Decentralized Double Stochastic Averaging Gradient

Aryan Mokhtari and Alejandro Ribeiro
Department of Electrical and Systems Engineering, University of Pennsylvania

Abstract—**This paper considers convex optimization problems where nodes of a network have access to summands of a global objective function. Each of these local objectives is further assumed to be an average of a finite set of functions. The motivation for this setup is to solve large scale machine learning problems where elements of the training set are distributed to multiple computational elements. The decentralized double stochastic averaging gradient (DSA) algorithm is proposed as a solution alternative that relies on: (i) The use of local stochastic averaging gradients instead of local full gradients. (ii) Determination of descent steps as differences of consecutive stochastic averaging gradients. The algorithm is shown to approach the optimal argument at a linear rate. This is in contrast to all other available methods for distributed stochastic optimization that converge at sublinear rates. Numerical experiments verify linear convergence of DSA and illustrate its advantages relative to these other alternatives.**

I. Introduction

Consider a variable $\mathbf{x} \in \mathbb{R}^p$ and a connected network of size N where each node n has access to a local objective function $f_n : \mathbb{R}^p \to \mathbb{R}$. The local objective function $f_n(\mathbf{x})$ is defined as the average of q_n functions $f_{n,i}(\mathbf{x})$ that can be individually evaluated at node n. Agents cooperate to solve the global optimization

$$\tilde{\mathbf{x}}^* := \operatorname*{argmin}_{\mathbf{x}} \sum_{n=1}^{N} f_n(\mathbf{x}) = \operatorname*{argmin}_{\mathbf{x}} \sum_{n=1}^{N} \frac{1}{q_n} \sum_{i=1}^{q_n} f_{n,i}(\mathbf{x}). \qquad (1)$$

The formulation in (1) models large scale machine learning problems where elements of the training set are distributed to multiple computational elements and $\tilde{\mathbf{x}}^*$ represents an optimal classifier [1], [2]. Analogous formulations are also of interest in decentralized control [3], [4] and sensor networks [5]–[7].

Our interest here is in solving (1) with a method that is distributed – nodes operate on their local functions and communicate with neighbors – and stochastic – nodes utilize only one out of the q_n functions $f_{n,i}$ to determine a descent direction. Distributed methods germane to this paper are decentralized gradient descent (DGD) and its variants [8]–[11], as well as the exact first order algorithm (EXTRA) [12]. An issue with the former that is solved by the latter is the lack of linear convergence rate guarantees that EXTRA achieves by using iterations that rely on information of two consecutive steps. Stochastic optimization methods related to the proposal in this paper are stochastic gradient descent [13]–[16] and stochastic averaging gradient methods [17], [18]. As in the case of distributed optimization, the former have sublinear convergence rates but the latter have linear convergence rates. They achieve these linear rates by using incremental gradients to reduce the stochastic gradient noise. This reduction follows from a memory trade that permits maintaining an average of past gradients in which only one term is updated per iteration.

The contribution of this paper is to develop the decentralized double stochastic averaging gradient (DSA) method, a novel decentralized stochastic algorithm for solving (1). The method exploits a new interpretation of EXTRA as a saddle point method (Section II) and uses stochastic averaging gradients in lieu of gradients (Section III). The proposed method converges linearly to the optimal argument in expectation (Section IV). This is in contrast to all other distributed stochastic methods to solve (1) that converge at sublinear rates. Numerical results verify that DSA is the only stochastic decentralized algorithm with linear convergence rate (Section V). Proofs of results in this paper are available in [19].

II. Decentralized double gradient descent

Consider a connected network that contains N nodes such that each node n can only communicate with nodes in its neighborhood $\mathcal{N}_n$. Define $\mathbf{x}_n \in \mathbb{R}^p$ as a local copy of the variable $\mathbf{x}$ that is kept at node n. In decentralized optimization, nodes try to minimize their local functions $f_n(\mathbf{x}_n)$ while ensuring that their local variables $\mathbf{x}_n$ coincide with the variables $\mathbf{x}_m$ of all neighbors $m \in \mathcal{N}_n$ – which, given that the network is connected, ensures that the variables $\mathbf{x}_n$ of all nodes are the same and renders the problem equivalent to (1). DGD is a well known method for decentralized optimization that relies on the introduction of nonnegative weights $w_{ij} \geq 0$ that are not null if and only if $m = n$ or if $m \in \mathcal{N}_n$. Letting $t \in \mathbb{N}$ be a discrete time index and α a given stepsize, DGD is defined by the recursion

$$\mathbf{x}_n^{t+1} = \sum_{m=1}^{N} w_{nm}\mathbf{x}_m^t - \alpha \nabla f_n(\mathbf{x}_n^t), \qquad n = 1, \ldots, N. \qquad (2)$$

Since $w_{nm} = 0$ when $m \neq n$ and $m \notin \mathcal{N}_n$, it follows from (2) that node n updates $\mathbf{x}_n$ by performing an average over the variables $\mathbf{x}_m^t$ of its neighbors $m \in \mathcal{N}_n$ and its own $\mathbf{x}_n^t$, followed by descent through the negative local gradient $-\nabla f_n(\mathbf{x}_n^t)$. If a constant stepsize is used, DGD iterates $\mathbf{x}_n^t$ approach a neighborhood of the optimal argument $\tilde{\mathbf{x}}^*$ of (1) but don't converge exactly. To achieve exact convergence diminishing stepsizes are used but the resulting convergence rate is sublinear [8].

EXTRA resolves either of these issues by mixing two consecutive DGD iterations with different weight matrices and opposite signs. To be precise, introduce a second set of weights $\tilde{w}_{nm}$ with the same properties as the weights w_{nm} and define EXTRA through the recursion

$$\begin{aligned}\mathbf{x}_n^{t+1} = \mathbf{x}_n^t &+ \sum_{m=1}^{N} w_{nm}\mathbf{x}_m^t - \sum_{m=1}^{N} \tilde{w}_{nm}\mathbf{x}_m^{t-1} \qquad (3)\\ &- \alpha\left[\nabla f_n(\mathbf{x}_n^t) - \nabla f_n(\mathbf{x}_n^{t-1})\right], \quad n = 1, \ldots, N.\end{aligned}$$

Observe that (3) is well defined for $t > 0$. For $t = 0$ we utilize the regular DGD iteration in (2). In the nomenclature of this paper we say that EXTRA performs a decentralized double gradient descent step because it operates in decentralized manner while utilizing a difference of two gradients as descent direction. Minor modification as it is, the use of this gradient difference in lieu of simple gradients endows extra with exact linear convergence to the optimal argument $\tilde{\mathbf{x}}^*$ [12].

To understand the rationality behind the EXTRA update, we define matrices and vectors to rewrite updates in (3) for different nodes as a single equation. To do so, define the vector $\mathbf{x} := [\mathbf{x}_1; \dots; \mathbf{x}_N] \in \mathbb{R}^{Np}$ which concatenates the local iterates, and the aggregate function $f : \mathbb{R}^{Np} \to \mathbb{R}$ as

$$f(\mathbf{x}) = f(\mathbf{x}_1, \dots, \mathbf{x}_N) := \sum_{n=1}^{N} f_n(\mathbf{x}_n). \tag{4}$$

Moreover, Consider the matrices $\mathbf{W} \in \mathbb{R}^{N \times N}$ and $\tilde{\mathbf{W}} \in \mathbb{R}^{N \times N}$ formed by components $[w_{nm}]$ and $[\tilde{w}_{nm}]$, respectively. Define the matrices $\mathbf{Z} := \mathbf{W} \otimes \mathbf{I} \in \mathbb{R}^{Np \times Np}$ and $\tilde{\mathbf{Z}} := \tilde{\mathbf{W}} \otimes \mathbf{I} \in \mathbb{R}^{Np \times Np}$ as the Kronecker products of the weight matrices $\mathbf{W} \in \mathbb{R}^{N \times N}$ and $\tilde{\mathbf{W}} \in \mathbb{R}^{N \times N}$ by the identity matrix $\mathbf{I} \in \mathbb{R}^{p \times p}$, respectively. Considering these definitions, we can rewrite the EXTRA's update for $t > 0$ in (3) as

$$\mathbf{x}^{t+1} = (\mathbf{I} + \mathbf{Z})\mathbf{x}^t - \tilde{\mathbf{Z}}\mathbf{x}^{t-1} - \alpha[\nabla f(\mathbf{x}^t) - \nabla f(\mathbf{x}^{t-1})], \tag{5}$$

and the initial step as

$$\mathbf{x}^1 = \mathbf{Z}\mathbf{x}^0 - \alpha \nabla f(\mathbf{x}^0). \tag{6}$$

By summing up the updates in (5) and (6) from step 0 to t and using the telescopic cancellation we obtain that

$$\mathbf{x}^{t+1} = \tilde{\mathbf{Z}}\mathbf{x}^t - \alpha \nabla f(\mathbf{x}^t) - \sum_{s=0}^{t} (\tilde{\mathbf{Z}} - \mathbf{Z})\mathbf{x}^s. \tag{7}$$

We introduce a primal-dual interpretation of the update in (7) by defining the sequence of vectors $\mathbf{v}^t = \sum_{s=0}^{t} (\tilde{\mathbf{Z}} - \mathbf{Z})^{1/2}\mathbf{x}^s$ as the accumulation of variables dissimilarities in different nodes over time. Note that if components of the vector $\mathbf{x}^s$ are equal to each other, i.e., $\mathbf{x}_1^s = \dots = \mathbf{x}_N^s$, the corresponding term of the sum in the definition of vector $\mathbf{v}^t$ is null, i.e. $(\tilde{\mathbf{Z}} - \mathbf{Z})^{1/2}\mathbf{x}^s = \mathbf{0}$. Considering the definition of $\mathbf{v}^t$ we can rewrite (7) as

$$\mathbf{x}^{t+1} = \mathbf{x}^t - \alpha \left[\nabla f(\mathbf{x}^t) + \frac{1}{\alpha}(\mathbf{I} - \tilde{\mathbf{Z}})\mathbf{x}^t + \frac{1}{\alpha}(\tilde{\mathbf{Z}} - \mathbf{Z})^{1/2}\mathbf{v}^t \right]. \tag{8}$$

Further, based on the definition of sequence $\mathbf{v}^t = \sum_{s=0}^{t} (\tilde{\mathbf{Z}} - \mathbf{Z})^{1/2}\mathbf{x}^s$ we can write $\mathbf{v}^{t+1}$ as

$$\mathbf{v}^{t+1} = \mathbf{v}^t + \alpha \left[\frac{1}{\alpha}(\tilde{\mathbf{Z}} - \mathbf{Z})^{1/2}\mathbf{x}^{t+1} \right]. \tag{9}$$

Consider $\mathbf{x}$ as the primal variable and $\mathbf{v}$ as the dual variable. Then, the EXTRA update is equivalent to a saddle point method with stepsize α for solving the Lagrangian

$$\mathcal{L}(\mathbf{x}, \mathbf{v}) = f(\mathbf{x}) + \frac{1}{\alpha}\mathbf{v}^T(\tilde{\mathbf{Z}} - \mathbf{Z})^{1/2}\mathbf{x} + \frac{1}{2\alpha}\mathbf{x}^T(\mathbf{I} - \tilde{\mathbf{Z}})\mathbf{x}, \tag{10}$$

where the the actual Lagrangian is augmented by the quadratic term $(1/2\alpha)\mathbf{x}^T(\mathbf{I} - \tilde{\mathbf{Z}})\mathbf{x}$. Observe that the optimization problem with the augmented Lagrangian in (10) is

$$\min_{\mathbf{x}} \; f(\mathbf{x}) \qquad \text{s.t. } \frac{1}{\alpha}(\tilde{\mathbf{Z}} - \mathbf{Z})^{1/2}\mathbf{x} = \mathbf{0}. \tag{11}$$

Observing that $\text{null}((\tilde{\mathbf{Z}} - \mathbf{Z})^{1/2}) = \text{null}(\tilde{\mathbf{Z}} - \mathbf{Z}) = \text{span}\{\mathbf{1}_N \otimes \mathbf{I}_p\}$, the constraint in (11) is equivalent to $\mathbf{x}_1 = \dots = \mathbf{x}_N$. Moreover, the definition of function $f(\mathbf{x})$ in (4) shows that the objectives of problems (11) and (1) are also identical. Hence, EXTRA is a saddle point method that solves (11) which is equivalent to (1). Considering the exact and linear convergence of saddle point methods, the convergence properties of EXTRA are justified.

III. Decentralized Double stochastic averaging gradient

Recall the definitions of the local functions $f_n(\mathbf{x}_n)$ and the instantaneous local functions $f_{n,i}(\mathbf{x}_n)$ available at node n. To implement EXTRA as in (3) each node computes the *full gradient* of its local objective function $\nabla f_n(\mathbf{x}_n) = (1/q_n)\sum_{i=1}^{q_n} \nabla f_{n,i}(\mathbf{x}_n)$ which is computationally expensive when the number of instantaneous functions q_n is extremely large. To resolve this issue the local objective gradients can be substituted by their stochastic approximations. The simplest approach for approximating the local objective functions gradient $\nabla f_n(\mathbf{x}_n) = (1/q_n)\sum_{i=1}^{q_n} \nabla f_{n,i}(\mathbf{x}_n)$ is choosing an instantaneous function $f_{n,i}(\mathbf{x}_n)$ randomly and using its gradient $\nabla f_{n,i}(\mathbf{x}_n)$ as an unbiased estimate of the gradient $\nabla f_n(\mathbf{x}_n) = (1/q_n)\sum_{i=1}^{q_n} \nabla f_{n,i}(\mathbf{x}_n)$. To be more precise, define vector $\boldsymbol{\theta} = [\theta_1 : \dots ; \theta_N] \in \{1, \dots, q_1\} \times \dots \times \{1, \dots, q_N\}$ as a random vector where each component $\theta_n \in \{1, \dots, q_n\}$ determines the associated instantaneous function $f_{n,\theta_n}(\mathbf{x}_n)$ that node n uses for gradient approximation. To be more precise, node n in lieu of computing the local function gradient $\nabla f_n(\mathbf{x})$ for updating the variable $\mathbf{x}_n$, approximates it by $\nabla f_{n,\theta_n}(\mathbf{x}_n)$. However, stochastic gradients lead to an algorithm with lower computation complexity, the noise of gradient approximation avoids exact convergence with constant stepsize as shown for stochastic gradient descent in centralized optimization. We study this observation in Section V.

To overcome the noise of gradient approximation we use the idea of unbiased stochastic averaging gradient as introduced in [18]. We introduce the auxiliary vectors $\boldsymbol{\phi}_{n,i} \in \mathbb{R}^p$ corresponding to i-th instantaneous function of node n which keeps track of the iterate $\mathbf{x}_n$ for the last step that i-th instantaneous function $f_{n,i}$ is chosen at node n. To be more precise, if the index identifier at time t for node n is $\theta_n^t = i$ then the corresponding auxiliary vector $\boldsymbol{\phi}_{n,i}^t$ is updated as $\boldsymbol{\phi}_{n,i}^{t+1} = \mathbf{x}_n^t$ and its corresponding instantaneous function gradient $\nabla f_{n,i}(\boldsymbol{\phi}_{n,i}^t)$ which is stored in a memory is replaced by $\nabla f_{n,i}(\mathbf{x}_n^t)$. All the other auxiliary vectors $\boldsymbol{\phi}_{n,j}^t$ for $j \neq i$ and their corresponding instantaneous gradients remain unchanged, i.e. $\boldsymbol{\phi}_{n,j}^{t+1} = \boldsymbol{\phi}_{n,j}^t$ and $\nabla f_{n,j}(\boldsymbol{\phi}_{n,j}^{t+1}) = \nabla f_{n,j}(\boldsymbol{\phi}_{n,j}^t)$. By storing the auxiliary variables gradients $\nabla f_{n,i}(\boldsymbol{\phi}_{n,i}^t)$, we can define an unbiased estimate of the local gradient $\nabla f_n(\mathbf{x}_n^t)$ as

$$\hat{\mathbf{g}}_n^t := \nabla f_{n,\theta_n^t}(\mathbf{x}_n^t) - \nabla f_{n,\theta_n^t}(\boldsymbol{\phi}_{n,\theta_n^t}^t) + \frac{1}{q_n}\sum_{i=1}^{q_n} \nabla f_{n,i}(\boldsymbol{\phi}_{n,i}^t). \tag{12}$$

Notice that the stochastic approximation $\hat{\mathbf{g}}_n^t$ is an unbiased estimate of the local gradient $\nabla f_n(\mathbf{x}_n^t)$, i.e., $\mathbb{E}\left[\hat{\mathbf{g}}_n^t \mid \mathcal{F}^t\right] = \nabla f_n(\mathbf{x}_n^t)$.

The proposed stochastic averaging gradient in (12) vanishes the noise of gradient approximation. To be more precise, as time progresses the auxiliary variables $\boldsymbol{\phi}_{n,i}^t$ approach to a neighborhood of the optimal variable $\tilde{\mathbf{x}}^*$, since they all get updated over time with a high probability. Therefore, roughly speaking we can write $\boldsymbol{\phi}_{n,i}^t \approx \mathbf{x}_n^t \approx \tilde{\mathbf{x}}^*$. This property implies that the stochastic gradient in (12) can be approximated by $\hat{\mathbf{g}}_n^t \approx (1/q_n)\sum_{i=1}^{q_n} \nabla f_{n,i}(\boldsymbol{\phi}_{n,i}^t) \approx \nabla f_n(\mathbf{x}_n^t)$. Therefore, the advantage of using stochastic approximation in (12) is the fact that the noise of stochastic gradients is diminishing when the sequence is close to convergence, while for the naive approximation $\nabla f_{n,\theta_n^t}(\mathbf{x}_n^t)$ the noise of stochastic approximation never vanishes.

We introduce Decentralized Double stochastic averaging gradient (DSA) as a stochastic version of EXTRA that approximates the local gradients by their stochastic averaging approximations

Algorithm 1 DSA algorithm at node n

```
Require: Vector $\mathbf{x}_n^0$ and stored gradients $\nabla f_{n,i}(\boldsymbol{\phi}_{n,i}^0)$ with $\boldsymbol{\phi}_{n,i}^0 = \mathbf{x}_n^0$.
 1: for $t = 0, 1, 2, \ldots$ do
 2:     Exchange variable $\mathbf{x}_n^t$ with neighboring nodes $m \in \mathcal{N}_n$.
 3:     Choose $\theta_n^t$ uniformly random from the set $\{1, \ldots, q_n\}$.
 4:     Compute and store stochastic averaging gradient
        $\hat{\mathbf{g}}_n^t = \nabla f_{n,\theta_n^t}(\mathbf{x}_n^t) - \nabla f_{n,\theta_n^t}(\boldsymbol{\phi}_{n,\theta_n^t}^t) + \frac{1}{q_n}\sum_{i=1}^{q_n} \nabla f_{n,i}(\boldsymbol{\phi}_{n,i}^t)$
 5:     Set $\boldsymbol{\phi}_{n,\theta_n^t}^{t+1} = \mathbf{x}_n^t$ and store gradient $\nabla f_{n,\theta_n^t}(\boldsymbol{\phi}_{n,\theta_n^t}^{t+1})$ in the
        table replacing $\nabla f_{n,\theta_n^t}(\boldsymbol{\phi}_{n,\theta_n^t}^t)$. Other vectors of the table remain
        unchanged, i.e. $\nabla f_{n,j}(\boldsymbol{\phi}_{n,j}^{t+1}) = \nabla f_{n,j}(\boldsymbol{\phi}_{n,j}^t)$ for $j \neq \theta_n^t$.
 6:     Update primal variable $\mathbf{x}_n^t$ as
 7:     if $t = 0$ then
 8:         $\mathbf{x}_n^{t+1} = \sum_{n=1}^{N} w_{nm}\mathbf{x}_n^{t+1} - \alpha\hat{\mathbf{g}}_n^t$.
 9:     else
10:         $\mathbf{x}_n^{t+1} = \mathbf{x}_n^t + \sum_{n=1}^{N} w_{nm}\mathbf{x}_n^t - \sum_{n=1}^{N} \hat{w}_{nm}\mathbf{x}_n^{t-1} - \alpha\left[\hat{\mathbf{g}}_n^t - \hat{\mathbf{g}}_n^{t-1}\right]$.
11:     end if
12: end for
```

as introduced in (12). The DSA update for $t > 0$ is given by

$$\mathbf{x}_n^{t+1} = \mathbf{x}_n^t + \sum_{m=1}^{N} w_{nm}\mathbf{x}_m^t - \sum_{m=1}^{N} \tilde{w}_{nm}\mathbf{x}_m^{t-1} - \alpha\left[\hat{\mathbf{g}}_n^t - \hat{\mathbf{g}}_n^{t-1}\right], \tag{13}$$

and the initial step is defined as

$$\mathbf{x}_n^1 = \sum_{m=1}^{N} w_{nm}\mathbf{x}_m^0 - \alpha\ \hat{\mathbf{g}}_n^0. \tag{14}$$

To write the DSA update for all the nodes in one equation, define the vector $\hat{\mathbf{g}}^t := [\hat{\mathbf{g}}_1^t; \ldots; \hat{\mathbf{g}}_N^t] \in \mathbb{R}^{Np}$ which contains all the local stochastic averaging gradients at step t. Considering this definition the updates for steps $t > 0$ in (13) can be simplified as

$$\mathbf{x}^{t+1} = (\mathbf{I} + \mathbf{Z})\mathbf{x}^t - \tilde{\mathbf{Z}}\mathbf{x}^{t-1} - \alpha\left[\hat{\mathbf{g}}^t - \hat{\mathbf{g}}^{t-1}\right], \tag{15}$$

and the initial updates in (14) are equivalent to

$$\mathbf{x}^1 = \mathbf{Z}\mathbf{x}^0 - \alpha\,\hat{\mathbf{g}}^0. \tag{16}$$

Comparing the DSA updates in (15) and (16) with EXTRA steps in (5) and (6) shows that DSA is different from EXTRA in using stochastic averaging gradients $\hat{\mathbf{g}}^t$ in lieu of full gradients $\nabla f(\mathbf{x}^t)$. Recall that EXTRA is a saddle point method for solving (11). Therefore, DSA is a stochastic saddle point method that solves problem (11) where the primal variable $\mathbf{x}^t$ is updated as

$$\mathbf{x}^{t+1} = \mathbf{x}^t - \alpha\hat{\mathbf{g}}^t - (\mathbf{I} - \tilde{\mathbf{Z}})\mathbf{x}^t - (\tilde{\mathbf{Z}} - \mathbf{Z})^{1/2}\mathbf{v}^t, \tag{17}$$

and the dual variable $\mathbf{v}^t$ is updated as

$$\mathbf{v}^{t+1} = \mathbf{v}^t + (\tilde{\mathbf{Z}} - \mathbf{Z})^{1/2}\mathbf{x}^{t+1}. \tag{18}$$

Notice that the initial primal variable $\mathbf{x}^0 \in \mathbb{R}^{Np}$ is an arbitrary vector, while according to the definition $\mathbf{v}^t = \sum_{s=0}^{t}(\tilde{\mathbf{Z}} - \mathbf{Z})^{1/2}\mathbf{x}^s$ the initial dual vector is set as $\mathbf{v}^0 = (\tilde{\mathbf{Z}} - \mathbf{Z})^{1/2}\mathbf{x}^0$. To implement DSA we use the update in (15) instead of using the primal-dual updates in (17) and (18). The latter requires exchange of the both primal $\mathbf{x}_n^t$ and dual $\mathbf{v}_n^t$ variables, while for the former only exchange of the primal variables $\mathbf{x}_n^t$ is required.

The DSA algorithm is summarized in Algorithm 1. The update of DSA for $t = 0$ and $t > 0$ are implemented in Steps 8 and 10, respectively. Steps 8 and 10 require access to the local iterates $\mathbf{x}_m^t$ of the neighboring nodes $m \in \mathcal{N}_n$ which are collected in Step 2. Further, implementation of the DSA update requires stochastic gradients $\hat{\mathbf{g}}_n^{t-1}$ and $\hat{\mathbf{g}}_n^t$ which are computed in Step 5 of iterations $t-1$ and t, respectively. In Step 3 the index θ_n^t is chosen randomly to distinguish the instantaneous function f_{n,θ_n^t} that we use its gradients at points $\mathbf{x}_n^t$ and $\boldsymbol{\phi}_{n,\theta_n^t}^t$ for computing the stochastic averaging gradient in Step 4. The table of auxiliary variables gradients is updated in Step 5 by replacing $\nabla f_{n,\theta_n^t}(\boldsymbol{\phi}_{n,\theta_n^t}^t)$ by $\nabla f_{n,\theta_n^t}(\mathbf{x}_n^t)$, while the other vectors remain unchanged.

IV. Convergence analysis

Our goal here is to show that as time progresses the sequence of iterates $\mathbf{x}^t$ approaches the optimal argument $\mathbf{x}^*$. In proving this result for the DSA algorithm we make the following assumptions.

Assumption 1. The wight matrices $\mathbf{W}$ and $\tilde{\mathbf{W}}$ satisfy

(a) If $m \neq n$ and $m \notin \mathcal{N}_n$, then $w_{nm} = \tilde{w}_{nm} = 0$.
(b) $\mathbf{W} = \mathbf{W}^T$ and $\tilde{\mathbf{W}} = \tilde{\mathbf{W}}^T$.
(c) null$\{\tilde{\mathbf{W}} - \mathbf{W}\} =$ span$\{\mathbf{1}\}$ and null$\{\mathbf{I} - \tilde{\mathbf{W}}\} \supseteq$ span$\{\mathbf{1}\}$.
(d) $\mathbf{0} \prec \tilde{\mathbf{W}}$ and $\mathbf{W} \preceq \tilde{\mathbf{W}} \preceq (\mathbf{I} + \mathbf{W})/2$.

Assumption 2. The instantaneous local functions $f_{n,i}(\mathbf{x}_n)$ are differentiable and strongly convex with parameter μ.

Assumption 3. The instantaneous local functions gradients $\nabla f_{n,i}$ are Lipschitz continuous with parameter L,

$$\|\nabla f_{n,i}(\mathbf{a}) - \nabla f_{n,i}(\mathbf{b})\| \leq L\ \|\mathbf{a} - \mathbf{b}\| \quad \mathbf{a}, \mathbf{b} \in \mathbb{R}^p. \tag{19}$$

The conditions imposed by Assumption 1(a) on the entries of the weight matrices $\mathbf{W}$ and $\tilde{\mathbf{W}}$ imply that nodes only have access to the local and neighboring information. Further, we assume the assigned weights are symmetric for both weight matrices $\tilde{\mathbf{W}}$ and $\mathbf{W}$ as mentioned in Assumption 1(b). Conditions on the spectral properties of matrices $\mathbf{W}$ and $\tilde{\mathbf{W}}$ in Assumptions 1(c) and 1(d) imply that null$\{\mathbf{I} - \mathbf{W}\} =$ span$\{\mathbf{1}\}$ – see Proposition 2.1 [12]. Assumption 2 implies that the local functions $f_n(\mathbf{x}_n)$ and the global cost function $f(\mathbf{x}) = \sum_{n=1}^{N} f_n(\mathbf{x}_n)$ are strongly convex with parameter μ. Likewise, the Lipschitz continuity of the local instantaneous gradients $\nabla f_{n,i}(\mathbf{x}_n)$ enforces Lipschitz continuity of the local functions gradients $\nabla f_n(\mathbf{x}_n)$ and the aggregate function gradients $\nabla f(\mathbf{x})$.

Define $0 < \gamma$ and $\Gamma < \infty$ as the smallest and largest eigenvalues of $\tilde{\mathbf{Z}}$, respectively. Likewise, define $\tilde{\gamma}$ as the smallest non-zero eigenvalue of the matrix $\tilde{\mathbf{Z}} - \mathbf{Z}$ and $\tilde{\Gamma}$ as the largest eigenvalue of $\tilde{\mathbf{Z}} - \mathbf{Z}$. Further, define vectors $\mathbf{u}^*, \mathbf{u}^t \in \mathbb{R}^{2Np}$ and matrix $\mathbf{G} \in \mathbb{R}^{2Np \times 2Np}$ as

$$\mathbf{u}^* := \begin{bmatrix}\mathbf{x}^* \\ \mathbf{v}^*\end{bmatrix}, \quad \mathbf{u}^t := \begin{bmatrix}\mathbf{x}^t \\ \mathbf{v}^t\end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix}\tilde{\mathbf{Z}} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}\end{bmatrix}. \tag{20}$$

The vector $\mathbf{u}^* \in \mathbb{R}^{2Np}$ concatenates the optimal primal and dual variables and the vector $\mathbf{u}^t \in \mathbb{R}^{2Np}$ contains primal and dual iterates at step t. Further, $\mathbf{G} \in \mathbb{R}^{2Np \times 2Np}$ is a block diagonal positive definite matrix. We study the convergence properties of the weighted norm $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2$ which is equivalent to $(\mathbf{u}^t - \mathbf{u}^*)^T\mathbf{G}(\mathbf{u}^t - \mathbf{u}^*)$. Our goal is to show that the sequence $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2$ converges linearly to null. To do this we show linear convergence of a Lyapunov function of the sequence $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2$.

The Lyapunov function is defined as $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2 + cp^t$ where

$$p^t := \sum_{n=1}^{N} \left[\frac{1}{q_n} \sum_{i=1}^{q_n} f_{n,i}(\boldsymbol{\phi}_{n,i}^t) - f_n(\tilde{\mathbf{x}}^*) - \frac{1}{q_n} \sum_{i=1}^{q_n} \nabla f_{n,i}(\tilde{\mathbf{x}}^*)^T (\boldsymbol{\phi}_{n,i}^t - \tilde{\mathbf{x}}^*) \right], \quad (21)$$

and $c > 0$ is a positive constant. Notice that based on the strong convexity of the local instantaneous functions $f_{n,i}$, each term $f_{n,i}(\boldsymbol{\phi}_{n,i}^t) - f_{n,i}(\tilde{\mathbf{x}}^*) - \nabla f_{n,i}(\tilde{\mathbf{x}}^*)^T (\boldsymbol{\phi}_{n,i}^t - \tilde{\mathbf{x}}^*)$ is positive and as a result the sequence p^t defined in (21) is always positive.

To prove linear convergence of the sequence $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2 + cp^t$ we first show an upper bound for the expected error $\mathbb{E}\left[\|\mathbf{u}^{t+1} - \mathbf{u}^*\|_{\mathbf{G}}^2 \mid \mathcal{F}^t\right]$ in terms of $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2$ and some parameters that capture the optimality gap.

Lemma 1. *Consider the DSA algorithm as defined in* (12)-(18). *Further, recall the definitions of* p^t *in* (21) *and* $\mathbf{u}^t$, $\mathbf{u}^*$, *and* $\mathbf{G}$ *in* (20). *If Assumptions 1-3 hold true, then for any positive constants* $\eta, \rho > 0$ *we can write*

$$\begin{aligned}
&\mathbb{E}\left[\|\mathbf{u}^{t+1} - \mathbf{u}^*\|_{\mathbf{G}}^2 \mid \mathcal{F}^t\right] \leq \|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2 \qquad (22)\\
&- \mathbb{E}\left[\|\mathbf{x}^{t+1} - \mathbf{x}^t\|_{\mathbf{Z}-\alpha(\eta+\rho)\mathbf{I}}^2 \,|\, \mathcal{F}^t\right] - 2\mathbb{E}\left[\|\mathbf{x}^{t+1} - \mathbf{x}^*\|_{\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}}}^2 \,|\, \mathcal{F}^t\right]\\
&- \mathbb{E}\left[\|\mathbf{v}^{t+1} - \mathbf{v}^t\|^2 \mid \mathcal{F}^t\right] + \frac{4\alpha L}{\rho} p^t - \alpha C_0 \left\|\mathbf{x}^t - \mathbf{x}^*\right\|^2,
\end{aligned}$$

where $C_0 = (2\mu^2/L) - (L^2/\eta) - (2(L^2 - \mu^2))/\rho$.

Likewise, we provide an upper bound for the other part of the Lyapunov function at time $t+1$ which is p^{t+1} in terms of p^t and some parameters that capture optimality gap. This bound is studied in the following lemma.

Lemma 2. *Consider the DSA algorithm as defined in* (12)-(18) *and the definition of* p^t *in* (21). *Further, define* $q_{\min}$ *and* $q_{\max}$ *as the smallest and largest values for the size of instantaneous functions at a node, respectively. If Assumptions 1-3 hold true, then for all* $t > 0$

$$\mathbb{E}\left[p^{t+1} \mid \mathcal{F}^t\right] \leq \left(1 - \frac{1}{q_{\max}}\right) p^t + \frac{L}{2q_{\min}} \left\|\mathbf{x}^t - \mathbf{x}^*\right\|^2. \quad (23)$$

Combining the results in Lemmata 1 and 2 we can show that the expected Lyapunov function $\mathbb{E}\left[\|\mathbf{u}^{t+1} - \mathbf{u}^*\|_{\mathbf{G}}^2 + c\, p^{t+1} \mid \mathcal{F}^t\right]$ is strictly smaller than its previous value $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2 + c\, p^t$.

Theorem 1. *Consider the DSA algorithm as defined in* (12)-(18). *Further, recall the definitions of* p^t *in* (21) *and* $\mathbf{u}^t$, $\mathbf{u}^*$, *and* $\mathbf{G}$ *in* (20). *Moreover, consider the results in* (22) *and* (23). *If Assumptions 1-3 hold true and the stepsize* α *and the parameter* c *are chosen properly then there exits* $0 < \delta < 1$ *such that*

$$\mathbb{E}\left[\|\mathbf{u}^{t+1} - \mathbf{u}^*\|_{\mathbf{G}}^2 + c\, p^{t+1} \mid \mathcal{F}^t\right] \leq (1-\delta) \left(\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2 + c\, p^t\right). \quad (24)$$

The conditions on α and c, and the explicit expression of δ are provided in [19]. The inequality in (24) shows that the expected value of the sequence $\|\mathbf{u}^{t+1} - \mathbf{u}^*\|_{\mathbf{G}}^2 + cp^{t+1}$ given the observations until step t is strictly smaller than the previous iterate $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2 + cp^t$. By taking the expected value with respect to the initial field $\mathbb{E}\left[. \mid \mathcal{F}^0\right] = \mathbb{E}\left[.\right]$ and applying the implied inequality recursively we obtain that

$$\mathbb{E}\left[\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2 + c\, p^t\right] \leq (1-\delta)^t \left(\|\mathbf{u}^0 - \mathbf{u}^*\|_{\mathbf{G}}^2 + c\, p^0\right). \quad (25)$$

According to (25), the sequence $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2 + c\, p^t$ converges linearly to null in expectation. Notice that the norm $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2$ is equal to $\|\mathbf{x}^t - \mathbf{x}^*\|_{\tilde{\mathbf{Z}}}^2 + \|\mathbf{v}^t - \mathbf{v}^*\|^2$. Hence, the inequality $\|\mathbf{x}^t - \mathbf{x}^*\|_{\tilde{\mathbf{Z}}}^2 \leq \|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2$ holds. Moreover, the sequence p^t is always non-negative. Considering these two observations the inequality $\|\mathbf{x}^t - \mathbf{x}^*\|_{\tilde{\mathbf{Z}}}^2 \leq \|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2 + c\, p^t$ holds true. Considering this inequality and the expression in (25), and observing that the term $\|\mathbf{x}^t - \mathbf{x}^*\|_{\tilde{\mathbf{Z}}}^2$ is lower bounded by $\gamma\|\mathbf{x}^t - \mathbf{x}^*\|^2$, we can write the following corollary.

Corollary 1. Consider the DSA algorithm as defined in (12)-(18). Recall the definitions of p^t in (21) and $\mathbf{u}^t$, $\mathbf{u}^*$, and $\mathbf{G}$ in (20). Further, recall γ as the smallest eigenvalue of $\tilde{\mathbf{Z}}$. If Assumptions 1-3 hold true, then there exits a constant $0 < \delta < 1$ such that

$$\mathbb{E}\left[\|\mathbf{x}^t - \mathbf{x}^*\|^2\right] \leq (1-\delta)^t \, \frac{\left(\|\mathbf{u}^0 - \mathbf{u}^*\|_{\mathbf{G}}^2 + c\, p^0\right).}{\gamma} \quad (26)$$

Corollary 1 states that $\mathbb{E}\left[\|\mathbf{x}^t - \mathbf{x}^*\|^2\right]$ linearly converges to null. Note that the sequence $\mathbb{E}\left[\|\mathbf{x}^t - \mathbf{x}^*\|^2\right]$ is not necessarily decreasing as the sequence $\mathbb{E}\left[\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2 + c\, p^t\right]$ is.

V. Numerical analysis

We numerically study the performance of the DSA algorithm for solving a logistic regression problem. Consider $q = \sum_{n=1}^{N} q_n$ training points where each node n has access to q_n of them. The training points at node n are denoted by $\mathbf{s}_{ni} \in \mathbb{R}^p$ for $i = 1, \ldots, q_n$ with the associated labels $l_{ni} \in \{-1, 1\}$. The goal is to solve the logistic regression problem

$$\tilde{\mathbf{x}}^* := \underset{\mathbf{x} \in \mathbb{R}^p}{\operatorname{argmin}} \frac{\lambda}{2}\|\mathbf{x}\|^2 + \sum_{n=1}^{N} \sum_{i=1}^{q_n} \log\left[1 + \exp(-l_{ni}\mathbf{s}_{ni}^T \mathbf{x})\right], \quad (27)$$

where the regularization term $(\lambda/2)\|\mathbf{x}\|^2$ is added to avoid overfitting the training model. The problem in (27) can be written in the form of (1) by defining the local objective functions f_n as

$$f_n(\mathbf{x}) = \frac{\lambda}{2N}\|\mathbf{x}\|^2 + \sum_{i=1}^{q_n} \log\left[1 + \exp(-l_{ni}\mathbf{s}_{ni}^T \mathbf{x})\right], \quad (28)$$

and the instantaneous local functions $f_{n,i}$ as

$$f_{n,i}(\mathbf{x}) = \frac{\lambda}{2N}\|\mathbf{x}\|^2 + q_n \log\left(1 + \exp\left(-l_{ni}\mathbf{s}_{ni}^T \mathbf{x}\right)\right), \quad (29)$$

for all $i = 1, \ldots, q_n$. In our experiments we use a synthetic dataset where components of the feature vectors $\mathbf{s}_{ni}$ with label $l_{ni} = 1$ are generated from a normal distribution with mean μ and standard deviation σ_+, while the distribution of the sample points with label $l_{ni} = -1$ is normal with mean $-\mu$ and standard deviation σ_-. The edges between nodes are generated randomly with probability p_c. The weight matrix $\mathbf{W}$ is generated using the Laplacian matrix $\mathbf{L}$ of the network as $\mathbf{W} = \mathbf{I} - \mathbf{L}/\tau$, where $\tau > (1/2)\lambda_{\max}(\mathbf{L})$. The convergence error is defined and computed as $e^t = \|\mathbf{x}^t - \mathbf{x}^*\|^2$. We set the total number of sample points $q = 500$, feature vectors dimension $p = 2$, regularization parameter $\lambda = 10^{-4}$, probability of existence of an edge $p_c = 0.3$, and $\tau = (2/3)\lambda_{\max}(\mathbf{L})$. To make the dataset *not* linearly separable we set the mean value as $\mu = 2$ and the standard deviations as $\sigma_+ = \sigma_- = 2$.

We provide a comparison of DSA with respect to DGD, EXTRA, stochastic EXTRA, and decentralized SAGA. The stochastic EXTRA is a stochastic version of EXTRA that uses naive stochastic gradients instead of gradients. DSA is different form stochastic EXTRA since it uses stochastic averaging gradients.

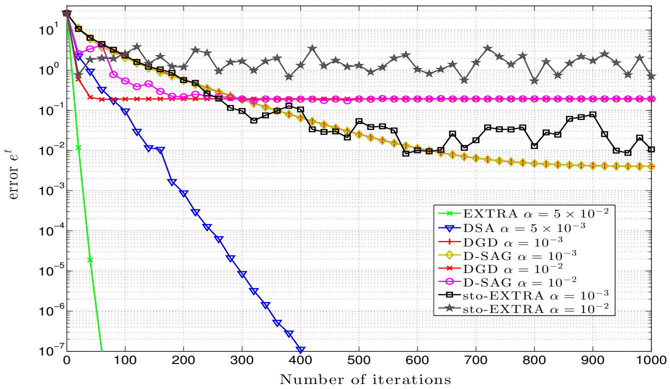

Fig. 1. Convergence paths of DSA, EXTRA, DGD, Stochastic EXTRA, and Decentralized SAGA with constant stepsizes. Relative distance to optimality $e^t = \|\mathbf{x}_t - \mathbf{x}^*\|^2$ is shown with respect to the number iterations t. DSA and EXTRA converge linearly to the optimum, while DGD, Stochastic EXTRA, and Decentralized SAGA converge to a neighborhood of the optimal solution. Smaller choice of stepsize leads to more accurate convergence for these algorithms.

The decentralized SAGA method is a stochastic version of DGD algorithm that uses stochastic averaging gradients instead of gradients which is a naive approach for developing a decentralized version of the SAGA algorithm. In our experiments we use $\tilde{\mathbf{W}} = (\mathbf{I} + \mathbf{W})/2$ for EXTRA, stochastic EXTRA, and DSA.

Fig. 1 illustrates the convergence paths of DSA, EXTRA, DGD, Stochastic EXTRA, and Decentralized SAGA with constant step sizes for $N = 20$ nodes. For EXTRA and DSA different stepsize are chosen and the best performance for EXTRA and DSA are achieved by $\alpha = 5 \times 10^{-2}$ and $\alpha = 5 \times 10^{-3}$, respectively. As shown in Fig. 1, DSA is the only stochastic algorithm that achieves linear convergence. Decentralized SAGA after couple of iterations achieves the performance of DGD and they both can not achieve exact convergence. By choosing smaller stepsize $\alpha = 10^{-3}$ they reach more accurate convergence relative to stepsize $\alpha = 10^{-2}$, however, the speed of convergence is slower for the smaller stepsize. Stochastic EXTRA also suffers from inexact convergence, but for a different reason. DGD and decentralized SAGA have inexact convergence since they solve a penalty version of (1), while stochastic EXTRA can not reach the optimal solution since the noise of stochastic gradient is not vanishing. DSA resolves both issues by combining the idea of stochastic averaging gradients to control the noise of stochastic gradients and using the double decentralized descent idea to solve the correct optimization problem. The convergence rate of EXTRA is faster than the one for DSA in terms of number of iterations, however, the complexity of EXTRA is higher than DSA. Hence, we also compare performances of these algorithms in terms of number of processed feature vectors. For instance, DSA requires 400 iterations or equivalently 400 feature vectors to achieve error $e^t = 10^{-7}$, while to achieve the same accuracy EXTRA requires 60 iterations which is equivalent to processing $60 \times 25 = 1440$ feature vectors. The difference can be more significant by increasing the number of instantaneous functions.

We also study the performance of DSA in different network topologies. We keep the parameters in Fig. 1 except we change the size of network to $N = 100$ which implies each node has $q_i = 5$ sample points. The linear convergence of DSA for random networks, complete graph, cycle, line and star are shown in Fig. 2. As we expect for the topologies that the graph is more connected and the diameter is smaller DSA converges faster.

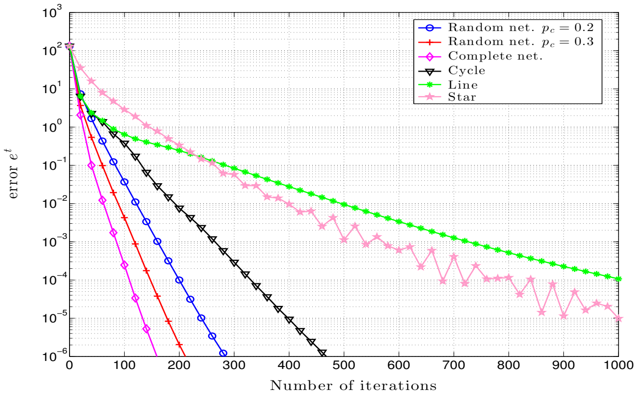

Fig. 2. Convergence paths of DSA for different network topologies. Relative distance to optimality $e^t = \|\mathbf{x}_t - \mathbf{x}^*\|^2$ is shown with respect to the number iterations t. DSA converges faster as the network connectivity increases.

References

[1] R. Bekkerman, M. Bilenko, and J. Langford, *Scaling up machine learning: Parallel and distributed approaches*. Cambridge University Press, 2011.

[2] K. Tsianos, S. Lawlor, and M. Rabbat, “Consensus-based distributed optimization: Practical issues and applications in large-scale machine learning,” *In: Proceedings of Allerton Conference on Communication, Control, and Computing*, 2012.

[3] Y. Cao, W. Yu, W. Ren, and G. Chen, “An overview of recent progress in the study of distributed multi-agent coordination,” *IEEE Transactions on Industrial Informatics*, vol. 9, pp. 427–438, 2013.

[4] C. G. Lopes and A. H. Sayed, “Diffusion least-mean squares over adaptive networks: Formulation and performance analysis,” *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 3122–3136, July 2008.

[5] 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.

[6] U. A. Khan, S. Kar, and J. M. Moura, “Diland: An algorithm for distributed sensor localization with noisy distance measurements,” *IEEE Transactions on Signal Processing*, vol. 58, no. 3, pp. 1940–1947, 2010.

[7] M. Rabbat and R. Nowak, “Distributed optimization in sensor networks,” *proceedings of the 3rd international symposium on Information processing in sensor networks*, pp. 20–27, ACM, 2004.

[8] A. Nedic and A. Ozdaglar, “Distributed subgradient methods for multiagent optimization,” *IEEE Transactions on Automatic Control*, vol. 54, pp. 48–61, 2009.

[9] D. Jakovetic, J. Xavier, and J. Moura, “Fast distributed gradient methods,” *IEEE Transactions on Automatic Control*, vol. 59, pp. 1131–1146, 2014.

[10] K. Yuan, Q. Ling, and W. Yin, “On the convergence of decentralized gradient descent,” *arXiv preprint arXiv*, 1310.7063, 2013.

[11] A. Mokhtari, Q. Ling, and A. Ribeiro, “Network newton,” in *Proc. Asilomar Conf. on Signals Systems Computers*, vol. (to appear). Pacific Grove CA, November 2-5 2014, available at http://arxiv.org/pdf/1412.3740.pdf.

[12] W. Shi, Q. Ling, G. Wu, and W. Yin, “Extra: An exact first-order algorithm for decentralized consensus optimization,” *arXiv preprint arXiv*, 1404.6264 2014.

[13] J. N. Tsitsiklis, D. P. Bertsekas, M. Athans *et al.*, “Distributed asynchronous deterministic and stochastic gradient optimization algorithms,” *IEEE transactions on automatic control*, vol. 31, no. 9, pp. 803–812, 1986.

[14] J. Duchi, A. Agarwal, and M. Wainwright, “Dual averaging for distributed optimization: Convergence analysis and network scaling,” *IEEE Transactions on Automatic Control*, vol. 57, pp. 592–606, 2012.

[15] S. S. Ram, A. Nedic, and V. V. Veeravalli, “Incremental stochastic subgradient algorithms for convex optimization,” *SIAM Journal on Optimization*, vol. 20, no. 2, pp. 691–717, 2009.

[16] S. S. Ram, A. Nedić, and V. V. Veeravalli, “Distributed stochastic subgradient projection algorithms for convex optimization,” *Journal of optimization theory and applications*, vol. 147, no. 3, pp. 516–545, 2010.

[17] M. Schmidt, N. L. Roux, and F. Bach, “Minimizing finite sums with the stochastic average gradient,” *arXiv preprint arXiv:1309.2388*, 2013.

[18] A. Defazio, F. Bach, and S. Lacoste-Julien, “Saga: A fast incremental gradient method with support for non-strongly convex composite objectives,” in *Advances in Neural Information Processing Systems*, 2014, pp. 1646–1654.

[19] A. Mokhtari and A. Ribeiro, “Decentralized double stochastic averaging gradient,” 2015, Online:http://www.seas.upenn.edu/ ∼aryanm/wiki/DSA.pdf.