Communication-Efficient Variance-Reduced Decentralized Stochastic Optimization over Time-Varying Directed Graphs

Yiyue Chen, *Student Member, IEEE,* Abolfazl Hashemi, and Haris Vikalo, *Senior Member, IEEE*

***Abstract*— We consider the problem of decentralized optimization over time-varying directed networks. The network nodes can access only their local objectives, and aim to collaboratively minimize a global function by exchanging messages with their neighbors. Leveraging sparsification, gradient tracking and variance-reduction, we propose a novel communication-efficient decentralized optimization scheme that is suitable for resource-constrained time-varying directed networks. We prove that in the case of smooth and strongly-convex objective functions, the proposed scheme achieves an accelerated linear convergence rate. To our knowledge, this is the first decentralized optimization framework for time-varying directed networks that achieves such a convergence rate and applies to settings requiring sparsified communication. Experimental results on both synthetic and real datasets verify the theoretical results and demonstrate efficacy of the proposed scheme.**

***Index Terms*— Convex optimization, decentralized optimization, stochastic optimization**

I. Introduction

Decentralized optimization problems are encountered in a number of settings in control, signal processing, and machine learning [2, 3, 4]. Formally, the goal of a decentralized optimization task is to minimize global objective in the form of a finite sum

$$\min_{\mathbf{x}\in\mathcal{X}} \left[f(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x}) \right], \tag{1}$$

where $f_i(\mathbf{x}) = \frac{1}{m_i}\sum_{j=1}^{m_i} f_{i,j}(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R}$ for $i \in [n] := \{1, ..., n\}$ denotes the local objective function that averages loss of m_i data points at node i, and $\mathcal{X}$ denotes a convex compact constraint set. The n nodes of the network exchange messages to collaboratively solve (1). Since the communication links between nodes in real-world networks are often uni-directional and dynamic (i.e., time-varying), we model the network by a sequence of directed graphs $\mathcal{G}(t) = (|n|, \mathcal{E}(t))$, where the existence of an edge $e_{i,j} \in \mathcal{E}(t)$ implies that node i can send messages to node j at time step t.

This work was supported in part by NSF grant 1809327.

Yiyue Chen and Haris Vikalo are with the Department of Electrical and Computer Engineering, University of Texas at Austin, Austin, TX 78712 USA. Abolfazl Hashemi is with the Oden Institute for Computational Engineering and Sciences, the University of Texas at Austin, Austin, TX 78712 USA.

A preliminary version of this article is presented at the 2021 International Conference on Acoustics, Speech, and Signal Processing (ICASSP) [1].

As the networks and datasets keep increasing in size, computational complexity and communication cost of decentralized optimization start presenting major challenges. To reduce the complexity of computing gradients, decentralized stochastic methods that allow each agent to perform gradient estimation by processing small subset of local data are preferred [5]. However, such techniques exhibit low convergence rates and suffer from high variance of the local stochastic gradients. Remedies for these impediments in decentralized optimization over networks and in stochastic optimization in centralized settings include gradient tracking [6, 7] and variance reduction [8, 9], respectively; however, no such remedies have been developed for decentralized optimization over time-varying directed networks. On another note, communication constraints often exacerbate large-scale decentralized optimization problems where the size of the network or the dimension of local model parameters may be on the order of millions. This motivates design of communication-efficient algorithms that compress messages exchanged between network nodes yet preserve fast convergence. In this paper, we study the general setting where a network is time-varying and directed, and present, to the best of our knowledge, the first variance-reduced communication-sparsified algorithm for decentralized convex optimization over such networks. Moreover, we theoretically establish that when the local objective functions are smooth and strongly-convex, the proposed algorithm achieves accelerated linear convergence rate. Note that while our focus is on communication-constrained settings, the proposed scheme readily applies to decentralized optimization problems where time-varying directed graphs operate under no communication constraints; in fact, to our knowledge, this is the first variance-reduced stochastic algorithm for such problems.

A. Related work

The first work on decentralized optimization over networks dates back to the 1980s [10]. A number of studies that followed in subsequent years was focused on the decentralized average consensus problem, where the network nodes work collaboratively to compute the average value of local vectors. Convergence conditions for achieving consensus over directed and undirected time-varying graphs were established in [3, 2, 11, 12, 13]. The first communication-efficient algorithm that achieves linear convergence over time-invariant (static) undirected graphs was proposed in [14].

The consensus problem can be viewed as a stepping stone towards more general decentralized optimization problems, where the nodes in a network aim to collaboratively minimize the sum of local objective functions. A number of solutions to this problem has been proposed for the setting where the network is undirected, including the well-known distributed (sub)gradient descent algorithm (DGD) [4, 15], distributed alternating direction method of multipliers (D-ADMM) [16], and decentralized dual averaging methods [17, 18, 19]. Recently, [20, 14] proposed a novel communication-efficient algorithm for decentralized convex optimization problems; the provably convergent algorithm relies on a message-passing scheme with memory and biased compression.

A key technical property required to ensure convergence of decentralized convex optimization algorithms over undirected networks is that the so-called mixing matrix characterizing the network connectivity is doubly-stochastic. However, in directed networks characterized by communication link asymmetry, doubly-stochastic mixing matrices are atypical. This motivated algorithms that rely on auxiliary variables to cancel out the imbalance in asymmetric directed networks in order to achieve convergence. For instance, the subgradient-push algorithm [21, 22] works with column-stochastic mixing matrices and introduces local normalization scalars to ensure converge. The directed distributed gradient descent (D-DGD) algorithm [23], on the other hand, keeps track of the variations of local models by utilizing auxiliary variables of the same dimension as the local model parameters. For convex objective functions, both algorithms achieve $\mathcal{O}(\frac{\ln T}{\sqrt{T}})$ convergence rate. When the objectives are strongly-convex with Lipshitz gradients, and assuming availability of only the stochastic gradient terms, the stochastic gradient-push algorithm proposed in [24] achieves $\mathcal{O}(\frac{\ln T}{T})$ convergence rate. A common feature of these algorithms is their reliance upon diminishing stepsizes to achieve convergence to the optimal solution; in comparison, using a fixed stepsize can accelerate the gradient search but cannot guarantee the exact convergence, only the convergence to a neighborhood of the optimal solution. The implied exactness-speed dilemma can be overcome using schemes that deploy gradient tracking (see, e.g., [6, 7, 25]). These schemes utilize fixed step sizes to achieve linear convergence rate when the objective functions are both smooth and strongly-convex. Among them, the Push-DIGing algorithm [7] follows the same basic ideas of the subgradient-push algorithm, while TV-AB [25] relies on column- and row-stochastic matrices to update model parameters and gradient terms, respectively.

The aforementioned linearly convergent methods rely on full gradient, i.e., each node is assumed to use all of its data to compute the local gradient. However, if the number of data points stored at each node is large, full gradient computation becomes infeasible. To this end, stochastic gradient descent was adapted to decentralized settings, but the resulting computational savings come at the cost of sublinear convergence rate [26, 24]. To accelerate traditional stochastic gradient methods in centralized settings, variance-reduction algorithms such as SAG [9] and SVRG [8] have been proposed; these schemes enable linear convergence when the objective functions are smooth and strongly-convex. In decentralized settings, GT-SVRG [26] and Network-SVRG [27] leverage variance-reduction techniques to achieve linear convergence rate. However, these algorithms are restricted to static and undirected networks, and a narrow class of directed networks where the mixing matrices can be rendered doubly stochastic.[1] The existing algorithms for decentralized optimization over directed networks, such as Push-SAGA [29], are restricted to static networks.

In recent years, decentralized learning tasks have experienced rapid growth in the amount of data and the dimension of the optimization problems, which may lead to practically infeasible demands for communication between network nodes. To this end, various communication compression schemes have been proposed; among them, the most frequently encountered are quantization and sparsification. Quantization schemes limit the number of bits encoding the messages, while the sparsification schemes select a subset of features and represent messages in lower dimensional space. For instance, [30, 31, 32, 20, 33, 34] propose algorithms for distributed training of (non)convex machine learning models in static master-worker settings (i.e., star graph topology) using quantized/compressed information, while [35, 14, 36, 37] develop communication-efficient algorithms for decentralized optimization over static and undirected networks. However, directed networks in general, and time-varying ones in particular, have received considerably less attention. Decentralized optimization over such networks faces technical challenges of developing an algorithmic framework conducive to theoretical analysis and establishing convergence guarantees, which is further exacerbated when the communication is compressed. Early steps in this direction were made in [38] by building upon the subgradient-push algorithm to develop a quantized communication framework for decentralized optimization over a static network.

Our proposed algorithm utilizes gradient tracking and variance reduction to achieve fast convergence, and relies on stochastic gradients to solve the decentralized convex optimization task at feasible computational cost. Preliminary results of this work, focused on a significantly slower full gradient framework ($\mathcal{O}(\frac{1}{\epsilon^2})$ vs. $\mathcal{O}(\ln \frac{1}{\epsilon})$), were presented in [1]. In Table I, we briefly summarize and contrast several algorithms for decentralized optimization over directed graphs.

B. Notation

We use lowercase bold letters to represent vectors and uppercase letters to represent matrices. $[A]_{ij}$ denotes the (i, j) entry of matrix A, while $\|\cdot\|$ denotes the standard Euclidean norm. The spectral radius of a matrix A is denoted by $\rho(A)$. The weighted infinity norm of $\mathbf{x}$ given a positive vector $\mathbf{w}$ is $\|\mathbf{x}\|_\infty^{\mathbf{w}} = \max_i |x_i|/w_i$ and the induced matrix norm is $|||\cdot|||_\infty^{\mathbf{w}}$. Finally, I denotes the identity matrix whose dimension is inferred from the context.

[1]To have doubly stochastic mixing matrices, directed graphs require weight balance, i.e., at each node of a graph, the sum of the weights from in-coming edges should be equal to that of the weights from out-coming edges [28].

TABLE I: The settings and convergence rates of algorithms for decentralized optimization over directed graphs.

Algorithm	Convergence	Digraph	Gradient	Convex Objective Setting	Compression
Subgradient-push [24]	$\mathcal{O}(\frac{1}{\epsilon})$	Time-varying	Stochastic	Strong convexity	✗
Push-DIGing [7]	$\mathcal{O}(\ln\frac{1}{\epsilon})$	Time-varying	Full	Strong convexity and smoothness	✗
TV-AB [25]	$\mathcal{O}(\ln\frac{1}{\epsilon})$	Time-varying	Full	Strong convexity and smoothness	✗
Quantized Push-sum [38]	$\mathcal{O}(\frac{1}{\epsilon^2})$	Static	Full	-	✓
This work	$\mathcal{O}(\ln\frac{1}{\epsilon})$	Time-varying	Stochastic	Strong convexity and smoothness	✓

II. Preliminaries

A. Problem Formulation

For convenience, we remind the reader of the problem formulation (1): In a network of n agents, where each node maintains a local model consisting of d parameters, the agents collaboratively solve the decentralized convex optimization

$$\min_{\mathbf{x}\in\mathbb{R}^d}\left[f(\mathbf{x}):=\frac{1}{n}\sum_{i=1}^{n}f_i(\mathbf{x})\right], \quad (2)$$

where $f_i(\mathbf{x})=\frac{1}{m_i}\sum_{j=1}^{m_i}f_{i,j}(\mathbf{x}):\mathbb{R}^d\to\mathbb{R}$ for $i\in[n]:=\{1,...,n\}$ denotes the local objective function at node i. Each component of f_i is assumed to be smooth and not accessible to nodes other than the i^{th} one, and the global objective f is assumed to be strongly-convex. We further assume existence of a unique optimal solution $\mathbf{x}^*\in\mathbb{R}^d$ and that each node can communicate to its neighbors; the nodes identify $\mathbf{x}^*$ by exchanging messages over a time-varying directed network. The network's connectivity properties are elaborated upon in Section 3.

B. Communication-Efficient Methods

In practice, bandwidth limitations may restrict the amount of data that the network nodes can communicate to each other; this is typical of high-dimensional scenarios where the dimension d of local parameters $\mathbf{x}_i$ is exceedingly large. To handle communication constraints, network nodes may employ *sparsification* to reduce the size of the messages. Typically, there are two approaches to sparsification: (i) each node selects and communicates d_q out of d components of a d-dimensional message; or (ii) each component of a d-dimensional message is selected to be communicated independently with probability d_q/d. Note that the former imposes a hard constraint on the number of communicated entries while the latter results in d_q communicated entries on expectation; both select a specific entry with probability d_q/d. Throughout this paper, we focus on and study the first approach.

Let $Q:\mathbb{R}^d\to\mathbb{R}^d$ denote the sparsification operator; we allow for biased Q with variance proportional to the argument norm, i.e., $\mathbb{E}[Q(\mathbf{x})]\neq\mathbf{x}$ and $\mathbb{E}[\|Q(\mathbf{x})-\mathbf{x}\|^2]\propto\|\mathbf{x}\|^2$. This stands in contrast to typical compression operators which aim to achieve no bias and have bounded variance (see, e.g., [30]). More recent works [20, 14, 36, 38] do consider biased compression operators but only for time-invariant communication networks – a setting that is more restrictive than the one considered in this paper.

III. Algorithm development

In this section, we first introduce a novel average consensus algorithm, an intermediate step towards the main (optimization) framework; then we present the optimization algorithm consisting of the consensus and the gradient components.

A. Foundations: Decentralized average consensus

We start by specifying a procedure for decentralized average consensus, an intermediate step towards decentralized optimization and, ultimately, an integral part thereof. The decentralized average consensus problem is formulated as the computation $\bar{\mathbf{x}}=\frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$, where $\mathbf{x}_i\in\mathcal{R}^d$ is the parameter vector at node i. Following the idea of [12], for each node of the network we define the so-called surplus vector, i.e., an auxiliary variable $\mathbf{y}_i\in\mathbb{R}^d$ which tracks local state vector variations over consecutive time steps; as shown later in this section, one can use the surplus vector to help provide guarantees of convergence to the optimal solution of the decentralized problem. The surplus vector is exchanged along with the state vector, i.e., at time t, node i sends both $\mathbf{y}_i^t$ and the state vector $\mathbf{x}_i^t$ to its out-neighbors. For the sake of having compact notation, let us introduce $\mathbf{z}_i^t\in\mathbb{R}^d$,

$$\mathbf{z}_i^t=\begin{cases}\mathbf{x}_i^t, & i\in\{1,...,n\}\\ \mathbf{y}_{i-n}^t, & i\in\{n+1,...,2n\}\end{cases}, \quad (3)$$

to represent messages node i communicates to its neighbors in the network at time t.

We assume that the time-varying graph is $\mathcal{B}$-jointly connected, i.e., that there exists a window size $\mathcal{B}\geq 1$ such that the aggregate graph $\bigcup_{l=t}^{t+\mathcal{B}-1}\mathcal{G}_l$ is strongly connected for all $t=k\mathcal{B}$, $k\in\mathcal{N}$. Note that if $\mathcal{B}=1$, each instance of the graph is strongly connected. This is a more general assumption than the often used $\mathcal{B}$-bounded strong-connectivity (see, e.g. [25]) which requires strong connectivity of the union graph $\bigcup_{l=t}^{t+\mathcal{B}-1}\mathcal{G}_l$ for all $t\geq 0$.

For $\mathcal{B}$-jointly connected graphs, the product of mixing matrices of graph instances over $\mathcal{B}$ consecutive time steps has a non-zero spectral gap. To formalize this statement, let us construct two weight matrices that reflect the network topology; in particular, let W_{in}^t (row-stochastic) and W_{out}^t (column-stochastic) denote the in-neighbor and out-neighbor weight matrices at time t, respectively. It holds that $[W_{in}^t]_{ij}>0$ if and only if $j\in\mathcal{N}_{in,i}^t$ and $[W_{out}^t]_{ij}>0$ if and only if $i\in\mathcal{N}_{out,j}^t$, where $\mathcal{N}_{in,i}^t$ denotes the set of nodes that may send information to node i (including i) whereas $\mathcal{N}_{out,j}^t$ denotes the set of nodes that may receive information from node j (including j) at time t. We assume W_{in}^t and W_{out}^t are given and that both $\mathcal{N}_{in,i}^t$ and $\mathcal{N}_{out,i}^t$ are known to node i. A common policy for designing W_{in}^t and W_{out}^t is to assign

$$[W_{in}^t]_{ij} = 1/|\mathcal{N}_{in,i}^t|, \qquad [W_{out}^t]_{ij} = 1/|\mathcal{N}_{out,j}^t|. \tag{4}$$

Recall that we are interested in sparsifying messages exchanged between nodes of a network; clearly, the sparsification should impact the structure of a mixing matrix. Indeed, if one attempts sparsifying messages used by existing methods, e.g. [21, 12, 13, 22], without any modifications of the mixing matrices therein, non-vanishing error terms induced by the compression operator will prevent those methods from converging. We note that the impact of sparsification on the components of a message vector is similar to the impact of link failures, and may therefore be captured by the structure of the weight matrices. To elaborate on this, observe that the vector-valued problem at time t can essentially be decomposed to d individual scalar-valued tasks with weight matrices $\{W_{in,m}^t\}_{m=1}^d$ and $\{W_{out,m}^t\}_{m=1}^d$. For the sparsified components of the message vector, i.e., those that are set to zero and not communicated, the corresponding entries in the weight matrices can be replaced by zeros; on the other hand, the entries in the weight matrices corresponding to the communicated components of the message vector remain unchanged, leading to the violation of the stochasticity of the weight matrices. To address this, we *re-normalize* the weight matrices $\{W_{in,m}^t\}_{m=1}^d$ and $\{W_{out,m}^t\}_{m=1}^d$, thus ensuring their row and column stochasticity. Note that the re-normalization of the i^{th} row of $\{W_{in,m}^t\}_{m=1}^d$ (i^{th} column of $\{W_{out,m}^t\}_{m=1}^d$) is performed by the i^{th} network node.

To specify the normalization rule, we first need to define the sparsification operation. Sparsification of $\mathbf{x}_i^t$ (and, consequently, $\mathbf{y}_i^t$) is done via the compression operator $Q(\cdot)$ applied to $\mathbf{z}_i^t$; we denote the result by $Q(\mathbf{z}_i^t)$. Let $[Q(\mathbf{z}_i^t)]_m$ denote the m^{th} component of $Q(\mathbf{z}_i^t)$. Let $\{A_m^t\}_{m=1}^d$ and $\{B_m^t\}_{m=1}^d$ be the weight matrices obtained after normalizing $\{W_{in,m}^t\}_{m=1}^d$ and $\{W_{out,m}^t\}_{m=1}^d$, respectively. To formalize the normalization procedure, we introduce the weight matrix

$$[A_m^t]_{ij} = \begin{cases} \frac{[W_{in,m}^t]_{ij}}{\sum_{j\in\mathcal{S}_m^t(i,j)}[W_{in,m}^t]_{ij}} & \text{if } j \in \mathcal{S}_m^t(i,j) \\ 0 & \text{otherwise,} \end{cases} \tag{5}$$

where $\mathcal{S}_m^t(i,j) := \{j | j \in \mathcal{N}_{in,i}^t, [Q(\mathbf{z}_j^t)]_m \neq 0\} \cup \{i\}$. Likewise, the weight matrix B_m^t is defined as

$$[B_m^t]_{ij} = \begin{cases} \frac{[W_{out,m}^t]_{ij}}{\sum_{i\in\mathcal{T}_m^t(i,j)}[W_{out,m}^t]_{ij}} & \text{if } i \in \mathcal{T}_m^t(i,j) \\ 0 & \text{otherwise,} \end{cases} \tag{6}$$

where $\mathcal{T}_m^t(i,j) := \{i | i \in \mathcal{N}_{out,j}^t, [Q(\mathbf{z}_i^t)]_m \neq 0\} \cup \{j\}$.

We can now define the *mixing matrix* of a directed network with sparsified messages.

Definition 1. *The m^{th} mixing matrix at time t of a time-varying directed network with sparsified messages, $\bar{M}_m^t \in \mathbb{R}^{2n\times 2n}$, is a matrix whose columns sum up to* 1 *and whose eigenvalues satisfy* $1 = |\lambda_1(\bar{M}_m^t)| = |\lambda_2(\bar{M}_m^t)| \geq |\lambda_3(\bar{M}_m^t)| \geq \cdots |\lambda_{2n}(\bar{M}_m^t)|$, *constructed from the current network topology as*

$$\bar{M}_m^t = \begin{bmatrix} A_m^t & \mathbf{0} \\ I - A_m^t & B_m^t \end{bmatrix}, \tag{7}$$

where A_m^t and B_m^t denote the m^{th} normalized in-neighbor and out-neighbor weight matrices at time t, respectively.

Algorithm 1 Directed Communication-Sparsified Average Consensus (Di-CS-AC)

```
1:  Input: $T$, $\mathbf{x}^0$, $\mathbf{y}^0 = \mathbf{0}$, $\gamma$
2:  set $\mathbf{z}^0 = [\mathbf{x}^0; \mathbf{y}^0]$, $\tilde{w}^0 = \mathbf{z}^0$ and
3:  for each $s \in [0, 1, ..., S]$ do
4:      generate non-negative matrices $\{W_{in,m}^t\}_{m=1}^d$ and $\{W_{out,m}^t\}_{m=1}^d$
5:      for each $m \in [1, ..., d]$ do
6:          construct a row-stochastic $A_m^t$ and a column-stochastic $B_m^t$ according to (5) and (6)
7:          construct $\bar{M}_m^t$ according to (7)
8:          for each $i \in [1, ..., 2n]$ do
9:              update $z_{im}^{t+1}$ according to (8)
10:         end for
11:     end for
12: end for
```

Given $\mathbf{z}_i^t$ and $\bar{M}_m^t$ in (3) and (7), respectively, we can formulate a compact recursive update rule for $\mathbf{z}_i^t$ as

$$z_{im}^{t+1} = \sum_{j=1}^{2n} [\bar{M}_m^t]_{ij}[Q(\mathbf{z}_j^t)]_m + \mathbb{1}_{\{t \bmod \mathcal{B}=\mathcal{B}-1\}}\gamma[F]_{ij} z_{jm}^{\mathcal{B}\lfloor t/\mathcal{B}\rfloor}, \tag{8}$$

where $F = \begin{bmatrix} \mathbf{0} & I \\ \mathbf{0} & -I \end{bmatrix}$ and m denotes the coordinate index.

As seen in (8), vectors $\mathbf{z}_i^t$ (which contain $\mathbf{x}_i^t$, the quantities to be averaged) are updated in a straightforward manner via sparsification and multiplication with the mixing matrix at all times t except those that satisfy

$$t \mod \mathcal{B} = \mathcal{B} - 1. \tag{9}$$

In particular, when (9) holds, vectors $\mathbf{z}_i^{\mathcal{B}\lfloor t/\mathcal{B}\rfloor}$, stored at time $\mathcal{B}\lfloor t/\mathcal{B}\rfloor$, are also used to update $\mathbf{z}_i^t$. The usage of the stored vectors is motivated by the observation that $\bar{M}_m^t$ may have zero spectral gap, which is undesirable since for such mixing matrices the convergence of the consensus algorithms will not be guaranteed. However, for a judiciously chosen perturbation parameter ϵ, which determines to what extent $\sum_{j=1}^{2n}[F]_{ij} z_{jm}^{\mathcal{B}\lfloor t/\mathcal{B}\rfloor}$ affects the update, we can ensure a nonzero spectral gap of the product of $\mathcal{B}$ consecutive mixing matrices starting from $t = k\mathcal{B}$. The described communication-sparsified average consensus procedure over directed graphs, referred to for convenience as Di-CS-AC, is formalized as Algorithm 1.

B. Decentralized gradient component

Going beyond the simple consensus problem and towards solving optimization (2), we re-define the recursive update rule for $\mathbf{z}_i^t$ as

$$z_{im}^{t+1} = \sum_{j=1}^{2n}[\bar{M}_m^t]_{ij}[Q(\mathbf{z}_j^t)]_m + \mathbb{1}_{\{t \bmod \mathcal{B}=\mathcal{B}-1\}}\gamma[F]_{ij}z_{jm}^{\mathcal{B}\lfloor t/\mathcal{B}\rfloor} - \mathbb{1}_{\{t \bmod \mathcal{B}=\mathcal{B}-1\}}\alpha g_{im}^{\mathcal{B}\lfloor t/\mathcal{B}\rfloor}, \tag{10}$$

where F and m denote the same objects as in (8), and g_{im} combines global gradient tracking with local stochastic reduction to achieve accelerated convergence (elaborated upon shortly). Note that (10) implies the following element-wise update rules for state and surplus vectors, respectively:

$$x_{im}^{t+1} = \sum_{j=1}^{n}[A_m^t]_{ij}[Q(\mathbf{x}_j^t)]_m + \mathbb{1}_{\{t \bmod \mathcal{B}=\mathcal{B}-1\}}\gamma y_{im}^{\mathcal{B}\lfloor t/\mathcal{B}\rfloor} - \mathbb{1}_{\{t \bmod \mathcal{B}=\mathcal{B}-1\}}\alpha g_{im}^{\mathcal{B}\lfloor t/\mathcal{B}\rfloor}, \tag{11}$$

$$y_{im}^{t+1} = \sum_{j=1}^{n}[B_m^t]_{ij}[Q(\mathbf{y}_j^t)]_m - (x_{im}^{t+1} - x_{im}^t). \tag{12}$$

Paralleling the basic consensus task discussed in the previous section, vectors $\mathbf{z}_i^t$ (containing state vectors to be averaged) are updated via sparsification and multiplication with the mixing matrix at all times t except those that satisfy (9). When (9) does hold, vectors $\mathbf{z}_i^{\mathcal{B}\lfloor t/\mathcal{B}\rfloor}$, stored at times $\mathcal{B}\lfloor t/\mathcal{B}\rfloor$, are also used to update $\mathbf{z}_i^t$; the motivation and reasoning for such special treatment are as same as in the consensus algorithm.[2]

In the proposed algorithm, updates of the gradient term $\mathbf{g}_i^t$ combine global gradient tracking with local stochastic variance reduction. In particular, the updates of $\mathbf{g}_i^t$ mix gradient messages while keeping track of the changes in gradient estimates $\mathbf{v}_i^t$; this guides $\mathbf{g}_i^t$ towards the gradient of the global objective, ultimately ensuring convergence to the optimal solution $\mathbf{x}^*$ (i.e., global gradient tracking helps avoid the pitfall of non-vanishing local gradients which would otherwise lead the search only to a neighborhood of $\mathbf{x}^*$). The mth entry of $\mathbf{g}_i^t$, g_{im}^t, is updated as

$$g_{im}^{\mathcal{B}(\lfloor t/\mathcal{B}\rfloor)} = \begin{cases} \sum_{j=1}^{n}[B_m(k\mathcal{B}-1:(k-1)\mathcal{B})]_{ij}g_{jm}^{(k-1)\mathcal{B}} \\ \quad +v_{im}^{\mathcal{B}(\lfloor t/\mathcal{B}\rfloor)} - v_{im}^{\mathcal{B}(\lfloor t/\mathcal{B}\rfloor-1)} & i \le n \\ 0 & \text{otherwise}, \end{cases} \tag{13}$$

where $k = \lfloor t/\mathcal{B}\rfloor$. The gradient estimate $\mathbf{v}_i^t$ in (13) is updated via the stochastic variance-reduction method [8]. Specifically,

$$\mathbf{v}_i^{\mathcal{B}(\lfloor t/\mathcal{B}\rfloor+1)} = \nabla f_{i,l_i}(\mathbf{z}_i^{\mathcal{B}\lfloor t/\mathcal{B}\rfloor}) - \nabla f_{i,l_i}(\tilde{w}_i) + \tilde{\mu}_i, \quad \forall i \in [n]. \tag{14}$$

One can interpret this update as being executed in a double loop fashion: when a local full gradient at node i, $\tilde{\mu}_i$, is computed (in what can be considered an outer loop), it is retained in the subsequent T iterations (the inner loop). In each iteration of the inner loop, if the time step satisfies (9), node i uniformly at random selects a local sample, l_i, for the calculation of two stochastic gradient estimates – an estimate of the current state, $\nabla f_{i,l_i}(\mathbf{z}_i^{\mathcal{B}\lfloor t/\mathcal{B}\rfloor})$, and an estimate of the state from the last outer loop, $\nabla f_{i,l_i}(\tilde{w}_i)$ – the terms needed to perform update of $\mathbf{v}_i^t$. By computing a full gradient periodically in the outer loop and estimating gradient stochastically in the inner loop, the described procedure trades computational cost for convergence speed, ultimately achieving linear convergence at fewer gradient computations per sample than the full gradient techniques.

The described procedure is formalized as Algorithm 2.

Remark 1. *We highlight a few important observations regarding Algorithm 2.*

(a) When there are no communication constraint and each agent in the network can send full information to out-neighboring agents, Algorithm 2 reduces to a novel stochastic variance-reduced scheme for decentralized convex optimization over such networks.

(b) For $\mathcal{B} = 1$, the problem reduces to decentralized optimization over networks that are strongly connected at all time steps, a typical connectivity assumption for many decentralized optimization algorithms [23, 14].

(c) Algorithm 2 requires each node in the network to store local vectors of size $4d$, including the current state vector, current and past surplus vector, and local gradient vector. While the current state vector and current surplus vector may be communicated to the neighboring nodes, past surplus vectors are only used locally to add local perturbations at the time steps satisfying (9).

(d) The columns of $\bar{M}_m^t$ sum up to one. However, $\bar{M}_m^t$ is not column-stochastic as it has negative entries, which stands in contrast to the stochasticity property of the mixing matrices appearing in the average consensus algorithms [39, 14].

IV. Convergence Analysis

For convenience, let us denote the product of a sequence of mixing matrices from time step s to T as

$$\bar{M}_m(T:s) = \bar{M}_m^T\bar{M}_m^{T-1}\cdots\bar{M}_m^s. \tag{15}$$

To further simplify notation, we also introduce

$$M_m((k+1)\mathcal{B}-1:k\mathcal{B}) = \bar{M}_m((k+1)\mathcal{B}-1:k\mathcal{B}) + \gamma F, \tag{16}$$

and

$$M_m(t:k_1\mathcal{B}) = \bar{M}_m(t:k_2\mathcal{B})M_m(k_2\mathcal{B}-1:(k_2-1)\mathcal{B})\cdots M_m((k_1+1)\mathcal{B}-1:k_1\mathcal{B}), \tag{17}$$

where $k_2\mathcal{B} \le t \le (k_2+1)\mathcal{B}-1$ and $k_1, k_2 \in \mathcal{N}, k_1 \le k_2$. Note that $M_m((k+1)\mathcal{B}-1:k\mathcal{B})$ is formed by adding a perturbation matrix γF to the product $\bar{M}_m((k+1)\mathcal{B}-1:k\mathcal{B})$. Finally, we also introduce shorthand notation for the product of the weight matrices B_m from time s to T,

$$B_m(T:s) = B_m^T B_m^{T-1}\cdots B_m^s. \tag{18}$$

Our analysis relies on several standard assumptions about the graph and network connectivity matrices as well as the characteristics of the local and global objective functions. These are given next.

Assumption 1. *Suppose the following conditions hold:*

[2]Note that F has all-zero matrices for its $(1,1)$ and $(2,1)$ blocks and thus we only need to store $\mathbf{z}_i^{\mathcal{B}\lfloor t/\mathcal{B}\rfloor}$ (equivalently, $\mathbf{y}_{i-n}^{\mathcal{B}\lfloor t/\mathcal{B}\rfloor}$), where $n+1 \le i \le 2n$.

Algorithm 2 Directed Communication-Sparsified Stochastic Variance-Reduced Gradient Descent (Di-CS-SVRG)

```
1: Input: $T$, $\mathbf{x}^0$, $\mathbf{y}^0 = \mathbf{0}$, $\alpha$, $\gamma$
2: set $\mathbf{z}^0 = [\mathbf{x}^0; \mathbf{y}^0]$, $\tilde{w}^0 = \mathbf{z}^0$ and $\mathbf{g}_i^0 = \mathbf{v}_i^0 = \nabla \mathbf{f}_i(\mathbf{x}_i^0) \quad \forall i \in [n]$
3: for each $s \in [0, 1, ..., S]$ do
4:   $\tilde{w} = \tilde{w}^s$
5:   $\tilde{\mu}_i = \nabla f_i(\tilde{w}) = \frac{1}{m_i} \sum_{j=1}^{m_i} \nabla f_{i,j}(\tilde{w})$
6:   for each $t \in [sT + 1, ..., (s+1)T - 1]$ do
7:     generate non-negative matrices $\{W_{in,m}^t\}_{m=1}^d$ and $\{W_{out,m}^t\}_{m=1}^d$
8:     for each $m \in [1, ..., d]$ do
9:       construct a row-stochastic $A_m^t$ and a column-stochastic $B_m^t$ according to (5) and (6)
10:      construct $\bar{M}_m^t$ according to (7)
11:      for each $i \in [1, ..., 2n]$ do
12:        update $z_{im}^{t+1}$ according to (10)
13:      end for
14:      if $t \mod \mathcal{B} = \mathcal{B} - 1$ then
15:        for each $i \in [1, ..., n]$ do
16:          select $l_i$ uniformly randomly from $[m_i]$:
17:            update $\mathbf{v}_i^{\mathcal{B}(\lfloor t/\mathcal{B} \rfloor + 1)}$ according to (14)
18:            update $g_{im}^{\mathcal{B}(\lfloor t/\mathcal{B} \rfloor + 1)}$ according to (13)
19:        end for
20:      end if
21:    end for
22:  end for
23:  $\tilde{w}^{s+1} = \mathbf{z}^{(s+1)T}$
24: end for
```

(a) *The product of consecutive mixing matrices $M_m((k+1)\mathcal{B} - 1 : k\mathcal{B})$ in (16) has a non-zero spectral gap for all $k \geq 0$, $1 \leq m \leq d$, and all $0 < \gamma < \gamma_0$ for some $0 < \gamma_0 < 1$.*

(b) *The collection of all possible mixing matrices $\{\bar{M}_m^t\}$ is a finite set.*

(c) *Each component of the local objective function $f_{i,j}$ is L-smooth and the global objective f is μ-strongly-convex*[3].

Remark 2. *Assumption 1(a) is readily satisfied for a variety of graph structures such as the $\mathcal{B}$-strongly connected directed graph introduced in [7], i.e., the setting where the union of graphs over B consecutive instances starting from $k\mathcal{B}$ forms a strongly connected graph for any non-negative integer k.*[4] *Furthermore, one can readily verify that Assumption 1(b) holds for the weight matrices defined in* (4).

Before stating the main theorem, we provide the following lemma which, under Assumption 1, establishes the consensus contraction of the product of mixing matrices and the product of normalized weight matrices.

[3]This implies that $\forall \mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^d$ there exists $L > 0$ such that $\|\nabla f_{i,j}(\mathbf{x}_1) - \nabla f_{i,j}(\mathbf{x}_2)\| \leq L\|\mathbf{x}_1 - \mathbf{x}_2\|$. Furthermore, $\forall \mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^d$ there exists $\mu > 0$ such that $f(\mathbf{x}_2) \geq f(\mathbf{x}_1) + \langle \nabla f(\mathbf{x}_1), \mathbf{x}_2 - \mathbf{x}_1 \rangle + \frac{\mu}{2}\|\mathbf{x}_2 - \mathbf{x}_1\|^2$.

[4]There are two versions of the definition of $\mathcal{B}$-strongly connected directed graphs, the difference being the window starting time. As noted in Section II, we consider the definition where the window may start at any time $t = k\mathcal{B}$; this differs from the (more demanding in regards to connectivity) definition in [25] where the starting time is an arbitrary non-negative integer.

Lemma 1. *Suppose Assumptions 1(a) and 1(b) hold. Let $\sigma = \max(|\lambda_{M,2}|, |\lambda_{B,2}|)$ denote the larger of the second largest eigenvalues of $M_m((k+1)\mathcal{B} - 1 : k\mathcal{B})$ and $B_m((k+1)\mathcal{B} - 1 : k\mathcal{B})$. Then,*

$$\|M_m((k+1)\mathcal{B} - 1 : k\mathcal{B})\mathbf{z} - \bar{\mathbf{z}}\| \leq \sigma\|\mathbf{z} - \bar{\mathbf{z}}\|, \ \forall \mathbf{z} \in \mathbb{R}^{2n} \quad \text{and}$$
$$\|B_m((k+1)\mathcal{B} - 1 : k\mathcal{B})\mathbf{y} - \bar{\mathbf{y}}\| \leq \sigma\|\mathbf{y} - \bar{\mathbf{y}}\|, \ \forall \mathbf{y} \in \mathbb{R}^n, \tag{19}$$

where $\bar{\mathbf{z}} = [\frac{1}{n}\sum_{i=1}^{2n} z_i, \cdots, \frac{1}{n}\sum_{i=1}^{2n} z_i]^T$ and $\bar{\mathbf{y}} = [\frac{1}{n}\sum_{i=1}^{n} y_i, \cdots, \frac{1}{n}\sum_{i=1}^{n} y_i]^T$.

Proof. The proof of the lemma is in the supplementary document. ∎

Our main result, stated in Theorem 1 below, establishes that Algorithm 2 provides linear convergence of local parameters to their average values, which itself converges linearly to the optimal solution of (1).

Theorem 1. *Suppose Assumption 1 holds. Denote the condition number of f by $\tilde{Q} = \frac{L}{\mu}$. If the step size α is chosen according to*

$$\alpha = \frac{(1 - \sigma^2)^2}{187\tilde{Q}L}, \tag{20}$$

the iterates of Algorithm 2 satisfy

$$\frac{1}{n}\sum_{i=1}^{n} \mathbb{E}\|\bar{\mathbf{z}}^{ST} - \mathbf{z}_i^{ST}\|^2 + \mathbb{E}\|\bar{\mathbf{z}}^{ST} - \mathbf{x}^*\|^2 \leq 2\lambda^S \times \left(\frac{1}{n}\sum_{i=1}^{n} \|\bar{\mathbf{z}}^0 - \mathbf{z}_i^0\|^2 + \|\bar{\mathbf{z}}^0 - \mathbf{x}^*\|^2 + \frac{(1 - \sigma^2)^2}{1457nL^2}\sum_{i=1}^{n}\sum_{m=1}^{d} \mathbb{E}|g_{im}^0 - \bar{g}_m^0|^2\right), \tag{21}$$

where

$$\lambda = 8\tilde{Q}^2 \exp\left(-\frac{(1 - \sigma^2)^2 T}{748\tilde{Q}^2}\right) + 0.66, \tag{22}$$

$\bar{\mathbf{z}}^t = \frac{1}{n}\sum_{i=1}^{2n} \mathbf{z}_i^t$, *and* $\bar{\mathbf{g}}^t = \frac{1}{n}\sum_{i=1}^{n} \mathbf{g}_i^t$.

Corollary 1.1. *Instate the notation and hypotheses of Theorem 1. If, in addition, the inner-loop duration T is chosen as*

$$T = \mathcal{B}\lceil \frac{1496\tilde{Q}^2}{(1 - \sigma^2)^2\mathcal{B}} \ln(200\tilde{Q}^2) \rceil, \tag{23}$$

the proposed algorithm achieves a linear convergence rate. Furthermore, to reach an ϵ-accurate solution, Algorithm 2 takes at most $\mathcal{O}(\frac{\tilde{Q}^2\mathcal{B}\ln\tilde{Q}}{(1-\sigma^2)^2}\ln 1/\epsilon)$ communication rounds and performs $\mathcal{O}((\frac{\tilde{Q}^2\ln\tilde{Q}}{(1-\sigma^2)^2} + \max_i\{m_i\})\ln 1/\epsilon)$ stochastic gradient computations.

Proof. It is straightforward to verify that for the stated value of T, $\lambda \leq 0.7 < 1$ and thus the algorithm converges linearly. ∎

Note that due to the gradient tracking step in Algorithm 2, in particular when constructing the linear system of inequalities (which includes the gradient tracking error), the rate of convergence is dependent upon $\tilde{Q}^2$ (through the factor in the coefficient matrix).

Remark 3. *Clearly, the communication cost of Algorithm 2 depends on the level of sparsification, i.e., the value of parameter d_q. Intuitively, if agents communicate fewer entries in each round, the communication cost per round decreases but the algorithm may take more rounds to reach the same accuracy. Therefore, the total communication cost until reaching a pre-specified ϵ-accuracy, found as the product of the number of communication rounds and the cost per round, is of interest. Let q denote the fraction of entries being communicated per iteration; smaller q implies more aggressive sparsification. This compression level parameter, q, impacts σ in Theorem 1; in particular, for a fixed network connectivity parameter $\mathcal{B}$, smaller q leads to sparser mixing matrices and, consequently, greater σ. Note that large $\mathcal{B}$ may be caused by sparsity of the instances of a time-varying network, thus leading to large values of σ.*

Remark 4. *It is worthwhile discussing and comparing the constants in Corollary 1.1 to those in the original SVRG [8] (centralized optimization) and GT-SVRG [26] (decentralized optimization over undirected graphs). For SVRG, this constant is $O(\frac{1}{\mu\alpha(1-2L\alpha)T}+\frac{2L\alpha}{1-2L\alpha})$, where α denotes the step size, L is the smoothness parameter, μ is the strong convexity parameter and T denotes the inner loop duration [8]. For both GT-SVRG and our proposed algorithm, the inner loop duration is $T = O(\frac{\tilde{Q}^2\log\tilde{Q}}{(1-\sigma)^2})$ and the linear convergence constant $O(\tilde{Q}^2\exp(-\frac{(1-\sigma^2)^2T}{\tilde{Q}^2}))$, where $\tilde{Q}$ is the condition number and σ is specified by the network topology and the applied compression, i.e., sparsification of the communicated quantities.*

A. Proof of Theorem 1

In this section, we prove Theorem 1 by analyzing various error terms that collectively impact the convergence rate of Algorithm 2. The main technical challenge in deriving the linear convergence result is the analysis of the vanishing errors formally introduced in the next paragraph: the consensus error, the optimality error and the gradient tracking error. Note that unlike in undirected graphs, the mixing matrices of directed graphs are not necessarily doubly stochastic; as a result, decentralized optimization schemes may produce state vectors that converge to a weighted average, rather than a consensus average. Furthermore, recall that in order to accelerate the convergence, we deploy two techniques: global gradient tracking and local variance reduction. Both the gradient tracking technique, which relies on the communication of gradient information over the network, and the variance reduction trick increase the difficulty of analyzing the vanishing gradient tracking error. The combination of these issues renders the analysis of the aforementioned errors challenging.

Specifically, the convergence rate depends on: (i) the expected consensus error, i.e., the expected squared difference between local vectors and the average vectors at time $(k+1)\mathcal{B}$, $\mathbb{E}[|z_{im}^{(k+1)\mathcal{B}}-\bar{z}_m^{(k+1)\mathcal{B}}|^2]$; (ii) the expected optimality error, i.e., the expected squared difference between the average vectors and the optimal vector, $\mathbb{E}[\|\bar{\mathbf{z}}^{(k+1)\mathcal{B}}-\mathbf{x}^*\|^2]$; and (iii) the expected gradient tracking error, $\mathbb{E}[\sum_{m=1}^d\sum_{i=1}^n|g_{im}^{(k+1)\mathcal{B}}-\bar{g}_m^{(k+1)\mathcal{B}}|^2$. Hence, it is critical to determine evolution of these sequences. Note that compared to the gradient-tracking based work, e.g. [7, 26], analyzing the proposed scheme is more involved due to its reliance upon a combination of variance reduction techniques and a communication-sparsified consensus; showing that the novel scheme achieves linear convergence on general directed time-varying graphs despite sparsified communication calls for a careful examination of the error terms in a manner distinct from the analysis found in prior work.

Dynamics of the aforementioned errors are clearly interconnected. Consequently, our analysis relies on deriving recursive bounds for the errors in terms of the linear combinations of their past values. The results are formally stated in Lemma 2, Lemma 3, and Lemma 4. Proofs of these lemmas are provided in the supplementary document.

Lemma 2. *Suppose Assumption 1 holds. Then $\forall i\le n$, $k\ge 0$ and $0<m\le d$, updates generated by Algorithm 2 satisfy*

$$\begin{aligned}\mathbb{E}[|z_{im}^{(k+1)\mathcal{B}}-\bar{z}_m^{(k+1)\mathcal{B}}|^2] \le{}& \frac{1+\sigma^2}{2}\mathbb{E}[|z_{im}^{k\mathcal{B}}-\bar{z}_m^{k\mathcal{B}}|^2]\\ &+\frac{2\alpha^2}{1-\sigma^2}\mathbb{E}[|g_{im}^{k\mathcal{B}}-\bar{g}_m^{k\mathcal{B}}|^2].\end{aligned}\tag{24}$$

Having established in Lemma 2 a recursive bound on the expected consensus error, we proceed by stating in Lemmas 3 and 4 recursive bounds on the expected optimality and gradient tracking errors, respectively. First, let us introduce (for $k\ge 0$)

$$\begin{aligned}\bar{\tau}^{k\mathcal{B}} &= \frac{1}{n}\sum_{i=1}^n\tau_i^{k\mathcal{B}},\\ \tau_i^{(k+1)\mathcal{B}} &= \begin{cases}\mathbf{x}_i^{(k+1)\mathcal{B}} & \text{if}\quad (k+1)\mathcal{B}\mod T=0\\ \tilde{w}_i & \text{otherwise.}\end{cases}\end{aligned}\tag{25}$$

Lemma 3. *Suppose Assumption 1 holds and let $0<\alpha<\frac{\mu}{8L^2}$. Then for all $k>0$ it holds that*

$$\begin{aligned}\mathbb{E}[n\|\bar{\mathbf{z}}^{(k+1)\mathcal{B}}-\mathbf{x}^*\|^2]\le{}&\frac{2L^2\alpha}{\mu}\mathbb{E}[\sum_{i=1}^n\|\bar{\mathbf{z}}^{k\mathcal{B}}-\mathbf{z}_i^{k\mathcal{B}}\|^2]\\ &+(1-\frac{\mu\alpha}{2})\mathbb{E}[n\|\bar{\mathbf{z}}^{k\mathcal{B}}-\mathbf{x}^*\|^2]\\ &+\frac{4L^2\alpha^2}{n}\mathbb{E}[\sum_{i=1}^n\|\tau_i^{k\mathcal{B}}-\bar{\tau}^{k\mathcal{B}}\|^2]\\ &+\frac{4L^2\alpha^2}{n}\mathbb{E}[n\|\bar{\tau}^{k\mathcal{B}}-\mathbf{x}^*\|^2].\end{aligned}\tag{26}$$

Lemma 4. *Suppose Assumption 1 holds. Then*

$$\begin{aligned}&\frac{1}{L^2}\mathbb{E}[\sum_{m=1}^{d}\sum_{i=1}^{n}|g_{im}^{(k+1)\mathcal{B}}-\bar{g}_m^{(k+1)\mathcal{B}}|^2]\\ &\leq \frac{120}{1-\sigma^2}\mathbb{E}[\sum_{i=1}^{n}\|\bar{\mathbf{z}}^{k\mathcal{B}}-\mathbf{z}_i^{k\mathcal{B}}\|^2]\\ &\quad+\frac{89}{1-\sigma^2}\mathbb{E}[n\|\bar{\mathbf{z}}^{k\mathcal{B}}-\mathbf{x}^*\|^2]\\ &\quad+\frac{3+\sigma^2}{4}\mathbb{E}[\frac{\sum_{m=1}^{d}\sum_{i=1}^{n}|g_{im}^{k\mathcal{B}}-\bar{g}_m^{k\mathcal{B}}|^2}{L^2}]\\ &\quad+\frac{38}{1-\sigma^2}\mathbb{E}[\sum_{i=1}^{n}\|\bar{\mathbf{z}}^{k\mathcal{B}}-\mathbf{z}_i^{k\mathcal{B}}\|^2]\\ &\quad+\frac{38}{1-\sigma^2}\mathbb{E}[n\|\bar{\tau}^{k\mathcal{B}}-\mathbf{x}^*\|^2].\end{aligned} \tag{27}$$

We proceed by defining a system of linear inequalities involving the three previously discussed error terms; study of the conditions for the geometric convergence of the powers of the resultant matrix in the system of linear inequalities leads to the linear convergence result in Theorem 1. To this end, we first state Proposition 1 whose proof follows by combining and re-organizing the inequalities in Lemmas 2-4 in a matrix form.

Proposition 1. *Suppose Assumption 1 holds. Define*

$$\mathbf{u}^{k\mathcal{B}}=\begin{bmatrix}\mathbb{E}[\sum_{i=1}^{n}\|\bar{\mathbf{z}}^{k\mathcal{B}}-\mathbf{z}_i^{k\mathcal{B}}\|^2]\\ \mathbb{E}[n\|\bar{\mathbf{z}}^{k\mathcal{B}}-\mathbf{x}^*\|^2]\\ \mathbb{E}[\frac{\sum_{m=1}^{d}\sum_{i=1}^{n}|g_{im}^{k\mathcal{B}}-\bar{g}_m^{k\mathcal{B}}|^2}{L^2}]\end{bmatrix}, \tag{28}$$

$$\tilde{\mathbf{u}}^{k\mathcal{B}}=\begin{bmatrix}\mathbb{E}[\sum_{i=1}^{n}\|\tau_i^{k\mathcal{B}}-\bar{\tau}^{k\mathcal{B}}\|^2]\\ \mathbb{E}[n\|\bar{\tau}^{k\mathcal{B}}-\mathbf{x}^*\|^2]\\ \mathbf{0}\end{bmatrix}, \tag{29}$$

$$J_\alpha=\begin{bmatrix}\frac{1+\sigma^2}{2} & 0 & \frac{2\alpha^2L^2}{1-\sigma^2}\\ \frac{2L^2\alpha}{\mu} & 1-\frac{\mu\alpha}{2} & 0\\ \frac{120}{1-\sigma^2} & \frac{89}{1-\sigma^2} & \frac{3+\sigma^2}{4}\end{bmatrix}, \tag{30}$$

$$H_\alpha=\begin{bmatrix}0 & 0 & 0\\ \frac{4L^2\alpha^2}{n} & \frac{4L^2\alpha^2}{n} & 0\\ \frac{38}{1-\sigma^2} & \frac{38}{1-\sigma^2} & 0\end{bmatrix}. \tag{31}$$

If $0\leq\alpha\leq\frac{\mu(1-\sigma^2)}{14\sqrt{2}L^2}$, *then for any* $k\geq 0$ *it holds that*

$$\mathbf{u}^{(k+1)\mathcal{B}}\leq J_\alpha\mathbf{u}^{k\mathcal{B}}+H_\alpha\tilde{\mathbf{u}}^{k\mathcal{B}}. \tag{32}$$

As a direct consequence of Proposition 1, for the iterations of the inner loop of Algorithm 2, for all $k\in[s\lfloor T/\mathcal{B}\rfloor,(s+1)\lfloor T/\mathcal{B}\rfloor-1]$ it holds that

$$\mathbf{u}^{(k+1)\mathcal{B}}\leq J_\alpha\mathbf{u}^{k\mathcal{B}}+H_\alpha\mathbf{u}^{sT}, \tag{33}$$

while for the outer loop of Algorithm 2 it holds for all $\forall s\geq 0$,

$$\mathbf{u}^{(s+1)T}\leq(J_\alpha^T+\sum_{l=0}^{T-1}J_\alpha^lH_\alpha)\mathbf{u}^{sT}. \tag{34}$$

Now, to guarantee linear decay of the outer loop sequence, we restrict the range of the inner loop duration T and the step size α according to

$$\rho(J_\alpha^T+\sum_{l=0}^{T-1}J_\alpha^lH_\alpha)<1, \tag{35}$$

where $\rho(\cdot)$ denotes the spectral radius of its argument.

In Lemma 5 below, we establish the range of α such that the weighted matrix norms of J_α^T and $\sum_{l=0}^{T-1}J_\alpha^lH_\alpha$ are small, thereby ensuring the geometric convergence of the powers of these matrices to $\mathbf{0}$.

Lemma 5. *Suppose Assumption 1 holds and let* $0<\alpha\leq\frac{(1-\sigma^2)^2}{187\tilde{Q}L}$, *where* $\tilde{Q}=\frac{L}{\mu}$. *Then*

$$\rho(J_\alpha)<|||J_\alpha|||_\infty^\delta<1-\frac{\mu\alpha}{4} \tag{36}$$

and

$$|||\sum_{l=0}^{T-1}J_\alpha^lH_\alpha|||_\infty^{\mathbf{q}}\leq|||(I-J_\alpha)^{-1}H_\alpha|||_\infty^{\mathbf{q}}<0.66, \tag{37}$$

where $\delta=\left[1,8\tilde{Q}^2,\frac{6656\tilde{Q}^2}{(1-\sigma^2)^2}\right]$ *and* $\mathbf{q}=[1,1,\frac{1457}{(1-\sigma^2)^2}]$.

Essentially, Lemma 5 establishes the range of the stepsize α such that the matrices involved in the system of linear inequalities (34) have small norm. Upon setting $\alpha=\frac{(1-\sigma^2)^2}{187\tilde{Q}L}$ (i.e., assigning α the largest feasible value), we proceed to determine the number of iterations in the outer loop such that the powers of matrix $J_\alpha^T+\sum_{l=0}^{T-1}J_\alpha^lH_\alpha$ in (34), and hence the components of $\mathbf{u}$ (i.e. the error terms), converge to zero at a geometric rate.

To this end, note that since J_α is non-negative, $\sum_{l=0}^{T-1}J_\alpha^l\leq\sum_{l=0}^{\infty}J_\alpha^l=(I-J_\alpha)^{-1}$. Hence, for all $s\geq 0$ it holds

$$\mathbf{u}^{(s+1)T}\leq(J_\alpha^T+(I-J_\alpha)^{-1}H_\alpha)\mathbf{u}^{sT}. \tag{38}$$

Since $\alpha=\frac{(1-\sigma^2)^2}{187QL}$, we may write

$$\begin{aligned}\|\mathbf{u}^{(s+1)T}\|_\infty^{\mathbf{q}}&\leq|||J_\alpha^T+(I-J_\alpha)^{-1}H_\alpha|||_\infty^{\mathbf{q}}\|\mathbf{u}^{sT}\|_\infty^{\mathbf{q}}\\ &\leq(|||J_\alpha^T|||_\infty^{\mathbf{q}}+0.66)\|\mathbf{u}^{sT}\|_\infty^{\mathbf{q}}\\ &\leq(8\tilde{Q}^2(|||J_\alpha^T|||_\infty^\delta)^T+0.66)\|\mathbf{u}^{sT}\|_\infty^{\mathbf{q}}\\ &\leq(8\tilde{Q}^2\exp(-\frac{(1-\sigma^2)^2T}{748\tilde{Q}^2})+0.66)\|\mathbf{u}^{sT}\|_\infty^{\mathbf{q}}\\ &:=\lambda\|\mathbf{u}^{sT}\|_\infty^{\mathbf{q}}.\end{aligned} \tag{39}$$

The result in (21) follows simply by noting the definitions of $\mathbf{u}^{sT}$ and the $\|\cdot\|_\infty^{\mathbf{q}}$ norm. Therefore, the proof of Theorem 1 is complete.

V. Experiments

In this section, we report results of benchmarking the proposed Algorithm 2; for convenience, we refer to Algorithm 2 as Di-CS-SVRG (Directed Communication-Sparsified Stochastic Variance-Reduced Gradient Descent). The results for the proposed Algorithm 1 are presented in the supplementary material.

We start with a network consisting of 10 nodes with randomly generated time-varying connections while ensuring strong connectivity at each time step. The construction begins with the Erdős–Rényi model [40] where each edge exists independently with probability 0.9; then, 2 directed edges are dropped from each strongly connected graph, leading to directed graphs. Building upon this basic structure, we can design networks with different connectivity profiles. Recall that the window size parameter $\mathcal{B}$, introduced in Assumption 1(a), may imply that the union graph over $\mathcal{B}$ consecutive instances, starting from any instance that is a multiple of $\mathcal{B}$, forms an almost-surely strongly connected Erdős–Rényi graph. When $\mathcal{B} = 1$, the network is strongly connected at each time step. The parameter q, the fraction of entries being communicated to neighboring nodes, characterizes the level of message sparsification; $q = 1$ implies communication without compression, while $q = 0$ indicates there is no communication in the network.

A. Decentralized Optimization Problem

We test the proposed Di-CS-SVRG on two tasks, linear and logistic regression, and benchmark it against several baseline algorithms. In particular, we compare Di-CS-SVRG with Decentralized Stochastic Gradient Descent (De-Stoc, stochastic variant), Push-DIGing (Push-DIG-Full) [7], Push-DIGing Stochastic (Push-DIG-Stoc, stochastic variant) and the TV-AB algorithm (AB Algorithm) [25]. In addition, we include comparisons to the full gradient schemes we considered in our preliminary work [1], Decentralized Full Gradient Descent (De-Full) and its communication-sparsified variant Sparsified De-Full (S-De-Full).

1) Decentralized Linear Regression: We consider the setting where n nodes collaboratively solve the optimization problem

$$\min_{\mathbf{x}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \|\mathbf{y}_i - D_i \mathbf{x}\|^2 \right\}, \tag{40}$$

where for the i^{th} node $D_i \in \mathbb{R}^{200\times 64}$ denotes the matrix with 200 local samples of size $d = 64$, and $\mathbf{y}_i \in \mathbb{R}^{200}$ denotes the corresponding measurement vector. The true value of $\mathbf{x}$, $\mathbf{x}^*$, is generated from a normal distribution, and the samples are generated synthetically. The measurements are generated as $\mathbf{y}_i = M_i \mathbf{x}^* + \eta_i$, where the entries of M_i are generated randomly from the standard normal distribution and then M_i is normalized such that its rows sum up to one. The local noise vector η_i is drawn at random from a zero-mean Gaussian distribution with variance 0.01. All algorithms are initialized using randomly generated local vectors $\mathbf{x}_i^0$, and utilize constant step size $\alpha_t = 0.002$ except the AB algorithm for which we followed the recommendation in [25] and set $\alpha_t = 0.0025$.

Performance of the algorithms is characterized using three metrics: residual over iterations, residual over average gradient computation and residual over communication cost, where the residual is computed as $\frac{\|\mathbf{x}^t - \mathbf{x}^*\|}{\|\mathbf{x}^0 - \mathbf{x}^*\|}$. The results are shown in Fig. 1. As seen in Fig. 1(a), Di-CS-SVRG (i.e., our Algorithm 2) with $q = 1$ converges at a linear rate and, while being a stochastic gradient algorithm, reaches the same residual floor as the full gradient method Push-DIG-Full and the AB algorithm. Di-CS-SVRG converges much faster than the two baseline algorithms employing SGD, Push-DIG-Stoc and De-Stoc. Fig. 1(b) shows that Di-CS-SVRG with varied compression levels q converges to the same residual floor, and that (as expected) larger q leads to faster convergence. Moreover, the figure shows that for a fixed q, Di-CS-SVRG achieves faster convergence than the benchmark algorithms.

Fig. 1(c) compares different algorithms in terms of the number of gradients computed per sample, demonstrating computation efficiency of Di-CS-SVRG. Finally, Fig. 1(d) shows the communication cost for varied q, computed as the total number of the (state, surplus and gradient) vector entries communicated across the network. As seen in the figure, to achieve a pre-specified level of the residual, Di-CS-SVRG with $q = 0.05$ incurs smaller communication cost than any other considered algorithm.

2) Decentralized Logistic Regression: To perform benchmarking on a logistic regression task, we solve a multi-class classification problem on the Stackoverflow dataset [41].

$$\min_{\mathbf{x}} \left\{ \frac{\mu}{2}\|\mathbf{x}\|^2 + \sum_{i=1}^{n}\sum_{j=1}^{N} \ln(1 + \exp(-(\mathbf{m}_{ij}^T \mathbf{x})\mathbf{y}_{ij})) \right\}, \tag{41}$$

where the training samples $(\mathbf{m}_{ij}, \mathbf{y}_{ij}) \in \mathbb{R}^{400+5}$, $\mathbf{m}_{ij}$ represents a vectorized text feature and $\mathbf{y}_{ij}$ represents the corresponding tag vector. We compare the performance of Di-CS-SVRG with the same benchmarking algorithms as in the linear regression problem, and use the same initialization setup. The logistic regression experiment is run with the stepsize $\alpha_t = 0.01$; the regularization parameter is set to $\mu = 10^{-5}$.

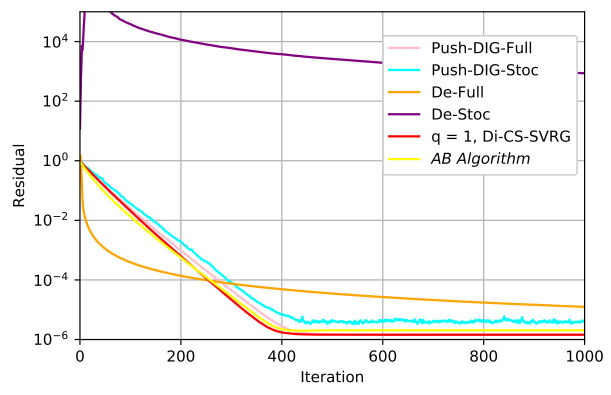
Performance of the algorithms on the logistic regression task is characterized by the classification correct rate. In particular, we evaluate the following three metrics: the correct rate vs. iterations, the correct rate vs. average gradient computation, and the correct rate vs. communication cost; they are all shown in Fig. 2. As seen in Fig. 2(a), Di-CS-SVRG converges and reaches the same residual floor as the full gradient method Push-DIG-Full and the AB Algorithm. Di-CS-SVRG converges much faster than then the two algorithms that rely on SGD, Push-DIG-Stoc and De-Stoc. Fig. 2(b) shows that for varied compression levels q, Di-CS-SVRG converges to the same residual floor. As expected, larger q leads to faster convergence. For a fixed q, Di-CS-SVRG converges faster than the benchmark algorithm.

Fig. 2(c) reports the average gradient computation, i.e., the number of gradients computed per sample. As can be seen, Di-CS-SVRG with different compression levels uses fewer gradient computation than the full gradient schemes (Push-DIG-Full, the AB algorithm and De-Full) to reach 90% correct classification rate.

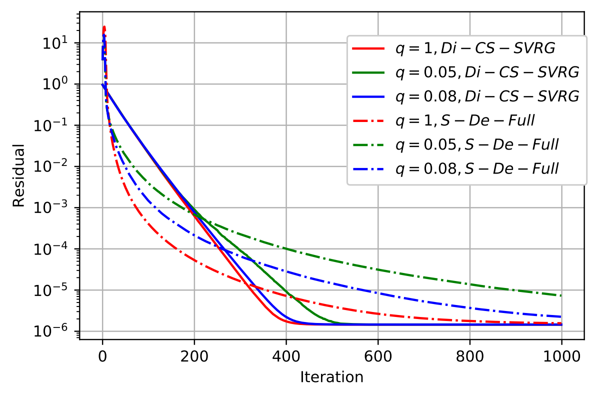
Fig. 2(d) shows the communication cost, defined as the total number of the (state, surplus and gradient) vector entries exchanged across the network, for various values of q. Among the considered schemes, Di-CS-SVRG with $q = 0.08$ reaches the pre-specified residual level with smaller communication cost than any stochastic scheme (Push-DIG-Stoc, De-Stoc as well as Di-CS-SVRG with other q's). Even though Push-DIG-Stoc reaches various accuracies slightly faster than Di-CS-SVRG, it requires considerably higher amount of gradient computation to do so (see Fig. 2).

B. Results on different network topologies

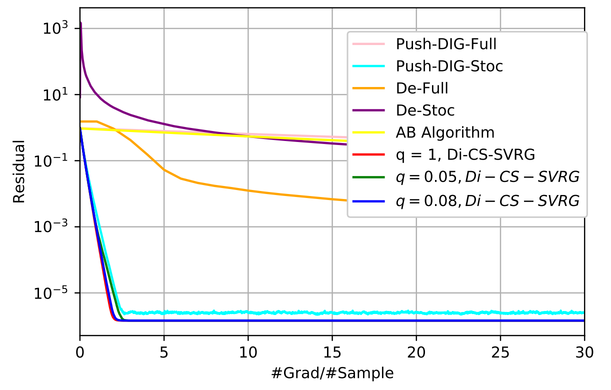
To further test Di-CS-SVRG in different settings, we apply it to decentralized optimization over networks with varied connectivity and sizes.



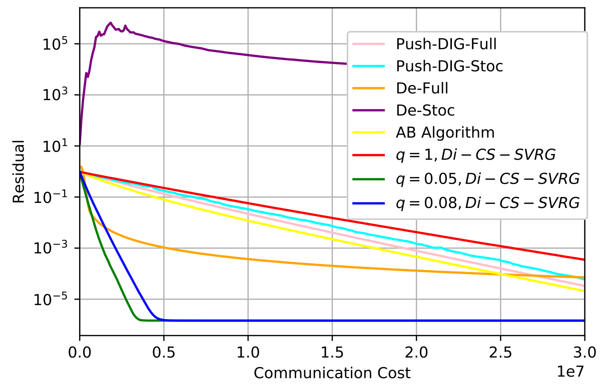
(a) Residual vs. iterations: full communication schemes.



(b) Residual vs. iterations: compressed communication schemes.



(c) Residual vs. gradient computations per sample.



(d) Residual vs. communication cost.

Fig. 1: Linear regression, $\mathcal{B} = 5$. (a) The residual achieved by full communication schemes and the residual of Di-CS-SVRG (Algorithm 2) with $q = 1$ vs. iterations. (b) The residual achieved by Di-CS-SVRG with different compression levels: $q = 1$, $q = 0.08$, and $q = 0.05$ vs. iterations. (c) The cumulative number of gradient computations needed to reach given level of the residual; showing both the compressed as well as full communication schemes. (d) The cumulative communication cost to reach given level of the residual; showing both the compressed as well as full communication schemes.

1) Varied network connectivity: We consider the linear regression problem and vary the values of the joint connectivity parameter $\mathcal{B}$. Fig. 3(a) shows the resulting residuals; as seen there, larger $\mathcal{B}$, implying the network takes longer time before the union of its instances forms a strongly connected graph, leads to slower convergence.

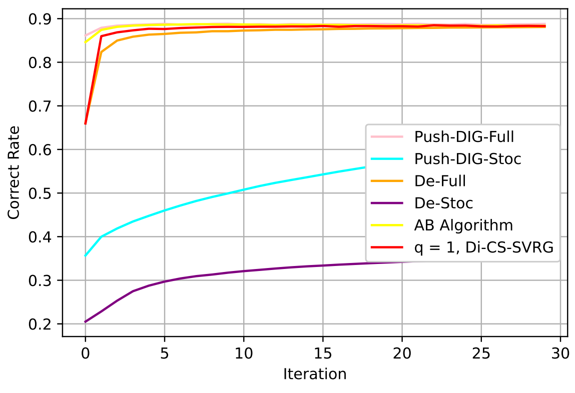
2) Varied network size: We now consider the logistic regression problem over networks of varied sizes. In particular, we fix the total number of data points and vary the number of nodes in the network. As the network grows, i.e., the number of nodes in the network becomes larger, each agent has fewer locally available data points. Fig. 3(b) shows the correct rate for compression levels $q = 1$, $q = 0.12$ and $q = 0.08$ as n grows from 10 to 30. For a pre-specified sparsification level, larger networks, in which each agent has fewer data points to train its local model, requires more communication rounds and therefore takes longer to converge.

VI. Conclusion

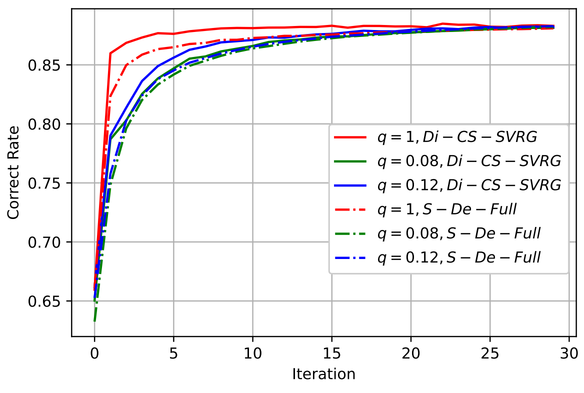
In this paper we studied decentralized convex optimization problems over time-varying directed networks and proposed a stochastic variance-reduced algorithm for solving them. The algorithm sparsifies messages exchanged between network nodes thus enabling collaboration in resource-constrained settings. We proved that the proposed algorithm, Di-CS-SVRG, enjoys linear convergence rate, and demonstrated its efficacy through simulation studies where it achieved the same accuracy as the best performing existing methods but at much lower communication and computation cost. As part of the future work, it is of interest to extend this work to decentralized non-convex optimization problems.

References

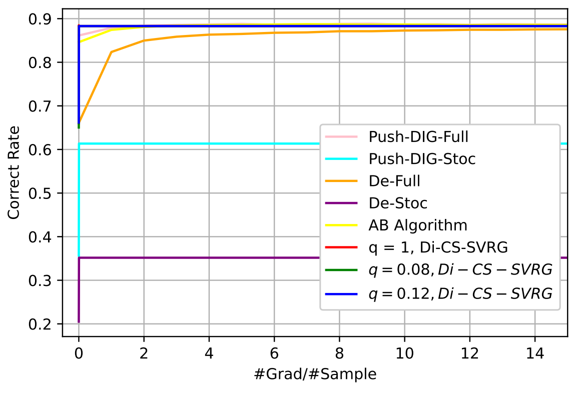
[1] Yiyue Chen, Abolfazl Hashemi, and Haris Vikalo. “Decentralized Optimization on Time-Varying Directed Graphs Under Communication Constraints”. In: *ICASSP 2021-2021 IEEE International Conference on Acoustics,*



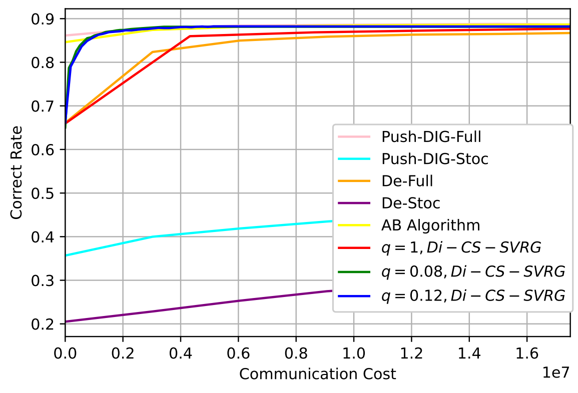
(a) Correct rate vs. iterations: full communication schemes.



(b) Correct rate vs. iterations: compressed schemes.



(c) Correct rate vs. gradient computations per sample.



(d) Correct rate vs. communication cost.

Fig. 2: Logistic regression, $\mathcal{B} = 1$. (a) The correct classification rate achieved by full communication schemes and the correct classification rate of Di-CS-SVRG vs. iterations. (b) The correct classification rate for Di-CS-SVRG with varied compression levels: $q = 1$, $q = 0.12$, $q = 0.08$ vs. iterations. (c) The cumulative number of gradient computations required to reach given levels of correct classification rate. (d) The cumulative communication cost required to reach given level of the correct classification rate.

Speech and Signal Processing (ICASSP). IEEE. 2021, pp. 3670–3674.

[2] Wei Ren and Randal W Beard. “Consensus seeking in multiagent systems under dynamically changing interaction topologies”. In: *IEEE Transactions on automatic control* 50.5 (2005), pp. 655–661.

[3] Ali Jadbabaie, Jie Lin, and A Stephen Morse. “Coordination of groups of mobile autonomous agents using nearest neighbor rules”. In: *IEEE Transactions on automatic control* 48.6 (2003), pp. 988–1001.

[4] Angelia Nedic and Asuman Ozdaglar. “Distributed subgradient methods for multi-agent optimization”. In: *IEEE Transactions on Automatic Control* 54.1 (2009), pp. 48–61.

[5] S Sundhar Ram, Angelia Nedić, and Venugopal V Veeravalli. “Distributed stochastic subgradient projection algorithms for convex optimization”. In: *Journal of optimization theory and applications* 147.3 (2010), pp. 516–545.

[6] Guannan Qu and Na Li. “Harnessing smoothness to accelerate distributed optimization”. In: *IEEE Transactions on Control of Network Systems* 5.3 (2017), pp. 1245–1260.

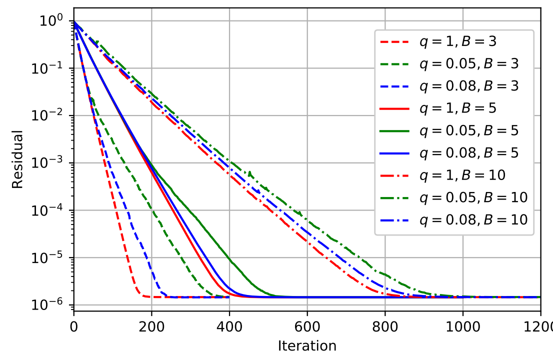
[7] Angelia Nedic, Alex Olshevsky, and Wei Shi. “Achieving geometric convergence for distributed optimization over time-varying graphs”. In: *SIAM Journal on Optimization* 27.4 (2017), pp. 2597–2633.

[8] Rie Johnson and Tong Zhang. “Accelerating stochastic gradient descent using predictive variance reduction”. In: *Advances in neural information processing systems*. 2013, pp. 315–323.

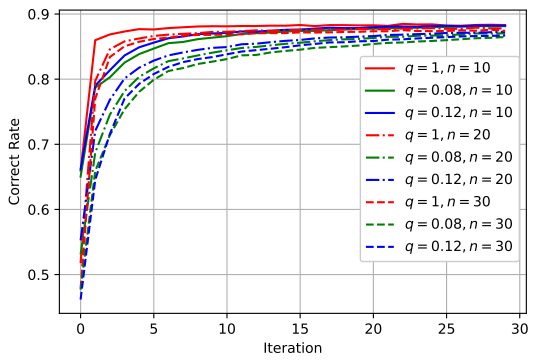
[9] Mark Schmidt, Nicolas Le Roux, and Francis Bach. “Minimizing finite sums with the stochastic average gradient”. In: *Mathematical Programming* 162.1-2 (2017), pp. 83–112.

[10] John Nikolas Tsitsiklis. *Problems in decentralized decision making and computation.* Tech. rep. Massachusetts Inst of Tech Cambridge Lab for Information and Decision Systems, 1984.

[11] Wei Ren, Randal W Beard, and Ella M Atkins. “Information consensus in multivehicle cooperative control”. In: *IEEE Control systems magazine* 27.2 (2007), pp. 71–82.



(a) Linear regression for varied network connectivity.



(b) Correct rate on logistic regression with varied network sizes.

Fig. 3: Varied network connectivity and size.

[12] Kai Cai and Hideaki Ishii. "Average consensus on general strongly connected digraphs". In: *Automatica* 48.11 (2012), pp. 2750–2761.

[13] Kai Cai and Hideaki Ishii. "Average consensus on arbitrary strongly connected digraphs with time-varying topologies". In: *IEEE Transactions on Automatic Control* 59.4 (2014), pp. 1066–1071.

[14] Anastasia Koloskova, Sebastian Stich, and Martin Jaggi. "Decentralized Stochastic Optimization and Gossip Algorithms with Compressed Communication". In: *International Conference on Machine Learning*. 2019, pp. 3478–3487.

[15] Björn Johansson, Maben Rabi, and Mikael Johansson. "A randomized incremental subgradient method for distributed optimization in networked systems". In: *SIAM Journal on Optimization* 20.3 (2010), pp. 1157–1170.

[16] Ermin Wei and Asuman Ozdaglar. "Distributed alternating direction method of multipliers". In: *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*. IEEE. 2012, pp. 5445–5450.

[17] John C Duchi, Alekh Agarwal, and Martin J Wainwright. "Dual averaging for distributed optimization: Convergence analysis and network scaling". In: *IEEE Transactions on Automatic control* 57.3 (2011), pp. 592–606.

[18] Angelia Nedić, Soomin Lee, and Maxim Raginsky. "Decentralized online optimization with global objectives and local communication". In: *2015 American Control Conference (ACC)*. IEEE. 2015, pp. 4497–4503.

[19] Lie He, An Bian, and Martin Jaggi. "Cola: Decentralized linear learning". In: *Advances in Neural Information Processing Systems*. 2018, pp. 4536–4546.

[20] Sebastian U Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. "Sparsified SGD with memory". In: *Advances in Neural Information Processing Systems*. 2018, pp. 4447–4458.

[21] David Kempe, Alin Dobra, and Johannes Gehrke. "Gossip-based computation of aggregate information". In: *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.* IEEE. 2003, pp. 482–491.

[22] Angelia Nedić and Alex Olshevsky. "Distributed optimization over time-varying directed graphs". In: *IEEE Transactions on Automatic Control* 60.3 (2014), pp. 601–615.

[23] Chenguang Xi, Qiong Wu, and Usman A Khan. "On the distributed optimization over directed networks". In: *Neurocomputing* 267 (2017), pp. 508–515.

[24] Angelia Nedić and Alex Olshevsky. "Stochastic gradient-push for strongly convex functions on time-varying directed graphs". In: *IEEE Transactions on Automatic Control* 61.12 (2016), pp. 3936–3947.

[25] Fakhteh Saadatniaki, Ran Xin, and Usman A Khan. "Optimization over time-varying directed graphs with row and column-stochastic matrices". In: *arXiv preprint arXiv:1810.07393* (2018).

[26] Ran Xin, Usman A Khan, and Soummya Kar. "Variance-reduced decentralized stochastic optimization with accelerated convergence". In: *arXiv preprint arXiv:1912.04230* (2019).

[27] Boyue Li et al. "Communication-efficient distributed optimization in networks with gradient tracking and variance reduction". In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2020, pp. 1662–1672.

[28] Bahman Gharesifard and Jorge Cortés. "When does a digraph admit a doubly stochastic adjacency matrix?" In: *Proceedings of the 2010 American Control Conference*. IEEE. 2010, pp. 2440–2445.

[29] Muhammad I Qureshi et al. "Push-SAGA: A decentralized stochastic algorithm with variance reduction over directed graphs". In: *IEEE Control Systems Letters* (2021).

[30] Hanlin Tang et al. "Communication compression for decentralized training". In: *Advances in Neural Information Processing Systems*. 2018, pp. 7652–7662.

[31] Wei Wen et al. "Terngrad: Ternary gradients to reduce communication in distributed deep learning". In: *Advances in neural information processing systems*. 2017, pp. 1509–1519.

[32] Hantian Zhang et al. "Zipml: Training linear models with end-to-end low precision, and a little bit of deep learning". In: *Proceedings of the 34th International*

Conference on Machine Learning-Volume 70. JMLR. org. 2017, pp. 4035–4043.

[33] Rudrajit Das et al. "Improved Convergence Rates for Non-Convex Federated Learning with Compression". In: *arXiv preprint arXiv:2012.04061* (2020).

[34] Amirhossein Reisizadeh et al. "Robust and communication-efficient collaborative learning". In: *Advances in Neural Information Processing Systems*. 2019, pp. 8386–8397.

[35] Zebang Shen et al. "Towards More Efficient Stochastic Decentralized Learning: Faster Convergence and Sparse Communication". In: *International Conference on Machine Learning*. 2018, pp. 4624–4633.

[36] Anastasia Koloskova et al. "Decentralized deep learning with arbitrary communication compression". In: *arXiv preprint arXiv:1907.09356* (2019).

[37] Abolfazl Hashemi et al. "On the Benefits of Multiple Gossip Steps in Communication-Constrained Decentralized Optimization". In: *arXiv preprint arXiv:2012.04061* (2020).

[38] Hossein Taheri et al. "Quantized Decentralized Stochastic Learning over Directed Graphs". In: *International Conference on Machine Learning (ICML)*. 2020.

[39] Lin Xiao and Stephen Boyd. "Fast linear iterations for distributed averaging". In: *Systems & Control Letters* 53.1 (2004), pp. 65–78.

[40] Paul Erdös, Alfréd Rényi, et al. "On random graphs". In: *Publicationes mathematicae* 6.26 (1959), pp. 290–297.

[41] *The Stack Overflow Data*. URL: `https://www.kaggle.com/stackoverflow/stackoverflow`.

Yiyue Chen recieved the B.S. degree from Wuhan University, China, in 2018, and the M.S.E. degree from the University of Texas at Austin in 2020. She is now pursuing her PhD degree in the University of Texas at Austin. Her research interests include machine learning and optimization.

Abolfazl Hashemi received the B.S. degree from Sharif University of Technology, Iran, in 2014, and the M.S.E. and PhD degrees from the University of Texas at Austin in 2016 and 2020, all in Electrical Engineering. He is now a Postdoctoral scholar at the Oden Institute for Computational Engineering and Sciences at the University of Texas at Austin. Abolfazl was the recipient of Iranian national elite foundation fellowship and a best student paper award finalist at 2018 American Control Conference. His research interests include machine learning, signal processing, and control.

Haris Vikalo received the B.S. degree from the University of Zagreb, Croatia, in 1995, the M.S. degree from Lehigh University in 1997, and the Ph.D. degree from Stanford University in 2003, all in electrical engineering. He held a short-term appointment at Bell Laboratories, Murray Hill, NJ, in the summer of 1999. From January 2003 to July 2003 he was a Postdoctoral Researcher, and from July 2003 to August 2007 he was an Associate Scientist at the California Institute of Technology. Since September 2007, he has been with the Department of Electrical and Computer Engineering, the University of Texas at Austin, where he is currently an Associate Professor. He is a recipient of the 2009 National Science Foundation Career Award. His research interests include signal processing, machine learning, communications and bioinformatics.