Variance-Reduced Stochastic Quasi-Newton Methods for Decentralized Learning

Jiaojiao Zhang, Huikang Liu, Anthony Man-Cho So, and Qing Ling

***Abstract*—In this work, we investigate stochastic quasi-Newton methods for minimizing a finite sum of cost functions over a decentralized network. We first develop a general algorithmic framework, in which each node constructs a local, inexact quasi-Newton direction that asymptotically approaches the global, exact one at each time step. To do so, a local gradient approximation is constructed using dynamic average consensus to track the average of variance-reduced local stochastic gradients over the entire network, followed by a proper local Hessian inverse approximation. We show that under standard convexity and smoothness assumptions on the cost functions, the methods obtained from our framework converge linearly to the optimal solution if the local Hessian inverse approximations used have uniformly bounded positive eigenvalues. To construct the Hessian inverse approximations with the said boundedness property, we design two fully decentralized stochastic quasi-Newton methods—namely, the damped regularized limited-memory DFP (Davidon-Fletcher-Powell) and the damped limited-memory BFGS (Broyden-Fletcher-Goldfarb-Shanno)—which use a fixed moving window of past local gradient approximations and local decision variables to adaptively construct Hessian inverse approximations. A noteworthy feature of these methods is that they do not require extra sampling or communication. Numerical results show that the proposed decentralized stochastic quasi-Newton methods are much faster than the existing decentralized stochastic first-order algorithms.**

***Index Terms*—Decentralized optimization, stochastic quasi-Newton methods, variance reduction, damped limited-memory DFP, damped limited-memory BFGS.**

I. Introduction

There has been steadily growing interest in machine learning over networks in various areas, such as large-scale learning [1], [2], privacy-preserving learning [3], [4], decentralized system control [5], [6], etc. These applications often can be formulated as decentralized learning problems. This paper focuses on decentralized learning over an undirected, connected network, where n nodes look for a minimizer of the average cost

$$x^* = \arg\min_{x\in\mathbb{R}^d} \; F(x) := \frac{1}{n}\sum_{i=1}^{n} f_i(x). \tag{1}$$

Here, $x \in \mathbb{R}^d$ is the decision variable and $f_i : \mathbb{R}^d \to \mathbb{R}$ is the local cost function of node i that is expressible as the average of m_i sample costs, i.e.,

$$f_i(x) := \frac{1}{m_i}\sum_{l=1}^{m_i} f_{i,l}(x) \tag{2}$$

with $f_{i,l} : \mathbb{R}^d \to \mathbb{R}$ being the l-th sample cost on node i, assumed to be differentiable. To obtain an optimal solution x^* to (1), the nodes are allowed to communicate with their neighbors and perform local computation. However, at each time step, node i may not have access to the local cost function f_i or the local full gradient ∇f_i since they may involve a large number of samples. Instead, at each time step, each node can access one or a mini-batch of sample costs and compute the local stochastic gradient.

In recent years, a large number of algorithms have been proposed for solving the learning problem (1). Among them, decentralized stochastic first-order methods are appealing due to their low computational complexity. By contrast, decentralized stochastic second-order methods are rarely studied. The goal of this paper is to devise computationally affordable decentralized stochastic second-order methods to accelerate the learning process.

Our methods are motivated by those used in the design of quasi-Newton methods in the centralized setting. Nevertheless, we need some new tools to ensure that the resulting Hessian inverse approximations can be computed in a decentralized manner. To better appreciate the novelty of our constructions, let us review existing techniques for approximating Hessian inverses in the design of quasi-Newton methods.

A. Centralized Stochastic Quasi-Newton Methods

In the centralized setting, the updates of deterministic quasi-Newton methods typically take the form

$$x^{k+1} = x^k - \alpha H^k \nabla F(x^k),$$

where H^k is an approximation of $(\nabla^2 F(x^k))^{-1}$ and $\alpha > 0$ is the step size. Two well-known quasi-Newton methods, DFP (Davidon-Fletcher-Powell) and BFGS (Broyden-Fletcher-Goldfarb-Shanno), update H^k via

$$\text{(DFP)}\quad H^{k+1} = H^k + \frac{s^k(s^k)^T}{(s^k)^T y^k} - \frac{H^k y^k (y^k)^T H^k}{(H^k y^k)^T y^k} \tag{3}$$

and

$$\begin{aligned}\text{(BFGS)}\quad H^{k+1} =& H^k - \frac{H^k y^k (s^k)^T + s^k (y^k)^T H^k}{(s^k)^T y^k} \\ &+ \frac{s^k (s^k)^T}{(s^k)^T y^k}\left(1 + \frac{(y^k)^T H^k y^k}{(s^k)^T y^k}\right),\end{aligned} \tag{4}$$

Jiaojiao Zhang and Anthony Man-Cho So are with the Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong. Huikang Liu is with the School of Information Management and Engineering, Shanghai University of Finance and Economics. Huikang Liu is supported by NSF China Grants 72192832. Qing Ling is with the School of Computer Science and Engineering and Guangdong Provincial Key Laboratory of Computational Science, Sun Yat-Sen University, Guangdong 510006, China, and also with Pazhou Lab, Guangdong 510300, China.

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors. The material includes additional mathematical derivations.

respectively. Here,

$$s^k = x^{k+1} - x^k, \quad y^k = \nabla F(x^{k+1}) - \nabla F(x^k).$$

If the cost F is strongly convex, then the curvature condition $(s^k)^T y^k > 0$ holds and thus the Hessian inverse approximations in (3) and (4) are positive definite for all $k \geq 1$, provided that the initialization H^0 is positive definite [7].

When the number of samples is large, computing the full gradient ∇F can be prohibitive. This motivates the development of stochastic quasi-Newton methods. A common approach is to combine stochastic gradient descent with carefully constructed curvature information [8]–[11]. The work [8] investigates how to construct a diagonal or low-rank matrix according to the so-called secant condition. The work [9] incorporates sub-sampled Hessian information in a Newton conjugate gradient method and a limited-memory quasi-Newton method for statistical learning. The work [10] employs the classic BFGS update formula in its limited memory form and proposes to collect curvature information at spaced intervals to get a stable estimate of the curvature of the objective function. The work [11] proposes an online limited-memory BFGS that uses stochastic gradients instead of full gradients. The convergence analysis of the method is given in [12]. The work [13] proposes a regularized stochastic BFGS method called RES, in which stochastic gradients are used both as descent directions and constituents of Hessian estimates. The regularization used in RES ensures that the eigenvalues of the Hessian approximations are uniformly bounded. The works [14] and [15] use variance reduction to eliminate the stochastic gradient noise, so that the resulting stochastic quasi-Newton methods are provably convergent at linear rates under standard convexity and smoothness assumptions on the cost.

The above-mentioned centralized stochastic quasi-Newton methods [8]–[15] cannot be directly applied to decentralized optimization. The work [8] only solves support vector machine problems. To stabilize the Hessian inverse approximation when stochastic noise presents, the works [9], [11]–[14] use $y^k = \nabla F_{S^k}(x^{k+1}) - \nabla F_{S^k}(x^k)$, which is the difference of stochastic gradients evaluated at the iterates x^{k+1} and x^k with the same sample set S^k. This guarantees that $(s^k)^T y^k > 0$ if each sample cost is strongly convex but doubles the evaluation of stochastic gradients. The works [10], [15] consider $y^k = \nabla^2 F_{S^k}(x^k) s^k$, where $\nabla^2 F_{S^k}(x^k)$ is the sub-sampled Hessian computed from the sample set S^k. This guarantees that $(s^k)^T y^k > 0$ if the sub-sampled Hessian is positive definite but incurs evaluation and sampling of true Hessians. By contrast, we only use local gradient approximations and local decision variables to construct Hessian inverse approximations without extra sampling or communication. The analyses in [8]–[15] use the properties of stochastic gradients or sub-sampled Hessians and do not apply to our methods.

B. Decentralized Deterministic and Stochastic Algorithms

When the numbers $m_1, \ldots, m_n$ of local samples are sufficiently small so that each node i can afford to compute the local full gradient ∇f_i or even the local full Hessian $\nabla^2 f_i$, there are many decentralized deterministic first-order and second-order algorithms for solving (1).

Distributed gradient descent (DGD) is a popular first-order method that combines local gradient descent with average consensus. However, with a fixed step size, it is unable to achieve exact convergence [16], [17]. The convergence error can be eliminated with the help of local historical information, as is done in, e.g., exact first-order algorithm (EXTRA) [18], primal-dual methods [19], [20], exact diffusion [21], and gradient tracking [22]–[24].

Although first-order algorithms are widely used in decentralized learning due to their low computational complexity, second-order methods are able to achieve faster convergence. Several works penalize the consensus constraints (namely, all the local decision variables must be eventually consensual) in the cost function and obtain approximate Newton directions that are computable in a decentralized manner [25]–[27]. A decentralized quasi-Newton method for problems in which the gradient components have a local structure is proposed in [28] and can be used to solve a penalized approximation of problem (1). However, these algorithms only converge to a neighborhood of an optimal solution with fixed step sizes. To achieve exact convergence, several second-order primal-dual methods have been developed [29]–[32]. A decentralized approximate Newton-type algorithm is proposed in [33], which adopts the gradient tracking technique so that the local gradients can track the global ones. A cubically-regularized Newton method with gradient tracking is explored in [34], which runs inexact, preconditioned Newton steps on each node. The work [35] proposes a decentralized adaptive Newton method, where each node runs a finite-time consensus inner loop at each time step.

When the numbers $m_1, \ldots, m_n$ of local samples are large, decentralized deterministic algorithms become prohibitive due to the time-consuming computation of local full gradients and Hessians. Hence, decentralized stochastic algorithms, where each node accesses only one or a mini-batch of sample costs at each time step and computes the local stochastic gradient, are favorable [2], [36], [37]. To remedy the gradient noise brought by the local stochastic gradients, variance reduction techniques such as stochastic variance-reduced gradient (SVRG) can be applied [38]–[45]. However, to the best of our knowledge, computationally affordable decentralized stochastic second-order methods have not been investigated.

C. Contributions

In this work, we first propose a general framework that incorporates decentralized stochastic quasi-Newton approximations with variance reduction to achieve fast convergence and then design two quasi-Newton methods to construct Hessian inverse approximations that fit into the framework. To be specific, at each time step, each node first uses a variance reduction technique to obtain a local corrected stochastic gradient and then uses dynamic average consensus [46] to approximate the global gradient. Further, the gradient approximations are used to construct suitable Hessian inverse approximations. We prove that if the constructed Hessian inverse approximations have uniformly bounded positive eigenvalues, then the methods obtained from our proposed general framework converge linearly to an exact optimal solution of (1).

TABLE I: Stochastic gradient computational complexity of decentralized methods to reach a Δ-optimal solution of (1).

Algorithm[1]	Step size	Stochastic gradient computational complexity	Batch size
DSA [38]	$\alpha=\mathcal{O}\left(\frac{\lambda_{\min}(\tilde{W})}{L\kappa_F}\right)^2$	$\mathcal{O}\left(\max\left\{m\kappa_F,\frac{\kappa_F^4}{1-\sigma},\frac{1}{(1-\sigma)^2}\right\}\log\frac{1}{\Delta}\right)$	$b=1$
GT-SVRG [39]	$\alpha=\mathcal{O}\left(\frac{(1-\sigma^2)^2}{L\kappa_F}\right)$	$\mathcal{O}\left(\left(m+\frac{\kappa_F^2\log\kappa_F}{(1-\sigma^2)^2}\right)\log\frac{1}{\Delta}\right)$	$b=1$
GT-SAGA [39]	$\alpha=\min\left\{\mathcal{O}\left(\frac{1}{\mu m}\right),\mathcal{O}\left(\frac{(1-\sigma^2)^2}{L\kappa_F}\right)\right\}$	$\mathcal{O}\left(\max\left\{m,\frac{\kappa_F^2}{(1-\sigma^2)^2}\right\}\log\frac{1}{\Delta}\right)$	$b=1$
VR-DIGing [40]	$\alpha=\mathcal{O}\left(\frac{1}{\max\left\{L,\frac{\mu}{(1-\sigma)^2}\right\}}\right)$	$\mathcal{O}\left((m+\kappa_F)\log\frac{1}{\Delta}\right)$	$b=\frac{\max\{L,m\mu\}}{\max\left\{L,\frac{\mu}{(1-\sigma)^2}\right\}}$
Acc-VR-DIGing [40]	$\alpha=\mathcal{O}(\frac{1}{L})$	$\mathcal{O}\left((m+\sqrt{m\kappa_F})\log\frac{1}{\Delta}\right)$	$b=\sqrt{\frac{m(1-\sigma)^2\max\{L,m\mu\}}{L}}$
DVR [42]	$\alpha=\mathcal{O}\left(2\lambda_{\min}^{+}\left(A_{\text{comm}}^T D_M^{-1}A_{\text{comm}}\right)\right)^3$	$\mathcal{O}\left((m+\kappa_F)\log\frac{1}{\Delta}\right)$	$b=1$
DFP and BFGS[4]	$\alpha=\mathcal{O}\left(\frac{(1-\sigma^2)^2}{LM_2\kappa_F\kappa_H}\right)$	$\mathcal{O}\left(\left(m+\frac{b\cdot\kappa_F^2\kappa_H^2\log\frac{\kappa_F\kappa_H}{1-\sigma^2}}{(1-\sigma^2)^2}\right)\log\frac{1}{\Delta}\right)$	$\frac{m-b}{(m-1)b}\leq\frac{1}{160}\min\left\{1,\frac{\zeta(1-\sigma^2)^2}{\gamma^2}\right\}$

Note that the use of gradient approximations to construct Hessian inverse approximations is quite adventurous, since the gradient approximations are not necessarily reliable due to stochastic gradient noise and disagreement among the nodes. In particular, if we naïvely adopt centralized quasi-Newton methods, then we may end up with almost-singular Hessian inverse approximations or even non-positive semidefinite ones. To address this issue, we propose two methods—the damped regularized limited-memory DFP (Davidon-Fletcher-Powell) and the damped limited-memory BFGS (Broyden-Fletcher-Goldfarb-Shanno)—which are able to adaptively construct positive definite Hessian inverse approximations. In particular, the proposed methods do not require extra sampling or communication. Further, we prove that the generated Hessian inverse approximations have uniformly bounded positive eigenvalues. Thus, they can be used in the general framework. For the ease of comparison, we summarize the convergence rates of two proposed methods and the existing decentralized stochastic methods in Table I.

Throughout this paper, the costs are assumed to be differentiable but not necessarily twice differentiable. The proposed quasi-Newton methods only use stochastic gradients to estimate Hessians, while the analysis does not make use of true Hessians. If the costs are not twice differentiable so that the constructed Hessian inverse approximations do not contain useful curvature information, our methods reduce to first-order methods but the analysis still holds.

Analyzing the general framework is challenging due to the general Hessian inverse approximations, for which the techniques developed for analyzing existing decentralized variance-reduced first-order algorithms [38]–[45] are insufficient. We design a novel convergence metric related to the function value and establish a proper recursion for it. In addition, due to the general Hessian inverse approximations we encounter two additional terms in the bound of the consensus error, one concerning the variance and the other concerning the optimality gap. Their effects persist throughout the analysis, and we successfully handle the complications caused by them. Finally, via tuning the sample size, we derive a tighter bound of the variance that enables our convergence analysis.

Notation. We use $\|\cdot\|$ to denote the Euclidean norm of a vector; $\text{tr}(\cdot)$, $\|\cdot\|_F$, and $\|\cdot\|_2$ to denote the trace, Frobenius norm, and spectral norm of a matrix, respectively. We use I_d to denote the $d\times d$ identity matrix, 1_n to denote the n-dimensional column vector of all ones, and $\otimes$ to denote the Kronecker product. For two matrices A,B of the same dimensions, we use $A\leq B$ to indicate that each entry of $B-A$ is non-negative; $A\succeq 0$ and $A\succ 0$ to indicate that A is positive semidefinite and positive definite, respectively, with A being symmetric; $A\succeq B$ and $A\succ B$ to mean $A-B\succeq 0$ and $A-B\succ 0$, respectively, with A,B being symmetric. We use $\lambda_{\max}(\cdot)$ and $\lambda_{\min}(\cdot)$ to denote the largest and smallest eigenvalue of a matrix, respectively; $\lambda_i(\cdot)$ to denote the i-th largest eigenvalue of a matrix; $\rho(\cdot)$ to denote the spectral radius of a matrix. For a positive vector $z=[z_1,\ldots,z_d]^T\in\mathbb{R}^d$, an arbitrary vector $a=[a_1,\ldots,a_d]^T\in\mathbb{R}^d$, and an arbitrary matrix $A\in\mathbb{R}^{m\times n}$, we use $\|a\|_\infty^z=\max_i|a_i|/z_i$ to denote the weighted infinity norm of a and $\|A\|_\infty^z$ to denote the weighted infinity norm of A induced by the vector norm $\|\cdot\|_\infty^z$. We use $A^{\frac{1}{2}}$ to denote the square root of a positive semidefinite matrix A, i.e., $A=A^{\frac{1}{2}}A^{\frac{1}{2}}$.

We define $\mathbf{W}=W\otimes I_d\in\mathbb{R}^{nd\times nd}$ and $\mathbf{W}_\infty=\frac{1_n1_n^T}{n}\otimes I_d\in\mathbb{R}^{nd\times nd}$. For $x_1,\ldots,x_n\in\mathbb{R}^d$, we define the aggregated variable $\mathbf{x}=[x_1;\ldots;x_n]\in\mathbb{R}^{nd}$. The aggregated variables $\mathbf{d},\mathbf{g},\mathbf{v}$, and τ are defined similarly. We define the average variable over all the nodes at time k as $\overline{\mathbf{x}}^k=\frac{1}{n}\sum_{i=1}^n x_i^k=\frac{1}{n}(1_n^T\otimes I_d)\mathbf{x}^k\in\mathbb{R}^d$. The average variables $\overline{\mathbf{d}}^k,\overline{\mathbf{g}}^k,\overline{\mathbf{v}}^k$, and $\overline{\tau}^k$ are defined similarly. We define the aggregated gradient at time k as $\nabla f(\mathbf{x}^k)=[\nabla f_1(x_1^k);\ldots;\nabla f_n(x_n^k)]\in\mathbb{R}^{nd}$, the average of all the local gradients at the local variables as $\overline{\nabla f}(\mathbf{x}^k)=\frac{1}{n}\sum_{i=1}^n\nabla f_i(x_i^k)\in\mathbb{R}^d$, and the average of all the local gradients at the common average $\overline{\mathbf{x}}^k$ as $\nabla F\left(\overline{\mathbf{x}}^k\right)=$

[1]For all algorithms, we set the number of local samples $m_i=m$ and the mini-batch sizes $b_i=b$ for all nodes i. Here, σ is the second largest singular value of the mixing matrix W defined in Assumption 3; L and μ are the smoothness and strong convexity constants, respectively; $\kappa_F=\frac{L}{\mu}$ is the condition number of the cost function F; γ and ζ are defined in Theorem 1.

[2]Here, $\tilde{W}=\frac{I+W}{2}$.

[3]Here, $\lambda_{\min}^{+}(\cdot)$ is the smallest positive eigenvalue of a matrix and A_{comm} and D_M are matrices defined by the augmented graph in [42].

[4]Here, $\kappa_H=\frac{M_2}{M_1}$ is the condition number of H defined in Theorem 1. DFP needs extra $\mathcal{O}(Md^2)$ computation and $\mathcal{O}(d^2+Md)$ storage per iteration, while BFGS needs extra $\mathcal{O}(Md)$ computation and $\mathcal{O}(Md)$ storage per iteration. This will be shown in Section III.

$\frac{1}{n}\sum_{i=1}^n \nabla f_i(\overline{\mathbf{x}}^k) \in \mathbb{R}^d$. Define the block diagonal matrix $\mathbf{H}^k = \mathrm{diag}\{H_i^k\} \in \mathbb{R}^{nd\times nd}$ whose i-th block is $H_i^k \in \mathbb{R}^{d\times d}$ and the average matrix $\overline{\mathbf{H}}^k = \frac{1}{n}\sum_{i=1}^n H_i^k \in \mathbb{R}^{d\times d}$. Given a random variable v, denote $\mathbb{E}[v]$ as expectation and $\mathbb{E}[v|\mathcal{F}]$ as conditional expectation with respect to (w.r.t.) the event $\mathcal{F}$.

II. Formulation and General Framework

We begin by introducing the problem formulation of decentralized learning and stating the basic assumptions. Then, we propose a general framework for designing variance-reduced decentralized stochastic quasi-Newton methods.

A. Problem Formulation

We consider an undirected, connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with node set $\mathcal{V} = \{1, \ldots, n\}$ and edge set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. The nodes $i \in \mathcal{V}$ and $j \in \mathcal{V}$ are allowed to send information to each other if they are connected by an edge $(i, j) \in \mathcal{E}$. Each node i has a local cost function f_i in the form of (2) and makes decisions based on stochastic gradients of f_i and information obtained from its neighbors. Let $\mathcal{N}_i$ be the set of neighbors of node i including itself and $x_i \in \mathbb{R}^d$ be the local copy of the decision variable x at node i. Since the network is bidirectionally connected, problem (1) is equivalent to

$$\begin{aligned} \mathbf{x}^* = \underset{\mathbf{x}=[x_1;\ldots;x_n]}{\arg\min}\ & f(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^n f_i(x_i) \\ \text{s.t.}\ \ & x_i = x_j,\ \forall j \in \mathcal{N}_i,\ \forall i, \end{aligned} \tag{5}$$

where $\mathbf{x}^* := [x^*; \ldots; x^*] \in \mathbb{R}^{nd}$. We note that (1) and (5) are equivalent in the sense that at optimality, the local variables $x_1^*, \ldots, x_n^*$ of (5) are all equal to the optimal solution x^* of (1), i.e., $x_1^* = \cdots = x_n^* = x^*$.

We make the following assumptions on the cost functions.

Assumption 1 (Convexity and L-smoothness)**.** *Each local sample cost $f_{i,l}$ is convex and has Lipschitz continuous gradient, i.e.,*

$$\begin{aligned} f_{i,l}(y) \geq & f_{i,l}(x) + \nabla f_{i,l}(x)^T(y-x), \\ f_{i,l}(y) \leq & f_{i,l}(x) + \nabla f_{i,l}(x)^T(y-x) + \frac{L}{2}\|y-x\|^2 \end{aligned}$$

for all $x, y \in \mathbb{R}^d$, where $L > 0$ is the Lipschitz constant.

Assumption 1 implies that the local cost functions f_i defined in (2) and the global cost function F defined in (1) are also convex and L-smooth. Under Assumption 1, we have the following fact, see [47, Theorem 2.1.5].

Fact 1. *Under Assumption 1, for all $x, y \in \mathbb{R}^d$, we have*

$$\begin{aligned} & \frac{1}{2L}\|\nabla f_{i,l}(x) - \nabla f_{i,l}(y)\|^2 \\ \leq & f_{i,l}(x) - f_{i,l}(y) - \nabla f_{i,l}(y)^T(x-y). \end{aligned}$$

Note that similar results hold for the local cost functions f_i and the global cost function F.

Assumption 2 (μ-strong convexity)**.** *The global cost function F is strongly convex, i.e.,*

$$F(y) \geq F(x) + \nabla F(x)^T(y-x) + \frac{\mu}{2}\|y-x\|^2$$

for all $x, y \in \mathbb{R}^d$, where $\mu > 0$ is the strong convexity constant.

To reach the consensual optimal solution, the nodes need to mix their local decision variables with those of their neighbors according to predefined weights. Let $w_{ij} \geq 0$ represent the weight that node i assigns to j and consider the mixing matrix $W = [w_{ij}] \in \mathbb{R}^{n\times n}$. We make the following assumption.

Assumption 3 (Mixing matrix)**.** *The mixing matrix W is non-negative, symmetric, and doubly stochastic (i.e., $w_{ij} \geq 0$ for all $i, j \in \{1, \ldots, n\}$, $W = W^T$, and $W1_n = 1_n$) with $w_{ij} = 0$ if and only if $j \notin \mathcal{N}_i$.*

One useful consequence of Assumption 3 is that the null space of $I_n - W$ is span(1_n). Mixing matrices satisfying Assumption 3 are commonly used in decentralized learning over an undirected, connected network, see, e.g., [48] for details. According to the Perron-Frobenius theorem [49], Assumption 3 implies that the eigenvalues of W lie in $(-1, 1]$ and the multiplicity of the eigenvalue 1 is one. It also implies that the second largest singular value σ of W is less than 1, i.e., $\sigma = \|W - \frac{1}{n}1_n 1_n^T\|_2 < 1$.

B. General Framework for Designing Variance-Reduced Decentralized Stochastic Quasi-Newton Methods

We propose a general framework for designing variance-reduced decentralized stochastic quasi-Newton methods to solve (5). At the k-th time step, node i updates its local decision variable x_i^{k+1} according to the decentralized stochastic quasi-Newton step

$$x_i^{k+1} = \sum_{j=1}^n w_{ij}x_j^k - \alpha d_i^k, \tag{6}$$

where $\alpha > 0$ is a constant step size. Consider the most ideal scenario, in which the network is fully connected and each node has sufficient computational power. If each local cost function f_i is twice differentiable, then the direction d_i^{k+1} can be chosen as the global negative Newton direction $\left(\nabla^2 F(\overline{\mathbf{x}}^{k+1})\right)^{-1}\nabla F(\overline{\mathbf{x}}^{k+1}) = \left(\frac{1}{n}\sum_{i=1}^n \nabla^2 f_i(\overline{\mathbf{x}}^{k+1})\right)^{-1}\left(\frac{1}{n}\sum_{i=1}^n \nabla f_i(\overline{\mathbf{x}}^{k+1})\right)$, which is expressible as the multiplication of the global Hessian inverse and the global gradient at the average variable $\overline{\mathbf{x}}^{k+1} = \frac{1}{n}\sum_{i=1}^n x_i^{k+1}$. However, computing the global negative Newton direction is expensive in the general decentralized stochastic learning setting for two reasons. First, computing the global Hessian inverse and the global gradient is impossible, because each node i only has access to the information from the neighborhood $\mathcal{N}_i$ instead of the entire network. Second, even computing the local Hessian inverse and the local gradient is unaffordable, because they involve all the sample costs on each node.

In the general framework, we update the direction d_i^{k+1} by

$$d_i^{k+1} = H_i^{k+1}g_i^{k+1}, \tag{7}$$

where H_i^{k+1} and g_i^{k+1} are carefully constructed Hessian inverse approximation and gradient approximation, respectively. Compared with the global negative Newton direction, the Hessian inverse approximation H_i^{k+1} is used to estimate the global Hessian inverse $\left(\frac{1}{n}\sum_{i=1}^n \nabla^2 f_i(\overline{\mathbf{x}}^{k+1})\right)^{-1} = \left(\nabla^2 F(\overline{\mathbf{x}}^{k+1})\right)^{-1}$

and the gradient approximation g_i^{k+1} is used to estimate the global gradient $\frac{1}{n}\sum_{i=1}^n \nabla f_i(\overline{\mathbf{x}}^{k+1}) = \nabla F(\overline{\mathbf{x}}^{k+1})$. For concreteness, the reader can simply let H_i^{k+1} be I_d throughout the general framework. Other constructions of H_i^{k+1} and the corresponding quasi-Newton methods are discussed in Section III.

For the gradient approximation g_i^{k+1}, we consider the case where the sample size m_i is large so that it is unaffordable to compute the local full gradient $\nabla f_i(x_i^{k+1})$. We proceed by choosing a subset $S_i^{k+1} \subseteq \{1,\dots,m_i\}$ with cardinality $|S_i^{k+1}| = b_i$ uniformly at random for node i, computing the stochastic gradients $\nabla f_{i,l}(x_i^{k+1})$, and obtaining a corrected stochastic gradient v_i^{k+1} with SVRG via

$$v_i^{k+1} = \sum_{l\in S_i^{k+1}} \frac{\nabla f_{i,l}(x_i^{k+1}) - \nabla f_{i,l}(\tau_i^{k+1})}{b_i} + \nabla f_i(\tau_i^{k+1}), \quad (8)$$

where $\tau_i^{k+1} \in \mathbb{R}^d$ is an auxiliary variable. Given a positive integer $\mathcal{T}$, we set $\tau_i^{k+1} = x_i^{k+1}$ if $\mod(k+1,\mathcal{T}) = 0$ and $\tau_i^{k+1} = \tau_i^k$ otherwise. In particular, node i calculates its local full gradient once every $\mathcal{T}$ time steps and saves it to correct the subsequent $\mathcal{T}$ local stochastic gradients. With SVRG, v_i^{k+1} is a reliable, unbiased estimate of the local full gradient $\nabla f_i(x_i^{k+1})$. However, we expect g_i^{k+1} to estimate the global gradient $\frac{1}{n}\sum_{i=1}^n \nabla f_i(\overline{\mathbf{x}}^{k+1})$. Inspired by the gradient tracking strategy [46], we construct g_i^{k+1} with a dynamic average consensus step, i.e.,

$$g_i^{k+1} = \sum_{j=1}^n w_{ij} g_j^k + v_i^{k+1} - v_i^k \quad (9)$$

with initialization $g_i^0 = v_i^0 = \nabla f_i(x_i^0)$.

Intuitively, the local corrected stochastic gradient v_i^{k+1} will gradually approach the local full gradient $\nabla f_i(x_i^{k+1})$ with the help of SVRG. If the local decision variables x_i^{k+1} are almost consensual, then the gradient approximations g_i^{k+1} will gradually approach the global gradient $\frac{1}{n}\sum_{i=1}^n \nabla f_i(\overline{\mathbf{x}}^{k+1})$ with the help of dynamic average consensus. With the gradient approximations g_i^{k+1} at hand, in Section III, we will show how to obtain a Hessian inverse approximation H_i^{k+1} that estimates the global Hessian inverse $\left(\frac{1}{n}\sum_{i=1}^n \nabla^2 f_i(\overline{\mathbf{x}}^{k+1})\right)^{-1}$. As we will see in Section III, the Hessian inverse approximation H_i^k is constructed locally using g_i^k and x_i^k without extra sampling or communication. Just as the first-order gradient tracking methods, our second-order methods require two rounds of communication of d-dimension vectors at each time step.

The proposed framework for designing variance-reduced decentralized stochastic quasi-Newton methods is described in Algorithm 1. The updates can be written compactly as

$$\begin{aligned} \mathbf{x}^{k+1} &= \mathbf{W}\mathbf{x}^k - \alpha \mathbf{d}^k, \\ \mathbf{g}^{k+1} &= \mathbf{W}\mathbf{g}^k + \mathbf{v}^{k+1} - \mathbf{v}^k, \\ \mathbf{d}^{k+1} &= \mathbf{H}^{k+1}\mathbf{g}^{k+1}. \end{aligned} \quad (10)$$

III. Damped Limited-Memory Stochastic Quasi-Newton Methods

In this section, we propose two fully decentralized stochastic quasi-Newton methods for solving the decentralized learning problem (1). To motivate our approach, recall in the centralized setting, the Hessian inverse approximations used in quasi-Newton methods are typically given by the solutions of certain optimization problems [7]. However, such approximations are costly to compute in the decentralized setting, as they require global information. To obtain an efficient decentralized implementation, we modify the said optimization problems so that each node i can solve the modified problems to get the local Hessian inverse approximations using only its own gradient approximations g_i^{k+1} and decision variables x_i^{k+1}.

Algorithm 1 Variance-reduced decentralized stochastic quasi-Newton method on node i

Require: α; K; $\mathcal{T}$; b_i; x_i^0; H_i^0; $\tau_i^0 = x_i^0$; $g_i^0 = v_i^0 = \nabla f_i(x_i^0)$; $d_i^0 = H_i^0 g_i^0$.

1: **for** $k = 0, 1, 2, \dots, K-1$ **do**
2: Update local decision variable x_i^{k+1} as in (6).
3: Select $S_i^{k+1} \subseteq \{1,\dots,m_i\}$ with batch size b_i.
4: Update auxiliary variable τ_i^{k+1} via

$$\tau_i^{k+1} = \begin{cases} x_i^{k+1}, & \text{if } \mod(k+1,\mathcal{T}) = 0, \\ \tau_i^k, & \text{otherwise.} \end{cases}$$

5: Update corrected stochastic gradient v_i^{k+1} as in (8).
6: Update gradient approximation g_i^{k+1} as in (9).
7: Construct Hessian inverse approximation H_i^{k+1}.
8: Update direction d_i^{k+1} as in (7).
9: **end for**

A. Damped Regularized Limited-Memory DFP

It is known that the DFP update is obtained by minimizing the Gaussian differential entropy subject to certain constraints. Inspired by [13], [50], to avoid $\lambda_{\min}(H^{k+1}) \to 0$, we add a regularization term with parameter $\rho > 0$ to the minimization problem and solve

$$\begin{aligned} H^{k+1} = \arg\min_{Z\in\mathbb{R}^{d\times d}} \ & \mathrm{tr}[(H^k)^{-1}(Z-\rho I_d)] \\ & - \log\det[(H^k)^{-1}(Z-\rho I_d)] \\ \text{s.t. } & Zy^k = s^k,\ Z \succeq 0, \end{aligned} \quad (11)$$

where $s^k = x^{k+1} - x^k$ is the variable variation and $y^k = g^{k+1} - g^k$ is the gradient approximation variation. If we let $\rho = 0$, then (11) reduces to the traditional DFP update.

Define the modified variable variation $\hat{s}^k$ as

$$\hat{s}^k = s^k - \rho y^k.$$

As shown in [13], the closed-form solution to (11) is given by

$$H^{k+1} = H^k + \frac{\hat{s}^k(\hat{s}^k)^T}{(\hat{s}^k)^T y^k} - \frac{H^k y^k (y^k)^T H^k}{(y^k)^T H^k y^k} + \rho I_d. \quad (12)$$

In the centralized stochastic setting where g^k is the stochastic gradient instead of the stochastic gradient approximation, the work [13] shows that if ρ is properly chosen and $H^0 \succ 0$, then $(\hat{s}^k)^T y^k > 0$ and thus $\lambda_{\min}(H^{k+1}) > \rho$ for all k. It also establishes an upper bound on the eigenvalues of H^{k+1} for all k. Unfortunately, these results no longer hold in the

decentralized stochastic setting, since the gradient approximation variation is affected by both the stochastic gradient noise and disagreement among the nodes. To address this issue, we introduce the damped regularized limited-memory DFP update on each node, which we now describe. Since our subsequent argument holds for any node, let us drop the node index i.

Given an initialization $H^0 \succ 0$, we define $\hat{y}^k$ as

$$\hat{y}^k = \theta^k y^k + (1-\theta^k)(H^0+\epsilon I_d)^{-1}\hat{s}^k, \tag{13}$$

where $\epsilon > 0$ is a parameter and θ^k is adaptively computed by

$$\theta^k = \min\left\{\tilde{\theta}^k, \frac{\tilde{L}\|\hat{s}^k\|}{\|y^k\|}\right\} \tag{14}$$

with $\tilde{\theta}^k$ being defined as [50]

$$\tilde{\theta}^k = \begin{cases} \frac{0.75(\hat{s}^k)^T\left(H^0+\epsilon I_d\right)^{-1}\hat{s}^k}{(\hat{s}^k)^T(H^0+\epsilon I_d)^{-1}\hat{s}^k-(\hat{s}^k)^T y^k}, \\ \quad \text{if } (\hat{s}^k)^T y^k \le 0.25(\hat{s}^k)^T\left(H^0+\epsilon I_d\right)^{-1}\hat{s}^k, \\ 1, \text{ otherwise} \end{cases}$$

and $\tilde{L} > 0$ being a parameter. As we will show later, with the added term $\frac{\tilde{L}\|\hat{s}^k\|}{\|y^k\|}$ in (14), $\|\hat{y}^k\|$ can be upper bounded in terms of $\|\hat{s}^k\|$. Then, we replace y^k with $\hat{y}^k$ in (12) to obtain the damped regularized DFP update

$$H^{k+1} = H^k + \frac{\hat{s}^k(\hat{s}^k)^T}{(\hat{s}^k)^T\hat{y}^k} - \frac{H^k\hat{y}^k(\hat{y}^k)^T H^k}{(\hat{y}^k)^T H^k\hat{y}^k} + \rho I_d.$$

We will prove in Lemma 3 that $(\hat{s}^k)^T\hat{y}^k > 0$, from which it follows that $\lambda_{\min}(H^{k+1}) > \rho$ for all k.

Next, we have to guarantee that $\lambda_{\max}(H^{k+1}) < \infty$. This is non-trivial since $\hat{y}$ is noisy and the regularization term ρI_d may accumulate as the algorithm progresses. Inspired by [50], [51], we use the limited-memory technique to tackle this issue. It is worth noting that the limited-memory technique is usually combined with BFGS to reduce the memory and computational costs. Here, we combine it with DFP to make the eigenvalues of H^k bounded, which, to the best of our knowledge, is new.

In our development below, we reinstate the node index i to emphasize that the computation is done on node i. At time step k, we set

$$H_i^{k,(0)} = \min\left\{\max\left\{\frac{(s_i^k)^T s_i^k}{(s_i^k)^T y_i^k} + \rho, \beta\right\}, \mathcal{B}\right\} I_d \tag{15}$$

to be the initial Hessian inverse approximation, where $\beta > 0$ and $\mathcal{B} > 0$ are two parameters. By construction, we have $\beta I_d \preceq H_i^{k,(0)} \preceq \mathcal{B} I_d$. Then, given two sequences $\{\hat{s}_i^p\}$ and $\{\hat{y}_i^p\}$, where $p = k+1-\tilde{M}, \ldots, k$, $\tilde{M} = \min\{k+1, M\}$, and M is the memory size, we compute

$$\begin{aligned} H_i^{k,(t+1)} =& H_i^{k,(t)} + \frac{\hat{s}_i^p(\hat{s}_i^p)^T}{(\hat{s}_i^p)^T\hat{y}_i^p} \\ & - \frac{H_i^{k,(t)}\hat{y}_i^p(\hat{y}_i^p)^T H_i^{k,(t)}}{(\hat{y}_i^p)^T H_i^{k,(t)}\hat{y}_i^p} + \rho I_d \end{aligned} \tag{16}$$

for $p = k+1-\tilde{M}+t$ and $t = 0, \ldots, \tilde{M}-1$. At the end of this inner loop, we set $H_i^{k+1} = H_i^{k,(\tilde{M})}$ to be the Hessian inverse approximation at time $k+1$.

We summarize the Hessian inverse approximation step of the damped regularized limited-memory DFP method on node i at time k in Algorithm 2.

Algorithm 2 Hessian inverse approximation step of damped regularized limited-memory DFP on node i at time k

Require: ρ; β; $\mathcal{B}$; ϵ; $\tilde{L}$; M.
1: Update variable variation $s_i^k = x_i^{k+1} - x_i^k$.
2: Update gradient variation $y_i^k = g_i^{k+1} - g_i^k$.
3: Update modified variable variation $\hat{s}_i^k = s_i^k - \rho y_i^k$.
4: Update modified gradient variation $\hat{y}_i^k$ as in (13).
5: Set $\tilde{M} = \min\{k+1, M\}$ and load $\{\hat{s}_i^p, \hat{y}_i^p\}_{p=k+1-\tilde{M}}^k$.
6: Initialize $H_i^{k,(0)}$ as in (15).
7: **for** $t = 0, \ldots, \tilde{M}-1$ **do**
8: Update $H_i^{k,(t+1)}$ as in (16).
9: **end for**
10: Output $H_i^{k+1} = H_i^{k,(\tilde{M})}$.

B. Damped Limited-Memory BFGS

Now, we introduce another decentralized stochastic quasi-Newton method for solving (1)—the damped limited-memory BFGS method. Recall that in the traditional BFGS method, the Hessian inverse approximation update is given by

$$\begin{aligned} H^{k+1} = \arg\min_Z \ & \|Z - H^k\|_O \\ \text{s.t. } & Zy^k = s^k, Z = Z^T, \end{aligned} \tag{17}$$

where $\|\cdot\|_O$ is the O-weighted norm defined as $\|H\|_O = \|O^{\frac{1}{2}} H O^{\frac{1}{2}}\|_F$ with O being an arbitrary positive semidefinite matrix satisfying $Os^k = y^k$ [7], $s^k = x^{k+1} - x^k$ is the variable variation, and $y^k = g^{k+1} - g^k$ is the gradient approximation variation. In particular, the solution H^{k+1} is the point that is closest, in terms of the O-weighted norm, to H^k among all symmetric matrices that satisfy the secant condition $Zy^k = s^k$. From [7], the closed-form solution of (17) is

$$\begin{aligned} H^{k+1} =& H^k - \frac{H^k y^k (s^k)^T + s^k (y^k)^T H^k}{(s^k)^T y^k} \\ & + \frac{s^k (s^k)^T}{(s^k)^T y^k}\left(1 + \frac{(y^k)^T H^k y^k}{(s^k)^T y^k}\right). \end{aligned} \tag{18}$$

In the decentralized setting, we need to compute a Hessian inverse approximation on each node at each time step. Since our subsequent argument holds for any node, let us omit the node index i for simplicity. As we have mentioned in Section III-A, the gradient approximations g^k are noisy, which makes it non-trivial to preserve the positivity and uniform boundedness of the eigenvalues of H^k. To overcome this difficulty, we combine the damping technique with the traditional BFGS update in (18). To be specific, given an initialization $H^0 \succ 0$, we define $\hat{y}^k$ as

$$\hat{y}^k = \theta y^k + (1-\theta)(H^0+\epsilon I)^{-1} s^k, \tag{19}$$

where $\epsilon > 0$ is a parameter and θ is adaptively computed by

$$\theta^k = \min\left\{\tilde{\theta}^k, \frac{\tilde{L}\|s^k\|}{\|y^k\|}\right\} \tag{20}$$

with $\tilde{\theta}^k$ being defined as [50]

$$\tilde{\theta}^k = \begin{cases} \frac{0.75(s^k)^T\left(H^0+\epsilon I_d\right)^{-1}s^k}{(s^k)^T(H^0+\epsilon I_d)^{-1}s^k-(s^k)^Ty^k}, \\ \quad \text{if } (s^k)^Ty^k \le 0.25(s^k)^T\left(H^0+\epsilon I_d\right)^{-1}s^k, \\ 1, \text{ otherwise} \end{cases}$$

and $\tilde{L} > 0$ being a parameter. Similar to damped regularized limited-memory DFP, with the added term $\frac{\tilde{L}\|s^k\|}{\|y^k\|}$ in (20), $\|\hat{y}^k\|$ can be upper bounded in terms of $\|s^k\|$. Then, we replace y^k in (18) with $\hat{y}^k$ to obtain damped limited-memory BFGS update

$$\begin{aligned} H^{k+1} =& H^k - \frac{H^k\hat{y}^k(s^k)^T + s^k(\hat{y}^k)^TH^k}{(s^k)^T\hat{y}^k} \\ &+ \frac{s^k(s^k)^T}{(s^k)^T\hat{y}^k}\left(1+\frac{(\hat{y}^k)^TH^k\hat{y}^k}{(s^k)^T\hat{y}^k}\right). \end{aligned}$$

As we will prove, the damping technique guarantees that $(s^k)^T\hat{y}^k > 0$, from which we have $\lambda_{\min}(H^{k+1}) > 0$ for all k.

Next, to guarantee that $\lambda_{\max}(H^{k+1}) < \infty$, we use the limited-memory technique. Compared with the damped regularized limited-memory DFP method in Section III-A, the limited-memory technique used here is not only for bounding the eigenvalues of the Hessian inverse approximations but also for reducing storage and computational costs. Again, let us reinstate the node index i to indicate that the computation is done on node i. At time k, we set

$$H_i^{k,(0)} = \min\left\{\max\left\{\frac{(s_i^k)^Ty_i^k}{(y_i^k)^Ty_i^k}, \beta\right\}, \mathcal{B}\right\} I_d \tag{21}$$

to be the initial Hessian inverse approximation. Given two sequences $\{s_i^p\}$ and $\{\hat{y}_i^p\}$, where $p = k+1-\tilde{M}, \ldots, k$, $\tilde{M} = \min\{k+1, M\}$, and M is the memory size, we compute

$$\begin{aligned} H_i^{k,(t+1)} &= H_i^{k,(t)} - \frac{H_i^{k,(t)}\hat{y}_i^p(s_i^p)^T + s_i^p(\hat{y}_i^p)^TH_i^{k,(t)}}{(s_i^p)^T\hat{y}_i^p} \\ &\quad + \frac{s_i^p(s_i^p)^T}{(s_i^p)^T\hat{y}_i^p}\left(1+\frac{(\hat{y}_i^p)^TH_i^{k,(t)}\hat{y}_i^p}{(s_i^p)^T\hat{y}_i^p}\right) \\ &= \left(I_d - \frac{s_i^p(\hat{y}_i^p)^T}{(s_i^p)^T\hat{y}_i^p}\right)H_i^{k,(t)}\left(I_d - \frac{\hat{y}_i^p(s_i^p)^T}{(s_i^p)^T\hat{y}_i^p}\right) + \frac{s_i^p(s_i^p)^T}{(s_i^p)^T\hat{y}_i^p} \end{aligned} \tag{22}$$

for $p = k+1-\tilde{M}+t$ and $t = 0, \ldots, \tilde{M}-1$. The second equality above will be useful for our later analysis. At the end of this inner loop, we set $H_i^{k+1} = H_i^{k,(\tilde{M})}$ to be the Hessian inverse approximation at time $k+1$.

One advantage of the proposed damped limited-memory BFGS method is that the update (22) can be realized by a two-loop recursion, where $H_i^{k,(t)}$ is not generated explicitly; rather, only its multiplications with certain vectors are computed. We summarize the Hessian inverse approximation step and the two-loop recursion step of the damped limited-memory BFGS method on node i at time k in Algorithms 3 and 4, respectively.

Remark 1. *Compared with the damped regularized limited-memory DFP method, the damped limited-memory BFGS method does not use regularization. The reason is that adding the regularization term ρI_d at the end of update* (22) *will make it difficult to realize the two-loop recursion. The memory requirement and computational cost per iteration of the proposed BFGS-type method are $\mathcal{O}(Md)$ and $\mathcal{O}(Md)$, respectively. By contrast, the memory requirement and computational cost per iteration of the proposed DFP-type method are $\mathcal{O}(d^2+Md)$ and $\mathcal{O}(Md^2)$, respectively.*

Algorithm 3 Hessian inverse approximation step of damped limited-memory BFGS on node i at time k

Require: β; $\mathcal{B}$; ϵ; $\tilde{L}$; M.

1: Update variable variation $s_i^k = x_i^{k+1} - x_i^k$.
2: Update gradient variation $y_i^k = g_i^{k+1} - g_i^k$.
3: Update modified gradient variation $\hat{y}_i^k$ as in (19).
4: Initialize $H_i^{k,(0)}$ as in (21).
5: Set $\tilde{M} = \min\{k+1, M\}$ and load $\{s_i^p, \hat{y}_i^p\}_{p=k+1-\tilde{M}}^k$.
6: Perform two-loop limited-memory BFGS in Algorithm 4.
7: Output direction $d_i^{k+1} = H_i^{k+1}g_i^{k+1}$.

Algorithm 4 Two-loop limited-memory BFGS on node i at time k

Set $q_i \leftarrow g_i^{k+1}$.
for $p = k, k-1, \ldots, k+1-\tilde{M}$ **do**
  $\alpha_i^p \leftarrow \frac{(s_i^p)^Tq_i}{(s_i^p)^T\hat{y}_i^p}$.
  $q_i \leftarrow q_i - \alpha_i^p\hat{y}_i^p$.
end for
$r_i \leftarrow H_i^{k,(0)}q_i$.
for $p = k+1-\tilde{M}, k-\tilde{M}, \ldots, k$ **do**
  $\beta_i \leftarrow \frac{(\hat{y}_i^p)^Tr_i}{(s_i^p)^T\hat{y}_i^p}$.
  $r_i \leftarrow r_i + s_i^p(\alpha_i^p - \beta_i)$.
end for
Output $H_i^{k+1}g_i^{k+1} = r_i$.

IV. Convergence Analysis of General Framework

In this section, we establish linear convergence of Algorithm 1 under the following assumption on the Hessian inverse approximations H_i^k.

Assumption 4 (Bounded Hessian inverse approximations)**.** *There exist constants M_1 and M_2 with $0 < M_1 \le M_2 < \infty$ such that*

$$M_1I_d \preceq H_i^k \preceq M_2I_d, \ \forall i \in \{1, \ldots, n\}, \ \forall k \ge 0.$$

Before we proceed, some remarks on Assumption 4 are in order. First, one obvious construction that satisfies the assumption is $H_i^k = I_d$ for all $i \in \{1, \ldots, n\}$ and $k \ge 0$, which corresponds to taking $M_1 = M_2 = 1$. However, there are other constructions of the H_i^k's that not only satisfy Assumption 4 but can also exploit the curvature of the cost function to accelerate the convergence of the method, including but not limited to the proposed quasi-Newton methods in Section III.

Second, when $H_i^k = I_d$ for all $i \in \{1, \ldots, n\}$ and $k \ge 0$, the general framework (10) reduces to the first-order algorithm GT-SVRG in [39]. In this case, we have $\overline{\mathbf{x}}^{k+1} = \overline{\mathbf{x}}^k - \alpha\overline{\mathbf{v}}^k$, where we substitute $\overline{\mathbf{g}}^k = \overline{\mathbf{v}}^k$ due to the dynamic average consensus step; see (31). Such an update is not too far away from that of centralized gradient descent, as $\overline{\mathbf{v}}^k$ is an unbiased estimator of $\overline{\nabla f}(\mathbf{x}^k)$ and $\overline{\nabla f}(\mathbf{x}^k) \to \nabla F(\overline{\mathbf{x}}^k)$ when the local

decision variables reach a consensus asymptotically. Based on this property, one can establish the convergence of $\|\overline{\mathbf{x}}^k - x^*\|$ as in the analysis of classic centralized gradient descent; see [39, Lemma 5]. However, for general H_i^k's, the convergence analysis of Algorithm 1 becomes much more challenging, and the techniques developed in [39] are insufficient. Indeed, in this case we can no longer characterize the convergence of $\|\overline{\mathbf{x}}^k - x^*\|$ directly. Instead, we establish the decrease of $F(\overline{\mathbf{x}}^k) - F(x^*)$; see Lemma 7 and Corollary 2. Moreover, we encounter two additional terms when we bound the consensus error $\|\mathbf{x}^{k+1} - \mathbf{W}_\infty \mathbf{x}^{k+1}\|^2$, one concerning the variance $\|\mathbf{v}^k - \nabla f(\mathbf{x}^k)\|^2$ and the other concerning the optimality gap $F(\overline{\mathbf{x}}^k) - F(x^*)$; see Lemma 5. Their effects persist throughout the analysis because the consensus error plays a fundamental role in bounding the other error terms. Thus, our recursion is more complicated, and we have to derive a tighter bound of the variance to establish convergence; see Lemma 6.

A. Main Theorem

Motivated by the analysis in [23], [39], we use the consensus error $\mathbb{E}[\|\mathbf{x}^k - \mathbf{W}_\infty \mathbf{x}^k\|^2]$, network optimality gap $\mathbb{E}[F(\overline{\mathbf{x}}^k) - F(x^*)]$, and the gradient tracking error $\mathbb{E}[\|\mathbf{g}^k - \mathbf{W}_\infty \mathbf{g}^k\|^2]$ to establish the convergence rate. To begin, let

$$\mathbf{u}^k = \begin{bmatrix} \mathbb{E}[\|\mathbf{x}^k - \mathbf{W}_\infty \mathbf{x}^k\|^2] \\ \frac{2n}{L}\mathbb{E}\left[F(\overline{\mathbf{x}}^k) - F(x^*)\right] \\ \frac{1-\sigma^2}{L^2}\mathbb{E}\left[\|\mathbf{g}^k - \mathbf{W}_\infty \mathbf{g}^k\|^2\right] \end{bmatrix} \in \mathbb{R}^3$$

be the aggregated error vector at time step k. Note that $\mathbf{u}^k$ can be viewed as a measure of the discrepancy between x_i^k and x^* since $\mathbf{u}^k = 0$ implies $x_i^k = \overline{\mathbf{x}}^k = x^*, \forall i$. Next, set

$$B := \max_{i\in\{1,\dots,n\}} \left\{ \frac{m_i - b_i}{(m_i - 1)b_i} \right\} < 1. \tag{23}$$

We call B non-sampling rate, since $B = 0$ means each node i uses all m_i samples to compute the local full gradient ∇f_i.

Our main theorem gives the conditions on the parameters, including the step size α, the non-sampling rate B, and the period $\mathcal{T}$ of SVRG, that can guarantee the linear convergence of Algorithm 1 to the optimal solution of (1).

Theorem 1. *Under Assumptions 1–4, if the parameters satisfy*

$$\begin{aligned} \alpha &\le \frac{(1-\sigma^2)^2 \mu M_1}{200 L^2 M_2^2}, \\ \beta := 16B &\le \frac{1}{10}\min\left\{1, \frac{\zeta(1-\sigma^2)^2}{\gamma^2}\right\}, \\ \mathcal{T} &\ge \frac{2\log(280/(\zeta(1-\sigma^2)^2))}{\zeta\widetilde{\alpha}}, \end{aligned} \tag{24}$$

where

$$\zeta := \left(\frac{\mu}{L}\right)^2 \left(\frac{M_1}{M_2}\right)^2,\ \gamma := 1 - \frac{M_1}{M_2},\ \widetilde{\alpha} := \frac{M_2^2 L^2}{M_1 \mu}\alpha,$$

then for $\mathbf{q} = [1; 10; \frac{200(\zeta+\beta)}{1-\sigma^2}]$ *and* $\forall \underline{t} \ge 0$*, we have*

$$\|\mathbf{u}^{(\underline{t}+1)\mathcal{T}}\|_\infty^{\mathbf{q}} \le 0.9 \|\mathbf{u}^{\underline{t}\mathcal{T}}\|_\infty^{\mathbf{q}}. \tag{25}$$

Theorem 1 implies that if the step size α and the non-sampling rate B are small enough and the period $\mathcal{T}$ of SVRG is long enough, then Algorithm 1 converges linearly to the optimal solution of (1) with a contraction rate of at most 0.9. Taking $\alpha = \mathcal{O}\left(\frac{(1-\sigma^2)^2 \mu M_1}{L^2 M_2^2}\right)$, we can see that

$$\mathcal{T} = \mathcal{O}\left(\frac{\kappa_F^2 \kappa_H^2 \log\frac{\kappa_F \kappa_H}{1-\sigma^2}}{(1-\sigma^2)^2}\right),$$

where $\kappa_F = L/\mu$ is the condition number of the global cost function F, $\kappa_H = M_2/M_1$ is the condition number of the Hessian inverse approximations, and $1 - \sigma^2$ represents the connectedness of the network. Therefore, to obtain a Δ-optimal solution, the total number of stochastic gradient evaluations required by Algorithm 1 is

$$\mathcal{O}\left(\left(\max_i\{m_i\} + \frac{\max_i\{b_i\} \cdot \kappa_F^2 \kappa_H^2 \log\frac{\kappa_F \kappa_H}{1-\sigma^2}}{(1-\sigma^2)^2}\right)\log\frac{1}{\Delta}\right).$$

Remark 2. *If* $m_i = m$ *and* $H_i^k = I_d$ *for all* $i \in \{1, \dots, n\}$ *and* $k \ge 0$*, then* $\gamma = 0$*,* $b_i = \mathcal{O}(1)$*, and* $\kappa_H = 1$*. The number of stochastic gradient evaluations of Algorithm 1 is*

$$\mathcal{O}\left(\left(m + \frac{\kappa_F^2 \log\frac{\kappa_F}{1-\sigma^2}}{(1-\sigma^2)^2}\right)\log\frac{1}{\Delta}\right),$$

which is similar to the bound

$$\mathcal{O}\left(\left(m + \frac{\kappa_F^2 \log\kappa_F}{(1-\sigma^2)^2}\right)\log\frac{1}{\Delta}\right)$$

obtained in [39]. Our analysis cannot show better dependence on κ_F*, but it applies to more general Hessian inverse approximations (i.e., ones that have uniformly bounded positive eigenvalues). The work [52] studies deterministic, quadratic cost functions and shows that communicating Hessians helps to achieve a better dependence on the condition number of cost functions. Our framework applies to stochastic, general cost functions and does not communicate Hessians. In addition, the work [39] suggests setting the batch sizes* $b_i = 1$ *for their variance-reduced stochastic first-order method. Note that smaller batch sizes reduce the number of stochastic gradient evaluations per iteration but increase the number of iterations. For the variance-reduced stochastic second-order methods obtained from our framework, our result suggests using moderate batch sizes. In fact, our numerical experiments will show that slightly larger batch sizes are beneficial. We conjecture that larger batch sizes lead to more stable gradient and Hessian estimators and hence better convergence behavior.*

B. Key Recursion for Establishing the Main Theorem

To prove Theorem 1, a key step is to establish the following recursion for the sequence $\{\mathbf{u}^k\}$.

Proposition 1. *Under the setting of Theorem 1, consider the updates in* (10)*. Let*

$$J := \begin{bmatrix} 1 - \frac{0.99(1-\sigma^2)}{2} & 0.011(1-\sigma^2)\zeta\widetilde{\alpha}\gamma^2 & 0.02\zeta\widetilde{\alpha} \\ 4.1\widetilde{\alpha} & 1 - 0.96\zeta\widetilde{\alpha} & \frac{0.51\widetilde{\alpha}\gamma^2}{1-\sigma^2} \\ 33 & c & 1 - \frac{0.99(1-\sigma^2)}{2} \end{bmatrix},$$

$$Q := \begin{bmatrix} 0.01\widetilde{\alpha}\beta\gamma^2(1-\sigma^2) & 0.01\widetilde{\alpha}\beta\gamma^2(1-\sigma^2) & 0 \\ 0.03\widetilde{\alpha}\zeta(1-\sigma^2)^2 & 0.03\widetilde{\alpha}\zeta(1-\sigma^2)^2 & 0 \\ 2.03\beta & 2.03\beta & 0 \end{bmatrix},$$

where $c := 0.162(1-\sigma^2)\widetilde{\alpha}\zeta + 2.01\beta$. *For all* $k \geq 0$, *we have*

$$\mathbf{u}^{k+1} \leq J\mathbf{u}^k + Q\tilde{\mathbf{u}}^k \tag{26}$$

with

$$\tilde{\mathbf{u}}^k := \begin{bmatrix} \mathbb{E}[\|\tau^k - \mathbf{W}_\infty \tau^k\|^2] \\ \frac{2n}{L}\mathbb{E}\left[F(\overline{\tau}^k) - F(x^*)\right] \\ 0 \end{bmatrix}.$$

Consequently, for all $\underline{t} \geq 0$, *we have*

$$\mathbf{u}^{(\underline{t}+1)\mathcal{T}} \leq \left(J^{\mathcal{T}} + \sum_{\tilde{t}=0}^{\mathcal{T}-1} J^{\tilde{t}} Q\right) \mathbf{u}^{\underline{t}\mathcal{T}}. \tag{27}$$

Proof. See Appendix VIII-B. □

C. Proof of Theorem 1

With the recursion (27) in Proposition 1, we prove (25) in Theorem 1. Let us first bound the spectral radius of J. The following fact from [53] is useful for such a purpose.

Fact 2. *Let* $A \in \mathbb{R}^{d\times d}$ *be non-negative and* $z \in \mathbb{R}^d$ *be positive. If* $Az \leq \bar{\omega} z$ *for some* $\bar{\omega} > 0$, *then* $\rho(A) \leq \|A\|_\infty^z \leq \bar{\omega}$.

Lemma 1. *Under the setting of Theorem 1, we have*

$$J\mathbf{z} \leq \left(1 - \frac{\zeta\widetilde{\alpha}}{2}\right)\mathbf{z},\ \mathbf{z} = [1; z_2; z_3],$$

where $z_2 := \frac{10}{\zeta} + \frac{1.2\gamma^2 z_3}{\zeta(1-\sigma^2)}$ *and* $z_3 := \frac{200(\zeta+\beta)}{\zeta(1-\sigma^2)}$. *Then, we have*

$$\rho(J) \leq \|J\|_\infty^{\mathbf{z}} \leq 1 - \frac{\zeta\widetilde{\alpha}}{2}. \tag{28}$$

Proof. See Part I [54] of the full version of this paper. □

Using (28) and the fact that J is non-negative, we have

$$\sum_{\tilde{t}=0}^{\mathcal{T}-1} J^{\tilde{t}} \leq \sum_{\tilde{t}=0}^{\infty} J^{\tilde{t}} = (I_3 - J)^{-1}.$$

Therefore, it suffices to show that (25) follows from

$$\mathbf{u}^{(\underline{t}+1)\mathcal{T}} \leq \left(J^{\mathcal{T}} + (I_3 - J)^{-1} Q\right) \mathbf{u}^{\underline{t}\mathcal{T}}. \tag{29}$$

Lemma 2. *Under the setting of Theorem 1, we have*

$$(I_3 - J)^{-1} Q\mathbf{q} \leq 0.8\mathbf{q},$$
$$\rho\left((I_3 - J)^{-1}Q\right) \leq \|(I_3 - J)^{-1}Q\|_\infty^{\mathbf{q}} \leq 0.8.$$

Proof. See Part I [54] of the full version of this paper. □

Now, since $\frac{10}{\zeta} \leq z_2 \leq \frac{280}{\zeta(1-\sigma^2)^2}$, we have $\mathbf{q} \leq \mathbf{z} \leq \frac{28}{\zeta(1-\sigma^2)^2}\cdot\mathbf{q}$. According to [53, Theorem 5.6.18], this yields

$$\|J^{\mathcal{T}}\|_\infty^{\mathbf{q}} \leq \frac{28}{\zeta(1-\sigma^2)^2}\cdot\|J^{\mathcal{T}}\|_\infty^{\mathbf{z}}.$$

Upon taking the norm $\|\cdot\|_\infty^{\mathbf{q}}$ on both sides of (29) and then invoking Lemmas 1 and 2, we have

$$\begin{aligned}
\|\mathbf{u}^{(\underline{t}+1)\mathcal{T}}\|_\infty^{\mathbf{q}} &\leq \left\|J^{\mathcal{T}} + (I_3 - J)^{-1} Q\right\|_\infty^{\mathbf{q}} \cdot \|\mathbf{u}^{\underline{t}\mathcal{T}}\|_\infty^{\mathbf{q}} \\
&\leq \left(\|J^{\mathcal{T}}\|_\infty^{\mathbf{q}} + 0.8\right)\left\|\mathbf{u}^{\underline{t}\mathcal{T}}\right\|_\infty^{\mathbf{q}} \\
&\leq \left(\frac{28}{\zeta(1-\sigma^2)^2}\cdot(\|J\|_\infty^{\mathbf{z}})^{\mathcal{T}} + 0.8\right)\left\|\mathbf{u}^{\underline{t}\mathcal{T}}\right\|_\infty^{\mathbf{q}} \\
&\leq \left(\frac{28}{\zeta(1-\sigma^2)^2}\cdot\exp\left\{-\frac{\zeta\widetilde{\alpha}\mathcal{T}}{2}\right\} + 0.8\right)\left\|\mathbf{u}^{\underline{t}\mathcal{T}}\right\|_\infty^{\mathbf{q}},
\end{aligned}$$

where we use the fact that $\|J^{\mathcal{T}}\|_\infty^{\mathbf{z}} \leq (\|J\|_\infty^{\mathbf{z}})^{\mathcal{T}}$ and $1 + a \leq \exp\{a\}, \forall a \in \mathbb{R}$. By setting

$$\mathcal{T} \geq \frac{2\log(280/(\zeta(1-\sigma^2)^2))}{\zeta\widetilde{\alpha}}$$

in the last inequality above, we get (25) and complete the proof of Theorem 1.

V. Convergence Analysis of Proposed Quasi-Newton Methods

In this section, we prove that the Hessian inverse approximations constructed by the proposed two quasi-Newton methods satisfy Assumption 4 and thus fit into the general framework.

A. Analysis of Damped Regularized Limited-Memory DFP

Now, let us prove that the Hessian inverse approximations constructed by the proposed damped regularized limited-memory DFP method have uniformly bounded positive eigenvalues. We begin with the following lemma, which shows the damping technique guarantees that $(\hat{s}_i^p)^T\hat{y}_i^p > 0$ and hence $\lambda_{\min}(H_i^k) > \rho$ for all $i \in \{1,\ldots,n\}$ and $k \geq 0$.

Lemma 3. *Consider the damped regularized limited-memory DFP update in* (16). *We have*

$$0 < \theta_i^p \leq 1 \text{ and } (\hat{s}_i^p)^T\hat{y}_i^p \geq 0.25(\hat{s}_i^p)^T(H_i^{k,(0)} + \epsilon I)^{-1}\hat{s}_i^p.$$

Moreover, with the initialization $H_i^{k,(0)}$ *defined in* (15), *the local Hessian inverse approximation* H_i^{k+1} *output by Algorithm 2 satisfies* $\lambda_{\min}(H_i^{k+1}) > \rho$.

Proof. See Part II [55] of the full version of this paper. □

Based on Lemma 3, the following theorem further gives the specific lower and upper bounds on the eigenvalues of the Hessian inverse approximations generated by Algorithm 2.

Theorem 2. *The local Hessian inverse approximations* $\{H_i^k\}$ *returned by Algorithm 2 satisfy*

$$M_1 I_d \preceq H_i^k \preceq M_2 I_d,\ \forall i \in \{1,\ldots,n\},\ \forall k \geq 0,$$

where $M_1 = \rho + (1+\omega)^{-2M}\left(\frac{1}{\beta} + \frac{1}{4(\mathcal{B}+\epsilon)}\right)^{-1}$, $M_2 = \mathcal{B} + M(4\mathcal{B} + 4\epsilon + \rho)$, *and* $\omega = 4(\mathcal{B}+\epsilon)\left(\tilde{L} + \frac{1}{\beta+\epsilon}\right)$.

Proof. See Appendix VIII-C. □

B. Analysis of Damped Limited-Memory BFGS

Our goal now is to prove that the Hessian inverse approximations constructed by the proposed damped limited-memory BFGS method have uniformly bounded positive eigenvalues. We begin with the following lemma, whose proof is similar to that of Lemma 3.

Lemma 4. *Consider the damped limited-memory BFGS update in* (22). *We have*

$$0 < \theta_i^p \leq 1 \text{ and } (s_i^p)^T\hat{y}_i^p \geq 0.25(s_i^p)^T(H_i^{k,(0)} + \epsilon I_d)^{-1}s_i^p.$$

Moreover, with the initialization $H_i^{k,(0)}$ *defined in* (21), *the local Hessian inverse approximation* H_i^{k+1} *output by Algorithm 3 satisfies* $\lambda_{\min}(H_i^{k+1}) > 0$.

Proof. See Part II [55] of the full version of this paper. □

Based on Lemma 4, the following theorem further gives the specific lower and upper bounds on the eigenvalues of the Hessian inverse approximations generated by Algorithm 3. The proof is similar to that of Theorem 2.

Theorem 3. *The local Hessian inverse approximations* $\{H_i^k\}$ *returned by Algorithm 3 satisfy*

$$M_1 I_d \preceq H_i^k \preceq M_2 I_d,\ \forall i \in \{1,\ldots,n\},\ \forall k \geq 0,$$

where we have $M_1 = \left(\frac{1}{\beta} + \frac{M\omega^2}{4(\mathcal{B}+\epsilon)}\right)^{-1}$, $M_2 = (1+\omega)^{2M}\left(\mathcal{B} + \frac{1}{\tilde{L}(\omega+2)}\right)$, *and* $\omega := 4(\mathcal{B}+\epsilon)\left(\tilde{L} + \frac{1}{\beta+\epsilon}\right)$.

Proof. See Part II [55] of the full version of this paper. □

Remark 3. *Before we proceed, several remarks are in order.*

1) The Hessian inverse approximations constructed by the proposed DFP- and BFGS-type methods satisfy Assumption 4. Thus, by Theorem 1, these methods converge linearly to the optimal solution of (1).

2) As the memory size M *gets larger, the Hessian inverse approximations constructed by the proposed BFGS-type method get closer to being singular. By contrast, the eigenvalues of the Hessian inverse approximations constructed by the proposed DFP-type method are uniformly bounded below by* $\rho > 0$*, regardless of what* M *is. Nevertheless, for both proposed methods, it is likely that the noise caused by randomness and disagreement accumulates more with a larger* M*. On the other hand, observe from the updates of the proposed methods that if the memory size* M *is too small, then there is insufficient second-order curvature information. Thus, we recommend using a moderate memory size* M*, so that the memory and computational costs can be kept low without sacrificing the performance of the methods.*

3) For the proposed DFP-type method, the regularization term ρI_d *in* (16) *lifts the lower bound* M_1 *by* ρ *and lifts the upper bound* M_2 *by* $M\rho$*. It is worth noting that the analysis of Theorem 2 also holds for* $\rho = 0$*. However, we observe from our numerical experiments that a suitably tuned* $\rho > 0$ *can improve the performance of the method.*

4) The analyses in Theorems 2 and 3 also hold for $\epsilon = 0$*. However, we observe from numerical experiments that a suitably tuned* $\epsilon > 0$ *can improve performance of the proposed DFP- and BFGS-type methods, especially the latter one.*

VI. Numerical Experiments

In this section, we use either the DFP-type update in Algorithm 2 or the BFGS-type update in Algorithm 3 to compute the local Hessian inverse approximations $\{H_i^k\}$ in the general framework as shown in Algorithm 1. We then evaluate the two resulting stochastic quasi-Newton methods by solving a least-squares problem with synthetic data in Section VI-A and a logistic regression problem with real data in Section VI-B–VI-E. We randomly generate an undirected, connected graph with n nodes and $\frac{\varrho n(n-1)}{2}$ edges, where $\varrho \in (0,1]$ is the connectivity ratio. The performance metric is the relative error defined as

$$\text{relative error} = \frac{1}{n}\left\|\mathbf{x}^k - \mathbf{x}^*\right\|^2 \Big/ \left\|\mathbf{x}^0 - \mathbf{x}^*\right\|^2,$$

where the optimal solution x^* is pre-computed through a centralized Newton method.

Just as the first-order gradient tracking methods, the proposed second-order methods require two rounds of communication of d-dimension vectors at each time step k. The total number of communication rounds doubles the number of the outer loops K in Algorithm 1. Instead of comparing the communication rounds, we compare the number of epochs on node i defined by $\frac{K}{\mathcal{T}m_i}[m_i + 2b_i(\mathcal{T}-1)]$, which means that within each period, node i access m_i sample costs once and access $2b_i$ sample costs for $\mathcal{T}-1$ times. We set $b_i = b$ and $m_i = m$ for all i in the numerical experiments.

A. Effect of Condition Number

We consider the least-squares problem

$$\min_{x\in\mathbb{R}^d} \frac{1}{2}\sum_{i=1}^{n} \|A_i x - b_i\|^2,$$

where $A_i \in \mathbb{R}^{m\times d}$ and $b_i \in \mathbb{R}^m$ are synthetic data privately owned by node i. Here, we set $m = 500$ and $d = 8$. For simplicity, we define the aggregated variables $A = [A_1; \ldots; A_n] \in \mathbb{R}^{nm\times d}$ and $b = [b_1; \ldots; b_n] \in \mathbb{R}^{nm}$ by vertically stacking the local data. We define the condition number of the problem as

$$\kappa_{LS} = \frac{\lambda_{\max}(A^TA)}{\lambda_{\min}(A^TA)}.$$

To show the effect of the condition number, we generate two groups of data with $\kappa_{LS} = 10$ and $\kappa_{LS} = 2000$, respectively as follows. For $\kappa_{LS} = 10$, we fix $\lambda_{\min}(A^TA) = 0.1$ and $\lambda_{\max}(A^TA) = 1$. For $\kappa_{LS} = 2000$, we fix $\lambda_{\min}(A^TA) = 0.001$ and $\lambda_{\max}(A^TA) = 2$. The other $(d-2)$ eigenvalues are randomly generated within the interval $\left[\lambda_{\min}(A^TA), \lambda_{\max}(A^TA)\right]$. Fig. 1 records the results for $\kappa_{LS} = 10$ and $\kappa_{LS} = 2000$, respectively. The parameters are set as follows. We set $n = 20$ and $\varrho = 0.5$ for the graph. We set $\mathcal{B} = 10^4$ for the two proposed quasi-Newton methods. When $\kappa_{LS} = 10$ (resp., 2000), for the proposed DFP-type method, we set $\alpha = 0.6$ (resp., 0.6), $\rho = 10^{-5}$ (resp., 10^{-5}), $\epsilon = 3$ (resp., 5), $\beta = 0.04$ (resp., 0.01), $\tilde{L} = 10$ (resp., 10), $M = 20$ (resp., 20), and $b_i = 10$ (resp., 15). For the proposed BFGS-type method, we set $\alpha = 0.6$ (resp., 0.6), $\epsilon = 3$ (resp., 37), $\beta = 0.04$ (resp., 0.01), $\tilde{L} = 10$ (resp., 10), $M = 20$ (resp., 50), and $b_i = 10$ (resp., 15). We compare the two proposed decentralized stochastic quasi-Newton methods with four existing decentralized stochastic first-order methods, namely DSA [38], GT-SVRG and GT-SAGA [39], and Acc-VR-DIGing [40]. For DSA, we set the step size $\alpha = 0.9$ (resp., 0.9) and batch size $b_i = 1$ (resp., 1). For GT-SVRG, we set the step size $\alpha = 0.9$ (resp., 0.9) and batch size $b_i = 1$ (resp., 1). For GT-SAGA, we set the step size $\alpha = 0.95$ (resp., 0.95) and batch size $b_i = 1$ (resp., 1). For Acc-VR-DIGing, we set the step size $\alpha = 0.9$ (resp., 0.9), the two parameters $\theta_1 = 0.2$ (resp., 0.1) and $\theta_2 = 0.01$ (resp., 0.01), and the batch size $b_i = 1$ (resp., 1). Note that all the first-order methods use the batch size $b_i = 1$, which yields the fastest convergence in terms of the number of epochs in this set of experiments.

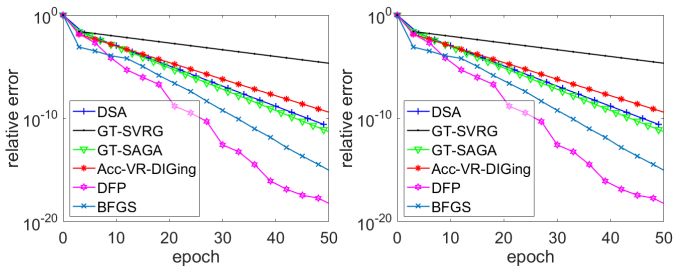

Fig. 1: Least-squares problem with $\kappa_{LS} = 10$ and 2000.

TABLE II: Datasets used in numerical experiments.

Dataset	# of Samples ($\sum_{i=1}^{n} m_i$)	# of Features (d)
covtype	40000	54
cod-rna	52000	8
a6a	11220	123
a9a	32560	123
ijcnn1	91700	22

From Fig. 1, we see the proposed quasi-Newton methods outperform the existing first-order methods, and their advantages are more obvious for the ill-conditioned problem. The DFP-type method performs better than the BFGS-type method in terms of the number of epochs, but the latter has lower storage and computational complexities, as we have discussed in Remark 3.

B. Comparison with First-Order and Second-Order Methods

The ensuing numerical experiments are performed on real datasets. We use the proposed quasi-Newton methods to solve the logistic regression problem

$$\min_{x\in\mathbb{R}^d} \frac{\iota}{2}\|x\|^2 + \frac{1}{n}\sum_{i=1}^{n}\frac{1}{m_i}\sum_{j=1}^{m_i}\ln\left(1+\exp\left(-\left(\mathbf{o}_{ij}^T x\right)\mathbf{p}_{ij}\right)\right),$$

where node i privately owns m_i training samples $(\mathbf{o}_{il}, \mathbf{p}_{il}) \in \mathbb{R}^d \times \{-1, +1\}$; $l = 1, \ldots, m_i$. We use five real datasets,[5] whose attributes are summarized in Table II. We normalize each sample so that $\|\mathbf{o}_{il}\| = 1, \forall i, l$. Note that another way is to normalize each feature, which yields better condition number but is non-trivial to implement in the decentralized setting. A regularization term $\frac{\iota}{2}\|x\|^2$ with $\iota > 0$ is used to avoid over-fitting. We set $n = 20$, $\iota = 0.001$, and $\mathcal{B} = 10^4$ throughout the following numerical experiments. The training samples are randomly and evenly distributed over all nodes.

We compare the two proposed decentralized stochastic quasi-Newton methods with the four decentralized stochastic first-order methods mentioned in the previous sub-section on four real datasets. Different from our previous numerical experiments on synthetic data, where the batch sizes are set as 1, we now use larger batch sizes to boost the performance of the four first-order methods. Fig. 2 depicts the results on covtype, cod-rna, a6a, and a9a, respectively. The parameters are set as follows. We set $n = 20$ and $\varrho = 0.5$ for the graph. When the dataset is covtype (resp., cod-rna, a6a, a9a), for the proposed DFP-type method, we set $\alpha = 0.32$ (resp., $0.3, 0.38, 0.38$), $\rho = 0.01$ (resp., $0.0002, 0.01, 0.001$), $\epsilon = 0.02$ (resp., $0.03, 0.005, 0.1$), $\beta = 0.002$ (resp., $0.002, 0.015, 0.5$), $\tilde{L} = 50$ $(50, 20, 50)$, $M = 3$ (resp., $20, 40, 50$), and $b_i/m_i = 10\%$ (resp., $8\%, 10\%, 6\%$). For the proposed BFGS-type method, we set $\alpha = 0.37$ (resp., $0.35, 0.38, 0.35$), $\epsilon = 0.001$ (resp., $100, 30, 30$), $\beta = 0.002$ (resp., $0.002, 1.2, 0.5$), $\tilde{L} = 50$ (resp., $50, 20, 20$), $M = 3$ (resp., $40, 50, 50$), and $b_i/m_i = 10\%$ (resp., $10\%, 10\%, 10\%$). For GT-SVRG, we set the step size $\alpha = 0.002$ (resp., $0.01, 0.009, 0.004$) and batch size $b_i = 5$ (resp., $2, 1, 2$). For DSA, we set the step size $\alpha = 0.001$ (resp., $0.03, 0.009, 0.008$) and batch size $b_i = 10$ (resp., $10, 1, 2$). For GT-SAGA, we set the step size $\alpha = 0.002$ (resp., $0.009, 0.009, 0.0035$) and batch size $b_i = 5$ (resp., $2, 1, 1$). For Acc-VR-DIGing, we set the step size $\alpha = 0.002$ (resp., $0.03, 0.04, 0.015$), the two parameters $\theta_1 = 0.9$ (resp., $0.07, 0.1, 0.1$) and $\theta_2 = 0.01$ (resp., $0.05, 0.1, 0.1$), and the batch size $b_i = 5$ (resp., $10, 5, 5$).

Fig. 2 shows the proposed methods outperform DSA, GT-SVRG, GT-SAGA, and Acc-VR-DIGing in all four datasets, demonstrating the gains of curvature information from the constructed Hessian inverse approximations. The DFP-type method is better than the BFGS-type method in all four datasets, but again the latter has lower memory requirement and lower computational cost.

Since there is no existing stochastic decentralized quasi-Newton method, we modify the deterministic decentralized BFGS method (abbreviated as D-BFGS) [28] to its stochastic variant (named D-BFGS-SGD) and compare the two on the a9a dataset. The parameters of D-BFGS and D-BFGS-SGD are hand-tuned to the best. We run the proposed methods with different memory sizes $M = 30$, 40, 50, and the other parameters are the same as those used in Fig. 2.

As shown in Fig. 3, our quasi-Newton methods outperform D-BFGS in terms of the number of epochs because the latter accesses all the sample costs in every iteration. D-BFGS-SGD is the slowest due to the existence of stochastic gradient noise. This result validates the effectiveness of our Hessian approximations constructed from the noisy stochastic gradients.

We also compare the computational complexity in terms of runtime in Table III. We report the runtime of Acc-VR-DIGing (which is the best among all the first-order methods), D-BFGS, D-BFGS-SGD, and the proposed DFP and BFGS with different memory sizes. D-BFGS-SGD converges slowly and we just run a fixed number of iterations. For the other methods, we report the runtime to reach below the accuracy of 10^{-3}. Acc-VR-DIGing requires a large number of iterations, although its per-iteration computational complexity is low. D-BFGS is slow due to the computation of inverses of neighboring Hessian approximations with size $\mathbb{R}^{|\mathcal{N}_i|d\times|\mathcal{N}_i|d}$, where $|\mathcal{N}_i|$ is the number of neighbors of node i. D-BFGS-SGD suffers from the same issue as D-BFGS and, in addition, progresses slowly due to the stochastic gradient noise. The proposed DFP and BFGS converge quickly with different memory sizes. Among the two, BFGS is matrix-free and takes the least time.

C. Effect of Batch Size

Here, we numerically evaluate the effect of different batch sizes on the performance of the proposed DFP- and BFGS-type

[5]https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

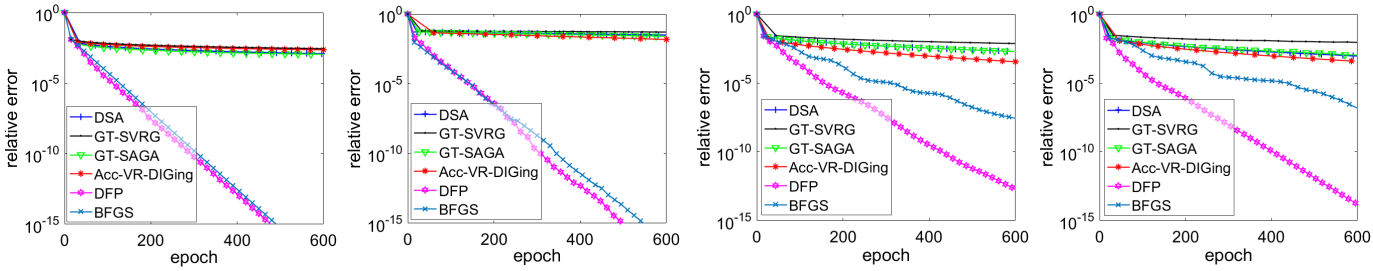

Fig. 2: Comparison with first-order methods on covtype, cod-rna, a6a, and a9a.

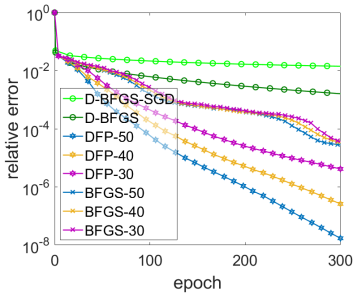

Fig. 3: Comparison with second-order methods on a9a.

TABLE III: Runtime comparison on a9a.

Method	Relative error	Runtime (s)
Acc-VR-DIGing	0.00099	775.2
D-BFGS-SGD	0.014	465.6
D-BFGS	0.00099	322.5
DFP $M = 50$	0.00097	27.9
DFP $M = 40$	0.00093	29.4
DFP $M = 30$	0.00092	34.9
BFGS $M = 50$	0.00099	23.5
BFGS $M = 40$	0.00099	**16.8**
BFGS $M = 30$	0.00095	17.0

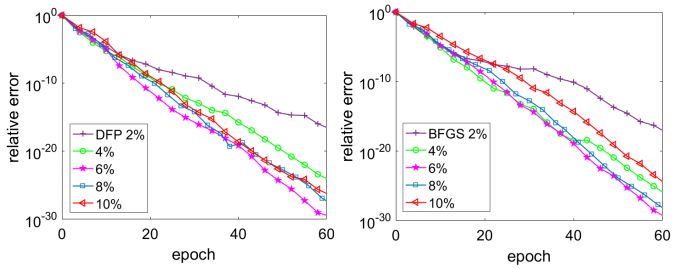

Fig. 4: Effect of batch size of DFP and BFGS on ijcnn1.

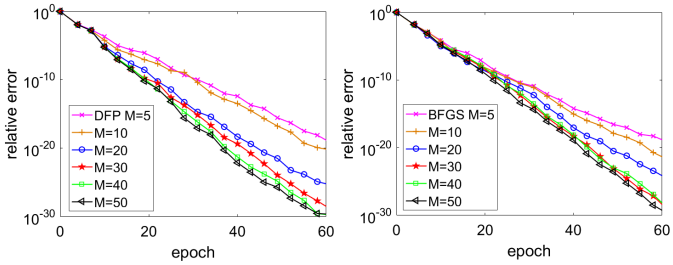

Fig. 5: Effect of memory size of DFP and BFGS on ijcnn1.

methods for solving the logistic regression problem using the real dataset ijcnn1. In Fig. 4, we show the effect of different batch size ratios ($b_i/m_i = 2\%, 4\%, 6\%, 8\%$, and 10%) on the performance of the proposed DFP- and BFGS-type methods, respectively. The parameters are set as follows. For the DFP-type (resp., BFGS-type) method, we set $\alpha = 0.32$ (resp., 0.31), $\rho = 0.005$ (resp., 0), $\epsilon = 0.005$ (resp., 0.005), $\beta = 0.1$ (resp., 0.1), and $M = 50$ (resp., 50). The other settings are the same as those used for Fig. 2.

From Fig.4, we observe that when the batch size is too large or too small, the two methods require more epochs. This is because a smaller batch size results in higher stochastic gradient noise, while a larger batch size calls for more gradient evaluations per iteration. For both proposed methods, a batch size ratio of $b_i/m_i = 6\%$ gives the best performance.

D. Effect of Memory Size

In Fig. 5, we show the effect of different memory sizes ($M = 5, 10, 20, 30, 40$, and 50) on the performance of the proposed DFP- and BFGS-type methods when solving the logistic regression problem using the real dataset ijcnn1, respectively. The parameters are set as follows. For the DFP-type (resp., BFGS-type) method, we set $\alpha = 0.32$ (resp., 0.31), $\rho = 0.004$ (resp., 0), $\epsilon = 0.005$ (resp., 0.002), $\beta = 0.001$ (resp., 0.1), and $b_i/m_i = 6\%$ (resp., 6%). The other settings are the same as those used for Fig. 2.

As Fig. 5 show, a larger memory size generally leads to faster convergence, but the improvement becomes marginal when the memory size is sufficiently large. Therefore, we can use a moderate memory size, which leads to low memory and computational costs.

E. Effect of Graph Topology

In Fig. 6, we show the effect of five different graph topologies (cycle, star, random graphs with connectivity ratios $\varrho = 0.2, 0.3, 0.5$) on the performance of the proposed DFP- and BFGS-type methods when solving the logistic regression problem using the real dataset ijcnn1, respectively. The second largest singular values σ of W (i.e., $\sigma = \|W - \frac{1}{n}1_n 1_n^T\|_2$) of the five graphs are $\sigma = 0.967, 0.950, 0.863, 0.797$, and 0.569, respectively. The parameters are set as follows. For the DFP-type method, we set $\alpha = 0.035$ (resp., $0.02, 0.2, 0.25, 0.32$), $\rho = 0.003$ (resp., $0.001, 0.001, 0.001, 0.005$), $\epsilon = 0.005$ (resp., $0.005, 0.005, 0.005, 0.005$), $\beta = 0.1$ (resp., $0.1, 0.1, 0.1, 0.1$), $M = 50$ (resp., $50, 50, 50, 50$), and $b_i/m_i = 6\%$ (resp., $6\%, 6\%, 6\%, 6\%$). For the BFGS-type method, we set

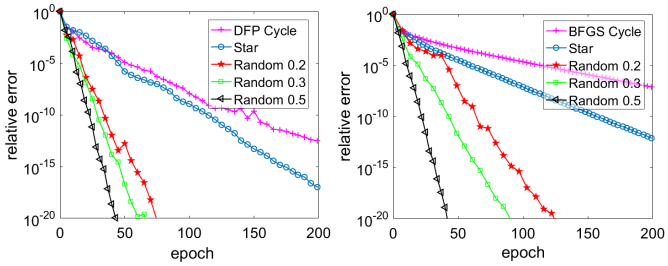

Fig. 6: Effect of topology of DFP and BFGS on ijcnn1.

$\alpha = 0.06$ (resp., $0.07, 0.2, 0.3, 0.31$), $\epsilon = 0.005$ (resp., $0.005, 0.002, 0.002, 0.002$), $\beta = 0.1$ (resp., $0.1, 0.1, 0.1, 0.1$), $M = 50$ (resp., $50, 50, 50, 50$), and $b_i/m_i = 11\%$ (resp., $10\%, 6\%, 6\%, 6\%$). The other settings are the same as those used for Fig. 2.

From Fig. 6, we observe that the proposed DFP- and BFGS-type methods converge linearly on different graphs. For both methods, graphs with smaller σ give faster convergence rates, which corroborates with the results in Theorem 1.

VII. Conclusion

The aim of this work is to develop viable stochastic quasi-Newton methods for decentralized learning. We develop a general algorithmic framework where each node adopts a local inexact quasi-Newton direction that approaches the global one asymptotically. To be specific, each node uses gradient tracking to estimate the average of variance-reduced local stochastic gradients and constructs a local Hessian inverse approximation to exploit the curvature information of the cost function. When the local Hessian inverse approximations are positive definite with uniformly bounded eigenvalues, we prove that the methods obtained from our general framework converge linearly to the exact solution. Moreover, we propose two fully decentralized quasi-Newton methods—namely, the damped regularized limited-memory DFP and the damped limited-memory BFGS—which exploit the curvature of the objective function by constructing locally computable Hessian inverse approximations. We use the damping and limited-memory techniques to ensure that the constructed Hessian inverse approximations have uniformly bounded positive eigenvalues. Our numerical experiments demonstrate that the proposed decentralized stochastic quasi-Newton methods are much faster than the existing decentralized stochastic first-order methods for solving the least-squares and logistic regression problems.

VIII. Appendix

A. Preliminaries

We start with some preliminaries. First, we have the following "averaging" property of the mixing step:

$$\begin{aligned}\|\mathbf{W}\mathbf{x}^k - \mathbf{W}_\infty\mathbf{x}^k\| =& \left\|\left((W - \frac{1}{n}1_n1_n^T)\otimes I_d\right)(\mathbf{x}^k - \mathbf{W}_\infty\mathbf{x}^k)\right\| \\ \leq& \sigma\|\mathbf{x}^k - \mathbf{W}_\infty\mathbf{x}^k\|. \end{aligned} \tag{30}$$

By Assumption 3, we know that $0 \leq \sigma < 1$. Thus, we see from (30) that $\mathbf{W}\mathbf{x}^k$ is closer to the average $\mathbf{W}_\infty\mathbf{x}^k$ than the unmixed $\mathbf{x}^k$. This "averaging" property will be frequently used in our later analysis.

Next, we recall that in (10) the gradient approximation $\mathbf{g}^k$ is updated by dynamic average consensus [46]. Under the initialization $\mathbf{g}^0 = \mathbf{v}^0 = \nabla f(\mathbf{x}^0)$, by taking average over all the nodes and using induction [23], we have

$$\overline{\mathbf{g}}^k = \overline{\mathbf{v}}^k, \ \forall k. \tag{31}$$

This implies that each g_i^k approximately tracks the average of the gradient estimators v_i^k when all g_i^k's are almost consensual.

To handle the randomness caused by sampling, let $\mathcal{F}^k$ be the event generated by $\bigcup_{i\in\{1,\ldots,n\}}^{t\in\{0,\ldots,k-1\}} S_i^t$. For each node i, the stochastic vector v_i^k is an unbiased estimator of the local gradient $\nabla f_i(x_i^k)$ conditioned on $\mathcal{F}^k$. In other words, we have

$$\mathbb{E}\left[v_i^k|\mathcal{F}^k\right] = \nabla f_i\left(x_i^k\right). \tag{32}$$

Further, by (31) and (32), we have

$$\mathbb{E}\left[\overline{\mathbf{g}}^k|\mathcal{F}^k\right] = \mathbb{E}\left[\overline{\mathbf{v}}^k|\mathcal{F}^k\right] = \overline{\nabla f}\left(\mathbf{x}^k\right). \tag{33}$$

On the other hand, under Assumption 1, we have

$$\left\|\overline{\nabla f}\left(\mathbf{x}^k\right) - \nabla F\left(\overline{\mathbf{x}}^k\right)\right\| \leq \frac{L}{\sqrt{n}}\left\|\mathbf{x}^k - \mathbf{W}_\infty\mathbf{x}^k\right\|, \ \forall k. \tag{34}$$

The proof can be found in [23, Lemma 8].

B. Proof of Proposition 1

The proof of Proposition 1 consists of four steps. We bound the consensus error $\mathbb{E}[\|\mathbf{x}^k - \mathbf{W}_\infty\mathbf{x}^k\|^2]$, the network optimality gap $\mathbb{E}[F(\overline{\mathbf{x}}^k) - F(x^*)]$, and the gradient tracking error $\mathbb{E}[\|\mathbf{g}^k - \mathbf{W}_\infty\mathbf{g}^k\|^2]$ in Steps I, II, and III, respectively. These bounds lead to (26). Step IV derives (27) from (26).

1) **Step I:** The following lemma establishes a recursion for the consensus error $\{\mathbb{E}[\|\mathbf{x}^k - \mathbf{W}_\infty\mathbf{x}^k\|^2]\}$.

Lemma 5. *Under the setting of Theorem 1, consider the updates in* (10)*. For all $k \geq 0$, we have*

$$\begin{aligned}&\mathbb{E}\left[\|\mathbf{x}^{k+1} - \mathbf{W}_\infty\mathbf{x}^{k+1}\|^2\right] \\ \leq& \left(\frac{1+\sigma^2}{2} + \frac{2\alpha^2\gamma^2M_2^2L^2}{1-\sigma^2}\right)\mathbb{E}\left[\|\mathbf{x}^k - \mathbf{W}_\infty\mathbf{x}^k\|^2\right] \\ &+ \frac{2\alpha^2M_2^2}{1-\sigma^2}\Big(2\mathbb{E}\left[\|\mathbf{g}^k - \mathbf{W}_\infty\mathbf{g}^k\|^2\right] + \frac{\gamma^2}{n}\mathbb{E}\left[\|\mathbf{v}^k - \nabla f(\mathbf{x}^k)\|^2\right] \\ &+ 2\gamma^2Ln\mathbb{E}\left[F(\overline{\mathbf{x}}^k) - F(x^*)\right]\Big).\end{aligned} \tag{35}$$

Proof. See Section I of the supplementary material. □

As seen from the right-hand side of (35), we need to bound the gradient tracking error $\mathbb{E}[\|\mathbf{g}^k - \mathbf{W}_\infty\mathbf{g}^k\|^2]$, the variance of the gradient estimators $\mathbb{E}[\|\mathbf{v}^k - \nabla f(\mathbf{x}^k)\|^2]$, and the network optimality gap $\mathbb{E}[F(\overline{\mathbf{x}}^k) - F(x^*)]$. Let us first bound the variance of the gradient estimators at time step k.

Lemma 6. *Under the setting of Theorem 1, consider the updates in* (10)*. For all $k \geq 0$, we have*

$$\begin{aligned}&\mathbb{E}\left[\|\mathbf{v}^k - \nabla f(\mathbf{x}^k)\|^2\right] \\ \leq& 4BL^2\cdot\Big(\mathbb{E}\left[\|\mathbf{x}^k - \mathbf{W}_\infty\mathbf{x}^k\|^2\right] + \frac{2n}{L}\mathbb{E}\left[F(\overline{\mathbf{x}}^k) - F(x^*)\right] \\ &+ \mathbb{E}\left[\|\tau^k - \mathbf{W}_\infty\tau^k\|^2\right] + \frac{2n}{L}\mathbb{E}\left[F(\overline{\tau}^k) - F(x^*)\right]\Big).\end{aligned} \tag{36}$$

Proof. See Section II of the supplementary material. □

Different from [39, Lemma 11], our bound on the variance of the gradient estimators in Lemma 6 is related to the non-sampling rate B and is tighter. This is vital for establishing the linear rate from the recursion (27). With Lemmas 5 and 6, we have the following corollary.

Corollary 1. *Under the setting of Theorem 1, consider the updates in* (10)*. For all $k \geq 0$, we have*

$$\begin{aligned}&\mathbb{E}\left[\|\mathbf{x}^{k+1}-\mathbf{W}_\infty\mathbf{x}^{k+1}\|^2\right] \quad (37)\\ \leq& J_{11}\mathbb{E}\left[\|\mathbf{x}^k-\mathbf{W}_\infty\mathbf{x}^k\|^2\right]+J_{12}\cdot\frac{2n}{L}\mathbb{E}\left[F(\overline{\mathbf{x}}^k)-F(x^*)\right]\\ &+J_{13}\cdot\frac{1-\sigma^2}{L^2}\mathbb{E}\left[\|\mathbf{g}^k-\mathbf{W}_\infty\mathbf{g}^k\|^2\right]\\ &+Q_{11}\mathbb{E}\left[\|\tau^k-\mathbf{W}_\infty\tau^k\|^2\right]+Q_{12}\cdot\frac{2n}{L}\mathbb{E}\left[F(\overline{\tau}^k)-F(x^*)\right].\end{aligned}$$

Proof. See Section III of the supplementary material. □

2) **Step II:** Next, we bound the network optimality gap $\mathbb{E}[F(\overline{\mathbf{x}}^k)-F(x^*)]$. We first prove the following lemma.

Lemma 7. *Under the setting of Theorem 1, consider the updates in* (10)*. For all $k \geq 0$, we have*

$$\begin{aligned}&\mathbb{E}\left[\|\overline{\mathbf{d}}^k-\overline{\mathbf{H}}^k\nabla F(\overline{\mathbf{x}}^k)\|^2\right] \quad (38)\\ \leq&\frac{2M_2^2}{n}\Big(L^2\mathbb{E}\left[\|\mathbf{x}^k-\mathbf{W}_\infty\mathbf{x}^k\|^2\right]+\frac{\gamma^2}{4}\mathbb{E}\left[\|\mathbf{g}^k-\mathbf{W}_\infty\mathbf{g}^k\|^2\right]\\ &+\frac{1}{n}\mathbb{E}\left[\|\mathbf{v}^k-\nabla f(\mathbf{x}^k)\|^2\right]\Big).\end{aligned}$$

Proof. See Section IV of the supplementary material. □

With Lemma 7, we are now ready to establish the following recursion for the network optimality gap $\{\mathbb{E}[F(\overline{\mathbf{x}}^k)-F(x^*)]\}$.

Corollary 2. *Under the setting of Theorem 1, consider the updates in* (10)*. For all $k \geq 0$, we have*

$$\begin{aligned}&\frac{2n}{L}\mathbb{E}\left[F(\overline{\mathbf{x}}^{k+1})-F(x^*)\right] \quad (39)\\ \leq&J_{21}\mathbb{E}\left[\|\mathbf{x}^k-\mathbf{W}_\infty\mathbf{x}^k\|^2\right]+J_{22}\cdot\frac{2n}{L}\mathbb{E}\left[F(\overline{\mathbf{x}}^k)-F(x^*)\right]\\ &+J_{23}\cdot\frac{1-\sigma^2}{L^2}\mathbb{E}\left[\|\mathbf{g}^k-\mathbf{W}_\infty\mathbf{g}^k\|^2\right]\\ &+Q_{21}\mathbb{E}\left[\|\tau^k-\mathbf{W}_\infty\tau^k\|^2\right]+Q_{22}\cdot\frac{2n}{L}\mathbb{E}\left[F(\overline{\tau}^k)-F(\tau^*)\right].\end{aligned}$$

Proof. See Section V of the supplementary material. □

3) **Step III:** We now move to establish the following recursion for the gradient tracking error $\{\mathbb{E}[\|\mathbf{g}^k-\mathbf{W}_\infty\mathbf{g}^k\|^2]\}$.

Lemma 8. *Under the setting of Theorem 1, consider the updates in* (10)*. For all $k \geq 0$, we have*

$$\begin{aligned}&\mathbb{E}\left[\|\mathbf{g}^{k+1}-\mathbf{W}_\infty\mathbf{g}^{k+1}\|^2\right] \quad (40)\\ \leq&\frac{1+\sigma^2}{2}\mathbb{E}\left[\|\mathbf{g}^k-\mathbf{W}_\infty\mathbf{g}^k\|^2\right]+\frac{4L^2}{1-\sigma^2}\mathbb{E}\left[\|\mathbf{x}^{k+1}-\mathbf{x}^k\|^2\right]\\ &+\frac{4}{1-\sigma^2}\left(\mathbb{E}\left[\|\mathbf{v}^{k+1}-\nabla f(\mathbf{x}^{k+1})\|^2\right]+\mathbb{E}\left[\|\mathbf{v}^k-\nabla f(\mathbf{x}^k)\|^2\right]\right).\end{aligned}$$

Proof. See Section VI of the supplementary material. □

The right-hand side of (40) suggests that we need to bound the difference of two successive iterations $\|\mathbf{x}^{k+1}-\mathbf{x}^k\|^2$ and the variance of the gradient estimators $\mathbb{E}[\|\mathbf{v}^{k+1}-\nabla f(\mathbf{x}^{k+1})\|^2]$. This is achieved in the following two lemmas.

Lemma 9. *Under the setting of Theorem 1, consider the updates in* (10)*. For all $k \geq 0$, we have*

$$\begin{aligned}&\mathbb{E}\left[\|\mathbf{x}^{k+1}-\mathbf{x}^k\|^2\right] \quad (41)\\ \leq&8.01\mathbb{E}\left[\|\mathbf{x}^k-\mathbf{W}_\infty\mathbf{x}^k\|^2\right]+\frac{4\alpha^2M_2^2}{n}\mathbb{E}\left[\|\mathbf{v}^k-\nabla f(\mathbf{x}^k)\|^2\right]\\ &+16\alpha^2M_2^2L\cdot n\mathbb{E}\left[F(\overline{\mathbf{x}}^k)-F(x^*)\right]\\ &+4\alpha^2M_2^2\mathbb{E}\left[\|\mathbf{g}^k-\mathbf{W}_\infty\mathbf{g}^k\|^2\right].\end{aligned}$$

Proof. See Section VII of the supplementary material. □

Lemma 10. *Under the setting of Theorem 1, consider the updates in* (10)*. For all $k \geq 0$, we have*

$$\begin{aligned}&\mathbb{E}\left[\|\mathbf{v}^{k+1}-\nabla f(\mathbf{x}^{k+1})\|^2\right] \quad (42)\\ \leq&4L^2B\Big(\mathbb{E}\left[\|\mathbf{x}^k-\mathbf{W}_\infty\mathbf{x}^k\|^2\right]+\frac{3\alpha M_2^2}{LM_1}\mathbb{E}\left[\|\mathbf{g}^k-\mathbf{W}_\infty\mathbf{g}^k\|^2\right]\\ &+1.01\cdot\frac{2n}{L}\mathbb{E}\left[F(\overline{\mathbf{x}}^k)-F(x^*)\right]+1.01\mathbb{E}\left[\|\tau^k-\mathbf{W}_\infty\tau^k\|^2\right]\\ &+1.01\cdot\frac{2n}{L}\mathbb{E}\left[F(\overline{\tau}^k)-F(x^*)\right]\Big).\end{aligned}$$

Proof. See Section VIII of the supplementary material. □

Corollary 3. *Under the setting of Theorem 1, consider the updates in* (10)*. For all $k \geq 0$, we have*

$$\begin{aligned}&\frac{1-\sigma^2}{L^2}\mathbb{E}\left[\|\mathbf{g}^{k+1}-\mathbf{W}_\infty\mathbf{g}^{k+1}\|^2\right] \quad (43)\\ \leq&J_{31}\mathbb{E}\left[\|\mathbf{x}^k-\mathbf{W}_\infty\mathbf{x}^k\|^2\right]+J_{32}\cdot\frac{2n}{L}\mathbb{E}\left[F(\overline{\mathbf{x}}^k)-F(x^*)\right]\\ &+J_{33}\cdot\frac{1-\sigma^2}{L^2}\mathbb{E}\left[\|\mathbf{g}^k-\mathbf{W}_\infty\mathbf{g}^k\|^2\right]\\ &+Q_{31}\mathbb{E}\left[\|\tau^k-\mathbf{W}_\infty\tau^k\|^2\right]+Q_{32}\frac{2n}{L}\mathbb{E}\left[F(\overline{\tau}^k)-F(\tau^*)\right].\end{aligned}$$

Proof. See Section IX of the supplementary material. □

4) **Step IV:** Combining Corollaries 1, 2, and 3 directly gives (26). Next, we establish (27). By unrolling (26), we get

$$\begin{aligned}\mathbf{u}^{(\underline{t}+1)\mathcal{T}}&\leq J^{\mathcal{T}}\mathbf{u}^{\underline{t}\mathcal{T}}+J^{\mathcal{T}-1}Q\tilde{\mathbf{u}}^{\underline{t}\mathcal{T}}+\cdots+Q\tilde{\mathbf{u}}^{\underline{t}\mathcal{T}+\mathcal{T}-1}\\ &\leq J^{\mathcal{T}}\mathbf{u}^{\underline{t}\mathcal{T}}+\left(J^{\mathcal{T}-1}Q+\cdots+J^0Q\right)\tilde{\mathbf{u}}^{\underline{t}\mathcal{T}}\\ &\leq\left(J^{\mathcal{T}}+\sum_{\tilde{t}=0}^{\mathcal{T}-1}J^{\tilde{t}}Q\right)\mathbf{u}^{\underline{t}\mathcal{T}},\end{aligned}$$

where we use the fact that $\tau^{\underline{t}\mathcal{T}}=\cdots=\tau^{\underline{t}(\mathcal{T}+1)-1}=\mathbf{x}^{\underline{t}\mathcal{T}}$ for SVRG in the second inequality and $\mathbb{E}[\|\tau^{\underline{t}\mathcal{T}}-\mathbf{W}_\infty\tau^{\underline{t}\mathcal{T}}\|^2]=\mathbb{E}[\|\mathbf{x}^{\underline{t}\mathcal{T}}-\mathbf{W}_\infty\mathbf{x}^{\underline{t}\mathcal{T}}\|^2]$ so that $Q\tilde{\mathbf{u}}^{\underline{t}\mathcal{T}}=Q\mathbf{u}^{\underline{t}\mathcal{T}}$ in the third inequality. This completes the proof of Proposition 1.

C. Proof of Theorem 2

Since our argument holds for any node, we again omit the node index i in the proof. First, we establish the upper

bound. According to (16), we know that $H^{k,(t+1)} \preceq H^{k,(t)} + \frac{\hat{s}^p(\hat{s}^p)^T}{(\hat{s}^p)^T\hat{y}^p} + \rho I_d$. This implies that

$$\begin{aligned}\|H^{k,(t+1)}\|_2 &\leq \|H^{k,(t)}\|_2 + \left\|\frac{\hat{s}^p(\hat{s}^p)^T}{(\hat{s}^p)^T\hat{y}^p}\right\|_2 + \rho \\ &\leq \|H^{k,(t)}\|_2 + \frac{\|\hat{s}^p\|^2}{(\hat{s}^p)^T\hat{y}^p} + \rho,\end{aligned} \tag{44}$$

where we use $(\hat{s}^p)^T\hat{y}^p > 0$ in the last inequality. Then, with Lemma 3, we have

$$(\hat{s}^p)^T\hat{y}^p \geq 0.25(\hat{s}^p)^T\left(H^{k,(0)} + \epsilon I_d\right)^{-1}\hat{s}^p \geq \frac{0.25\|\hat{s}^p\|^2}{\mathcal{B}+\epsilon}, \tag{45}$$

where the last inequality holds since $H^0 \preceq \mathcal{B}I_d$. Substituting (45) into (44), we get

$$\|H^{k,(t+1)}\|_2 \leq \|H^{k,(t)}\|_2 + 4(\mathcal{B}+\epsilon) + \rho.$$

Unrolling the above recurrence gives

$$\begin{aligned}\|H^{k+1}\|_2 = \|H^{k,(\tilde{M})}\|_2 &\leq \|H^{k,(0)}\|_2 + \tilde{M}(4\mathcal{B}+4\epsilon+\rho) \\ &\leq \mathcal{B} + M(4\mathcal{B}+4\epsilon+\rho).\end{aligned}$$

This establishes the upper bound $M_2 = \mathcal{B} + M(4\mathcal{B}+4\epsilon+\rho)$.

Now, using (13), (14), and the fact that $0 < \theta^p \leq \frac{\tilde{L}\|\hat{s}^p\|}{\|y^p\|}$, we bound

$$\begin{aligned}\|\hat{y}^p\| &\leq \theta^p\|y^p\| + (1-\theta^p)\|(H^{k,(0)}+\epsilon I_d)^{-1}\hat{s}^p\| \\ &\leq \tilde{L}\|\hat{s}^p\| + \frac{1}{\beta+\epsilon}\|\hat{s}^p\|.\end{aligned} \tag{46}$$

Furthermore, using the Sherman-Morrison-Woodbury formula on (16), we get

$$\begin{aligned}&\left(H^{k,(t+1)} - \rho I_d\right)^{-1} \\ =&\left(I_d - \frac{\hat{y}^p(\hat{s}^p)^T}{(\hat{s}^p)^T\hat{y}^p}\right)\left(H^{k,(t)}\right)^{-1}\left(I_d - \frac{\hat{s}^p(\hat{y}^p)^T}{(\hat{s}^p)^T\hat{y}^p}\right) + \frac{\hat{y}^p(\hat{y}^p)^T}{(\hat{s}^p)^T\hat{y}^p}.\end{aligned} \tag{47}$$

Consider the two terms on the right-hand side of (47). Observe that

$$\left\|\frac{\hat{y}^p(\hat{s}^p)^T}{(\hat{s}^p)^T\hat{y}^p}\right\|_2 \leq \frac{\|\hat{y}^p\|\cdot\|\hat{s}^p\|}{(\hat{s}^p)^T\hat{y}^p} \leq 4(\mathcal{B}+\epsilon)\left(\tilde{L} + \frac{1}{\beta+\epsilon}\right), \tag{48}$$

where we use (45) and (46) to get the last inequality. Moreover, we have

$$\left\|\frac{\hat{y}^p(\hat{y}^p)^T}{(\hat{s}^p)^T\hat{y}^p}\right\|_2 \leq \frac{\|\hat{y}^p\|^2}{(\hat{s}^p)^T\hat{y}^p} \leq 4(\mathcal{B}+\epsilon)\left(\tilde{L} + \frac{1}{\beta+\epsilon}\right)^2, \tag{49}$$

where we again use (45) and (46) to get the last inequality. Letting $\omega := 4(\mathcal{B}+\epsilon)\left(\tilde{L} + \frac{1}{\beta+\epsilon}\right)$ and taking the norm on both sides of (47), we get

$$\begin{aligned}&\left\|\left(H^{k,(t+1)} - \rho I_d\right)^{-1}\right\|_2 \\ \leq&\left(1 + \left\|\frac{\hat{y}^p(\hat{s}^p)^T}{(\hat{s}^p)^T\hat{y}^p}\right\|_2\right)^2 \cdot \left\|\left(H^{k,(t)}\right)^{-1}\right\|_2 + \left\|\frac{\hat{y}^p(\hat{y}^p)^T}{(\hat{s}^p)^T\hat{y}^p}\right\|_2 \\ \leq&(1+\omega)^2\left\|\left(H^{k,(t)}\right)^{-1}\right\|_2 + \frac{\omega^2}{4(\mathcal{B}+\epsilon)} \\ \leq&(1+\omega)^2\left\|\left(H^{k,(t)} - \rho I_d\right)^{-1}\right\|_2 + \frac{\omega^2}{4(\mathcal{B}+\epsilon)},\end{aligned} \tag{50}$$

where the second inequality follows from (48) and (49) and the last inequality follows from the fact that $H^{k,(t)} \succ H^{k,(t)} - \rho I_d \succ 0$. Unrolling the above recurrence gives

$$\begin{aligned}&\left\|\left(H^{k,(\tilde{M})} - \rho I_d\right)^{-1}\right\|_2 \\ \leq&(1+\omega)^{2(M-1)}\left(\left\|\left(H^{k,(1)} - \rho I_d\right)^{-1}\right\|_2 + \frac{\frac{\omega^2}{4(\mathcal{B}+\epsilon)}}{(1+\omega)^2-1}\right) \\ \leq&(1+\omega)^{2(M-1)}\left(\left\|\left(H^{k,(1)} - \rho I_d\right)^{-1}\right\|_2 + \frac{1}{4(\mathcal{B}+\epsilon)}\right),\end{aligned} \tag{51}$$

where the second inequality follows from

$$\frac{\frac{\omega^2}{4(\mathcal{B}+\epsilon)}}{(1+\omega)^2-1} = \frac{\omega}{(\omega+2)4(\mathcal{B}+\epsilon)} < \frac{1}{4(\beta+\epsilon)}.$$

Besides, by setting $t=0$ in the second inequality of (50) and using (15), we know that

$$\left\|\left(H^{k,(1)} - \rho I_d\right)^{-1}\right\|_2 \leq (1+\omega)^2\beta^{-1} + \frac{\omega^2}{4(\mathcal{B}+\epsilon)}.$$

Substituting the above inequality into (51), we get

$$\begin{aligned}&\left\|\left(H_i^{k,(\tilde{M})} - \rho I_d\right)^{-1}\right\|_2 \\ \leq&(1+\omega)^{2(M-1)}\left((1+\omega)^2\beta^{-1} + \frac{1+\omega^2}{4(\mathcal{B}+\epsilon)}\right) \\ \leq&(1+\omega)^{2M}\left(\beta^{-1} + \frac{1}{4(\mathcal{B}+\epsilon)}\right),\end{aligned} \tag{52}$$

where we use $1+\omega^2 \leq (1+\omega)^2$ in the last inequality. By taking the inverse on both sides of (52), we get

$$\lambda_{\min}\left(H^{k,(\tilde{M})}\right) \geq \rho + (1+\omega)^{-2M}\left(\frac{1}{\beta} + \frac{1}{4(\mathcal{B}+\epsilon)}\right)^{-1}.$$

This establishes the lower bound $M_1 = \rho + (1+\omega)^{-2M}\left(\frac{1}{\beta} + \frac{1}{4(\mathcal{B}+\epsilon)}\right)^{-1}$ and completes the proof.

REFERENCES

[1] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, "Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 5336–5346.

[2] H. Tang, X. Lian, M. Yan, C. Zhang, and J. Liu, "D^2: Decentralized training over decentralized data," in *Proceedings of International Conference on Machine Learning*, 2018, pp. 4848–4856.

[3] T. S. Brisimi, R. Chen, T. Mela, A. Olshevsky, I. C. Paschalidis, and W. Shi, "Federated learning of predictive models from federated electronic health records," *International Journal of Medical Informatics*, vol. 112, pp. 59–67, 2018.

[4] S. Warnat-Herresthal, H. Schultze, K. L. Shastry, S. Manamohan, S. Mukherjee, V. Garg, R. Sarveswara, K. Händler, P. Pickkers, N. A. Aziz *et al.*, "Swarm learning for decentralized and confidential clinical machine learning," *Nature*, vol. 594, no. 7862, pp. 265–270, 2021.

[5] D. K. Molzahn, F. Dörfler, H. Sandberg, S. H. Low, S. Chakrabarti, R. Baldick, and J. Lavaei, "A survey of distributed optimization and control algorithms for electric power systems," *IEEE Transactions on Smart Grid*, vol. 8, no. 6, pp. 2941–2962, 2017.

[6] G. Tsaousoglou, P. Pinson, and N. G. Paterakis, "Transactive energy for flexible prosumers using algorithmic game theory," *IEEE Transactions on Sustainable Energy*, vol. 12, no. 3, pp. 1571–1581, 2021.

[7] J. Nocedal and S. Wright, *Numerical optimization*. Springer Science & Business Media, 2006.

[8] A. Bordes, L. Bottou, and P. Gallinari, "SGD-QN: Careful quasi-Newton stochastic gradient descent," *Journal of Machine Learning Research*, vol. 10, pp. 1737–1754, 2009.

[9] R. H. Byrd, G. M. Chin, W. Neveitt, and J. Nocedal, "On the use of stochastic Hessian information in optimization methods for machine learning," *SIAM Journal on Optimization*, vol. 21, no. 3, pp. 977–995, 2011.

[10] R. H. Byrd, S. L. Hansen, J. Nocedal, and Y. Singer, "A stochastic quasi-Newton method for large-scale optimization," *SIAM Journal on Optimization*, vol. 26, no. 2, pp. 1008–1031, 2016.

[11] N. N. Schraudolph, J. Yu, and S. Günter, "A stochastic quasi-Newton method for online convex optimization," in *Artificial Intelligence and Statistics*, 2007, pp. 436–443.

[12] A. Mokhtari and A. Ribeiro, "Global convergence of online limited memory BFGS," *The Journal of Machine Learning Research*, vol. 16, no. 1, pp. 3151–3181, 2015.

[13] ——, "RES: Regularized stochastic BFGS algorithm," *IEEE Transactions on Signal Processing*, vol. 62, no. 23, pp. 6089–6104, 2014.

[14] A. Lucchi, B. McWilliams, and T. Hofmann, "A variance reduced stochastic Newton method," *arXiv preprint arXiv:1503.08316*, 2015.

[15] P. Moritz, R. Nishihara, and M. Jordan, "A linearly-convergent stochastic L-BFGS algorithm," in *Artificial Intelligence and Statistics*, 2016, pp. 249–258.

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

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

[18] W. Shi, Q. Ling, G. Wu, and W. Yin, "EXTRA: An exact first-order algorithm for decentralized consensus optimization," *SIAM Journal on Optimization*, vol. 25, no. 2, pp. 944–966, 2015.

[19] T. H. Chang, M. Hong, and X. Wang, "Multi-agent distributed optimization via inexact consensus ADMM," *IEEE Transactions on Signal Processing*, vol. 63, no. 2, pp. 482–497, 2014.

[20] H. Sun and M. Hong, "Distributed non-convex first-order optimization and information processing: Lower complexity bounds and rate optimal algorithms," *IEEE Transactions on Signal processing*, vol. 67, no. 22, pp. 5912–5928, 2019.

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

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

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

[24] G. Scutari and Y. Sun, "Distributed nonconvex constrained optimization over time-varying digraphs," *Mathematical Programming*, vol. 176, no. 1, pp. 497–544, 2019.

[25] A. Mokhtari, Q. Ling, and A. Ribeiro, "Network Newton distributed optimization methods," *IEEE Transactions on Signal Processing*, vol. 65, no. 1, pp. 146–161, 2016.

[26] F. Mansoori and E. Wei, "A fast distributed asynchronous Newton-based optimization algorithm," *IEEE Transactions on Automatic Control*, vol. 65, no. 7, pp. 2769–2784, 2019.

[27] N. K. Jerinkić, D. Jakovetić, N. Krejić, and D. Bajović, "Distributed second-order methods with increasing number of working nodes," *IEEE Transactions on Automatic Control*, vol. 65, no. 2, pp. 846–853, 2019.

[28] M. Eisen, A. Mokhtari, and A. Ribeiro, "Decentralized quasi-Newton methods," *IEEE Transactions on Signal Processing*, vol. 65, no. 10, pp. 2613–2628, 2017.

[29] A. Mokhtari, W. Shi, Q. Ling, and A. Ribeiro, "DQM: Decentralized quadratically approximated alternating direction method of multipliers," *IEEE Transactions on Signal Processing*, vol. 64, no. 19, pp. 5158–5173, 2016.

[30] M. Eisen, A. Mokhtari, and A. Ribeiro, "A decentralized quasi-Newton method for dual formulations of consensus optimization," in *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, 2016, pp. 1951–1958.

[31] ——, "A primal-dual quasi-Newton method for exact consensus optimization," *IEEE Transactions on Signal Processing*, vol. 67, no. 23, pp. 5983–5997, 2019.

[32] J. Zhang, Q. Ling, and A. M.-C. So, "A Newton tracking algorithm with exact linear convergence for decentralized consensus optimization," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 7, pp. 346–358, 2021.

[33] B. Li, S. Cen, Y. Chen, and Y. Chi, "Communication-efficient distributed optimization in networks with gradient tracking and variance reduction," in *Proceedings of International Conference on Artificial Intelligence and Statistics*, 2020, pp. 1662–1672.

[34] A. Daneshmand, G. Scutari, P. Dvurechensky, and A. Gasnikov, "Newton method over networks is fast up to the statistical precision," in *International Conference on Machine Learning*, 2021, pp. 2398–2409.

[35] J. Zhang, K. You, and T. Başar, "Distributed adaptive Newton methods with global superlinear convergence," *Automatica*, vol. 138, p. 110156, 2022.

[36] S. Pu and A. Nedić, "Distributed stochastic gradient tracking methods," *Mathematical Programming*, vol. 187, p. 409–457, 2021.

[37] S. Pu, A. Olshevsky, and I. C. Paschalidis, "A sharp estimate on the transient time of distributed stochastic gradient descent," *IEEE Transactions On Automatic Control*, 2021.

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

[39] R. Xin, U. A. Khan, and S. Kar, "Variance-reduced decentralized stochastic optimization with accelerated convergence," *IEEE Transactions on Signal Processing*, vol. 68, pp. 6255–6271, 2020.

[40] H. Li, Z. Lin, and Y. Fang, "Variance reduced EXTRA and DIGing and their optimal acceleration for strongly convex decentralized optimization," *Journal of Machine Learning Research*, vol. 23, pp. 1–41, 2022.

[41] R. Xin, U. A. Khan, and S. Kar, "Fast decentralized nonconvex finite-sum optimization with recursive variance reduction," *SIAM Journal on Optimization*, vol. 32, no. 1, pp. 1–28, 2022.

[42] H. Hendrikx, F. Bach, and L. Massoulié, "Dual-free stochastic decentralized optimization with variance reduction," *Advances in Neural Information Processing Systems*, vol. 33, pp. 19 455–19 466, 2020.

[43] R. Xin, S. Kar, and U. A. Khan, "Decentralized stochastic optimization and machine learning: A unified variance-reduction framework for robust performance and fast convergence," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 102–113, 2020.

[44] H. Sun, S. Lu, and M. Hong, "Improving the sample and communication complexity for decentralized non-convex optimization: Joint gradient estimation and tracking," in *International Conference on Machine Learning*, 2020, pp. 9217–9228.

[45] R. Xin, U. Khan, and S. Kar, "A hybrid variance-reduced method for decentralized stochastic non-convex optimization," in *International Conference on Machine Learning*. PMLR, 2021, pp. 11 459–11 469.

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

[47] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*. Springer Science & Business Media, 2003, vol. 87.

[48] S. Boyd, P. Diaconis, and L. Xiao, "Fastest mixing Markov chain on a graph," *SIAM Review*, vol. 46, no. 4, pp. 667–689, 2004.

[49] S. U. Pillai, T. Suel, and S. Cha, "The Perron-Frobenius theorem: Some of its applications," *IEEE Signal Processing Magazine*, vol. 22, no. 2, pp. 62–75, 2005.

[50] H. Chen, H.-C. Wu, S.-C. Chan, and W.-H. Lam, "A stochastic quasi-Newton method for large-scale nonconvex optimization with applications," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 11, pp. 4776–4790, 2019.

[51] X. Wang, S. Ma, D. Goldfarb, and W. Liu, "Stochastic quasi-Newton methods for nonconvex stochastic optimization," *SIAM Journal on Optimization*, vol. 27, no. 2, pp. 927–956, 2017.

[52] E. Berglund, S. Magnusson, and M. Johansson, "Distributed Newton method over graphs: Can sharing of second-order information eliminate the condition number dependence," *IEEE Signal Processing Letters*, vol. 28, pp. 1180–1184, 2021.

[53] R. A. Horn and C. R. Johnson, *Matrix analysis*. Cambridge University Press, 2012.

[54] J. Zhang, H. Liu, A. M.-C. So, and Q. Ling, "Variance-reduced stochastic quasi-Newton methods for decentralized learning—Part I: General framework," *arXiv preprint arXiv:2201.07699*, 2022.

[55] ——, "Variance-reduced stochastic quasi-Newton methods for decentralized learning—Part II: Damped limited-memory DFP and BFGS methods," *arXiv preprint arXiv:2201.07733*, 2022.