Variance-Reduced Stochastic Quasi-Newton Methods for Decentralized Learning: Part II

Jiaojiao Zhang\textsuperscript{1}, Huikang Liu\textsuperscript{2}, Anthony Man-Cho So\textsuperscript{1}, and Qing Ling\textsuperscript{3}

Abstract—In Part I of this work, we have proposed a general framework of decentralized stochastic quasi-Newton methods, which converge linearly to the optimal solution under the assumption that the local Hessian inverse approximations have bounded positive eigenvalues. In Part II, we specify two fully decentralized stochastic quasi-Newton methods, damped regularized limited-memory DFP (Davidon-Fletcher-Powell) and damped limited-memory BFGS (Broyden-Fletcher-Goldfarb-Shanno), to locally construct such Hessian inverse approximations without extra sampling or communication. Both of the methods use a fixed moving window of \( M \) past local gradient approximations and local decision variables to adaptively construct positive definite Hessian inverse approximations with bounded eigenvalues, satisfying the assumption in Part I for the linear convergence. For the proposed damped regularized limited-memory DFP, a regularization term is added to improve the performance. For the proposed damped limited-memory BFGS, a two-loop recursion is applied, leading to low storage and computation complexity. Numerical experiments demonstrate that the proposed quasi-Newton methods are much faster than the existing decentralized stochastic first-order algorithms.

Index Terms—decentralized optimization, stochastic quasi-Newton methods, damped limited-memory DFP, damped limited-memory BFGS

I. INTRODUCTION

With the explosive growth of big data and the urgent need for privacy protection, decentralized learning has become attractive. In decentralized learning, local machines store large-scale data and collaboratively train models. In Part I of this work, we have considered a decentralized learning problem over an undirected and connected network with \( n \) nodes, in the form of

\[
x^* = \arg \min_{x \in \mathbb{R}^d} F(x) = \frac{1}{n} \sum_{i=1}^{n} f_i(x),
\]

Here, \( x \) is the decision variable and \( f_i \) is the average of \( m_i \) sample costs such that

\[
f_i(x) = \frac{1}{m_i} \sum_{l=1}^{m_i} f_{i,l}(x),
\]

where \( f_{i,l}: \mathbb{R}^d \to \mathbb{R} \) is the \( l \)-th sample cost on node \( i \) and \( f_{i,l} \) is not accessible by any other nodes. The network is described by an undirected and connected graph \( \mathcal{G} = (\mathcal{V}, \mathcal{E}) \) with node set \( \mathcal{V} = \{1, \ldots, n\} \) and edge set \( \mathcal{E} \subseteq \mathcal{V} \times \mathcal{V} \). Nodes \( i \) and \( j \) are neighbors and allowed to communicate with each other if and only if they are connected with an edge \((i, j) \in \mathcal{E}\). We define \( \mathcal{N}_i \) as the set of neighbors of node \( i \) including itself. All the nodes cooperate to find the optimal solution \( x^* \) to \((1)\) using computation on their local costs \( f_i \) and information received from their neighbors.

In Part I, we have established a general framework to solve \((1)\), incorporating quasi-Newton approximations with variance reduction so as to achieve fast convergence. With initializations \( \tau_i^0 = x_i^0 \) and \( g_i^0 = \nabla f_i(x_i^0) \), at time \( k+1 \) node \( i \) updates its local decision variable \( x_i^{k+1} \) as

\[
x_i^{k+1} = \sum_{j=1}^{n} w_{ij} x_j^k - \alpha d_i^k,
\]

\[
v_i^{k+1} = \frac{1}{b_i} \sum_{l \in S_i^{k+1}} \left( \nabla f_{i,l}(x_i^{k+1}) - \nabla f_{i,l}(\tau_i^{k+1}) \right) + \nabla f_i(\tau_i^{k+1}),
\]

\[
g_i^{k+1} = \sum_{j=1}^{n} w_{ij} g_j^k + v_i^{k+1} - v_i^k,
\]

\[
d_i^{k+1} = H_i^{k+1} g_i^k.
\]

Here, \( \alpha > 0 \) is the step size, \( W = [w_{ij}] \in \mathbb{R}^{n \times n} \) is the mixing matrix, \( S_i^{k+1} \subseteq \{1, \ldots, m_i\} \) with batch size \( b_i \), while \( \tau_i^{k} = \tau_i^{k+1} \) or \( \tau_i^{k+1} = x_i^{k+1} \) if \( \text{mod}(k+1, T) = 0 \). The general framework \((2)\) can be written in a compact form, as

\[
x^{k+1} = W x^k - \alpha d^k,
\]

\[
g^{k+1} = W g^k + v^{k+1} - v^k,
\]

\[
d^{k+1} = H^{k+1} g^k,
\]

where the notations can be found in Part I.

At each time \( k \), each node \( i \) computes a local approximate Newton direction \( d_i^k \) by the local gradient approximation \( g_i^k \) and the local Hessian inverse approximation \( H_i^k \). The local gradient approximation \( g_i^k \) is obtained by the dynamic average consensus method \((1)\) to track the average of the variance-reduced local stochastic gradient \( v_i^k \), while the local Hessian inverse approximation \( H_i^k \) is constructed by quasi-Newton methods with the local decision variable \( x_i^k \) and the gradient approximation \( g_i^k \). Part I proves that the proposed general framework \((3)\) converges linearly to the optimal solution of \((1)\), given that the Hessian inverse approximations \( H_i^k \) satisfy the following assumption.

Assumption 1. There exist two constants \( M_1 \) and \( M_2 \) with \( 0 < M_1 \leq M_2 < 2 \) such that

\[
M_1 I_d \preceq H_i^k \preceq M_2 I_d, \quad \forall \ i = 1, \ldots, n, \ k \geq 0.
\]

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 Business School, Imperial College London.

Qing Ling is with the School of Computer Science and Engineering and Guangdong Province Key Laboratory of Computational Science, Sun Yat-Sen University, as well as the Pazhou Lab.
In Part II of this work, we focus on how to construct Hessian inverse approximations $H^k_i$ satisfying Assumption [1] with fully decentralized quasi-Newton methods.

We first review deterministic and stochastic quasi-Newton methods for solving (1) in the centralized setting and then move on to the decentralized setting. In the centralized deterministic setting, quasi-Newton methods usually update in the form of

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

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

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

and

$$(BFGS)\quad H^{k+1} = H^k - \frac{H^k y^k (s^k)^T + s^k (y^k)^T H^k}{(s^k)^T y^k} \left(1 + \frac{y^k (H^k y^k)}{(s^k)^T y^k}\right),$$

respectively. Here, $s^k$ and $y^k$ are defined as

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

If the cost function $F(x)$ is strongly convex, the curvature condition $(s^k)^T y^k > 0$ holds and thus the Hessian inverse approximations $H^k$ via [5] and [6] preserve positive definiteness given a positive definite initialization such that $H^0 > 0$ [2].

When the number of samples is very large, computing the full gradient $\nabla F$ is prohibitive, which motivates the development of stochastic methods. In the centralized stochastic setting, there are many works which combine stochastic gradient descent with carefully constructed curvature information [3]-[6]. For example, [3] investigates how to construct a diagonal or low-rank matrix according to the secant condition. The work of [4] incorporates sub-sampled Hessian information in a Newton conjugate gradient method and a limited-memory quasi-Newton method for statistical learning. An online limited-memory BFGS using stochastic gradients is proposed in [6], in lieu of the full gradient in BFGS update; the convergence analysis is given in [7]. The work of [8] proposes a regularized stochastic BFGS (RES) method where stochastic gradients are used both as descent directions and constituents of Hessian estimates. The regularization technique ensures that the eigenvalues of the Hessian approximations are uniformly bounded. The works of [9] and [10] take advantage of variance reduction to eliminate the stochastic gradient noise, such that the resultant stochastic quasi-Newton methods are provably convergent at linear rates.

Although the deterministic and stochastic quasi-Newton methods have been widely used in the centralized setting, they cannot be used directly in decentralized optimization. Taking the decentralized network topology into consideration, each node is only allowed to communicate with its neighbors, which leads to the lack of global gradient and Hessian information. In the decentralized deterministic setting, there are a few works exploring the decentralized quasi-Newton methods with the penalization technique [11], [12], in the dual domain [13], and in the primal-dual domain [14]. However, to the best of our knowledge, computationally affordable decentralized stochastic second-order methods have not been investigated.

In Part II of this work, we propose two fully decentralized quasi-Newton methods to construct local Hessian inverse approximations $H^k_i$ fitting into the general framework: All $H^k_i$ satisfy Assumption [1] and are constructed only with the local decision variables $x^k_i$ and the local gradient approximations $g^k_i$. Note that using the gradient approximations to construct the Hessian inverse approximations is quite adventurous, since the gradient approximations are noisy due to stochastic gradient noise and disagreement among the nodes. Naively adopting centralized quasi-Newton methods may end up with almost-singular Hessian inverse approximations, or even non-positive semidefinite ones. To tackle these issues, the proposed methods use the damping and limited-memory techniques so as to adaptively construct positive definite Hessian inverse approximations with bounded eigenvalues.

**Notations.** We use $\| \cdot \|$ to denote the Euclidean norm of a vector. $\text{tr}(\cdot)$, $\| \cdot \|_F$ and $\| \cdot \|_2$ denote the trace, the Frobenius norm and the spectral norm of a matrix, respectively. $I_d \in \mathbb{R}^{d \times d}$ denotes the $d \times d$ identity, and $1_n \in \mathbb{R}^n$ denotes the $n$-dimensional column vector of all ones. $A \succeq 0$ and $A \succ 0$ mean that $A$ is positive semidefinite and positive definite, respectively. We use $A \succeq B$ and $A \succ B$ to denote $A - B \succeq 0$ and $A - B \succ 0$, respectively. $\lambda_{\text{max}}(\cdot)$, $\lambda_{\text{min}}(\cdot)$ denote the largest and smallest eigenvalues of a matrix, respectively. The $i$-th largest eigenvalue of a matrix is denoted by $\lambda_i(\cdot)$. We use $A^2$ to denote the square root of a positive semidefinite matrix $A$ such that $A = A^2 A^2$. Define the aggregated variable $x = [x_1; \cdots; x_n] \in \mathbb{R}^{nd}$ for $x_1, \cdots, x_n \in \mathbb{R}^d$, and similar aggregation rules apply to other variables $d$, $g$, and $v$. Define a block diagonal matrix $H^R_i \in \mathbb{R}^{nd \times nd}$ whose $i$-th block is $H^k_i \in \mathbb{R}^{d \times d}$.

**II. DAMPED REGULARIZED LIMITED-MEMORY DFP FOR HESSIAN INVERSE APPROXIMATION**

In this section, we propose a fully decentralized stochastic quasi-Newton approximation approach called the damped regularized limited-memory DFP, to locally construct the Hessian inverse approximations. Each node $i$ only uses its own gradient approximations $g^{k+1}_i$ and decision variables $x^{k+1}_i$ to construct $H^{k+1}_i$, and does not need extra communication with its neighbors. Therefore, from now on we do not specify the node index $i$ until the end of derivation.

As we have emphasized in Section II, constructing a reliable Hessian inverse approximation is challenging in the decentralized stochastic setting. Since the gradient approximations $g^{k+1}_i$ are noisy due to stochastic gradient noise and disagreement among the nodes, naively adopting centralized quasi-Newton methods may lead to almost-singular Hessian inverse approximations (i.e., either $\lambda_{\text{min}}(H^{k+1}) \to 0$ or $\lambda_{\text{max}}(H^{k+1}) \to \infty$) or even non-positive semidefinite ones.
A. Damped Regularized Limited-memory DFP

It is known that the update of DFP is obtained by minimizing the Gaussian differential entropy subject to certain constraints. Inspired by [8], [15], to avoid \( \lambda_{\min}(H^{k+1}) \to 0 \), we add a regularization term with parameter \( \rho > 0 \) to the minimization problem, given by

\[
H^{k+1} = \arg \min_{Z \in \mathbb{R}^{d \times d}} \text{tr}[(H^k)^{-1}(Z - \rho I_d)] - \log \det((H^k)^{-1}(Z - \rho I_d)),
\]

s.t. \( Z y^k = s^k, \ Z \succeq 0, \)

where \( s^k = x^{k+1} - x^k \) is variable variation and \( y^k = g^{k+1} - g^k \) is gradient approximation variation. If we let \( \rho = 0 \), then \[7\] reduces to the traditional DFP.

Define a modified variable variation \( \tilde{s}^k \) as

\[
\tilde{s}^k = s^k - \rho y^k,
\]

The work [8] has proved that the closed-form solution to \[7\] is given by

\[
H^{k+1} = H^k + \frac{\tilde{s}^k(\tilde{s}^k)^T}{\tilde{s}^k y^k} - \frac{H^k y^k (y^k)^T H^k}{(y^k)^T H^k y^k} + \rho I_d.
\]

In the centralized stochastic setting where \( g^k \) is the stochastic gradient instead of the stochastic gradient approximation, [8] shows that if \( \rho \) is properly chosen and \( H^0 > 0 \), then \( (\hat{z}^k)^T y^k > 0 \) and thus \( \lambda_{\min}(H^{k+1}) > \rho \) for all \( k \). In addition, [8] also establishes the upper bound on the eigenvalues of \( H^{k+1} \) for all \( k \). Unfortunately, these satisfactory results no longer hold in the decentralized stochastic setting, since \( y^k = g^{k+1} - g^k \) suffers from both stochastic gradient noise and disagreement among the nodes. To address this issue, we further propose the damped regularized limited-memory DFP as follows.

To preserve positive semidefiniteness and lower boundedness of \( H^k \), we combine a damping technique with the regularized DFP in [3]. To be specific, given a suitable \( H^0 > 0 \), we define \( \hat{y}^k \) as

\[
\hat{y}^k = \theta^k y^k + (1 - \theta^k)(H^0 + \epsilon I_d)^{-1} \tilde{s}^k,
\]

where \( \epsilon > 0 \) is a constant and \( \theta^k \) is adaptively computed by

\[
\theta^k = \min \left\{ \hat{\theta}^k, \frac{\hat{L} \| \tilde{s}^k \|}{\| y^k \|} \right\},
\]

with a parameter \( \hat{L} > 0 \) and the widely used parameter \( \hat{\theta}^k \) defined as [15]

\[
\hat{\theta}^k = \begin{cases} 
0.75 (\hat{s}^k)^T (H^0 + \epsilon I_d)^{-1} \tilde{s}^k, \\
(\hat{s}^k)^T (H^0 + \epsilon I_d)^{-1} \tilde{s}^k - (\hat{s}^k)^T y^k, \\
1, \text{ otherwise.}
\end{cases}
\]

As we will show later, with the added term \( \frac{\hat{L} \| \tilde{s}^k \|}{\| y^k \|} \) in [11], \( \hat{y}^k \) can be upper bounded in terms of \( \| y^k \| \). Then, we replace \( y^k \) with \( \hat{y}^k \) in \[9\] to obtain the damped regularized DFP update

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

As we will prove in Lemma [1] the damping technique guarantees that \( (\hat{s}^k)^T \hat{y}^k > 0 \), which yields \( \lambda_{\min}(H^k) > \rho \).

On the other hand, we also have to guarantee \( \lambda_{\max}(H^{k+1}) < \infty \). This is nontrivial since \( \hat{y}^k \) is noisy and the regularization term \( \rho I_d \) may accumulate when the algorithm evolves. Inspired by [15], [16], we use the limited-memory technique to tackle this issue. Usually, the limited-memory technique is combined with BFGS to reduce the memory and computation costs. In contrast, here we combine it with DFP to make the eigenvalues of \( H^k \) bounded.

From now on, we specify the node index \( i \). For time \( k \), set an initial Hessian inverse approximation

\[
H^{k,(0)}_i = \min \left\{ \max \left\{ \frac{(s_i^k)^T s_i^k}{(s_i^k)^T y_i^k} + \rho, \beta \right\}, B \right\} I_d,
\]

where \( \beta > 0 \) and \( B > 0 \) are two parameters to guarantee that \( \beta I_d \leq H^{k,(0)}_i \leq BI_d \) in initialization. Given the two sequences \( \{s_i^k\} \) and \( \{y_i^p\}, p = k + 1 - M, \ldots, k \), where \( M = \min\{k + 1, M\} \) and \( M \) is the memory size, the damped regularized limited-memory DFP at node \( i \) updates as

\[
H^{k,(t+1)}_i = H^{k,(t)}_i + \frac{s_i^p (s_i^p)^T}{(s_i^p)^T y_i^p} - \frac{H^{k,(t)}_i y_i^p (y_i^p)^T H^{k,(t)}_i}{(y_i^p)^T H^{k,(t)}_i y_i^p} + \rho I_d,
\]

where \( p = k + 1 - M + t \) and \( t = 0, \ldots, M - 1 \). At the end of this inner loop, we set \( H^{k+1}_i = H^{k,(M)}_i \).

The damped regularized limited-memory DFP for Hessian inverse approximation at node \( i \) for time \( k \) is summarized in Algorithm [1].

Algorithm 1 Damped regularized limited-memory DFP for Hessian inverse approximation at node \( i \) for time \( k \)

**Require:** \( \rho; \beta; B; c; L; M \).

1. Update variable variation \( s_i^k = x_i^{k+1} - x_i^k \).
2. Update gradient variation \( y_i^k = y_i^{k+1} - y_i^k \).
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 [10].
5. Set \( M = \min\{k + 1, M\} \) and load \( \{s_i^p, y_i^p\}_{p=k+1-M} \).
6. Initialize \( H^{k,(0)}_i \) as in [14].
7. for \( t = 0, \ldots, M - 1 \) do
8. Update \( H^{k,(t+1)}_i \) as in [15].
9. end for
10. Output \( H^{k+1}_i = H^{k,(M)}_i \).

B. Bounded Positive Eigenvalues of \( H^k_i \) Constructed by DFP

In this section, we will prove that the Hessian inverse approximations constructed by the proposed damped regularized limited-memory DFP are positive definite and have bounded eigenvalues.

Consider the update [15]. The following Lemma shows the damping technique guarantees that \( (\hat{s}_i^k)^T \hat{y}_i^k > 0 \), which yields \( \lambda_{\min}(H^k_i) > \rho \).
Lemma 1. Considering the update in (15), with the corrected \(\hat{y}^p\) by the damping technique, we have

\[
0 < \theta^p_i \leq 1 \text{ and } (\hat{y}^p)^T \hat{y}^p \geq 0.25(\hat{y}^p)^T (H_i^{k(0)} + \epsilon I)^{-1} \hat{y}^p.
\]

Moreover, with the initialization \(H_i^{k(0)}\) defined in (14), \(H_i^{k+1}\) generated by the damped regularized limited-memory DFP in Algorithm 7 on node \(i\) keeps positive definite, such that \(\lambda_{\min}(H_i^{k+1}) > \rho\).

**Proof.** Since the analysis holds for any node, we omit the node index \(i\) in the proof. With \(\theta^p\) defined in (12), if \((\hat{y}^p)^T y^p > 0.25(\hat{y}^p)^T (H_i^{k(0)} + \epsilon I)_i^{-1} \hat{y}^p\), we have \(\theta^p = 1\). If \((\hat{y}^p)^T y^p \leq 0.25(\hat{y}^p)^T(H_i^{k(0)} + \epsilon I)_i^{-1} \hat{y}^p\), substituting this inequality into the definition of \(\bar{\theta}^p\), we have

\[
\bar{\theta}^p = \frac{0.75(\hat{y}^p)^T(H_i^{k(0)} + \epsilon I)_i^{-1} \hat{y}^p}{(\hat{y}^p)^T(H_i^{k(0)} + \epsilon I)_i^{-1} \hat{y}^p - 0.25(\hat{y}^p)^T (H_i^{k(0)} + \epsilon I)_i^{-1} \hat{y}^p}.
\]

\[
\leq 0.75(\hat{y}^p)^T(H_i^{k(0)} + \epsilon I)_i^{-1} \hat{y}^p = 1.
\]

Obviously, with \(H_i^{k(0)} \succ 0\), we have \(\bar{\theta}^p > 0\). Since \(\theta^p = \min \left\{ \theta^p, \frac{\hat{y}^p}{\hat{y}^p} \right\}\), with \(0 < \bar{\theta}^p \leq 1\), we have \(0 < \theta^p \leq 1\).

Moreover, substituting the definitions of \(\bar{y}^p\) in (10) and \(\theta^p\) in (11), we compute

\[
(\hat{y}^p)^T \hat{y}^p = (\hat{y}^p)^T (\theta^p y^p + (1 - \theta^p)(H_i^{k(0)} + \epsilon I)_i^{-1} \hat{y}^p)
\]

\[
= \theta^p \left[ (\hat{y}^p)^T y^p - (\hat{y}^p)^T (H_i^{k(0)} + \epsilon I)_i^{-1} \hat{y}^p \right] + (\hat{y}^p)^T (H_i^{k(0)} + \epsilon I)_i^{-1} \hat{y}^p
\]

\[
= \begin{cases} 
0.25(\hat{y}^p)^T (H_i^{k(0)} + \epsilon I)_i^{-1} \hat{y}^p, & \text{if } (\hat{y}^p)^T y^p \leq 0.25(\hat{y}^p)^T (H_i^{k(0)} + \epsilon I)_i^{-1} \hat{y}^p, \\
(\hat{y}^p)^T \hat{y}^p, & \text{otherwise},
\end{cases}
\]

which implies

\[
(\hat{y}^p)^T \hat{y}^p \geq 0.25(\hat{y}^p)^T (H_i^{k(0)} + \epsilon I)_i^{-1} \hat{y}^p.
\]

By the initialization \(H_i^{k(0)}\) in (14), we have \((\hat{y}^p)^T \hat{y}^p > 0\).

To show that \(H_i^{k+1}\) is positive definite, we rewrite (15) as

\[
H_i^{k,(t+1)} = \frac{\hat{s}^T \hat{y}}{\hat{s}^T \hat{y}} + H_i^{k} \left( I_d - \frac{H_i^{k} \hat{y} (H_i^{k} \hat{y})^T}{(H_i^{k} \hat{y})^T H_i^{k} \hat{y}} \right) H_i^{k} \hat{y} + \rho I_d,
\]

where we omit all the time indexes at the right-hand side of (15) for simplicity. Since \(0 < \hat{s}^T \hat{y} \leq H_i^{k}\), \(\hat{s}^T \hat{y} > 0\), we conclude that \(H_i^{k,(t+1)}\) is positive definite and \(\lambda_{\min}(H_i^{k,(t+1)}) > \rho\), which completes the proof.

Based on Lemma 1, the following theorem further gives the specific lower and upper bounds for the eigenvalues of \(H_i^{k}\) generated by Algorithm 7.

**Theorem 1.** Consider the damped regularized limited-memory DFP in Algorithm 7. We have

\[
M_1 I_d \preceq H_i^{k} \preceq M_2 I_d, \forall i,
\]

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

**Proof.** Since the analysis holds for any node, we omit the node index \(i\) in the proof. First, we establish the upper bound. According to (15), we know that \(H_i^{k,(t+1)} \leq H_i^{k,(t)} + \tilde{y}^p(\hat{y}^p)^T \hat{y}^p + \rho I_d\), which implies

\[
\| H_i^{k,(t+1)} \|_2 \leq \| H_i^{k,(t)} \|_2 + \| \tilde{y}^p(\hat{y}^p)^T \hat{y}^p + \rho \|
\]

\[
\leq \| H_i^{k,(t)} \|_2 + \| \tilde{y}^p \| \| \hat{y}^p \| + \rho,
\]

where we use \((\hat{y}^p)^T \hat{y}^p > 0\) in the last inequality. Then, with Lemma 1 we have

\[
(\hat{y}^p)^T \hat{y}^p \geq 0.25(\hat{y}^p)^T (H_i^{k(0)} + \epsilon I)_i^{-1} \hat{y}^p
\]

\[
\geq 0.25\| \hat{y}^p \|^2/B + \epsilon,
\]

where the last inequality holds since \(H_i^{0} \preceq B I_d\). Substituting (18) into (17), we get

\[
\| H_i^{k,(t+1)} \|_2 \leq \| H_i^{k,(t)} \|_2 + 4(B + \epsilon) + \rho.
\]

Following the standard argument for recurrence, from (19) we get

\[
\| H_i^{k+1} \|_2 = \| H_i^{k(M)} \|_2 \leq \| H_i^{k(0)} \|_2 + M(4B + 4\epsilon + \rho)
\]

\[
\leq B + M(4B + 4\epsilon + \rho).
\]

Thus, we get the upper bound \(M_2 = B + M(4B + 4\epsilon + \rho)\).

Next, we establish the lower bound. According to (10) and (11), with the added term \(\tilde{y}^p \| \hat{y}^p \| \) in (11), \(\| \hat{y}^p \| \) can be upper bounded in terms of \(\| \hat{y}^p \| \) such that

\[
\| \hat{y}^p \| \leq \theta^p \| y^p \| + (1 - \theta^p) \| (H_i^{k(0)} + \epsilon I)_i^{-1} \hat{y}^p \|
\]

\[
\leq \tilde{L} \| \hat{y}^p \| + \frac{1}{\beta + \epsilon} \| \hat{y}^p \|,
\]

where we use \(0 < \theta^p \leq \tilde{L} \| \hat{y}^p \| \) in the second inequality. Using Sherman-Morrison-Woodbury formula (2) on (15), we get

\[
H_i^{k,(t+1)} - \rho I_d
\]

\[
= I_d - \tilde{y}^p(\hat{y}^p)^T (\hat{y}^p)^T \hat{y}^p + \rho I_d
\]

\[
= \tilde{y}^p(\hat{y}^p)^T (\hat{y}^p)^T \hat{y}^p + \rho I_d
\]

\[
\frac{1}{(\hat{y}^p)^T \hat{y}^p}.
\]

With (18) and (20), we bound the two terms at the right-hand side of (21) as follows. The first term at the right-hand side of (21) satisfies

\[
\| \tilde{y}^p(\hat{y}^p)^T \|_2 \leq \| \hat{y}^p \| \| \hat{y}^p \| + 4(B + \epsilon) \left( \tilde{L} + \frac{1}{\beta + \epsilon} \right),
\]

where we substitute (18) and (20) in the last inequality. For the second term at the right-hand side of (21), we have

\[
\| \tilde{y}^p(\hat{y}^p)^T \|_2 \leq \| \hat{y}^p \| \| \hat{y}^p \| + 4(B + \epsilon) \left( \tilde{L} + \frac{1}{\beta + \epsilon} \right)^2.
\]
where we also substitute (15) and (20) in the last inequality. For simplicity, let \( \omega = 4(B + \epsilon) \left( \tilde{L} + \frac{1}{\beta + \epsilon} \right) \), then taking norm on both sides of (21), we have
\[
\left\| (H^{k,(t+1)} - \rho I_d)^{-1} \right\|_2 \\
\leq \left( 1 + \left\| \tilde{y}^T \right\|_2 \right)^2 \cdot \left\| (H^{k,(t)})^{-1} \right\|_2 + 2 \cdot \left\| \tilde{y}^T \right\|_2 \cdot \left\| \tilde{y}^T \right\|_2
\]
\[
\leq (1 + \omega)^2 \left\| (H^{k,(t)})^{-1} \right\|_2 + \frac{\omega^2}{4(B + \epsilon)},
\]
where we substitute (22) and (23) in the second inequality, and use the fact that \( H^{k,(t)} > H^{k,(t)} - \rho I_d > 0 \) in the last inequality. A standard argument for recurrence on (24) shows that
\[
\left\| (H^{k,(M)})^{-1} - \rho I_d \right\|_2 \\
\leq (1 + \omega)^2 \left( (H^{k,(1)})^{-1} \right) + \frac{\omega^2}{4(B + \epsilon)}
\]
\leq (1 + \omega)^2 \left( (H^{k,(1)})^{-1} \right) + \frac{\omega^2}{4(B + \epsilon)},
\]
where to derive the second inequality we use
\[
\frac{\omega^2}{(1 + \omega)^2 - 1} = \frac{\omega}{(1 + \omega) + 4(B + \epsilon)} < \frac{1}{4(B + \epsilon)}.
\]
Besides, by setting \( t = 0 \) in the second inequality of (24), we know that
\[
\left\| (H^{k,(1)})^{-1} \right\|_2 \leq (1 + \omega)^2 \beta^{-1} + \frac{\omega^2}{4(B + \epsilon)}.
\]
Substituting the above inequality into (25), we get
\[
\left\| (H^{k,(M)})^{-1} - \rho I_d \right\|_2 \\
\leq (1 + \omega)^2 \left( \left\| (H^{k,(1)})^{-1} \right\|_2 + \frac{\omega^2}{4(B + \epsilon)} \right)
\]
\leq (1 + \omega)^2 \left( \beta^{-1} + \frac{1}{4(B + \epsilon)} \right),
\]
where we use \( 1 + \omega^2 \leq (1 + \omega)^2 \) in the last inequality. By taking inverse on both sides of (26), we get
\[
\lambda_{\min} \left( H^{k,(M)} \right) \geq \rho + (1 + \omega)^{-2M} \left( \frac{1}{\beta} + \frac{1}{4(B + \epsilon)} \right)^{-1}.
\]
Thus, we obtain the lower bound of \( M_1 = \rho + (1 + \omega)^{-2M} \left( \frac{1}{\beta} + \frac{1}{4(B + \epsilon)} \right)^{-1} \), which completes the proof.

III. DAMPED LIMITED-MEMORY BFGS FOR HESSIAN INVERSE APPROXIMATION

In this section, we propose a damped limited-memory BFGS method to construct the local Hessian inverse approximations. Compared with the damped regularized limited-memory DFP method in Section II, the limited-memory technique here is used not only for bounding the Hessian inverse approximations, but also for reducing storage and computation costs. To be specific, the proposed method can be implemented by a two-loop recursion, where \( H^k \) is not generated explicitly, and only its multiplication with vectors are computed.

A. Damped Limited-memory BFGS

Since the discussion holds for any node, from now on we omit the node index \( i \) until the end of derivation. The traditional BFGS is the solution to an optimization problem given by
\[
H^{k+1} = \arg \min_Z \| Z - H^k \|_Q, \quad \text{s.t. } Zy^k = s^k, Z = Z^T.
\]
where \( \| \cdot \|_Q \) is a weighted Frobenius norm defined as \( \| H \|_Q = \| Q^{1/2} H Q^{1/2} \|_F \) and \( Q \) is an arbitrary positive semidefinite matrix satisfying the relation \( Qs^k = y^k \) [2]. Note that in our decentralized setting, \( s^k = x^{k+1} - x^k \) is variable variation and \( y^k = g^k + 1 - g^k \) is gradient approximation variation. The solution of the semidefinite optimization in (27) is therefore the closest to \( H^k \) in the sense of weighted norm among all symmetric matrices that satisfy the secant condition. The closed form solution of (27) is given by
\[
H^{k+1} = H^k - \frac{H^k y^k (s^k)^T + s^k (y^k)^T H^k}{(s^k)^T y^k} \\
+ \frac{s^k (s^k)^T y^k}{(s^k)^T y^k} \left( \frac{s^k (s^k)^T}{(s^k)^T y^k} \right) y^k, \quad \text{s.t. } Zy^k = s^k, Z = Z^T.
\]
As we have mentioned in Section II, the gradient approximations \( g^k \) are noisy such that it is nontrivial to preserve positive semidefiniteness and lower boundedness of \( H^k \) (i.e., \( \lambda_{\min}(H^k) \) and \( \lambda_{\max}(H^k) \) must be both positive and finite). Naively adopting the BFGS update in (28) is unable to achieve this goal. Therefore, we combine the damping technique with the traditional BFGS method in (28). To be specific, given a suitable \( H^k > 0 \), we define \( \tilde{y}^k \) as
\[
\tilde{y}^k = \theta y^k + (1 - \theta) (H^0 + \epsilon I)^{-1} s^k,
\]
where \( \epsilon > 0 \) is a parameter and \( \theta \) is adaptively computed by
\[
\theta^k = \min \left\{ \tilde{y}^k, \| s^k \| \right\},
\]
with a parameter \( \tilde{L} > 0 \) and the widely used parameter \( \tilde{L} \) defined as [15]
\[
\tilde{L}^k = \frac{0.75(s^k)^T (H^0 + \epsilon I)^{-1} s^k}{(s^k)^T y^k},
\]
if \( (s^k)^T y^k \leq 0.25(s^k)^T (H^0 + \epsilon I)^{-1} s^k, \)
1, otherwise.

Similar to the proposed DFP-based method, with the added term \( \frac{\tilde{L}^k s^k}{\| s^k \|} \) in (30), \( \tilde{y}^k \) can be upper bounded in terms of \( \| s^k \| \). Then, we replace \( \tilde{y}^k \) in (28) with \( \tilde{y}^k \), such that the damped BFGS becomes
\[
H^{k+1} = H^k - \frac{H^k \tilde{y}^k (s^k)^T + s^k (\tilde{y}^k)^T H^k}{(s^k)^T \tilde{y}^k} \\
+ \frac{s^k (s^k)^T \tilde{y}^k}{(s^k)^T \tilde{y}^k} \left( 1 + (\tilde{y}^k)^T H^k \tilde{y}^k \right).
\]
As we will prove later, the damping technique guarantees that \((s^k)^T y^k > 0\), such that \(\lambda_{\min}(H^{k+1}) > 0\).

On the other hand, to guarantee that \(\lambda_{\max}(H^{k+1}) < \infty\), we use the limited-memory technique. From now on, we specify the node index \(i\). For time \(k\), set an initial Hessian inverse approximation as

\[
H_i^{(0)} = \min \left\{ \max \left\{ \frac{(s_i^k)^T y_i^k}{(y_i^k)^T y_i^k}, \beta \right\}, B \right\} I_d. \tag{33}
\]

Given the two sequences \(\{s_i^k\}\) and \(\{p_i^k\}\), \(p = k + 1 - M, \ldots, k\), where \(M = \min\{k + 1, M\}\) and \(M\) is the memory size, the damped limited-memory BFGS at node \(i\) updates as

\[
H_i^{k,(t+1)} = H_i^{k,(t)} - \frac{H_i^{k,(t)} p_i^T (p_i s_i^T) H_i^{k,(t)} p_i}{s_i^T p_i (p_i s_i^T) H_i^{k,(t)} p_i + s_i^T p_i (p_i s_i^T) H_i^{k,(t)} p_i}
\]

where \(p = k + 1 - M + t\) and \(t = 0, \ldots, M - 1\). The second equality will be used for the analysis. At the end of this inner loop, we set \(H_i^{k+1} = H_i^{k,(M)}\).

Algorithm 2

**Damped limited-memory BFGS for Newton direction approximation run on agent \(i\)**

- **Require**: \(\beta; B; \epsilon; L; M\).
  1. Update variable variation \(s_i^k = s_i^{k+1} - s_i^k\).
  2. Update the gradient variation \(y_i^k = y_i^{k+1} - y_i^k\).
  3. Update modified gradient variation \(\tilde{y}_i^k\) as in (29).
  4. Initialize \(H_i^{k,(0)}\) as in (33).
  5. Set \(M = \min\{k + 1, M\}\) and load \(\{s_i^k, \tilde{y}_i^k\}_{p=k+1-M}^k\).
  6. Perform two-loop limited-memory BFGS in Algorithm 3.
  7. Output direction \(d_i^{k+1} = H_i^{k+1} y_i^{k+1}\).

**Remark 1.** Compared with the damped regularized limited-memory DFP method, the proposed damped limited-memory BFGS does not use the regularization term parameterized by \(p\). The reason is that adding such a regularization term \(p I_d\) at the end of update (34) will make it difficult to realize the two-loop recursion. How to develop decentralized regularized BFGS which is friendly to two-loop recursion will be our future work. The memory requirement and computation cost per iteration of the proposed BFGS are \(O(Md)\) and \(O(Md^2)\), respectively. In contrast, the memory requirement and computation cost per iteration of the proposed DFP are \(O(d^2 + Md)\) and \(O(Md^2)\), respectively.

**Algorithm 3 Two-loop limited-memory BFGS run on agent \(i\)**

Set \(q_i \leftarrow y_i^{k+1}\).

For \(p = k, k-1, \ldots, k + 1 - M\),

\[
\alpha_i^p \leftarrow (s_i^p)^T q_i^{p+1},
\]

\[
q_i \leftarrow q_i - \alpha_i^p p_i^p.
\]

**end for**

\[
r_i \leftarrow H_i^{k,(0)} q_i,
\]

For \(p = k + 1 - M, k - M, \ldots, k\),

\[
\beta_i \leftarrow (s_i^p)^T r_i,
\]

\[
r_i \leftarrow r_i + s_i^p (\alpha_i^p - \beta_i).
\]

**end for**

Output \(H_i^{k+1} y_i^{k+1} = r_i\).

B. Bounded Positive Eigenvalues of \(H_i^k\) Constructed by BFGS

In the following analysis, we will prove that the Hessian inverse approximations constructed by the proposed damped limited-memory BFGS are positive definite and have bounded eigenvalues.

Consider the update (34). The following Lemma shows the damping technique guarantees that \((s_i^p)^T y_i^p > 0\), which yields \(\lambda_{\min}(H_i^k) > 0\). The proof is similar to that of Lemma 1; we write it down here for completeness.

**Lemma 2.** Considering the update in (34), with the corrected \(\tilde{y}_i^p\) by the damping technique, we have

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

Moreover, with the initialization \(H_i^{k,(0)}\) defined in (35), \(H_i^{k+1}\) generated by the damped limited-memory BFGS in Algorithm 2 on each node keeps positive definite and \(\lambda_{\min}(H_i^{k+1}) > 0\).

**Proof.** Since the analysis holds for any node, we omit the node index \(i\) in the proof. With \(\tilde{y}_i^p\) defined in (31), if \((s_i^p)^T y_i^p > 0.25(s_i^p)^T (H_i^{k,(0)} + \epsilon I_d)^{-1} s_i^p\), we have \(\theta_i^p = 1\). If \((s_i^p)^T y_i^p \leq 0.25(s_i^p)^T (H_i^{k,(0)} + \epsilon I_d)^{-1} s_i^p\), substituting this inequality into the definition of \(\theta_i^p\), we have

\[
\tilde{\theta}_i^p = \frac{0.75(s_i^p)^T (H_i^{k,(0)} + \epsilon I_d)^{-1} s_i^p}{(s_i^p)^T (H_i^{k,(0)} + \epsilon I_d)^{-1} s_i^p - 0.25(s_i^p)^T (H_i^{k,(0)} + \epsilon I_d)^{-1} s_i^p}
\]

\[
= 1.
\]

Obviously, with \(H_i^{k,(0)} > 0\), we have \(\tilde{\theta}_i^p > 0\). Since \(\theta_i^p = \min \left\{ \tilde{\theta}_i^p, \frac{L_{\|s_i^p\|}}{\|s_i^p\|} \right\}\), with \(0 < \tilde{\theta}_i^p \leq 1\), we have \(0 < \theta_i^p \leq 1\).

Moreover, substituting the definitions of \(\tilde{y}_i^p\) in (29) and \(\theta_i^p\)
in (30), we compute
\[
(s^p)^T \hat{y}^p
= \left( (s^p)^T \left( 0 \right) + (1 - \theta^p)(H^{k,(0)} + \epsilon I)^{-1} s^p \right)
= \theta^p \left[ (s^p)^T \hat{y}^p - (s^p)^T (H^{k,(0)} + \epsilon I)^{-1} s^p \right]
+ (s^p)^T (H^{k,(0)} + \epsilon I)^{-1} s^p
= \begin{cases} 
0.25(s^p)^T (H^{k,(0)} + \epsilon I)^{-1} s^p, \\
(s^p)^T \hat{y}^p, \text{ otherwise},
\end{cases}
\]
which implies
\[
(s^p)^T \hat{y}^p \geq 0.25(s^p)^T (H^{k,(0)} + \epsilon I)^{-1} s^p.
\]
By the initialization $H^{k,(0)}$ in (33), we have $(s^p)^T \hat{y}^p > 0$. To show that $H^{k+1}$ is positive definite, for the second inequality (34) and any nonzero $z \in \mathbb{R}^d$, we have
\[
= z^T (I_d - \frac{s^p \hat{y}^T}{s^T \hat{y}}) H (I_d - \frac{\hat{y} s_T}{s^T \hat{y}}) z + \left( \frac{s^T \hat{y}}{s^T \hat{y}} \right)^2 \frac{1}{s^T \hat{y}} \nabla^2 f (x_k)
\]
where we omit all the time indexes at the right-hand side of (34) for simplicity. With $s^T \hat{y} > 0$, we know that $H^{k,(t)}$ is positive definite and $\lambda_{\text{min}}(H^{k,(t)}) > 0$, which completes the proof.

Based on Lemma 3, the following theorem further gives the specific lower and upper bounds for the eigenvalues of $H^k$ generated by Algorithm 2.

**Theorem 2.** Consider the damped limited-memory BFGS method in Algorithm 2. We have
\[
M_1 I_d \preceq H^k \preceq M_2 I_d,
\]
where $M_1 = \left( \frac{1}{\beta} + \frac{M \omega^2}{4 + (\omega + 2)} \right)^{-1}$, $M_2 = (1 + \omega)^{2M} \left( B + \frac{1}{L(\omega + 2)} \right)$ and $\omega \triangleq 4(\omega + B + \epsilon)$.

**Proof.** Since the analysis holds for any node, we omit the node index $i$ in the proof. Note that the inequalities (18), (20), (22) and (23) for the proposed DFP method also hold for the proposed BFGS method, as long as we replace $\hat{s}$ with $s$. Thus, we directly use these inequalities by replacing $\hat{s}$ with $s$ and omit the proof.

First, we establish the upper bound. According to the update (34), we have
\[
\left\| H^{k,(t+1)} \right\|_2 \leq \left\| H^{k,(t)} \right\|_2 \cdot \left\| I - (s^p \hat{y}^p)^T (s^p \hat{y}^p)^T \right\|_2 + \left\| (s^p \hat{y}^p)^T \right\|_2
\leq (1 + \omega)^2 \left\| H^{k,(t)} \right\|_2 + 4(\omega + B + \epsilon),
\]
where we use (22) and (18) in the last inequality. Following the standard argument for recurrence on (36), we have
\[
\left\| H^{k,(\omega t)} \right\|_2 \leq (1 + \omega)^{2M} \left( \left\| H^{k,(0)} \right\|_2 + \frac{4(\omega + B + \epsilon)}{(1 + \omega)^2 - 1} \right)
\leq (1 + \omega)^{2M} \left( B + \frac{1}{L(\omega + 2)} \right),
\]
where where to derive the last inequality we use
\[
\frac{4(\omega + B + \epsilon)}{(1 + \omega)^2 - 1} = \frac{1}{L + \frac{1}{\beta + \epsilon}} < \frac{1}{L(\omega + 2)}.
\]

Thus, we obtain the upper bound given by $M_2 = (1 + \omega)^{2M} \left( B + \frac{1}{L(\omega + 2)} \right)$.

Next, we establish the lower bound. Using the Sherman-Morrison-Woodbury formula on (34), we get
\[
\left( H^{k,(t+1)} \right)^{-1} \leq \left( H^{k,(t)} \right)^{-1} + \frac{(\hat{y}^p)^T}{(s^p)^T \hat{y}^p} - \frac{(s^p)^T \hat{y}^p}{(s^p)^T (s^p)^T (H^{k,(t)})^{-1}} - \frac{(s^p)^T \hat{y}^p}{(s^p)^T \hat{y}^p},
\]
which implies
\[
\left\| \left( H^{k,(t+1)} \right)^{-1} \right\|_2 \leq \left\| \left( H^{k,(t)} \right)^{-1} \right\|_2 + \left\| \frac{(\hat{y}^p)^T}{(s^p)^T \hat{y}^p} \right\|_2
\leq \left\| \left( H^{k,(t)} \right)^{-1} \right\|_2 + \frac{\omega^2}{4(\omega + B + \epsilon)},
\]
where we use (22) in the last inequality. Following the standard argument for recurrence on (39), we have
\[
\lambda_{\text{min}} \left( H^{k,(\omega t)} \right) \geq \left( \frac{1}{\beta} + \frac{M \omega^2}{4(\omega + B + \epsilon)} \right)^{-1}.
\]
Therefore, we obtain the lower bound given by $M_1 = \left( \frac{1}{\beta} + \frac{M \omega^2}{4(\omega + B + \epsilon)} \right)^{-1}$ and complete the proof.

**Remark 2.** Regarding the theoretical results given by Theorems 7 and 2, we have the following comments.

1) The Hessian inverse approximations $H^k$ constructed by the proposed DFP and BFGS methods satisfy Assumption 4 and thus fit into the general framework in Part I for exact linear convergence.

2) With a large memory size $M$, the eigenvalues of $H^k$ have a wide range and thus $H^k$ may be almost singular. One conjecture is that noise caused by randomness and disagreement accumulates more with larger $M$. On the other hand, observe from the updates of the proposed quasi-Newton methods that a too small memory size $M$ may lead to insufficient second-order curvature information. Thus, we recommend to use a moderate memory size $M$, which leads to low computation and memory costs without sacrificing the performance.

3) For the proposed DFP method, the regularization term $\rho I_d$ in (15) lifts the lower bound $M_1$ by $\rho$ and lifts the upper bound $M_2$ by $M \rho$. It is the limited-memory technique that prevents the regularization term $\rho I_d$ from accumulating to infinity. The analysis of Theorem 7 also holds for $\rho = 0$. However, we observe from the numerical experiments that a suitably tuned $\rho > 0$ can improve the performance.

4) The analysis in Theorems 2 and 8 also hold for $\epsilon = 0$. However, we observe from the numerical experiments
that a suitably tuned $\epsilon > 0$ can improve the performance, especially for the proposed BFGS method.

IV. NUMERICAL EXPERIMENTS

In this section, we embed the proposed DFP method in Algorithm 1 and the proposed BFGS method in Algorithm 2 into the general framework of Part I and evaluate their performance. We use the two resultant algorithms to solve a least-squares problem with synthetic data in Section IV-A and a logistic regression problem with real data in Section IV-B. We randomly generate a connected and undirected network with $n$ nodes and $\binom{n(n-1)}{2}$ edges, where $\rho \in (0, 1]$ is the connectivity ratio. The performance metric is the relative error defined as

$$\text{relative error} = \frac{\|x^b - x^*\|^2}{n \|x^0 - x^*\|^2},$$

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

A. Effects of Condition Number

We consider a least-squares problem defined as

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

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 aggregated variables $A = [A_1; \cdots; A_n] \in \mathbb{R}^{nm \times d}$ and $b = [b_1; \cdots; b_n] \in \mathbb{R}^{nm}$ by vertically stacking the local data. We define the condition number of the problem as

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

To show the effects of the condition number, we generate two groups of data with $\kappa_{LS} = 10$ and $\kappa_{LS} = 2000$ as follows. For $\kappa_{LS} = 10$, we fix $\lambda_{\min}(A^T A) = 0.1$ and $\lambda_{\max}(A^T A) = 1$. For $\kappa_{LS} = 2000$, we fix $\lambda_{\min}(A^T A) = 0.001$ and $\lambda_{\max}(A^T A) = 2$. The other $(d-2)$ eigenvalues are randomly generated within the interval $[\lambda_{\min}(A^T A), \lambda_{\max}(A^T A)]$. Figs. 1 and 2 record the results for $\kappa_{LS} = 10$ and $\kappa_{LS} = 2000$, respectively. The parameters are set as follows. We set $n = 20$ and the connectivity ratio is $\rho = 0.5$. We set $B = 10^4$ for the two proposed quasi-Newton methods. When $\kappa_{LS} = 10$ (2000), for the proposed DFP method, we set $\alpha = 0.6 (0.06), \rho = 10^{-5} (10^{-3}), \epsilon = 3 (5), \beta = 0.04 (0.01), L = 10 (10), \text{memory size } M = 20 (20), \text{the batch size } b_i = 10 (15).$ For the proposed BFGS method, we set $\alpha = 0.6 (0.6), \epsilon = 3 (37), \beta = 0.04 (0.01), L = 10 (10), \text{the memory size } M = 20 (50), \text{the batch size } b_i = 10 (15), \text{respectively.}$ We compare with the existing first-order methods, DSA [17], GT-SVRG and GT-SAGA [18], and Acc-VR-DIGing [19]. For DSA, we set the stepsize $\alpha = 0.9 (0.9)$ and the batch size $b_i = 1 (1)$. For GT-SVRG, we set the stepsize $\alpha = 0.9 (0.9)$ and the batch size $b_i = 1 (1)$. For GT-SAGA, we set the stepsize $\alpha = 0.95 (0.95)$ and the batch size $b_i = 1 (1)$. For Acc-VR-DIGing, we set the stepsize $\alpha = 0.9 (0.9)$ and the batch size $b_i = 1 (1)$.

B. Comparison with First-order Algorithms: Real Datasets

The ensuing numerical experiments evaluate on the real datasets. We use the proposed quasi-Newton methods to solve a logistic regression problem in the form of

$$\min_{x \in \mathbb{R}^d} \frac{1}{2} \|x\|^2 + \frac{1}{n} \sum_{i=1}^n \frac{1}{m_i} \sum_{j=1}^{m_i} \ln (1 + \exp(- (\omega_{ij}^T x) p_{ij}))),$$

where node $i$ privately owns $m_i$ training samples $(\omega_{il}, p_{ij}) \in \mathbb{R}^d \times \{-1, +1\}; l = 1, \ldots, m_i$. We use five real datasets whose attributes are summarized in Table III. We normalize each sample such that $|\omega_{il}| = 1, \forall i, l.$ Note that another way is to normalize each feature, which yields better condition number but is nontrivial to implement in the decentralized setting. A

---

1https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/
TABLE I
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              |
| icnn1   | 91700                             | 22               |

The regularization term $\frac{\lambda}{2} \| x \|_2^2$ with $\lambda > 0$ is used to avoid overfitting. We set $n = 20$, $\lambda = 0.001$ and $B = 10^4$ throughout the following numerical experiments. The training samples are randomly and evenly distributed over all the $n$ nodes.

We compare the proposed two decentralized stochastic quasi-Newton methods with four decentralized stochastic first-order algorithms, DSA in [17], GT-SVRG and GT-SAGA in [18], and Acc-VR-DIGing in [19] on four real datasets. Different to the numerical experiments on the synthetic data where the batch sizes are set as 1, now we use larger batch sizes to boost the performance of the four first-order methods. Figs. 3–6 depict the results on covtype, cod-rna, a6a and a9a, respectively. The parameters are set as follows. We set $n = 20$ and the connectivity ratio $\rho = 0.5$. When the dataset is covtype (cod-rna, a6a, a9a), for the proposed DFP method, we set $\alpha = 0.32$ (0.3, 0.38, 0.38), $\rho = 0.01$ (0.0002, 0.01, 0.001), $\epsilon = 0.02$ (0.03, 0.005, 0.1), $\beta = 0.002$ (0.002, 0.015, 0.5), $L = 50$ (50, 20, 50), the memory size $M = 3$ (20, 40, 50), the batch size ratio $b_i / m_i = 10\%$ (8\%, 10\%, 6\%), respectively. For the proposed BFGS method, we set $\alpha = 0.37$ (0.35, 0.38, 0.35), $\epsilon = 0.001$ (100, 30, 30), $\beta = 0.002$ (0.002, 1.2, 0.5), $L = 50$ (50, 20, 20), the memory size $M = 3$ (40, 50, 50), the batch size ratio $b_i / m_i = 10\%$ (10\%, 10\%, 10\%), respectively. For GT-SVRG, we set the stepsize $\alpha = 0.002$ (0.01, 0.009, 0.004) and the batch size $b_i = 5$ (2, 1, 2). For DSA, we set the stepsize $\alpha = 0.001$ (0.03, 0.009, 0.008) and the batch size $b_i = 10$ (10, 1, 2). For GT-SAGA, we set the stepsize $\alpha = 0.002$ (0.009, 0.009, 0.0035) and the batch size $b_i = 5$ (2, 1, 1). For Acc-VR-DIGing, we set the stepsize $\alpha = 0.002$ (0.03, 0.04, 0.015), the two parameters $\theta_1 = 0.9$ (0.07, 0.1, 0.1) and $\theta_2 = 0.01$ (0.05, 0.1, 0.1), while the batchsize $b_i = 5$ (10, 5, 5).

As Figs. 3–6 show, the proposed two decentralized stochastic quasi-Newton methods outperform the real dataset icnn1. GT-SVRG, GT-SAGA and Acc-VR-DIGing in all the four datasets, demonstrating the gains of curvature information from the constructed Hessian inverse approximations. Generally, the proposed DFP method is better than the proposed BFGS method in all the four datasets, but the BFGS method has lower memory requirement and lower computation cost.

C. Effects of Batch Size

Here, we numerically show the effects of batch size on the performance of proposed DFP and BFGS methods for solving the logistic regression problem using the real dataset icnn1. In Figs. 7 and 8, we evaluate the effects of different batch size ratios ($b_i / m_i = 2\%, 4\%, 6\%, 8\%$ and $10\%$), on the performance of the proposed DFP and BFGS methods, respectively. The parameters are set as follows. For the proposed DFP (BFGS)
method, we set $\alpha = 0.32 (0.31)$, $\rho = 0.005 (0)$, $\epsilon = 0.005 (0.005)$, $\beta = 0.1 (0.1)$, and the memory size $M = 50 (50)$. The other settings are the same as those used in Fig. 3.

From Figs. 7 and 9, we observe that too larger or smaller batch sizes lead to more epochs, because a smaller batch size causes higher stochastic gradient noise, while a larger batch size calls for more gradient evaluations per iteration. For both of the proposed DFP and BFGS methods, a batch size ratio of $b_i/m_i = 6\%$ gives the best performance.

**D. Effects of Memory Size**

In Figs. 9 and 10, we evaluate the effects of different memory sizes ($M = 5, 10, 20, 30, 40, $ and 50) on the performance of the proposed DFP and BFGS methods, respectively. The problem is logistic regression and the dataset is ijcnn1. The parameters are set as follows. For the proposed DFP (BFGS) method, we set $\alpha = 0.32 (0.31)$, $\rho = 0.004 (0)$, $\epsilon = 0.005 (0.002)$, $\beta = 0.001 (0.1)$, and the batch size ratio $b_i/m_i = 6\% (6\%)$. The other settings are the same as those used in Fig. 3.

As Figs. 9 and 10 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 computation costs.

**E. Effects of Topology**

In Figs. 11 and 12, we evaluate the effects of five different topologies (cycle, star, random graphs with connectivity ratios $\rho = 0.2, 0.3, 0.5$) on the performance of the proposed DFP and BFGS methods, respectively. The second largest singular values $\sigma$ of $W$ i.e., $\sigma = ||W - \frac{1}{n}1_n1_n^T||_2$, 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 proposed DFP method, we set $\alpha = 0.035 (0.02, 0.2, 0.25, 0.32)$, $\rho = 0.003 (0.001, 0.001, 0.001, 0.005)$, $\epsilon = 0.005 (0.005, 0.005, 0.005, 0.005)$, $\beta = 0.1 (0.1, 0.1, 0.1, 0.1)$, the memory size $M = 50 (50, 50, 50, 50)$, and the batch size ratio $b_i/m_i = 6\% (6\%, 6\%, 6\%, 6\%)$. For the proposed BFGS method, we set $\alpha = 0.06 (0.07, 0.2, 0.3, 0.31)$, $\epsilon = 0.005 (0.005, 0.002, 0.002, 0.002)$, $\beta = 0.1 (0.1, 0.1, 0.1, 0.1)$, $M = 50 (50, 50, 50, 50)$, and the batch size ratio $b_i/m_i = 11\% (10\%, 6\%, 6\%, 6\%)$. The other settings are the same as used in Fig. 3.

From Figs. 11 and 12, we observe that the proposed two decentralized quasi-Newton methods converge linearly on different graphs. For both methods, graphs with smaller $\sigma$ give faster convergence rates, which corroborate with the theoretical results in Part I.

**V. CONCLUSIONS**

In Part II of this work, we propose two fully decentralized quasi-Newton methods, damp regularized limited-memory DFP and damp limited-memory BFGS, to locally construct the Hessian inverse approximations. We use the damping and limited-memory techniques to ensure that the constructed Hessian inverse approximations are positive definite with bounded eigenvalues. For the DFP-based method,
we add a regularization term to improve the performance. For the BFGS-based method, we use a two-loop recursion to reduce the memory and computation costs. We prove that quasi-Newton approximations satisfy the assumption in Part I for the exact linear convergence. Numerical experiments in Part II demonstrate that the proposed quasi-Newton methods are much faster than the existing decentralized stochastic first-order methods.

REFERENCES

[1] M. Zhu and S. Martínez, “Discrete-time dynamic average consensus,” *Automatica*, vol. 46, no. 2, pp. 322–329, 2010.
[2] J. Nocedal and S. Wright, *Numerical optimization*. Springer Science & Business Media, 2006.
[3] 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.
[4] 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.
[5] 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.
[6] 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.
[7] 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.
[8] ——, “RES: Regularized stochastic BFGS algorithm,” *IEEE Transactions on Signal Processing*, vol. 62, no. 23, pp. 6089–6104, 2014.
[9] A. Lucchi, B. McWilliams, and T. Hofmann, “A variance reduced stochastic Newton method,” *arXiv preprint arXiv:1503.08316*, 2015.
[10] P. Moritz, R. Nishihara, and M. Jordan, “A linearly-convergent stochastic L-BFGS algorithm,” in *Artificial Intelligence and Statistics*, 2016, pp. 249–258.
[11] M. Eisen, A. Mokhtari, and A. Ribeiro, “Decentralized quasi-Newton methods,” *IEEE Transactions on Signal Processing*, vol. 65, no. 10, pp. 2613–2628, 2017.
[12] 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.
[13] 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.
[14] ——, “A primal-dual quasi-Newton method for exact consensus optimization,” *IEEE Transactions on Signal Processing*, vol. 67, no. 23, pp. 5983–5997, 2019.
[15] 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.
[16] 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.
[17] 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.
[18] 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.
[19] H. Li, Z. Lin, and Y. Fang, “Optimal accelerated variance reduced EXTRA and DIGing for strongly convex and smooth decentralized optimization,” *arXiv preprint arXiv:2009.04373*, 2020.