STOCHASTIC PROXIMAL GRADIENT CONSENSUS OVER TIME-VARYING NETWORKS

Mingyi Hong[†] *and Tsung-Hui Chang*[‡]

[†] Dept. of IMSE and ECE
Iowa State University
Ames, IA, 50011
E-mail: mingyi@iastate.edu

[‡] School of Science & Engineering
The Chinese University of Hong Kong, Shenzhen
Shenzhen, China 518172
E-mail: tsunghui.chang@ieee.org

ABSTRACT

We consider solving a convex, nonsmooth and stochastic optimization problem over a multi-agent network. Each agent has access to a local objective function and can communicate with its immediate neighbors only. We develop a dynamic stochastic proximal-gradient consensus (DySPGC) algorithm, featuring: *i)* it works for both the static and randomly time-varying networks; *ii)* it can deal with either the exact or the stochastic gradient information; *iii)* it has provable rate of convergence. Interestingly, the developed algorithm includes as special cases many existing (and seemingly unrelated) first-order algorithms for distributed optimization over static networks, such as the EXTRA (Shi *et al* 2014), the PG-EXTRA (Shi *at* 2015), the IC/IDC-ADMM (Chang *et al* 2014), and the DLM (Ling *et al* 2015). It is also closely related to the classical distributed gradient method.

Index Terms— Consensus optimization, alternating direction method of multipliers, stochastic optimization

1. INTRODUCTION

Consider the following classical global consensus problem

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

where $f_i(y)$ is a convex function. Suppose N agents are distributed over a network defined by an *undirected* graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, with $|\mathcal{V}| = N$ vertices and $|\mathcal{E}| = E$ edges. Each agent can communicate with its immediate neighbors, and it is responsible for optimizing one component function f_i. For applications of this model, see a recent survey [1]. The key research question is: how to enable the agents to distributedly compute an optimal solution of (1), using only local, and possibly inexact and stochastic, gradient information.

Suppose each agent i has a local copy of y, denoted as $x_i \in \mathbb{R}^M$, then the classical distributed subgradient (DSG) method is given by

$$x_i^{r+1} = \sum_{j=1}^N w_{ij}^r x_j^r - \gamma^r d_i^r, \ \forall\, i \in \mathcal{V}, \tag{2}$$

where r denotes the iteration counter; $d_i^r \in \partial f_i(x_i^r)$ is a subgradient vector; $w_{ij}^r \geq 0$ is the weight for the edge $e_{ij} \in \mathcal{E}$ at iteration r; and $\gamma^r > 0$ is the stepsize. The convergence of the DSG was first analyzed in [2], and it has been extended to many scenarios, e.g., when there are local constraints [3], or when the messages exchanged among the agents are quantized [4]. The DSG is known to converge with rate $\mathcal{O}(\ln(r)/\sqrt{r})$ [5]. Under certain smoothness assumption on f, Shi *et al* [6] propose an EXTRA algorithm which adds certain *error-correction* terms to the DSG iteration (2). EXTRA and its extension [7] achieve an $\mathcal{O}(1/r)$ convergence rate for smooth convex problem and linear convergence for smooth strongly convex problems. Analysis on related algorithms can be found in [5, 8, 9].

Another popular approach for distributed optimization is based on the alternating direction method of multipliers (ADMM) [10–13]. The $\mathcal{O}(1/r)$ rate of convergence for decentralized ADMM has been shown by [14] with stochastic communication graph, and the linear convergence is shown in [15] for smooth strongly convex problems over static networks. Almost all the ADMM-based methods require that each agent solves its local problem exactly (cf. [11, 13, 16–18]), which can be very expensive, except two related works [19, 20]. Recently, [21] demonstrates that the ADMM is also capable of solving certain *nonconvex* global consensus problem.

In this work, we consider the following popular structured version of the global consensus problem (1)

$$\min_{y\in\mathbb{R}^M} \ f(y) := \sum_{i=1}^N f_i(y) = \sum_{i=1}^N g_i(y) + h_i(y), \tag{3}$$

where each $g_i : \mathbb{R}^M \to \mathbb{R}$ is a smooth convex function; each $h_i : \mathbb{R}^M \to \mathbb{R}$ is a convex but possibly nonsmooth lower semi-continuous function. We propose an ADMM based method, named the *dynamic stochastic proximal-gradient consensus (DySPGC)*, featuring: *i)* When only an unbiased estimate of ∇g_i is known, it converges with a rate $\mathcal{O}(1/\sqrt{r})$; *ii)* When the exact ∇g_i is known, the rate improves to $\mathcal{O}(1/r)$; *iii)* The algorithm works for both the static and certain randomly time-varying networks.

What is more interesting is our insight into the connection between the proposed DySPGC and a few DSG-type methods. In particular, we show that EXTRA/PG-EXTRA [6, 7] are in fact special instances of the proposed DySPGC algorithm (when applied to a static network with symmetric weights and exact gradients). Further, we also establish a close connection between the DSG iteration (2) and the proposed DySPGC. Additionally our method generalizes other distributed ADMM-type methods such as the DLM [20] and the IC-ADMM [19].

2. SYSTEM MODEL

We begin by assuming that each h_i admits an "easy prox" operator [22], i.e., the following problem is easily solvable

$$\text{prox}_h^\beta(u) := \min_y h_i(y) + \frac{\beta}{2}\|y-u\|^2. \tag{4}$$

M. Hong is supported by NSF, Grant No. CCF-1526078. T.-H. Chang is supported by NSFC, China, Grant No. 61571385.

Assume that ∇g_i is Lipschitz continuous, i.e., for some $P_i > 0$,

$$\|\nabla g_i(y) - \nabla g_i(v)\| \leq P_i\|y - v\|, \ \forall\, y, v \in \mathrm{dom}(h), \ \forall\, i. \quad (5)$$

Suppose N agents are defined over a connected *undirected* graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$. Define a companion *symmetric directed* graph given by $\mathcal{G}_d = \{\mathcal{V}, \mathcal{A}, W\}$, where $\mathcal{A}$ is the set of directed arcs with $|\mathcal{A}| = 2E$, and for every edge $(i,j) \in \mathcal{E}$, there are two edges $e_{ij}, e_{ji} \in \mathcal{A}$; $W \in \mathbb{R}_+^{N\times N}$ is a *weight matrix*. Let us use $\mathcal{N}_i$ to denote the neighborhood of node i, i.e., $\mathcal{N}_i = \{j \mid e_{ij} \in \mathcal{A}\}$. Generally we assume that the weight matrix W is a (row) stochastic matrix, its diagonal elements are all positive, and its off-diagonal elements satisfy

$$W[i,j] > 0, \ \text{if } e_{ij} \in \mathcal{A}, \quad W[i,j] = 0, \ \text{otherwise.} \quad (6)$$

Reformulation: It is well-known that (3) can be reformulated by

$$\min \sum_{i=1}^N g_i(x_i) + h_i(x_i), \quad \text{s.t. } x_i = z_{ij}, \quad x_j = z_{ij}, \ \forall\, e_{ij} \in \mathcal{A}.$$

Define $x := \{x_i\} \in \mathbb{R}^{NM}$, and $z = \{z_{ij}\} \in \mathbb{R}^{2EM}$. Also introduce two matrices $A = [A_1; A_2] \in \mathbb{R}^{4EM\times NM}$ and $B = -[I_{2EM}; I_{2EM}] \in \mathbb{R}^{4EM\times 2EM}$, where the (q,i)th block of A_1 (resp. (q,j)th block of A_2) is I_M (M by M identity matrix) if the qth block of z is z_{ij}. Then, the previous problem can be transformed to the following compact representation [1, 13, 19, 20]

$$\min \ f(x) := \sum_{i=1}^N g_i(x_i) + h_i(x_i), \quad \text{s.t.} \quad Ax + Bz = 0. \quad \text{(P)}$$

Random Graph: We will use the following random graph [9, 14].

Definition 2.1 *(The randomly activated graph) At each time r, each link $e \in \mathcal{E}$ has a probability $p_e \in (0,1]$ of being active. The set of "active" nodes is: $\mathcal{V}^r := \{i \mid \exists e_{ij} \in \mathcal{A}^r, \ \forall\, i \in \mathcal{V}\}$. Further, assume that $\mathcal{G}$ is connected, and the graph realizations $\mathcal{G}^r$ and $\mathcal{G}^t$ are independent $\forall\, r \neq t$.*

Define a vector of positive constants $\rho := \{\rho_{ij} > 0\}_{e_{ij}\in\mathcal{A}}$. For a given graph $\mathcal{G}_d^r$ at each time r, construct a time-dependent diagonal matrix $\Gamma^r \succeq 0$ by $\Gamma^r = \mathrm{blkdg}[\Xi^r \otimes I_M, \Xi^r \otimes I_M] \in \mathbb{R}^{4EM\times 4EM}$, where $\Xi^r \in \mathbb{R}^{2E\times 2E}$ is a diagonal matrix induced by the graph $\mathcal{G}_d^r$ with $\Xi^r[q,q] = \rho_{ij}$ if link $e_{ij} \in \mathcal{A}^r$ and the qth block of z is z_{ij}; otherwise $\Xi^r[q,q] = 0$. Also define matrices $\Gamma \succ 0$ and $\Xi \succ 0$ similarly, but over the original graph $\mathcal{G}_d$.

Gradient Information: Assume that each agent i is responsible for a single component function $g_i + h_i$, and it only has an estimate of $\nabla g_i(x_i)$, denoted as $\widetilde{g}_i(x_i, \xi_i)$. Such estimate satisfies

$$\mathbb{E}[\widetilde{g}_i(x_i,\xi_i)] = \nabla g_i(x_i), \ \ \mathbb{E}\left[\|\widetilde{g}_i(x_i,\xi_i) - \nabla g_i(x_i)\|^2\right] \leq \sigma^2, \ \forall\, i,$$

where ξ_i is a random variable following an unknown distribution; σ^2 is the variance of the error. Let $\widetilde{G}(x,\xi) := \sum_{i=1}^N \widetilde{g}_i(x_i,\xi_i)$.

3. THE PROPOSED ALGORITHMS

Our proposed algorithm is based on the ADMM [11,23]. To proceed, we express the augmented Lagrangian of (P) as

$$L(x,z,\lambda) = \sum_{i=1}^N g_i(x_i) + h_i(x_i) + \langle \lambda, Ax + Bz\rangle + \frac{1}{2}\|Ax + Bz\|_\Gamma^2,$$

where $\lambda \in \mathbb{R}^{4EM}$ is the dual variable. Note that here a *matrix* penalization parameter Γ replaces the single parameter used in conventional ADMM. For each $i \in \mathcal{V}$, define $\Omega_i := \omega_i I_M \succeq 0$ as a proximal matrix. Define $\Omega := \mathrm{blkdiag}[\Omega_1, \cdots, \Omega_N] \succeq 0$. Let

$$M_+ := A_1^T + A_2^T \in \mathbb{R}^{NM\times 2EM}, \ M_- := A_1^T - A_2^T \in \mathbb{R}^{NM\times 2EM}.$$

To model the time-varying network, let us define $\widetilde{G}^{r+1}(x^r, \xi^{r+1}) := [a_1; \ a_2; \ \cdots; a_N] \in \mathbb{R}^{MN}$ with

$$a_i = \begin{cases} \widetilde{g}_i(x^r, \xi^{r+1}) & \text{if } \ i \in \mathcal{V}^{r+1} \\ 0 & \text{otherwise} \end{cases}$$

Define $h^{r+1}(x) := \sum_{i\in\mathcal{V}^{r+1}} h_i(x_i)$. Also define the matrices $A^r, B^r, \Omega^r, M_+^r, M_-^r$ similarly as A, B, Ω, M_+ and M_-, but over the graph realization $\mathcal{G}_d^r$ (i.e., entries corresponding to the inactive links/nodes are set to zeros). Using these definitions, we present in the table below the proposed algorithm, named the dynamic stochastic proximal-gradient consensus (DySPGC) algorithm.

Algorithm 1. The DySPGC Algorithm

At iteration 0, let $B^T\lambda^0 = 0$, $z^0 = \frac{1}{2}M_+^T x^0$.
At each iteration $r+1$, update the variable blocks by:

$$\begin{aligned} x^{r+1} &= \arg\min \ \left\langle \widetilde{G}^{r+1}(x^r,\xi^{r+1}), x - x^r\right\rangle + h^{r+1}(x) \\ &\quad + \frac{1}{2}\left\|A^{r+1}x + B^{r+1}z^r + \Gamma^{-1}\lambda^r\right\|_\Gamma^2 \\ &\quad + \frac{1}{2}\|x - x^r\|^2_{\Omega^{r+1}+\eta^{r+1}I_{MN}} \qquad (7a) \\ x_i^{r+1} &= x_i^r, \quad \text{if } i \notin \mathcal{V}^{r+1} \qquad (7b) \\ z^{r+1} &= \arg\min \frac{1}{2}\left\|A^{r+1}x^{r+1} + B^{r+1}z + \Gamma^{-1}\lambda^r\right\|_\Gamma^2 \qquad (7c) \\ z_{ij}^{r+1} &= z_{ij}^r, \quad \text{if } e_{ij} \notin \mathcal{A}^{r+1} \qquad (7d) \\ \lambda^{r+1} &= \lambda^r + \Gamma\left(A^{r+1}x^{r+1} + B^{r+1}z^{r+1}\right) \qquad (7e) \end{aligned}$$

When the network is static and the exact gradient is known, i.e., $\mathcal{G}_d^r = \mathcal{G}_d$ and $\widetilde{G}^{r+1}(x^r,\xi^{r+1}) = \nabla g(x^r)$ for all r, we can set $\eta^{r+1} = 0$ for all r. The DySCPA reduces to the following proximal gradient consensus (PGC) algorithm.

Algorithm 2. The PGC Algorithm

At iteration 0, let $B^T\lambda^0 = 0$, $z^0 = \frac{1}{2}M_+^T x^0$.
At each iteration $r+1$, update the variable blocks by:

$$\begin{aligned} x^{r+1} &= \arg\min_x \ \langle \nabla g(x^r), x - x^r\rangle + h(x) + \langle \lambda^r, Ax + Bz^r\rangle \\ &\quad + \frac{1}{2}\|Ax + Bz^r\|_\Gamma^2 + \frac{1}{2}\|x - x^r\|_\Omega^2 \qquad (8a) \\ z^{r+1} &= \arg\min_z \ \frac{1}{2}\left\|Ax^{r+1} + Bz + \Gamma^{-1}\lambda^r\right\|_\Gamma^2 \qquad (8b) \\ \lambda^{r+1} &= \lambda^r + \Gamma\left(Ax^{r+1} + Bz^{r+1}\right) \qquad (8c) \end{aligned}$$

Below we present distributed implementation of the proposed algorithms. Define a stepsize parameter β_i^{r+1} as

$$\beta_i^{r+1} := 2\Big(\sum_{j\in\mathcal{N}_i^{r+1}} \widehat{\rho}_{ij} + \frac{w_i}{2}\Big), \ \text{with } \widehat{\rho}_{ij} := \frac{\rho_{ij} + \rho_{ji}}{2}, \ \forall\, i.$$

Let us define a new *stepsize matrix* $\Upsilon^{r+1} := \text{diag}([\beta_1^{r+1}, \cdots, \beta_N^{r+1}]) \otimes I_M \succ 0$ and specialize the weight matrix $W^{r+1} \in \mathbb{R}^{N\times N}$ as

$$(W[i,j])^{r+1} = \begin{cases} \frac{\rho_{ji}+\rho_{ij}}{\sum_{\ell\in\mathcal{N}_i^{r+1}}(\rho_{\ell i}+\rho_{i\ell})+\omega_i} = \frac{\rho_{ji}+\rho_{ij}}{\beta_i^{r+1}}, & \text{if } e_{ij}\in\mathcal{A}^{r+1}, \\ \frac{\omega_i}{\sum_{\ell\in\mathcal{N}_i^{r+1}}(\rho_{\ell i}+\rho_{i\ell})+\omega_i} = \frac{w_i}{\beta_i^{r+1}}, & i=j, i\in\mathcal{V}^{r+1} \\ 0, & \text{otherwise,} \end{cases} \tag{9}$$

Clearly, for any given r, W^r is a row stochastic matrix (but not doubly stochastic) and it satisfies (6).

Let us split λ^r by $\lambda^r = [\delta^r; \gamma^r]$ where $\delta^r, \gamma^r \in \mathbb{R}^{2EM}$. It can be show that the DySPCA is equivalent to (see [24] for a proof):

$$x_i^{r+1} + \frac{1}{\beta_i^{r+1}}\zeta_i^{r+1} + \frac{\eta^{r+1}}{\beta_i^{r+1}}(x_i^{r+1}-x_i^r) + \frac{\sum_{j\in\mathcal{N}_i^{r+1}}(\delta_{ij}^r-\delta_{ji}^r)}{\beta_i^{r+1}}$$
$$= \frac{-1}{\beta_i^{r+1}}\left(\widetilde{g}_i(x_i^r,\xi_i^{r+1}) + \sum_{j\in\mathcal{N}_i^{r+1}}\left(\rho_{ij}z_{ij}^r+\rho_{ji}z_{ji}^r\right)+\omega_i x_i^r\right),\ \forall\, i\in\mathcal{V}^{r+1} \tag{10a}$$

$$x_i^{r+1} = x_i^r, \quad \forall\, i\notin\mathcal{V}^{r+1} \tag{10b}$$

$$z_{ij}^{r+1} = \begin{cases} \frac{1}{2}\left(x_i^{r+1}+x_j^{r+1}\right), & \text{if } e_{ij}\in\mathcal{A}^{r+1} \\ z_{ij}^r, & \text{otherwise} \end{cases} \tag{10c}$$

$$\delta_{ij}^{r+1} = \begin{cases} \delta_{ij}^r + \frac{\rho_{ij}}{2}(x_i^{r+1}-x_j^{r+1}), & \text{if } e_{ij}\in\mathcal{A}^{r+1} \\ \delta_{ij}^r & \text{otherwise.} \end{cases} \tag{10d}$$

for some $\zeta_i^{r+1}\in h_i(x_i^{r+1})$.

Surprisingly, when the network is static and $\widetilde{G}(x;\xi)=\nabla g(x)$, the PGC algorithm admits a single-variable characterization.

Proposition 3.1 *The iteration* (8a) – (8c) *has the following compact characterization for all* $r\geq 1$*:*

$$x^{r+1} - x^r + \Upsilon^{-1}(\zeta^{r+1}-\zeta^r) - \Upsilon^{-1}\left(-\nabla g\left(x^r\right)+\nabla g\left(x^{r-1}\right)\right)$$
$$= (W\otimes I_M)x^r - \frac{1}{2}(I_{MN}+W\otimes I_M)x^{r-1}. \tag{11}$$

In particular, each agent i *implements the following iteration*

$$x_i^{r+1} - x_i^r + \frac{1}{\beta_i}(\zeta_i^{r+1}-\zeta_i^r) - \frac{1}{\beta_i}\big(-\nabla g_i\big(x_i^r\big)+\nabla g_i\big(x_i^{r-1}\big)\big)$$
$$= \frac{1}{\sum_{j\in\mathcal{N}_i}\widehat{\rho}_{ij}+\omega_i}\big(\sum_{j\in\mathcal{N}_i}\widehat{\rho}_{ij}x_j^r+\omega_i x_i^r\big)$$
$$-\frac{1}{2}\big(x_i^{r-1}+\frac{1}{\sum_{j\in\mathcal{N}_i}\widehat{\rho}_{ij}+\omega_i}\big(\sum_{j\in\mathcal{N}_i}\widehat{\rho}_{ij}x_j^{r-1}+\omega_i x_i^{r-1}\big)\big)$$

Remark 3.1 *If* $h\equiv 0$ *(no nonsmooth term), the iteration* (10a)–(10d) *can be implemented in a straightforward manner (i.e., in closed-form). Specifically, at iteration* $r+1$*, each node* i *updates* x_i *and* $\{z_{ij},\delta_{ij}\mid\forall\, j\in\mathcal{N}_i^{r+1}\}$*. As long as node* i *can communicate with its neighbors, these updates can be easily performed. When* h *is present, it can be shown that the iteration* (11) *is equivalent to*

$$x_i^{r+1} = \text{prox}_{h_i}^{\beta_i}\big(-\frac{1}{\beta_i}\nabla g_i(x_i^r)+\widehat{W}_i x^r + \sum_{t=1}^{r}(\widehat{W}_i-\widetilde{W}_i)x^{t-1}\big).$$

where $\widehat{W}_i$ *and* $\widetilde{W}_i$ *are respectively the* i*th block-column of*

$$\widehat{W} := W\otimes I_M, \quad \widetilde{W} := \frac{1}{2}(I_{MN}+W\otimes I_M), \tag{12}$$

and W *is given in* (9)*;* $\text{prox}_{h_i}^{\beta_i}$ *is the usual proximity operator.*

4. CONVERGENCE RATE ANALYSIS

4.1. Convergence Analysis

We begin analyzing the (rate of) convergence of the proposed methods. Our main results are summarized in the following table. The proofs of various results can be found in [24].

Table 1. Main Convergence Results.

Algorithm	Conv. Condition	Conv. Rate
Network/Gradient		
Static/Exact	$\Omega+\frac{1}{2}M_+(\Xi\otimes I_M)M_+^T\succ\widetilde{P}/2$	$\mathcal{O}(1/r)$
Static/Inexact	$\Omega+\frac{1}{2}M_+(\Xi\otimes I_M)M_+^T\succ\widetilde{P}$	$\mathcal{O}(1/\sqrt{r})$
Random/Exact	$\Omega\succ\widetilde{P}/2$	$\mathcal{O}(1/r)$
Random/Inexact	$\Omega\succ\widetilde{P}$	$\mathcal{O}(1/\sqrt{r})$

Due to its relative simplicity, we first analyze Algorithm 2 in which the exact gradient is available and the network is static.

Theorem 4.1 *Suppose problem* (3) *has a nonempty optimal solution set. Let* $\mathcal{G}^r=\mathcal{G}$ *for all* r*, where* $\mathcal{G}$ *is connected. Then Algorithm 2 converges to a primal-dual optimal solution of problem* (P) *if*

$$2\Omega + M_+(\Xi\otimes I_M)M_+^T = \Upsilon W+\Upsilon\succ\widetilde{P}. \tag{13}$$

where $\widetilde{P} := \text{diag}([P_1,\cdots,P_N])\otimes I_M\in\mathbb{R}^{MN\times MN}$.

A sufficient condition for (13) is that $2\Omega\succ\widetilde{P}$, which is equivalent to $\omega_i > P_i/2$ for all $i\in\mathcal{V}$. Comparing with existing convergence results on proximal-based ADMM such as [25] and [26], our bound for the penalty parameter ω_i is reduced by half. More importantly, no global information is needed at each agent to verify this condition, as it is neither related to the network structure nor any information about the global objective function.

Next we analyze the case where the graph is static and the gradient is stochastic.

Theorem 4.2 *Suppose that problem* (3) *has a nonempty optimal solution set, and the graph is static and connected (with* $\mathcal{G}^r=\mathcal{G}$ *for all* r*). Assume that* $\text{dom}(h)$ *is a bounded set, i.e., there exists a finite* $C>0$ *such that* $d_x := \sup_{\hat{x},\,\tilde{x}\in\text{dom}(h)}\|\hat{x}-\tilde{x}\|\leq C$*. Let* $w:=[x;z;\lambda]$*. Define* $\bar{w}^{r+1} := \frac{1}{r+1}\sum_{t=0}^r w^t$*,*

$$d_z := \sup_{\hat{x},\,\tilde{x}\in\text{dom}(h)}\sqrt{\sum_{ij:e_{ij}\in\mathcal{A}}2\rho_{ij}\|\hat{x}_i-\tilde{x}_j\|^2},$$

where $\{w^t\}$ *are the iterates generated by Algorithm 1. Suppose that* $\eta^{r+1}=\sqrt{r+1},\ \forall\, r$*, and the stepsize matrix satisfies*

$$2\Omega + M_+(I_M\otimes\Xi)M_+^T = \Upsilon W+\Upsilon\succ 2\widetilde{P}. \tag{14}$$

Then at a given iteration r*, we have*

$$\mathbb{E}\left[f(\bar{x}^r)-f(x^*)\right]+\rho\|A\bar{x}^r+B\bar{z}^r\|$$
$$\leq\frac{\sigma^2}{\sqrt{r}}+\frac{d_x^2}{2\sqrt{r}}+\frac{1}{2r}\left(d_z^2+d_\lambda^2(\rho)+\max_i\omega_i d_x^2\right)$$

where $d_\lambda(\rho) := \sup_{\lambda\in\mathcal{B}_\rho}\|\lambda-\lambda^0\|_{\Gamma^{-1}}^2$*,* $\mathcal{B}_\rho=\{\lambda\mid\|\lambda\|\leq\rho\}$*, and* $\rho>0$ *is any finite constant.*

Specializing the above result to the exact gradient case, we can easily show that Algorithm 2 converges with a faster rate of $\mathcal{O}(1/r)$. Further we can analyze the convergence of the DySPCA with random network activation.

Theorem 4.3 *Define d_x, d_z and $\bar{w}^{r+1}$ as in the statement of Theorem 4.2. Suppose $\{w^t\} = \{x^t, z^t, \lambda^t\}$ is a sequence generated by Algorithm 1 (DySPCA), and that*

$$\eta^{r+1} = \sqrt{r+1},\ \forall\, r,\ \textit{and}\quad \Omega \succ \widetilde{P}.$$

Suppose that the graph $\{\mathcal{G}^r\}$ follows Definition 2.1. Then we have

$$\mathbb{E}\left[f(\bar{x}^r) - f(x^*) + \rho\|A\bar{x}^r + B\bar{z}^r\|\right] \le \frac{\sigma^2}{\sqrt{r}} + \frac{d_x^2}{2\sqrt{r}} + \frac{1}{2r}\left(2d_J + d_z^2 + d_\lambda^2(\rho) + \max_i \omega_i d_x^2\right)$$

where $d_\lambda(\rho) := \sup_{\lambda\in\mathcal{B}_\rho} \|\lambda - \lambda^0\|^2_{\Gamma^{-1}}$, $\mathcal{B}_\rho = \{\lambda \mid \|\lambda\| \le \rho\}$, and $\rho > 0$ is any finite constant, and $d_J := \sup_{\lambda\in\mathcal{B}_\rho} J(x^0, z^0, \lambda)$ for some function $J(\cdot)$.

4.2. Connection with Existing Algorithms

We briefly discuss the connection of the proposed DySPCA with a few existing algorithms; See Table 2 for a summary.

Table 2. Comparison with Different Algorithms with DySPGC.

Algorithm	Relation	Special Setting
EXTRA	Special Case	Static, $h \equiv 0$, $W = W^T$, $\widetilde{G} = \nabla g$
DSG	Different x-step	Static, g smooth, $\widetilde{G} = \nabla g$
ICADMM	Special Case	Static, $\widetilde{G} = \nabla g$, g composite
DLM	Special Case	Static, $h \equiv 0$, $W = W^T$, $\widetilde{G} = \nabla g$

First, one can show that the DySPCA is a generalization of the EXTRA algorithm [6]. Consider applying Algorithm 2 to problem (P) with a smooth objective. According to Proposition 3.1, the resulting iterates become (the weight matrices are given in (12))

$$x^{r+1} = x^r + \Upsilon^{-1}\left(\nabla g(x^{r-1}) - \nabla g(x^r)\right) + \widehat{W}x^r - \widetilde{W}x^{r-1}$$

This is precisely the EXTRA algorithm [6], except that here a more general *matrix stepsize* Υ^{-1} is used instead of a *scalar stepsize*. If one insists on having a scalar stepsize $\beta = \beta_i = \beta_j > 0, \forall\, i, j$, then this implies that the weight matrix W must be symmetric. There are at least two ways to construct such single scalar stepsize; see [24]. To compare the convergence result in Theorem 4.1 and that of [6, Theorem 3.3], note that when a single stepsize is used, we have $\Upsilon = \beta I_{MN}$. Therefore a sufficient condition to guarantee the condition given in Theorem 4.1 is that $\beta\lambda_{\min}(I_{MN} + W) > \max_i P_i$. This is precisely the condition set forth in [6, Theorem 3.3].

Second, we can show that when the problem is smooth ($h_i \equiv 0$), and the x-step of the PGC algorithm (8a) is replaced by

$$x^{r+1} = \arg\min \langle \nabla g(x^r), x - x^r\rangle + \frac{1}{2}\|Ax + Bz^r\|^2_\Gamma + \frac{1}{2}\|x - x^r\|^2_\Omega$$

then we recover the DSG iteration (2). Obviously, our convergence analysis does not work for this variant, as the x-update is no longer related to the dual variable λ. Nevertheless, the above observation reveals a fundamental connection between the ADMM-based method and the classical DSG. We can further show that DySPCA generalizes the IC-ADMM proposed in [19] and the PG-EXTRA [7]. Due to space limitations we do not further expand our discussion.

5. NUMERICAL RESULTS

We show some preliminary numerical results of the proposed algorithms by solving a LASSO problem

$$\min_x \tfrac{1}{2}\textstyle\sum_{i=1}^N \|A_i x - b_i\|^2 + \nu\|x\|_1 \qquad (15)$$

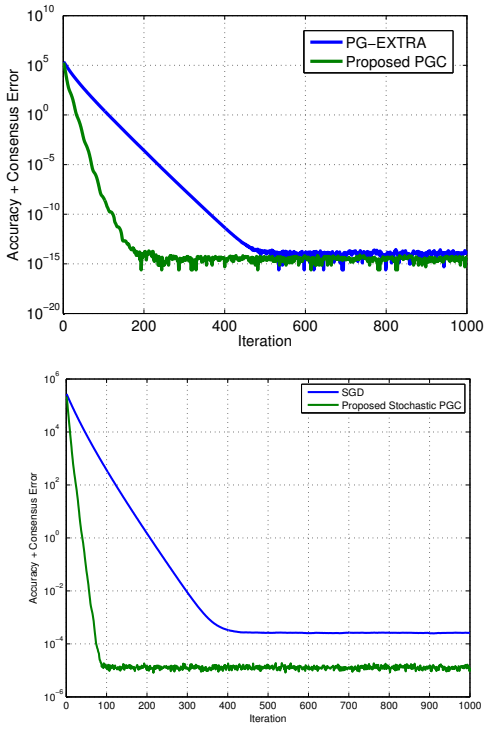

Fig. 1. Top: Comparison of PGC with PG-EXTRA. Bottom: Comparison of SPGC with distributed SGD

where $A_i \in \mathbb{R}^{K\times M}$, $b_i \in \mathbb{R}^K$, where the parameters of the problem are given by: $N = 16$, $M = 100$, $\nu = 0.1$, $K = 200$. Each data matrix A_i is randomly generated as $A_i = L_i \times Q_i$ where $L_i \sim$ Uniform$[0,\ 10]$, and $Q_i \in \mathbb{R}^{K\times M}$ whose entries are iid standard Gaussian random variables; $b_i = A_i c + d_i$ where $c \in \mathbb{R}^N$ is a sparse random vector with 0.01 percent of uniformly distributed non-zero entries; $d_i \in \mathbb{R}^K$ is a vector of iid zero-mean Gaussian random variables with standard deviation 0.01. Note that here $P_i = \|A_i A_i^T\|$, $\forall\, i$. Due to space limitation, we only consider static graphs which are generated according to the method proposed in [27], with a radius parameter set to 0.4.

In our simulation, we compare Algorithm 2 (PGC) with the PG-EXTRA [7], and compare the static version of Algorithm 1 (the stochastic PGC) with the D-SGD [28]. The stepsize for the EXTRA is chosen according to the sufficient condition suggested in [7], and the weight matrix W is the Metropolis constant edge weight matrix. For Algorithm 1 (resp. Algorithm 2), $\omega_i = P_i/2$ (resp. $\omega_i = P_i$) and $\rho_{ij} = 10^{-3}$ for all i, j. For the D-SGD, the stepsize is set as 10^{-5}. The error of the gradient estimate is $\sigma^2 = 0.1$. To measure the progress of different algorithms, we define the following

$$\text{accuracy} = \tfrac{|f(\bar{x}^r) - f^*|}{f^*},\ \text{where } \bar{x}^r = \tfrac{1}{N}\textstyle\sum_{i=1}^N x_i^r,$$
$$\text{consensus error} = \textstyle\sum_{i=1}^N \|x_i^r - \bar{x}^r\|^2.$$

The performance of different algorithms is shown in Fig. 1. Clearly the proposed algorithm outperforms both the EXTRA and the D-SGD. This is expected since compared with the EXTRA, the PGC is able to use larger and more flexible stepsizes, while it is known that the D-SGD with constant stepsize does not converge to the global optimal solution, and D-SGD with diminishing stepsizes has very slow convergence (without convergence rate guarantee).

6. REFERENCES

[1] G. B. Giannakis, Q. Ling, G. Mateos, I. D. Schizas, and H. Zhu, "Proximal splitting methods in signal processing," in *Splitting Methods in Communication and Imaging*. Springer New York, 2015.

[2] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, 2009.

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

[4] A. Nedic, A. Olshevsky, A. Ozdaglar, and J. N. Tsitsiklis, "Distributed subgradient methods and quantization effects," in *IEEE Conference on Decision and Control*, Dec 2008, pp. 4177–4184.

[5] I. Chen, "Fast distributed first-order methods," 2012, Master's thesis, Massachusetts Institute of Technology.

[6] W. Shi, Q. Ling, G. Wu, and W. Yin, "EXTRA: An exact first-order algorithm for decentralized consensus optimization," 2014, online at arXiv:1404.6264.

[7] W. Shi, Q. Ling, G. Wu, and W. Yin, "A proximal gradient algorithm for decentralized nondifferentiable optimization," in *International Conference on Acoustics, Speech and Signal Processing*, 2015.

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

[9] J. C. Duchi, A. Agarwal, and M. J. Wainwright, "Dual averaging for distributed optimization: Convergence analysis and network scaling," *IEEE Transactions on Automatic Control*, vol. 57, no. 3, pp. 592–606, March 2012.

[10] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, Athena-Scientific, second edition, 1999.

[11] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[12] R. Glowinski, *Numerical methods for nonlinear variational problems*, Springer-Verlag, New York, 1984.

[13] 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, no. 1, pp. 350 – 364, 2008.

[14] E. Wei and A. Ozdaglar, "On the O(1/k) convergence of asynchronous distributed alternating direction method of multipliers," 2013, Preprint, available at arXiv:1307.8254.

[15] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "On the linear convergence of the ADMM in decentralized consensus optimization," *IEEE Transactions on Signal Processing*, vol. 62, pp. 1750–1761, 2014.

[16] 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 Transactions on Signal Processing*, vol. 61, no. 10, pp. 2718–2723, May 2013.

[17] H. Zhu, A. Cano, and G.B. Giannakis, "Distributed consensus-based demodulation: algorithms and error analysis," *IEEE Transactions on Wireless Communications*, vol. 9, no. 6, pp. 2044–2054, June 2010.

[18] T. Erseghe, D. Zennaro, E. Dall'Anese, and L. Vangelista, "Fast consensus by the alternating direction multipliers method," *IEEE Transactions on Signal Processing*, vol. 59, no. 11, pp. 5523–5537, Nov 2011.

[19] T.-H. Chang, M. Hong, and X. Wang, "Multi-agent distributed optimization via inexact consensus admm," *IEEE Transactions on Signal Processing*, vol. 63, no. 2, pp. 482–497, Jan 2015.

[20] Q. Ling, W. Shi, G. Wu, and A. Ribeiro, "DLM: Decentralized linearized alternating direction method of multipliers," *IEEE Transactions on Signal Processing*, vol. 63, no. 15, pp. 4051–4064, Aug 2015.

[21] M. Hong, Z.-Q. Luo, and M. Razaviyayn, "Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems," 2014, submitted for publication.

[22] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends in Optimization*, vol. 1, no. 3, pp. 1–112, 2013.

[23] J. Eckstein, "Some saddle-function splitting methods for convex programming," *Optimization Methods and Software*, vol. 4, no. 1, pp. 75–83, 1994.

[24] M. Hong and T.-H. Chang, "Stochastic proximal gradient consensus over time-varying networks," 2015, Technical Report.

[25] X. Gao, B. Jiang, and S. Zhang, "On the information-adaptive variants of the admm: An iteration complexity perspective," 2014, Preprint.

[26] Y. Ouyang, Y. Chen, G. Lan, and Jr. E. Pasiliao, "An accelerated linearized alternating direction method of multipliers," *SIAM Journal on Imaging Sciences*, vol. 8, no. 1, pp. 644–681, 2015.

[27] M. E. Yildiz and A. Scaglione, "Coding with side information for rate-constrained consensus," *IEEE Trans. Signal Process.*, vol. 56, no. 8, pp. 3753–3764, 2008.

[28] S. Sundlhar Ram, A. Nedić, and V. V. Veeravalli, "Distributed stochastic subgradeint projection algorithms for convex optimization," *J. Optim. Theory Appl.*, vol. 147, pp. 516–545, 2010.