IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 20 August 2023, accepted 10 September 2023, date of publication 19 September 2023,
date of current version 27 September 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3317298

RESEARCH ARTICLE

Hybrid Distributed Optimization for Learning Over Networks With Heterogeneous Agents

MOHAMMAD H. NASSRALLA[1,2], NAEEM AKL[3], AND ZAHER DAWY[1], (Senior Member, IEEE)

[1]Department of Electrical and Computer Engineering, American University of Beirut, Beirut 1107 2020, Lebanon
[2]Infineon Technologies AG, 81726 Gleichen, Germany
[3]Qualcomm, Bridgewater, NJ 08807, USA

Corresponding author: Zaher Dawy (zd03@aub.edu.lb)

This work was supported by the Qatar National Research Fund (a member of Qatar Foundation) funded by National Priorities Research Program (NPRP) under Grant NPRP12S-0305-190231. The findings achieved herein are solely the responsibility of the authors.

ABSTRACT This paper considers distributed optimization for learning problems over networks with heterogeneous agents having different computational capabilities. The heterogeneity of computational capabilities implies that a subset of the agents may run computationally-intensive learning algorithms like Newton's method or full gradient descent, while the other agents can only run lower-complexity algorithms like stochastic gradient descent. This leads to opportunities for designing hybrid distributed optimization algorithms that rely on cooperation among the network agents in order to enhance overall performance, improve the rate of convergence, and reduce the communication overhead. We show in this work that hybrid learning with cooperation among heterogeneous agents attains a stable solution. For small step-sizes μ, the proposed approach leads to small estimation error in the order of $O(\mu)$. We also provide the theoretical analysis of the stability of the first, second, and fourth order error moments for learning over networks with heterogeneous agents. Finally, results are presented and analyzed for case study scenarios to demonstrate the effectiveness of the proposed approach.

INDEX TERMS Distributed optimization, diffusion strategy, gradient descent, heterogeneous networks, Newton's method, stochastic gradient descent, stability analysis.

I. INTRODUCTION

Distributed optimization is a key enabler for collaborative learning in intelligent networks in which the edge devices only have access to their local data streams. It provides a scalable solution to the distributed inference problem in scenarios where communication between nodes is costly and data centralization raises privacy concerns [1], [2], [3], [4], [5], [6], [7]. Most prior literature focuses on distributed optimization in homogeneous networks where all agents have similar computational capabilities and apply the same learning algorithm. However, real network deployments have much richer structure and may comprise agents with various energy constraints and hardware complexities [8], [9], [10], [11]. These agents may cooperatively solve one optimization problem, each using their own distinct learning method. Stability and convergence of this hybrid optimization problem must be guaranteed. The goal of this work is to investigate this issue and evaluate the performance gains of distributed learning in heterogeneous networks.

Learning over networks induces many degrees of freedom and there are many factors that can improve or degrade the distributed learning performance. For instance, the following design options have an imperative impact on the performance of distributed learning algorithms: the optimization algorithm that is being run by every agent, the distributed learning strategy, the selection of the weights over the edges of the network, the synchronous/asynchronous reception of shared parameters over the network [12], [13], [14], the topology of the network [15], [16] (strongly connected, weakly connected, etc.), the link failures [17], the exchange of noisy information [18], [19], [20], the malicious behavior [21], etc. In particular, the weights over the edges of the network play an important role in improving the minimization of the cost function of every agent and they are chosen according to different rules like the metropolis [22], [23],

The associate editor coordinating the review of this manuscript and approving it for publication was Abdel-Hamid Soliman.

Hasting [24], [25] relative degree [26], Laplacian [27], [28], [29], etc.

The proposed algorithms in the literature for optimization, estimation and learning over networks assume that the distributed agents have similar or homogeneous computational capabilities. Having similar computational capabilities means that the distributed agents should normally run the same iterative optimization algorithm. For instance, the methods of incremental [16], consensus [16], [30], diffusion [16], [31], enlarged cooperation [16] and spatio-temporal [32] distributed strategies are proposed in the literature where all agents are assumed to have homogeneous computational capabilities. In this work, we study learning over networks when the distributed agents have different or heterogeneous computational capabilities and can run different iterative optimization algorithms while cooperating among each other. For instance, a subset of the capable agents can run Newton's method, full gradient descent, while the other subset may only be able to run stochastic gradient descent (SGD) due to their limited computational capabilities.

There exist many applications for optimizing learning over multiple agents with heterogeneous computational capabilities [33], [34], [35]. For instance, in wireless cellular networks with cooperative and distributed learning, some mobile phones may have more computational capabilities than others, and base stations may serve as edge computing nodes with much higher capabilities than cell phones [5], [36], [37], [38]. In wireless sensor networks (WSN), sensor devices normally have varying levels of computational, energy, and storage resources, which restricts their ability to run different classes of distributed learning algorithms [39], [40], [41], [42], [43]. Mixed models of hybrid optimization algorithms can also find application use cases in distributed power systems, computer networks, industrial control systems, etc.

The main contribution of this paper is in proposing a hybrid distributed optimization approach for learning over networks with heterogeneous agents and in proving theoretically its stability. Although there are many papers in the literature that deal with heterogeneous computing [44], [45], [46], [47], [48], [49], this is the first work that theoretically and experimentally investigates the cooperation between agents when they perform different optimization algorithms. The theoretical and simulation results show that the first, second and fourth order moments converge to a stationary point and attain a stable solution. This paper shows that the cooperation among agents that can run fast algorithms like Newton's method and slow ones like SGD can improve the rate of convergence and performance of all agents. Improving the rate of convergence and performance for agents with limited computational capabilities leads to reducing the local power consumption, decreasing the communication overhead over the network, and saving financial costs resulting from running the network for a long time.

The rest of the paper is organized as follows. Section II presents the network model and the problem formulation. Section III provides the stability of the first, second, and fourth order moments of learning among heterogeneous multi-agent systems. Simulation results and analysis are presented in Section IV. Finally conclusions are drawn in Section V.

II. PROBLEM FORMULATION

We assume a network of N interacting agents with different computational capabilities, which allows the distributed agents to run heterogeneous optimization algorithms. Different algorithms may be run over the network agents and depending on the the available computational resources, the agent can determine which algorithm it can handle. We define a parameter δ to distinguish among the optimization algorithms over the network as follows:

$$\delta = \begin{cases} NM & \text{for agents with high computational capabilities and run the Newton's method,} \\ QN & \text{for agents with high computational capabilities and run the Quasi-Newton method,} \\ GD & \text{for agents with high computational capabilities and run the gradient descent method,} \\ SGD & \text{for agents with low computational capabilities and run the stochastic gradient descent method.} \end{cases} \tag{1}$$

Without loss of generality, we have only focused on Newton's, Quasi-Newton, gradient descent, and SGD methods since these are among the most widely used techniques in iterative optimization algorithms.

Many distributed learning strategies can be considered, however we adopt in this work the diffusion method due to its stability and superior adaptation performance. Diffusion techniques have shown better performance in adaptive situations where it is essential to track drifts in the underlying models through constant step-size adaptation [50], [51].

In diffusion method, every agent over the network runs iteratively the following algorithm:

$$\begin{cases} \psi_{k,i-1} &= \sum_{l \in \mathcal{N}_k} a_{lk} \omega_{l,i-1}, \\ \omega_{k,i} &= \psi_{k,i-1} + \mu_k d_k^{\delta}(\psi_{k,i-1}) \end{cases} \tag{2}$$

where $\omega \in R^M$ is the global parameter, which all agents over the network must decide upon, ψ combines linearly the previous iterates $\omega_{l,i-1}$ received from the neighborhood $\mathcal{N}_k$ of agent k and then $\psi_{k,i-1}$ is used to update the descent direction optimization algorithm, $d_k^{\delta}(\cdot)$ is the descent direction, and μ_k is the learning rate or the step size. The coefficient a_{lk} that appears in (2) represents the weight that agent k assigns to the received iterate $\omega_{l,i-1}$ from agent l, and these weights are usually selected to satisfy the following conditions [22], [23], [24], [25], [26], [27], [28], [29]:

$$a_{lk} \geq 0, \quad \sum_{l=1}^{N} a_{lk} = 1, \quad \text{and } a_{lk} = 0 \text{ if } l \notin \mathcal{N}_k, \quad k = 1, 2, \ldots, N \tag{3}$$

where $\mathcal{N}_k$ represents the neighborhood of the agent k. The entries $\{a_{lk}\}$ can be collected into an $N \times N$ matrix A, such that the k-th column of A consists of $\{a_{lk},\ l = 1, 2, \ldots, N\}$.

The individual cost function of every agent k is defined to be $J_k(\omega)$. Every agent k holds a given cost function $J_k(\omega)$, its own data, iterative optimization algorithm and some shared parameters over the network from agents in its neighborhood. The objective is to determine the unique minimizer ω^o of the weighted aggregate cost $J^{glob}(\omega)$ that is given by

$$J^{glob}(\omega) = \sum_{k=1}^{N} \alpha_k J_k(\omega) \tag{4}$$

where N is the number of agents over the network, α_k is some positive scalar ($\alpha_k \geq 0$), and ω^o can be written as

$$\omega^o = \underset{\omega}{\text{argmin}} J^{glob}(\omega) = \underset{\omega}{\text{argmin}} \sum_{k=1}^{N} \alpha_k J_k(\omega). \tag{5}$$

Note that $d_k^{\delta}(\cdot)$ in (2) can take different forms of descent directions like Newton's direction, gradient descent, SGD, etc. We suggest the following definition for $d_k^{\delta}(\cdot)$ to accommodate all possibilities of the descent directions that may be adopted by the distributed agents:

$$d_k^{\delta}(\psi_{k,i-1}) = -Y_{k,i-1}^{\delta}\{\nabla J_k(\psi_{k,i-1}) + s_{k,i}^{\delta}(\psi_{k,i-1})\} \tag{6}$$

The matrix $Y_{k,i-1}^{\delta} \in R^{M\times M}$ and its value depends on the adopted iterative optimization algorithm as shown below

$$Y_{k,i-1}^{\delta} = \begin{cases} [\nabla^2 J_{k,i-1}(\psi_{k,i-1})]^{-1} & \text{when } \delta = NM, \\ B_{k,i-1}(\psi_{k,i-1}) & \text{when } \delta = QN, \\ I_M & \text{when } \delta = GD, \\ I_M & \text{when } \delta = SGD. \end{cases} \tag{7}$$

where $B_{k,i-1}(\psi_{k,i-1})$ is an approximation of the inverse of the Hessian matrix $\nabla^2 J_{k,i-1}(\psi_{k,i-1})]^{-1}$ as will be described in Section II-B. Computing the matrix $Y_{k,i-1}^{\delta}$ at the minimizer ω^o produces a matrix $Y_{k,o}^{\delta}$ which is defined as follows:

$$Y_{k,o}^{\delta} = \begin{cases} [\nabla^2 J_{k,i-1}(\omega^o)]^{-1} & \text{when } \delta = NM, \\ B_{k,i-1}(\omega^o) & \text{when } \delta = QN, \\ I_M & \text{when } \delta = GD, \\ I_M & \text{when } \delta = SGD. \end{cases} \tag{8}$$

The $s_{k,i}^{\delta}(\cdot)$ in (6) is called the gradient noise and it represents the difference between the full gradient $\nabla J_k(\cdot)$ and the approximated gradient $\widehat{\nabla J}_k(\cdot)$ and it is defined as

$$s_{k,i}^{\delta}(\psi_{k,i-1}) = \begin{cases} 0, \quad \text{when } \delta = NM,\ QN,\ \text{or } GD, \\ \widehat{\nabla J}_k(\psi_{k,i-1}) - \nabla J_k(\psi_{k,i-1}), \\ \text{when } \delta = SGD. \end{cases} \tag{9}$$

The gradient noise plays an important role in the stability analysis as will be shown in Section III. The approximated gradient $\widehat{\nabla J}_k(\cdot)$ may be the mini-batch-gradient descent, SGD, etc. These approximating algorithms [52] are widely used for online learning, learning with big data [53], and when agents have limited computational capabilities [54], [55], [56], [57], [58], [59], [60]. Using (6), we can write (2) as follows:

$$\begin{cases} \psi_{k,i-1} = \sum_{l\in\mathcal{N}_k} a_{lk}\omega_{l,i-1}, \\ \omega_{k,i} = \psi_{k,i-1} \\ -\mu_k Y_{k,i-1}^{\delta}\{\nabla J_k(\psi_{k,i-1}) + s_{k,i}^{\delta}(\psi_{k,i-1})\} \end{cases} \tag{10}$$

The following is an assumption about the individual and the aggregate cost functions $J_k(\omega)$ and $J^{glob}(\omega)$.

Assumption 1 (Conditions on Aggregate and Individual Costs): Assume that the individual costs, $J_k(\omega)$, are each twice-differentiable and convex, with at least one of them being ν-strongly convex (the agent k_0). It follows that $J^{glob}(\omega)$ is strongly convex and attains a global minimizer ω^o and satisfies:

$$\nabla J^{glob}(\omega^o) = \sum_{k=1}^{N} \alpha_k \nabla J_k(\omega^o) = 0 \tag{11}$$

In addition, we assume that the gradient $\nabla J_k(\omega)$ is δ_c-Lipschitz. It follows that the Hessian of the individual cost function $\nabla^2 J_k(\omega)$ and the Hessian of the aggregate cost function $\nabla^2 J^{glob}(\omega)$ satisfy the following conditions:

$$\nabla^2 J_k(\omega) \leq \delta_c I_M \tag{12}$$

$$\nu I_M \leq \nabla^2 J^{glob}(\omega) \leq \delta_c I_M \tag{13}$$

$$\nabla^2 J_{k_0}(\omega) \geq \nu I_M > 0, \quad \nabla^2 J_k(\omega) \geq 0, \quad k \neq k_0 \tag{14}$$

for some positive parameter $\nu \leq \delta_c$. ■

Let $\widetilde{\omega}_{k,i}$ and $\widetilde{\psi}_{k,i-1}$ be defined as follows:

$$\widetilde{\omega}_{k,i} = \omega^o - \omega_{k,i-1} \tag{15}$$

$$\widetilde{\psi}_{k,i-1} = \omega^o - \psi_{k,i-1} \tag{16}$$

where ω^o is the optimal minimizer of the cost function given in (4). By using the mean value theorem [61], it can be shown that $\nabla J_k(\psi_{k,i-1})$ can be written as

$$\nabla J_k(\psi_{k,i-1}) = -H_{k,i-1}\widetilde{\psi}_{k,i-1} - b_k \tag{17}$$

where b_k is the gradient of the cost function at the optimal value of the network and it is given by

$$b_k = -\nabla J_k(\omega^o) \tag{18}$$

$H_{k,i-1}$ is defined to be

$$H_{k,i-1} = \int_0^1 \nabla^2 J_k(\omega^o - t\widetilde{\psi}_{k,i-1})dt \tag{19}$$

From (12) and (19) and since each individual Hessian matrix is positive semidefinite, we get:

$$H_{k,i-1} \leq \delta_c I_M \tag{20}$$

Moreover, we deduce from (14) and (19) that

$$H_{k_o}(\omega) \geq \nu I_M > 0, \quad H_k \geq 0, \quad k \neq k_o \tag{21}$$

Subtracting ω^o from both sides of (10) and using (15), (16) and (17), then (10) can be rewritten as

$$\begin{cases} \widetilde{\psi}_{k,i-1} = \sum_{l\in\mathcal{N}_k} a_{lk}\widetilde{\omega}_{l,i-1}, \\ \widetilde{\omega}_{k,i} = \widetilde{\psi}_{k,i-1} - \mu_k Y^{\delta}_{k,i-1} H_{k,i-1}\widetilde{\psi}_{k,i-1} \\ -\mu_k Y^{\delta}_{k,i-1} b_k + \mu_k Y^{\delta}_{k,i-1} s^{\delta}_{k,i}(\psi_{k,i-1}) \end{cases} \tag{22}$$

To assess the evolution of the error dynamics across the entire network, equations (22) of all agents can be collected and written into a network-error recursion formula as follows:

$$\widetilde{\boldsymbol{\omega}}_i = \boldsymbol{\mathcal{B}}_{i-1}\widetilde{\boldsymbol{\omega}}_{i-1} + \mathcal{M}\boldsymbol{s}_i(\boldsymbol{\omega}_{i-1}) - \mathcal{M}b \tag{23}$$

where

$$\widetilde{\boldsymbol{\omega}}_i = col\{\widetilde{\boldsymbol{\omega}}_{1,i},\ \widetilde{\boldsymbol{\omega}}_{2,i},\ \ldots,\ \widetilde{\boldsymbol{\omega}}_{N,i}\} \tag{24}$$

$$\boldsymbol{\mathcal{B}}_{i-1} = (\mathcal{I} - \mathcal{M}\boldsymbol{\mathcal{H}'}_{i-1})\mathcal{A}^T \tag{25}$$

$$\mathcal{I} = I_N \otimes I_M \tag{26}$$

$$\mathcal{A} = A \otimes I_M \tag{27}$$

where the convention *col* is used to represent a column vector, A is a left stochastic matrix that includes the weights $\{a_{lk}\}$ over the edges of the network, and the term $\otimes$ is the Kronecker product. Note that the weights α_k that are used in (4) are chosen to be

$$\alpha_k = \mu_k p_k \tag{28}$$

where p_k represents the k-th entry of the Perron eigenvector of A ($Ap = p,\ \mathbb{1}^T p = 1$).

$$\mathcal{M} = \text{diag}\{\mu_1 I_M, \mu_2 I_M, \ldots, \mu_N I_M\} \tag{29}$$

$$\boldsymbol{s}_i(\boldsymbol{\omega}_{i-1}) = col\{Y^{\delta}_{k,i-1}s^{\delta}_{1,i}(\psi_{1,i-1}), \ldots, Y^{\delta}_{k,i-1}s^{\delta}_{N,i}(\psi_{N,i-1})\} \tag{30}$$

$$b = col\{Y^{\delta}_{1,o}b_1,\ Y^{\delta}_{2,o}b_2, \ldots, Y^{\delta}_{N,o}b_N\} \tag{31}$$

$$\boldsymbol{\mathcal{H}'}_{i-1} = \text{diag}\{\boldsymbol{H'}^{\delta}_{1,i-1}, \boldsymbol{H'}^{\delta}_{2,i-1}, \ldots, \boldsymbol{H'}^{\delta}_{N,i-1}\} \tag{32}$$

$$\boldsymbol{H'}^{\delta}_{k,i-1} = Y^{\delta}_{k,i-1}H_{k,i-1} \tag{33}$$

According to (1), different forms can be taken by $d^{\delta}_k(\psi_{k,i-1})$ and $Y^{\delta}_{k,i-1}$ as will be shown in the following sections. In addition, we state the lower and upper bounds of $Y^{\delta}_{k,i-1}$, which will be useful in the remaining part of the paper.

A. DISTRIBUTED AGENTS UNDER NEWTON'S METHOD

When a subset of agents run the Newton's method, then the descent direction $d^{\delta=NM}_k(\psi_{k,i-1})$ in (2) becomes [62], [63]

$$d^{\delta=NM}_k(\psi_{k,i-1}) = -Y^{\delta=NM}_{k,i-1}\nabla J_{k,i-1}(\cdot) \tag{34}$$

where $s^{\delta=NM}_{k,i}(\psi_{k,i-1})$ is equal to zero and $Y^{\delta=NM}_{k,i-1}$ in (10) is given in this case by

$$Y^{\delta=NM}_{k,i-1} = [\nabla^2 J_{k,i-1}(\psi_{k,i-1})]^{-1} \tag{35}$$

and $[\nabla^2 J_{k,i-1}(\cdot)]^{-1}$ is the inverse of the Hessian matrix at every iteration. The induced 2-norm of $Y^{\delta=NM}_{k,i-1}$ can be upper bounded as follows

$$\|Y^{\delta=NM}_{k,i-1}\| = \|[\nabla^2 J_{k,i-1}(\psi_{k,i-1})]^{-1}\|$$

$$\overset{(a)}{\leq} 2\|[\nabla^2 J_k(\omega^o_k)]^{-1}\| = \widetilde{L}_k \tag{36}$$

where $\widetilde{L}_k$ is some positive number and (a) follows since $\nabla^2 J_k(\cdot)$ is assumed to be positive definite at every iteration, then $\nabla^2 J_k(\omega^o_k)$ is non-singular and thus there is a radius $r > 0$ such that $\|[\nabla^2 J_k(\psi_{k,i-1})]^{-1}\| \leq 2\|[\nabla^2 J_k(\omega^o_k)]^{-1}\|$ for all ω_k with $\|\psi_{k,i-1} - \omega^o_k\| \leq r$ [64]. Note that if $[\nabla^2 J_k(\psi_{k,i-1})]^{-1}$ is not positive definite at every iteration, then it can be replaced by related positive-definite matrix like modified Hessian matrices to guarantee that condition [62], [63], [64], [65]. This means that $Y^{\delta=NM}_{k,i-1}$ can be upper bounded as follows

$$Y^{\delta=NM}_{k,i-1} \leq c_2 I_M \tag{37}$$

where c_2 is some positive number. In addition, since $\nabla^2 J_{k,i-1} \leq \delta_c I_M$, and based on (35) we get

$$Y^{\delta=NM}_{k,i-1} \geq \frac{1}{\delta_c} I_M = c_1 I_M \tag{38}$$

where c_1 is some positive number. Consequently

$$c_1 I_M \leq Y^{\delta=NM}_{k,i-1} \leq c_2 I_M \tag{39}$$

B. DISTRIBUTED AGENTS UNDER QUASI-NEWTON METHOD

Quasi-Newton method is an approximation technique for the Newton's method and it can compute efficiently the inverse of the Hessian matrix by relying only on the gradient of the cost function [64], [65], [66]. There are many different versions of Quasi-Newton methods (Broyden-Fletcher-Goldfarb-Shanno (BFGS), Davidon-Fletcher-Powell (DFP), etc), but they are all based on approximating the inverse of the Hessian $[\nabla^2 J_{k,i}(\psi_{k,i-1})]^{-1}$ by another matrix $B_{k,i}(\psi_{k,i-1})$ which may take different forms depending on the update formula. For instance, the BFGS method computes $B_{k,i}(\psi_{k,i-1})$ as follows

$$B_{k,i} = B_{k,i-1} - \frac{(B_{k,i-1}s_{k,i-1})(B_{k,i-1}s_{k,i-1})^T}{s^T_{k,i-1}B_{k,i-1}s_{k,i-1}} + \frac{y_{k,i-1}y^T_{k,i-1}}{y^T_{k,i-1}s_{k,i-1}} \tag{40}$$

where $y_{k,i-1}$ and $s_{k,i-1}$ are given by

$$y_{k,i-1} = \nabla J_{k,i}(\psi_{k,i}) - \nabla J_{k,i-1}(\psi_{k,i-1}) \tag{41}$$

$$s_{k,i-1} = \psi_{k,i} - \psi_{k,i-1} \tag{42}$$

So, $Y^{\delta=QN}_{k,i-1}$ usually takes the following form under Quasi-Newton method:

$$Y^{\delta=QN}_{k,i-1} = B_{k,i-1} \tag{43}$$

Then the descent direction $d^{\delta=QN}_k(\psi_{k,i-1})$ in (2) becomes

$$d^{\delta=QN}_k(\psi_{k,i-1}) = -B_{k,i-1}\nabla J_{k,i-1}(\cdot) \tag{44}$$

Note that $s^{\delta=QN}_{k,i}(\psi_{k,i-1})$ is equal to zero, and since B_k is designed to be symmetric and positive definite at every iteration, then $Y^{\delta=QN}_{k,i-1}$ can be bounded as follows:

$$c_{lq} I_M \leq Y^{\delta=QN}_{k,i-1} \leq c_{uq} I_M \tag{45}$$

where c_{lq} and c_{uq} are some positive numbers ($c_{lq} < c_{uq}$).

C. DISTRIBUTED AGENTS UNDER GRADIENT DESCENT METHOD

When the gradient descent is being implemented by any of the distributed agents, then the $s_{k,i}^{\delta=GD}(\psi_{k,i-1})$ is equal to zero, and the $Y_{k,i-1}^{\delta=GD}$ in (10) is given in this case by

$$Y_{k,i-1}^{\delta=GD} = I_M \tag{46}$$

The descent direction $d_k^{\delta=GD}(\psi_{k,i-1})$ in (2) becomes

$$d_k^{\delta=GD}(\psi_{k,i-1}) = -\nabla J_{k,i-1}(\psi_{k,i-1}) \tag{47}$$

The induced 2-norm of $Y_{k,i-1}^{\delta=GD}$ is given by

$$\|Y_{k,i-1}^{\delta=GD}\| = 1 \tag{48}$$

D. DISTRIBUTED AGENTS UNDER STOCHASTIC GRADIENT DESCENT METHOD

Stochastic Gradient Descent (*SGD*) is an approximation algorithm of GD that requires low computational complexity [53], [55], [58], [59], [60], [67], [68], however it introduces non-zero gradient noise $s_{k,i}^{\delta=SG}(\psi_{k,i-1})$. The $Y_{k,i-1}^{\delta=SG}$ in (10) is given in this case by

$$Y_{k,i-1}^{\delta=SGD}(\psi_{k,i-1}) = I_M \tag{49}$$

Then, the descent direction $d_k^{\delta=SGD}(\psi_{k,i-1})$ in (2) for agents which implement the SGD is given by

$$d_k^{\delta=SGD}(\psi_{k,i-1}) = -[\nabla J_k(\psi_{k,i-1}) + s_{k,i}^{\delta=SGD}(\psi_{k,i-1})] \tag{50}$$

The induced 2-norm of $Y_{k,i-1}^{s}$ is given by

$$\|Y_{k,i-1}^{\delta=SGD}\| = 1 \tag{51}$$

E. UNIFYING BOUNDS OF $Y_{K,I-1}^{\delta}$

We capture in this section the various bounds of $Y_{k,i-1}^{\delta}$ by a single unifying description under different iterative optimization strategies. We conclude from (39), (45), (46), and (49) that $Y_{k,i-1}^{\delta}$ is bounded as follows

$$\begin{aligned} c_1' I_M &= \min\{1, c_1, c_{lq}\} I_M \le Y_{k,i-1}^{\delta} \le \max\{1, c_2, c_{uq}\} I_M \\ &= c_2' I_M \end{aligned} \tag{52}$$

where c_1' and c_2' are defined to be equal to $\min\{1, c_1, c_{lq}\}$ and $\max\{1, c_2, c_{uq}\}$, respectively. In addition, we conclude from (36), (45), (48), and (51) that

$$\|Y_{k,i-1}^{\delta}\| \le max\{\widetilde{L}_k, c_{uq}, 1\} = \widetilde{L}_{max} \tag{53}$$

There are still four important terms which include the matrix $Y_{k,i-1}^{\delta}$ and need to be bounded. The first term is $\|I - \boldsymbol{H}_{k,i-1}'^{\delta}\|$, while the remaining ones are the first, second, and the fourth moments of $Y_{k,i-1}^{\delta} s_{k,i}^{\delta}(\psi_{k,i-1})$. The upper bound of $\|I - \boldsymbol{H}_{k,i-1}'^{\delta}\|$ is computed below

$$\begin{aligned} &\|I - \boldsymbol{H}_{k,i-1}'^{\delta}\| = \|I - Y_{k,i-1}^{\delta} H_{k,i-1}\| \\ &= \|Y_{k,i-1}^{\delta}((Y_{k,i-1}^{\delta})^{-1} - H_{k,i-1})\| \\ &\le \|Y_{k,i-1}^{\delta}\| \|((Y_{k,i-1}^{\delta})^{-1} - H_{k,i-1})\| \\ &\le \|Y_{k,i-1}^{\delta}\| \|(\nabla^2 J_k(\psi_{k,i-1}) - H_{k,i-1})\| \\ &= \|Y_{k,i-1}^{\delta}\| \\ &* \| \int_0^1 \nabla^2 J_k(\psi_{k,i-1}) - \nabla^2 J_k(\omega^o - t\widetilde{\psi}_{k,i-1}) dt \| \\ &\le L_{max} \\ &* \int_0^1 \|\nabla^2 J_k(\psi_{k,i-1}) - \nabla^2 J_k(\omega^o - t\widetilde{\psi}_{k,i-1})\| dt \\ &\le L_{max} \int_0^1 \mathcal{K}_d' \|\psi_{k,i-1} - \omega^o + t\widetilde{\psi}_{k,i-1}\| dt \\ &= L_{max} \mathcal{K}_d' \int_0^1 \|\widetilde{\psi}_{k,i-1}(t-1)\| dt \\ &= L_{max} \mathcal{K}_d' C \|\widetilde{\psi}_{k,i-1}\| \\ &= L_{max} \mathcal{K}_d' C \| \sum_{l \in \mathcal{N}_k} \widetilde{\omega}_{l,i-1} \| \\ &\le L_{max} \mathcal{K}_d' C N \|\widetilde{\omega}_{i-1}\| \end{aligned} \tag{54}$$

where $\mathcal{K}_d'$ is a positive scalar that follows from the Lipschitz condition [16] and C is some positive constant.

■

Now, regarding the moments of $Y_{k,i-1}^{\delta} s_{k,i}^{\delta}(\psi_{k,i-1})$, it is useful to recall that the gradient noise is non-zero only when the distributed agents are applying the SGD. The following is an important assumption about the gradient noise for agents that apply the SGD.

Lemma 1 (Moments of Gradient Noise): The gradient noise process $s_{k,i}^{\delta=SGD}(\psi_{k,i-1})$ has non-zero value over agents that implement the SGD as given in (9). In addition, their matrix $Y_{k,i-1}^{\delta=SGD} = I_M$ is given in (49). Then, the moments of the gradient noise in [16] remain unchanged for the term $Y_{k,i-1}^{\delta=SGD} s_{k,i}^{\delta=SGD}(\psi_{k,i-1})$ and thus for any $\omega \in \mathcal{F}_{i-1}$ and for all $k = 1, 2, \ldots, N$, we have

$$\mathbf{E}[Y_{k,i-1}^{\delta=SGD} s_{k,i}^{\delta=SGD}(\psi_{k,i-1}) | \mathcal{F}_{i-1}] = 0 \tag{55}$$

$$\mathbf{E}[\|Y_{k,i-1}^{\delta=SGD} s_{k,i}^{\delta=SGD}(\psi_{k,i-1})\|^2 | \mathcal{F}_{i-1}] \le \beta_k^2 \|\widetilde{\psi}_{k,i-1}\|^2 + \sigma_{s,k}^2 \tag{56}$$

$$\mathbf{E}[Y_{k,i-1}^{\delta=SGD} s_{k,i}^{\delta=SGD}(\psi_{k,i-1})(s_{l,i}^{\delta=SGD})^T(\psi_{k,i-1}) | \mathcal{F}_{i-1}] = 0, \quad k \ne l \tag{57}$$

$$\mathbf{E}[\|s_{k,i}^{\delta=SGD}(\psi_{k,i-1})\|^4 | \mathcal{F}_{i-1}] \le \beta_k^4 \|\widetilde{\psi}_{k,i-1}\|^4 + \sigma_{s,k}^4 \tag{58}$$

for some $\beta_k^2 \ge 0$, $\sigma_{s,k}^2 \ge 0$, and $\mathcal{F}_{i-1}$ represents the set of the previous iterates $\{\omega_{i-1}\}$.

■

As explained in [16], these conditions are automatically satisfied in many circumstances of interest in learning and adaptation. Condition (55) states that the gradient noise is unbiased conditioned on the past iterates. Condition (56) states that the second-order moment of the gradient noise is bounded by the squared norm of the iterates. Condition (57) shows that the gradient noises across the agents are uncorrelated.

F. COMPUTATIONAL COMPLEXITY

It is noteworthy to mention the computational complexity of the different optimization methods adopted by the distributed

TABLE 1. Computational complexity for different optimization algorithms.

Algorithm	Computational Complexity
Newton's method	$\mathcal{O}(\text{ixL}^3)$
Quasi-Newton	$\mathcal{O}(\text{ixL}^2)$
Gradient Descent	$\mathcal{O}(\text{ixL})$
Stochastic Gradient Descent	$\mathcal{O}(\text{i})$

agents. The objective function $J_k(\omega_k)$ is usually expressed as a summation of loss function $Q_k(\omega_k)$ as follows:

$$J(\omega_k) = \frac{1}{L}\sum_{l=0}^{L-1} Q_k(\omega_k, \gamma_k(l), h_k(l)), \quad h_k \in R^m, \gamma_k(l) \in R. \tag{59}$$

where L, h_k, and γ_k are the size of the data-set, one sample of the data-set, and the class of the samples of the data-set respectively. The computational complexity for i iterations for Newton's method is $\mathcal{O}(i \times L^3)$ [69], while the quasi-Newton method has a computational complexity of $\mathcal{O}(i \times L^2)$ [70]. The gradient descent based methods have lower convergence rates than Newton's based methods, but they require lower computational complexity. The GD has a computational complexity of $\mathcal{O}(i \times L)$ [64], [69], while the SGD computational complexity is $\mathcal{O}(i)$ [60], [69]. Table 1 summarizes the computational complexity for the adopted optimization algorithms.

III. STABILITY ANALYSIS

Using the results of the previous section, we examine the stability of the mean-error process $\|\mathbf{E}\,\widetilde{\boldsymbol{\omega}}_i\|$, the mean-square-error $\mathbf{E}\|\widetilde{\boldsymbol{\omega}}_i\|^2$ and the fourth-order moment $\mathbf{E}\|\widetilde{\boldsymbol{\omega}}_i\|^4$ of the distributed strategy in (23).

A. STABILITY OF FIRST, SECOND AND FOURTH ORDER ERROR MOMENTS

In this section, we study how well the distributed strategies in (22) and (23) approach the optimal solution ω^o of the global cost in (4). This can be measured by the first, second and fourth order moments of the error between the iterate $\omega_{k,i}$ and the optimal point ω^o. The following theorems are introduced in this section, and the proofs are moved to the appendices.

1) THEOREM 1: NETWORK MEAN-SQUARE-ERROR STABILITY

Assume that the aggregate cost $J^{glob}(\omega)$, the individual costs $J_k(\omega)$, and the network topology satisfy the conditions in **Assumption 1** and **Lemma 1**. Then, the network with heterogeneous agents is mean-square stable for sufficiently small step-sizes, specifically, it holds that

$$\limsup_{i\to\infty} \mathbf{E}\|\widetilde{\omega}_{k,i}\|^2 = O(\mu_{max}), \text{ for } k = 1, 2, \ldots, N \tag{60}$$

for any $\mu_{max} < \mu_o$, for some small enough μ_o.

Proof: See Appendix B.

2) THEOREM 2: NETWORK FOURTH-ORDER MOMENT STABILITY

Assume that the aggregate cost $J^{glob}(\omega)$, the individual costs $J_k(\omega)$, and the network topology satisfy the conditions in **Assumption 1** and **Lemma 1**. Then, the fourth-order moment, $\mathbf{E}\|\widetilde{\omega}_{k,i}\|^4$, of the network with heterogeneous agents is stable for sufficiently small step-sizes, specifically, it holds that

$$\limsup_{i\to\infty} \mathbf{E}\|\widetilde{\omega}_{k,i}\|^4 = O(\mu_{max}^2), \text{ for } k = 1, 2, \ldots, N \tag{61}$$

for any $\mu_{max} < \mu_o$, for some small enough μ_o.

Proof: See Appendix C.

3) THEOREM 3: NETWORK MEAN-ERROR STABILITY

Assume that the aggregate cost $J^{glob}(\omega)$, the individual costs $J_k(\omega)$, and the network topology satisfy the conditions in **Assumption 1** and **Lemma 1**. Then, the first-order moment, $\|\mathbf{E}\widetilde{\omega}_{k,i}\|$, of the network with heterogeneous agents are stable for sufficiently small step-sizes, specifically, it holds that

$$\limsup_{i\to\infty} \|\mathbf{E}\widetilde{\omega}_{k,i}\| = O(\mu_{max}), \text{ for } k = 1, 2, \ldots, N \tag{62}$$

for any $\mu_{max} < \mu_o$, for some small enough μ_o.

Proof: See Appendix D.

IV. SIMULATION RESULTS

In this section, we illustrate the performance of the proposed work in comparison with existing algorithms that assume the homogeneous computational capabilities among the dispersed agents. Figure 1 presents an example of a strongly connected network of $N = 10$ agents. Each agent owns a regularized multinomial logistic cost function $J_k(\omega)$ that is given by

$$J_k(\omega_k) = \frac{\rho}{2}\|\omega_k\|^2 - \frac{1}{L}\sum_{n=0}^{L-1}[\omega_{k,\gamma_k(n)}^T h_k(n) - \log(\sum_{l=1}^{K} e^{\omega_{k,l}^T h_k(n)})] \tag{63}$$

where L, $\gamma_k(n) \in R$, $h_k(n) \in R^M$ and ρ represent the size of the data-set, the n^{th} label, the n^{th} data sample and the regularization parameter, respectively. The gradient and the Hessian matrix of (63) can be found in [71]. The objective of every agent is to minimize (63) in a distributed manner. All agents are assumed to have the same cost function $J_k(\omega)$. We choose $\alpha_k = 1$ in (28) for all agents.

Two metrics are used to assess the performance of the proposed algorithm. The first metric measures the mean-square-deviation (MSD) at agent k and a set of agents $\mathcal{S}$ using the following formulas

$$MSD_k = \mathbf{E}\|\widetilde{\omega}_{k,i}\|^2 \tag{64}$$

$$MSD_{\mathcal{S}} = \frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}} MSD_k \tag{65}$$

where $|\mathcal{S}|$ is the size of set $\mathcal{S}$. The second metric measures the excess risk (ER) at every agent k and a set of agents $\mathcal{S}$.

ER represents the average fluctuation of $J^{glob}(\omega)$ around its minimum value $J^{glob}(\omega^o)$. ER is defined as

$$ER_k = \mathbf{E}\{J^{glob}(\omega_{k,i-1}) - J^{glob}(\omega^o)\} \tag{66}$$

$$ER_{\mathcal{S}} = \frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}} ER_k \tag{67}$$

A. KEY PARAMETERS

The algorithm in (2) entails several parameters (step-size μ_k and weight matrix A) to be predetermined. A constant step-size $\mu_k = 10^{-3}$ is assumed for agents running SGD, and a variant μ_k using a backtracking line search method based on Armijo condition [64], [65] is selected for agents running Newton's method or the full gradient descent method. On the other hand, the weights of matrix A are chosen as

$$a_{lk} = \begin{cases} \dfrac{1}{|\mathcal{N}_k|}, & \text{if } l \in \mathcal{N}_k \\ 0, & \text{otherwise} \end{cases} \tag{68}$$

where $|\mathcal{N}_k|$ denotes the degree of the agent k, which is equal to the size of its neighborhood.

B. DATASET

The MNIST dataset [72] is used to evaluate the performance of the proposed work. The MNIST dataset is a collection of handwritten digit images used extensively in pattern recognition and machine learning research. It has a training set of 60,000 examples, and a test set of 10,000 examples. The digits have been size-normalized and centered in fixed-size images.

C. IMPACT ON AGENTS WITH HIGH COMPUTATIONAL CAPABILITIES

The proposed distributed optimization for learning over networks has an imperative impact on both capable and non-capable agents. The convergence rate of agents with low computational capabilities will increase after its cooperation with agents with high computational capabilities, whereas the capable agents will undergo a decrease in their rate of convergence. To assess the impact of the proposed work on the capable agents, six scenarios are considered:

- Scenarios 1, 2, and 3: All agents implement Newton's method, Quasi-Newton method, and GD method per respective scenario.
- Scenarios 4, 5, and 6: Agents three and eight implement Newton's method, Quasi-Newton method, and GD method per respective scenario, while other agents run SGD.

Agents three and eight in the network of Figure 1 are assumed to own high computational capabilities. So, we will evaluate the impact of such high computational capabilities on the remaining agents over the network. Figures 2 to 4 present the normalized curves for $MSD_{\{3,8\}} = \frac{1}{2}(MSD_3 + MSD_8)$ in (65) for agents three and eight in the aforementioned six scenarios. The curves are attained by running the distributed strategy in (2) and averaging the

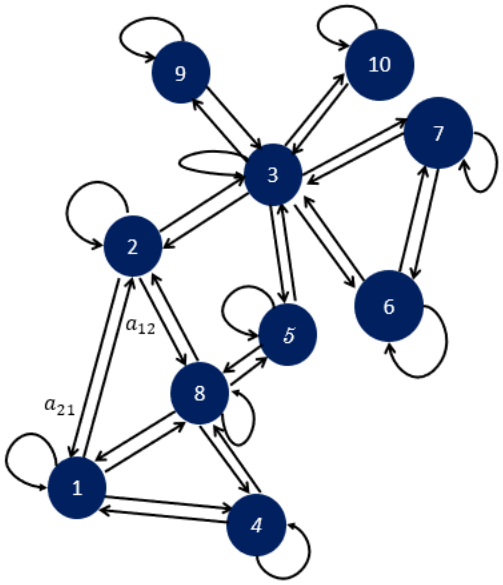


FIGURE 1. A strongly connected network with $N = 10$ agents which own heterogeneous computational capabilities. Subset of the agents can run the Newton's methods, or the GD, and the remaining ones can only implement the SGD.

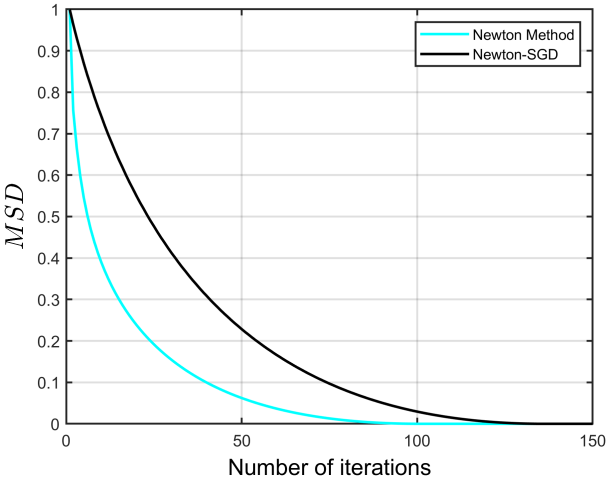


FIGURE 2. Impact of heterogeneous distributed optimization algorithms on capable agents running Newton's method.

trajectories of each agent over 200 repeated experiments. The following conclusions can be derived from Figures 2 to 4:

- The learning curves for capable agents converge to a stationary point and attain a stable solution, which proves that the cooperation with nodes which implement SGD does not affect the stability of Newton's method, Quasi-Newton method, or GD method.
- The learning curves for capable agents converge faster in homogeneous networks than heterogeneous networks. This shows that the performance of agents running Newton's method, Quasi-Newton method, or GD method degrades upon cooperating with agents running SGD. However, this degradation at the capable agents is acceptable since it is outweighed by the performance improvement of the other agents as will be shown in the next section.

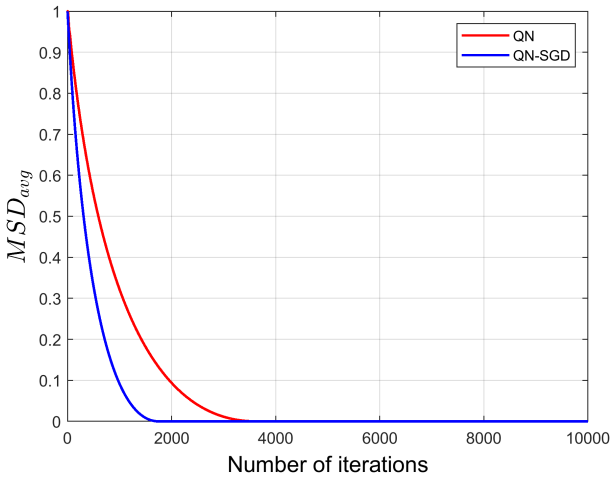


FIGURE 3. Impact of heterogeneous distributed optimization algorithms on capable agents running Quasi-Newton method.

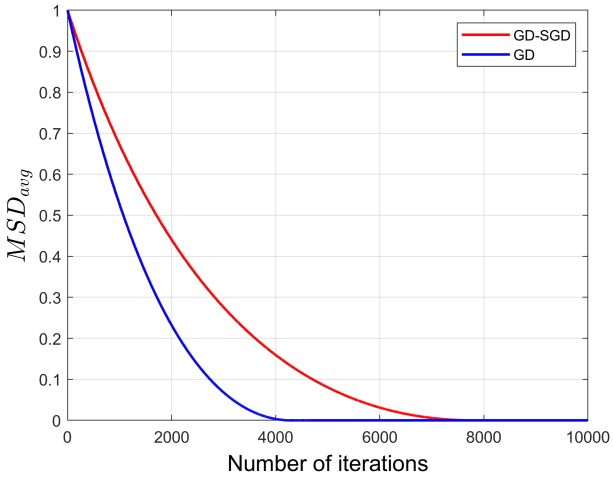


FIGURE 4. Impact of heterogeneous distributed optimization algorithms on capable agents running the GD method.

D. IMPACT ON AGENTS WITH LOW COMPUTATIONAL CAPABILITIES

This section shows the impact of cooperative learning in heterogeneous networks on agents with low computational capabilities. Four scenarios are considered:

- Scenarios 4, 5 and 6 of Section IV-C.
- Scenario 7: All agents implement SGD.

Figures 5 and 6 present the normalized curves for non-capable agents (all agents except three and eight) for the aforementioned scenarios. Figure 5 presents the evolution of the ensemble-average learning curves $MSD_{\{1-10\}\backslash\{3,8\}}$ in (65) for the incapable agents using the distributed learning method in (2). The following conclusions can be made:

- The proposed heterogeneous learning method converges to a stationary point and attains a stable solution, which proves that the cooperation with nodes who implement Newton's method, Quasi-Newton method, or GD method does not affect the stability of the SGD method.
- The convergence rate of the proposed work is much faster than learning in a homogeneous network with all agents running the SGD. For instance, convergence in hybrid-learning Scenarios 4 to 6 is attained within 150 and 7000 iterations respectively, whereas homogeneous-learning Scenario 7 requires more than 10000 iterations per agent to converge.
- The proposed hybrid method significantly reduces the communication overhead among the learning agents and the overall network power consumption during learning due to the faster convergence rate.

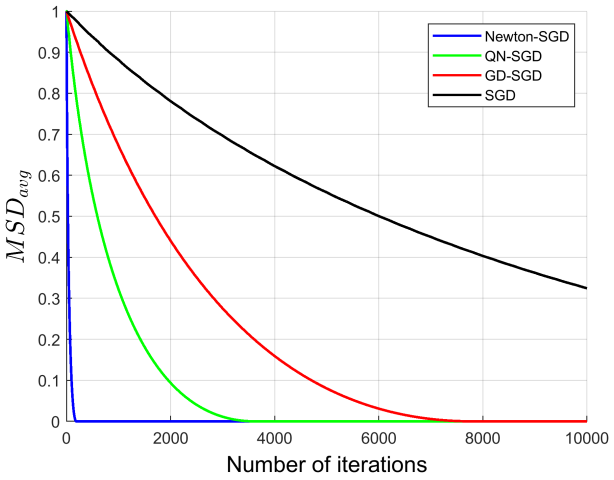


FIGURE 5. Evolution of the mean-square deviation learning curves for incapable agents in homogeneous and heterogeneous networks using the diffusion learning strategy. Incapable agents use step-size $\mu = 10^{-3}$.

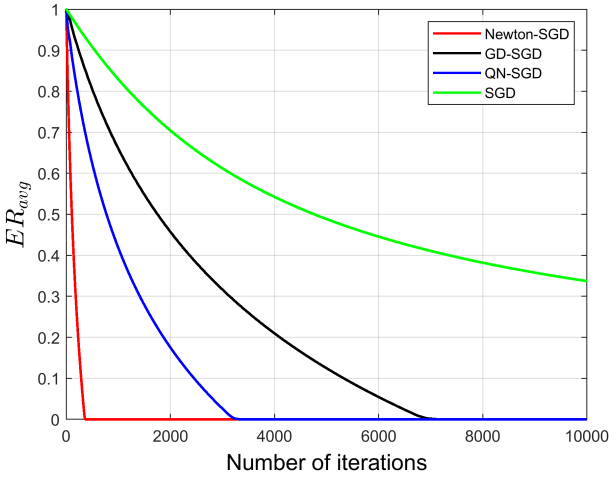


FIGURE 6. Evolution of the empirical risk learning curves for incapable agents in homogeneous and heterogeneous networks using the diffusion learning strategy. Incapable agents use step-size $\mu = 10^{-3}$.

Figure 6 presents the evolution of the ensemble-average learning curves $ER_{\{1-10\}\backslash\{3,8\}}$ in (67) for the incapable agents using the distributed learning method in (2). Figure 6 shows that there is a large gap between the ER curves of the incapable agents in the homogeneous network of Scenario 7 and those in the heterogeneous network of Scenarios 4 to 6. Again, this clearly illustrates that with hybrid learning convergence is achieved much faster at the incapable agents due to cooperation with the capable agents.

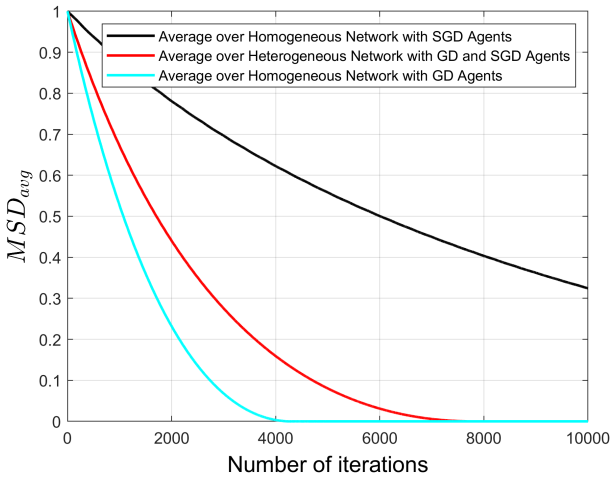


FIGURE 7. Evolution of the average mean-square deviation learning curves over all agents of Figure 1 in homogeneous and heterogeneous scenarios.

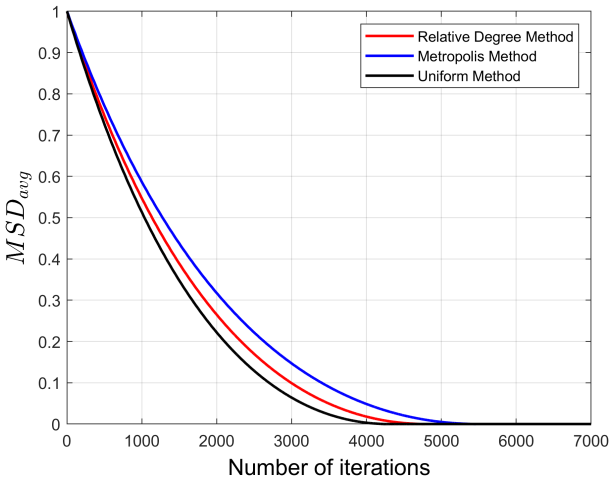


FIGURE 8. Evolution of the average mean-square deviation learning curves over the agents of Figure 1 for different combination policies. Agents three and eight run Newton's method, while other agents run SGD.

E. OVERALL PERFORMANCE

Figure 7 illustrates $MSD_{\{1-10\}}$ over all the network agents of Figure 1 in Scenarios 3, 6 and 7 of Sections IV-C and IV-D. The figure shows that network performance in Scenario 6 (GD+SGD) is much closer to that of Scenario 3 (all GD) than that of Scenario 7 (all SGD). This shows that the performance improvement of the incapable agents in Section IV-D outweighs the performance reduction of the capable agents in Section IV-C. This further shows that it is sufficient to selectively deploy few capable agents in a network to approximate the performance of a fully capable network through hybrid learning.

F. IMPACT OF THE COMBINATION WEIGHTS ON THE PERFORMANCE

The weights over the edges of the network play a central role in stability, convergence and performance of the distributed optimization algorithms [16]. There are many different algorithms in literature on the optimal selection of the combination weights over the edges of the network [16]. In this part of the paper, we show experimentally that the weights over the edges of the network affect also the distributed learning among agents with heterogeneous computational capabilities. Figure 8 illustrates the impact of the choice of the combination weights $\{a_{lk}\}$ in (2) on the performance of the hybrid distributed learning method. Referring to Scenario 4 in Section IV-C, Figure 8 shows the evolution of $MSD_{\{1-10\}}$ for three choices of $\{a_{lk}\}$:

- Uniform combination weights [16] as in (68)
- Combination weights based on relative degree [26]:

$$a_{lk} = \begin{cases} \dfrac{n_l}{\sum\limits_{m\in\mathcal{N}_k} n_m}, & \text{if } k \text{ and } l \text{ are neighbors or } k = l, \\ 0, & \text{otherwise} \end{cases} \tag{69}$$

- Combination weights using the metropolis method [22]:

$$a_{lk} = \begin{cases} \dfrac{1}{max(n_l, n_k)}, & \text{if } k \neq l \text{ are neighbors,} \\ 1 - \sum\limits_{l\in\mathcal{N}_k\setminus\{k\}} a_{lk}, & k = l, \\ 0, & \text{otherwise} \end{cases} \tag{70}$$

As expected, the convergence rate changes based on the combination strategy. The problem of optimally selecting combination weights $\{a_{lk}\}$ in hybrid learning sceharios is an element for future study.

In summary, learning over heterogeneous networks allows agents with limited computational capabilities to run simple learning methods such as SGD but still enjoy faster convergence based on cooperation with capable agents implementing more advanced techniques. This helps incapable agents to both perform better and save power. Additionally, the faster convergence reduces the communication overhead for parameter exchange in cooperative learning.

V. CONCLUSION

This work studied the distributed learning problem among network agents with heterogeneous computational capabilities. We proposed an iterative learning method where a subset of incapable agents run SGD, while other capable agents run computationally demanding techniques such as Newton's method or full GD. Theoretical and simulation results show that for small step-size parameter μ, the distributed learning based on cooperation between the two types of agents is stable and converges faster than learning algorithms in homogeneous networks. As part of the future work, we will study the convergence analysis, and the trade-off between computational and communication cost for the distributed learning among agents with heterogeneous computational capabilities.

APPENDIX A
ANALYSIS OF THE MATRIX $\mathcal{B}_{i-1}$

Matrix $\boldsymbol{\mathcal{B}}_{i-1}$ in (25) plays an important role in the stability and the convergence analysis of (22). In this section, we will present a decomposition and some upper bounds for the elements of the matrix $\boldsymbol{\mathcal{B}}_{i-1}$ which will be useful for the upcoming proofs in appendices B, C and D. A is a left stochastic matrix and assumed to be primitive. The Jordan canonical decomposition of the $N \times N$ matrix A is given by

$$A_1 = V_\epsilon J V_\epsilon^{-1} \tag{71}$$

where J, V_ϵ, and V_ϵ^{-1} have dimensions of $N \times N$ and they are given by

$$J = \begin{pmatrix} 1 & 0 \\ 0 & J_\epsilon \end{pmatrix} \tag{72}$$

$$V_\epsilon = \begin{pmatrix} p & V_R \end{pmatrix} \tag{73}$$

$$V_\epsilon^{-1} = \begin{pmatrix} \mathbb{1}^T \\ V_L^T \end{pmatrix} \tag{74}$$

where the matrices V_L, J_ϵ, V_R have dimensions of $(N-1) \times (N-1)$. Since $V_\epsilon^{-1} V_\epsilon = I_N$, then it holds that,

$$\begin{cases} \mathbb{1}^T V_R = \mathbf{0}^T \\ V_L^T p = \mathbf{0} \\ V_L^T V_R = I_{N-1} \end{cases} \tag{75}$$

Substituting (71) into (25), we get

$$\boldsymbol{\mathcal{B}}_{i-1} = ((V_\epsilon^{-1})^T \otimes I_M)\{(J^T \otimes I_M) - \boldsymbol{\mathcal{D}}_{i-1}^T\}(V_\epsilon^T \otimes I_M) \tag{76}$$

where $\boldsymbol{\mathcal{D}}_{i-1}^T$ is given by

$$\begin{aligned} \boldsymbol{\mathcal{D}}_{i-1}^T &= ((V_\epsilon)^T \otimes I_M)\mathcal{M}\boldsymbol{\mathcal{H}'}_{i-1}\mathcal{A}^T((V_\epsilon^{-1})^T \otimes I_M) \\ &= \begin{pmatrix} \boldsymbol{D}_{11,i-1}^T & \boldsymbol{D}_{21,i-1}^T \\ \boldsymbol{D}_{12,i-1}^T & \boldsymbol{D}_{22,i-1}^T \end{pmatrix} \end{aligned} \tag{77}$$

The objective of this section is to find the upper bounds of the terms $\|\boldsymbol{D}_{11,i-1}^T\|$, $\|\boldsymbol{D}_{12,i-1}^T\|$, $\|\boldsymbol{D}_{21,i-1}^T\|$, and $\|\boldsymbol{D}_{22,i-1}^T\|$ of the matrix $\boldsymbol{\mathcal{B}}_{i-1}$. Substituting (73), (74) and (75) in (77), we get

$$\boldsymbol{D}_{11,i-1} = \sum_{k=1}^{N} \mu_k p_k \boldsymbol{H'}_{k\delta,i-1}^T \tag{78}$$

$$\boldsymbol{D}_{12,i-1} = (\mathbb{1}^T \otimes I_M)\boldsymbol{\mathcal{H}}_{i-1}'^T \mathcal{M}(V_R \otimes I_M) \tag{79}$$

$$\boldsymbol{D}_{21,i-1} = (V_L^T A \otimes I_M)\boldsymbol{\mathcal{H}}_{i-1}'^T \mathcal{M}(p \otimes I_M) \tag{80}$$

$$\boldsymbol{D}_{22,i-1} = (V_L^T A \otimes I_M)\boldsymbol{\mathcal{H}}_{i-1}'^T \mathcal{M}(V_R \otimes I_M) \tag{81}$$

Let μ_k be defined as

$$\mu_k = \tau_k \mu_{max} \tag{82}$$

where τ_k is some positive scalar, then the matrix sequence $\boldsymbol{D_{11,i-1}}$ can be upper bounded as follows

$$\begin{aligned} \boldsymbol{D}_{11,i-1} &= \sum_{k=1}^{N} \mu_k p_k Y_{k,i-1}^{\delta} \boldsymbol{H}_{k,i-1}^{\delta T} \\ &\overset{(a)}{\leq} p_{max}\tau_{k_{max}}\mu_{max} N c_2' \delta_c I_M \\ &= c_2'' \mu_{max} I_M \\ &= O(\mu_{max}) \end{aligned} \tag{83}$$

where (a) follows from (20) and (52), c_2' is defined in (52), μ_{max} and p_{max} represent the maximum learning rate and the maximum entry of the vector p, respectively, while c_2'' is a constant that is equal to $p_{max}\tau_{k_{max}} N c_2' \delta_c$.

The lower bound of $D_{11,i-1}$ can be derived as follows

$$\begin{aligned} \boldsymbol{D}_{11,i-1} &= \sum_{k=1}^{N} \mu_k p_k Y_{k,i-1}^{\delta} \boldsymbol{H}_{k,i-1}^{\delta T} \\ &\overset{(a)}{\geq} c_1' p_{k_o} \tau_{k_o} \mu_{max} \nu I_M \\ &= c_1'' \mu_{max} I_M \\ &= O(\mu_{max}) \end{aligned} \tag{84}$$

where (a) follows from (21) and (39), c_1' is defined in (52), and c_1'' is a constant that is equal to $c_1' p_{k_o} \tau_{k_o} \nu$. The induced 2-norm of $\|I_M - \boldsymbol{D}_{11,i-1}^T\|$ can be derived as follows. It can be deduced from (83) and (84) that

$$\begin{aligned} c_1'' \mu_{max} I_M &\leq \boldsymbol{D}_{11,i-1} \leq c_2'' \mu_{max} I_M \\ -c_2'' \mu_{max} I_M &\leq -\boldsymbol{D}_{11,i-1} \leq -c_1'' \mu_{max} I_M \\ (1 - c_2'' \mu_{max}) I_M &\leq I_M - \boldsymbol{D}_{11,i-1} \leq (1 - c_1'' \mu_{max}) I_M \\ \|I_M - \boldsymbol{D}_{11,i-1}^T\| &\leq \max\{1 - c_1'' \mu_{max}, 1 - c_2'' \mu_{max}\} \\ &= 1 - O(\mu_{max}) \end{aligned} \tag{85}$$

$\|\boldsymbol{D}_{21,i-1}\|$ can be bounded as follows

$$\begin{aligned} \|\boldsymbol{D}_{21,i-1}\| &= \|(V_L^T A \otimes I_M)\boldsymbol{\mathcal{H}}_{i-1}'^T \mathcal{M}(p \otimes I_M)\| \\ &\leq \|(V_L^T A \otimes I_M)\| \|\boldsymbol{\mathcal{H}}_{i-1}'^T\| \|\mathcal{M}\| \|p \otimes I_M\| \\ &= \|(V_L^T A \otimes I_M)\| \|\boldsymbol{\mathcal{H}}_{i-1}'^T\| \|\mathcal{M}\| \sqrt{N p_{max}^2} \\ &= \alpha_{21} \|\boldsymbol{\mathcal{H}}_{i-1}'^T\| \|\mathcal{M}\| \\ &\overset{(a)}{\leq} \alpha_{21} \|\mathcal{M}\| (\max_{1 \leq k \leq N} \|\boldsymbol{H}_{k,i-1}'^{\delta}\|) \\ &\overset{(b)}{\leq} \alpha_{21} \|\mathcal{M}\| \widetilde{L}_{max} \delta_c \\ &= \alpha_{21} \|\mathcal{M}\| \widetilde{L}_{max} \delta_c \\ &\leq \sigma_{21} \mu_{max} = O(\mu_{max}) \end{aligned} \tag{86}$$

where (a) follows from the property of the induced 2-norm of block diagonal matrices, and (b) follows from (20) and (53). α_{21} and σ_{21} are positive constants that are defined to be equal to $\|(V_L^T A \otimes I_M)\| \sqrt{N p_{max}^2}$ and $\alpha_{21} \|\mathcal{M}\| \widetilde{L}_{max} \delta_c$, respectively. Similarly, it can be shown that

$$\begin{aligned} \|\boldsymbol{D}_{12,i-1}\| &\leq \sigma_{12} \mu_{max} = O(\mu_{max}) \\ \|\boldsymbol{D}_{22,i-1}\| &\leq \sigma_{22} \mu_{max} = O(\mu_{max}) \end{aligned} \tag{87}$$

APPENDIX B
NETWORK MEAN-SQUARE-ERROR STABILITY

Multiplying both sides of (23) by $\mathcal{V}_\epsilon^T$ we get:

$$\mathcal{V}_\epsilon^T \widetilde{\boldsymbol{\omega}}_i = \mathcal{V}_\epsilon^T \boldsymbol{\mathcal{B}}_{i-1} \widetilde{\boldsymbol{\omega}}_{i-1} + \mathcal{V}_\epsilon^T \mathcal{M} \boldsymbol{s}_i(\boldsymbol{\omega}_{i-1}) - \mathcal{V}_\epsilon^T \mathcal{M} b \tag{88}$$

$$\begin{bmatrix}\overline{\omega}_i\\ \check{\omega}_i\end{bmatrix}=\begin{bmatrix} I-D_{11,i-1}^T & -D_{21,i-1}^T\\ -D_{12,i-1}^T & \mathcal{J}_\epsilon^T-D_{22,i-1}^T\end{bmatrix}\begin{bmatrix}\overline{\omega}_{i-1}\\ \check{\omega}_{i-1}\end{bmatrix}+\begin{bmatrix}\overline{s}_{i-1}\\ \check{s}_{i-1}\end{bmatrix}-\begin{bmatrix}\bar{b}\\ \check{b}\end{bmatrix} \tag{89}$$

where

$$\mathcal{V}_\epsilon^T\widetilde{\omega}_i=\begin{pmatrix}(p^T\otimes I_M)\widetilde{\omega}_i\\ (V_R^T\otimes I_M)\widetilde{\omega}_i\end{pmatrix}=\begin{pmatrix}\bar{\omega}_i\\ \check{\omega}_i\end{pmatrix} \tag{90}$$

$$\mathcal{V}_\epsilon^T\mathcal{M}s(\psi_{i-1})=\begin{pmatrix}(p^T\otimes I_M)\mathcal{M}s(\psi_{i-1})\\ (V_R^T\otimes I_M)\mathcal{M}s(\psi_{i-1})\end{pmatrix}=\begin{pmatrix}\bar{s}_i\\ \check{s}_i\end{pmatrix} \tag{91}$$

$$\mathcal{V}_\epsilon^T\mathcal{M}b=\begin{pmatrix}(p^T\otimes I_M)\mathcal{M}b\\ (V_R^T\otimes I_M)\mathcal{M}b\end{pmatrix}=\begin{pmatrix}\bar{b}\\ \check{b}\end{pmatrix} \tag{92}$$

From the expression of $\bar{b}$ and $\check{b}$, we note that they depend on $\mathcal{M}$ and b. Recall from (31) that the entries of b are defined in terms of the gradient vectors and $\nabla J_k(\omega^o)$ and the matrix $Y_{k,o}^\delta=Y_{k,i-1}^\delta(\omega^o)$. Since each $J_k(\omega)$ is assumed to be twice-differentiable, then each $\nabla J_k(\omega)$ is a differentiable function and therefore bounded. In addition, $Y_{k,i-1}^\delta$ is also bounded as discussed in Section II-E. It follows that

$$\bar{b}=-\sum_{k=1}^N p_k\mu_k Y_{k,o}^\delta\nabla J_k(\omega^o)=O(\mu_{max}) \tag{93}$$

Similarly,

$$\check{b}=O(\mu_{max}) \tag{94}$$

From (89) we can write:

$$\overline{\omega}_i=(I-D_{11,i-1}^T)\overline{\omega}_{i-1}-(D_{21,i-1}^T)\check{\omega}_{i-1}+\overline{s}_{i-1} \tag{95}$$

$$\check{\omega}_i=(-D_{12,i-1}^T)\overline{\omega}_{i-1}+(\mathcal{J}_\epsilon^T-D_{22,i-1}^T)\check{\omega}_{i-1}+\check{s}_{i-1}-\check{b} \tag{96}$$

Conditioning both sides of (95) on $\mathcal{F}_{i-1}$ and using the condition on gradient noise that is given in (55),

$$\begin{aligned}\mathbf{E}[\|\overline{\omega}_i\|^2|\mathcal{F}_{i-1}]&=\mathbf{E}[\|(I-D_{11,i-1}^T)\overline{\omega}_{i-1}-(D_{21,i-1}^T)\check{\omega}_{i-1}\\&\quad+\overline{s}_{i-1}-\bar{b}\|^2|\mathcal{F}_{i-1}]\\&=\|(I-D_{11,i-1}^T)\overline{\omega}_{i-1}-(D_{21,i-1}^T)\check{\omega}_{i-1}\\&\quad-\bar{b}\|^2+\mathbf{E}[\|\overline{s}_{i-1}\|^2|\mathcal{F}_{i-1}]\end{aligned} \tag{97}$$

Conditioning again on both sides of (97) we get:

$$\begin{aligned}\mathbf{E}[\|\overline{\omega}_i\|^2]&=\mathbf{E}[\|(I-D_{11,i-1}^T)\overline{\omega}_{i-1}-(D_{21,i-1}^T)\check{\omega}_{i-1}\\&\quad-\bar{b}\|^2]+\mathbf{E}[\|\overline{s}_{i-1}\|^2]\\&\le\frac{1}{1-t}\mathbf{E}[\|(I-D_{11,i-1}^T)\overline{\omega}_{i-1}\|^2]\\&\quad+\frac{1}{t}\mathbf{E}[\|(D_{21,i-1}^T)\check{\omega}_{i-1}-\bar{b}\|^2]+\mathbf{E}[\|\overline{s}_{i-1}\|^2]\\&\le\frac{1}{1-t}\mathbf{E}[\|(I-D_{11,i-1}^T)\|^2\|\overline{\omega}_{i-1}\|^2]\\&\quad+\frac{2}{t}\mathbf{E}[\|(D_{21,i-1}^T)\|^2\|\check{\omega}_{i-1}\|^2]+\frac{2}{t}\|\bar{b}\|^2\\&\quad+\mathbf{E}[\|\overline{s}_{i-1}\|^2]\\&\overset{(a)}{=}\frac{(1-\sigma_{11}\mu_{max})^2}{1-\sigma_{11}\mu_{max}}\mathbf{E}[\|\overline{\omega}_{i-1}\|^2]\\&\quad+\frac{2\sigma_{21}^2\mu_{max}^2}{\sigma_{11}\mu_{max}}\mathbf{E}[\|\check{\omega}_{i-1}\|^2]+\frac{2\|\bar{b}\|^2}{\sigma_{11}\mu_{max}}\\&\quad+\mathbf{E}[\|\overline{s}_{i-1}\|^2]\\&=(1-O(\mu_{max}))\mathbf{E}[\|\overline{\omega}_{i-1}\|^2]\\&\quad+O(\mu_{max})\mathbf{E}[\|\check{\omega}_{i-1}\|^2]+O(\mu_{max})\\&\quad+\mathbf{E}[\|\overline{s}_{i-1}\|^2]\end{aligned} \tag{98}$$

where (a) follows from (85), and (86). It is shown in [16] that $\mathbf{E}[\|\overline{s}_{i-1}\|^2$ and $\mathbf{E}[\|\check{s}_{i-1}\|^2$ are upper bounded by

$$\begin{aligned}&\mathbf{E}[\|\overline{s}_{i-1}\|^2,\ \ \mathbf{E}[\|\check{s}_{i-1}\|^2\\&\le\nu_1^2\nu_2^2\beta_d^2\mu_{max}^2[\mathbf{E}[\|\bar{\omega}_i\|^2]+\mathbf{E}[\|\check{\omega}_i\|^2]]+\nu_1^2\mu_{max}^2\sigma_s^2\end{aligned} \tag{99}$$

Substituting (99) in (98) we get:

$$\begin{aligned}\mathbf{E}\|\overline{\omega}_i\|^2&\le(1-O(\mu_{max}))\mathbf{E}\|\overline{\omega}_{i-1}\|^2+O(\mu_{max})\mathbf{E}\|\check{\omega}_{i-1}\|^2\\&\quad+O(\mu_{max}^2)\end{aligned} \tag{100}$$

Similarly by following the same steps that led to (100) and by using the results of (87), we get:

$$\begin{aligned}\mathbf{E}[\|\check{\omega}_i\|^2]&\le(\rho(J_\epsilon)+\epsilon+O(\mu_{max}^2))\mathbf{E}[\|\check{\omega}_{i-1}\|^2]\\&\quad+O(\mu_{max}^2)\mathbf{E}[|\overline{\omega}_{i-1}\|^2]+O(\mu_{max}^2)\end{aligned} \tag{101}$$

Equations (100) and (101) can be written in a matrix form as follows:

$$\begin{aligned}\begin{bmatrix}\mathbf{E}\|\overline{\omega}_i\|^2\\ \mathbf{E}\|\check{\omega}_i\|^2\end{bmatrix}&\le\Gamma\begin{bmatrix}\mathbf{E}[\|\overline{\omega}_{i-1}\|^2]\\ \mathbf{E}[\|\check{\omega}_{i-1}\|^2]\end{bmatrix}+\begin{bmatrix}O(\mu_{max}^2)\\ O(\mu_{max}^2)\end{bmatrix}\\&=\Gamma\begin{bmatrix}\mathbf{E}[\|\overline{\omega}_{i-1}\|^2]\\ \mathbf{E}[\|\check{\omega}_{i-1}\|^2]\end{bmatrix}+\begin{bmatrix}O(\mu_{max}^2)\\ O(\mu_{max}^2)\end{bmatrix}\end{aligned} \tag{102}$$

where Γ is a matrix that is given by

$$\Gamma=\begin{bmatrix}1-O(\mu_{max}) & O(\mu_{max}^2)\\ O(\mu_{max}^2) & \rho(J_\epsilon)+\epsilon+O(\mu_{max}^2)\end{bmatrix} \tag{103}$$

Since $\rho(J_\epsilon)<1$ is independent of μ_{max}, and since ϵ and μ_{max} are small positive numbers that can be chosen arbitrarily small and independently of each other, then it can be easily shown that

$$\begin{aligned}\limsup_{i\to\infty}\begin{bmatrix}\mathbf{E}\|\overline{\omega}_i\|^2\\ \mathbf{E}\|\check{\omega}_i\|^2\end{bmatrix}&\le(I-\Gamma)^{-1}\begin{bmatrix}O(\mu_{max}^2)\\ O(\mu_{max}^2)\end{bmatrix}\\&=\begin{bmatrix}O(\frac{1}{\mu_{max}}) & O(1)\\ O(\mu_{max}) & O(1)\end{bmatrix}\begin{bmatrix}O(\mu_{max}^2)\\ O(\mu_{max}^2)\end{bmatrix}\\&=\begin{bmatrix}O(\mu_{max})\\ O(\mu_{max}^2)\end{bmatrix}\end{aligned} \tag{104}$$

from which we conclude that

$$\limsup_{i\to\infty}\mathbf{E}\|\overline{\omega}_i\|^2=O(\mu_{max}) \tag{105}$$

$$\limsup_{i\to\infty}\mathbf{E}\|\check{\omega}_i\|^2=O(\mu_{max}^2) \tag{106}$$

and therefore

$$\limsup_{i\to\infty}\mathbf{E}\|\widetilde{\boldsymbol{\omega}}_i\|^2 = \limsup_{i\to\infty}\mathbf{E}\|(\mathcal{V}_\epsilon^{-1})^T\begin{bmatrix}\overline{\omega}_i\\ \check{\omega}_i\end{bmatrix}\|^2 \le \limsup_{i\to\infty}\|(\mathcal{V}_\epsilon^{-1})^T\|^2[\mathbf{E}\|\overline{\omega}_i\|^2 + \mathbf{E}\|\check{\omega}_i\|^2] = O(\mu_{max}). \quad (107)$$

■

APPENDIX C
NETWORK FOURTH-ORDER MOMENT STABILITY

Using (95), and (101), we can write:

$$\|\overline{\omega}_i\|^4 = \|(I - D_{11,i-1}^T)\overline{\omega}_{i-1} - (D_{21,i-1}^T)\check{\omega}_{i-1} + \overline{s}_{i-1}\|^4 \quad (108)$$

$$\|\check{\omega}_i\|^4 = \| - D_{12,i-1}^T\overline{\omega}_{i-1} + (\mathcal{J}_\epsilon^T - D_{22,i-1}^T)\check{\omega}_{i-1} + \check{s}_{i-1} - \check{b}\|^4 \quad (109)$$

Using the following inequality:

$$\|a+b\|^4 \le \|a\|^4 + 3\|b\|^4 + 8\|a\|^2\|b\|^2 + 4\|a\|^2\mathrm{Re}(a*b) \quad (110)$$

Let

$$a = (I - D_{11,i-1}^T)\overline{\omega}_{i-1} - (D_{21,i-1}^T)\check{\omega}_{i-1} \quad (111)$$

$$b = \overline{s}_{i-1} \quad (112)$$

then decompose the right-hand side of (108) according to (110), and take the expectations on both sides and follow exactly the same steps in [16], and use the results of (85), (86) and (87), we get:

$$\limsup_{i\to\infty}\mathbf{E}\|\bar{\omega}_i\|^4 = O(\mu_{max}^2) \quad (113)$$

Similarly, let

$$a = (-D_{12,i-1}^T)\overline{\omega}_{i-1} + (\mathcal{J}_\epsilon^T - D_{22,i-1}^T)\check{\omega}_{i-1} - \check{b} \quad (114)$$

$$b = \check{s}_{i-1} \quad (115)$$

and decompose the right-hand side of (109) according to (110), and take the expectations on both sides and follow exactly the same steps in [16], we get:

$$\limsup_{i\to\infty}\mathbf{E}\|\check{\omega}_i\|^4 = O(\mu_{max}^4) \quad (116)$$

and, therefore

$$\limsup_{i\to\infty}\mathbf{E}\|\widetilde{\boldsymbol{\omega}}_i\|^4 = \limsup_{i\to\infty}\mathbf{E}\left(\|(\mathcal{V}_\epsilon^{-1})^T\begin{bmatrix}\overline{\omega}_i\\ \check{\omega}_i\end{bmatrix}\|^2\right)^2 \le \limsup_{i\to\infty}\|(\mathcal{V}_\epsilon^{-1})^T\|^4[\mathbf{E}\|\overline{\omega}_i\|^2 + \mathbf{E}\|\check{\omega}_i\|^2] = O(\mu_{max}^2). \quad (117)$$

■

APPENDIX D
NETWORK MEAN-ERROR STABILITY

$$\widetilde{\omega}_i = \mathcal{B}_{i-1}\widetilde{\omega}_{i-1} + \mathcal{M}s(\psi_{i-1}) - \mathcal{M}b \ , i \ge 0 \quad (118)$$

$$\boldsymbol{\mathcal{B}}_{i-1} = (\mathcal{I} - \mathcal{M}\boldsymbol{\mathcal{H}'}_{i-1})\mathcal{A}^T \quad (119)$$

Let

$$\widetilde{\mathcal{H}} = \mathcal{H} - \boldsymbol{\mathcal{H}'}_{i-1} \quad (120)$$

where $\mathcal{H}$ is a constant matrix defined as

$$\mathcal{I} = diag\{I_1, I_2, \ldots, I_N\} \quad (121)$$

and $I_k,\ k = 1, \ldots, N$ represents the eye matrix. Substituting (120) in (119), we can write:

$$\mathcal{B}_{i-1} = \mathcal{B} + \mathcal{M}\widetilde{\mathcal{H}}\mathcal{A}^T \quad (122)$$

where $\mathcal{B}$ is given by

$$\mathcal{B} = (I_{MN} - \mathcal{M}\mathcal{H})\mathcal{A}^T \quad (123)$$

Substituting (122) in (118), we get:

$$\widetilde{\omega}_i = \mathcal{B}\widetilde{\omega}_{i-1} + \mathcal{M}\widetilde{\mathcal{H}}\mathcal{A}^T\widetilde{\omega}_{i-1} + \mathcal{M}s(\psi_{i-1}) - \mathcal{M}b \ , i \ge 0 \quad (124)$$

Taking conditional expectations on both sides of (124)

$$\mathbf{E}[\widetilde{\omega}_i|\mathcal{F}_{i-1}] \overset{(a)}{=} \mathcal{B}\widetilde{\omega}_{i-1} + \mathcal{M}\widetilde{\mathcal{H}}\mathcal{A}^T\widetilde{\omega}_{i-1} - \mathcal{M}b \ , i \ge 0 \quad (125)$$

where (a) follows from $\mathbf{E}[s(\psi_{i-1})|\mathcal{F}_{i-1}] = 0$. Taking the expectations of both sides of (125), we can write:

$$\mathbf{E}[\widetilde{\omega}_i] = \mathcal{B}\mathbf{E}[\widetilde{\omega}_{i-1}] + \mathcal{M}\mathbf{E}[\widetilde{\mathcal{H}}\mathcal{A}^T\widetilde{\omega}_{i-1}] - \mathcal{M}b \ , i \ge 0 = \mathcal{B}\mathbf{E}[\widetilde{\omega}_{i-1}] + \mathcal{M}c_{i-1} - \mathcal{M}b \ , i \ge 0 \quad (126)$$

where c_{i-1} is defined as follows:

$$c_{i-1} = \mathbf{E}[\widetilde{\boldsymbol{\mathcal{H}}}\mathcal{A}^T\widetilde{\omega}_{i-1}] \quad (127)$$

Multiplying both sides of (126) by $\mathcal{V}_\epsilon^T$, we can write:

$$\begin{bmatrix}\mathbf{E}\overline{\omega}_i\\ \mathbf{E}\check{\omega}_i\end{bmatrix} = \overline{\mathcal{B}}\begin{bmatrix}\mathbf{E}\overline{\omega}_{i-1}\\ \mathbf{E}\check{\omega}_{i-1}\end{bmatrix} - \begin{bmatrix}0\\ \check{b}\end{bmatrix} + \mathcal{V}_\epsilon^T\mathcal{M}c_{i-1} \quad (128)$$

Note that (128) has the same form as in [16], with the exception that c_{i-1} and $\widetilde{\mathcal{H}}$ take different forms in this work from that in [16]. So, we present below the upper bounds of $\widetilde{\mathcal{H}}$, and $\|\mathcal{V}_\epsilon^T\mathcal{M}c_{i-1}\|$ and then the remaining steps of the proof follow similarly as in [16].

$$\widetilde{H}_{k,i-1}^\delta = I - \boldsymbol{H}_{k,i-1}'^\delta$$

$$\|\widetilde{H}_{k,i-1}^\delta\| = \|I - \boldsymbol{H}_{k,i-1}'^\delta\| \overset{(a)}{\le} L_{max}\mathcal{K}_d'CN\|\widetilde{\omega}_{i-1}\| \quad (129)$$

where (a) follows from (54), then

$$\|\widetilde{\mathcal{H}}_{i-1}\| = \max_{1\le k\le N}\|\widetilde{H}_{k,i-1}^\delta\| \le L_{max}\mathcal{K}_d'CN\|\widetilde{\omega}_{i-1}\| \quad (130)$$

therefore

$$\|\mathcal{V}_\epsilon^T\mathcal{M}c_{i-1}\| \le \|\mathcal{V}_\epsilon^T\|\|\mathcal{M}\|\|c_{i-1}\| \le \|\mathcal{V}_\epsilon^T\|\|\mathcal{M}\|\|\mathbf{E}(\widetilde{\mathcal{H}}_{i-1}\mathcal{A}^T\widetilde{\omega}_{i-1})\|$$

$$\leq (\|\mathcal{V}_\epsilon^T\|\|\mathcal{M}\|\|\mathcal{A}^T\|)\mathbf{E}[\|(\widetilde{\mathcal{H}}_{i-1}\|\|\widetilde{\omega}_{i-1})\|]$$
$$\overset{(a)}{\leq} L_{max}\mathcal{K}_d' CN\|\mathcal{V}_\epsilon^T\|\|\mathcal{M}\|\|\mathcal{A}^T\|\mathbf{E}\|\widetilde{\omega}_{i-1}\|^2$$
$$\overset{(b)}{=} r\mu_{max}\mathbf{E}\|\widetilde{\omega}_{i-1}\|^2 \tag{131}$$

for some constant r that is independent of μ_{max}. It follows from (62) that

$$\limsup_{i\to\infty}\|\mathcal{V}_\epsilon^T\mathcal{M}c_{i-1}\|^2 = O(\mu_{max}^2) \tag{132}$$

Rewrite (128) as

$$\begin{bmatrix}\overline{z}_i\\ \check{z}_i\end{bmatrix} = \overline{\mathcal{B}}\begin{bmatrix}\overline{z}_{i-1}\\ \check{z}_{i-1}\end{bmatrix} - \begin{bmatrix}0\\ \check{b}\end{bmatrix} - \begin{bmatrix}\overline{b}'\\ \check{b}'\end{bmatrix} - \begin{bmatrix}\overline{\mathcal{B}}'\overline{z}_{i-1}\\ \check{\mathcal{B}}'\check{z}_{i-1}\end{bmatrix} + \begin{bmatrix}\overline{c}_{i-1}\\ \check{c}_{i-1}\end{bmatrix} + \begin{bmatrix}\overline{c}'_{i-1}\\ \check{c}'_{i-1}\end{bmatrix} \tag{133}$$

then continue as in [16], and use the results of (85), (86) and (87), we get:

$$\limsup_{i\to\infty}\|\mathbf{E}\widetilde{\boldsymbol{\omega}}_i\| = O(\mu_{max}),\ \text{for } k = 1, 2, \ldots, N. \tag{134}$$

■

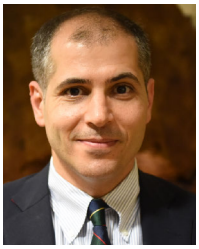
REFERENCES

[1] X. Wang, H. Ishii, L. Du, P. Cheng, and J. Chen, "Privacy-preserving distributed machine learning via local randomization and ADMM perturbation," *IEEE Trans. Signal Process.*, vol. 68, pp. 4226–4241, 2020.
[2] C. Li, P. Zhou, L. Xiong, Q. Wang, and T. Wang, "Differentially private distributed online learning," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 8, pp. 1440–1453, Aug. 2018.
[3] M. Wang, C. Xu, X. Chen, H. Hao, L. Zhong, and S. Yu, "Differential privacy oriented distributed online learning for mobile social video prefetching," *IEEE Trans. Multimedia*, vol. 21, no. 3, pp. 636–651, Mar. 2019.
[4] I. Ahmad, S. Shahabuddin, H. Malik, E. Harjula, T. Leppänen, L. Lovén, A. Anttonen, A. H. Sodhro, M. Mahtab Alam, M. Juntti, A. Ylä-Jääski, T. Sauter, A. Gurtov, M. Ylianttila, and J. Riekki, "Machine learning meets communication networks: Current trends and future challenges," *IEEE Access*, vol. 8, pp. 223418–223460, 2020.
[5] M. S. Elbamby, C. Perfecto, C.-F. Liu, J. Park, S. Samarakoon, X. Chen, and M. Bennis, "Wireless edge computing with latency and reliability guarantees," *Proc. IEEE*, vol. 107, no. 8, pp. 1717–1737, Aug. 2019.
[6] J. Wang and K. D. Pham, "An approximate distributed gradient estimation method for networked system optimization under limited communications," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 50, no. 12, pp. 5142–5151, Dec. 2020.
[7] Y. Guan and X. Ge, "Distributed attack detection and secure estimation of networked cyber-physical systems against false data injection attacks and jamming attacks," *IEEE Trans. Signal Inf. Process. Over Netw.*, vol. 4, no. 1, pp. 48–59, Mar. 2018.
[8] J. Zhang, Z. Jiang, Z. Chen, and X. Hu, "WMGCN: Weighted meta-graph based graph convolutional networks for representation learning in heterogeneous networks," *IEEE Access*, vol. 8, pp. 40744–40754, 2020.
[9] C. Shi, Y. Li, J. Zhang, Y. Sun, and P. S. Yu, "A survey of heterogeneous information network analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 1, pp. 17–37, Jan. 2017.
[10] W. Choi, K. Duraisamy, R. G. Kim, J. R. Doppa, P. P. Pande, D. Marculescu, and R. Marculescu, "On-chip communication network for efficient training of deep convolutional networks on heterogeneous manycore systems," *IEEE Trans. Comput.*, vol. 67, no. 5, pp. 672–686, May 2018.
[11] M. Ma, A. Zhu, S. Guo, and Y. Yang, "Intelligent network selection algorithm for multiservice users in 5G heterogeneous network system: Nash Q-learning method," *IEEE Internet Things J.*, vol. 8, no. 15, pp. 11877–11890, Aug. 2021.
[12] K. Srivastava and A. Nedic, "Distributed asynchronous constrained stochastic optimization," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 4, pp. 772–790, Aug. 2011.
[13] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2508–2530, Jun. 2006.
[14] D. Jakovetic, J. Xavier, and J. M. F. Moura, "Cooperative convex optimization in networked systems: Augmented Lagrangian algorithms with directed gossip communication," *IEEE Trans. Signal Process.*, vol. 59, no. 8, pp. 3889–3902, Aug. 2011.
[15] M. J. Wainwright and M. I. Jordan, *Graphical Models, Exponential Families, and Variational Inference*. Boston, MA, USA: Now, 2008.
[16] A. H. Sayed, *Adaptation, Learning, and Optimization Over Networks*. Boston, MA, USA: Now, 2014.
[17] S. Kar and J. M. F. Moura, "Distributed consensus algorithms in sensor networks with imperfect communication: Link failures and channel noise," *IEEE Trans. Signal Process.*, vol. 57, no. 1, pp. 355–369, Jan. 2009.
[18] T. Can Aysal, M. J. Coates, and M. G. Rabbat, "Distributed average consensus with dithered quantization," *IEEE Trans. Signal Process.*, vol. 56, no. 10, pp. 4905–4918, Oct. 2008.
[19] A. G. Dimakis, S. Kar, J. M. F. Moura, M. G. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proc. IEEE*, vol. 98, no. 11, pp. 1847–1864, Nov. 2010.
[20] V. Saligrama, M. Alanyali, and O. Savas, "Distributed detection in sensor networks with packet losses and finite capacity links," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4118–4132, Nov. 2006.
[21] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A.-R. Sadeghi, "DÏoT: A federated self-learning anomaly detection system for IoT," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2019, pp. 756–767.
[22] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *J. Chem. Phys.*, vol. 21, no. 6, pp. 1087–1092, Jun. 1953.
[23] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," in *Proc. 42nd IEEE Int. Conf. Decis. Control*, vol. 5, Dec. 2003, pp. 4997–5002.
[24] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, Apr. 1970.
[25] L. J. Billera and P. Diaconis, "A geometric interpretation of the Metropolis–Hastings algorithm," *Stat. Sci.*, vol. 16, no. 4, pp. 335–339, Nov. 2001.
[26] A. Sayed, *Adaptive Filters*. Hoboken, NJ, USA: Wiley, 2011.
[27] B. Bollobás, *Modern Graph Theory* (Graduate Texts in Mathematics), vol. 184, 1st ed. New York, NY, USA: Springer-Verlag, 1998.
[28] D. M. Cvetković, M. Doob, and H. Sachs, *Spectra of Graphs: Theory and Applications*. New York, NY, USA: Academic Press, 1980.
[29] W. Kocay and D. L. Kreher, *Graphs, Algorithms and Optimization*. London, U.K.: Chapman & Hall, 2004.
[30] V. Solo, "Stability of distributed adaptive algorithms I: Consensus algorithms," in *Proc. 54th IEEE Conf. Decis. Control (CDC)*, Dec. 2015, pp. 7422–7427.
[31] D. Jin, J. Chen, C. Richard, J. Chen, and A. H. Sayed, "Affine combination of diffusion strategies over networks," *IEEE Trans. Signal Process.*, vol. 68, pp. 2087–2104, 2020.
[32] J.-W. Lee, S.-E. Kim, W.-J. Song, and A. H. Sayed, "Spatio-temporal diffusion strategies for estimation and detection over networks," *IEEE Trans. Signal Process.*, vol. 60, no. 8, pp. 4017–4034, Aug. 2012.
[33] C. P. Subha and S. Malarkkan, "Optimization of energy efficient cellular learning automata algorithm for heterogeneous wireless sensor networks," in *Proc. 10th Int. Conf. Intell. Syst. Control (ISCO)*, Jan. 2016, pp. 1–6.
[34] Y.-F. Huang, T.-H. Tan, N.-C. Wang, Y.-L. Chen, and Y.-L. Li, "Resource allocation for D2D communications with a novel distributed Q-learning algorithm in heterogeneous networks," in *Proc. Int. Conf. Mach. Learn. Cybern. (ICMLC)*, vol. 2, Jul. 2018, pp. 533–537.
[35] W. Tashan, I. Shayea, S. Aldirmaz-Colak, M. Ergen, M. H. Azmi, and A. Alhammadi, "Mobility robustness optimization in future mobile heterogeneous networks: A survey," *IEEE Access*, vol. 10, pp. 45522–45541, 2022.
[36] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Trans. Signal Inf. Process. Over Netw.*, vol. 1, no. 2, pp. 89–103, Jun. 2015.

[37] J. Park, S. Samarakoon, M. Bennis, and M. Debbah, "Wireless network intelligence at the edge," *Proc. IEEE*, vol. 107, no. 11, pp. 2204–2239, Nov. 2019.
[38] A. E. Eshratifar, M. S. Abrishami, and M. Pedram, "JointDNN: An efficient training and inference engine for intelligent mobile cloud computing services," *IEEE Trans. Mobile Comput.*, vol. 20, no. 2, pp. 565–576, Feb. 2021.
[39] W. Kim, M. S. Stankovic, K. H. Johansson, and H. J. Kim, "A distributed support vector machine learning over wireless sensor networks," *IEEE Trans. Cybern.*, vol. 45, no. 11, pp. 2599–2611, Nov. 2015.
[40] P. Ray and P. K. Varshney, "Estimation of spatially distributed processes in wireless sensor networks with random packet loss," *IEEE Trans. Wireless Commun.*, vol. 8, no. 6, pp. 3162–3171, Jun. 2009.
[41] S. Jayaprakasam, S. K. A. Rahim, and C. Y. Leow, "Distributed and collaborative beamforming in wireless sensor networks: Classifications, trends, and research directions," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2092–2116, 4th Quart., 2017.
[42] Q. Zhou, D. Li, S. Kar, L. M. Huie, H. V. Poor, and S. Cui, "Learning-based distributed detection-estimation in sensor networks with unknown sensor defects," *IEEE Trans. Signal Process.*, vol. 65, no. 1, pp. 130–145, Jan. 2017.
[43] P. A. Forero, A. Cano, and G. B. Giannakis, "Distributed clustering using wireless sensor networks," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 4, pp. 707–724, Aug. 2011.
[44] Y. Cui, K. Cao, J. Zhou, and T. Wei, "Optimizing training efficiency and cost of hierarchical federated learning in heterogeneous mobile-edge cloud computing," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 42, no. 5, pp. 1518–1531, May 2023.
[45] C. Park and J. Lee, "Mobile edge computing-enabled heterogeneous networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 1038–1051, Feb. 2021.
[46] B. Wang, J. Xie, K. Lu, Y. Wan, and S. Fu, "On batch-processing based coded computing for heterogeneous distributed computing systems," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 3, pp. 2438–2454, Jul. 2021.
[47] N. Shakya, F. Li, and J. Chen, "On distributed computing with heterogeneous communication constraints," *IEEE/ACM Trans. Netw.*, vol. 30, no. 6, pp. 2776–2787, Dec. 2022.
[48] N. Woolsey, R.-R. Chen, and M. Ji, "Coded elastic computing on machines with heterogeneous storage and computation speed," *IEEE Trans. Commun.*, vol. 69, no. 5, pp. 2894–2908, May 2021.
[49] Z. Zhang, T. Song, L. Lin, Y. Hua, X. He, Z. Xue, R. Ma, and H. Guan, "Towards ubiquitous intelligent computing: Heterogeneous distributed deep neural networks," *IEEE Trans. Big Data*, vol. 8, no. 3, pp. 644–657, Jun. 2022.
[50] S.-Y. Tu and A. H. Sayed, "Diffusion strategies outperform consensus strategies for distributed estimation over adaptive networks," *IEEE Trans. Signal Process.*, vol. 60, no. 12, pp. 6217–6234, Dec. 2012.
[51] J. Chen, C. Richard, and A. H. Sayed, "Multitask diffusion adaptation over networks with common latent representations," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 3, pp. 563–579, Apr. 2017.
[52] H. Robbins and S. Monro, "A stochastic approximation method," *Ann. Math. Statist.*, vol. 22, no. 3, pp. 400–407, Sep. 1951.
[53] N. Qian, "On the momentum term in gradient descent learning algorithms," *J. Int. Neural Netw. Soc.*, vol. 12, no. 1, pp. 145–151, Jan. 1999.
[54] Y. Nesterov, "A method for unconstrained convex minimization problem with the rate of convergence o($1/k^2$)," *Doklady AN USSR*, vol. 269, no. 3, pp. 543–547, 1983.
[55] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *J. Mach. Learn. Res.*, vol. 12, pp. 2121–2159, Jul. 2011.
[56] M. D. Zeiler, "ADADELTA: An adaptive learning rate method," 2012, *arXiv:1212.5701*.
[57] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[58] T. Dozat, "Incorporating Nesterov momentum into Adam," in *Proc. ICLR Workshop*, 2016, pp. 1–4.
[59] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of Adam and beyond," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–23.
[60] M. H. Nassralla, N. Akl, A. Mohamed, and Z. Dawy, "A low complexity approximation of gradient descent for learning over single and multi-agent systems," in *Proc. 19th Int. Conf. Wireless Mobile Comput., Netw. Commun. (WiMob)*, Jun. 2023, pp. 350–355.
[61] B. T. Polyak and Y. Z. Tsypkin, "Pseudogradient adaptation and training algorithms," *Automat. Remote Control*, vol. 34, no. 3, pp. 377–397, Jan. 1973.
[62] M. Wu, N. Xiong, A. V. Vasilakos, V. C. M. Leung, and C. L. P. Chen, "RNN-K: A reinforced Newton method for consensus-based distributed optimization and control over multiagent systems," *IEEE Trans. Cybern.*, vol. 52, no. 5, pp. 4012–4026, May 2022.
[63] A. Mokhtari, Q. Ling, and A. Ribeiro, "Network Newton distributed optimization methods," *IEEE Trans. Signal Process.*, vol. 65, no. 1, pp. 146–161, Jan. 2017.
[64] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York, NY, USA: Springer, 2006.
[65] I. Griva, S. G. Nash, and A. Sofer, *Linear and Nonlinear Optimization*, 2nd ed. Philadelphia, PA, USA: SIAM, 2008.
[66] M. Eisen, A. Mokhtari, and A. Ribeiro, "Decentralized quasi-Newton methods," *IEEE Trans. Signal Process.*, vol. 65, no. 10, pp. 2613–2628, May 2017.
[67] L. Bottou, F. E. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *SIAM Rev.*, vol. 60, no. 2, pp. 223–311, Jan. 2018.
[68] X.-L. Li, "Preconditioned stochastic gradient descent," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 1454–1466, May 2018.
[69] N. Costilla-Enriquez, Y. Weng, and B. Zhang, "Combining Newton–Raphson and stochastic gradient descent for power flow analysis," *IEEE Trans. Power Syst.*, vol. 36, no. 1, pp. 514–517, Jan. 2021.
[70] Q. Jin, T. Ren, N. Ho, and A. Mokhtari, "Statistical and computational complexities of BFGS quasi-Newton method for generalized linear models," 2022, *arXiv:2206.00207v1*.
[71] D. B¨hning, "Multinomial logistic regression algorithm," *Ann. Inst. Stat. Math.*, vol. 44, no. 1, pp. 197–200, Oct. 1992.
[72] L. Deng, "The MNIST database of handwritten digit images for machine learning research," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 141–142, Nov. 2012.

MOHAMMAD H. NASSRALLA received the degree from the American University of Beirut. He is currently a Senior Staff Digital Design Engineer with Infineon AG Company, Germany. He has several years of research and industrial experience in signal processing, machine learning, distributed optimization, and hardware implementation on FPGA.

NAEEM AKL received the B.E. degree in computer and communication engineering from the American University of Beirut, Beirut, Lebanon, in 2013, and the M.E. and Ph.D. degrees in electrical and computer engineering from The University of Texas at Austin, Austin, TX, USA, in 2015 and 2018, respectively. He is currently a Staff Systems Engineer with Qualcomm Wireless Research and Development, Bridgewater, NJ, USA, working on the development of 5G/6G cellular technologies. His research interests include the design and standardization of wireless systems, signal processing, and machine learning.



ZAHER DAWY (Senior Member, IEEE) received the B.E. degree in computer and communications engineering from the American University of Beirut (AUB), in 1998, and the Ph.D. degree in communications engineering from the Munich University of Technology (TUM), in 2004. He joined AUB, in 2004, where he is currently a Professor with the Electrical and Computer Engineering Department. His teaching and research interests include wireless communications, the Internet of Things, computational biology, and biomedical engineering.