GT-SAGA: A fast incremental gradient method for decentralized finite-sum minimization

Ran Xin[†], Boyue Li[†], Soummya Kar[†], and Usman A. Khan[‡]
[†]Carnegie Mellon University, Pittsburgh, PA, USA, [‡]Tufts University, Medford, MA, USA

***Abstract*—In this paper, we study decentralized solutions for finite-sum minimization problems when the underlying training data is distributed over a network of nodes. In particular, we describe the GT-SAGA algorithm that combines variance reduction and gradient tracking to achieve both robust performance and fast convergence. Variance reduction is implemented locally to asymptotically estimate each local batch gradient at each node, while gradient tracking fuses the local estimated gradients across the nodes. Combining variance reduction and gradient tracking thus enables linear convergence to the optimal solution of strongly-convex problems while keeping a low per-iteration computation complexity at each node. We cast the convergence and behavior of GT-SAGA and related methods in the context of certain practical tradeoffs and further compare their performance over a logistic regression problem with strongly convex regularization.**

***Index Terms*—Stochastic optimization, first-order methods, decentralized algorithms, variance reduction.**

I. Introduction

Many machine learning, inference, and data science problems entail massive amounts of data collected or stored at a large number of devices. Any such practical setup comes with its own set of communication constraints that limit the amount of data that can be communicated to a potentially far-away server for centralized processing. Decentralized training of the corresponding machine learning models are thus found to be of significant interest where the nodes (devices) in a network cooperate to solve the underlying optimization and learning problems without relying on any central processor. Moreover, in many large-scale problems, each node typically possesses hundreds of thousands to millions of data samples comprising the *batch* data at each device. Handling all this data at once is practically infeasible and stochastic methods where the batch data is used efficiently are preferable.

In this paper, we consider finite-sum minimization problems, i.e., $\min_{\mathbf{x}} \sum_{j=1}^{m} f_j(\mathbf{x})$, where $f_j : \mathbb{R}^p \to \mathbb{R}$ is a loss function that quantifies the modeling error incurred by the jth data sample. Such problems commonly arise in machine learning [2], signal processing [3], [4], and control [5]–[7]. Among first-order methods for finite-sum minimization, the well known gradient descent (**GD**) operates on the entire batch of data and computes m gradients to compute the descent direction $\sum_j \nabla f_j$. It can be shown that [8] **GD** requires $\mathcal{O}(m\kappa \ln\epsilon^{-1})$ to reach an ϵ-accuracy of the optimal $\mathbf{x}^*$, when $f = \frac{1}{m}\sum_{j=1}^{m} f_j$ is smooth and strongly convex, where κ is the condition number of f. Clearly, a challenge with **GD** is when m is large in which case the amount of computation required is quite extensive.

A computationally-efficient alternative of **GD** is stochastic gradient descent (**SGD**) that operates on random samples of the true gradient. In finite-sum problems, **SGD** samples uniformly at random one (or more) data samples from the batch and compute one gradient at each iteration. It can be shown that [8] **SGD** requires $\mathcal{O}(\kappa^2\epsilon^{-1})$ gradient computations to reach an ϵ-accuracy of the optimal for smooth and strongly convex objectives. It can be argued that **SGD** is often more preferable in the big data regimes where m is very large and **GD** may be practically infeasible to implement [2]. Clearly, **SGD** is sublinear, and in order to recover the linear convergence of **GD**, variance reduction methods have been developed. The key idea behind variance reduction is to estimate the batch gradient from randomly sampled stochastic gradients. Relevant variance reduced methods include **SAG** [9], **SAGA** [10], **SVRG** [11], **SARAH** [12], and **SPIDER** [13]. It can be shown that **SAGA**, for example, achieves ϵ-accuracy of the optimal with $\mathcal{O}(\max\{m,\kappa\}\ln\epsilon^{-1})$ and thus linearly converges to $\mathbf{x}^*$.

The aforementioned centralized methods face certain challenges when the data is distributed as in many large-scale learning and control problems. Decentralized stochastic first-order methods are thus preferable where the decentralized nature ensures that long-distance communication of large-dimensional vectors is avoided while the stochastic nature ensures computational efficiency. The minimization problem now is described over a network of n agents and is given by $\min \sum_{i=1}^{n}\sum_{j=1}^{m_1} f_{i,j}$, where $f_{i,j}$ is the loss incurred by the jth data sample at node i. Early work along these lines includes distributed stochastic gradient descent (**DSGD**) and can be found in [14]–[17], which extend **SGD** by adding a network fusion term on the iterates. More recent progress includes **GT-DSGD** [18]–[20] that incorporates a certain gradient tracking technique [21]–[25], where the (spatial) uncertainty across the nodes is overcome by implementing gradient fusion over the network. It can be shown that **DSGD** achieves ϵ-accuracy of the solution in $\mathcal{O}(\epsilon^{-1})$ gradient computations and certain aspects of it are improved with the addition of gradient tracking, see [4] for a detailed tutorial.

The work of RX and SK has been partially supported by NSF under grant CCF-1513936. The work of UAK has been partially supported by NSF under grants CMMI-1903972 and CBET-1935555. The author email addresses are: {ranx,boyuel,soummyak}@andrew.cmu.edu and khan@ece.tufts.edu.

An expanded version of this paper is available on Arxiv [1], where we discuss a unified framework to cast different variance reduction methods in a decentralized framework. In contrast, this paper only discusses one particular instantiation (**GT-SAGA**) of the framework.

GT-DSGD, by adding gradient tracking, shows performance improvement over **DSGD**, however, the convergence is still sublinear. This is because a local stochastic gradient is employed at each iteration whose variance does not vanish. In this paper, we describe a novel algorithm **GT-SAGA** that uses variance reduction to remove the uncertainty due to local stochastic gradients. In particular, **GT-SAGA** is based on the SAGA method [10] for variance reduction. We show that **GT-SAGA** requires $\mathcal{O}(\max\{m, \frac{\kappa^2}{(1-\lambda)^2}\}\ln\epsilon^{-1})$ parallel component gradient evaluations to achieve an ϵ-accuracy of $\mathbf{x}^*$, where m is the number of data samples at each node and $\lambda \in [0,1)$ is a network parameter. We note that in a big-data regime, i.e., $m \gg \frac{\kappa^2}{1-\lambda^2}$, the complexity of **GT-SAGA** becomes $\mathcal{O}(m\ln\epsilon^{-1})$, which is independent of the network, and is n times faster than that of centralized SAGA. Clearly, in this "big-data" regime, **GT-SAGA** acts effectively as a means for parallel computation and achieves a linear speed-up compared with its centralized counterpart.

Existing decentralized VR methods include **DSA** [26] that combines **EXTRA** [27] with **SAGA** [10], diffusion-**AVRG** that combines exact diffusion [28] and **AVRG** [29], **DSBA** [30] that adds proximal mapping [31] to each iteration of **DSA**, **ADFS** [32] that applies an accelerated randomized proximal coordinate gradient method [33] to the a dual problem, and Network-**SVRG/SARAH** [34] that implements variance-reduction in the decentralized **DANE** framework based on gradient tracking. As discussed before, in a big-data scenario where m is very large, **GT-SAGA** improves upon the convergence rate of these methods in terms of the joint dependence on κ and m, with the exception of **DSBA** and **ADFS**, which require a computation of the proximal maps.

We now describe the rest of the paper. Section II describes the decentralized stochastic optimization problem and the necessary assumptions. Section III provides the **GT-SAGA** algorithm and the main results on convergence and performance. The convergence analysis is available in Section IV. Finally, numerical experiments are detailed in Section V while Section VI concludes the paper.

II. Problem Formulation

Consider a network of n nodes such that each node i possesses a local loss function $f_i : \mathbb{R}^p \rightarrow \mathbb{R}$ that is further decomposed over m_i component functions associated to its local data samples. The nodes cooperate to solve the following finite-sum minimization problem:

$$\text{P1:} \qquad \min_{\mathbf{x}\in\mathbb{R}^p} \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x}), \qquad f_i \triangleq \frac{1}{m_i}\sum_{j=1}^{m_1} f_{i,j}(\mathbf{x}),$$

where $F \triangleq \sum_i f_i$ denotes the global loss. We assume that each local f_i is private to node i and is thus cannot be communicated to any other node. The nodes exchange information over a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V} = \{1, \ldots, n\}$ is the set of nodes and $\mathcal{E}$ is the set of ordered pairs (i, r) such that node i and j can exchange information. In order to solve Problem P1, we make the following assumptions:

Assumption 1. *There exists a set of weights $\{w_{ir}\}$ such that $w_{ir} = 0$, for each $(i, r) \notin \mathcal{E}$, and $W = \{w_{ir}\}$ is primitive and doubly-stochastic.*

We thus have $W^\infty = \frac{1}{n}\mathbf{1}\mathbf{1}^\top$. Note that the weight matrix is not required to be symmetric and is applicable to directed graphs that admit doubly-stochastic weights [35].

Assumption 2. *The global function F is μ-strongly-convex, i.e., $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^p$ and for some $\mu > 0$, we have*

$$F(\mathbf{y}) \geq F(\mathbf{x}) + \langle \nabla F(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle + \frac{\mu}{2}\|\mathbf{x} - \mathbf{y}\|^2,$$

where $\nabla F : \mathbb{R}^p \rightarrow \mathbb{R}^p$ is the gradient of F and $\langle \mathbf{x}, \mathbf{y} \rangle$ is the inner product of two vectors.

Assumption 3. *Each local function $f_{i,j}$ is L-smooth, i.e., $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^p$ and for some $L > 0$, we have*

$$\|\nabla f_{i,j}(\mathbf{x}) - \nabla f_{i,j}(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|.$$

Clearly, under Assumption 2, the global function F has a unique minimizer, denoted as $\mathbf{x}^*$, which is the optimal solution of Problem P1. Moreover, under Assumption 3, the global objective F is also L-smooth and $L \geq \mu$. We use $\kappa \triangleq L/\mu$ to denote the condition number of F. With the help of these assumptions, we now describe the proposed **GT-SAGA** algorithm and main results next.

III. **GT-SAGA**: Algorithm and Main Results

We now systematically introduce the **GT-SAGA** algorithm. To this aim, let $\mathbf{x}_k^i$ denote the estimate of the optimal solution $\mathbf{x}^*$ at node i and iteration k and consider **DGD** [14]:

$$\mathbf{x}_i^{k+1} = \sum_{r=1}^{n} w_{ir}\mathbf{x}_r^k - \alpha_k \nabla f_i(\mathbf{x}_i^k), \tag{1}$$

where α_k are the step-sizes. **DGD** converges linearly to an error ball around $\mathbf{x}^*$ with a constant step-size and converges sublinearly to $\mathbf{x}^*$ with decaying step-sizes [14], [15]. The reason behind this *inexact* linear convergence is that $\mathbf{x}^*$ is not a fixed point of **DGD** because $\nabla f_i(\mathbf{x}^*) \neq \mathbf{0}_p$, in general; recall that $\sum_i \nabla f_i(\mathbf{x}^*) = \mathbf{0}_p$. One way to recover the exact linear convergence is to replace the local gradient $\nabla f_i(\mathbf{x}_i^k)$ with an estimate of the global gradient ∇F. The resulting algorithm **GT-DGD** is written as

$$\mathbf{x}_i^{k+1} = \sum_{r=1}^{n} w_{ir}\mathbf{x}_r^k - \alpha \mathbf{y}_i^k, \tag{2}$$

$$\mathbf{y}_i^{k+1} = \sum_{r=1}^{n} w_{ir}\mathbf{y}_r^k + \nabla f_i(\mathbf{x}_i^{k+1}) - f_i(\mathbf{x}_i^k), \tag{3}$$

where $\mathbf{y}_i^k \in \mathbb{R}^p$ estimates the global gradient [21]–[24], [36]. It can be shown that **GT-DGD** linearly converges to the optimal solution and requires $\mathcal{O}(m \max\{\kappa, (1-\lambda)^{-1}\}\ln\epsilon^{-1})$ gradient computations per node per iteration to achieve an ϵ-accurate solution (for positive semi-definite weights) [37], where $m = m_i$, $\forall i$, is the total data samples at each node and λ is the second-largest singular value of W. Clearly, the complexity of **GT-DGD** increases with the number of data samples m and thus may be infeasible when m is large.

A stochastic implementation of **GT-DGD** that does not require computing m gradients at each iteration is thus of significant value. Along these lines, stochastic extensions have been studied in [18] over undirected graphs and in [19], [20] over directed graphs. The **GT-DSGD** algorithm [18] over undirected graphs is given as follows:

$$\mathbf{x}_i^{k+1} = \sum_{r=1}^{n} w_{ir}\mathbf{x}_r^k - \alpha_k \mathbf{y}_i^k, \tag{4}$$

$$\mathbf{y}_i^{k+1} = \sum_{r=1}^{n} w_{ir}\mathbf{y}_r^k + \nabla f_{i,s_i^k}(\mathbf{x}_i^{k+1}) - f_{i,s_i^k}(\mathbf{x}_i^k), \tag{5}$$

where s_i^k is an index chosen uniformly at random from the index set $\{1,\ldots,m_i\}$ at each node i. We note here that $\mathbf{y}_k^i$ does not estimate the global gradient anymore due to the use of stochastic gradients. As a result, **GT-DSGD** loses the exact linear convergence with a constant step-size and converges sublinearly to the optimal with decaying step-sizes.

A. *GT-SAGA*

Recall that variance-reduced methods estimate the local batch gradient ∇f_i from stochastic gradients $\nabla f_{i,s_i^k}$. Thus, a natural extension of **GT-DSGD** is to replace the stochastic gradient $\nabla f_{i,s_i^k}$ with a gradient estimator $\mathbf{g}_k^i$. Any gradient estimation technique potentially works here and in this paper we explore the gradient estimator that comes from SAGA [10]. Intuitively, $\mathbf{g}_i^k \to \nabla f_i$ due to which $\mathbf{y}_k^i \to \nabla F$ and **GT-SAGA** linearly converges to $\mathbf{x}^*$.

Algorithm 1 **GT-SAGA** at each node i

```
Require: $\mathbf{x}_i^0 \in \mathbb{R}^p$; $\alpha$; $\{w_{ir}\}_{r=1}^n$; $\mathbf{y}_i^0 = \mathbf{g}_i^0 = \nabla f_i(\mathbf{x}_i^0)$;
    $\widehat{\mathbf{x}}_{i,j} = \mathbf{x}_0^i, \forall j$; Gradient table $\{\nabla f_{i,j}(\widehat{\mathbf{x}}_{i,j})\}_{j=1}^{m_i}$.
1: for $k = 0,1,2,\cdots$ do
2:     Update the local estimate of the solution:
           $\mathbf{x}_i^{k+1} = \sum_{r=1}^n \widetilde{w}_{ir}\mathbf{x}_r^k - \alpha\mathbf{y}_i^k$;
3:     Select $s_i^{k+1}$ uniformly at random from $\{1,\cdots,m_i\}$;
4:     Update the local stochastic gradient estimator:
           $\mathbf{g}_i^{k+1} = \nabla f_{i,s_i^{k+1}}(\mathbf{x}_i^{k+1}) - \nabla f_{i,s_i^{k+1}}(\mathbf{z}_{i,s_i^{k+1}}^{k+1})$
               $+ \frac{1}{m_i}\sum_{j=1}^{m_i}\nabla f_{i,j}(\mathbf{z}_{i,j}^{k+1})$;
5:     Update the $s_i^{k+1}$-th entry in the gradient table:
           $\nabla f_{i,s_i^{k+1}}(\widehat{\mathbf{x}}_{i,s_i^{k+1}}) \to \nabla f_{i,s_i^{k+1}}(\mathbf{x}_i^{k+1})$;
6:     Update the local gradient tracker:
           $\mathbf{y}_i^{k+1} = \sum_{r=1}^n \widetilde{w}_{ir}\mathbf{y}_r^k + \mathbf{g}_i^{k+1} - \mathbf{g}_i^k$;
7: end for
```

GT-SAGA, formally described in Algorithm 14, requires a gradient table that stores m_i gradients at each node i, i.e., $\{\nabla f_{i,j}(\widehat{\mathbf{x}}_{i,j})\}_{i=1}^{m_i}$, where $\widehat{\mathbf{x}}_{i,j}$ represents the most recent iterate where the gradient of $f_{i,j}$ was last evaluated. At each iteration k, a random index s_i^{k+1} is generated and $\nabla f_{i,s_i^{k+1}}$, last evaluated and recorded at some $\widehat{\mathbf{x}}_{i,s_i^{k+1}}$, is replaced with $\nabla f_{i,s_i^{k+1}}$ now evaluated at the current iterate $\mathbf{x}_i^{k+1}$.

B. *Main results*

The main convergence result of **GT-SAGA** is summarized in the following theorem. To this aim, let $m \triangleq \min_i m_i$ and $M \triangleq \max_i m_i$.

Theorem 1. *Let Assumptions 1, 2 and 3 hold. If the step-size α is such that*

$$0 < \alpha \leq \overline{\alpha} \triangleq \min\left\{\mathcal{O}\left(\frac{1}{\mu M}\right), \mathcal{O}\left(\frac{m}{M}\frac{(1-\lambda^2)^2}{L\kappa}\right)\right\},$$

*then **GT-SAGA** linearly converges (in the mean-square sense) to the optimal solution $\mathbf{x}^*$ of Problem P1. If $\alpha = \overline{\alpha}$, then **GT-SAGA** achieves ϵ-accuracy of $\mathbf{x}^*$ in*

$$\mathcal{O}\left(\max\left\{M, \frac{M}{m}\frac{\kappa^2}{(1-\lambda^2)^2}\right\}\ln\frac{1}{\epsilon}\right)$$

parallel local component gradient computations.

The formal proof of **GT-SAGA** is deferred to the next section. We now discuss some of its salient features.

(i) Intuitively, **GT-SAGA** performs two fusion operations. One is a fusion at the node-level, where the stochastic gradients are used to estimate the entire local gradient ∇f_i. In fact, it can be shown that as $\mathbf{x}_i^k$ and $\mathbf{z}_{i,j}^k$ approach to an agreement on $\mathbf{x}^*$, the variance of the gradient estimator decays to zero [1]. The other is fusion across the nodes where gradient tracking is used to estimate the global gradient ∇F from the local estimated gradients; we note here that gradient tracking is related to dynamic average consensus [38]. The variable $\mathbf{y}_k^i$ thus goes to ∇F, at each node.

(ii) **GT-SAGA** relies on stochastic gradients and thus has a low per-iteration computation complexity and converges linearly to the exact optimal $\mathbf{x}^*$. It can be thus considered as an appropriate counterpart of centralized SAGA [10]. Note that both gradient tracking and variance reduction are needed to establish linear convergence, i.e., the classical **DSGD** does not ensure linear convergence with any one of these two features.

(iii) It was shown in [37] that the best case complexity of **GT-DGD** is $\mathcal{O}(M\max\{\kappa,(1-\lambda)\}\ln\epsilon^{-1})$, which reduces to $\mathcal{O}(m\kappa\ln\epsilon^{-1})$ for well-connected networks. **GT-SAGA**, with complexity of $\mathcal{O}(M\ln\epsilon^{-1})$, thus improves the joint dependency on the number of samples M and the condition number κ.

(iv) In large-scale problems, i.e., when $M \approx m \gg \frac{\kappa^2}{(1-\lambda^2)^2}$, the convergence is independent of the network and is n times faster than SAGA. **GT-SAGA** thus achieves a linear speedup in terms of the total number of nodes.

(v) The performance improvement in **GT-SAGA** comes at a price of additional storage $\mathcal{O}(m_i p)$ at each node. It is noteworthy that this storage cost can be reduced to $\mathcal{O}(m_i)$ for certain problems of interest, for example, logistic regression and least squares, by exploiting the structure of the objective functions [9], [10]. When additional storage is not feasible, other variance-reduction methods like SVRG [1], [11] can be employed that do not require extra storage but they have their own implementation constraints.

IV. Convergence Analysis

We now provide the convergence analysis of **GT-SAGA**. Our approach to is to first use standard arguments to obtain an LTI system to describe the algorithm. We then study the convergence properties of this LTI system to establish convergence guarantees of **GT-SAGA**. The said LTI system describes the evolution of the following vector, with $p = 1$ assumed for simplicity,

$$\mathbf{u}^k = \begin{bmatrix} \mathbb{E}\left[\left\|\mathbf{x}^k - W^\infty \mathbf{x}^k\right\|^2\right] \\ \mathbb{E}\left[n\left\|\overline{\mathbf{x}}^k - \mathbf{x}^*\right\|^2\right] \\ \mathbb{E}\left[t^k\right] \\ \mathbb{E}\left[L^{-2}\left\|\mathbf{y}^k - W_\infty \mathbf{y}^k\right\|^2\right], \end{bmatrix}$$

where: the first term, the agreement error, quantifies how far the network vector $\mathbf{x}^k \triangleq [\mathbf{x}_k^1 \ \ldots \ \mathbf{x}_k^n]^\top$ is from the average $\overline{\mathbf{x}}^k \triangleq W^\infty \mathbf{x}^k$; the second term, the optimality gap, quantifies the gap between the average and the optimal $\mathbf{x}^*$; and the last term is the gradient tracking error with $\mathbf{y}^k \triangleq [\mathbf{y}_k^1 \ \ldots \ \mathbf{y}_k^n]^\top$. For the third term, we define $\widehat{\mathbf{x}}_{i,j}^k$ as the most recent iterate ($\mathbf{x}_{i}^{\underline{k}}$) where the component gradient $\nabla f_{i,j}$ was evaluated before iteration k (i.e., at some iteration $\underline{k} \leq k$) and define

$$t_i^k \triangleq \tfrac{1}{m_i}\textstyle\sum_{j=1}^{m_i}\left\|\widehat{\mathbf{x}}_{i,j}^k - \mathbf{x}^*\right\|^2, \qquad t^k \triangleq \textstyle\sum_{i=1}^n t_i^k. \tag{6}$$

In other words, t_k^i quantifies the average optimality gap w.r.t to the iterates that are present in the gradient table at time k. We have the following proposition on the evolution of $\mathbf{u}_k$.

Proposition 1. *Let Assumptions 1, 2, and 3 hold. If the step-size α follows $0 < \alpha \leq \frac{\mu(1-\lambda)}{16L^2}$, we have, $\forall k \geq 1$,*

$$\mathbf{u}^{k+1} \leq \begin{bmatrix} \dfrac{1+\lambda^2}{2} & 0 & 0 & \dfrac{2\alpha^2 L^2}{1-\lambda^2} \\ \dfrac{2L^2\alpha}{\mu} & 1 - \dfrac{\mu\alpha}{2} & \dfrac{2L^2\alpha^2}{n} & 0 \\ \dfrac{2}{m} & \dfrac{2}{m} & 1 - \dfrac{1}{M} & 0 \\ \dfrac{104}{1-\lambda^2} & \dfrac{71}{1-\lambda^2} & \dfrac{19}{1-\lambda^2} & \dfrac{3+\lambda^2}{4} \end{bmatrix} \mathbf{u}^k, \tag{7}$$

compactly written as $\mathbf{u}^{k+1} \triangleq G_\alpha \mathbf{u}^k$, where $m = \min_i m_i$ and $M = \max_i m_i$.

The proof of the constituent inequalities in (7) is beyond the scope of this paper. In [1], we provide a unified framework to incorporate different variance reduction methods in the **GT-DSGD** framework and the proof of (7) can be deduced from there. The corresponding arguments use the strong convexity (Assumption 2) and L-smooth (Assumption 3) inequalities in addition to some standard bounds and inequalities from linear systems. It is however clear that an R-linear convergence of **GT-SAGA** is readily established if we can show that $\rho(G_\alpha) < 1$ for a certain range of the step-size α. To this aim, we recall the following lemma from [39], which will be used to show that $\rho(G_\alpha) < 1$.

Lemma 1. *Let A be a non-negative matrix and $\mathbf{x}$ be positive, i.e., $\mathbf{x} > 0$. If $A\mathbf{x} \leq \beta\mathbf{x}$ for $\beta > 0$, then $\rho(A) \leq \beta$.*

A. *Proof of Theorem 1*

To show $\rho(G_\alpha) < \beta$, for some $\beta < 1$, we note from Lemma 1 that it suffices to show that $G_\alpha \boldsymbol{\epsilon} \leq \beta\boldsymbol{\epsilon}$, for a positive vector $\boldsymbol{\epsilon} = [\epsilon_1 \ \epsilon_2 \ \epsilon_3 \ \epsilon_4]^\top$ and for a valid range of the step-size α. Here, we choose $\beta = 1 - \frac{\mu\alpha}{4}$ leading to find α such that the following inequalities hold:

$$\frac{\mu\alpha}{4} + \frac{2L^2}{1-\lambda^2}\frac{\epsilon_4}{\epsilon_1}\alpha^2 \leq \frac{1-\lambda^2}{2} \tag{8}$$

$$\frac{2L^2}{n}\epsilon_3\alpha \leq \frac{\mu}{4}\epsilon_2 - \frac{2L^2}{\mu}\epsilon_1 \tag{9}$$

$$\frac{\mu\alpha}{4} \leq \frac{1}{M} - \frac{2}{m}\frac{\epsilon_1}{\epsilon_3} - \frac{2}{m}\frac{\epsilon_2}{\epsilon_3} \tag{10}$$

$$\frac{\mu\alpha}{4} \leq \frac{1-\lambda^2}{4} - \frac{104}{1-\lambda^2}\frac{\epsilon_1}{\epsilon_4} - \frac{71}{1-\lambda^2}\frac{\epsilon_2}{\epsilon_4} - \frac{19}{1-\lambda^2}\frac{\epsilon_3}{\epsilon_4} \tag{11}$$

Clearly, that (9)–(11) hold for some feasible range of α is equivalent to the RHS of (9)–(11) being positive. Based on this observation, we will next fix the values of $\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4$ that are independent of α.

First, for the RHS of (9) to be positive, we set $\epsilon_1 = 1, \epsilon_2 = 8.5\kappa^2$, where recall that $\kappa = L/\mu \geq 1$. Second, the RHS of (10) being positive is equivalent to

$$\epsilon_3 > \frac{2M}{m}\epsilon_1 + \frac{2M}{m}\epsilon_2 = \frac{2M}{m} + \frac{17M\kappa^2}{m}. \tag{12}$$

We therefore set $\epsilon_3 = \frac{20M\kappa^2}{m}$. Third, we note that the RHS of (11) being positive is equivalent to

$$\epsilon_4 > \frac{4}{(1-\lambda^2)^2}\left(104\epsilon_1 + 71\epsilon_2 + 19\epsilon_3\right),$$

which is satisfied by $\epsilon_4 = \frac{8700}{(1-\lambda^2)^2}\frac{M\kappa^2}{m}$.

We now solve for the range of α from (8)–(11) given the previously fixed $\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4$. From (9), we have that

$$\alpha \leq \frac{n}{2L^2\epsilon_3}\left(\frac{\mu}{4}\epsilon_2 - \frac{2L^2}{\mu}\epsilon_1\right) = \frac{m}{M}\frac{n}{320\kappa L}. \tag{13}$$

Moreover, it is straightforward to verify that if α satisfies

$$0 < \alpha \leq \frac{m}{M}\frac{(1-\lambda^2)^2}{320\kappa L} \tag{14}$$

then (8) holds. Next, to make (10) hold, it suffices to have

$$\alpha \leq \frac{1}{5\mu M}. \tag{15}$$

Finally, to make (11) hold, it suffices to make

$$\alpha \leq \frac{1-\lambda^2}{2\mu}. \tag{16}$$

To summarize, combining (13)–(16), we conclude that if the step-size α satisfies

$$0 < \alpha \leq \overline{\alpha} := \min\left\{\frac{1}{5\mu M}, \frac{m}{320M}\frac{(1-\lambda^2)^2}{L\kappa}\right\}, \tag{17}$$

then $\rho\left(G_\alpha\right) \leq 1 - \frac{\mu\alpha}{4}$ by Lemma 1, and if $\alpha = \overline{\alpha}$, we have

$$\rho\left(G_\alpha\right) \leq 1 - \min\left\{\frac{1}{20M}, \frac{m}{1280M}\frac{(1-\lambda^2)^2}{\kappa^2}\right\},$$

which completes the proof. □

V. Numerical Experiments

We next consider decentralized logistic regression, where the goal is to classify hand-written digits $\{3, 8\}$ from the MNIST dataset [40]. For image classification, each node i possesses m_i labeled images. The jth image at node i is vectorized as a feature vector $\boldsymbol{\theta}_j \in \mathbb{R}^{784}$ and ζ_j is its corresponding binary label ($+1$ or -1). The nodes cooperate to solve the following smooth and strongly-convex problem:

$$\min_{\mathbf{w},b} F(\mathbf{w}, b) = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} f_{i,j}(\mathbf{w}, b),$$

where each local cost function is

$$f_{i,j}(\mathbf{w}, b) = \ln\left[1 + \exp\left\{-(\mathbf{w}^\top \boldsymbol{\theta}_j + b)\zeta_j\right\}\right] + \frac{\lambda}{2}\|\mathbf{w}\|_2^2,$$

where the optimization variable is $\mathbf{x} = [\mathbf{w}^\top \; b]^\top$.

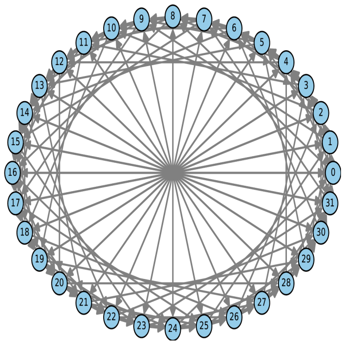
Fig. 1. Directed exponential graph with $n = 32$ nodes.

In the first experiment, we consider the directed exponential graphs [41], shown in Fig. 1. The class of directed exponential graphs is weight-balanced (therefore admits doubly-stochastic weights), sparsely-connected (low-communication per node), and possesses a strong algebraic connectivity. We divide the total number of 11968 training samples evenly among all nodes, i.e., $m_i = m = 374$. Each training sample is normalized to a unit vector. We set the regularization parameter $\lambda = \frac{1}{nm}$ [9]. The optimal solution is found by centralized Nesterov gradient descent. Fig. 2 plots the average residual $\frac{1}{n}\sum_{i=1}^{n}\left(F(x_k^i) - F^*\right)$ across all nodes and compares the proposed **GT-SAGA** (with both gradient tracking and variance reduction) with **GT-DSGD** (without variance reduction) and **DSGD** (without gradient tracking and variance reduction). The hyper-parameters for all algorithms are tuned manually for best performances. We can clearly observe that although **DSGD** and **GT-DSGD** have comparable performance, **GT-SAGA** linearly converges to the optimal solution. Next, we compare the accuracy of the three algorithms in Fig. 3 over test data with 1200 images. Here again, we observe that **GT-SAGA** outperforms the other algorithms.

In the next experiment, we consider a geometric (nearest-neighbor) graph with $n = 100$ nodes, see Fig. 4 (left). Each node possesses a different number of data samples; the sample distribution at the nodes is shown in Fig. 4 (right). This unbalanced data distribution over a geometric graph models ad hoc wireless networks where the connectivity is range-based and the nodes have resource constraints. Performance comparison among **GT-SAGA**, **GT-DSGD**, and **DSGD** is shown in Fig. 5. As before, **GT-SAGA** linearly converges to the optimal and outperforms the other algorithms. Of importance here is that, in this unbalanced data scenario, **GT-DSGD** outperforms **DSGD**; this is consistent with the theory, see [4] for a detailed discussion and precise technical statements.

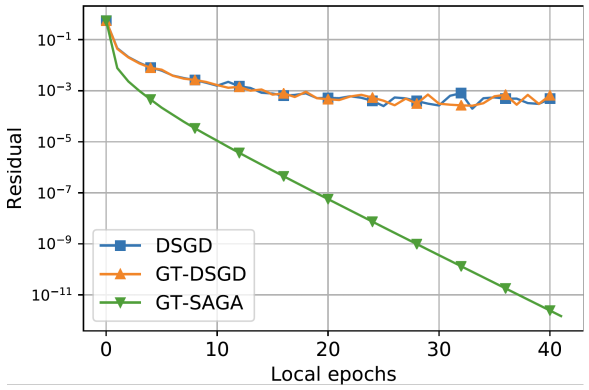

Fig. 2. Convergence rate comparison over the directed exponential graph.

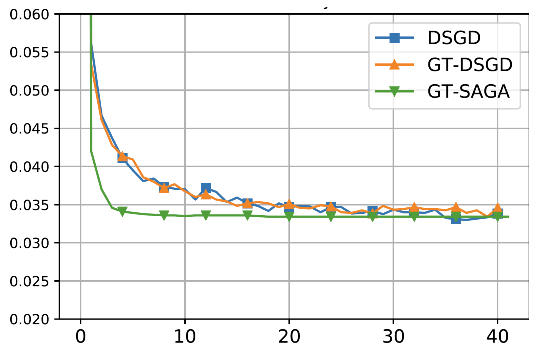

Fig. 3. Test accuracy comparison over the directed exponential graph.

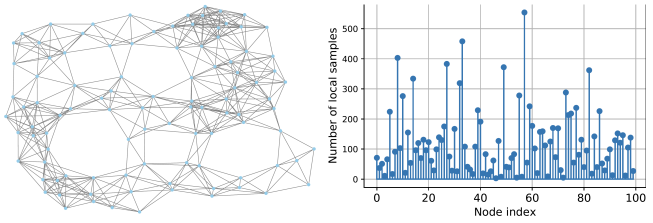

Fig. 4. (Left) Random undirected geometric graph with $n = 100$ nodes. (Right) Unbalanced data distribution across the nodes.

VI. Conclusions

In this paper, we discuss decentralized, stochastic, first-order methods to solve finite-sum minimization problems. In particular, we describe **GT-SAGA** where each node employs a stochastic gradient at each iteration computed from random samples of its local data batch. **GT-SAGA** uses variance reduction to asymptotically estimate each local batch gradient and fuses the local estimated gradients across the nodes with the help of gradient tracking. We show that **GT-SAGA** converges linearly to the optimal solution of smooth and strongly-convex problems while maintaining a low per-iteration computation complexity. Finally, we note that non-convex extensions of the variance reduction and gradient tracking have been recently developed in [42], [43] and references therein.

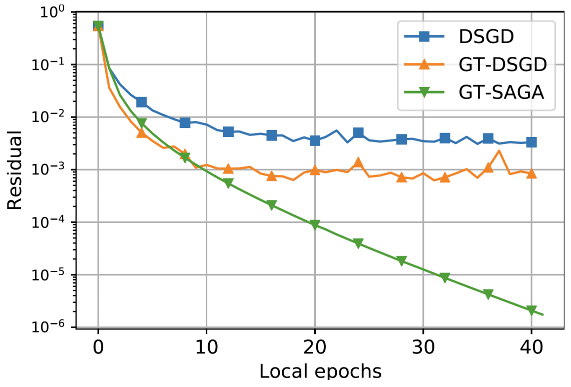

Fig. 5. Convergence rate comparison over the geometric graph.

References

[1] R. Xin, U. A. Khan, and S. Kar, "Variance-reduced decentralized stochastic optimization with accelerated convergence," *arXiv:1912.04230*, Dec. 2019.

[2] L. Bottou, F. E. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *SIAM Review*, vol. 60, no. 2, pp. 223–311, 2018.

[3] H. Raja and W. U. Bajwa, "[cloud k-svd: A] collaborative dictionary learning algorithm for big, distributed data," *IEEE Transactions on Signal Processing*, vol. 64, no. 1, pp. 173–188, 2016.

[4] R. Xin, S. Kar, and U. A. Khan, "Decentralized stochastic optimization and machine learning," *IEEE Signal Processing Magazine*, May 2020, to appear.

[5] S. Safavi, U. A. Khan, S. Kar, and J. M. F. Moura, "Distributed localization: A linear theory," *Proceedings of the IEEE*, vol. 106, no. 7, pp. 1204–1223, Jul. 2018.

[6] S. Pu and A. Garcia, "A flocking-based approach for distributed stochastic optimization," *Operations Research*, vol. 1, pp. 267–281, 2018.

[7] T. Yang, X. Yi, J. Wu, Y. Yuan, D. Wu, Z. Meng, Y. Hong, H. Wang, Z. Lin, and K. H. Johansson, "A survey of distributed optimization," *Annual Reviews in Control*, 2019.

[8] Y. Nesterov, *Lectures on convex optimization*, vol. 137, Springer, 2018.

[9] M. Schmidt, N. Le Roux, and F. Bach, "Minimizing finite sums with the stochastic average gradient," *Mathematical Programming*, vol. 162, no. 1-2, pp. 83–112, 2017.

[10] A. Defazio, F. Bach, and S. Lacoste-Julien, "SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives," in *NeurIPS*, 2014, pp. 1646–1654.

[11] R. Johnson and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," in *Advances in neural information processing systems*, 2013, pp. 315–323.

[12] L. M. Nguyen, J. Liu, K. Scheinberg, and M. Takáč, "SARAH: a novel method for machine learning problems using stochastic recursive gradient," in *ICML*. JMLR. org, 2017, pp. 2613–2621.

[13] C. Fang, C. J. Li, Z. Lin, and T. Zhang, "Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator," in *NeurIPS*, 2018, pp. 689–699.

[14] A. Nedić and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Trans. on Autom. Control*, vol. 54, no. 1, pp. 48, 2009.

[15] K. Yuan, Q. Ling, and W. Yin, "On the convergence of decentralized gradient descent," *SIAM J. on Optim.*, vol. 26, no. 3, pp. 1835–1854, Sep. 2016.

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

[17] J. Chen and A. H. Sayed, "Diffusion adaptation strategies for distributed optimization and learning over networks," *IEEE Trans. on Sig. Process.*, vol. 60, no. 8, pp. 4289–4305, Aug. 2012.

[18] Shi Pu and Angelia Nedić, "A distributed stochastic gradient tracking method," in *2018 IEEE Conference on Decision and Control (CDC)*. IEEE, 2018, pp. 963–968.

[19] U. A. Khan R. Xin, A. K. Sahu and S. Kar, "Distributed stochastic optimization with gradient tracking over strongly-connected networks," in *58th IEEE Conference of Decision and Control*, Nice, France, 2019.

[20] M. I. Qureshi, R. Xin, S. Kar, and U A. Khan, "On the convergence of decentralized stochastic optimization over strongly-connected directed graphs," *IEEE Letters to Control System Society*, 2020.

[21] J. Xu, S. Zhu, Y. C. Soh, and L. Xie, "Augmented distributed gradient methods for multi-agent optimization under uncoordinated constant stepsizes," in *IEEE 54th Annual Conference on Decision and Control*, 2015, pp. 2055–2060.

[22] P. Di Lorenzo and G. Scutari, "Next: In-network nonconvex optimization," *IEEE Trans. on Sig. and Information Processing over Networks*, vol. 2, no. 2, pp. 120–136, 2016.

[23] C. Xi, R. Xin, and U. A. Khan, "ADD-OPT: Accelerated distributed directed optimization," *IEEE Transactions on Automatic Control*, vol. 63, no. 5, pp. 1329–1339, May 2018.

[24] R. Xin and U. A. Khan, "A linear algorithm for optimization over directed graphs with geometric convergence," *IEEE Control Systems Letters*, vol. 2, no. 3, pp. 315–320, 2018.

[25] F. Saadatniaki, R. Xin, and U. A. Khan, "Decentralized optimization over time-varying directed graphs with row and column-stochastic matrices," *IEEE Transactions on Automatic Control*, Dec. 2019.

[26] A. Mokhtari and A. Ribeiro, "DSA: decentralized double stochastic averaging gradient algorithm," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2165–2199, 2016.

[27] W. Shi, Q. Ling, G. Wu, and W. Yin, "EXTRA: an exact first-order algorithm for decentralized consensus optimization," *SIAM J. Optim.*, vol. 25, no. 2, pp. 944–966, 2015.

[28] K. Yuan, B. Ying, X. Zhao, and A. H. Sayed, "Exact diffusion for distributed optimization and learning Part I: Algorithm development," *IEEE Trans. on Sig. Process.*, vol. 67, no. 3, pp. 708–723, 2018.

[29] B. Ying, K. Yuan, and A. H. Sayed, "Variance-reduced stochastic learning under random reshuffling," *arXiv:1708.01383*, 2017.

[30] Z. Shen, A. Mokhtari, T. Zhou, P. Zhao, and H. Qian, "Towards more efficient stochastic decentralized learning: Faster convergence and sparse communication," *arXiv:1805.09969*, 2018.

[31] A. Defazio, "A simple practical accelerated method for finite sums," in *Advances in neural information processing systems*, 2016, pp. 676–684.

[32] H. Hendrikx, F. Bach, and L. Massoulié, "Asynchronous accelerated proximal stochastic gradient for strongly convex distributed finite sums," *arXiv:1901.09865*, 2019.

[33] Q. Lin, Z. Lu, and L. Xiao, "An accelerated randomized proximal coordinate gradient method and its application to regularized empirical risk minimization," *SIAM J. Optim.*, vol. 25, no. 4, pp. 2244–2273, 2015.

[34] B. Li, S. Cen, Y. Chen, and Y. Chi, "Communication-efficient distributed optimization in networks with gradient tracking," *arXiv:1909.05844*, 2019.

[35] B. Gharesifard and J. Cortés, "Distributed strategies for generating weight-balanced and doubly stochastic digraphs," *European Journal of Control*, vol. 18, no. 6, pp. 539–557, 2012.

[36] G. Qu and N. Li, "Harnessing smoothness to accelerate distributed optimization," *IEEE Trans. on Control of Network Systems*, vol. 5, no. 3, pp. 1245–1260, 2017.

[37] S. A. Alghunaim, K. Yuan, and A. H. Sayed, "A linearly convergent proximal gradient algorithm for decentralized optimization," *arXiv:1905.07996*, 2019.

[38] M. Zhu and S. Martínez, "Discrete-time dynamic average consensus," *Automatica*, vol. 46, no. 2, pp. 322–329, 2010.

[39] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, Cambridge, 1985.

[40] Y. LeCun, "The MNIST database of handwritten digits," *http://yann.lecun.com/exdb/mnist/*, 1998.

[41] M. Assran, N. Loizou, N. Ballas, and M. Rabbat, "Stochastic gradient push for distributed deep learning," *arXiv:1811.10792*, 2018.

[42] R. Xin, U. A. Khan, and S. Kar, "An improved convergence analysis for decentralized online stochastic non-convex optimization," *arXiv preprint arXiv:2008.04195*, 2020.

[43] R. Xin, U. A. Khan, and S. Kar, "A near-optimal stochastic gradient method for decentralized non-convex finite-sum optimization," *arXiv preprint arXiv:2008.07428*, 2020.