, “Supervised dictionary learning,” in Proceedings of Advances in Neural Information Processing Systems (NIPS), pp. 1033–1040, December 2008.
[7] J. A. Bagnell and D. M. Bradley, “Differentiable sparse coding,” in Advances in Neural Information Processing Systems 21, pp. 113–120, Curran Associates, 2009.
[8] H. Zou and T. Hastie, “Regularization and variable selection via the elastic net,” Journal of the Royal Statistical Society, Series B, vol. 67, pp. 301–320, 2005.
[9] M. Aharon, M. Elad, and A. Bruckstein, “K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation,” IEEE Transactions on Signal Processing, vol. 54, pp. 4311–4322, November 2006.
[10] S. Arora, R. Ge, and A. Moitra, “New algorithms for learning incoherent and overcomplete dictionaries,” in Proceedings of 27th Annual Conference on Learning Theory (COLT), pp. 779–806, June 2014.
[11] Q. Zhang and B. Li, “Discriminative k-svd for dictionary learning in face recognition,” 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2691–2698, June 2010.
[12] M. Yaghoobi, T. Blumensath, and M. E. Davies, “Dictionary learning for sparse approximations with the majorization method,” IEEE Transactions on Signal Processing, vol. 57, pp. 2178–2191, June 2009.
[13] S. P. Kasiviswanathan, H. Wang, A. Banerjee, and P. Melville, “Online 11-dictionary learning with application to novel document detection,” in Advances in Neural Information Processing Systems 25, pp. 2258–2266, Curran Associates, December 2012.
[14] M. Razaviyayn, H. W. Tseng, and Z. Q. Luo, “Dictionary learning for sparse representation: Complexity and algorithms,” in 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5247–5251, May 2014.
[15] F. Facchinei, G. Scutari, and S. Sagratella, “Parallel selective algorithms for nonconvex big data optimization,” IEEE Transactions on Signal Processing, vol. 63, pp. 1874–1889, April 2015.
[16] A. Daneshmand, F. Facchinei, V. Kungurtsev, and G. Scutari, “Hybrid random/deterministic parallel algorithms for convex and nonconvex big data optimization,” IEEE Transactions on Signal Processing, vol. 63, pp. 3914–3929, August 2015.
[17] J. Liang, M. Zhang, X. Zeng, and G. Yu, “Distributed dictionary learning for sparse representation in sensor networks,” IEEE Transactions on Image Processing, vol. 23, pp. 2528–2541, June 2014.
[18] Z. Shakeri, H. Raja, and W. U. Bajwa, “Dictionary learning based nonlinear classifier training from distributed data,” in 2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP), pp. 759–763, December 2014.
[19] J. Chen, Z. J. Towfic, and A. H. Sayed, “Dictionary learning over distributed models,” IEEE Transactions on Signal Processing, vol. 63, pp. 1001–1016, February 2015.
[20] H. T. Wai, T. H. Chang, and A. Scaglione, “A consensus-based decentralized algorithm for non-convex optimization with application to dictionary learning,” in 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 3546–3550, April 2015.
[21] P. Chainais and C. Richard, “Distributed dictionary learning over a sensor network.” 2013. Available online at http://arxiv.org/pdf/1304.3568
[22] M. Zhu and S. Martinez, “An approximate dual subgradient algorithm for distributed non-convex constrained optimization,” IEEE Transactions on Automatic Control, vol. 58, pp. 1534–1539, June 2013.
[23] P. Bianchi and J. Jakubowicz, “Convergence of a multi-agent projective stochastic gradient algorithm for non-convex optimization,” IEEE Transactions on Automatic Control, vol. 58, pp. 391–405, February 2013.
[24] P. D. Lorenzo and G. Scutari, “NEXT: In-network nonconvex optimization,” IEEE Transactions on Signal and Information Processing over Networks, vol. 2, pp. 120–136, June 2016.
[25] D. P. Bertsekas and J. N. Tsitsiklis, Parallel and distributed computation: numerical methods. Athena Scientific, 1997.
[26] L. Xiao, S. Boyd, and S.-J. Kim, “Distributed average consensus with least-mean-square deviation,” Journal of Parallel and Distributed Computing, vol. 67, pp. 33–46, January 2007.