# Prox-PDA: The Proximal Primal-Dual Algorithm for Fast Distributed Nonconvex Optimization and Learning Over Networks

**Mingyi Hong** [1] **Davood Hajinezhad** [1] **Ming-Min Zhao** [2]

## Abstract

In this paper we consider nonconvex optimization and learning over a network of distributed nodes. We develop a Proximal Primal-Dual Algorithm (Prox-PDA), which enables the network nodes to distributedly and collectively compute the set of first-order stationary solutions in a global sublinear manner [with a rate of $\mathcal{O}(1/r)$, where $r$ is the iteration counter]. To the best of our knowledge, this is the first algorithm that enables distributed nonconvex optimization with global sublinear rate guarantees. Our numerical experiments also demonstrate the effectiveness of the proposed algorithm.

## 1. Introduction

We consider the following optimization problem

$$\min_{z\in\mathbb{R}^M} \; g(z) := \sum_{i=1}^{N} f_i(z), \tag{1}$$

where each $f_i$, $i \in \{1, \cdots, N\} := [N]$ is a nonconvex cost function, and we assume that it is smooth and has Lipschitz continuous gradient.

Such a *finite sum* problem plays a central role in machine learning and signal/information processing (Cevher et al., 2014; Hong et al., 2016). In particular, in the class of empirical risk minimization (ERM) problem, $z$ represents the feature vectors to be learned, and each $f_i$ can represent: 1) a mini-batch of (possibly nonconvex) loss functions modeling data fidelity (Antoniadis et al., 2009); 2) nonconvex activation functions of neural networks (Allen-Zhu & Hazan, 2016); 3) nonconvex utility functions used in applications such as resource allocation (Bjornson & Jorswieck, 2013). Recently, a number of works in machine learning community have been focused on designing fast algorithms for solving problem (1) in centralized setting; e.g., SAG (Defazio et al., 2014), SAGA (Schmidt et al., 2013), and SVRG (Johnson & Zhang, 2013) for convex problems, and (Reddi et al., 2016; Allen-Zhu & Hazan, 2016; Hajinezhad et al., 2016b; Rahimpour et al., 2016) for nonconvex problems.

In this work, we are interested in designing algorithms that solve problem (1) in a distributed manner. In particular, we focus on the scenario where each $f_i$ (or equivalently, each subset of data points in the ERM problem) is available locally at a given computing node $i \in [N]$, and the nodes are connected via a network. Clearly, such distributed optimization and learning scenario is important for machine learning, because in contemporary applications such as document topic modeling and/or social network data analysis, oftentimes data corporas are stored in geographically distributed locations without any central controller managing the entire network of nodes; see (Forero et al., 2010; Yan et al., 2013; Rahmani & Atia, 2015; Aybat & Hamedani, 2016).

**Related Works.** Distributed *convex* optimization and learning has been thoroughly investigated in the literature. In (Nedic & Ozdaglar, 2009b), the authors propose a distributed subgradient algorithm (DSG), which allows the agents to jointly optimize problem (1). Subsequently, many variants of DSG have been proposed, either with special assumptions on the underlying graph, or having additional structures of the problem; see, e.g., (Lobel & Ozdaglar, 2011; Lobel et al., 2011; Nedic & Olshevsky, 2015). The rate of convergence for DSG is $\mathcal{O}(\log(r)/\sqrt{r})$ under certain diminishing stepsize rules. Recently, a number of algorithms such as the exact first-order algorithm (EXTRA) (Shi et al., 2014) and DLM (Ling et al., 2015) have been proposed, which use constant stepsize and achieve faster $\mathcal{O}(1/r)$ rate for convex problems. Recent works that apply distributed optimization algorithms to machine learning applications include (Scardapane et al., 2016; Aybat & Hamedani, 2016; Scardapane & Lorenzo, 2016).

On the other hand, there has been little work for dis-

[1]Department of Industrial and Manufacturing Systems Engineering, Iowa State University, Ames, IA, USA [2]College of Information Science and Electronic Engineering, Zhejiang University, China. Correspondence to: Mingyi Hong <mingyi@iastate.edu>, Davood Hajinezhad <dhaji@iastate.edu>, Ming-Min Zhao <zmmblack@zju.edu.cn>.

*Proceedings of the $34^{th}$ International Conference on Machine Learning*, Sydney, Australia, PMLR 70, 2017. 

tributed optimization and learning when the objective function involves nonconvex problems. A dual subgradient method has been proposed in (Zhu & Martinez, 2010), which relaxes the exact consensus constraint. In (Bianchi & Jakubowicz, 2013) a stochastic projection algorithm using diminishing stepsizes has been proposed. An ADMM based algorithm has been presented in (Hong et al., 2014; Hajinezhad & Hong, 2015) for a special type of problem called *global consensus*, where all distributed nodes are directly connected to a central controller. Utilizing certain convexification decomposition technique the authors of (Lorenzo & Scutari, 2016) designed an algorithm named NEXT, which converges to the set of stationary solutions when using diminishing stepsizes. To the best of our knowledge, no multi agent distributed algorithm is able to guarantee global sublinear convergence rate for problem (1).

**Our Contributions**. In this work, we propose a proximal primal-dual algorithm (Prox-PDA) for problem (1) over an undirected connected network. We show that Prox-PDA converges to the set of stationary solutions of problem (1) (satisfying the first-order optimality condition) in a globally sublinear manner. We also show that Prox-PDA can be extended in several directions to improve its practical performance. To the best of our knowledge, this is the first algorithm that is capable of achieving global sublinear convergence rate for distributed non-convex optimization.

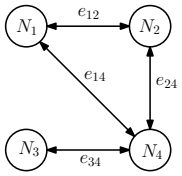
Further, our work reveals an interesting connection between the *primal-dual* based algorithm Prox-PDA and the *primal-only* fast distributed algorithms such as EXTRA (Shi et al., 2014). Such new insight of the connection between primal-dual and primal-only algorithms could be of independent interest for the optimization community. Finally, we generalize the theory for Prox-PDA based algorithms to a challenging distributed matrix factorization problem.

## 2. System Model

Define a graph $\mathcal{G} := \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$ and $\mathcal{E}$ are the node and edge sets; Let $|\mathcal{V}| = N$ and $|\mathcal{E}| = E$. Each node $v \in \mathcal{V}$ represents an agent in the network, and each edge $e_{ij} = (i,j) \in \mathcal{E}$ indicates that node $i$ and $j$ are neighbors; see Fig.1(Left). Assume that each node $i$ can only communicate with its immediate neighbors, defined as $\mathcal{N}_i := \{j \mid (i,j) \in \mathcal{V}\}$, with $|\mathcal{N}_i| = d_i$. The distributed version of problem (1) is given as below

$$\min_{x_i \in \mathbb{R}^M} f(x) := \sum_{i=1}^{N} f_i(x_i), \text{ s.t. } x_i = x_j, \ \forall\, (i,j) \in \mathcal{E}. \quad (2)$$

Clearly the above problem is equivalent to (1) as long as $\mathcal{G}$ is connected. For notational simplicity, define $x := \{x_i\} \in \mathbb{R}^{NM \times 1}$, and $Q := N \times M$.




$$\tilde{A} = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \end{bmatrix}.$$

*Figure 1.* (Left) An undirected Connected Network, (Right) Incidence Matrix.

To proceed, let us introduce a few useful quantities related to graph $\mathcal{G}$.

• The *incidence matrix* $\tilde{A} \in \mathbb{R}^{E \times N}$ is a matrix with entires $\tilde{A}(k,i) = 1$ and $\tilde{A}(k,j) = -1$ if $k = (i,j) \in \mathcal{E}$ with $j > i$, and all the rest of the entries being zero. For example, for the network in Fig.1 (Left); the incidence matrix is given in Fig.1 (Right). Define the *extended incidence matrix* as

$$A := \tilde{A} \otimes I_M \in \mathbb{R}^{EM \times Q}, \quad (3)$$

where $\otimes$ denotes the Kronecker product.

• The *Degree matrix* $\tilde{D} \in \mathbb{R}^{N \times N}$ is given by $\tilde{D} := \text{diag}[d_1, \cdots, d_N]$; Let $D := \tilde{D} \otimes I_M \in \mathbb{R}^{Q \times Q}$.

• The signed and the signless Laplacian matrices (denoted as $L^-$ and $L^+$ respectively), are given below

$$L^- := A^\top A \in \mathbb{R}^{Q \times Q}, \ L^+ := 2D - A^\top A \in \mathbb{R}^{Q \times Q}. \quad (4)$$

Using the above notations, one can verify that problem (2) can be written in the following compact form

$$\min_{x \in \mathbb{R}^Q} f(x), \quad \text{s.t. } Ax = 0. \quad (5)$$

## 3. The Prox-PDA Algorithm

The proposed algorithm builds upon the classical augmented Lagrangian (AL) method (Bertsekas, 1982; Powell, 1969). Let us define the AL function for (5) as

$$L_\beta(x, \mu) = f(x) + \langle \mu, Ax \rangle + \frac{\beta}{2}\|Ax\|^2, \quad (6)$$

where $\mu \in \mathbb{R}^Q$ is the dual variable; $\beta > 0$ is a penalty parameter. Let $B \in \mathbb{R}^{Q \times Q}$ be some arbitrary matrix to be determined shortly. Then the proposed algorithm is given in the table below (Algorithm 1).

In Prox-PDA, the primal iteration (7a) minimizes the augmented Lagrangian plus a proximal term $\frac{\beta}{2}\|x - x^r\|^2_{B^TB}$. We emphasize that the proximal term is critical in both the algorithm implementation and the analysis. It is used to ensure the following key properties:

(1). The primal problem is strongly convex;

(2). The primal problem is decomposable over different network nodes, hence distributedly implementable.

To see the first point, suppose $B^TB$ is chosen such that $A^TA + B^TB \succeq I_Q$, and that $f(x)$ has Lipschitz gradient. Then by a result in (Zlobec, 2005)[Theorem 2.1], we know

**Algorithm 1** The Prox-PDA Algorithm

1: At iteration 0, initialize $\mu^0 = 0$ and $x^0 \in \mathbb{R}^Q$.
2: At each iteration $r+1$, update variables by:

$$x^{r+1} = \arg\min_{x\in\mathbb{R}^Q} \; f(x) + \langle \mu^r, Ax\rangle + \frac{\beta}{2}\|Ax\|^2 + \frac{\beta}{2}\|x - x^r\|^2_{B^TB}; \tag{7a}$$

$$\mu^{r+1} = \mu^r + \beta A x^{r+1}. \tag{7b}$$

that there exists $\beta > 0$ large enough such that the objective function of (7a) is strongly convex. To see the second point, Let $B := |A|$, where the absolute value is taken for each component of $A$. It can be verified that $B^TB = L^+$, and step (7a) becomes

$$\begin{aligned} x^{r+1} &= \arg\min_x \sum_{i=1}^N f_i(x_i) + \langle \mu^r, Ax\rangle \\ &+ \frac{\beta}{2}x^T L^- x + \frac{\beta}{2}(x - x^r)^T L^+ (x - x^r) \\ &= \arg\min_x \sum_{i=1}^N f_i(x_i) + \langle \mu^r, Ax\rangle + \beta x^T D x - \beta x^T L^+ x^r. \end{aligned}$$

Clearly this problem is *separable* over the nodes, therefore it can be solved completely distributedly.

## 4. The Convergence Analysis

In this section we provide convergence analysis for Algorithm 1. The key in the analysis is the construction of a novel potential function, which decreases at every iteration of the algorithm. In particular, the constructed potential function is a *conic combination* of the AL function and the size of the violation of the consensus constraint, therefore it measures the progress of *both* the primal and dual updates.

We first state our main assumptions below.

**[A1.]** The function $f(x)$ is differentiable and has Lipschitz continuous gradient, i.e.,

$$\|\nabla f(x) - \nabla f(x)\| \le L\|x - y\|, \quad \forall\, x, y \in \mathbb{R}^Q.$$

Further assume that $A^TA + B^TB \succeq I_Q$.

**[A2.]** There exists a constant $\delta > 0$ such that

$$\exists\, \underline{f} > -\infty, \quad \text{s.t. } f(x) + \frac{\delta}{2}\|Ax\|^2 \ge \underline{f},\ \forall\, x \in \mathbb{R}^Q.$$

Without loss of generality we will assume that $\underline{f} = 0$. Below we provide a few nonconvex smooth functions that satisfy our assumptions, all of which are commonly used as activation functions for neural networks.

- **The sigmoid function** $\texttt{sigmoid}(x) = \frac{1}{1+e^{-x}}$.
- **The** arctan **and** tanh **function**.
- **The** logit **function** $\text{logit}(x) = \frac{e^x}{e^x+1}$.

### 4.1. The Analysis Steps

Below we provide the analysis of Prox-PDA. First we provide a bound on the size of the constraint violation using a quantity related to the primal iterates. Let $\sigma_{\min}$ denotes the smallest *non-zero* eigenvalue of $A^TA$, and we define $w^r := (x^{r+1} - x^r) - (x^r - x^{r-1})$ for notational simplicity. We have the following result.

**Lemma 1** *Suppose Assumptions [A1] and [A2] are satisfied. Then the following is true for Prox-PDA*

$$\frac{1}{\beta}\|\mu^{r+1} - \mu^r\|^2 \le \frac{2L^2}{\beta\sigma_{\min}}\left\|x^r - x^{r+1}\right\|^2 + \frac{2\beta}{\sigma_{\min}}\|B^TBw^r\|^2. \tag{8}$$

Then we bound the descent of the AL function.

**Lemma 2** *Suppose Assumptions [A1] and [A2] are satisfied. Then the following is true for Algorithm 1*

$$L_\beta(x^{r+1}, \mu^{r+1}) - L_\beta(x^r, \mu^r) \le \frac{2\beta\|B^TB\|}{\sigma_{\min}}\|w^r\|^2_{B^TB} - \left(\frac{\beta - L}{2} - \frac{2L^2}{\beta\sigma_{\min}}\right)\|x^{r+1} - x^r\|^2. \tag{9}$$

A key observation from Lemma 2 is that no matter how large $\beta$ is, the rhs of (9) cannot be made negative. This observation suggests that in contrast to (Hong et al., 2014; Hajinezhad et al., 2016a) the augmented Lagrangian alone cannot serve as the potential function for Prox-PDA. In search for an appropriate potential function, we need a new object that is decreasing in the order of $\beta\|w^r\|^2_{B^TB}$. The following lemma shows that the descent of the sum of the constraint violation and the proximal term has the desired property.

**Lemma 3** *Suppose Assumption [A1] is satisfied. Then the following is true*

$$\begin{aligned} &\frac{\beta}{2}\left(\|Ax^{r+1}\|^2 + \|x^{r+1} - x^r\|^2_{B^TB}\right) \\ &\le L\|x^{r+1} - x^r\|^2 + \frac{\beta}{2}\left(\|Ax^r\|^2 + \|x^r - x^{r-1}\|^2_{B^TB}\right) \\ &- \frac{\beta}{2}\left(\|w^r\|^2_{B^TB} + \|A(x^{r+1} - x^r)\|^2\right). \end{aligned} \tag{10}$$

It is interesting to observe that the new object, $\beta/2\left(\|Ax^{r+1}\|^2 + \|x^{r+1} - x^r\|^2_{B^TB}\right)$, *increases* in $L\|x^{r+1} - x^r\|^2$ and *decreases* in $\beta/2\|w^r\|^2_{B^TB}$, while the AL behaves in an opposite manner (cf. Lemma 2). More importantly, in our new object, the constant in front of $\|x^{r+1} - x^r\|^2$ is *independent* of $\beta$. Although neither of these two objects decreases by itself, quite surprisingly, a proper *conic combination* of these two objects decreases at every iteration of Prox-PDA. To precisely state the claim, let us define the *potential function* for Algorithm 1 as

$$\begin{aligned} P_{c,\beta}(x^{r+1}, x^r, \mu^{r+1}) &:= L_\beta(x^{r+1}, \mu^{r+1}) \\ &+ \frac{c\beta}{2}\left(\|Ax^{r+1}\|^2 + \|x^{r+1} - x^r\|^2_{B^TB}\right), \end{aligned} \tag{11}$$

where $c > 0$ is some constant to be determined later. We have the following result.

**Lemma 4** *Suppose the assumptions made in Lemmas 1 – 3 are satisfied. Then we have the following*

$$P_{c,\beta}(x^{r+1}, x^r, \mu^{r+1}) \le P_{c,\beta}(x^r, x^{r-1}, \mu^r) - \left(\frac{\beta - L}{2} - \frac{2L^2}{\beta\sigma_{\min}} - cL\right)\|x^{r+1} - x^r\|^2 - \left(\frac{c\beta}{2} - \frac{2\beta\|B^TB\|_F}{\sigma_{\min}}\right)\|w^r\|^2_{B^TB}. \quad (12)$$

From the above analysis, it is easy to see that as long as $c$ and $\beta$ are sufficiently large, the potential function decreases at each iteration of Prox-PDA. Below we derive the precise bounds for $c$ and $\beta$. First, a sufficient condition for $c$ is given below (note, that $\delta > 0$ is defined in Assumption [A2])

$$c \ge \max\left\{\frac{\delta}{L}, \frac{4\|B^TB\|_F}{\sigma_{\min}}\right\}. \quad (13)$$

Second, for any given $c$, we need $\beta$ to satisfy $\frac{\beta - L}{2} - \frac{2L^2}{\beta\sigma_{\min}} - cL > 0$, which implies the following

$$\beta > \frac{L}{2}\left(2c + 1 + \sqrt{(2c+1)^2 + \frac{16L^2}{\sigma_{\min}}}\right). \quad (14)$$

We conclude that if both (13) and (14) are satisfied, then the potential function $P_{c,\beta}(x^{r+1}, x^r, \mu^{r+1})$ decreases at every iteration.

Our next step shows that by using the particular choices of $c$ and $\beta$ in (13) and (14), the constructed potential function is lower bounded.

**Lemma 5** *Suppose [A1] - [A2] are satisfied, and $(c, \beta)$ are chosen according to (13) and (14). Then the following statement holds true*

$$\exists\ \underline{P} > -\infty \text{ s.t. } P_{c,\beta}(x^{r+1}, x^r, \mu^{r+1}) \ge \underline{P},\ \forall\ r > 0.$$

Now we are ready to present the main result of this section. To this end, define $Q(x^{r+1}, \mu^r)$ as the optimality gap of problem (5), given by

$$Q(x^{r+1}, \mu^r) := \|\nabla_x L_\beta(x^{r+1}, \mu^r)\|^2 + \|Ax^{r+1}\|^2. \quad (15)$$

It is easy to see that $Q(x^{r+1}, \mu^r) \to 0$ implies that any limit point $(x^*, \mu^*)$, if it exists, is a KKT point of (5) that satisfies the following conditions

$$0 = \nabla f(x^*) + A^T\mu^*, \quad Ax^* = 0. \quad (16)$$

In the following we show that the gap $Q(\cdot)$ not only decreases to zero, but does so in a sublinear manner.

**Theorem 1** *Suppose Assumption A and the conditions (13) and (14) are satisfied. Then we have*

- **Eventual Consensus:**

$$\lim_{r\to\infty} \mu^{r+1} - \mu^r \to 0,\ \lim_{r\to\infty} Ax^r \to 0.$$

- **Convergence to Stationary Points:** *Every limit point of the iterates $\{x^r, \mu^r\}$ generated by Algorithm 1 converges to a KKT point of problem (5). Further, $Q(x^{r+1}, \mu^r) \to 0$.*
- **Sublinear Convergence Rate:** *For any given $\varphi > 0$, let us define $T$ to be the first time that the optimality gap reaches below $\varphi$, i.e.,*

$$T := \arg\min_r Q(x^{r+1}, \mu^r) \le \varphi.$$

*Then for some $\nu > 0$, we have $\varphi \le \frac{\nu}{T-1}$. That is, the optimality gap $Q(x^{r+1}, \mu^r)$ converges sublinearly.*

## 5. Variants of Prox-PDA

In this section, we discuss two important extensions of the Prox-PDA, one allows the $x$-problem (7a) to be solved inexactly, while the second allows the use of increasing penalty parameter $\beta$. In many practical applications, exactly minimizing the augmented Lagrangian may not be easy. Therefore, we propose the proximal gradient primal-dual algorithm (Prox-GPDA), whose main steps are given below

$$x^{r+1} = \arg\min_{x\in\mathbb{R}^Q} \langle \nabla f(x^r), x - x^r\rangle + \langle \mu^r, Ax\rangle + \frac{\beta}{2}\|Ax\|^2 + \frac{\beta}{2}\|x - x^r\|^2_{B^TB}; \quad (17)$$

$$\mu^{r+1} = \mu^r + \beta Ax^{r+1}. \quad (18)$$

The analysis of this algorithm follows similar steps as that for Prox-PDA. For detailed discussion see (Hong, 2016).

Our second variant do not require to explicitly compute the bound for $\beta$ given in (14). Indeed, the bounds on $\beta$ derived in the previous sections are the worst case bounds, and algorithms that use stepsizes that strictly satisfy such bounds may be slow at the beginning. In practice, one may prefer to start with a small penalty parameter and gradually increase it. We denote this algorithm by Prox-PDA-IP, whose main steps are given below

$$x^{r+1} = \arg\min_{x\in\mathbb{R}^Q} f(x) + \langle \mu^r, Ax\rangle + \frac{\beta^{r+1}}{2}\|Ax\|^2 + \frac{\beta^{r+1}}{2}\|x - x^r\|^2_{B^TB}; \quad (19)$$

$$\mu^{r+1} = \mu^r + \beta^{r+1}Ax^{r+1}. \quad (20)$$

Note that one can also replace $f(x)$ in (19) by $\langle \nabla f(x^r), x - x^r\rangle$ to obtain a similar variant for Prox-GPDA denoted by Prox-GPDA-IP. Throughout this section we will still as-

sume that Assumption A holds true. Further, we will assume that $\beta^r$ satisfies the following conditions

$$\frac{1}{\beta^r} \to 0,\ \sum_{r=1}^{\infty} \frac{1}{\beta^r} = \infty,\ \beta^{r+1} \geq \beta^r,$$
$$\max_r\,(\beta^{r+1} - \beta^r) < \kappa,\ \text{for some finite}\ \ \kappa > 0. \qquad (21)$$

Also without loss of generality we will assume that

$$B^T B \succ 0, \quad \text{and} \quad \|B^T B\|_F > 1. \qquad (22)$$

Note that this is always possible, by adding an identity matrix to $B^T B$ if necessary.

The analysis for Prox-PDA-IP is long and technical, therefore we refer the readers to (Hong, 2016). The key step is to construct a new potential function, given below

$$P_{\beta^{r+1},c}(x^{r+1}, x^r, \mu^{r+1}) = L_{\beta^{r+1}}(x^{r+1}, \mu^{r+1}) + \frac{c\beta^{r+1}\beta^r}{2}\|Ax^{r+1}\|^2 + \frac{c\beta^{r+1}\beta^r}{2}\|x^r - x^{r+1}\|^2_{B^T B}.$$

The insight here is that in order to achieve the desired descent, in the potential function the coefficients for $\|x^{r+1} - x^r\|^2_{B^T B}$ and $\|Ax^{r+1}\|^2$ should be proportional to $\mathcal{O}\left((\beta^r)^2\right)$. We have the following theorem regarding to the convergence of Prox-PDA-IP.

**Theorem 2** *Suppose Assumption A and (21) are satisfied. Suppose that $B$ is selected such that (22) holds true. Then the following hold for Prox-PDA-IP*

• **Eventual Consensus:**

$$\lim_{r\to\infty} \mu^{r+1} - \mu^r \to 0,\ \lim_{r\to\infty} Ax^r \to 0.$$

• **Convergence to KKT Points:** *Every limit point of the iterates $\{x^r, \mu^r\}$ generated by Prox-PDA-IP converges to a KKT point of problem (5). Further, $Q(x^{r+1}, \mu^r) \to 0$.*

## 6. Connections and Discussions

In this section we present an interesting observation which establishes links between the so-called EXTRA algorithm (Shi et al., 2014) (developed for distributed, but *convex* optimization) and the Prox-GPDA.

Specifically, the optimality condition of the $x$-update step (17) is given by

$$\nabla f(x^r) + A^T(\mu^r + \beta Ax^{r+1}) + \beta(B^T B(x^{r+1} - x^r)) = 0.$$

Utilizing the fact that $A^T A = L^-$, $B^T B = L^+$ and $L^+ + L^- = 2D$, we have

$$\nabla f(x^r) + A^T\mu^r + 2\beta D x^{r+1} - \beta L^+ x^r = 0.$$

Subtracting the same equation evaluated at the previous iteration, we obtain

$$\nabla f(x^r) - \nabla f(x^{r-1}) + \beta L^- x^r + 2\beta D(x^{r+1} - x^r) - \beta L^+(x^r - x^{r-1}) = 0,$$

where we have used the fact that $A^T(\mu^r - \mu^{r-1}) = \beta A^T A x^r = \beta L^- x^r$. Rearranging terms, we have

$$\begin{aligned} x^{r+1} &= x^r - \frac{1}{2\beta} D^{-1}\left(\nabla f(x^r) - \nabla f(x^{r-1})\right) \\ &+ \frac{1}{2} D^{-1}(L^+ - L^-)x^r - \frac{1}{2} D^{-1} L^+ x^{r-1} \\ &= x^r - \frac{1}{2\beta} D^{-1}\left(\nabla f(x^r) - \nabla f(x^{r-1})\right) + Wx^r \\ &- \frac{1}{2}(I + W)x^{r-1}, \end{aligned} \qquad (23)$$

where in the last equality we have defined the *weight matrix* $W := \frac{1}{2} D^{-1}(L^+ - L^-)$, which is a row stochastic matrix.

Iteration (23) has the same form as the EXTRA algorithm given in (Shi et al., 2014), therefore we can conclude that EXTRA is a special case of Prox-GPDA. Moreover, by appealing to our analysis in Section 5, it readily follows that EXTRA works for the nonconvex distributed optimization problem as well.

We remark that each node $i$ can distributedly implement iteration (23) by performing the following

$$\begin{aligned} x_i^{r+1} &= x_i^r - \frac{1}{2\beta d_i}\left(\nabla f_i(x_i^r) - \nabla f_i(x_i^{r-1})\right) \\ &+ \sum_{j\in\mathcal{N}(i)} \frac{1}{d_i} x_j^r - \frac{1}{2}\left(\sum_{j\in\mathcal{N}(i)} \frac{1}{d_i} x_j^{r-1} + x_i^{r-1}\right). \end{aligned} \qquad (24)$$

Clearly, at iteration $r+1$, besides the local gradient information, node $i$ only needs the aggregated information from its neighbors, $\sum_{j\in\mathcal{N}(i)} x_j^r$. Therefore the algorithm is distributedly implementable.

## 7. Distributed Matrix Factorization

In this section we study a variant of the Prox-PDA/Prox-PDA-IP for the following distributed matrix factorization problem (Ling et al., 2012)

$$\begin{aligned} \min_{X,Y} \quad & \frac{1}{2}\|XY - Z\|_F^2 + \eta\|X\|_F^2 + h(Y) \qquad (25) \\ & = \sum_{i=1}^{N} \frac{1}{2}\|Xy_i - z_i\|^2 + \gamma\|X\|_F^2 + h_i(y_i), \\ \text{s.t.} \quad & \|y_i\|^2 \leq \tau,\ \forall\, i \end{aligned}$$

where $X \in \mathbb{R}^{M\times K}$, $Y \in \mathbb{R}^{K\times N}$; for each $i$, $y_i \in \mathbb{R}^K$ consists of one column of $Y$; $Z \in \mathbb{R}^{M\times N}$ is some known matrix; $h(Y) := \sum_{i=1}^N h_i(y_i)$ is some convex but possibly nonsmooth penalization term; $\eta > 0$ is some given constant; for notation simplicity we have defined $\gamma := 1/N\eta$.

It is easy to extend the above formulation to the case where $Y$ and $Z$ both have $NP$ columns, and each $y_i$ and $z_i$ consists of $P$ columns of $Y$ and $Z$ respectively.

We assume that $h(Y)$ is lower bounded over dom $(h)$. One application of problem (25) is the distributed *sparse dictionary learning* problem where $X$ is the dictionary to be learned, each $z_i$ is a training data sample, and each $y_i$ is the sparse coefficient corresponding to the particular training sample $z_i$. The constraint $\|y_i\|^2 \leq \tau$ simply says that the size of the coefficient must be bounded.

Consider a distributed scenario where $N$ agents form a graph $\{\mathcal{V}, \mathcal{E}\}$, each having a column of $Y$. We reformulate problem (25) as

$$\min_{\{X_i\},\{y_i\}} \sum_{i=1}^{N} \left( \frac{1}{2}\|X_i y_i - z_i\|^2 + h_i(y_i) + \gamma \|X_i\|_F^2 \right)$$
$$\text{s.t. } \|y_i\|^2 \leq \tau, \ \forall\, i \quad X_i = X_j, \ \forall\, (i,j) \in \mathcal{E}.$$

Let us stack all the variables $X_i$, and define $\boldsymbol{X} := [X_1; X_2; \cdots; X_N] \in \mathbb{R}^{NM \times K}$. Define the block signed incidence matrix as $\boldsymbol{A} = \tilde{A} \otimes I_M \in \mathbb{R}^{EM \times NM}$, where $A$ is the standard graph incidence matrix. Define the block signless incidence matrix $\boldsymbol{B} \in \mathbb{R}^{EM \times NM}$ similarly. If the graph is connected, then the condition $\boldsymbol{AX} = \boldsymbol{0}$ implies network-wide consensus. We formulate the distributed matrix factorization problem as

$$\begin{aligned} &\min_{\{X_i\},\{y_i\}} \ f(\boldsymbol{X}, Y) + h(Y) \\ &:= \sum_{i=1}^{N} \left( \frac{1}{2}\|X_i y_i - z_i\|^2 + \gamma \|X_i\|_F^2 + h_i(y_i) \right) \\ &\text{s.t. } \|y_i\|^2 \leq \tau, \ \forall\, i \quad \boldsymbol{AX} = \boldsymbol{0}. \end{aligned} \tag{26}$$

Clearly the above problem does not satisfy Assumption A, because the objective function is not smooth, and neither $\nabla_{\mathbf{X}} f(\mathbf{X}, Y)$ nor $\nabla_Y f(\mathbf{X}, Y)$ is Lipschitz continuous. The latter fact poses significant difficulty in algorithm development and analysis. Define the block-signed/signless Laplacians as

$$\boldsymbol{L}^- = \boldsymbol{A}^T \boldsymbol{A}, \ \boldsymbol{L}^+ = \boldsymbol{B}^T \boldsymbol{B}. \tag{27}$$

The AL function for the above problem is given by

$$\begin{aligned} L_\beta(\boldsymbol{X}, Y, \boldsymbol{\Omega}) &= \sum_{i=1}^{N} \left( \frac{1}{2}\|X_i y_i - z_i\|^2 + \gamma \|X_i\|_F^2 + h_i(y_i) \right) \\ &+ \langle \boldsymbol{\Omega}, \boldsymbol{AX} \rangle + \frac{\beta}{2} \langle \boldsymbol{AX}, \boldsymbol{AX} \rangle, \end{aligned} \tag{28}$$

where $\boldsymbol{\Omega} := \{\Omega_e\} \in \mathbb{R}^{EM \times K}$ is the matrix of the dual variable, with $\Omega_e \in \mathbb{R}^{M \times K}$ being the dual variable for the consensus constraint on link $e$, i.e, $X_i = X_j$, $e = (i,j)$.

Let us generalize Algorithm 1 for distributed matrix factorization given in Algorithm 2. In Algorithm 2 we have introduced a sequence $\{\theta_i^r \geq 0\}$ which measures the size of the local factorization error. We note that including the proximal term $\frac{\theta_i^r}{2}\|y_i - y_i^r\|^2$ is the key to achieve convergence for Algorithm 2.

**Algorithm 2** Prox-PDA for Distr. Matrix Factorization

1: At iteration 0, initialize $\boldsymbol{\Omega}^0 = \boldsymbol{0}$, and $\boldsymbol{X}^0, y^0$
2: At each iteration $r+1$, update variables by:

$$\theta_i^r = \|X_i^r y_i^r - z_i\|^2, \quad \forall\, i; \tag{29a}$$

$$y_i^{r+1} = \arg \min_{\|y_i\|^2 \leq \tau} \frac{1}{2}\|X_i^r y_i - z_i\|^2 + h_i(y_i) + \frac{\theta_i^r}{2}\|y_i - y_i^r\|^2, \quad \forall\, i; \tag{29b}$$

$$\boldsymbol{X}^{r+1} = \arg\min_{\boldsymbol{X}} \ f(\boldsymbol{X}, Y^{r+1}) + \langle \boldsymbol{\Omega}^r, \boldsymbol{AX} \rangle + \frac{\beta}{2}\langle \boldsymbol{AX}, \boldsymbol{AX} \rangle + \frac{\beta}{2}\langle \boldsymbol{B}(\boldsymbol{X} - \boldsymbol{X}^r), \boldsymbol{B}(\boldsymbol{X} - \boldsymbol{X}^r) \rangle; \tag{29c}$$

$$\boldsymbol{\Omega}^{r+1} = \boldsymbol{\Omega}^r + \beta \boldsymbol{AX}^{r+1}. \tag{29d}$$

Let us comment on the distributed implementation of the algorithm. First note that the $y$ subproblem (29b) is naturally distributed to each node, that is, only local information is needed to perform the update. Second, the $\boldsymbol{X}$ subproblem (29c) can also be decomposed into $N$ subproblems, one for each node. To be more precise, let us examine the terms in (29c) one by one. First, the term $f(\boldsymbol{X}, Y^{r+1}) = \sum_{i=1}^N \left( \frac{1}{2}\|X_i y_i^{r+1} - z_i\|^2 + h_i(y_i^{r+1}) + \gamma \|X_i\|_F^2 \right)$, hence it is decomposable. Second, the term $\langle \boldsymbol{\Omega}^r, \boldsymbol{AX} \rangle$ can be expressed as

$$\langle \boldsymbol{\Omega}^r, \boldsymbol{AX} \rangle = \sum_{i=1}^{N} \sum_{e \in U(i)} \langle \Omega_e^r, X_i \rangle - \sum_{e \in H(i)} \langle \Omega_e^r, X_i \rangle$$

where the sets $U(i)$ and $H(i)$ are defined as

$$U(i) := \{e \mid e = (i,j) \in \mathcal{E}, i \geq j\},$$
$$H(i) := \{e \mid e = (i,j) \in \mathcal{E}, j \geq i\}.$$

Similarly, we have

$$\begin{aligned} \langle \boldsymbol{BX}^r, \boldsymbol{BX} \rangle &= \sum_{i=1}^{N} \left\langle X_i, d_i X_i^r + \sum_{j \in N(i)} X_j^r \right\rangle \\ \frac{\beta}{2} \left( \langle \boldsymbol{AX}, \boldsymbol{AX} \rangle + \langle \boldsymbol{BX}, \boldsymbol{BX} \rangle \right) &= \beta \langle \boldsymbol{DX}, \boldsymbol{X} \rangle = \beta \sum_{i=1}^{N} d_i \|X_i\|_F^2, \end{aligned}$$

where $\boldsymbol{D} := \tilde{D} \otimes I_M \in \mathbb{R}^{NM \times NM}$ with $\tilde{D}$ being the degree matrix. It is easy to see that the $\boldsymbol{X}$ subproblem (29c) is separable over the distributed agents. Finally, one can verify that the $\boldsymbol{\Omega}$ update step (29d) can be implemented by each edge $e \in \mathcal{E}$ as follows

$$\Omega_e^{r+1} = \Omega_e^r + \beta \left( X_i^{r+1} - X_j^{r+1} \right), \quad e = (i,j), i \geq j.$$

To show convergence rate of the algorithm, we need the following definition

$$Q(\boldsymbol{X}^{r+1}, Y^{r+1}, \boldsymbol{\Omega}^r) := \beta\|\boldsymbol{A}\boldsymbol{X}^{r+1}\|^2 + \|[\boldsymbol{Z}_1^{r+1}; \boldsymbol{Z}_2^{r+1}]\|^2,$$

where we have defined

$$\boldsymbol{Z}_1^{r+1} := \nabla_{\boldsymbol{X}} L_\beta(\boldsymbol{X}^{r+1}, Y^{r+1}, \boldsymbol{\Omega}^r);$$
$$\boldsymbol{Z}_2^{r+1} := Y^{r+1} - \text{prox}_{h+\iota(\mathcal{Y})}\left[Y^{r+1} - \nabla_Y\left(L_\beta(\boldsymbol{X}^{r+1}, Y^{r+1}, \boldsymbol{\Omega}^r) - h(Y)\right)\right].$$

In the above expression, we have used $\mathcal{Y} := \bigcup_i \left\{\|y_i\|^2 \le \tau\right\}$ to denote the feasible set of $Y$, and used $\iota(\mathcal{Y})$ to denote the indicator function of such set. Similarly as in Section 4, we can show that $Q(\boldsymbol{X}^{r+1}, Y^{r+1}, \boldsymbol{\Omega}^r) \to 0$ implies that every limit point of $(\boldsymbol{X}^{r+1}, Y^{r+1}, \boldsymbol{\Omega}^r)$ is a KKT point of problem (26).

Next we present the main convergence analysis for Algorithm 2. The proof is long and technical, therefore we refer the readers to (Hong, 2016).

**Theorem 3** *Consider using Algorithm 2 to solve the distributed matrix factorization problem (26). Suppose that $h(Y)$ is lower bounded over dom $h(x)$, and that the penalty parameter $\beta$, together with two positive constants $c$ and $d$, satisfies the following conditions*

$$\begin{aligned} &\frac{\beta+2\gamma}{2} - \frac{8(\tau^2+4\gamma^2)}{\beta\sigma_{\min}} - \frac{cd}{2} > 0, \\ &\frac{1}{2} - \frac{8}{\sigma_{\min}\beta} - \frac{c}{d} > 0, \ \frac{1}{2} - \frac{8\tau}{\sigma_{\min}\beta} - \frac{c\tau}{d} > 0, \\ &\frac{c\beta}{2} - \frac{2\beta\|B^TB\|}{\sigma_{\min}} > 0. \end{aligned} \tag{30}$$

*Then in the limit, consensus will be achieved, i.e.,*

$$\lim_{r\to\infty} \|X_i^r - X_j^r\| = 0, \quad \forall\, (i,j) \in \mathcal{E}.$$

*Further, the sequences $\{\boldsymbol{X}^{r+1}\}$ and $\{\boldsymbol{\Omega}^{r+1}\}$ are both bounded, and every limit point generated by Algorithm 2 is a KKT point of problem (25).*

*Additionally, Algorithm 2 converges sublinearly. Specifically, for any given $\varphi > 0$, define $T$ to be the first time that the optimality gap reaches below $\varphi$, i.e.,*

$$T := \arg\min_r \ Q(\boldsymbol{X}^{r+1}, Y^{r+1}, \boldsymbol{\Omega}^r) \le \varphi.$$

*Then for some constant $\nu > 0$ we have $\varphi \le \frac{\nu}{T-1}$.*

We can see that it is always possible to find the tuple $\{\beta, c, d > 0\}$ that satisfies (30): $c$ can be solely determined by the last inequality; for fixed $c$, the constant $d$ needs to be chosen large enough such that $1/2 - \frac{c}{d} > 0$ and $1/2 - \frac{c\tau}{d} > 0$ are satisfied. After $c$ and $d$ are fixed, one can always choose $\beta$ large enough to satisfy the first three conditions. In practice, we typically prefer to choose $\beta$ as small as possible to improve the convergence speed. Therefore empirically one can start with (for some small $\nu > 0$): $c = \frac{4\|B^TB\|}{\sigma_{\min}} + \nu, \quad d = \max\{4, 2c\tau\}$, and then gradually increase $d$ to find an appropriate $\beta$ that satisfies the first three conditions.

We remark that Algorithm 2 can be extended to the case with increasing penalty. Due to the space limitation we omit the details here.

## 8. Numerical Results

In this section, we demonstrate the performance of the proposed algorithms. All experiments are performed in Matlab (2016b) on a desktop with an Intel Core(TM) i5-4690 CPU (3.50 GHz) and 8GB RAM running Windows 7.

### 8.1. Distributed Binary Classification

In this subsection, we study the problem of binary classification using nonconvex regularizers in the mini-bach setup i.e. each node stores $b$ (batch size) data points, and each component function is given by

$$f_i(x_i) = \frac{1}{Nb}\left[\sum_{j=1}^{b} \log(1 + \exp(-y_{ij}x_i^T v_{ij})) + \sum_{k=1}^{M} \frac{\lambda\alpha x_{i,k}^2}{1+\alpha x_{i,k}^2}\right]$$

where $v_{ij} \in \mathbb{R}^M$ and $y_{ij} \in \{1, -1\}$ are the feature vector and the label for the $j$th date point in $ith$ agent (Antoniadis et al., 2009). We use the parameter settings of $\lambda = 0.001$, $\alpha = 1$ and $M = 10$. We randomly generated $100,000$ data points and distribute them into $N = 20$ nodes (i.e. $b = 5000$). We use the optimality gap (opt-gap) and constraint violation (con-vio), displayed below, to measure the quality of the solution generated by different algorithms

$$\text{opt-gap} := \left\|\sum_{i=1}^{N} \nabla f_i(z_i)\right\|^2 + \|Ax\|^2, \ \text{con-vio} = \|Ax\|^2.$$

We compare the the Prox-GPDA, and Prox-GPDA-IP with the distributed subgradient (DSG) method (Nedic & Ozdaglar, 2009a;b) (which is only known to work for convex cases) and the Push-sum algorithm (Tatarenko & Touri, 2015). The performance of all three algorithms in terms of the consensus error and the optimality gap (averaged over 30 problem instances) are presented in Fig. 2. The penalty parameter for Prox-GPDA is chosen such that satisfy (14), and $\beta^r$ for Prox-GPDA-IP is set as $0.05\log(r)$, the stepsizes of the DSG algorithm and the Push-sum algorithm are chosen as $1/0.05\log(r)$ and $1/r$, respectively. Note that these parameters are tuned for each algorithm to achieve the best results. All Algorithms will stop after 1000 iterations. It can be observed that the Prox-GPDA with constant stepsize outperforms other algorithms. The Push-sum algorithm does not seem to converge within 1000 iterations.

To see more results, we compare different algorithms with different number of agents in the network ($N$). The problem and the algorithms setup are set as the previous case. We measure the optimality gap as well as the constraint violation and the results are respectively reported in Table 1 and Table 2. In the tables Alg1, Alg2, Alg3, Alg4

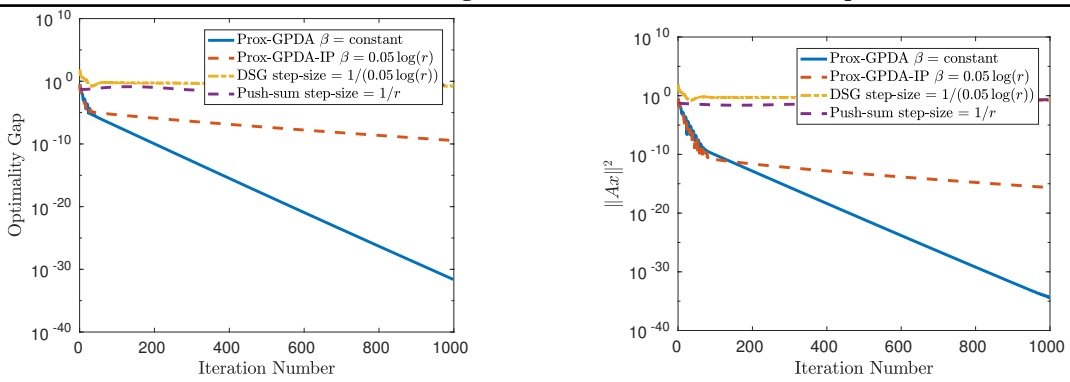



*Figure 2.* Results for the matrix factorization problem.

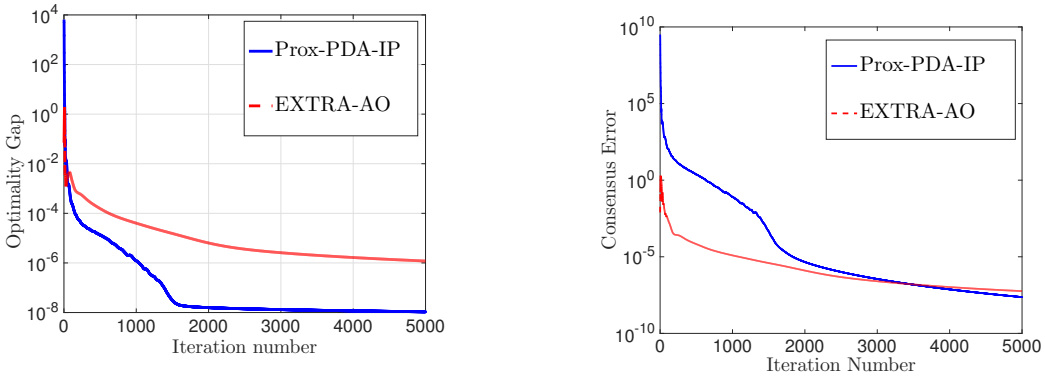



*Figure 3.* Results for the matrix factorization problem.

*Table 1.* Optimality Gap for different Algorithms

| N | **Alg1** | **Alg2** | **Alg3** | **Alg4** |
|---|---|---|---|---|
| **10** | 5.1e-36 | 2.4e-22 | 1.34 | 2.79 |
| **20** | 4.7e-32 | 5.0e-9 | 0.04 | 0.42 |
| **30** | 2.3e-21 | 5.1e-8 | 0.008 | 0.20 |
| **40** | 1.3e-12 | 2.9e-7 | 0.007 | 0.21 |
| **50** | 5.5e-10 | 4.2e-6 | 0.005 | 0.40 |

*Table 2.* Constraint Violation for different Algorithms

| N | **Alg1** | **Alg2** | **Alg3** | **Alg4** |
|---|---|---|---|---|
| **10** | 1.3e-36 | 3.4e-27 | 0.35 | 0.65 |
| **20** | 1.2e-34 | 3.7e-16 | 0.02 | 0.40 |
| **30** | 2.3e-24 | 7.8e-15 | 0.01 | 0.18 |
| **40** | 2.2e-16 | 2.1e-14 | 0.03 | 0.20 |
| **50** | 2.2e-14 | 2.2e-12 | 0.01 | 0.12 |

denote Prox-GPDA, Prox-GPDA-IP, DGS, and Push-sum algorithms respectively. As seen, the performance of the proposed algorithms (Alg1, Alg2) are significantly better than DGS and Push-Sum.

### 8.2. Distributed Matrix Factorization

In this section we consider the distributed matrix factorization problem (25). The training data is constructed by randomly extracting 300 overlapping patches from the $512 \times 512$ image of *barbara.png*, each with size $16 \times 16$ pixels. Each of the extracted patch is vectorized, resulting a training data set $Z$ of size $256 \times 300$. We consider a network of $N = 10$ agents, and the columns of $Z$ are evenly distributed among the agents (each having $P = 30$ columns). We compare Prox-PDA-IP with the EXTRA-AO algorithm proposed in (H.-T. Wai & Scaglione, 2015). Note that the EXTRA-AO is also designed for a similar distributed matrix factorization problem and it works well in practice. However, it does not have formal convergence proof. We initialize both algorithms with $X$ being the 2D discrete cosine transform (DCT) matrix. We set $\gamma = 0.05$, $\tau = 10^5$ and $\beta = 0.001r$, and the results are averaged over 30 problem instances. The stepsizes of the EXTRA-AO is set as $\alpha_{\rm AO} = 0.03$ and $\beta_{\rm AO} = 0.002$. In Fig. 3, we compare the performance of the proposed Prox-PDA-IP and the EXTRA-AO. It can be observed that our proposed algorithm converges faster than the EXTRA-AO. We have observed that the EXTRA-AO does have reasonably good practical performance, however it lacks formal convergence proof.

## Acknowledgment

The authors supported by National Natural Science Foundation (Grant No. CCF-1526078).

## References


Allen-Zhu, Z. and Hazan, E. Variance Reduction for Faster Non-Convex Optimization. In *Proceedings of the 33rd International Conference on Machine Learning*, ICML, 2016.

Antoniadis, A., Gijbels, I., and Nikolova, M. Penalized likelihood regression for generalized linear models with non-quadratic penalties. *Annals of the Institute of Statistical Mathematics*, 63(3):585–615, 2009.

Aybat, N-S. and Hamedani, E-Y. A primal-dual method for conic constrained distributed optimization problems. *Advances in Neural Information Processing Systems*, 2016.

Bertsekas, D. P. *Constrained Optimization and Lagrange Multiplier Method*. Academic Press, 1982.

Bianchi, P. and Jakubowicz, J. Convergence of a multi-agent projected stochastic gradient algorithm for non-convex optimization. *IEEE Transactions on Automatic Control*, 58(2):391–405, 2013.

Bjornson, E. and Jorswieck, E. Optimal resource allocation in coordinated multi-cell systems. *Foundations and Trends in Communications and Information Theory*, 9, 2013.

Cevher, V., Becker, S., and Schmidt, M. Convex optimization for big data: Scalable, randomized, and parallel algorithms for big data analytics. *IEEE Signal Processing Magazine*, 31(5): 32–43, 2014.

Defazio, A., Bach, F., and Lacoste-Julien, S. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *The Proceeding of NIPS*, 2014.

Forero, P. A., Cano, A., and Giannakis, G. B. Consensus-based distributed support vector machines. *Journal of Machine Learning Research*, 11(May):1663–1707, 2010.

H.-T. Wai, T.-H. Chang and Scaglione, A. A consensus-based decentralized algorithm for non-convex optimization with application to dictionary learning. In *the Proceedings of the IEEE ICASSP*, 2015.

Hajinezhad, D. and Hong, M. Nonconvex alternating direction method of multipliers for distributed sparse principal component analysis. In *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. IEEE, 2015.

Hajinezhad, D., Chang, T-H., Wang, X., Shi, Q., , and Hong, M. Nonnegative matrix factorization using admm: Algorithm and convergence analysis. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016a.

Hajinezhad, D., Hong, M., Zhao, T., and Wang, Z. Nestt: A nonconvex primal-dual splitting method for distributed and stochastic optimization. In *Advances in Neural Information Processing Systems 29*, pp. 3215–3223. 2016b.

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

Hong, M., Luo, Z.-Q., and Razaviyayn, M. Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems. 2014. technical report, University of Minnesota.

Hong, M., Razaviyayn, M., Luo, Z.-Q., and Pang, J.-S. A unified algorithmic framework for block-structured optimization involving big data. *IEEE Signal Processing Magazine*, 33(1): 57–77, 2016.

Johnson, R. and Zhang, T. Accelerating stochastic gradient descent using predictive variance reduction. In *the Proceedings of the Neural Information Processing (NIPS)*. 2013.

Ling, Q., Xu, Y., Yin, W., and Wen, Z. Decentralized low-rank matrix completion. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2925–2928, March 2012.

Ling, Q., Shi, W., Wu, G., and Ribeiro, A. DLM: Decentralized linearized alternating direction method of multipliers. *IEEE Transactions on Signal Processing*, 63(15):4051–4064, Aug 2015.

Lobel, I. and Ozdaglar, A. Distributed subgradient methods for convex optimization over random networks. *Automatic Control, IEEE Transactions on*, 56(6):1291–1306, 2011.

Lobel, I., Ozdaglar, A., and Feijer, D. Distributed multi-agent optimization with state-dependent communication. *Mathematical Programming*, 129(2):255–284, 2011.

Lorenzo, P. Di and Scutari, G. Next: In-network nonconvex optimization. *IEEE Transactions on Signal and Information Processing over Networks*, 2(2):120–136, 2016.

Nedic, A. and Olshevsky, A. Distributed optimization over time-varying directed graphs. *IEEE Transactions on Automatic Control*, 60(3):601–615, 2015.

Nedic, A. and Ozdaglar, A. Cooperative distributed multi-agent optimization. In *Convex Optimization in Signal Processing and Communications*. Cambridge University Press, 2009a.

Nedic, A. and Ozdaglar, A. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, 2009b.

Powell, M. M. D. An efficient method for nonlinear constraints in minimization problems. In *Optimization*. Academic Press, 1969.

Rahimpour, Alireza, Taalimi, Ali, Luo, Jiajia, and Qi, Hairong. Distributed object recognition in smart camera networks. In *IEEE International Conference on Image Processing, Phoenix, Arizona, USA*. IEEE, 2016.

Rahmani, M. and Atia, G. A decentralized approach to robust subspace recovery. In *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 802–807. IEEE, 2015.

Reddi, S., Sra, S., Poczos, B., and Smola, A. Fast incremental method for nonconvex optimization. *arXiv preprint arXiv:1603.06159*, 2016.

Scardapane, S. and Lorenzo, P. Di. A framework for parallel and distributed training of neural networks. *arXiv preprint arXiv:1610.07448*, 2016.

Scardapane, S., Fierimonte, R., Lorenzo, P. Di, Panella, M., and Uncini, A. Distributed semi-supervised support vector machines. *Neural Networks*, 80:43–52, 2016.

Schmidt, M., Roux, N. Le, and Bach., F. Minimizing finite sums with the stochastic average gradient. 2013. Technical report, INRIA.

Shi, W., Ling, Q., Wu, G., and Yin, W. Extra: An exact first-order algorithm for decentralized consensus optimization. *SIAM Journal on Optimization*, 25(2):944–966, 2014.

Tatarenko, T. and Touri, B. Non-convex distributed optimization. 2015. arXiv Preprint: arXiv:1512.00895.

Yan, F., Sundaram, S., Vishwanathan, S. V. N., and Qi, Y. Distributed autonomous online learning: Regrets and intrinsic privacy-preserving properties. *IEEE Transactions on Knowledge and Data Engineering*, 25(11):2483–2493, 2013.

Zhu, M. and Martinez, S. An approximate dual subgradient algorithm for multi-agent non-convex optimization. In *49th IEEE Conference on Decision and Control (CDC)*, pp. 7487–7492, 2010.

Zlobec, S. On the liu-floudas convexification of smooth programs. *Journal of Global Optimization*, 32(3):401–407, 2005.