Network Gradient Descent Algorithm for Decentralized Federated Learning

Shuyuan Wu, Danyang Huang & Hansheng Wang

Taylor & Francis
Taylor & Francis Group

Check for updates

Network Gradient Descent Algorithm for Decentralized Federated Learning

Shuyuan Wu[a], Danyang Huang[b,c], and Hansheng Wang[a]

[a]Guanghua School of Management, Peking University, Beijing, China; [b]Center for Applied Statistics, Renmin University of China, Beijing, China; [c]School of Statistics, Renmin University of China, Beijing, China

ABSTRACT

We study a fully decentralized federated learning algorithm, which is a novel gradient descent algorithm executed on a communication-based network. For convenience, we refer to it as a network gradient descent (NGD) method. In the NGD method, only statistics (e.g., parameter estimates) need to be communicated, minimizing the risk of privacy. Meanwhile, different clients communicate with each other directly according to a carefully designed network structure without a central master. This greatly enhances the reliability of the entire algorithm. Those nice properties inspire us to carefully study the NGD method both theoretically and numerically. Theoretically, we start with a classical linear regression model. We find that both the learning rate and the network structure play significant roles in determining the NGD estimator's statistical efficiency. The resulting NGD estimator can be statistically as efficient as the global estimator, if the learning rate is sufficiently small and the network structure is weakly balanced, even if the data are distributed heterogeneously. Those interesting findings are then extended to general models and loss functions. Extensive numerical studies are presented to corroborate our theoretical findings. Classical deep learning models are also presented for illustration purpose.

1. Introduction

We study here a gradient descent algorithm for decentralized federated learning. Our methodology has two important elements. The first is gradient descent, which is arguably the most popularly used optimization method for complex statistical learning (e.g., deep learning). The second is federated learning, which is a novel collective learning paradigm. The objective here is to train a global model collectively by a large number of local devices (or clients). Often those devices can be naturally connected with each other through a network (e.g., a wireless communication network). Then, how to conduct a gradient descent algorithm on this network and study the resulting estimator's statistical properties become problems of great interest.

Specifically, we consider a standard statistical learning problem with a total of N observations. For each observation, there is a response of interest and a predictor with a fixed dimension. There are also parameters of interest, which need to be estimated by minimizing an appropriately defined loss function. Traditionally, the whole sample is placed on one single computer and then processed conveniently. However, in federated learning, data are often distributed across a large number of clients (e.g., mobile devices). One major concern is the privacy issue. Typically, the data generated from different clients could contain highly sensitive private information. Passing the original data from local clients to a central computer may incur a high risk

of privacy disclosure. Nevertheless, to train the global model, we need the information contained in the datasets distributed across different clients. This inspires the novel idea of federated learning to fix the privacy disclosure problem.

The original idea of federated learning was first proposed by Konečnỳ et al. (2016). They assumed that there exists a central computer, which can be connected with a large number of clients. To train the global model, the central computer collects the local parameter estimators computed on each local client after local gradient updating iteratively. Then, the aggregating parameter estimators are passed to each local client. Transporting local parameter estimators protects privacy better than directly passing the raw data from local clients to the central computer. This leads to the classical federated learning algorithm (McMahan et al. 2017), which has been extended by many researchers. For example, Smith et al. (2017) proposed a novel strategy to handle federated multi-task learning problems. Cheu et al. (2019) proposed a distributed differentially private algorithm for stricter privacy protection. Hashimoto et al. (2018) developed a robust optimization method to handle worst-case risk.

These federated learning algorithms require a central machine, which is responsible for communicating with every client. Such an type of architecture is easy to implement. However, it suffers from several serious limitations. First, it is not the best choice for privacy protection, because the central machine itself could be vulnerable. In fact, if the central machine

is conquered, the attacker is given the chance to communicate with every client (Bellet et al. 2018). Second, this centralized network structure is extremely fragile for stable operation. In other words, the central machine has an overly important role in the network. When the central machine stops working, the entire learning task stops. Third, centralized network structure has a high requirement for network bandwidth. This is because the central machine has to communicate with a huge number of clients (Li et al. 2021).

To fix these problems, a number of researchers advocate the idea of fully decentralized federated learning (Colin et al. 2016; Vanhaesebrouck, Bellet, and Tommasi 2017; Tang et al. 2018). The key feature is that there is no central computer involved for model training and communication. Although one central computer may be needed to set up the entire learning task, it is not responsible for actual computation. Consequently, all the computation-related communications should occur only between individual clients. This leads to a communication-based network structure. Each node represents a client and each edge represents the communication relationship between the clients. Subsequently, local gradient steps are conducted for each client. The corresponding local parameters are updated by aggregating the information from their one-hop neighbors (i.e., directly connected neighbors) in the network. This leads to the Gossip SGD method of Blot et al. (2016) and Decentralized (Stochastic-)GD method of Yuan, Ling, and Yin (2016), Lian et al. (2017), and Lian et al. (2018). Nedic, Olshevsky, and Shi (2017) and Savazzi, Nicoli, and Rampa (2020) further proposed new algorithms based on the exchange of both parameters and gradients in each iteration for better convergence speed. In addition, Lalitha et al. (2018, 2019) proposed a Bayesian-type approach to estimate the interested parameter for additive models in the aperiodic strongly connected network. Richards and Rebeschini (2019) and Richards, Rebeschini, and Rosasco (2020) analyzed the statistical rates of Decentralized GD based on nonparametric regression with a squared loss function.

The above literature about decentralized federated learning could be summarized from different aspects according to the: (a) assumptions imposed on the network structure; (b) assumptions imposed on the data distribution across different clients; and (c) types of theoretical convergence; see Table 1 for the details. By Table 1, we find that many existing literature imposed stringent assumptions on the network structure, for example, doubly stochastic matrix (Yuan, Ling, and Yin 2016; Lian et al. 2017; Tang et al. 2018; Richards and Rebeschini 2019; Richards, Rebeschini, and Rosasco 2020). In addition, some literature imposed restrictive assumptions on the data distribution pattern, for example, homogeneous distribution (Richards and Rebeschini 2019; Richards, Rebeschini, and Rosasco 2020). Moreover, most existing literature studied the algorithm convergence either theoretically or numerically. It seems that little has been done about the statistical convergence theory.

In our work, we develop a novel methodology for decentralized federated learning. The new methodology allows the data distribution pattern across different clients to be either homogeneous or heterogeneous. Moreover, the proposed method only requires the network structure to be weakly balanced. This is an assumption weaker than the double stochastic matrix assumption, which has been popularly used in the past literature (Yuan,

Ling, and Yin 2016; Tang et al. 2018; Richards and Rebeschini 2019; Richards, Rebeschini, and Rosasco 2020). Meanwhile, both the numerical convergence and statistical efficiency are studied. Specifically, our methodology is based on the classical structure of the decentralized federated learning (Yuan, Ling, and Yin 2016; Blot et al. 2016). For convenience, we call the method network gradient descent (NGD) method. First, we assume that different clients are connected with each other by an appropriately defined network. The corresponding network structure can be mathematically described by a network adjacency matrix. Second, we conduct a gradient descent algorithm on each local client. To calculate the local gradients, the target client needs to update the current estimator to be the next one by the method of gradient descent. The gradient is computed based on the current estimator, which is taken as the averaged estimators generated by the network neighbors. This leads to an interesting gradient descent algorithm, which can be executed on a fully connected network.

We theoretically prove that the NGD algorithm numerically converges to a limiting estimator under appropriate conditions for the learning rate. The resulting estimator is referred to as the NGD estimator and its asymptotic properties are investigated. We show theoretically that the statistical efficiency of the NGD estimator is jointly affected by three factors. They are, respectively, the learning rate, the network structure, and the data distribution across different clients. Our theory suggests that a statistically efficient estimator can be obtained even if the data are heterogeneously distributed across different clients, as long as the learning rate is sufficiently small and the network structure is weakly balanced. This seems to us a very interesting or even surprising theoretical finding (Lian et al. 2017, 2018). However, if the data distribution across different clients are well homogeneous, the technical requirement for both the learning rate and network structure can be further relaxed. Extensive numerical experiments are conducted to demonstrate our theoretical findings.

The rest of the article is organized as follows. Section 2 develops the NGD algorithm and presents its numerical and asymptotic statistical properties. The numerical studies are presented in Section 3, including the simulation experiments and a real data example of image data analysis by deep learning. Finally, Section 4 concludes with a brief discussion. All technical details are relegated to the Appendix, supplementary materials.

2. Methodology

2.1. Network Gradient Descent

We first introduce the model and notations. Let $X_i \in \mathbb{R}^p, Y_i \in \mathbb{R}^1$ be the information collected from the ith subject with $1 \leq i \leq N$. Here Y_i is the response of interest and X_i is the associated predictor with finite moments. For simplicity, we start with the following classical linear regression model

$$Y_i = X_i^\top \theta_0 + \varepsilon_i,$$

where ε_i is the independently and identically distributed residual with mean 0 and variance σ^2. Here θ_0 is the regression coefficient parameter to be estimated. To estimate θ_0, we need to optimize the following global least-squares loss function

Table 1. The summarization of existing literature from three aspects according to the assumptions on the network structure, data distribution pattern, and types of theoretical convergence.

Existing Literature	Network Structure Assumption	Data Distribution Assumption	Theoretical Convergence Type
Yuan, Ling, and Yin (2016)	DSM	HOMO/HETE	NCR
Tang et al. (2018)			
Nedic, Olshevsky, and Shi (2017)	DSM/SC	HOMO/HETE	NCR
Lian et al. (2017, 2018)	DSM	BV	NCR
Richards and Rebeschini (2019)	DSM	HOMO	NCR&SCR
Richards, Rebeschini, and Rosasco (2020)			
Vanhaesebrouck, Bellet, and Tommasi (2017)	s-SM	HOMO/HETE	NCR
Lalitha et al. (2018, 2019)	aperiodic-SC	HOMO/HETE	SEB
Blot et al. (2016)	Algorithm Driven		
Savazzi, Nicoli, and Rampa (2020)			
The Proposed NGD Method	**WBM**	**HOMO/HETE**	**NCR&SCR**

NOTE: The definition of the abbreviations are given as follows. DSM: doubly stochastic matrix; SC: strongly connected; s-SM: symmetric similarity matrix (i.e., the matrix value reflects distribution similarity); WBM: weakly balanced matrix; BV: bounded variance (i.e., the variance of the stochastic gradient is bounded across different clients). HOMO: homogeneous (i.e., independent and identically distributed across different clients); HETE: heterogeneous; SEB: statistical error bound; NCR/SCR: numerical/statistical convergence rate. The assumptions of the proposed method are marked in bold.

as $\mathcal{L}_N(\theta) = N^{-1} \sum_{i=1}^{N} (Y_i - X_i^\top \theta)^2$. Then, it leads to the corresponding ordinary least squares (OLS) estimator $\widehat{\theta}_{\text{ols}} = \arg\min_\theta \mathcal{L}_N(\theta) = \widehat{\Sigma}_{xx}^{-1} \widehat{\Sigma}_{xy}$, where $\widehat{\Sigma}_{xx} = N^{-1} \sum_i X_i X_i^\top$ and $\widehat{\Sigma}_{xy} = N^{-1} \sum_i X_i Y_i$. It is remarkable that based on the least squares loss function, we are able to obtain theoretical results with deep insights. Those results are then extended to general loss functions in Section 2.5.

By the classical linear regression theory, we know that $\widehat{\theta}_{\text{ols}}$ is asymptotic normal with $\sqrt{N}(\widehat{\theta}_{\text{ols}} - \theta_0) \to_d N(0, \sigma^2 \Sigma_{xx}^{-1})$ with $\Sigma_{xx} = E(X_i X_i^\top)$; see Rao et al. (1973) and Shao (2003). Consequently, as long as $\widehat{\Sigma}_{xx}^{-1}$ and $\widehat{\Sigma}_{xy}$ can be computed directly, the global OLS estimator can be easily obtained. However, such a straightforward method is often practically infeasible in federated learning. This is mainly because that data are distributed across different clients. To compute $\widehat{\Sigma}_{xx}^{-1} \widehat{\Sigma}_{xy}$ directly, one must aggregate all the local data from different clients to the central computer. As mentioned in the introduction, this leads to high risk of privacy disclosure and is practically undesirable. As a result, various federated learning algorithms are practically attractive.

Specifically, we consider a fully decentralized federated learning framework with the whole data distributed across M clients, which are also nodes in the network. The clients are indexed by $\mathcal{M} = \{1, 2, \ldots, M\}$. Those clients are connected by a communication network, whose adjacency matrix is given by $A = (a_{m_1 m_2}) \in \mathbb{R}^{M \times M}$ for $1 \leq m_1, m_2 \leq M$. Here, $a_{m_1 m_2} = 1$ if client m_1 can receive information from m_2, and $a_{m_1 m_2} = 0$ otherwise. For completeness, we assume $a_{mm} = 0$ for $1 \leq m \leq M$. We define the weighting matrix $W = (w_{m_1 m_2}) \in \mathbb{R}^{M \times M}$ with $w_{m_1 m_2} = a_{m_1 m_2}/d_{m_1}$, where $d_{m_1} = \sum_{m_2} a_{m_1 m_2}$ is the in-degree for client (i.e., node) m_1 and we assume $d_{m_1} \geq 1$. Define the whole dataset $\mathcal{S}_F = \{1, 2, \ldots, N\} = \bigcup_{m=1}^{M} \mathcal{S}_{(m)}$, where $\mathcal{S}_{(m)}$ is the index set of the sample distributed to the mth client. Then, the global loss function could be rewritten as

$$\mathcal{L}_N(\theta) = N^{-1} \sum_{i=1}^{N} \left(Y_i - X_i^\top \theta\right)^2 = M^{-1} \sum_{m=1}^{M} \mathcal{L}_{(m)}(\theta),$$

where $\mathcal{L}_{(m)}(\theta) = n^{-1} \sum_{i \in \mathcal{S}_{(m)}} \left(Y_i - X_i^\top \theta\right)^2$ is the loss function evaluated for the mth client. For simplicity, we assume that $|\mathcal{S}_{(m)}| = n = N/M$.

We then introduce the network gradient algorithm as follows. Specifically, we follow the idea of Yuan, Ling, and Yin (2016), Colin et al. (2016), and Lian et al. (2017), and execute the following NGD algorithm:

$$\widehat{\theta}^{(t+1,m)} = \widetilde{\theta}^{(t,m)} - \alpha \dot{\mathcal{L}}_{(m)}\left(\widetilde{\theta}^{(t,m)}\right),$$

where $\widetilde{\theta}^{(t,m)}$ is the neighborhood averaged estimator obtained in the tth iteration for the mth client. Specifically, we have $\widetilde{\theta}^{(t,m)} = \sum_{k=1}^{M} w_{mk} \widehat{\theta}^{(t,k)}$. The optimization process is a decentralized federated learning process. Write $\widehat{\Sigma}_{xx}^{(m)} = \sum_{i \in \mathcal{S}_{(m)}} X_i X_i^\top / n$ and $\widehat{\Sigma}_{xy}^{(m)} = \sum_{i \in \mathcal{S}_{(m)}} X_i Y_i / n$. Then, we have,

$$
\begin{aligned}
\widehat{\theta}^{(t+1,m)} &= \widetilde{\theta}^{(t,m)} - \alpha \dot{\mathcal{L}}\left(\widetilde{\theta}^{(t,m)}\right) \\
&= \widetilde{\theta}^{(t,m)} - \alpha \left(\widehat{\Sigma}_{xx}^{(m)} \widetilde{\theta}^{(t,m)} - \widehat{\Sigma}_{xy}^{(m)}\right) \\
&= \left(I_p - \alpha \widehat{\Sigma}_{xx}^{(m)}\right) \widetilde{\theta}^{(t,m)} + \alpha \widehat{\Sigma}_{xy}^{(m)} \\
&= \left(I_p - \alpha \widehat{\Sigma}_{xx}^{(m)}\right) \left(\sum_{k=1}^{M} w_{mk} \widehat{\theta}^{(t,k)}\right) + \alpha \widehat{\Sigma}_{xy}^{(m)}, \quad (2.1)
\end{aligned}
$$

where $I_p \in \mathbb{R}^{p \times p}$ is a p-dimensional identity matrix.

The estimator obtained by the NGD algorithm on each client could be reorganized into a vector form. Then, we define $\widehat{\theta}^{*(t)} = \left(\widehat{\theta}^{(t,1)\top}, \ldots, \widehat{\theta}^{(t,M)\top}\right)^\top \in \mathbb{R}^q$, $\widehat{\Sigma}_{xy}^* = \left(\widehat{\Sigma}_{xy}^{(1)\top}, \ldots, \widehat{\Sigma}_{xy}^{(M)\top}\right)^\top \in \mathbb{R}^q$, and $\Delta^* = \text{diag}\left(\Delta^{(1)}, \Delta^{(2)}, \ldots, \Delta^{(M)}\right) \in \mathbb{R}^{q \times q}$ with $\Delta^{(m)} = I_p - \alpha \widehat{\Sigma}_{xx}^{(m)}$ and $q = Mp$. Then, (2.1) can be rewritten into a matrix form as

$$\widehat{\theta}^{*(t+1)} = \Delta^*(W \otimes I_p)\widehat{\theta}^{*(t)} + \alpha \widehat{\Sigma}_{xy}^*, \quad (2.2)$$

where $\otimes$ denotes the Kronecker product. By Equation (2.2), we find that the NGD estimation process can be represented by a linear dynamic system. This representation immediately leads to two interesting questions. First, does a stable solution exist for this linear dynamic system? Second, if there is a stable solution, does the aforementioned NGD algorithm converge to it? These problems are considered in the next section.

2.2. Numerical Convergence

To study the numerical properties of the NGD algorithm, we temporarily assume that a stable solution indeed exists. We then study its analytical expression. This leads to sufficient conditions about its existence. Before we discuss the stable solution, we first introduce the notations. Define $\widehat{\theta}^*$ as the stable solution of Equation (2.2). Then, we have

$$\widehat{\theta}^* = \Delta^*(W \otimes I_p)\widehat{\theta}^* + \alpha \widehat{\Sigma}_{xy}^*. \qquad (2.3)$$

Denote $\widehat{\Omega} = I_p - \Delta^*(W \otimes I_p) \in \mathbb{R}^{q \times q}$. Its analytical form is given by

$$\widehat{\Omega} = \begin{pmatrix} I_p - w_{11}\Delta^{(1)}, & -w_{12}\Delta^{(1)}, & \ldots, & -w_{1M}\Delta^{(1)} \\ -w_{21}\Delta^{(2)}, & I_p - w_{22}\Delta^{(2)}, & \ldots, & -w_{2M}\Delta^{(2)} \\ -w_{31}\Delta^{(3)}, & -w_{32}\Delta^{(3)}, & \ldots, & -w_{3M}\Delta^{(3)} \\ \ldots, & \ldots, & \ldots, & \ldots \\ -w_{M1}\Delta^{(M)}, & -w_{M2}\Delta^{(M)}, & \ldots, & I_p - w_{MM}\Delta^{(M)} \end{pmatrix}.$$

If $\widehat{\Omega}$ is invertible, the stable solution is uniquely determined, and it is given by $\widehat{\theta}^* = \alpha \widehat{\Omega}^{-1} \widehat{\Sigma}_{xy}^*$. Accordingly, the invertibility of $\widehat{\Omega}$ determines the existence of the stable solution $\widehat{\theta}^*$. If $\widehat{\theta}^*$ can indeed be obtained by the NGD algorithm (2.2), we refer to it as the NGD estimator.

Next, we discuss the convergence conditions of the NGD algorithm. To this end, we take the difference between Equations (2.2) and (2.3). This leads to the following equation

$$\widehat{\theta}^{*(t+1)} - \widehat{\theta}^* = \Delta^*(W \otimes I_p)\left(\widehat{\theta}^{*(t)} - \widehat{\theta}^*\right). \qquad (2.4)$$

We refer to $\Delta^*(W \otimes I_p)$ as a contraction operator. As (2.4) shows, the contraction operator plays an important role in determining the numerical convergence of the NGD algorithm. Specifically, we should have $\widehat{\theta}^{*(t+1)} - \widehat{\theta}^* = \left\{\Delta^*(W \otimes I_p)\right\}^{t+1}(\widehat{\theta}^{*(0)} - \widehat{\theta}^*)$ by (2.4), where $\widehat{\theta}^{*(0)}$ is the initial value specified for every client. Then, for the numerical convergence of the NGD algorithm, we should require $\lambda_{\max}\left(\Delta^*\{W \otimes I_p\}\right) < 1$, where $\lambda_{\max}(B)$ is the largest absolute eigenvalue of an arbitrary matrix B. Notice that the eigenvalue of B could be a complex number, if B is asymmetric. Moreover, under the condition $\lambda_{\max}\left(\Delta^*\{W \otimes I_p\}\right) < 1$, $\widehat{\Omega}$ is also invertible. Thus, the stable solution $\widehat{\theta}^*$ (now equal to the NGD estimator) does exist. Finally, define a sequence $\{x^t\}$ as converging linearly if there are some $t_0 \in \mathbb{N}$ such that $\|x^t\| \leq c_0 \nu^t$ for all $t > t_0$, with some constants $\nu \in [0, 1)$ and $c_0 > 0$. We then obtain the following theorem to assure the numerical linear convergence of the NGD algorithm.

Theorem 1 (Numerical Convergence for Linear Regression). Assume that $n > p$ and $0 < \alpha < 2\min_{1 \leq m \leq M}\left\{\lambda_{\max}^{-1}(\widehat{\Sigma}_{xx}^{(m)})\right\}$. Then, the stable solution $\widehat{\theta}^*$ exists, and $\widehat{\theta}^{*(t)} \to \widehat{\theta}^*$ linearly.

The detailed proof of Theorem 1 is given in Appendix B.1, supplementary materials. Based on Theorem 1, we draw the following two important conclusions. First, as long as the learning rate α is sufficiently small, and the sample size n of each client is larger than the dimension of parameter p, the stable solution $\widehat{\theta}^*$ should exist, and the NGD algorithm should converge to it linearly. This leads to the NGD estimator. Nevertheless, we should note that there exists a situation, where $\widehat{\Omega}$ is invertible but

$\lambda_{\max}\left(\Delta^*\{W \otimes I_p\}\right) > 1$. In this case, the stable solution exists, but the NGD algorithm might not converge to it. Thus, the NGD estimator does not exist because it cannot be computed. Second, surprisingly, we find that the numerical convergence of the NGD algorithm can be fully assured by the learning rate only. In other words, the network structure is not very important in determining the NGD algorithm's numerical convergence. However, as we show subsequently, the network structure does play a critical role in determining the NGD estimators' statistical efficiency. This is demonstrated in the next section.

2.3. The NGD Estimator versus the OLS Estimator

By the results given in Section 2.2, we know that, under appropriate regularity conditions, the NGD algorithm should numerically converge to the NGD estimator $\widehat{\theta}^*$. Then, it is of great interest to investigate its statistical efficiency. To this end, the relationship between $\widehat{\theta}^*$ and the global estimator (i.e., the OLS estimator $\widehat{\theta}_{ols}$), is investigated. Write $\widehat{\theta}_{ols}^* = I^*\widehat{\theta}_{ols}$ as the stacked global OLS estimator, where $I^* = \mathbf{1}_M \otimes I_p$, and $\mathbf{1}_M = (1, \ldots, 1)^\top \in \mathbb{R}^M$. Furthermore, define $\widehat{SE}^2(W) = M^{-1}\|W^\top \mathbf{1}_M - \mathbf{1}_M\|^2$, where $\|v\| = (v^\top v)^{1/2}$ for an arbitrary vector v. Note that the mean for the column sum of W is 1. Thus, $\widehat{SE}^2(W)$ measures the variability of the column sum of W. Intuitively, $\widehat{SE}^2(W)$ can be considered as a measure for the balance of the network structure. A small $\widehat{SE}^2(W)$ value suggests that different clients are connected with each other with approximately equal likelihood. For example, if the network structure is of a doubly stochastic structure (Yuan, Ling, and Yin 2016; Tang et al. 2018), then $\widehat{SE}^2(W) = 0$. This implies a perfectly balanced network structure. Otherwise, some clients must be overly preferred by the network, which means imbalance to some extent. In this regard, several classical network structures should be discussed in the following Section 2.4. Write $\widehat{SE}^2(\widehat{\Sigma}_{xx}) = \text{tr}\left[M^{-1}\sum_{m=1}^M (\widehat{\Sigma}_{xx}^{(m)} - \widehat{\Sigma}_{xx})^2\right]$ and $\widehat{SE}^2(\widehat{\Sigma}_{xy}) = M^{-1}\sum_{m=1}^M \|\widehat{\Sigma}_{xy}^{(m)} - \widehat{\Sigma}_{xy}\|^2$. If observations are distributed across different clients in a completely random way, we expect $\widehat{SE}^2(\widehat{\Sigma}_{xx})$ and $\widehat{SE}^2(\widehat{\Sigma}_{xy})$ to be of $O_p(1/n)$ order; Otherwise, the data distribution across different clients should be heterogeneous; see Lemma 3 in Appendix A, supplementary materials for the detailed proof. Thus, both $\widehat{SE}^2(\widehat{\Sigma}_{xx})$ and $\widehat{SE}^2(\widehat{\Sigma}_{xy})$ can be viewed as measures for data distribution randomness. Finally, define $\lambda_{\min}(B)$, $\lambda_{\min}^+(B)$ as the smallest eigenvalue and the smallest positive eigenvalue of an arbitrary matrix B, respectively. Let $\sigma_{\max}^w = \lambda_{\max}^{1/2}(W^\top W)$, $\sigma_{\min}^{I-w} = (\lambda_{\min}^+\{(I_p - W)^\top(I_p - W)\})^{1/2}$. We then obtain the following theorem.

Theorem 2 (Statistical Efficiency for Linear Regression). Assume the stable solution $\widehat{\theta}^*$ exists. Further, assume that (1) W is irreducible, (2) there exist some positive constants $0 < \kappa_1 \leq \kappa_2 < \infty$, such that $\kappa_1 \leq \lambda_{\min}(\widehat{\Sigma}_{xx}) \leq \lambda_{\max}(\widehat{\Sigma}_{xx}^{(m)}) \leq \kappa_2$ for any $1 \leq m \leq M$, (3) $\widehat{SE}(W)$ and α are sufficiently small such that $\alpha\kappa_2\sigma_{\max}^w + \widehat{SE}(W) < \kappa_1\kappa_2^{-1}\sigma_{\min}^{I-w}/(4\sigma_{\max}^w)$. Then it holds that

$$\frac{\|\widehat{\theta}^* - \widehat{\theta}_{ols}^*\|}{\sqrt{M}} \leq c_1\{\widehat{SE}(W) + \alpha\}\left[\widehat{SE}(\widehat{\Sigma}_{xx}) + \widehat{SE}(\widehat{\Sigma}_{xy})\right]$$

for some constants $c_1 > 0$ with probability tending to one.

The detailed proof of Theorem 2 is given in Appendix B.2, supplementary materials. By Theorem 2, the discrepancy between $\widehat{\theta}^*$ and $\widehat{\theta}^*_{\text{ols}}$ is upper bounded by $\{\widehat{\text{SE}}(W) + \alpha\}\{\widehat{\text{SE}}(\widehat{\Sigma}_{xx}) + \widehat{\text{SE}}(\widehat{\Sigma}_{xy})\}$. As a result, the optimal statistical efficiency can be guaranteed, as long as this term is of $o_p(1/\sqrt{N})$ order. This can be easily satisfied by: either (1) $\alpha = o_p(1/\sqrt{N})$ and $\widehat{\text{SE}}(W) = o_p(1/\sqrt{N})$; or (2) $\widehat{\text{SE}}(\widehat{\Sigma}_{xx}) = O_p(1/\sqrt{n})$, $\widehat{\text{SE}}(\widehat{\Sigma}_{xy}) = O_p(1/\sqrt{n})$ and $\alpha = o_p(1/\sqrt{M})$, $\widehat{\text{SE}}(W) = o_p(1/\sqrt{M})$. Note that (2) represents the case where data cross different clients are homogeneously, which means identically and independently distributed. In this case, M can play its role through $\widehat{\text{SE}}(\widehat{\Sigma}_{xx}) + \widehat{\text{SE}}(\widehat{\Sigma}_{xy}) = O_p(\sqrt{M/N})$. This suggests that a larger M leads to a larger value for $\widehat{\text{SE}}(\widehat{\Sigma}_{xx}) + \widehat{\text{SE}}(\widehat{\Sigma}_{xy})$, and then a worse statistical convergence rate. It is remarkable that a doubly stochastic matrix has been typically assumed in previous literature (Yuan, Ling, and Yin 2016; Tang et al. 2018; Richards and Rebeschini 2019; Richards, Rebeschini, and Rosasco 2020). It requires $\widehat{\text{SE}}(W) = 0$. However, here we only require that $\widehat{\text{SE}}(W)$ to be relatively small, that is $o_p(1/\sqrt{M})$ or $o_p(1\sqrt{N})$. For convenience, we define W that satisfies this condition to be a *weakly balanced network structure*.

To summarize, by Theorem 2, the statistical efficiency of the final estimator is determined by the following three factors: (a) the learning rate, (b) the network structure, and (c) the data distribution pattern. Regarding learning rate, we find that the statistical efficiency of $\widehat{\theta}^*$ improves as α decreases. Regarding W, we find that $\widehat{\theta}^*$ becomes more efficient if W is more balanced (e.g., W is of a doubly stochastic structure). Regarding data distribution pattern, we find that homogeneous data distribution leads to smaller distance between the resulting estimator and $\widehat{\theta}^*_{\text{ols}}$.

When α is fixed or W is arbitrarily specified, the NGD algorithm might not converge numerically to the global OLS estimator at all, even if the data are distributed independently and identically among different clients. This is particularly true if the sample size of each client is relatively small and the number of clients is relatively large (Kairouz et al. 2021). Consequently, to obtain an NGD estimator as efficient as the OLS estimator, both α and $\widehat{\text{SE}}(W)$ are critically important. To this end, the learning rate α should be sufficiently small and the network structure should be weakly balanced. This inspires us to study different network structures in this regard.

2.4. Network Structures

As demonstrated in the previous section, the network structure plays a very important role in the statistical efficiency of the proposed NGD method through $\widehat{\text{SE}}(W)$. In this section, we study the $\widehat{\text{SE}}(W)$ values of a number of important network structures. Specifically, we consider here three important network structures: the central-client network, the circle-type network, and the fixed-degree network.

Case 1 (Central-Client Network). We first consider the most typically used central-client structure. Without loss of generality, we assume that the first client with $m_1 = 1$ is the central computer, which should be connected with all the other clients. However, not all the other clients are connected with each other.

This leads to the network structure $A = (a_{m_1 m_2}) \in \mathbb{R}^{M \times M}$ with $a_{1 m_2} = a_{m_1 1} = 1$ for every $m_1, m_2 \neq 1$ and $a_{m_1 m_2} = 0$ otherwise. Accordingly, we have the weighting matrix $W = (w_{m_1 m_2}) \in \mathbb{R}^{M \times M}$ given by $W = [0, (M-1)^{-1} \mathbf{1}_{M-1}^{\top}; \mathbf{1}_{M-1}, \mathbf{0}]$. See the left panel of Figure 1 for an example with $M = 10$. Next, we evaluate $\widehat{\text{SE}}(W)$. It can be verified that $\widehat{\text{SE}}^2(W) = (M-1)^{-1}(M-2)^2$. The details of this derivation are given in Appendix B.3, supplementary materials, showing that the resulting estimator should be inconsistent when $M > 2$, and its performance could be even worse as M increases.

Case 2 (Circle-Type Network). In this case, assume the clients are already appropriately ordered with a fixed in-degree $d_{m_1} = D > 0$ for any $1 \leq m_1 \leq M$. Specifically, define a network structure as $A = (a_{m_1 m_2}) \in \mathbb{R}^{M \times M}$ with $a_{m_1 m_2} = 1$ if $m_2 = \{(m_1 + d - 1) \bmod M\} + 1$ for $1 \leq d \leq D$, where $a \bmod b$ represents the remainder after dividing a by b. Otherwise, we define $a_{m_1 m_2} = 0$. The resulting network structure should be of a circle type. See the middle panel of Figure 1 for an example with $M = 10$ and $D = 2$. The associated W matrix is given by

$$W = \begin{pmatrix} 0, & 1/D, & 1/D, \ldots, & 0 \\ 0, & 0, & 1/D, \ldots, & 0 \\ \ldots, & \ldots, & \ldots, & \ldots, \ldots \\ 1/D, & 1/D, & 1/D, \ldots, & 0 \end{pmatrix}. \tag{2.5}$$

Then, one can find W satisfying $\sum_{i=1}^{M} w_{ij} = \sum_{j=1}^{M} w_{ij} = 1$. This further implies that $\widehat{\text{SE}}^2(W) = 0$. As demonstrated in Section 2.3, this is an extremely well-balanced network structure. Nevertheless, it is challenging in practice to develop and maintain such a circle-type network structure, when M is relatively large.

Case 3 (Fixed-Degree Network). We next consider a fixed degree but a random-sampled network structure. Specifically, we fix the in-degree $d_{m_1} = D > 0$ for any $1 \leq m_1 \leq M$ with some prespecified D. Next, for each client m, we conduct simple random sampling without replacement by other clients. Write this sample as $\mathcal{M}_D$. We then define $a_{m_1 m_2} = 1$ if $m_2 \in \mathcal{M}_D$ and 0 otherwise. This leads to a network structure A, which is then fixed throughout the whole iteration process. See the right panel of Figure 1 for an example with $M = 10$. Next, we investigate its $\widehat{\text{SE}}(W)$ value. In this case, $\widehat{\text{SE}}(W)$ is a random variable. Take the expectation on $\sum_{i=1}^{M} w_{ij}$, we have $E(\sum_{i=1}^{M} w_{ij}) = \sum_{i=1}^{M} E(w_{ij}) = \sum_{i=1, i \neq j}^{M} D^{-1} P(w_{ij} = 1/D) = 1$. We next compute the variance of $\sum_{i=1}^{M} w_{ij}$. Because $P(w_{ij} = D^{-1}) = P(w_{ij}^2 = D^{-2}) = D/(M-1)$ for any $i \neq j$ and $P(w_{ii} = 0) = P(w_{ii} w_{ij} = 0) = 1$. Then, we have $\text{SE}(\sum_{i=1}^{M} w_{ij}) = E\{\widehat{\text{SE}}^2(W)\} = E(\sum_{i=1}^{M} w_{ij} - 1)^2 = E(\sum_{i=1}^{M} w_{ij})^2 - 1$. On the other hand, it can be verified (see Appendix B.3, supplementary materials for the details) that

$$E\left(\sum_{i=1}^{M} w_{ij}\right)^2 = \sum_{i=1}^{M} E(w_{ij}^2) + \sum_{i \neq k} E(w_{ij} w_{kj})$$

$$= (M-1)\frac{1}{D^2}\frac{D}{M-1}$$

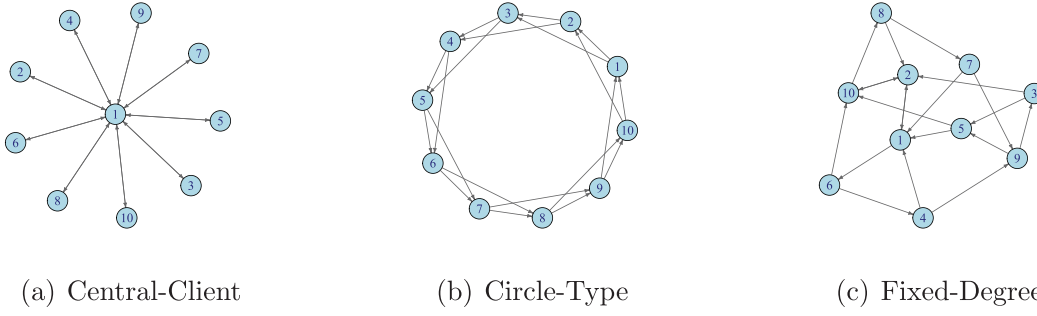(a) Central-Client   (b) Circle-Type   (c) Fixed-Degree

Figure 1. Examples of three important network structures. Left Panel: the central-client network; middle panel: the circle-type network; and right panel: the fixed-degree network.

$$+(M-1)(M-2)\frac{1}{D^2}\frac{D^2}{(M-1)^2}$$

$$=\frac{1}{D}-\frac{1}{M-1}+1.$$

Consequently, $\mathrm{SE}\left(\sum_{i=1}^{M} w_{ij}\right) = E\{\widehat{\mathrm{SE}}^2(W)\} = D^{-1} - (M-1)^{-1} \geq 0$. The statistical efficiency of the fixed degree network may be slightly worse when M is small than that in the circle-type network. However, fixed degree network is easy to be implemented in practice and is more robust against network attacks (Kairouz et al. 2021).

2.5. General Loss Functions

The abovementioned theoretical results are developed for the linear regression models and the OLS loss functions. We next extend those nice results to more general models and loss functions. In this regard, we assume $\mathcal{L}_N(\theta)$ is a general loss function. The true parameter $\theta_0 = \operatorname{argmin}_\theta E\{\mathcal{L}_N(\theta)\}$ and the global estimator $\widehat{\theta}_{\mathrm{ge}} = \operatorname{argmin}_\theta \mathcal{L}_N(\theta)$ are defined accordingly. Here we assume $\mathcal{L}_N(\cdot)$, and $E\{\mathcal{L}_N(\cdot)\}$ satisfy the standard regularity conditions in classical statistical analysis of M-estimators such that $\widehat{\theta}_{\mathrm{ge}} - \theta_0 = O_p(1/\sqrt{N})$ (Van der Vaart 2000; Jordan, Lee, and Yang 2019). For example, $\mathcal{L}_N(\theta)$ can be defined as a twice negative log-likelihood function of a generalized linear model. Accordingly, $\widehat{\theta}_{\mathrm{ge}}$ should be the maximum likelihood estimation (MLE). Then, the NGD algorithm can be executed as

$$\widehat{\theta}^{(t+1,m)} = \widetilde{\theta}^{(t,m)} - \alpha \dot{\mathcal{L}}_{(m)}\left(\widetilde{\theta}^{(t,m)}\right)$$

$$= \widetilde{\theta}^{(t,m)} - \alpha\left\{\dot{\mathcal{L}}_{(m)}(\widehat{\theta}_{\mathrm{ge}}) + \ddot{\mathcal{L}}_{(m)}(\widetilde{\xi}^{(t,m)})\left(\widetilde{\theta}^{(t,m)} - \widehat{\theta}_{\mathrm{ge}}\right)\right\}$$

$$= \left\{I_p - \alpha\ddot{\mathcal{L}}_{(m)}(\widetilde{\xi}^{(t,m)})\right\}\left(\sum_{k=1}^{M} w_{mk}\widehat{\theta}^{(t,k)}\right)$$

$$+\alpha\left\{\ddot{\mathcal{L}}_{(m)}(\widetilde{\xi}^{(t,m)})\widehat{\theta}_{\mathrm{ge}} - \dot{\mathcal{L}}_{(m)}(\widehat{\theta}_{\mathrm{ge}})\right\},$$

where $\dot{\mathcal{L}}_{(m)}(\theta)$ and $\ddot{\mathcal{L}}_{(m)}(\theta)$ represents the first- and second-order derivatives of $\mathcal{L}_{(m)}(\theta)$ with respect to θ. The algorithm can be rewritten in a matrix form as

$$\widehat{\theta}^{*(t+1)} = \widetilde{\Delta}^{*(t)}(W \otimes I_p)\widehat{\theta}^{*(t)} + \alpha\left\{\ddot{\mathcal{L}}^*(\widetilde{\xi}^{(t)})\widehat{\theta}_{\mathrm{ge}}^* - \dot{\mathcal{L}}^*(\widehat{\theta}_{\mathrm{ge}})\right\},$$

where $\ddot{\mathcal{L}}^*(\widetilde{\xi}^{(t)}) = \operatorname{diag}(\ddot{\mathcal{L}}_{(1)}(\widetilde{\xi}^{(t,1)}), \ddot{\mathcal{L}}_{(2)}(\widetilde{\xi}^{(t,2)}), \ldots, \ddot{\mathcal{L}}_{(M)}(\widetilde{\xi}^{(t,M)})) \in \mathbb{R}^{q\times q}$, $\widetilde{\Delta}^{*(t)} = I_q - \alpha\ddot{\mathcal{L}}^*(\widetilde{\xi}^{(t)})$, $\dot{\mathcal{L}}^*(\widehat{\theta}_{\mathrm{ge}}) =$

$(\dot{\mathcal{L}}_{(1)}^\top(\widehat{\theta}_{\mathrm{ge}}), \dot{\mathcal{L}}_{(2)}^\top(\widehat{\theta}_{\mathrm{ge}}), \ldots, \dot{\mathcal{L}}_{(M)}^\top(\widehat{\theta}_{\mathrm{ge}}))^\top \in \mathbb{R}^q$ and $\widehat{\theta}_{\mathrm{ge}}^* = I^*\widehat{\theta}_{\mathrm{ge}}$. Then, take the difference between $\widehat{\theta}^{*(t+1)}$ and the global estimator $\widehat{\theta}_{\mathrm{ge}}^*$, yielding

$$\widehat{\theta}^{*(t+1)} - \widehat{\theta}_{\mathrm{ge}}^* = \widetilde{\Delta}^{*(t)}(W \otimes I_p)\{\widehat{\theta}^{*(t)} - \widehat{\theta}_{\mathrm{ge}}^*\} - \alpha\dot{\mathcal{L}}^*(\widehat{\theta}_{\mathrm{ge}}), \quad (2.6)$$

Compare (2.6) for a general loss function with (2.4), and obtain the following key differences. First, the contraction operator $\widetilde{\Delta}^{*(t)}(W \otimes I_p)$ for a general loss changes as t changes. Second, there is an extra remainder term in (2.6) due to $\dot{\mathcal{L}}_{(m)}(\widehat{\theta}_{\mathrm{ge}}) \neq 0$. Nevertheless, we should have $\sum_{m=1}^{M} \dot{\mathcal{L}}_{(m)}(\widehat{\theta}_{\mathrm{ge}}) = 0$. All these differences make the corresponding theoretical investigation extremely challenging. Finally, denote $\Sigma_w = W^\top(I_M - M^{-1}\mathbf{1}_M\mathbf{1}_M^\top)W$, and write $\widehat{\mathrm{SE}}(\dot{\mathcal{L}}(\theta_0)) = \left\{M^{-1}\sum_{m=1}^{M}\|\dot{\mathcal{L}}_{(m)}(\theta_0)\|^2\right\}^{1/2}$. Similarly with $\widehat{\mathrm{SE}}(\widehat{\Sigma}_{xx})$ and $\widehat{\mathrm{SE}}(\widehat{\Sigma}_{xy})$, we could treat $\widehat{\mathrm{SE}}(\dot{\mathcal{L}}(\theta_0))$ as a measure for data distribution randomness. If observations are independent and identically distributed across different clients, $\widehat{\mathrm{SE}}(\dot{\mathcal{L}}(\theta_0))$ should be of $O_p(1/n)$ order. Otherwise, the value of $\widehat{\mathrm{SE}}(\dot{\mathcal{L}}(\theta_0))$ could be relatively larger. Based on the above notations, we obtain the following theorem to describe the discrepancy between the resulting NGD estimator and the global estimator.

Theorem 3 (Statistical Efficiency for General Loss Functions). Assume (C1) there are some positive constants $0 < \kappa_3 \leq \kappa_4 < \infty$, such that $\kappa_3 \leq \lambda_{\min}(\ddot{\mathcal{L}}_{(m)}(\theta)) \leq \lambda_{\max}(\ddot{\mathcal{L}}_{(m)}(\theta)) \leq \kappa_4$ for any $\theta \in \mathbb{R}^p$ and $1 \leq m \leq M$; (C2) $\widehat{\mathrm{SE}}(W)$ and α are sufficiently small such that $\lambda_{\max}^{1/2}(\Sigma_w) \leq \rho < 1$ and $\alpha\sigma_{\max}^w\kappa_4 + \widehat{\mathrm{SE}}(W) < 0.5(\kappa_4)^{-1}\kappa_3(1-\rho)$. Then, we have

$$\lim_{t\to\infty}\|\widehat{\theta}^{*(t)} - \widehat{\theta}_{\mathrm{ge}}^*\|/\sqrt{M} \leq c_2\left\{\widehat{\mathrm{SE}}(W) + \alpha\right\}\widehat{\mathrm{SE}}(\dot{\mathcal{L}}(\theta_0)) \quad (2.7)$$

for some positive constants $c_2 > 0$ with probability tending to one.

The detailed proof of Theorem 3 is given in Appendix B.4, supplementary materials. This theorem suggests that the good theoretical results for linear regression models and the OLS loss functions can be extended to more general models and loss functions. We find again that the distance between the NGD estimator $\widehat{\theta}^*$ and the global estimator $\widehat{\theta}_{\mathrm{ge}}^*$ is linearly bounded by $\{\alpha + \widehat{\mathrm{SE}}(W)\}\widehat{\mathrm{SE}}(\dot{\mathcal{L}}(\theta_0))$. To summarize, we find that the three factors (i.e., the learning rate, the network structure, and the data distribution pattern) together as a whole affect the convergence rate of the NGD estimator. Specifically, a statistical

efficient estimator can be obtained by either (a) $\alpha = o_p(1/\sqrt{N})$ and $\widehat{SE}(W) = o_p(1/\sqrt{N})$; or (b) $\widehat{SE}(\dot{\mathcal{L}}(\theta_0)) = O_p(1/\sqrt{n})$ and $\alpha = o_p(1/\sqrt{M})$, $\widehat{SE}(W) = o_p(1/\sqrt{M})$.

Theorem 3 studies the NGD algorithm under the global strong convexity and smoothness conditions. To further extend its applicability, we follow Nesterov (1998) and Jordan, Lee, and Yang (2019) and relax those technical conditions from the globally strong convexity to locally strong convexity. This leads to the following Corollary 1. The detailed proof of Corollary 1 is given in Appendix B.5, supplementary materials. We find that the key results of Theorem 3 remains valid.

Corollary 1 (Statistical Efficiency for Locally Strong Convex Functions). Assume conditions (C1′): there exist some positive constants $0 < \kappa_3' \leq \kappa_4' < \infty$, such that $\kappa_3' \leq \lambda_{\min}(\ddot{\mathcal{L}}_{(m)}(\widehat{\theta}_{ge})) \leq \lambda_{\max}(\ddot{\mathcal{L}}_{(m)}(\widehat{\theta}_{ge})) \leq \kappa_4'$, and $\|\ddot{\mathcal{L}}_{(m)}(\theta) - \ddot{\mathcal{L}}_{(m)}(\theta')\| \leq \kappa_4'\|\theta - \theta'\|$ for all θ, θ' in the neighborhood of $\widehat{\theta}_{ge}$ and (C2) hold. Further assume that the initial value $\widehat{\theta}^{(0,m)}$ lies close to $\widehat{\theta}_{ge}$, and $\alpha + \widehat{SE}(W)$ is sufficiently small. Then (2.7) in Theorem 3 holds.

Remark. It is remarkable that Corollary 1 assumed locally strong convexity. To deal with more general locally convex settings, a novel method has been developed by Ho et al. (2020) and Ren et al. (2022) for studying the estimation error and the computational complexity of algorithm-based estimators. The key idea is to construct an interesting reference estimator sequence for the actual algorithm-based estimator sequence. The reference sequence is constructed based on the population version of the interested algorithm. Next, by studying this population version reference sequence, the optimization error of the interested algorithm can be investigated. Furthermore, by studying the difference between the actual and reference estimator sequences, the stability error of the interested estimator can be controlled. However, how to apply this interesting technique to our NGD algorithm seems not immediately straightforward. It should be an interesting topic for future study.

Next, note that Theorem 3 is developed using standard asymptotic analysis techniques. The merit of this approach is that elegant theoretical results can be obtained in a more convenient way, with the help of a large sample size and infinite iterations. We next develop a parallel but non-asymptotic theoretical result with both finite sample size n and number of iterations t. To this end, denote $\delta_{\max}^0 = \max_{1 \leq m \leq M} \|\widehat{\theta}^{(0,m)} - \widehat{\theta}_{ge}\|$ represents the initial distance. This leads to the following non-asymptotic results.

Corollary 2 (Non-asymptotic Estimation Error). Under the conditions of Theorem 3, there exist constant $c_3 > 0$, such that

$$\frac{\|\widehat{\theta}^{*(t)} - \widehat{\theta}_{ge}^*\|}{\sqrt{M}} \leq \delta_{\max}^0(1 - \alpha\kappa_3)^t + c_3\{\widehat{SE}(W) + \alpha\}$$
$$\{\widehat{SE}(\dot{\mathcal{L}}(\theta_0)) + \|\widehat{\theta}_{ge} - \theta_0\|\}. \quad (2.8)$$

The detailed proof of Corollary 2 is given in Appendix B.4, supplementary materials. By (2.8), we are able to decompose the estimation error of $\widehat{\theta}^{*(t)}$ into three components. They are, respectively, the optimization error $\delta_{\max}^0(1 - \alpha\kappa_3)^t$, the statistical error due to network structure, learning rate, data distribution

pattern $\{\widehat{SE}(W) + \alpha\}\widehat{SE}(\dot{\mathcal{L}}(\theta_0))$, and the statistical error of the global estimator $\|\widehat{\theta}_{ge} - \theta_0\|$. Note that, the term $\delta_{\max}^0(1 - \alpha\kappa_3)^t$ is the optimization error. It linearly converges to 0 as t increases, which is consistent with the classical gradient descent methods (Nesterov 1998; Boyd, Boyd, and Vandenberghe 2004; Karimi, Nutini, and Schmidt 2016). Moreover, if the sample size is sufficiently large to support the asymptotic statistical inference, the statistical error of the global estimator $\|\widehat{\theta}_{ge} - \theta_0\|$ should be of $O_p(1/\sqrt{N})$ order. When both the sample size n and number of iterations t go to infinity, this result degenerates to that of Theorem 3.

2.6. Locally Over-Parameterized Problems

We study an interesting problem about the locally over-parameterized models in this section. By local over-parameterization, we mean that the number of parameters (i.e., p) is larger than the locally averaged sample size n but smaller than the whole sample size N. In contrast to a locally over-parameterized problem, a globally over-parameterized problem refers to the situation with $N > p$. In this case, the stable solution cannot be uniquely determined. This makes the discussion about the resulting estimator's theoretical statistical efficiency impossible. Consequently, we would like to focus on the locally over-parameterized problem in this section.

For illustration purpose, we consider here a linear regression model but under a locally over-parameterized framework (i.e., $n < p < N$). Following the same technique in Sections 2.1 and 2.2, we find that whether the NGD algorithm numerically converges to the stable solution is fully determined by the contraction operator $\Delta^*(W \otimes I_p)$. Once again, whether $\lambda_{\max}\{\Delta^*(W \otimes I_p)\} < 1$ is the key condition. Nevertheless, this problem becomes considerably more challenging with $n > p$. In this case, we have $\lambda_{\max}(W \otimes I_p) = 1$ and $\lambda_{\max}(\Delta^*) = 1$. Consequently, there indeed exists the theoretical possibility that $\lambda_{\max}\{\Delta^*(W \otimes I_p)\} = 1$; see Appendix C.1, supplementary materials for an example. If this happens, then the numerical convergence of the NGD algorithm can no longer be assured. Thus, the key question here is: how likely this undesirable situation can happen in real practice. To address this interesting question, we study here two important network structures very carefully. They are, respectively, the central-client network and the circle network.

Case 1 (Central-Client Network). As our first example, we consider here the central-client network structure. Recall that $W = [0, (M-1)^{-1}\mathbf{1}_{M-1}^\top; \mathbf{1}_{M-1}, 0]$. We can prove that for any eigenvalue λ of $\Delta^*(W \otimes I_p)$, there exists $\mathbf{v}_1 \in \mathbb{R}^p$, such that $(M-1)^{-1}\Delta^{(1)} \sum_{k=2}^M \Delta^{(k)} \mathbf{v}_1 = \lambda^2 \mathbf{v}_1$; see detailed proof in Appendix B.6, supplementary materials. As a consequence, it suffices to verify that $\lambda_{\max}(\Delta^{(1)}(M-1)^{-1} \sum_{k=2}^M \Delta^{(k)}) < 1$. Then it could be proved that

$$\Delta^{(1)}(M-1)^{-1} \sum_{k=2}^M \Delta^{(k)} = \left(I_p - \alpha\widehat{\Sigma}_{xx}^{(1)}\right)$$
$$\left\{I_p - \alpha(M-1)^{-1} \sum_{m=2}^M \widehat{\Sigma}_{xx}^{(m)}\right\}$$

$$= I_p - \alpha \left\{ \frac{M}{M-1} \widehat{\Sigma}_{xx} + \frac{M-2}{M-1} \widehat{\Sigma}_{xx}^{(1)} \right\}$$
$$+ O(\alpha^2).$$

Note that $\widehat{\Sigma}_{xx}$ is a positive definite matrix, and $\widehat{\Sigma}_{xx}^{(1)}$ is a semipositive definite matrix. Thus, when $0 < \alpha < 2(M-1)/\left\{ M\lambda_{\max}(\widehat{\Sigma}_{xx}) + (M-2)\lambda_{\max}(\widehat{\Sigma}_{xx}^{(1)}) \right\}$, the largest absolute eigenvalue of the leading term of $\Delta^{(1)}(M-1)^{-1}\sum_{k=2}^{M}\Delta^{(k)}$ (i.e., $I_p - \alpha(M-1)^{-1}\left\{ M\widehat{\Sigma}_{xx} + (M-2)\widehat{\Sigma}_{xx}^{(1)} \right\}$) is smaller than 1. Therefore, as long as α is sufficiently small, such that the $O(\alpha^2)$ is ignorable, we would have $\lambda_{\max}\left\{ \Delta^*(W \otimes I_p) \right\} < 1$.

Case 2 (Circle Network). Next, we consider circle-type network in (2.5) but with $D = 1$. Similarly, we can prove that for any eigenvalue λ of $\Delta^*(W \otimes I_p)$, there exists $v_1 \in \mathbb{R}^p$, such that $\left(\prod_{m=1}^{M} \Delta^{(m)} \right) v_1^{(p)} = \lambda^M v_1^{(p)}$. In addition, $\left(\prod_{m=1}^{M} \Delta^{(m)} \right)$ could be expanded as follows

$$\prod_{m=1}^{M} \Delta^{(m)} = \prod_{m=1}^{M} \left(I_p - \alpha \widehat{\Sigma}_{xx}^{(m)} \right) = I_p - \alpha M \widehat{\Sigma}_{xx} + O(\alpha^2).$$

Consequently, $\lambda_{\max}\left(\prod_{m=1}^{M} \Delta^{(m)} \right) < 1$ as long as α is sufficiently small.

Subsequently, we consider building a simulation study to check the numerical convergence of the NGD algorithm under more network structures (including the interesting counter-example we constructed). The detailed setting and results are shown in Appendix C.1, supplementary materials. From the results, it could be found that in the locally over-parameterized regime, the proposed NGD algorithm could still converge numerically under most of the network structures.

3. Numerical Studies

3.1. The Basic Setups

We present here a number of numerical studies to demonstrate the finite sample performances of the NGD method based on both simulation studies and a real data example. For simulation studies, several simulation models are used to generate the whole sample data. Once the sample data are generated, they are then distributed to different clients. We consider two different distribution patterns. The first one is *homogeneous*. All the observations are distributed to different clients randomly. Thus, the sample distribution across different clients should be homogeneous. The second one is *heterogeneous*. We sort all the observations according to their response values. Then, we sequentially distribute them to different clients according to their sorted response values. Thus, the observations allocated to the same clients share similar response values, and then, distributions should be heterogeneous across different clients. As a result, the estimators produced by different clients separately can hardly be consistent. Once all the simulated data are generated and distributed across different clients, the clients are then connected with each other by one particular type of network structure, as described in Section 2.4.

For the entire simulation study, we fix the whole sample size as $N = 10{,}000$ and the number of clients as $M = 200$. Thus,

the sample size for each client is given by $n = N/M = 50$. For each simulation study, the experiments are replicated for a total of $R = 500$ times for each parameter setup. Let $\widehat{\theta}_r^{*(t)}$ be the estimator obtained in the rth replicate ($1 \leq r \leq R$) on the tth iteration with the initial value given by $\widehat{\theta}_r^{*(0)} = \mathbf{0}$ for every $1 \leq r \leq R$. We then compute its mean squared error (MSE) values averaged from the whole clients as $\|\widehat{\theta}_r^{*(t)} - \theta_0^*\|^2/M$, where $\theta_0^* = I^*\theta_0$ represents the stacked true parameters. This leads to a total of R MSE values in log-scale. We then demonstrate the performance of the proposed algorithm for different models and network structures in terms of the log(MSE) values.

3.2. Least Squares Loss Functions

We start with a linear regression model. Following Tibshirani (1996), we set $p = 8$ and $\theta_0 = (3, 1.5, 0, 0, 2, 0, 0, 0)^\top$. Here, X_i is generated from a multivariate normal distribution with mean 0 and $\text{cov}(X_{ij_1}, X_{ij_2}) = \rho^{|j_1 - j_2|}$ with $\rho = 0.5$ for $1 \leq j_1, j_2 \leq p$. The residual term ε_i is independently generated from the standard normal distribution. For comparison, we fix the in-degrees in the circle-type and the fixed-degree network structures to be 1 and 2, respectively. Various learning rates (i.e., $\alpha = 0.005, 0.01, 0.02, 0.05$) are considered. Subsequently, the NGD estimators are computed. The median log(MSE) values are reported in Figure 2.

By Figure 2, we obtain the following interesting findings. First, we find that the circle-type network structure seems to be the best network structure, with the lowest median log(MSE) values. This is not surprising, because it has the smallest $\widehat{\text{SE}}(W)$ values as 0. Instead, the central-client network structure demonstrates the worst performance based on its largest $\widehat{\text{SE}}(W)$ values. Unfortunately, this is one of the most popularly used network structures in practice. The fixed-degree network structure performs slightly worse than that of the circle-type network structure. Recall that the in-degree of the fixed-degree network structure is fixed to be 2 in this example. As demonstrated in the next section, its performance can be greatly improved by allowing the in-degree to be slightly larger. Second, comparing the upper and bottom panels, we find that little difference is detected. This might be due to the fact α is set to be sufficiently small and M is relatively large. Third, for each network structure, we find that the smaller the learning rate α is, the slower the algorithm converges and the more efficient the NGD estimator is. All these results are in line with our theoretical findings in Theorem 2 very well.

3.3. General Loss Functions

In this section, we further demonstrate the finite sample performance of the NGD methods on different models and general loss functions. Compared with the previous section, the key diffidence is that the loss function is replaced by more general ones. Specifically, we consider the negative two-times log-likelihood function as the general loss function with the following simulation examples.

Example 1 (Logistic Regression). In this example, we consider a logistic regression, which is one of the most popularly used
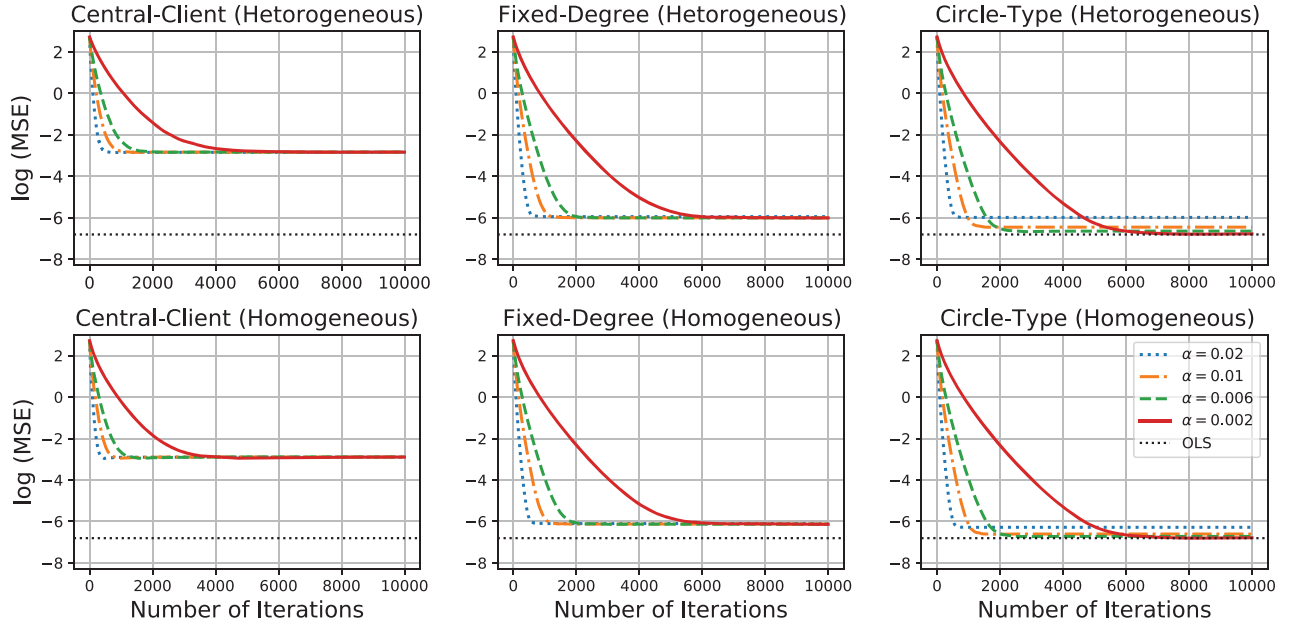
Figure 2. The median log(MSE) values for different learning rates, network structures, and distribution patterns based on linear regression models. The black dotted line represents the global OLS estimators.

models for classification. Consider an example in Barut, Fan, and Verhasselt (2016). Set $p = 6$ and $\theta_0 = (1/2, 1/2, 1/2, 1/2, 1/2, -2.5/2)^\top$. The covariate X_i is generated from a multivariate normal distribution with $E(X_i) = 0$, var$(X_i) = 1$, and cov$(X_{ij_1}, X_{ij_2}) = 0.5$ for $j_1 \neq j_2$. Given X_i, the response Y_i is then generated according to $P(Y_i = 1|X_i, \theta_0) = \{1 + \exp(-X_i^\top \theta_0)\}^{-1}$. The learning rates are set to 0.02, 0.05, 0.1, and 0.2.

Example 2 (Poisson Regression). This is an example revised from Fan and Li (2001). Specifically, the feature dimension is $p = 8$ and $\theta_0 = (1.2, 0.6, 0, 0, 0.8, 0, 0, 0)^\top$. The first six components of X_i are generated in the same way as in Section 3.1 but with $\rho = 0.2$. The last two components of X_i are independently and identically distributed as a Bernoulli distribution with probability of success 0.5. All covariates are standardized with mean 0 and variance 1. Conditional on X_i, the response Y_i is generated from a Poisson distribution with $E(Y_i|X_i) = \exp(X_i^\top \theta_0)$. The learning rates are set to 2×10^{-4}, 3×10^{-4}, 5×10^{-4}, and 8×10^{-4}.

The detailed results of the logistic regression are given in Figure 3, while those of the Poisson regression are summarized in Figure 4. All the results are qualitatively similar to those in Figure 2. Specifically, we find that larger α leads to faster numerical convergence but worse statistical efficiency. Furthermore, the network structure plays a significant role in determining the statistical efficiency of the NGD estimators. The circle-type network structure is the best, and the central-client network structure is the worst. In addition, data heterogeneity has little influence on the statistical efficiency of the NGD estimators.

3.4. Nodal Degree Effect for Fixed Degree Networks

As mentioned in Section 2.3, the network structure plays an important role in determining the NGD estimators' statistical efficiency. Among various possible network structures, the

fixed-degree type network is easy to implement in practice. Thus, we study its finite sample performance in this section. In particular, we focus on the effect of nodal degrees. Accordingly, the simulation example with the fixed-degree network structure in Section 2.4 is replicated for both OLS problem and general loss functions but with different nodal degrees. To ensure that the NGD algorithm converges to the global estimator, the learning rate considered for these three models is set to be sufficiently small as $\alpha = 2 \times 10^{-3}$ for the linear regression, 2×10^{-2} for the logistic regression, and 2×10^{-4} for the Poisson regression. The log(MSE) values are then box-plotted in Figure 5. We find that, as the in-degree increases, the statistical efficiency of the resulting estimator improves. In particular, the median log(MSE) decreases greatly when the in-degree increases from 1 to 2. We also find that, when the in-degree is no less than 6, the statistical efficiency of the resulting NGD estimator becomes very comparable with that of the global estimator.

3.5. Deep Learning Models

In this section, we apply the proposed NGD method to more sophisticated deep learning models for large-scale datasets. Specifically, we conduct the following two experiments. The first experiment uses the MNIST dataset (LeCun et al. 1998, http://yann.lecun.com/exdb/mnist/). It contains 70,000 photos of hand-written digits (0–9), and each class contains about 7000 images. 60,000 of the dataset are for training and the rest are for validation. For this dataset, we consider the LeNet model (LeCun et al. 1998) with $p = 61,706$ parameters. The second experiment studies the CIFAR10 dataset (Krizhevsky and Hinton 2009, http://www.cs.toronto.edu/~kriz/cifar.html), which contains 60,000 color images. The dataset forms 10 equal-size classes. Among the dataset, 50,000 are for training and 10,000 are for validation. For this dataset, we consider the MobileNet model (Howard et al. 2017) with $p = 3,488,522$ parameters.
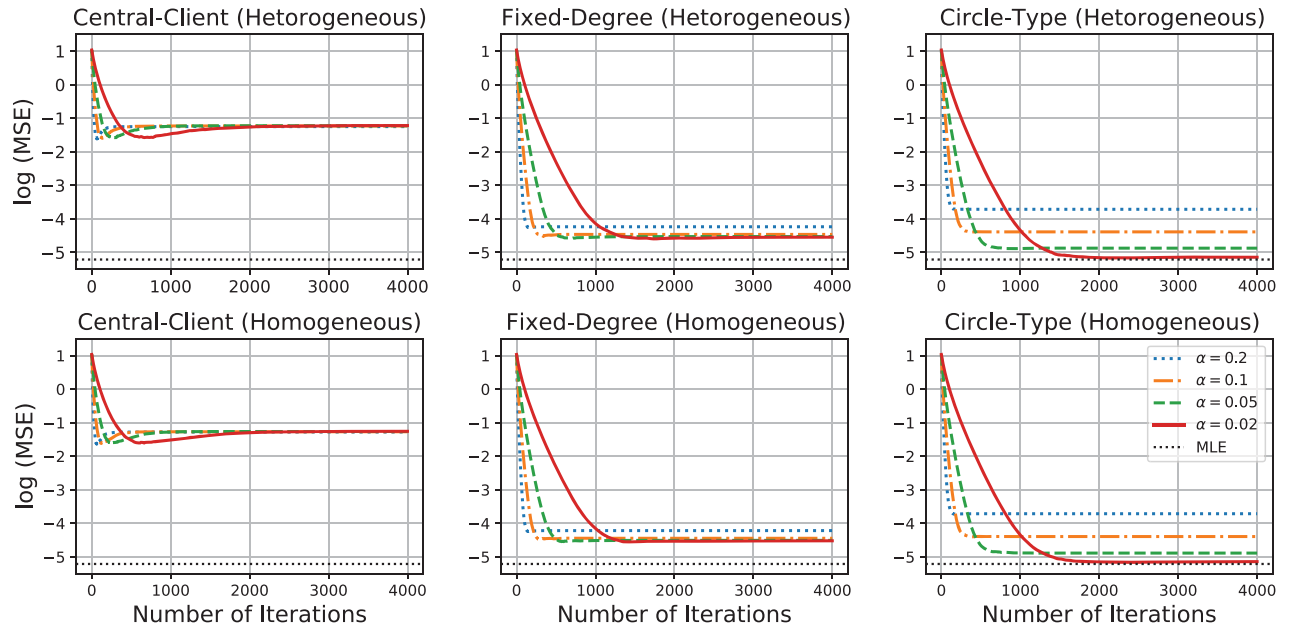
Figure 3. The median log(MSE) values for different learning rates, network structures, and distribution patterns based on a logistic regression model. The black dotted line represents the global MLE estimators.
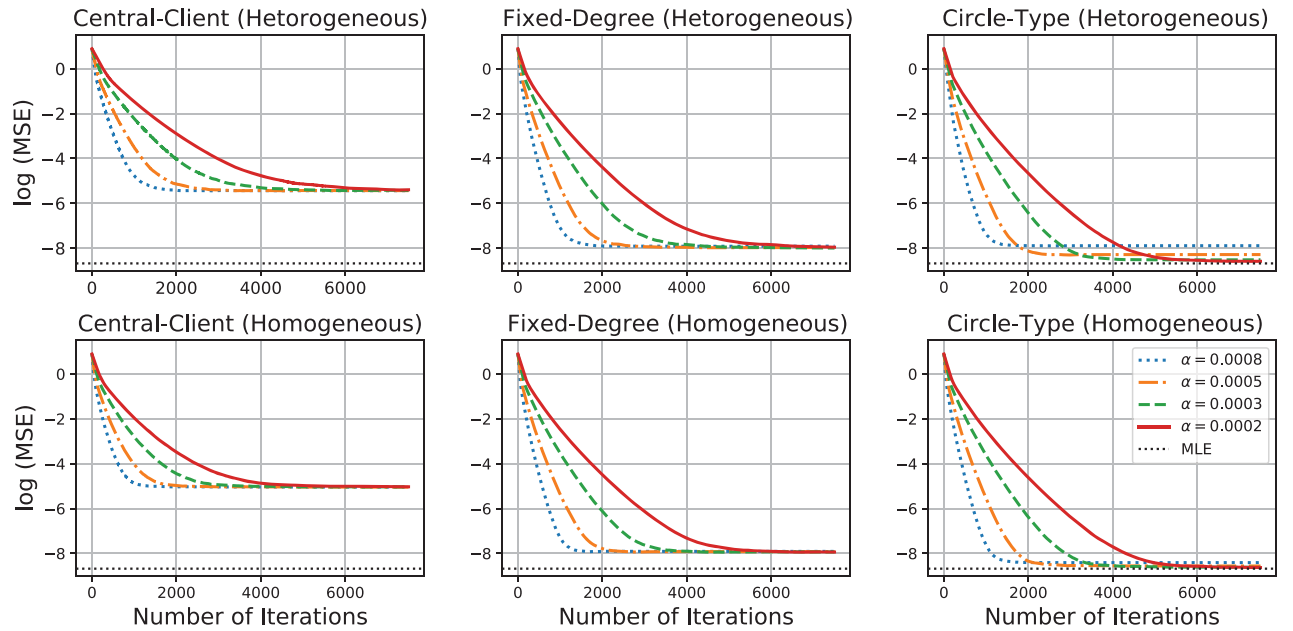

Figure 4. The median log(MSE) values for different learning rates, network structures, and distribution patterns based on a Poisson regression model. The black dotted line represents the global MLE estimators.

To train the model, we distribute the training data into M clients with $M = 40$ for MNIST and $M = 25$ for CIFAR10. The data are sorted by response labels first and then distributed to different clients. Thus, most of the clients contain only one class of response label, which is extremely heterogeneous across different clients. Three different network structures are studied, they are respectively: (a) the central-client network structure, (b) the circle type network structure with $D = 2$, and (c) the fixed degree network structure with $D = 6$. To accelerate the numerical convergence, the constant-and-cut learning rate scheduling strategy is adopted (Wu et al. 2018; Lang, Xiao, and Zhang 2019) for MNIST with initial learning rate $\alpha = 0.01$. Then, it drops to 0.005 and 0.001 after 1000 and 4000

iterations, respectively. For CIFAR10, since MobileNet requires larger amount of GPU memory, we use the mini-batch strategy (Cotter et al. 2011; Li et al. 2014) with batch size 128 and initial learning rate 0.002. Define one epoch to represent the process that the local gradient has been updated for every client based on all of its batches. The learning rate decreases to 0.001 after 2800 epochs. As to the initial value $\widehat{\theta}^{(0,m)}$, we adopt Xavier uniform initializer (Glorot and Bengio 2010) for MNIST, and the pretrained weights on ImageNet dataset for CIFAR10.

The prediction performance of $\widehat{\theta}^{(t,m)}$ is then evaluated by the prediction error based on the validation set, which is denoted by $\text{Err}^{(t,m)}$. Then, for any given network structure and a given t, we should obtain a total of M Err-values. Their means and
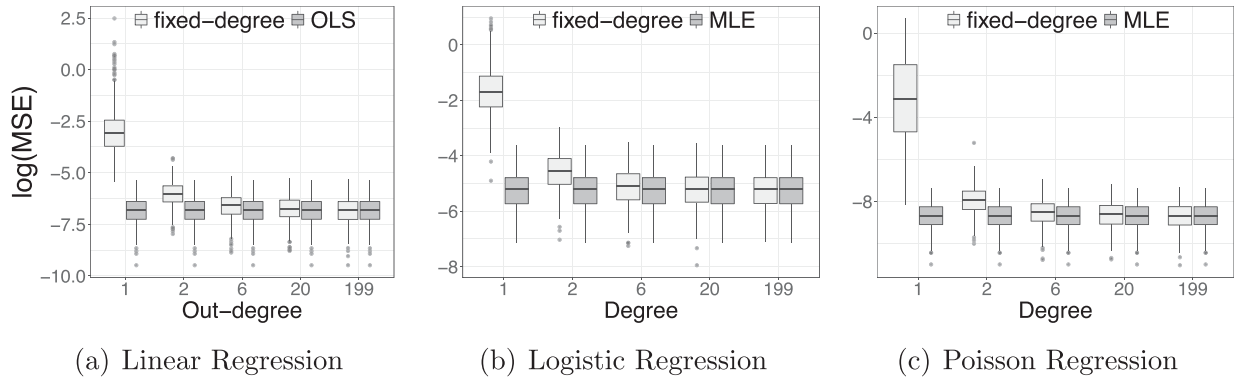
(a) Linear Regression   (b) Logistic Regression   (c) Poisson Regression

Figure 5. The log(MSE) values for the fixed degree networks. The dark gray box represents the global estimator (OLS or MLE). From the left to the right panel, the learning rate is fixed to be $\alpha = 2 \times 10^{-3}, 2 \times 10^{-2}$, and 2×10^{-4}.
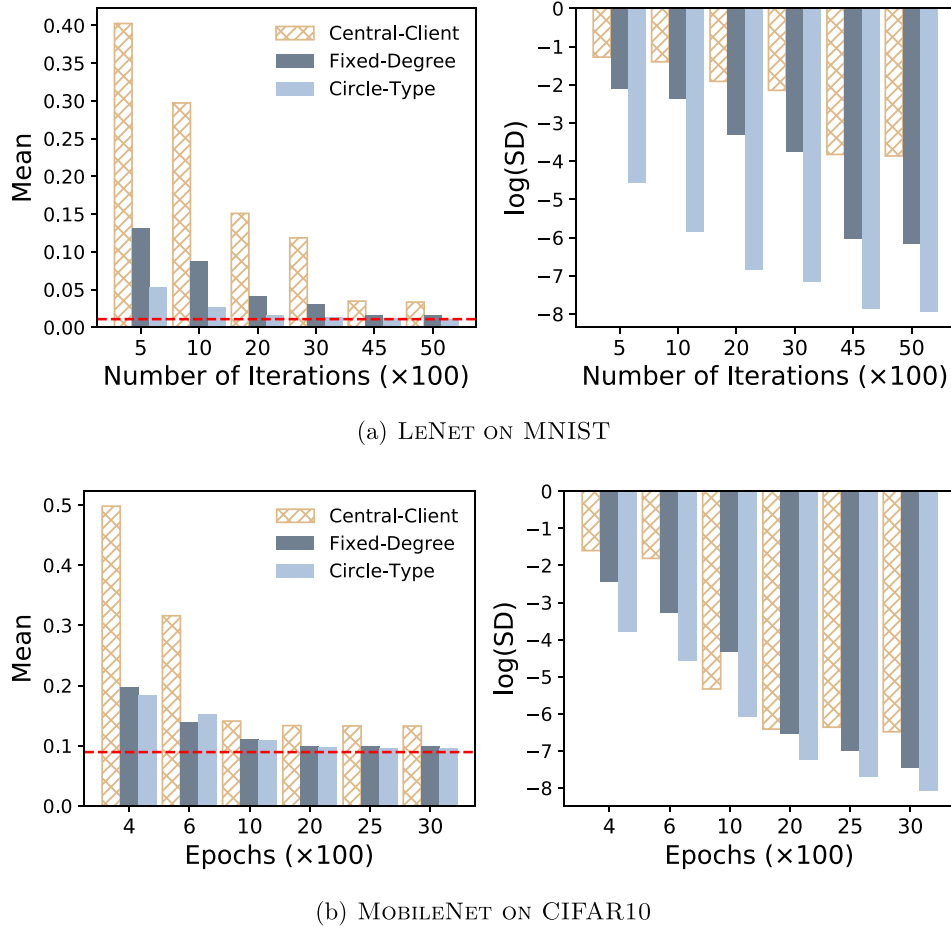

(a) LᴇNᴇᴛ ᴏɴ MNIST

(b) MᴏʙɪʟᴇNᴇᴛ ᴏɴ CIFAR10

Figure 6. The mean and log(SD) for the Err values obtained by deep learning models. Different network structures are considered. The dotted line in the left panel represents the optimal Err value calculated based on the whole dataset using one single computer.

log-transformed standard deviations are bar-plotted in Figure 6. For comparison, we also report the *optimal Err value* calculated based on the whole dataset using one single computer. By Figure 6, we obtain two interesting findings. First, from the left panel, we find that the mean Err values of the central-client type network structure are the largest. In the contrast, those of both the circle-type and fixed-degree networks are much smaller. By the time of convergence, the prediction errors based on those two network structures could be as small as the optimal one. While the circle-type network slightly outperforms the

fixed-degree network. Second, from the right panel, it could be concluded the standard deviation of the Err values (in log-scale) for the central-client network is always the largest, while that of the circle-type network is the smallest.

4. Concluding Remarks

In this study, we develop a methodology for a fully decentralized federated learning algorithm. Both the theoretical and numerical properties of the algorithm are carefully studied. We find that

the numerical convergence properties are mainly determined by the learning rate. However, the statistical properties of the resulting estimators are related to the learning rate, network structure and data distribution pattern. A sufficiently small learning rate and balanced network structure are required for better statistical efficiency, even if data are distributed heterogeneously. Extensive numerical studies are presented to demonstrate the finite sample performance.

Finally, we discuss interesting topics for future study. First, our study analyses a synchronous NGD algorithm, which ignores asynchronous problems, which should be the subject of further study. Second, our theorem suggests that a sufficiently small learning rate should be used for the best statistical efficiency. However, in practice, an unnecessarily small learning rate would lead to painfully slow numerical convergence. Then, an interesting research problem is how to practically schedule the learning rate to balance between statistical efficiency and numerical convergence speed. In addition, our methodology focuses on the situation when the loss function is globally or locally strong convex. How to generalize our theoretical results to more general locally convex settings (Ho et al. 2020; Ren et al. 2022) remains to be a challenging but also an interesting topic for future study.

Supplementary Materials

Supplementary_Material.pdf: This document provides the extensions of the proposed method, the proofs of the theoretical results in the main text, and some additional simulation results. Appendix A provides for technical lemmas which are useful to prove the results in the main text. Appendix B contains the detailed proofs of the main theorems and corollaries developed in the main text. Appendix C reports some extensions and discussions of the proposed method.

Code.zip: This file is the python code for the proposed method. Please see the "README.md" in the file for using the code.

Funding

References

Barut, E., Fan, J., and Verhasselt, A. (2016), "Conditional Sure Independence Screening," *Journal of the American Statistical Association*, 111, 1266–1277. [814]

Bellet, A., Guerraoui, R., Taziki, M., and Tommasi, M. (2018), "Personalized and Private Peer-to-Peer Machine Learning," in *International Conference on Artificial Intelligence and Statistics*, PMLR, pp. 473–481. [807]

Blot, M., Picard, D., Cord, M., and Thome, N. (2016), "Gossip Training for Deep Learning," arXiv preprint arXiv:1611.09726. [807,808]

Boyd, S., Boyd, S. P., and Vandenberghe, L. (2004), *Convex Optimization*, Cambridge: Cambridge University Press. [812]

Cheu, A., Smith, A., Ullman, J., Zeber, D., and Zhilyaev, M. (2019), "Distributed Differential Privacy via Shuffling," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, pp. 375–403. [806]

Colin, I., Bellet, A., Salmon, J., and Clémençon, S. (2016), "Gossip Dual Averaging for Decentralized Optimization of Pairwise Functions," in *International Conference on Machine Learning*, PMLR, pp. 1388–1396. [807,808]

Cotter, A., Shamir, O., Srebro, N., and Sridharan, K. (2011), "Better Mini-Batch Algorithms via Accelerated Gradient Methods," *NIPS*, 24, 1647–1655. [815]

Fan, J., and Li, R. (2001), "Variable Selection via Nonconcave Penalized Likelihood and its Oracle Properties," *Journal of the American Statistical Association*, 96, 1348–1360. [814]

Glorot, X., and Bengio, Y. (2010), "Understanding the Difficulty of Training Deep Feedforward Neural Networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, JMLR Workshop and Conference Proceedings, pp. 249–256. [815]

Hashimoto, T., Srivastava, M., Namkoong, H., and Liang, P. (2018), "Fairness Without Demographics in Repeated Loss Minimization," in *International Conference on Machine Learning*, PMLR, pp. 1929–1938. [806]

Ho, N., Khamaru, K., Dwivedi, R., Wainwright, M. J., Jordan, M. I., and Yu, B. (2020), "Instability, Computational Efficiency and Statistical Accuracy," arXiv preprint arXiv:2005.11411. [812,817]

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. (2017), "Mobilenets: Efficient Convolutional Neural Networks for Mobile Vision Applications," arXiv preprint arXiv:1704.04861. [814]

Jordan, M. I., Lee, J. D., and Yang, Y. (2019), "Communication-Efficient Distributed Statistical Inference," *Journal of the American Statistical Association*, 114, 668–681. [811,812]

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., D'Oliveira, R. G. L., Eichner, H., Rouayheb, S. E., Evans, D., Gardner, J., Garrett, Z., Gascón, A., Ghazi, B., Gibbons, P. B., Gruteser, M., Harchaoui, Z., He, C., He, L., Huo, Z., Hutchinson, B., Hsu, J., Jaggi, M., Javidi, T., Joshi, G., Khodak, M., Konecný, J., Korolova, A., Koushanfar, F., Koyejo, S., Lepoint, T., Liu, Y., Mittal, P., Mohri, M., Nock, R., Özgür, A., Pagh, P., Qi, H., Ramage, D., Raskar, R., Raykova, M., Song, D., Song, W., Stich, S. U., Sun, Z., Suresh, A. T., Tramér, F., Vepakomma, P., Wang, J., Xiong, L., Xu, Z., Yang, Q., Yu, F. X., Yu, H., and Zhao, S. (2021), "Advances and Open Problems in Federated Learning," *Foundations and Trends® in Machine Learning*, 14, 1–210. [810,811]

Karimi, H., Nutini, J., and Schmidt, M. (2016), "Linear Convergence of Gradient and Proximal-Gradient Methods Under the Polyak-Łojasiewicz Condition," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, pp. 795–811. [812]

Konečný, J., McMahan, H. B., Yu, F. X., Richtárik, P., Suresh, A. T., and Bacon, D. (2016), "Federated Learning: Strategies for Improving Communication Efficiency," arXiv preprint arXiv:1610.05492. [806]

Krizhevsky, A. and Hinton, G. (2009), "Learning Multiple Layers of Features from Tiny Images," *Handbook of Systemic Autoimmune Diseases*, 1(4). [814]

Lalitha, A., Kilinc, O. C., Javidi, T., and Koushanfar, F. (2019), "Peer-to-peer Federated Learning on Graphs," arXiv preprint arXiv:1901.11173. [807,808]

Lalitha, A., Shekhar, S., Javidi, T., and Koushanfar, F. (2018), "Fully Decentralized Federated Learning," in *Third Workshop on Bayesian Deep Learning (NeurIPS)*. [807,808]

Lang, H., Xiao, L., and Zhang, P. (2019), "Using Statistics to Automate Stochastic Optimization," *Advances in Neural Information Processing Systems*, 32, 9540–9550. [815]

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998), "Gradient-Based Learning Applied to Document Recognition," *Proceedings of the IEEE*, 86, 2278–2324. [814]

Li, M., Zhang, T., Chen, Y., and Smola, A. J. (2014), "Efficient Mini-Batch Training for Stochastic Optimization," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 661–670. [815]

Li, Y., Chen, C., Liu, N., Huang, H., Zheng, Z., and Yan, Q. (2021), "A Blockchain-Based Decentralized Federated Learning Framework with Committee Consensus," *IEEE Network*, 35, 234–241. [807]

Lian, X., Zhang, C., Zhang, H., Hsieh, C.-J., Zhang, W., and Liu, J. (2017), "Can Decentralized Algorithms Outperform Centralized Algorithms?

A Case Study for Decentralized Parallel Stochastic Gradient Descent," arXiv preprint arXiv:1705.09056. [807,808]

Lian, X., Zhang, W., Zhang, C., and Liu, J. (2018), "Asynchronous Decentralized Parallel Stochastic Gradient Descent," in *International Conference on Machine Learning*, PMLR, pp. 3043–3052. [807,808]

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. (2017), "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Artificial Intelligence and Statistics*, PMLR, pp. 1273–1282. [806]

Nedic, A., Olshevsky, A., and Shi, W. (2017), "Achieving Geometric Convergence for Distributed Optimization Over Time-Varying Graphs," *SIAM Journal on Optimization*, 27, 2597–2633. [807,808]

Nesterov, Y. (1998), "Introductory Lectures on Convex Programming Volume I: Basic Course," *Lecture notes*, 3, 5. [812]

Rao, C. R., Rao, C. R., Statistiker, M., Rao, C. R., and Rao, C. R. (1973), *Linear Statistical Inference and its Applications* (Vol. 2), New York: Wiley. [808]

Ren, T., Cui, F., Atsidakou, A., Sanghavi, S., and Ho, N. (2022), "Towards Statistical and Computational Complexities of Polyak Step Size Gradient Descent," arXiv preprint arXiv:2110.07810. [812,817]

Richards, D., and Rebeschini, P. (2019), "Optimal Statistical Rates for Decentralised Non-parametric Regression with Linear Speed-Up," arXiv preprint arXiv:1905.03135. [807,808,810]

Richards, D., Rebeschini, P., and Rosasco, L. (2020), "Decentralised Learning with Random Features and Distributed Gradient Descent," in *International Conference on Machine Learning*, PMLR, pp. 8105–8115. [807,808,810]

Savazzi, S., Nicoli, M., and Rampa, V. (2020), "Federated Learning with Cooperating Devices: A Consensus Approach for Massive IoT Networks," *IEEE Internet of Things Journal*, 7, 4641–4654. [807,808]

Shao, J. (2003), *Mathematical Statistics*, Springer Texts in Statistics, New York: Springer. [808]

Smith, V., Chiang, C.-K., Sanjabi, M., and Talwalkar, A. (2017), "Federated Multi-Task Learning," arXiv preprint arXiv:1705.10467. [806]

Tang, H., Lian, X., Yan, M., Zhang, C., and Liu, J. (2018), "Decentralized Training Over Decentralized Data," in *International Conference on Machine Learning*, PMLR, pp. 4848–4856. [807,808,809,810]

Tibshirani, R. (1996), "Regression Shrinkage and Selection via the Lasso," *Journal of the Royal Statistical Society*, Series B, 58, 267–288. [813]

Van der Vaart, A. W. (2000), *Asymptotic Statistics* (Vol. 3), Cambridge: Cambridge University Press. [811]

Vanhaesebrouck, P., Bellet, A., and Tommasi, M. (2017), "Decentralized Collaborative Learning of Personalized Models Over Networks," in *Artificial Intelligence and Statistics*, PMLR, pp. 509–517. [807,808]

Wu, Y., Ren, M., Liao, R., and Grosse, R. (2018), "Understanding Short-Horizon Bias in Stochastic Meta-Optimization," arXiv preprint arXiv:1803.02021. [815]

Yuan, K., Ling, Q., and Yin, W. (2016), "On the Convergence of Decentralized Gradient Descent," *SIAM Journal on Optimization*, 26, 1835–1854. [807,808,809,810]