A CONSENSUS-BASED DECENTRALIZED ALGORITHM FOR NON-CONVEX OPTIMIZATION WITH APPLICATION TO DICTIONARY LEARNING

Hoi-To Wai[†], *Tsung-Hui Chang*[‡], *Anna Scaglione*[†]

[†]School of Elect., Comp. and Energy. Eng., Arizona State University, Tempe, Arizona, USA.
[‡]Dept. of Elect. and Comp. Eng., National Taiwan Univ. of Science and Technology, Taiwan (R.O.C.).
Emails: htwai@asu.edu, tsunghui.chang@ieee.org, Anna.Scaglione@asu.edu

ABSTRACT

In handling massive-scale signal processing problems arising from 'big-data' applications, key technologies could come from the development of decentralized algorithms. In this context, consensus-based methods have been advocated because of their simplicity, fault tolerance and versatility. This paper presents a new consensus-based decentralized algorithm for a class of non-convex optimization problems that arises often in inference and learning problems, including 'sparse dictionary learning' as a special case. For the proposed algorithm, we provide sufficient conditions for convergence to a stationary point. Numerical results demonstrate the efficacy of the proposed algorithm and provide evidence that validates our convergence claim.

Index Terms— dictionary learning, decentralized algorithm, non-convex optimization

1. INTRODUCTION

In recent years, one of the biggest challenges in signal processing is on finding ways to deal with enormous amount of data generated by sources such as online social media, mobile apps, cyber physical systems, etc [1,2]. The massive memory and computational requirements of these problems calls for a paradigm shift in the design of signal processing algorithms. Specifically, it calls for decentralized methods that can leverage on the collective computational and storage power of a cluster of computers (a.k.a. the cloud) to expedite the process.

In light of the need for large scale signal processing, the idea of consensus-based optimization provides an attractive option for developing decentralized methods [3, 4]. The main idea is to enforce consensus via information diffusion over a network of computing nodes, while simultaneously performing locally first-order updates, like a gradient descent. Such methods have the advantages that they require only local computation and can often be implemented with low complexity [3, 4].

This paper considers the problem of dictionary learning (DL) [5, 6] as an example of the large-scale signal processing problems that may be applied to massive amounts of data. In DL, the goal is to find an appropriate basis (a.k.a. the dictionary) for the signals observed (e.g., a sequence of images, or a collection of articles) such that a sparse representation is possible for all the information contained in the database. Having learnt the dictionary from a set of training data, we can go on to compress the whole set of data, storing only the sparse coefficients vectors. The DL problem can be cast as a *bi-convex* optimization problem, tackled by many existing methods (see e.g. the overview in [5, Chapter 12]). However, most of the existing algorithms for DL are centralized. Motivated by the impending need of finding parallel computation methods for these problems that would scale as the amount of data grows large, we study the decentralized implementation of *bi-convex* optimization problems.

This work is supported by NSF grant CCF 1018111 and Taiwan Ministry of Science and Technology NSC 102-2221-E-011-005-MY3.

More specifically, in this paper our aim is to develop a consensus-based optimization algorithm for a class of large-scale, non-convex optimization problems. The latter includes the DL problem as a special case. Our method combines the exact first-order algorithm (EXTRA) algorithm in [7] with a local proximal gradient update. Through theoretical analysis and numerical experiments, we show that the developed algorithm converges to a stationary point of the DL problem. To the best of our knowledge, the algorithm we proposed is the first of this kind that comes with evidence (at least empirically) of reaching convergence to stationary points.

1.1. Relation to Prior Work

Consensus-based decentralized algorithms to solving optimization problems have been proposed in a number of previous works, e.g., [3, 4, 7–12]. The convergence of these algorithms have been established in [3, 4, 7, 11, 12]. Some recent developments include the EXTRA algorithm proposed in [7], which guarantees convergence of the consensus-based gradient descent algorithm using fixed step size. Combined algorithms that mix consensus technique with other optimization methods have also been developed, e.g., the alternating direction method of multipliers (ADMM) [13–15], the primal-dual method [16] and the Gauss-Newton method [17]. Most of this prior art focuses on convex optimization problems. Only a few related works on consensus-based decentralized algorithms exist for non-convex optimization, e.g., [14, 17–19]. In particular, a stochastic decentralized algorithm have been proposed in [18] for tackling a class of non-convex problems.

In this work, we focus on the commonly encountered class of *bi-convex* optimization problem [20]. Examples of such optimization problems include the non-negative matrix factorization [21], low-rank matrix completion [14], DL problem [5, 6], etc. For the DL problem, [19] proposes an algorithm that most resembles ours, since it employs a consensus-based (proximal) gradient descent. This method will be described in detail later. Lastly, [14] has studied a consensus-based decentralized algorithm for low-rank matrix completion, which can be applied to the DL problem. However, the algorithm in [14] has higher complexity compared to ours.

2. PROBLEM FORMULATION

Consider N networked agents for which the interconnectivity between the agents is described by an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Each agent i is assigned with a local cost function $f_i(\mathbf{x}, \mathbf{y}_i)$ that depends on a global variable $\mathbf{x}$ and a local variable $\mathbf{y}_i$. The global cost function is defined as the sum of these local cost function. Our aim is to tackle the following optimization problem:

$$\min_{\mathbf{x},\{\mathbf{y}_i\}_{i=1}^N} \sum_{i=1}^{N} \big(f_i(\mathbf{x},\mathbf{y}_i) + h_i(\mathbf{y}_i)\big), \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^m$ and $\mathbf{y}_i \in \mathbb{R}^n$ are the optimization variables. We have assumed the following regarding (1):

1. The function $f_i(\mathbf{x},\mathbf{y}_i)$ is bi-convex and it may not be jointly convex in $(\mathbf{x},\mathbf{y}_i)$.
2. The function $h_i(\mathbf{y}_i)$ is convex and can possibly be non-smooth.
3. The function $f_i(\mathbf{x},\mathbf{y}_i)$ is continuously differentiable with respect to (w.r.t.) $\mathbf{x}$ and $\mathbf{y}_i$. Moreover, its gradient is Lipschitz continuous with the constants L_x and L_y, respectively.

This paper focuses on a decentralized algorithm for solving (1). Our study is motivated by the following dictionary learning (DL) problem:

$$\min_{\mathbf{X},\mathbf{Y}} \frac{1}{2}\|\mathbf{S}-\mathbf{X}\mathbf{Y}\|_F^2 + \lambda\|\mathbf{Y}\|_1 + \sum_{\ell=1}^{M} \gamma_\ell \|\mathbf{X}_{:,\ell}\|_2^2. \tag{2}$$

where $[\mathbf{X}]_{:,\ell}$ denotes the ℓth column in matrix $\mathbf{X}$. The regularization terms $\|\mathbf{Y}\|_1$ and $\|\mathbf{X}_{:,\ell}\|_2^2$ are introduced respectively to promote sparsity and to ensure that the solution $\mathbf{X}$ is bounded. In the data model underlying (2), the matrix $\mathbf{S} \in \mathbb{R}^{m\times M}$ contains M columns of training data, in which each of them is assumed to be a linear combination of the column vectors in the dictionary $\mathbf{X} \in \mathbb{R}^{m\times n}$, with the coefficients contained in the columns of $\mathbf{Y} \in \mathbb{R}^{n\times M}$. It is assumed that $\mathbf{S}$ is sparse such that each column of $\mathbf{Y}$ is formed by combining only a few columns from the dictionary $\mathbf{X}$, i.e., we have $\mathbf{S} = \mathbf{X}\mathbf{Y}$.

As an application for the algorithm developed, this paper considers a distributed implementation of (2) in a sensor network setting. In particular, each agent collects training data $\mathbf{S}_i$ independently while all agents want to learn a common dictionary $\mathbf{X}$. We write $\mathbf{S} = [\mathbf{S}_1\ \mathbf{S}_2\ \cdots\ \mathbf{S}_N]$ and $\mathbf{Y} = [\mathbf{Y}_1\ \mathbf{Y}_2\ \cdots\ \mathbf{Y}_N]$. To this end, we can rewrite (2) as follows:

$$\min_{\mathbf{X},\{\mathbf{Y}_i\}_{i=1}^N} \frac{1}{2}\sum_{i=1}^{N}\Big(\|\mathbf{S}_i-\mathbf{X}\mathbf{Y}_i\|_F^2+\lambda\|\mathbf{Y}_i\|_1\Big)+\sum_{\ell=1}^{n}\gamma_\ell\|\mathbf{X}_{:,\ell}\|_2^2. \tag{3}$$

This is a special case of (1) with $f_i(\mathbf{X},\mathbf{Y}_i) = (1/2)\|\mathbf{S}_i-\mathbf{X}\mathbf{Y}_i\|_F^2 + (1/N)\sum_{\ell=1}^{n}\gamma_\ell\|\mathbf{X}_{:,\ell}\|_2^2$ and $h_i(\mathbf{Y}_i) = \lambda\|\mathbf{Y}_i\|_1$.

3. PROPOSED ALGORITHM

Next we describe the proposed algorithm, called EXTRA-AO, for solving (1) using consensus-based information exchange.

3.1. Decentralized Alternating Optimization Algorithms

We first observe that (1) is a *non-separable* and *non-convex* problem. The non-separability prevents us from solving (1) directly via decentralized optimization, while the non-convexity prevents us from applying safely existing decentralized algorithms.

A natural way to dealing with (1) is to employ an alternating optimization (AO) approach — we fix $\mathbf{x}$ while optimizing $\{\mathbf{y}_i\}_{i=1}^N$; then we fix $\{\mathbf{y}_i\}_{i=1}^N$ while optimizing $\mathbf{x}$. As the objective function is bi-convex, algorithms such as (proximal) gradient methods can be applied to each of the two sub-optimizations above. For instance, the following recursion is frequently used, e.g., [22]:

$$\mathbf{x}^k = \mathbf{x}^{k-1} - \alpha_k \textstyle\sum_{i=1}^{N} \nabla_{\mathbf{x}} f_i(\mathbf{x}^{k-1},\mathbf{y}_i^{k-1}) \tag{4}$$

$$\mathbf{y}_i^k = \mathbf{prox}_{\beta_k h_i(\cdot)}\big(\mathbf{y}_i^{k-1} - \beta_k \nabla_{\mathbf{y}} f_i(\mathbf{x}^k,\mathbf{y}_i^{k-1})\big),\ \forall\, i, \tag{5}$$

where $\alpha_k, \beta_k > 0$ are the step sizes to be specified later. Notice that in (5), the proximal operator is defined as:

$$\mathbf{prox}_{\beta_k h_i(\cdot)}(\mathbf{y}) = \arg\min_{\mathbf{z}\in\mathcal{C}_i} \frac{1}{2}\|\mathbf{y}-\mathbf{z}\|_2^2 + \beta_k h_i(\mathbf{z}). \tag{6}$$

A popular example is $h_i(\mathbf{y}_i) = \|\mathbf{y}_i\|_1$. In this case, the proximal operator is equivalent to the soft thresholding operator [23]. The latter can be computed in closed form.

It is obvious that once $\mathbf{x}^{k-1}$ becomes available at agent i, the update for $\mathbf{y}_i^k$ is distributable (cf. (5)). The important issue is how to tackle the optimization w.r.t. $\mathbf{x}$ in a decentralized manner.

We observe that $\mathbf{x}$ is a global variable that has to be agreed upon with the other agents in the network. A possible solution to optimizing $\mathbf{x}$ is to apply the decentralized gradient descent algorithms, e.g., in [24]. In fact, this is the method proposed in [19] to tackle the distributed DL problem (cf. (3)). The following adapt-then-combine (ATC) strategy is taken to compute $\mathbf{x}_i^k$ at agent i. We replace (4) with:

$$\begin{cases}\hat{\mathbf{x}}_i^k = \mathbf{x}_i^{k-1} - \alpha_k \nabla_{\mathbf{x}} f_i(\mathbf{x}_i^{k-1},\mathbf{y}_i^{k-1}),\\ \mathbf{x}_i^k = \sum_{j=1}^{N} W_{ij}\hat{\mathbf{x}}_j^k,\end{cases} \tag{7}$$

where the symmetric matrix $\mathbf{W} \in \mathbb{R}^{N\times N}$ satisfying $\mathbf{W}\mathbf{1} = \mathbf{1}$ and $\mathbf{1}^T\mathbf{W} = \mathbf{1}^T$ is a mixing matrix. The choice of $\mathbf{W}$ is constrained by the network topology $\mathcal{G}$ such that $W_{ij} \neq 0$ only when $(i,j) \in \mathcal{E}$[1].

We note that the first equation in (7) computes a local update for $\mathbf{x}_i$ based on the local variable $\mathbf{y}_i^k$, then the second equation promotes consensus amongst the local updates computed by the neighboring agents. Due to the fact that $\mathbf{W}$ is constrained by the network topology, (7) can be implemented via local computations and information exchange. Furthermore, parallel processing is also possible as (5) and (7) are equations that can be processed locally except for the combination step in (7).

However, the recursions (5) and (7) are not guaranteed to converge to a stationary point of (1) in general. Under a fixed step size rule, i.e., $\alpha_k = \alpha$ for all k, it can be shown that the recursion (5) and (7) may converge to a solution such that $\mathbf{x}_i^k \neq \mathbf{x}_j^k$, i.e., consensus cannot be reached; on the other hand, when the step size α_k is diminishing, our numerical experiments suggest that the algorithm may not converge to a stationary point of (1) at all. An example of lack of convergence is the numerical simulations shown in Fig. 1, illustrating that the ATC-AO with a fixed step size does not lead to a solution that reaches consensus.

3.2. EXTRA-AO Algorithm

The previous discussion suggests that one has to apply a different strategy for the decentralized update of $\mathbf{x}$. For this reason we leverage an alternative idea proposed in [7], called the exact first order algorithm (EXTRA). The algorithm is described in the next page by (8) together with the pseudo code of EXTRA-AO in Algorithm 1. An important feature of the EXTRA update is that a fixed step size is used throughout the algorithm. As seen in [7], this strategy can achieve consensus and optimality simultaneously when applied to convex optimization problems.

[1]Note that self-edge is assumed to be present in $\mathcal{G}$ such that $(i,i) \in \mathcal{E}$.

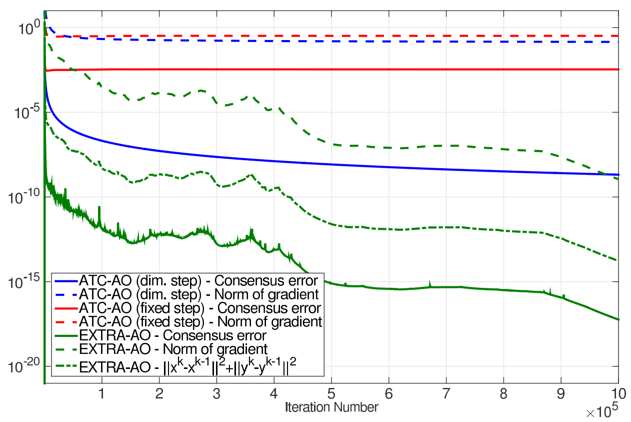

Fig. 1: Convergence behavior of the algorithms for distributed DL (3). The simulation details are given in Section 4. We compare the 'consensus error', $(1/N)\sum_{i=1}^N \|\mathbf{x}_i^k - (1/N)\sum_{j=1}^N \mathbf{x}_j^k\|^2$, and the 'norm of gradient' w.r.t. $\mathbf{x}$, $\|\sum_{i=1}^N \nabla_{\mathbf{x}} f_i(\mathbf{x}_i^k, \mathbf{y}_i^k)\|^2$, against the iteration number. In addition, the 'norm of difference' for EXTRA-AO denotes $\|\boldsymbol{x}^k - \boldsymbol{x}^{k-1}\|_2^2 + \|\boldsymbol{y}^k - \boldsymbol{y}^{k-1}\|_2^2$.

We now discuss about the EXTRA update for $\mathbf{x}$. The EXTRA step (8) combines both consensus and gradient descent, where the optimization variables from the previous two iterations are required. Note that, in (8), $\tilde{\mathbf{W}} = (\mathbf{I} + \mathbf{W})/2$ has the same sparsity as $\mathbf{W}$, therefore similar to the ATC strategy described previously, the EXTRA update can also be computed via local computations and information exchange with the neighboring agents.

Next, we provide some insights into the convergence of EXTRA-AO. To facilitate our discussions, let us introduce the following variables/functions:

$$\boldsymbol{x}^k \triangleq [\mathbf{x}_1^k\ \mathbf{x}_2^k\ \cdots\ \mathbf{x}_N^k]^T,\ \boldsymbol{y}^k \triangleq [\mathbf{y}_1^k\ \mathbf{y}_2^k\ \cdots\ \mathbf{y}_N^k]^T, \tag{10}$$

$$\mathbf{f}(\boldsymbol{x},\boldsymbol{y}) \triangleq [f_1(\mathbf{x}_1,\mathbf{y}_1)\ \cdots\ f_N(\mathbf{x}_N,\mathbf{y}_N)]^T, \tag{11}$$

$$\nabla_{\boldsymbol{x}}\mathbf{f}(\boldsymbol{x},\boldsymbol{y}) \triangleq [\nabla_{\mathbf{x}} f_1(\mathbf{x}_1,\mathbf{y}_1)\ \cdots\ \nabla_{\mathbf{x}} f_N(\mathbf{x}_N,\mathbf{y}_N)]^T, \tag{12}$$

$$f(\boldsymbol{x},\boldsymbol{y}) \triangleq \textstyle\sum_{i=1}^N f_i(\mathbf{x}_i,\mathbf{y}_i) = \mathbf{1}^T\mathbf{f}(\boldsymbol{x},\boldsymbol{y}). \tag{13}$$

Notice that both $\mathbf{x}_i$ and $\mathbf{y}_i$ are given as column vectors, therefore $\boldsymbol{x}^k \in \mathbb{R}^{N\times m}$ and $\boldsymbol{y}^k \in \mathbb{R}^{N\times n}$. A sufficient condition for EXTRA-AO to reach a stationary point of (1) is as follows:

Proposition 1 *Assume that* $\text{null}\{\mathbf{I} - \mathbf{W}\} = \text{span}\{\mathbf{1}\}$. *Suppose that the sequence* $\{(\boldsymbol{x}^k, \boldsymbol{y}^k)\}_k$ *generated by EXTRA-AO converges to a point* $(\boldsymbol{x}^\infty, \boldsymbol{y}^\infty)$, *then* $(\hat{\mathbf{x}}^\infty, \boldsymbol{y}^\infty)$ *is a stationary point to problem* (1), *where* $\hat{\mathbf{x}}^k \triangleq (1/N)\mathbf{1}^T\boldsymbol{x}^k$.

Proof. We observe that the update equation (8) can be rewritten as follows. For example, at $k = 2$, we have

$$\begin{aligned}\boldsymbol{x}^2 &= \boldsymbol{W}\boldsymbol{x}^1 - \alpha\nabla_{\boldsymbol{x}}\mathbf{f}(\boldsymbol{x}^1,\boldsymbol{y}^1) + \boldsymbol{x}^1 - (\tilde{\mathbf{W}}\boldsymbol{x}^0 - \alpha\nabla_{\boldsymbol{x}}\mathbf{f}(\boldsymbol{x}^0,\boldsymbol{y}^0))\\ &= \mathbf{W}\boldsymbol{x}^1 - \alpha\nabla_{\boldsymbol{x}}\mathbf{f}(\boldsymbol{x}^1,\boldsymbol{y}^1) + (\mathbf{W} - \tilde{\mathbf{W}})\boldsymbol{x}^0,\end{aligned}$$

where the second equality is due to (8) with $k = 1$. By induction on $k = 3, 4, ...$, we obtain:

$$\boldsymbol{x}^{k+1} = \mathbf{W}\boldsymbol{x}^k - \alpha\nabla\mathbf{f}(\boldsymbol{x}^k,\boldsymbol{y}^k) + (\mathbf{W} - \tilde{\mathbf{W}})\textstyle\sum_{t=0}^{k-1}\boldsymbol{x}^t \tag{14}$$

Since $\mathbf{1}^T(\mathbf{W} - \tilde{\mathbf{W}}) = \mathbf{0}$, multiplying $(1/N)\mathbf{1}^T$ from the left of both side yields

$$\hat{\mathbf{x}}^{k+1} = \hat{\mathbf{x}}^k - (1/N)\mathbf{1}^T\nabla_{\boldsymbol{x}}\mathbf{f}(\boldsymbol{x}^k,\boldsymbol{y}^k), \tag{15}$$

Note that (15) is analogous to the centralized gradient descent in (4).

Algorithm 1 The EXTRA-AO algorithm for (1).

```
1: Initialize: $\{\mathbf{x}_i^0\}_{i=1}^N$, $\{\mathbf{y}_i^0\}_{i=1}^N$;
2: for $k = 1, 2, ...$ do
3:     for $i = 1, 2, ..., N$ do
4:         Agent $i$ computes the following EXTRA update for $\mathbf{x}_i$:
           $$\mathbf{x}_i^k = \begin{cases}\sum_{j=1}^N W_{ij}\mathbf{x}_j^{k-1} - \alpha\nabla_{\mathbf{x}} f_i(\mathbf{x}_i^{k-1},\mathbf{y}_i^{k-1}),\ \text{if } k = 1,\\ \mathbf{x}_i^{k-1} + \sum_{j=1}^N W_{ij}\mathbf{x}_j^{k-1} - \alpha\nabla_{\mathbf{x}} f_i(\mathbf{x}_i^{k-1},\mathbf{y}_i^{k-1})\\ -\sum_{j=1}^N \tilde{W}_{ij}\mathbf{x}_j^{k-2} + \alpha\nabla_{\mathbf{x}} f_i(\mathbf{x}_i^{k-2},\mathbf{y}_i^{k-2}),\ \text{if } k > 1,\end{cases} \tag{8}$$
           where $\alpha > 0$ is a fixed step size and $\tilde{\mathbf{W}} = (\mathbf{I} + \mathbf{W})/2$. Notice that $\tilde{\mathbf{W}}$ can also take a different form with more relaxed conditions, see [7].
5:         Agent $i$ computes the following update for $\mathbf{y}_i$:
           $$\mathbf{y}_i^k = \mathbf{prox}_{\beta h_i(\cdot)}\left(\mathbf{y}_i^{k-1} - \beta\nabla_{\mathbf{y}} f_i(\mathbf{x}_i^k,\mathbf{y}_i^{k-1})\right) \tag{9}$$
6:     end for
7: end for
8: Return: $\{\mathbf{x}_i^k\}_{i=1}^N$, $\{\mathbf{y}_i^k\}_{i=1}^N$.
```

Now, if the sequence $\{\boldsymbol{x}^k, \boldsymbol{y}^k\}_k$ converges to a unique limit point $(\boldsymbol{x}^\infty, \boldsymbol{y}^\infty)$ as $k \to \infty$, in the limit the EXTRA update in (8) is

$$(\mathbf{W} - \tilde{\mathbf{W}})\boldsymbol{x}^\infty = \mathbf{0}. \tag{16}$$

As $\text{null}\{\mathbf{W} - \tilde{\mathbf{W}}\} = \text{span}\{\mathbf{1}\}$[2], the above implies $\mathbf{x}_i^\infty = \hat{\mathbf{x}}^\infty$ for all i, i.e., consensus is achieved as $k \to \infty$. Lastly, applying this to (15) with $k \to \infty$ gives:

$$\mathbf{0} = (1/N)\mathbf{1}^T\nabla_{\boldsymbol{x}}\mathbf{f}(\boldsymbol{x}^\infty,\boldsymbol{y}^\infty), \tag{17}$$

which implies that $\hat{\mathbf{x}}^\infty$ is a stationary point of (1), given $\boldsymbol{y}^\infty$.

On the other hand, if $\boldsymbol{y}^k$ is convergent, its limit $\boldsymbol{y}^\infty$ is a fixed point to Eq. (9), given $\mathbf{x}_i^\infty$, i.e.,

$$\mathbf{y}_i^\infty = \mathbf{prox}_{\beta h_i(\cdot)}\big(\mathbf{y}_i^\infty - \beta\nabla_{\mathbf{y}} f_i(\mathbf{x}_i^\infty,\mathbf{y}_i^\infty)\big),\ \forall\, i. \tag{18}$$

As $\beta > 0$, the above guarantees that $\mathbf{y}_i^\infty$ is a stationary point of (1) given $\mathbf{x}_i^\infty = \hat{\mathbf{x}}^\infty$. Combining this observation with (17) implies that $(\hat{\mathbf{x}}^\infty, \boldsymbol{y}^\infty)$ is a stationary point to (1). **Q.E.D.**

Showing that the sufficient condition in Proposition 1 holds is part of the on-going research. We conclude this section with the following lemma on the choice of step size, whose proof is skipped.

Lemma 1. *Suppose that the step sizes* α, β *in EXTRA-AO satisfy*

$$0 < \alpha < (2\lambda_{min}(\tilde{\mathbf{W}})/L_x),\ 0 < \beta < (1/L_y), \tag{19}$$

[3]*then the following inequalities hold for the objective values of* (1) *at each iteration:*

$$\begin{aligned}f(\boldsymbol{x}^{k+1},\boldsymbol{y}^k) - f(\boldsymbol{x}^k,\boldsymbol{y}^k) &\le -\delta\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|_2^2\\ &\quad -\tfrac{1}{\alpha}\Big\langle(\tilde{\mathbf{W}} - \mathbf{W})\textstyle\sum_{t=0}^{k+1}\boldsymbol{x}^t, \boldsymbol{x}^{k+1} - \boldsymbol{x}^k\Big\rangle,\end{aligned} \tag{20}$$

$$f(\boldsymbol{x}^{k+1},\boldsymbol{y}^{k+1}) - f(\boldsymbol{x}^{k+1},\boldsymbol{y}^k) \le -(1/2)\|\boldsymbol{y}^k - \boldsymbol{y}^{k+1}\|_2^2, \tag{21}$$

where $\delta = (\lambda_{min}(\tilde{\mathbf{W}})/\alpha - L_x/2) > 0$ *is a constant.*

[2]Note that as $\mathbf{W} - \tilde{\mathbf{W}} = (\mathbf{W} - \mathbf{I})/2$, we have $\boldsymbol{x} \in \text{null}\{\mathbf{W} - \tilde{\mathbf{W}}\} = \text{null}\{\mathbf{W} - \mathbf{I}\} = \text{span}\{\mathbf{1}\}$ under the premise of Proposition 1.

[3]In practice, a small stepsize would suffice for the algorithm to converge.

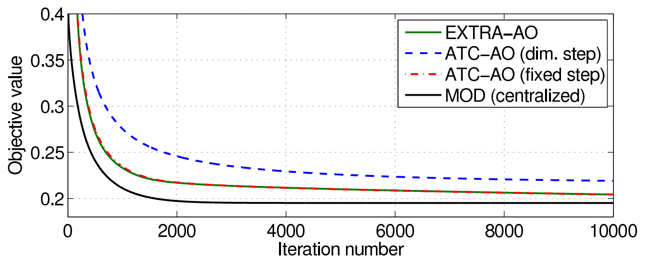

Fig. 2: The objective value against the iteration number for the distributed DL algorithms. The same step size selection as in Fig. 1 is used. Notice that the solution of ATC-AO with fixed step size does not reach consensus and is infeasible to (3).

The key to proving the above Lemma is to apply the descent lemma [25] and the global inequality [23] to Eqs. (8) and (9), respectively and exploiting the simplified EXTRA update (14).

The above lemma provides a guideline for choosing the step size for EXTRA-AO. In addition, when the latter term in (20) is non-negative or vanishing, the objective value achieved by EXTRA-AO is non-increasing, combining this with (21) imply that $\boldsymbol{y}^k$ converges as $\|\boldsymbol{y}^k - \boldsymbol{y}^{k+1}\| \to 0$.

3.3. Distributed DL using EXTRA-AO

To apply EXTRA-AO to the distributed DL problem (3), we note that the global variable $\mathbf{x}$ should be taken as the dictionary $\mathbf{X}$; while the local variable $\mathbf{y}_i$ is taken as the sparse coefficients $\mathbf{Y}_i$. For simplicity, we also take $\gamma_\ell = \gamma$ for all ℓ. Notice that the latter term in the objective function of (3) can now be written as $\gamma\|\mathbf{X}\|_F^2$. The gradient matrices of $f_i(\mathbf{A}_i, \mathbf{S}_i)$ w.r.t. $\mathbf{A}_i$ and $\mathbf{S}_i$ are needed in (8) and (9). They are given as follows:

$$\nabla_{\mathbf{X}} f_i(\mathbf{X}_i^k, \mathbf{Y}_i^k) = (\mathbf{X}_i^k \mathbf{Y}_i^k - \mathbf{S}_i)(\mathbf{Y}_i^k)^T + (\gamma/N)\mathbf{X}_i^k, \quad (22)$$

$$\nabla_{\mathbf{Y}} f_i(\mathbf{X}_i^k, \mathbf{Y}_i^k) = (\mathbf{X}_i^k)^T(\mathbf{X}_i^k \mathbf{Y}_i^k - \mathbf{S}_i) \quad (23)$$

Eq. (22) can be computed efficiently as $\mathbf{Y}_i$ is sparse. As for the proximal operation in (9), since $h_i(\mathbf{Y}_i) = \|\mathbf{Y}_i\|_1$, it can be replaced by the soft shrinkage operator [23].

4. NUMERICAL RESULTS

This subsection presents numerical results to demonstrate the efficacy of the proposed EXTRA-AO algorithm for DL. To prepare the training data, we have randomly extracted 300 overlapping patches, each with size 16×16, from the 512×512 image of `barbara.png`, as shown in Fig. 4. Each of the extracted patch is vectorized, thereby giving $\mathbf{S}$ a size of 256×300. We assume that there are $n = 64$ atoms, thereby giving a compression ratio of $1/4$. The size of the common dictionary $\mathbf{X}$ is 256×64. Notice that our algorithm is scalable to handle problems of larger scale.

For the distributed DL problem (3), we set $\lambda = 0.03$ and $\gamma = \gamma_\ell = 0.1$ as the regularization parameters. The columns of the training data matrix $\mathbf{Y}$ is divided into $N = 10$ equally sized partitions $\mathbf{S}_i \in \mathbb{R}^{256 \times 30}$. It corresponds to the scenario when 10 sensors are taking samples from the image for dictionary learning. In addition to learning the dictionary $\mathbf{X}$ distributively, each agent is responsible for computing the sparse matrix $\mathbf{Y}_i$ of size 64×30 only. The network connectivity graph $\mathcal{G}$ is generated as an Erdos-Renyi random graph with a connection probability of $p = 0.6$ as well as self-edges. The matrix $\mathbf{W}$ is constructed using the Metropolis-Hastings rule [26].

The performance of EXTRA-AO is compared to ATC-AO in [19] and the Method of Optimal Directions (MOD) in [27], where the latter is a centralized algorithm for DL. We initialized the algorithms with $\mathbf{X}$ set as the 2D discrete cosine transform (DCT) matrix. For EXTRA-AO and ATC-AO with fixed step size, we set

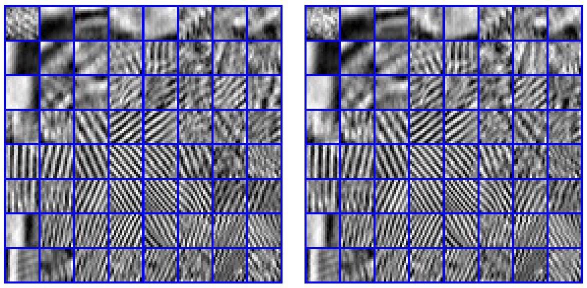

Fig. 3: The dictionary learnt from the image: (Left) using the ATC-AO algorithm after 2×10^4 iterations. (Right) using the EXTRA-AO algorithm after 2×10^4 iterations. Each 8×8 patch represents an atom in the dictionary.

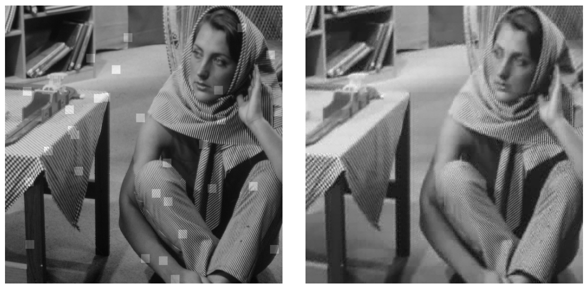

Fig. 4: Reconstructing the images: (Left) the original image with the 30 shaded masks representing the training samples taken by one agent. (Right) by sparse coding using the dictionary learnt in EXTRA-AO after 2×10^4 iterations.

$\alpha = 0.03$ and $\beta = 0.02$, ATC-AO with diminishing step size is set with $\alpha_k = \beta_k = 0.02 \cdot \frac{10}{(k/100)+10}$.

We first verify that the EXTRA-AO algorithm achieves convergence to a stationary point of (3). As shown in Fig. 1, the norm of difference between successive iterations decreases to 0 as $k \to \infty$, thereby satisfying the sufficient condition in Proposition 1. We also note that the solution $\mathbf{X}_i$ obtained at ATC-AO (with fixed step size) does not achieve consensus and is thus infeasible to (1).

In Fig. 2, we compare the objective value against the number of iteration. As seen, EXTRA-AO achieves a comparable objective value to the MOD algorithm. The estimated dictionary is depicted in Fig. 3, which shows a combination of rotated tiles that correspond to the common features found in `barbara.png`. Upon careful observation, we also found that some of the atoms (e.g., the $(3,4)$th atom) learnt by EXTRA-AO shows clearer edges than ATC-AO.

Lastly, Fig. 4 shows the reconstruction result after sparse decoding using the dictionary learnt before. To promote sparsity, the sparse decoding is performed with $\lambda = 0.1$, the resulting sparse code contains only 32.12% of non-zero entries, i.e., there are about 21 non-zero coefficients out of 64 for every 16×16 patch. As seen, the image reconstructed shows only a reasonable amount of artifacts compared to the original image.

5. CONCLUSION

In this paper, we have proposed a new consensus-based decentralized algorithm called EXTRA-AO. The algorithm is applicable to a class of non-convex optimization problems, which has only been considered in a few applications of consensus-based algorithms before. We have also provided a sufficient condition for the proposed EXTRA-AO to reach a stationary point. It was checked empirically to hold in our numerical results on the dictionary learning problem.

6. REFERENCES

[1] K. Slavakis, G. B. Giannakis, and G. Mateos, "Modeling and Optimization for Big Data Analytics: (Statistical) learning tools for our era of data deluge," *IEEE Signal Process. Mag.*, vol. 31, no. 5, pp. 18–31, Sep. 2014.

[2] V. Cevher, S. Becker, and M. Schmidt, "Convex Optimization for Big Data: Scalable, randomized, and parallel algorithms for big data analytics," *IEEE Signal Process. Mag.*, vol. 31, no. 5, pp. 32–43, Sep. 2014.

[3] S. S. Ram, A. Nedic, and V. V. Veeravalli, "A new class of distributed optimization algorithms : application to regression of distributed data," *Optimization Methods and Software*, no. 1, pp. 37–41, Feb. 2012.

[4] A. H. Sayed, S.-Y. Tu, J. Chen, X. Zhao, and Z. J. Towfic, "Diffusion strategies for adaptation and learning over networks: an examination of distributed strategies and network behavior," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 155–171, May 2013.

[5] M. Elad, *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*, 1st ed. Springer Publishing Company, 2010.

[6] I. Tosic and P. Frossard, "Dictionary Learning," *IEEE Signal Process. Mag.*, vol. 28, no. 2, pp. 27–38, Mar. 2011.

[7] W. Shi, Q. Ling, G. Wu, and W. Yin, "EXTRA: An Exact First-Order Algorithm for Decentralized Consensus Optimization," pp. 1–23, Apr. 2014. [Online]. Available: http://arxiv.org/abs/1404.6264

[8] A. G. Dimakis, S. Kar, J. M. F. Moura, M. G. Rabbat, and A. Scaglione, "Gossip Algorithms for Distributed Signal Processing," *Proc. IEEE*, vol. 98, no. 11, pp. 1847–1864, Nov. 2010.

[9] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and Cooperation in Networked Multi-Agent Systems," *Proc. IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007.

[10] J. Chen, Z. J. Towfic, and A. H. Sayed, "Dictionary Learning over Distributed Models," pp. 1–16, Feb. 2014. [Online]. Available: http://arxiv.org/abs/1402.1515

[11] S. S. Ram, A. Nedić, and V. V. Veeravalli, "Distributed Stochastic Subgradient Projection Algorithms for Convex Optimization," *Journal of Optimization Theory and Applications*, vol. 147, no. 3, pp. 516–545, Jul. 2010.

[12] D. Jakovetic, J. Xavier, and J. M. F. Moura, "Fast Distributed Gradient Methods," *IEEE Trans. Autom. Control*, vol. 59, no. 5, pp. 1131–1146, May 2014.

[13] J. F. C. Mota, J. M. F. Xavier, P. M. Q. Aguiar, and M. Puschel, "D-ADMM: A Communication-Efficient Distributed Algorithm for Separable Optimization," *IEEE Trans. Signal Process.*, vol. 61, no. 10, pp. 2718–2723, May 2013.

[14] Q. Ling, Y. Xu, W. Yin, and Z. Wen, "Decentralized low-rank matrix completion," in *Proc' IEEE ICASSP*, no. 2, Mar. 2012, pp. 2925–2928.

[15] T.-H. Chang, M. Hong, and X. Wang, "Multi-agent distributed optimization via inexact consensus ADMM," *IEEE Trans. Signal Process.*, vol. 63, no. 2, pp. 482–497, Jan. 2015.

[16] T.-H. Chang, A. Nedic, and A. Scaglione, "Distributed Constrained Optimization by Consensus-Based Primal-Dual Perturbation Method," *IEEE Trans. Autom. Control*, vol. 59, no. 6, pp. 1524–1538, Jun. 2014.

[17] X. Li and A. Scaglione, "Convergence and Applications of a Gossip-Based Gauss-Newton Algorithm," *IEEE Trans. Signal Process.*, vol. 61, no. 21, pp. 5231–5246, Nov. 2013.

[18] P. Bianchi and J. Jakubowicz, "Convergence of a Multi-Agent Projected Stochastic Gradient Algorithm for Non-Convex Optimization," *IEEE Trans. Autom. Control*, vol. 58, no. 2, pp. 391–405, Feb. 2013.

[19] P. Chainais and C. Richard, "Distributed dictionary learning over a sensor network," pp. 1–6, Apr. 2013. [Online]. Available: http://arxiv.org/abs/1304.3568

[20] J. Gorski, F. P., and K. Klamroth, "Biconvex sets and optimization with biconvex functions: a survey and extensions," *Mathematical Methods of Operations Research*, vol. 66, no. 3, pp. 373–407, Jun. 2007.

[21] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization." *Nature*, vol. 401, no. 6755, pp. 788–791, October 1999.

[22] Y. Xu and W. Yin, "A Block Coordinate Descent Method for Regularized Multiconvex Optimization with Applications to Nonnegative Tensor Factorization and Completion," *SIAM Journal on Imaging Sciences*, vol. 6, no. 3, pp. 1758–1789, Sep. 2013.

[23] A. Beck and M. Teboulle, "Gradient-based algorithms with applications to signal recovery," *Convex Optimization in Signal Processing*, pp. 3–51, 2009.

[24] S.-Y. Tu and A. H. Sayed, "Diffusion Strategies Outperform Consensus Strategies for Distributed Estimation Over Adaptive Networks," *IEEE Trans. Signal Process.*, vol. 60, no. 12, pp. 6217–6234, Dec. 2012.

[25] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, Sep. 1999.

[26] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Systems & Control Letters*, vol. 53, no. 1, pp. 65–78, Sep. 2004.

[27] K. Engan, S. O. Aase, and J. H. Husøy, "Multi-frame compression: theory and design," *Signal Processing*, vol. 80, no. 10, pp. 2121–2140, Oct. 2000.