A Linearized Bregman Algorithm for Decentralized Basis Pursuit

Kun Yuan, Qing Ling, Wotao Yin, Alejandro Ribeiro

***Abstract*—In this paper we solve a decentralized basis pursuit problem in a multiagent system where each agent holds part of the linear observations on a common sparse vector. The agents collaborate to recover the sparse vector through limited neighboring communication. The proposed decentralized linearized Bregman algorithm solves the Lagrange dual of an augmented ℓ_1 model that is equivalent to basis pursuit. The fact that this dual problem is unconstrained and differentiable enables a lightweight yet efficient decentralized gradient algorithm. We prove nearly linear convergence of the dual and primal variables to their optima. Numerical experiments demonstrate the effectiveness of the proposed algorithm.**

***Index Terms*—Basis pursuit, linearized Bregman, decentralized computation**

I. Introduction

Consider a multiagent system of n distributed agents who collaboratively recover a sparse signal $x \in \mathbb{R}^p$ from the linear measurements that they individually collect. Agent i collects measurements $b_i = A_i x$, where $b_i \in \mathbb{R}^{q_i}$ and $A_i \in \mathbb{R}^{q_i \times p}$. Let

$$b \triangleq \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix} \in \mathbb{R}^q, \quad A \triangleq \begin{bmatrix} —A_1— \\ \vdots \\ —A_n— \end{bmatrix} \in \mathbb{R}^{q \times p},$$

where $q = \sum_{i=1}^n q_i$. We propose a decentralized algorithm for the agents to collaboratively solve the basis pursuit problem [1]:

$$\begin{array}{ll} \min\limits_x & ||x||_1 \\ s.t. & Ax = b. \end{array} \tag{1}$$

According to the compressive sensing theory [2], [3], if A satisfies certain properties and x is sufficiently sparse, model (1) exactly recovers x. If b is contaminated by noise or x is only approximately sparse, stable recovery can be guaranteed if one replaces the constraints $Ax = b$ by $\|Ax-b\| \le \epsilon$, where $\|\cdot\|$ is the ℓ_2-norm and ϵ is the estimated noise level, or stops our algorithm when $\|Ax - b\| \approx \epsilon$.

Distributed basis pursuit is applied in collaborative spectrum sensing [4], [5], [6], [7], where the goal is to recover a wideband spectrum signal x, which is sparse. Agent i takes its measurement $b_i = A_i x$ with sensing matrix A_i, and all agents collaborate to recover x through solving model (1).

K. Yuan and Q. Ling are with Dept. of Automation, University of Science and Technology of China, Hefei, Anhui, China 230026. W. Yin is with Dept. of Computational and Applied Mathematics, Rice University, Houston, TX, USA 77005. A. Ribeiro is with Dept. of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA, USA 19104. Q. Ling is supported by NSFC grant 61004137. W. Yin is supported by ARL and ARO grant W911NF-09-1-0383, NSF grants DMS-0748839 and ECCS-1028790. A. Ribeiro is supported by NSF CAREER CCF-0952867, NSF CCF-1017454, and AFOSR MURI FA9550-10-1-0567.

Decentralized optimization has several advantages for multiagent systems. In *centralized computation*, the agents need to transmit all their data, A_i and b_i in our case, to a fusion center via multi-hop communication and the fusion center solves (1) and broadcasts the solutions to the agents. This approach is energy-consuming and vulnerable to network and fusion center failures. In *decentralized computation*, each agent only exchanges a limited amount of information with its one-hop neighbors and keeps its data (e.g., A_i and b_i) private. The optimization is completed without a fusion center [8], [9].

Consider the decentralized basis pursuit problem where the sensing matrix A is partitioned by rows. The optimization variable is common to all agents and each agent holds part of the objective function and part of the constraint. For this kind of problem, existing decentralized algorithms include distributed subgradient descent [10], distributed stochastic subgradient projection [11], and alternating direction algorithm of multipliers (ADMM) [8], [12], [13]. In these algorithms, each agent holds a local solution; at each iteration, agents exchange the local solutions with their one-hop neighbors.

For distributed subgradient descent, each agent first computes a new solution through combining local solutions of itself and its one-hop neighbors with weighted average, and then descends along its local negative subgradient direction [10]. Distributed stochastic subgradient projection is similar to distributed subgradient descent but includes an extra projection operation to a common constraint set [11]. Though easy to implement and suit for asynchronous networks, these two algorithms are not competitive in solving the decentralized basis pursuit problem since they do not handle the constraints $Ax = b$ efficiently and subgradient descent does not take advantage of the structure of $\|x\|_1$.

The ADMM approach for (1) explicitly introduces consensus constraints. Solving this consensus-constrained problem with skillful variable splitting leads to decentralized algorithms. ADMM-based decentralized algorithms converge globally for convex problems, and have linear convergence when each local objective function is differentiable, strongly convex and having Lipschitz continuous gradient [14]. For (1), the ADMM-based approach requires each agent to solve a least absolute shrinkage and selection operator (LASSO) subproblem with p unknown at each iteration, requiring a rather significant amount of computation on each agent [5], [7]. [6] elegantly simplifies the subproblem for each agent but its algorithm requires much more iterations to converge.

This paper proposes a decentralized linearized Bregman algorithm for solving (1). The proposed algorithm is very easy to implement, converges fast at a nearly linear rate, and applies to asynchronous networks. In particular, the main computation of each at each step is two matrix-vector multiplications involving A_i, which are much cheaper than solving a LASSO subproblem involving A_i. The basic idea is to apply a decentralized gradient method to a smoothed dual problem of (1), where the smoothing does not change the solution yet ensures the applicability and fast convergence of decentralized gradient iterations. In addition to showing that the decentralized linearized Bregman algorithm has nearly linear convergence to the solution of (1), the number of iterations needed is comparable to that of the ADMM-based algorithms. Hence, the proposed algorithm appears to have the state-of-the-art performance.

Notation $\mathcal{N}_i$ is the set of one-hop neighbors of agent i. Shrink(x) is an operator equal to $\max\{|x_i|, 0\}\text{sign}(x_i)$ element-wise. $\|\cdot\|$ is vector ℓ_2-norm or matrix spectral norm.

II. Background of Linearized Bregman

Linearized Bregman solves model (1) by solving

$$\begin{array}{ll} \min\limits_{x} & ||x||_1 + \frac{1}{2\alpha}||x||^2 \\ s.t. & Ax = b, \end{array} \tag{2}$$

where $\alpha > 0$ is chosen so that (2) returns a solution to (1). In fact, there exists $\alpha_{\min} > 0$ such that the solution to (2) is also a solution to (1) for any $\alpha \geq \alpha_{\min}$ [15]. For compressive sensing, $\alpha_{\min} = 10\|x^o\|_\infty$, where x^o is the original signal, is shown to work well [16]. Model (2) is easier to solve than (1) since the Lagrange dual of (2) is unconstrained and differentiable, subject to efficient gradient algorithms (c.f. [16], [17]). The Lagrange dual of (2) (posted as a minimization problem instead of a maximization one) is

$$\min_{y} \quad f(y) \triangleq \frac{\alpha}{2}||A^T y - \text{Proj}_{[-1,1]}(A^T y)||^2 - b^T y, \tag{3}$$

where $y \in \mathbb{R}^q$ is the dual variable and Proj$_{[-1,1]}$ denotes element-wise projection to interval $[-1, 1]$. Its gradient is

$$\nabla f(y) = \alpha A \text{Shrink}(A^T y) - b. \tag{4}$$

The linearized Bregman algorithm solves (3) by gradient descent. The updates at iteration k are

$$\begin{gathered} y(k+1) = y(k) - h(k)(Ax(k) - b), \\ \text{where } x(k) = \alpha \text{Shrink}(A^T y(k)), \end{gathered} \tag{5}$$

and $h(k)$ is the stepsize. [16] shows that (3) is strongly convex in a restricted sense and thus, if the stepsizes are fixed or chosen by line search, both $x(k)$ and $y(k)$ converge linearly.

III. Decentralized Linearized Bregman

Suppose that the multi-agent system lays over a bidirectionally connected network. For simplicity, we describe a synchronous version of our algorithm though it can run asynchronously. Each agent i keeps $x_i \in \mathbb{R}^p$, which a local estimate of the common x. Every x_i will converge to x.

Algorithm 1 Decentralized linearized Bregman at agent i

```
Require: Sensing matrix $A_i$ and measurements $b_i$.
Require: Doubly stochastic weight matrix $W$.
1: Initialize $v_i(0) = 0$;
2: for $k = 0, 1, 2, \ldots, K$, agent $i$ do
3:     Compute $u_i(k+1)$ according to (6a);
4:     Transmit $u_i(k+1)$ to, and receive $u_j(k+1)$ from $j \in \mathcal{N}_i$;
5:     Compute $v_i(k+1)$ according to (6b);
6: end for
7: Return $x_i$ in (6a).
```

Algorithm 1 gives the proposed algorithm. Agent i does

$$\textit{descent}\text{: } u_i(k+1) = v_i(k) - h(k)A_i^T(A_i x_i(k) - b_i), \tag{6a}$$
$$\text{where } x_i(k) = \alpha \,\text{Shrink}(n v_i(k)),$$
$$\textit{averaging}\text{: } v_i(k+1) = \sum_{j \in \mathcal{N}_i} w_{ij} u_j(k+1), \tag{6b}$$

where $u_i, v_i \in \mathbb{R}^p$ are two auxiliary variables, and $W = [w_{ij}]$ is a doubly stochastic weight matrix satisfying: $\sum_{j=1}^n w_{ij} = 1$, $\sum_{j=1}^n w_{ji} = 1$, and $w_{ij} \neq 0$ if and only if $j \in \mathcal{N}_i \cup i$. Prior to (6b), agent i transmits $u_i(k+1)$ to, and receives $u_j(k+1)$ from, its one-hop neighbors $j \in \mathcal{N}_i$. Its raw data (i.e., A_i and b_i) is kept locally.

To see how algorithm (6) is related to (5), we partition $y = [y_1; \ldots; y_n]$ where $y_i \in \mathbb{R}^{q_i}$. We similarly partition (5) and then multiply A_i^T to both sides, arriving at

$$A_i^T y_i(k+1) = A_i^T y_i(k) - h(k)A_i^T(A_i x(k) - b_i)$$

over $i = 1, \ldots, n$. Comparing the left-hand side of this with that of (6a), we see $u_i \sim A_i^T y_i$ ("$\sim$" means "is an local estimate of"). Suppose for moment that (6b) is repeated infinitely many times at each k, then $v_i(k) = \frac{1}{n}\sum_{j=1}^n u_j(k) \sim \frac{1}{n}\sum_{i=1}^n A_i^T y_i(k) = \frac{1}{n}A^T y$ and thus $x_i(k) = \alpha \,\text{Shrink}(n v_i(k)) \sim \alpha \,\text{Shrink}(A^T y(k)) = x(k)$. Putting together, we have $u_i \sim A_i^T y_i$, $v_i \sim \frac{1}{n}A^T y$, and $x_i \sim x$. If "$\sim$" was "$=$", summing up (6a) over i gives (5).

The choice of W affects the diffusion speed of (6b) and thus the convergence of the algorithm. In an asynchronous network, W can vary over iterations. As we focus exclusively on the synchronous case, we fix W according to either the maximum degree (MD) rule or the Metropolis-Hastings (MH) rule [18].

IV. Convergence Analysis

We first introduce a lemma which describes the *restricted strong convexity property* of the dual function $f(y)$ (c.f. [16]).

Lemma 1 *Consider $f(y)$ in* (3) *where A and b are nonzero. Assume that $Ax = b$ is consistent. Proj$_*(y)$ denotes the projection of y onto the solution set of* (3). *Then $\exists \nu > 0$ such that the objective function $f(y)$ in* (3) *satisfies*

$$\nabla^T f(y)(y - \textit{Proj}_*(y)) \geq \nu\|y - \textit{Proj}_*(y)\|^2, \text{ for all } y. \tag{7}$$

An explicit formula of ν can be found in [16].

Definition We define some variables that appear in the theorem below. Let $z_i(k) \triangleq A_i x_i(k) - b_i \in \mathbb{R}^{q_i}$ and $y_i(k+1) \triangleq$

$y_i(k) - h(k)z_i(k) \in \mathbb{R}^{q_i}$ with $y_i(0) = 0$; the recursion implies that $y_i(k) = -\sum_{s=0}^{k-1} h(s)z_i(s)$. Defining

$$z = \begin{bmatrix} z_1 \\ \vdots \\ z_n \end{bmatrix} \in \mathbb{R}^q \text{ and } y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \in \mathbb{R}^q,$$

$y(k)$ satisfies equations $y(k+1) = y(k) - h(k)z(k)$ and $y(k) = -\sum_{s=0}^{k-1} h(s)z(s)$. Define $\bar{x}(k) \triangleq \alpha\text{Shrink}(n\bar{v}(k))$ where $\bar{v}(k) \triangleq -\frac{1}{n}\sum_{s=0}^{k-1}\sum_{j=1}^{n} h(s)A_j^T z_j(s)$. Substituting the equation $y_i(k) = -\sum_{s=0}^{k-1} h(s)z_i(s)$ into the definition of $\bar{v}(k)$, we have $\bar{v}(k) = \frac{1}{n}\sum_{j=1}^{n} A_j^T r_j(k) = \frac{1}{n}A^T y(k)$. Let be the unique solution to problem (2).

Theorem 1 *Consider Algorithm 1 defined by* (6a)-(6b). *Assume that the largest eigenvalue of the weigh matrix W is* 1 *and the second largest is $\beta < 1$. Assume a fixed stepsize $h(k) = h$ and $\|z(k)\| \leq L$ where L is a positive constant. Then*

$$\begin{aligned}&\|y(k+1) - \textit{Proj}_*(y(k+1))\| \\ \leq&\rho\|y(k) - \textit{Proj}_*(y(k))\| + \gamma,\end{aligned} \tag{8}$$

$\rho = \sqrt{1 + \delta^2 - 2h\nu} < 1$ *and* $\gamma = \frac{h^2 L\alpha n^{\frac{3}{2}}}{\delta(1-\beta)}(\max_i \|A_i\|)^2 + hL$ *with δ being an arbitrary positive constant. Further*

$$\begin{aligned}&\|x_i(k) - x^*\| \\ \leq&\alpha\|A\|\|y(k) - \textit{Proj}_*(y(k))\| + \frac{\alpha nhL}{1-\beta}\max_j \|A_j\|.\end{aligned} \tag{9}$$

Proof: *Step 1: Bounding $\|v_i(k) - \bar{v}(k)\|$.*

Combining (6a) and (6b) and using $v_i(0) = 0$, we eliminate u_i and obtain the expression of $v_i(k)$ as

$$v_i(k) = -\sum_{s=0}^{k-1}\sum_{j=1}^{n}(W^{k-s})_{ij} hA_j^T z_j(s), \tag{10}$$

where $(W^{s+1})_{ij}$ denotes the (i,j)th entry of the matrix W^{s+1}. Recall that the largest eigenvalue of W is 1 and the second largest is $\beta < 1$. Then

$$\begin{aligned}&\|\sum_{j=1}^{n}(W^{k-s})_{ij} hA_j^T z_j(s) - \frac{1}{n}\sum_{j=1}^{n} hA_j^T z_j(s)\| \\ \leq&\|\frac{1}{n}\sum_{j=1}^{n}\beta^{k-s} hA_j^T z_j(s)\|.\end{aligned} \tag{11}$$

Because $\bar{v}(k) = -\frac{1}{n}\sum_{s=0}^{k-1}\sum_{j=1}^{n} hA_i^T z_i(s)$, we know

$$\begin{aligned}&\|v_i(k) - \bar{v}(k)\| \\ \leq&\sum_{s=0}^{k-1}\|\frac{1}{n}\sum_{j=1}^{n}\beta^{k-s} hA_j^T z_j(s)\| \\ \leq&\sum_{s=0}^{k-1}\beta^{k-s} h\max_j\|A_j\|\|\frac{1}{n}\sum_{j=1}^{n} z_j(s)\| \\ \leq&\sum_{s=0}^{k-1}\beta^{k-s} h\max_j\|A_j\|\|[z_1(s);\ldots;z_n(s)]\| \\ =&h\max_j\|A_j\|\sum_{s=0}^{k-1}\beta^{k-s}\|z(s)\|.\end{aligned} \tag{12}$$

By assumption $\|z\| \leq L$, therefore

$$\begin{aligned}&\|v_i(k) - \bar{v}(k)\| \\ =&h\max_j\|A_j\|\sum_{s=0}^{k-1}\beta^{k-s}\|z(s)\| \\ \leq&hL\max_j\|A_j\|\sum_{s=0}^{k-1}\beta^{k-s} \\ \leq&\frac{hL}{1-\beta}\max_j\|A_j\|.\end{aligned} \tag{13}$$

The right-hand-side of (13) is irrelevant with i, hence

$$\begin{aligned}&\max_i\|v_i(k) - \bar{v}(k)\| \\ \leq&\frac{hL}{1-\beta}\max_i\|A_i\|.\end{aligned} \tag{14}$$

Step 2: Bounding $\|y(k) - \textit{Proj}_*(y(k))\|$.

Utilizing the property of projection as well as the recursion $y(k+1) = y(k) - hz(k)$, we know that

$$\begin{aligned}&\|y(k+1) - \text{Proj}_*(y(k+1))\|^2 \\ \leq&\|y(k+1) - \text{Proj}_*(y(k))\|^2 \\ =&\|y(k) - \text{Proj}_*(y(k)) - hz(k)\|^2.\end{aligned} \tag{15}$$

Expanding (15) results in

$$\begin{aligned}&\|y(k+1) - \text{Proj}_*(y(k+1))\|^2 \\ \leq&\|y(k) - \text{Proj}_*(y(k))\|^2 + h^2\|z(k)\|^2 \\ &- 2hz(k)^T[y(k) - \text{Proj}_*(y(k))] \\ =&\|y(k) - \text{Proj}_*(y(k))\|^2 + h^2\|z(k)\|^2 \\ &- 2h[A\bar{x}(k) - b]^T[y(k) - \text{Proj}_*(y(k))] \\ &+ 2h[A\bar{x}(k) - b - z(k)]^T[y(k) - \text{Proj}_*(y(k))].\end{aligned} \tag{16}$$

Consider $-2h[A\bar{x}(k) - b]^T[y(k) - \text{Proj}_*(y(k))]$. Recall that $\bar{v}(k) = \frac{1}{n}A^T y(k)$ and $\bar{x}(k) = \alpha\text{Shrink}(n\bar{v}(k))$. According to (4), the gradient of $f(y)$ at $y(k)$ is

$$\begin{aligned}&\nabla f(y(k)) \\ =&\alpha A\text{Shrink}(A^T y(k)) - b \\ =&\alpha A\text{Shrink}(n\bar{v}(k)) - b \\ =&A\bar{x}(k) - b.\end{aligned} \tag{17}$$

Replacing $\nabla f(y(k)) = A\bar{x}(k) - b$ to (7) leads to

$$\begin{aligned}&[A\bar{x}(k) - b]^T[y(k) - \text{Proj}_*(y(k))] \\ \geq&\nu\|y(k) - \text{Proj}_*(y(k))\|^2.\end{aligned} \tag{18}$$

For $2h[A\bar{x}(k) - b - z(k)]^T[y(k) - \text{Proj}_*(y(k))]$, we have the inequality

$$\begin{aligned}&2h[A\bar{x}(k) - b - z(k)]^T[y(k) - \text{Proj}_*(y(k))] \\ \leq&\frac{h^2}{\delta^2}\|A\bar{x}(k) - b - z(k)\|^2 + \delta^2\|y(k) - \text{Proj}_*(y(k))\|^2,\end{aligned} \tag{19}$$

where δ is an arbitrary positive constant.

Substituting (18) and (19) to (16) and collecting terms, we have

$$\begin{aligned}&\|y(k+1)-\text{Proj}_*(y(k+1))\|^2\\ \leq&[1+\delta^2-2h\nu]\|y(k)-\text{Proj}_*(y(k))\|^2+h^2\|z(k)\|^2\\ &+\frac{h^2}{\delta^2}\|A\bar{x}(k)-b-z(k)\|^2. \end{aligned}\tag{20}$$

By definition $z_i(k)\triangleq A_ix_i(k)-b_i$ and $z\triangleq[z_1;\ldots;z_n]$

$$\begin{aligned}&\|A\bar{x}(k)-b-z(k)\|^2\\ \leq&n\max_i\|A_i\bar{x}(k)-b_i-z_i(k)\|^2\\ =&n\max_i\|A_i\bar{x}(k)-A_ix_i(k)\|^2\\ \leq&n\max_i[\|A_i\|^2\|x_i(k)-\bar{x}(k)\|^2]\\ \leq&n(\max_i\|A_i\|)^2(\max_i\|x_i(k)-\bar{x}(k)\|)^2. \end{aligned}\tag{21}$$

Since $x_i(k)=\alpha\text{Shrink}(nv_i(k))$ and $\bar{x}(k)=\alpha\text{Shrink}(n\bar{v}(k))$

$$\begin{aligned}&\max_i\|x_i(k)-\bar{x}(k)\|\\ =&\max_i\|\alpha\text{Shrink}(nv_i(k))-\alpha\text{Shrink}(n\bar{v}(k))\|\\ =&\max_i\alpha\|\text{Shrink}(nv_i(k))-\text{Shrink}(n\bar{v}(k))\|\\ \leq&\max_i\alpha\|nv_i(k)-n\bar{v}(k)\|\\ =&\max_i\alpha n\|v_i(k)-\bar{v}(k)\|, \end{aligned}\tag{22}$$

then we have

$$\begin{aligned}&\|A\bar{x}(k)-b-z(k)\|^2\\ \leq&\alpha^2n^3(\max_i\|A_i\|)^2(\max_i\|v_i(k)-\bar{v}(k)\|)^2. \end{aligned}\tag{23}$$

Substituting (23) to (20), we can obtain the upper bound of $\|y(k+1)-\text{Proj}_*(y(k+1))\|^2$

$$\begin{aligned}&\|y(k+1)-\text{Proj}_*(y(k+1))\|^2\\ \leq&[1+\delta^2-2h\nu]\|y(k)-\text{Proj}_*(y(k))\|^2+h^2\|z(k)\|^2\\ &+\frac{h^2\alpha^2n^3}{\delta^2}(\max_i\|A_i\|)^2(\max_i\|v_i(k)-\bar{v}(k)\|)^2, \end{aligned}\tag{24}$$

which implies that

$$\begin{aligned}&\|y(k+1)-\text{Proj}_*(y(k+1))\|\\ \leq&\sqrt{1+\delta^2-2h\nu}\|y(k)-\text{Proj}_*(y(k))\|+h\|z(k)\|\\ &+\frac{h\alpha n^{3/2}}{\delta}\max_i\|A_i\|\max_i\|v_i(k)-\bar{v}(k)\|. \end{aligned}\tag{25}$$

Substituting (14) and $\|z(k)\|\leq L$ to (25) yields

$$\begin{aligned}&\|y(k+1)-\text{Proj}_*(y(k+1))\|\\ \leq&\sqrt{1+\delta^2-2h\nu}\|y(k)-\text{Proj}_*(y(k))\|+h\|z(k)\|\\ &+\frac{h\alpha n^{3/2}}{\delta}\max_i\|A_i\|\max_i\|v_i(k)-\bar{v}(k)\|\\ \leq&\sqrt{1+\delta^2-2h\nu}\|y(k)-\text{Proj}_*(y(k))\|+hL\\ &+\frac{h^2L_z\alpha n^{3/2}}{\delta(1-\beta)}(\max_i\|A_i\|)^2\\ =&\rho\|y(k)-\text{Proj}_*(y(k))\|+\gamma. \end{aligned}\tag{26}$$

Step 3: Bounding $\|x_i(k)-x^*\|$.

Given any dual solution $y(k)$, the primal optimum of (2) is $x^*=\alpha\text{Shrink}(A^T\text{Proj}_*(y(k)))$. According to the iterate of $x_i(k)=\alpha\text{Shrink}(nv_i(k))$

$$\begin{aligned}&\|x_i(k)-x^*\|\\ =&\|\alpha\text{Shrink}(nv_i(k))-\alpha\text{Shrink}(A^T\text{Proj}_*(y(k)))\|\\ =&\|\alpha\text{Shrink}(A^Ty(k))-\alpha\text{Shrink}(A^T\text{Proj}_*(y(k)))\\ &+\alpha\text{Shrink}(nv_i(k))-\alpha\text{Shrink}(A^Ty(k))\|\\ \leq&\|\alpha\text{Shrink}(A^Ty(k))-\alpha\text{Shrink}(A^T\text{Proj}_*(y(k)))\|\\ &+\|\alpha\text{Shrink}(nv_i(k))-\alpha\text{Shrink}(A^Ty(k))\|. \end{aligned}\tag{27}$$

Using the nonexpansive property of the shrinkage operator

$$\begin{aligned}&\|\alpha\text{Shrink}(A^Ty(k))-\alpha\text{Shrink}(A^T\text{Proj}_*(y(k)))\|\\ \leq&\alpha\|A^Ty(k)-A^T\text{Proj}_*(y(k))\|\\ \leq&\alpha\|A\|\|y(k)-\text{Proj}_*(y(k))\|. \end{aligned}\tag{28}$$

Since $\bar{v}=\frac{1}{n}A^Ty(k)$, we have

$$\begin{aligned}&\|\alpha\text{Shrink}(nv_i(k))-\alpha\text{Shrink}(A^Ty(k))\|\\ \leq&\alpha\|nv_i(k)-A^Ty(k)\|\\ =&\alpha\|nv_i(k)-n\bar{v}(k)\|\\ =&\alpha n\|v_i(k)-\bar{v}(k)\|. \end{aligned}\tag{29}$$

Substituting (28) and (29) to (27) leads to

$$\begin{aligned}&\|x_i(k)-x^*\|\\ \leq&\alpha\|A\|\|y(k)-\text{Proj}_*(y(k))\|+\alpha n\|v_i(k)-\bar{v}(k)\|. \end{aligned}\tag{30}$$

Further using (13), we obtain (9). ■

Remark Let us consider the dual convergence result (8). If we choose $\delta^2=h\nu$, then the two constants $\rho=\sqrt{1-h\nu}$ and $\gamma=\frac{h^{\frac{3}{2}}L\alpha n^{\frac{3}{2}}}{\nu^{\frac{1}{2}}(1-\beta)}(\max_i\|A_i\|)^2+hL$. Setting stepsize h to be small enough γ is in the order of h. This way, the algorithm converges to a small neighborhood of the dual optimum set with linear rate $\rho=\sqrt{1-h\nu}$.

In the primal convergence result (9), if we ignore the term $\frac{\alpha nhL}{1-\beta}\max_j\|A_j\|$, a local solution x_i converges to the primal optimum with nearly R-linear rate. The term $\frac{\alpha nhL}{1-\beta}\max_j\|A_j\|$ is proportional to h, suggesting that a small stepsize h enables x_i to converge to a small neighborhood of the primal optimum.

Theorem 1 assumes that $\|z(k)\|\leq L$. By definition $z(k)=[A_1x_1(k)-b_1;\ldots;A_nx_n(k)-b_n]$, $\|z(k)\|$ is upper bounded if every $\|x_i(k)\|$ is upper bounded. This assumption is common in convergence analysis.

V. Numerical Experiments

In the simulation, we generate a multiagent system with $n=50$ agents. The agents are uniformly randomly deployed in a 100×100 area and two agents are one-hop neighbors if their distance is within 30. The generated network is connected. The sparse signal to recover is $x\in\mathbb{R}^{200}$ and its sparsity is 20; nonzero elements of x are generated following the Gaussian distribution. The sensing matrix $A\in\mathbb{R}^{100\times200}$ is generated following the Gaussian distribution. The measurements $b=$

Ax are noise-free. Each agent holds two rows of A and b, i.e., $A_i \in \mathbb{R}^{2\times 200}$ and $b_i \in \mathbb{R}^2$.

We compare the proposed Algorithm 1 (DLB) with two existing algorithms, DLASSO in [6] and DADMM in [7]. We consider the standard DLB (DLB-STD) and its variant with restarted Nesterov acceleration (DLB-ACC) [19]. Recall from Section I that DADMM has the most expensive iteration since each agent solves a LASSO subproblem at each iteration. In DLASSO, each agent solves a ridge regression subproblem at each iteration. The two DLB iterations have the lowest per-iteration cost.

In the numerical experiments, we first compare the convergence rates of DLB, DLASSO, and DADMM, and then study how the weight matrix W influences the convergence rate of DLB. The convergence is measured by relative error, which is defined as

$$\frac{1}{n}\sum_{i=1}^{n}\|x_i(k) - x^*\|.$$

The parameter α in (2) is set as 4 as suggested in [16]. In DLB-ACC the Nesterov acceleration is restarted every 50 iterations. The parameters in DLASSO and DADMM are both hand-tuned to the best.

Fig. 1 compares the convergence rates of DLB, DLASSO, and DADMM. The stepsize is set as $h(k) = 0.03$ for DLB-ACC and $h(k) = \min(0.06, \frac{1}{k})$ for DLB-STD; the weight matrix W is set according to the Metropolis-Hastings (MH) rule. DLB-ACC is the fastest among the four algorithms. Compared to DLB-STD, DLB-ACC reduces the number of iterations from $\sim$ 700 to $\sim$ 200 to reach 10^{-6} accuracy. DADMM is slower than DLB-ACC; further, each agent takes much more time at each iteration. DLASSO has modest per-iteration but is the slowest.

In Fig. 2 we compare the performance of DLB with two different weight matrices W: maximum degree (MD) and Metropolis-Hastings (MH). The stepsizes are adjusted to the best. In DLB-ACC, 0.02 for MD and 0.03 for MH; in DLB-STD, $h(k) = \min(0.04, \frac{1}{k})$ for MD and $h(k) = \min(0.06, \frac{1}{k})$ for MH. Experimental results suggest that MH is better than MD in both the standard and the accelerated DLB algorithms.

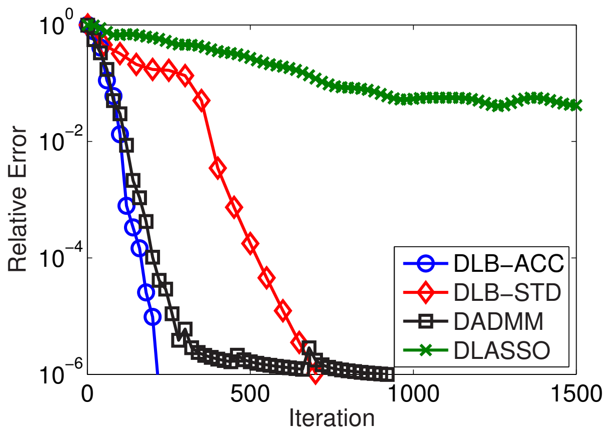

Fig. 1. Comparison of DLB, DLASSO, and DADMM.

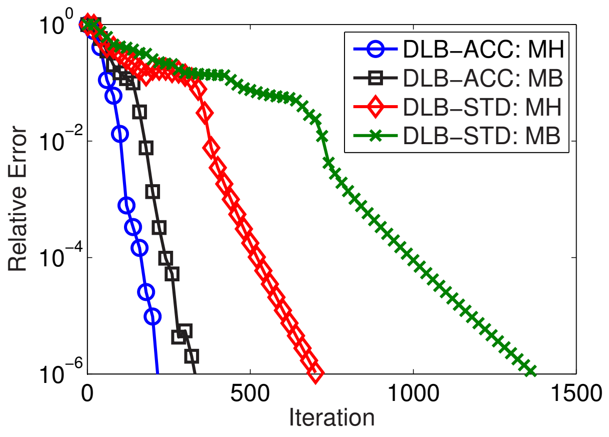

Fig. 2. DLB with different weight matrix W: maximum degree (MD) and Metropolis-Hastings (MH).

References

[1] S. Chen, D. Donoho, and M. Saunders, "Atomic decomposition by basis pursuit," SIAM Review, vol. 43, pp. 129–159, 2001

[2] D. Donoho, "Compressed sensing," IEEE Transactions on Information Theory, vol. 52, pp. 1289–1306, 2006

[3] E. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," IEEE Transaction on Information Theory, vol. 52, pp. 5406–5425, 2006

[4] Z. Tian, "Compressed wideband sensing in cooperative cognitive radio networks," Proceedings of GLOBECOM, 2008

[5] J. Bazerque and G. Giannakis, "Distributed spectrum sensing for cognitive radio networks by exploiting sparsity," IEEE Transactions on Signal Processing, vol. 58, pp. 1847–1862, 2010

[6] G. Mateos, J. Bazerque, and G. Giannakis, "Distributed sparse linear regression," IEEE Transactions on Signal Processing, vol. 58, pp. 5262–5276, 2010

[7] J. Mota, J. Xavier, P. Aguiar, and M. Puschel, "Distributed basis pursuit," IEEE Transactions on Signal Processing, vol. 60, pp. 1942–1956, 2012

[8] I. Schizas, A. Ribeiro, and G. Giannakis, "Consensus in ad hoc WSNs with noisy links - Part I: distributed estimation of deterministic signals," IEEE Transactions on Signal Processing, vol. 56, pp. 350–364, 2008

[9] J. Predd, S. Kulkarni, and H. Poor, "A collaborative training algorithm for distributed learning," IEEE Transactions on Information Theory, vol. 55, pp. 1856–1871, 2009

[10] A. Nedic and A. Ozdaglar, "Distributed subgradient algorithms for multi-agent optimization," IEEE Transactions on Automatic Control, vol. 54, pp. 48–61, 2009

[11] S. Ram, A. Nedic, and V. Veeravalli, "Distributed stochastic subgradient projection algorithms for convex optimization," Journal of Optimization Theory and Applications, vol. 147, pp. 516–545, 2010

[12] D. Bertsekas and J. Tsitsiklis, *Parallel and Distributed Computation: Numerical Algorithms*, Second Edition, Athena Scientific, 1997

[13] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction algorithm of multipliers," Foundations and Trends in Machine Learning, vol. 3, pp. 1–122, 2010

[14] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "Linearly convergent decentralized consensus optimization with the alternating direction algorithm of multipliers," Proceedings of ICASSP, 2013

[15] M. Friedlander and P. Tseng, "Exact regularization of convex programs," SIAM Journal on Optimization, vol. 18, pp. 1326–1350, 2007

[16] M. Lai and W. Yin, "Augmented ℓ_1 and nuclear-norm models with a globally linearly convergent algorithm," Manuscript

[17] W. Yin, "Analysis and generalizations of the linearized Bregman algorithm," SIAM Journal on Imaging Sciences, vol. 3, pp. 856–877, 2010

[18] S. Boyd, P. Diaconis, and L. Xiao, "Fastest mixing Markov chain on a graph," SIAM Review, vol. 46, pp. 667–689, 2004

[19] Y. Nesterov, "Gradient algorithms for minimizing composite objective function," CORE Discussion Paper, 2007