DISH: A Distributed Hybrid Primal-Dual Optimization Framework to Utilize System Heterogeneity

Xiaochun Niu and Ermin Wei

***Abstract*— We consider solving distributed consensus optimization problems over multi-agent networks. Current distributed methods fail to capture the heterogeneity among agents' local computation capacities. We propose DISH as a *distributed hybrid primal-dual algorithmic framework* to handle and utilize system heterogeneity. Specifically, DISH allows those agents with higher computational capabilities or cheaper computational costs to implement Newton-type updates locally, while other agents can adopt the much simpler gradient-type updates. We show that DISH is a general framework and includes EXTRA, DIGing, ESOM-0 as special cases. Moreover, when all agents take both primal and dual Newton-type updates, DISH approximates the Newton's method by estimating both primal and dual Hessians. Theoretically, we show that DISH achieves a linear (Q-linear) convergence rate to the exact optimal solution for strongly convex functions, regardless of agents' choices of gradient-type and Newton-type updates. Finally, we perform numerical studies to demonstrate the efficacy of DISH in practice. To the best of our knowledge, DISH is the first hybrid method allowing heterogeneous local updates for distributed consensus optimization under general network topology with provable convergence and rate guarantees.**

I. Introduction

Distributed optimization problems over a connected network with multiple agents have gained significant attention recently. This is motivated by a wide range of applications such as power grids [1], [2], sensor networks [3], [4], communication networks [5], [6], and machine learning [7], [8]. In such problems, each agent only has access to its local data and only communicates with its neighbors in the network due to privacy issues or communication budgets [9]. All agents in the system aim to optimize an objective function collaboratively by employing a distributed procedure. Formally, we denote by $\mathcal{G} = \{\mathcal{N}, \mathcal{E}\}$ a connected undirected network with the node set $\mathcal{N} = \{1, \cdots, n\}$ and the edge set $\mathcal{E} \subseteq \{\{i, j\} \,|\, i, j \in \mathcal{N}, i \neq j\}$. We study the distributed optimization problem over $\mathcal{G}$,

$$\min_{\omega \in \mathbb{R}^d} \quad \sum_{i=1}^{n} f_i(\omega), \tag{1}$$

where $\omega \in \mathbb{R}^d$ is the decision variable and $f_i : \mathbb{R}^d \to \mathbb{R}$ is the local objective function corresponding to the i^{th} agent. For instance, if we consider an empirical risk minimization problem in a supervised learning setting, the goal of the system is to learn a shared model over all the data in the network without exchanging local data, where local f_i denotes expected loss over the local data at the i^{th} agent.

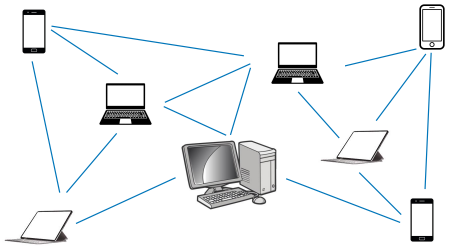


Fig. 1. A Heterogeneous System.

In order to develop a distributed method for solving Problem 1, we decouple the computation of individual agents by introducing the local copy of the decision variable at the i^{th} agent as $x_i \in \mathbb{R}^d$. We formulate Problem 1 over the network $\mathcal{G}$ as a *consensus optimization* problem [10], [11],

$$\min_{x_1, \cdots, x_n} \quad \sum_{i=1}^{n} f_i(x_i) \quad \text{s.t. } x_i = x_j, \text{ for } \{i, j\} \in \mathcal{E}. \tag{2}$$

The consensus constraint $x_i = x_j$ for $\{i, j\} \in \mathcal{E}$ enforces the equivalence of Problems 1 and 2 for a connected network $\mathcal{G}$.

While there is growing literature on developing distributed optimization algorithms to solve Problem 2, most existing methods require all agents to take the same type of updates. Such methods include gradient-type methods [11]–[14] and Newton-type methods [15]–[18]. With these methods, if any agent in a system cannot handle high-order computation, a fast-converging method utilizing higher-order information will not be applicable to the whole system. As a result, the system could not fully utilize the distributed computation capability when faced with heterogeneity. This is in stark contrast to the fact that many practical distributed systems have heterogeneous agents. There can be drastically varying computation and communication capabilities among the agents due to different hardware, network connectivity, and battery power. [19]. Figure 1 shows an example of a heterogeneous system. Moreover, due to the recent global chip shortage, processors with advanced computation capability have very limited availability. Consequently, many distributed computation systems have only a few agents with advanced hardware co-existing with many older processors. Therefore, it is imperative to provide a flexible and efficient hybrid method to utilize heterogeneous agents. To the best of our knowledge, this paper takes the first step in this direction.

In order to handle and utilize the system heterogeneity,

The authors are with the Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL 60201 USA.

This work was supported in part by the National Science Foundation (NSF) under Grant ECCS-2030251 and CMMI-2024774.

we propose a *distributed hybrid primal-dual algorithmic framework* named DISH. DISH allows agents to choose gradient-type updates or Newton-type updates based on their computation capabilities. Specifically, there can be both gradient-type and Newton-type agents in the same communication round and each agent can switch to either type of updates based on its current situation. We show that DISH include primal-dual gradient-type methods such as EXTRA [12], DIGing [13], and [14] and primal-Newton-dual-gradient methods like ESOM [20] as special cases. Theoretically, we show that DISH achieves a linear (Q-linear) convergence rate to the exact optimal solution for strongly convex functions, regardless of agents' choices of gradient-type and Newton-type updates. Finally, we conduct numerical experiments on decentralized least squares problems and logistic regression problems and demonstrate the efficacy of the DISH algorithmic framework. We observe that when all agents always take primal-dual Newton-type updates, DISH offers faster convergence speed over gradient methods.

Related Works. Our work is related to the proliferating literature on distributed optimization methods to solve Problem 2. There are first-order primal iterative methods, like distributed (sub)-gradient descent (DGD) [11], which takes a linear combination of a local gradient descent step and a weighted average among local neighbors. DGD finds a near-optimal solution with constant stepsize. Based on DGD, other related methods including [12]–[14], [21] use gradient tracking technique, which can find the exact solution with constant stepsize and be viewed as primal-dual gradient methods with respect to augmented Lagrangian formulation. Second-order primal methods, including Network Newton [22] and Distributed Newton method [23], rely on an inner loop to iteratively approximate a Newton step. [24] derives a DGD based method with the inclusion of first and second-order updates in the continuous-time setting. Their method cannot be directly applied in discrete-time and lacks convergence rate analysis. Another popular approach is to use dual decomposition-based methods such as ADMM [25], [26], CoCoA [27], ESOM [20], and PD-QN [28]. Among these, PD-QN is a primal-dual quasi-Newton method with a linear convergence guarantee. ESOM is most related to our approach, which proposes to perform second-order updates in the primal space and first-order updates in the dual space and has a provable linear convergence rate. However, none of these methods allow different types of updates for heterogeneous agents. Our earlier work [29] develops a linearly converging distributed primal-dual hybrid method that allows different types of updates, but relies on the structure of a server-client (federated) network. To the best of our knowledge, DISH is the first hybrid method allowing heterogeneous local updates for distributed consensus optimization under general network topology with provable convergence and rate guarantees.

Contributions. Our main contributions are fourfold:

- We propose DISH as a distributed hybrid primal-dual algorithmic framework, which allows agents to employ both gradient-type and Newton-type information to harvest system heterogeneity.
- For the agents capable of second order computation in DISH, we develop a Newton-type method that approximates Newton's step in both the primal and the dual spaces with a distributed implementation.
- We show a linear convergence analysis of DISH to find the optimal solution regardless of agents' choice of gradient-type or Newton-type updates.
- We conduct numerical experiments and demonstrate the efficacy of DISH in practice.

Due to space considerations, proofs have been omitted. Interested readers are referred to Section Appendix for details at https://github.com/xniu1/DISH/blob/main/Appendix.pdf.

Notations. We denoted by $\otimes$ the Kronecker product. For any $m \in \mathbb{Z}^+$, we denote by $I_m \in \mathbb{R}^{m\times m}$ the identity matrix and $\mathbb{1}_m = (1, \cdots, 1)^\intercal \in \mathbb{R}^m$ the vector of all ones. For any symmetric matrix S, we denote by $\rho(S)$ its spectral radius. For any positive semidefinite matrix $M \in \mathbb{R}^{p\times p}$, we denote by $\sigma^+_{\min}(M)$ its smallest positive eigenvalue and $\|y\|^2_M = y^\intercal My$ for any $y \in \mathbb{R}^p$. For any positive definite matrix A, we denote by $\theta_{\min}(A)$ its smallest eigenvalue.

II. Preliminaries

In this section, we reformulate Problem 2 in a compact form and introduce its dual problem based on the augmented Lagrangian [30], which prepares our derivation of DISH.

Equivalent Reformulation. For compactness, we reformulate Problem 2 in the following equivalent form,

$$\min_{\mathbf{x}\in\mathbb{R}^{nd}} \quad f(\mathbf{x}) = \sum_{i=1}^{n} f_i(x_i) \quad \text{s.t. } (Z \otimes I_d)\mathbf{x} = \mathbf{x}, \tag{3}$$

where $\mathbf{x} = (x_1^\intercal, \cdots, x_n^\intercal)^\intercal$ is the concatenation of local variables, $f : \mathbb{R}^{nd} \to \mathbb{R}$ is the aggregate function, and $Z \in \mathbb{R}^{d\times d}$ with elements z_{ij} is a consensus matrix corresponding to $\mathcal{G}$. We emphasize that Z satisfies the following assumption.

Assumption 1. *The consensus matrix Z satisfies that:*

(a) *Off-diagonal elements: $z_{ij} \neq 0$ if and only if $\{i, j\} \notin \mathcal{E}$;*
(b) *Diagonal elements: $z_{ii} > 0$ for all $i \in \mathcal{N}$;*
(c) *$z_{ij} = z_{ji}$ for all $i \neq j$ and $i, j \in \mathcal{N}$;*
(d) *$Z\mathbb{1}_n = \mathbb{1}_n$.*

Assumption 1 is standard for consensus matrices, where (a) states the right sparsity pattern of Z, (b) ensures the aperiodicity of $\mathcal{G}$, and (c) and (d) impose that Z is symmetric and doubly stochastic. We denote by γ the second largest eigenvalue of Z. With the irreducibility of Z guaranteed by the connectness of $\mathcal{G}$, by Perron-Frobenius theorem, we have $\rho(Z) = 1$, $\gamma < 1$, and $\text{null}(I - Z) = \text{span}\{\mathbb{1}_n\}$. A matrix Z under Assumption 1 is known as the consensus matrix due to its property that $(Z \otimes I_d)\mathbf{x} = \mathbf{x}$ if and only if $x_i = x_j$ for all $i, j \in \mathcal{N}$ [11]. If we denote $W = (I_n - Z) \otimes I_d$, we have $\sigma^+_{\min}(W) = 1 - \gamma$ and $\text{null}(W) = \text{span}\{\mathbb{1}_n \otimes y : y \in \mathbb{R}^d\}$.

We can rewrite Problem 3 using the matrix W as follows,

$$\min_{\mathbf{x}\in\mathbb{R}^{nd}} \quad f(\mathbf{x}) = \sum_{i=1}^{n} f_i(x_i) \quad \text{s.t. } W\mathbf{x} = 0. \tag{4}$$

We will impose the next assumption throughout the paper.

Assumption 2. *The local function f_i is twice differentiable, s_i-strongly convex, and ℓ_i-Lipschitz smooth with positive constants $0 < s_i \leq \ell_i < \infty$ for any agent $i \in \mathcal{N}$.*

Assumption 2 postulates that the local Hessian is bounded by $s_i I_d \preceq \nabla^2 f_i(\cdot) \preceq \ell_i I_d$ for any $i \in \mathcal{N}$. For convenience, we denote by $s = \min_{i\in\mathcal{N}}\{s_i\}$ and $\ell = \max_{i\in\mathcal{N}}\{\ell_i\}$.

Augmented Lagrangian and Dual Problem. In order to develop primal-dual methods for solving Problem 4 with the consensus constraint, we introduce the dual function based on the augmented Lagrangian. We denote by $\boldsymbol{\lambda} = (\lambda_1^\intercal, \cdots, \lambda_n^\intercal)^\intercal$ the dual variable with $\lambda_i \in \mathbb{R}^d$ associated with the constraint $z_{ii}x_i - \sum_{j\in\mathcal{N}} z_{ij}x_j = 0$ at agent i. We define the augmented Lagrangian $L(\mathbf{x}, \boldsymbol{\lambda})$ of Problem 4 as

$$L(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) + \boldsymbol{\lambda}^\intercal W\mathbf{x} + \frac{\mu}{2}\mathbf{x}^\intercal V\mathbf{x}, \tag{5}$$

where $\mu \geq 0$ and $V \in \mathbb{R}^{nd\times nd}$ is positive semi-definite with $\text{null}(V) = \text{span}\{\mathbb{1}_n \otimes y : y \in \mathbb{R}^d\}$. The augmentation term $\mu\mathbf{x}^\intercal V\mathbf{x}/2$ serves as a penalty for the violation of the consensus constraint. Examples of choices for V include W and W^2. For convenience, throughout the paper, we will use $V = W$. We remark that $L(\mathbf{x}, \boldsymbol{\lambda})$ is the (unaugmented) Lagrangian function when $\mu = 0$. The augmented Lagrangian in (5) can also be viewed as the (unaugmented) Lagrangian associated with the penalized problem

$$\min_{\mathbf{x}\in\mathbb{R}^{nd}} f(\mathbf{x}) + \frac{\mu}{2}\mathbf{x}^\intercal V\mathbf{x} \quad \text{s.t. } W\mathbf{x} = 0. \tag{6}$$

Problem 6 is equivalent to Problem 4 since $\mu\mathbf{x}^\intercal V\mathbf{x}/2$ is zero for any feasible $\mathbf{x}$. By the convexity condition in Assumption 2 and the Slater's condition, strong duality holds for Problem 6 [31]. Thus, Problem 6, as well as Problem 4, are equivalent to the following dual problem,

$$\max_{\boldsymbol{\lambda}\in\mathbb{R}^{nd}} g(\boldsymbol{\lambda}), \text{ where } g(\boldsymbol{\lambda}) = \min_{\mathbf{x}\in\mathbb{R}^{nd}} L(\mathbf{x}, \boldsymbol{\lambda}), \tag{7}$$

where we refer to $g : \mathbb{R}^{nd} \to \mathbb{R}$ as the dual function. For any $\boldsymbol{\lambda} \in \mathbb{R}^{nd}$, as we will show in Lemma 5, the function $L(\cdot, \boldsymbol{\lambda})$ is strongly convex with a unique minimizer defined as

$$\mathbf{x}^*(\boldsymbol{\lambda}) = \operatorname*{argmin}_{\mathbf{x}\in\mathbb{R}^{nd}} L(\mathbf{x}, \boldsymbol{\lambda}). \tag{8}$$

By the definition of g in (7), we have $L(\mathbf{x}^*(\boldsymbol{\lambda}), \boldsymbol{\lambda}) = g(\boldsymbol{\lambda})$. We show the explicit forms of the gradient and the Hessian of the dual function g in the following lemma [32].

Lemma 3. *Under Assumption 2, with $\mathbf{x}^*(\boldsymbol{\lambda})$ defined in (8), the gradient and the Hessian of the dual function $g(\boldsymbol{\lambda})$ defined in Problem 7 are given by*

$$\nabla g(\boldsymbol{\lambda}) = W\mathbf{x}^*(\boldsymbol{\lambda}),$$
$$\nabla^2 g(\boldsymbol{\lambda}) = -W(\nabla^2_{\mathbf{xx}} L(\mathbf{x}^*(\boldsymbol{\lambda}), \boldsymbol{\lambda}))^{-1} W.$$

For the rest of the paper, we focus on developing distributed methods for solving Problem 7.

III. Algorithm

This section proposes DISH as a distributed hybrid primal-dual algorithmic framework for solving Problem 7, which allows choices of gradient-type and Newton-type updates for each agent at each iteration based on their current battery/computation capabilities and provides flexibility to handle and utilize heterogeneity in the network.

A. DISH to Handle and Utilize System Heterogeneity

Specifically, we propose the following hybrid updates. At each iteration k,

$$\mathbf{x}^{k+1} = \mathbf{x}^k - AP^k\nabla_{\mathbf{x}} L(\mathbf{x}^k, \boldsymbol{\lambda}^k),$$
$$\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + BQ^k\nabla_{\boldsymbol{\lambda}} L(\mathbf{x}^k, \boldsymbol{\lambda}^k), \tag{9}$$

where stepsize matrices $A = \text{diag}\{a_1, \cdots, a_n\} \otimes I_d$ and $B = \text{diag}\{b_1, \cdots, b_n\} \otimes I_d$ consist of personalized stepsizes a_i and $b_i > 0$ for $i \in \mathcal{N}$ and block diagonal update matrices $P^k = \text{diag}\{P_1^k, \cdots, P_n^k\}$ and $Q^k = \text{diag}\{Q_1^k, \cdots, Q_n^k\}$ are composed of positive definite local update matrices P_i^k and $Q_i^k \in \mathbb{R}^{d\times d}$ for $i \in \mathcal{N}$. Here are some examples of possible local update matrices. For the primal updates, we can take

$$\text{Gradient-type: } P_i^k = I_d;$$
$$\text{Newton-type: } P_i^k = (\nabla^2 f_i(x_i^k) + \mu I_d)^{-1}. \tag{10}$$

As for the dual updates, we can use

$$\text{Gradient-type: } Q_i^k = I_d;$$
$$\text{Newton-type: } Q_i^k = \nabla^2 f_i(x_i^k) + \mu I_d. \tag{11}$$

As we go through the following sections, we will explain such choices of local update matrices. We remark that as we will show in Theorem 7, the analysis of DISH only requires the update matrices P_i^k and Q_i^k to be positive definite. Thus, the agents can take other local updates such as quasi-Newton methods like BFGS [33] and the scaled gradient method [10]. These can be future directions. Nevertheless, this paper mainly focuses on gradient-type and Newton-type updates.

By substituting the partial derivatives in (9), we can write DISH in a compact form as follows, at iteration k,

$$\mathbf{x}^{k+1} = \mathbf{x}^k - AP^k(\nabla f(\mathbf{x}^k) + W\boldsymbol{\lambda}^k + \mu W\mathbf{x}^k),$$
$$\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + BQ^k W\mathbf{x}^k, \tag{12}$$

Based on (12), we provide the distributed implementation of DISH in Algorithm 1. DISH in Algorithm 1 consists of a primal (Line 5) and a dual step (Line 6) for each agent, where both steps can be either gradient-type or Newton-type based on the agent's choice in each iteration. We note that this is a very flexible framework. An agent may use different types of updates across iterations, and between primal and dual spaces within the same iteration. The primal and dual updates can be computed simultaneously as they both depend on values from the previous iteration. In the sequel, we provide interpretation on DISH with some specific choices of updates.

B. Relation of DISH to Existing Methods

Now we illustrate how the introduced DISH algorithm is related to some other distributed optimization methods.

Algorithm 1 DISH: Distributed Hybrid Primal-dual Algorithmic Framework for Consensus Optimization

Input: Initialization $x_i^0, \lambda_i^0 \in \mathbb{R}^d$, stepsizes $a_i, b_i \in \mathbb{R}^+$ for all $i \in \mathcal{N}$, and the penalty parameter μ.
for $k = 0, \ldots, K-1$ **do**
  for each agent $i \in \mathcal{N}$ in parallel **do**
    Send x_i^k and λ_i^k to its neighbors j for $\{i,j\} \in \mathcal{E}$;
    $$x_i^{k+1} = x_i^k - a_i P_i^k \Big\{ \nabla f_i(x_i^k) + (1 - z_{ii})\lambda_i^k - \sum_{\{j,i\}\in\mathcal{E}} z_{ji}\lambda_j^k + \mu\big[(1-z_{ii})x_i^k - \sum_{\{i,j\}\in\mathcal{E}} z_{ij}x_j^k\big] \Big\};$$
    $$\lambda_i^{k+1} = \lambda_i^k + b_i Q_i^k \big[(1-z_{ii})x_i^k - \textstyle\sum_{\{i,j\}\in\mathcal{E}} z_{ij}x_j^k\big];$$
  end for
end for

1) Primal-Dual Gradient-type Method (EXTRA, DIGing, and [14]): When all agents in the network perform primal and dual gradient-type updates, that is, $P^k = Q^k = I_{nd}$, the compact form (12) of Algorithm 1 at iteration k is as follows,

$$\begin{aligned} \mathbf{x}^{k+1} &= \mathbf{x}^k - A(\nabla f(\mathbf{x}^k) + W\boldsymbol{\lambda}^k + \mu W\mathbf{x}^k), \\ \boldsymbol{\lambda}^{k+1} &= \boldsymbol{\lambda}^k + BW\mathbf{x}^k. \end{aligned} \tag{13}$$

This recovers the Arrow-Hurwicz-Uzawa method [34]. For convenience, we refer to updates in (13) as DISH-G. We remark that some exact distributed first-order methods with gradient tracking techniques, such as EXTRA [12], DIGing [13], and [14], are also equivalent to primal-dual gradient-type methods similar to DISH-G [21]. The only difference between these methods and DISH-G occurs in different choices of consensus constraints or penalty terms used in the augmented Lagrangian, or whether the dual step adopts the previous primal variable x^k or the updated x^{k+1} (also referred to as Jacobi or Gauss-Seidel updates).

While such gradient-type primal-dual methods lead to simple distributed implementation, they suffer from slow convergence due to their first-order nature. This motivates us to involve Newton-type updates in DISH as a speedup.

2) Primal-Newton-Dual-Gradient Method (ESOM [20]): ESOM is a second-order method that each iteration approximates a Newton's step by an inner loop in the primal space and a gradient ascent step in the dual space. ESOM-0 is a variant of ESOM without the primal inner loop and can be viewed as a special case of DISH with a different choice of positive definite update matrix. In particular, we define $P^k_{\text{ESOM}} = \nabla^2 f(\mathbf{x}^k) + \mu(I_n - \text{diag}(Z)) \otimes I_d$, which is a diagonal approximation of the primal Hessian $\nabla^2_{\mathbf{xx}} L_\mu(\mathbf{x}^k, \boldsymbol{\lambda}^k) = \nabla^2 f(\mathbf{x}^k) + \mu W$ when $\mu > 0$ and exact when $\mu = 0$. When all agents in DISH perform primal Newton-type and dual gradient-type updates with $P^k = P^k_{\text{ESOM}}$ and $Q^k = I_{nd}$, respectively, the updates of DISH in (12) at iteration k is

$$\begin{aligned} \mathbf{x}^{k+1} &= \mathbf{x}^k - AP^k_{\text{ESOM}}(\nabla f(\mathbf{x}^k) + W\boldsymbol{\lambda}^k + \mu W\mathbf{x}^k), \\ \boldsymbol{\lambda}^{k+1} &= \boldsymbol{\lambda}^k + BW\mathbf{x}^k, \end{aligned}$$

which coincide with ESOM-0. Other variants of ESOM iteratively approximates the off-diagonal parts of the primal Hessian. ESOM enjoys the speedup brought by the primal Hessian's information. Our numerical study shows that DISH with Newton-type updates in both primal and dual spaces can further benefit from the dual Hessian's information.

C. DISH-N as an Approximated Newton's Method

Now we take a close inspection of DISH when all agents in the network always take both primal and dual Newton-type updates. In this case, DISH in (12) shows as follows,

$$\begin{aligned} \mathbf{x}^{k+1} &= \mathbf{x}^k - A(H^k)^{-1}(\nabla f(\mathbf{x}^k) + W\boldsymbol{\lambda}^k + \mu W\mathbf{x}^k), \\ \boldsymbol{\lambda}^{k+1} &= \boldsymbol{\lambda}^k + BH^kW\mathbf{x}^k, \end{aligned} \tag{14}$$

where $H^k = \nabla^2 f(\mathbf{x}^k) + \mu I_{nd} = \text{diag}\{\nabla^2 f_1(x_1^k) + \mu I_d, \cdots, \nabla^2 f_n(x_n^k) + \mu I_d\}$ approximates the primal Hessian $\nabla^2_{\mathbf{xx}} L_\mu(\mathbf{x}^k, \boldsymbol{\lambda}^k)$. We will refer to updates in (14) as DISH-N. In the sequel, we present that DISH-N can be viewed as an approximation of a Newton's method that takes Newton's steps in both the primal and the dual spaces.

Primal Update. The primal Newton's step for solving the inner problem $\min_{\mathbf{x}} L(\mathbf{x}, \boldsymbol{\lambda}^k)$ in (7) at iteration k is as follows,

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \big(\nabla^2_{\mathbf{xx}} L(\mathbf{x}^k, \boldsymbol{\lambda}^k)\big)^{-1} \nabla_{\mathbf{x}} L(\mathbf{x}^k, \boldsymbol{\lambda}^k). \tag{15}$$

We note that when $\mu = 0$, the primal update in (14) coincides with the exact Newton's step in (15). As when $\mu > 0$, the primal Hessian $\nabla^2_{\mathbf{xx}} L(\mathbf{x}^k, \boldsymbol{\lambda}^k) = \nabla^2 f(\mathbf{x}^k) + \mu W$ is nonseparable due to the penalty term μW, which makes it difficult to compute the exact Hessian inverse in a distributed way. Here we approximate $W = (I_n - Z) \otimes I_d$ by the identity I_{nd}. We remark that $(I_n - \text{diag}(Z)) \otimes I_d$ used in P^k_{ESOM} is another approximation of W. Adopting either of them is guaranteed to provide linear convergence rate by Theorem 7. By substituting $H^k = \nabla^2 f(\mathbf{x}^k) + \mu I_{nd}$ in (15), we obtain the Newton-type primal update in (14).

Dual Update. Now we consider the dual Newton's update for $\max_{\boldsymbol{\lambda}} g(\boldsymbol{\lambda})$ in (7) at iteration k. Since we cannot get the exact primal minimizer $\mathbf{x}^*(\boldsymbol{\lambda}^k)$ used in $\nabla g(\boldsymbol{\lambda}^k)$ and $\nabla^2 g(\boldsymbol{\lambda}^k)$ by Lemma 3, we replace $\mathbf{x}^*(\boldsymbol{\lambda}^k)$ by the current primal iterate $\mathbf{x}^k$ and define $\widehat{\nabla} g(\boldsymbol{\lambda}^k)$ and $\widehat{\nabla}^2 g(\boldsymbol{\lambda}^k)$ as estimators of $\nabla g(\boldsymbol{\lambda}^k)$ and $\nabla^2 g(\boldsymbol{\lambda}^k)$, respectively, as follows,

$$\widehat{\nabla} g(\boldsymbol{\lambda}^k) = W\mathbf{x}^k, \;\; \widehat{\nabla}^2 g(\boldsymbol{\lambda}^k) = -W(\nabla^2_{\mathbf{xx}} L(\mathbf{x}^k, \boldsymbol{\lambda}^k))^{-1} W.$$

We remark that $\widehat{\nabla}^2 g(\boldsymbol{\lambda}^k)$ is not full-rank due to the matrix W. We denote by $\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \Delta\boldsymbol{\lambda}^k$ the dual update at iteration k, where $\Delta\boldsymbol{\lambda}^k$ is an approximated dual Newton's step satisfying

$$\widehat{\nabla}^2 g(\boldsymbol{\lambda}^k) \Delta\boldsymbol{\lambda}^k = \widehat{\nabla} g(\boldsymbol{\lambda}^k). \tag{16}$$

We define a Hessian weighted average of local primal variables y^k as follows,

$$y^k = \big(\sum_{i\in\mathcal{N}} \nabla^2 f_i(x_i^k)\big)^{-1} \sum_{i\in\mathcal{N}} \nabla^2 f_i(x_i^k) x_i^k. \tag{17}$$

Now we introduce a lemma to characterize $\Delta\boldsymbol{\lambda}^k$ using y^k.

Lemma 4. *Under Assumption 2, with y^k defined in (17), the dual Newton's step $\Delta\boldsymbol{\lambda}^k$ in (16) satisfies*

$$W\Delta\boldsymbol{\lambda}^k = \nabla^2_{\mathbf{xx}} L(\mathbf{x}^k, \boldsymbol{\lambda}^k)\left(\mathbb{1}_n \otimes y^k - \mathbf{x}^k\right).$$

We note that calculating y^k in (17) directly is impractical in a distributed manner since communicating $d \times d$ local Hessians can be prohibitively expensive. Thus, at the i^{th} agent, we estimate y^k by the weighted average of x_j^k from its neighbors in the network, that is, $\sum_{j\in\mathcal{N}} z_{ij}x_j^k = [(Z \otimes I_d)x^k]_i$. Such estimators are more accurate when either the local Hessians are similar to each other or the local decision variables x_i^k are close to consensus. This includes the scenarios when the original problem is generated by an empirical risk minimization problem with i.i.d. samples at each node, or when the underlying graph has good algebraic connectivity or when the method is close to its limit point (a consensed point).

If we substitute $(Z \otimes I_d)\mathbf{x}^k$ and $H^k = \nabla^2 f(\mathbf{x}^k) + \mu I_{nd}$ as estimators of $\mathbb{1}_n \otimes y^k$ and $\nabla^2_{\mathbf{xx}} L(\mathbf{x}^k, \boldsymbol{\lambda}^k)$ in Lemma 4, respectively, we obtain an estimator $\Delta\widehat{\boldsymbol{\lambda}}^k$ of $\Delta\boldsymbol{\lambda}^k$ satisfying

$$W\Delta\widehat{\boldsymbol{\lambda}}^k = H^k((Z \otimes I_d)\mathbf{x}^k - \mathbf{x}^k) = -H^k W\mathbf{x}^k.$$

We highlight that only $W\boldsymbol{\lambda}^k$ is used in the primal update in (14). Therefore, we do not need to compute $\boldsymbol{\lambda}$ accurately, but rather focus on approximating $W\boldsymbol{\lambda}^k$ and $W\Delta\boldsymbol{\lambda}^k$ instead. In order to ensure that $W\Delta\boldsymbol{\lambda}^k$ lies in the subspace range(W), we introduce an additional communication round and use $W^2\Delta\widehat{\boldsymbol{\lambda}}^k = -WH^kW\mathbf{x}^k$ as an estimator of $W\Delta\widehat{\boldsymbol{\lambda}}^k$. We remark that such estimation is exact under a complete graph with $Z = \mathbb{1}_n\mathbb{1}_n^T/n$, and it is more accurate if the underlying graph is a closer-to-complete one with all eigenvalues of Z closer to either 1 or 0. Thus, by substituting the estimator $W^2\Delta\widehat{\boldsymbol{\lambda}}^k$, we obtain the dual update,

$$\begin{aligned} W\boldsymbol{\lambda}^{k+1} &= W\boldsymbol{\lambda}^k - \beta_2 W^2\Delta\widehat{\boldsymbol{\lambda}}^k \\ &= W\boldsymbol{\lambda}^k + \beta_2 WH^kW\mathbf{x}^k. \end{aligned}$$

There are multiple equivalent $\boldsymbol{\lambda}$ solutions satisfying the above equation, all corresponding to the same primal update. One of these equivalent solutions can be obtained by omitting W on both sides, which leads to the Newton-type dual update in (14). Thus, DISH-N with distributed implementation approximates the primal-dual Newton's method. Previous works [30], [32] have shown that the primal-dual Newton's method improves the convergence performance by utilizing second-order information. Thus, DISH-N convergences efficiently when the approximations are good, i.e., with i.i.d. data distribution among agents for an empirical risk minimization problem and/or a closer-to-complete graph for the underlying topology.

IV. Theoretical Convergence

In this section, we present the linear convergence rate for DISH in Algorithm 1, regardless of agents' choices of gradient-type or Newton-type updates.

Properties of Primal and Dual Functions. We first show some pivotal properties of the primal function $L(\cdot, \boldsymbol{\lambda})$.

Lemma 5. *Under Assumption 2, for any $\boldsymbol{\lambda} \in \mathbb{R}^{nd}$, $L(\cdot, \boldsymbol{\lambda})$ is s-strongly convex and ℓ_L-Lipschitz smooth with $\ell_L = \ell + 2\mu$.*

Now we show the properties of the dual function $g(\boldsymbol{\lambda})$. In particular, we denote by Λ^{OPT} the dual optimal set to Problem 7, that is, for any $\boldsymbol{\lambda}^* \in \Lambda^{\mathsf{OPT}}$, we have $g(\boldsymbol{\lambda}^*) = \max_{\boldsymbol{\lambda}} g(\boldsymbol{\lambda})$. The next lemma shows the properties of g.

Lemma 6. *Under Assumption 2, the dual function $g(\cdot)$ is ℓ_g-Lipschitz smooth and it satisfies the PL inequality [35] with $\boldsymbol{\lambda}^* \in \Lambda^{OPT}$ that*

$$g(\boldsymbol{\lambda}^*) - g(\boldsymbol{\lambda}) \le \frac{1}{2p_g}\|\nabla g(\boldsymbol{\lambda})\|^2,$$

where constants $p_g = (1-\gamma)/(\ell + 2\mu)$ and $\ell_g = 4/s$.

Merit Function. Now we introduce the merit function used in the analysis. We first define two performance metrics, the dual optimality gap and the primal tracking error, as follows,

$$\begin{aligned} \Delta^k_{\boldsymbol{\lambda}} &= g(\boldsymbol{\lambda}^*) - g(\boldsymbol{\lambda}^k), \\ \Delta^k_{\mathbf{x}} &= L(\mathbf{x}^k, \boldsymbol{\lambda}^k) - L(\mathbf{x}^*(\boldsymbol{\lambda}^k), \boldsymbol{\lambda}^k), \end{aligned} \tag{18}$$

where $x^*(\boldsymbol{\lambda})$ is defined in (8) and $\boldsymbol{\lambda}^* \in \Lambda^{\mathsf{OPT}}$ is a dual optimal point. We remark that $\Delta^k_{\mathbf{x}}$ and $\Delta^k_{\boldsymbol{\lambda}}$ are both nonnegative by definition. Now we define a merit function to be used in the analysis by combing the performance metrics in (18) as

$$\Delta^k = 9\Delta^k_{\boldsymbol{\lambda}} + \Delta^k_{\mathbf{x}}. \tag{19}$$

We remark that $\Delta^k \ge 0$ for all $k \ge 0$. We define $\mathbf{x}^{\mathsf{OPT}}$ as the optimal solution of Problem 4. The strong duality implies that $\mathbf{x}^*(\boldsymbol{\lambda}^*) = \mathbf{x}^{\mathsf{OPT}}$ for any $\boldsymbol{\lambda}^* \in \Lambda^{\mathsf{OPT}}$. We will show that using DISH, Δ^k converges to zero at a linear rate in Theorem 7 and therefore, the primal sequence $\mathbf{x}^k$ goes to the exact optimal solution $\mathbf{x}^{\mathsf{OPT}}$ linearly in Corollary 10.

Linear Convergence of DISH. Now we present the theoretical linear convergence of DISH. We first define constants $0 < \underline{p}_i \le \overline{p}_i$ and $0 < \underline{q}_i \le \overline{q}_i$ as bounds on the positive definite local update matrices P_i^k and Q_i^k for $i \in \mathcal{N}$,

$$\underline{p}_i I_d \preceq P_i^k \preceq \overline{p}_i I_d, \quad \underline{q}_i I_d \preceq Q_i^k \preceq \overline{q}_i I_d. \tag{20}$$

Specifically, with the options of gradient-type and Newton-type updates in (10) and (11), we have $\underline{p}_i = \min\{1, 1/(\ell_i + \mu)\}$, $\overline{p}_i = \max\{1, 1/(m_i + \mu)\}$, $\underline{q}_i = \min\{1, m_i + \mu\}$, and $\overline{q}_i = \max\{1, \ell_i + \mu\}$. For convenience, we also define constants $\underline{\alpha}$ and $\underline{\beta}$ as lower bounds to eigenvalues of matrices AP^k and BQ^k, respectively, as follows,

$$\begin{aligned} \underline{\alpha} &= \min_{i\in\mathcal{N}}\{a_i\underline{p}_i\} \le \min_{i\in\mathcal{N}}\{a_i\theta_{\min}(P_i^k)\} = \theta_{\min}(AP^k), \\ \underline{\beta} &= \min_{i\in\mathcal{N}}\{b_i\underline{q}_i\} \le \min_{i\in\mathcal{N}}\{b_i\theta_{\min}(Q_i^k)\} = \theta_{\min}(BQ^k). \end{aligned} \tag{21}$$

Now we show the main theorem stating that DISH in Algorithm 1 convergence linearly in terms of Δ^k.

Theorem 7 (Linear Convergence of DISH)**.** *For any given $\mu \ge 0$, under Assumption 2, we suppose that the stepsizes $\{a_i, b_i\}_{i\in\mathcal{N}}$ satisfy the following conditions,*

$$\begin{aligned} &0 < a_i \le 1/[2\overline{p}_i(s/16 + \ell + 2\mu)], \\ &0 < b_i \le \min\{s/64, \underline{\alpha}s^2/60\}/\overline{q}_i, \end{aligned} \tag{22}$$

where $\overline{p}_i$ and $\overline{q}_i$ are defined in (20) and $\underline{\alpha}$ is defined in (21). Then for all $k = 0, 1, \cdots, K-1$, the iterates generated from DISH in Algorithm 1 satisfy

$$\Delta^{k+1} \leq (1-\rho)\,\Delta^k,$$

where $\rho = \min\{(1-\gamma)\underline{\beta}/[9(\ell+4\mu)], s\underline{\alpha}/2\}$ with $\underline{\beta}$ defined in (21) and Δ^k is the merit function defined in (19).

Proof Sketch of Theorem 7. Now we sketch the proof of Theorem 7. Due to the coupled nature of primal and dual updates in DISH in (12), our main idea for analyzing the primal-dual framework is to bound the dual optimality gap $\Delta_{\boldsymbol{\lambda}}^k$ and the primal tracking error $\Delta_{\mathbf{x}}^k$ through coupled inequalities. We decompose our analysis into three steps.

Step 1: Bounding the Dual Optimality Gap $\Delta_{\boldsymbol{\lambda}}^{k+1}$. We first bound the updated dual optimality gap $\Delta_{\boldsymbol{\lambda}}^{k+1}$ with an alternative primal tracking error $\|\nabla_{\mathbf{x}} L(\mathbf{x}^k, \boldsymbol{\lambda}^k)\|^2$ using the Lipschitz smoothness of the dual function g. For convenience, we define a constant β as an upper bound of $\|BQ^k\|$ as follows,

$$\beta = \max_{i\in\mathcal{N}}\{b_i\overline{q}_i\} \geq \max_{i\in\mathcal{N}}\{b_i\|Q_i^k\|\} = \|BQ^k\|. \tag{23}$$

The following proposition shows the obtained inequality.

Proposition 8. *Under Assumption 2, given a constant $\mu > 0$ and stepsizes $\{a_i, b_i\}_{i\in\mathcal{N}} > 0$, for all $k = 0, 1, \cdots, K-1$, the iterates generated from DISH in Algorithm 1 satisfy*

$$\Delta_{\boldsymbol{\lambda}}^{k+1} \leq \Delta_{\boldsymbol{\lambda}}^k - \left(\frac{1}{2} - \beta\ell_g\right)\left\|\nabla g(\boldsymbol{\lambda}^k)\right\|_{BQ^k}^2 + \left(\frac{1}{2} + \beta\ell_g\right)\frac{4\beta}{s^2}\left\|\nabla_{\mathbf{x}} L(\mathbf{x}^k, \boldsymbol{\lambda}^k)\right\|^2,$$

where β is defined in (23) and ℓ_g is defined in Lemma 6.

Step 2: Bounding the Primal Tracking Error $\Delta_{\mathbf{x}}^{k+1}$. Next, we derive a bound of the updated primal tracking error $\Delta_{\mathbf{x}}^{k+1}$ by an alternative dual optimality gap $\nabla g(\boldsymbol{\lambda}^k)$. We use the Lipschitz smoothness of $L(\cdot, \boldsymbol{\lambda})$ to show the following result.

Proposition 9. *Under Assumption 2, given a constant $\mu > 0$ and stepsizes $\{a_i, b_i\}_{i\in\mathcal{N}} > 0$, for all $k = 0, 1, \cdots, K-1$, the iterates generated from DISH in Algorithm 1 satisfy*

$$\Delta_{\mathbf{x}}^{k+1} \leq \Delta_{\mathbf{x}}^k + 3\|\nabla g(\boldsymbol{\lambda}^k)\|_{BQ^k}^2 - \|\nabla_{\mathbf{x}} L(\mathbf{x}^k, \boldsymbol{\lambda}^k)\|_{D^k}^2 + \Delta_{\boldsymbol{\lambda}}^k - \Delta_{\boldsymbol{\lambda}}^{k+1},$$

where matrix $D^k = AP^k - (2\beta + \ell_L/2)A^2(P^k)^2 - 12\beta/s^2 I_{nd}$ with β and ℓ_L defined in (23) and Lemma 5, respective.

Step 3: Putting Things Together. Finally, we take a linear combination of the coupled inequalities in Propositions 8 and 9. We use the strong convexity of $L(\cdot, \boldsymbol{\lambda})$ and the PL inequality satisfied by $g(\boldsymbol{\lambda})$ in Lemmas 5 and 6. By some algebraic manipulations, when the stepsizes satisfy (22), we prove the linear convergence of DISH in Theorem 7. □

The following corollary shows that DISH finds the optimal solution $\mathbf{x}^{\mathsf{OPT}}$ at a linear rate.

Corollary 10. *Under Assumption 2, when the stepsizes $\{a_i, b_i\}_{i\in\mathcal{N}}$ satisfy conditions in (22), for all $k = 0, \cdots, K-1$, the iterates generated from DISH in Algorithm 1 satisfy*

$$\|\mathbf{x}^k - \mathbf{x}^{\mathsf{OPT}}\|^2 \leq c(1-\rho)^k,$$

where the constant $c = 4\ell_L\Delta^0/[s\cdot\min\{\ell_L, 9s\}]$ with ℓ_L defined in Lemma 5.

Theorem 7 shows a linear (Q-linear) convergence rate of DISH in terms of the merit function Δ^k, regardless of agents' choices of gradient-type or Newton-type updates. Corollary 10 guarantees that DISH converges linearly to the exact optimal solution $\mathbf{x}^{\mathsf{OPT}}$. We can also show that the dual sequence goes to the optimal point. We remark that since only constants $\underline{p}_i$, $\overline{p}_i$, $\underline{q}_i$, $\overline{q}_i$ defined in (20) are needed in Theorem 7, agents can adopt any local updates as long as the update matrices are positive definite with uniform upper and lower bounds.

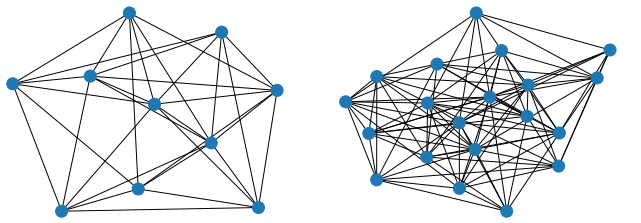
The provable linear rate of DISH recovers the linear rate of some existing distributed methods with exact convergence. Such methods include gradient-type methods such as EXTRA [12], DIGing [13], and [14] and other methods that adopt Newton or quasi-Newton information like PD-QN [28] and ESOM [20] as discussed in Section III-B. We remark that when all agents take Newton-type updates in both primal and dual spaces, DISH does not give faster than linear rate. This is due to the distributed approximations made in both primal and dual Newton steps.

The linear rate $1-\rho$ in Theorem 7 depends on the network structure γ, objective function properties ℓ and s, the augmentation penalty μ, and the worst case of update matrices $\underline{\alpha}$ and $\underline{\beta}$. Although the theorem is conservative relying on the worst agents' updates, as numerical experiments will show in Section V, DISH can achieve faster performance when more agents adopt Newton-type updates since the local information is more fully utilized.

V. Numerical Experiments

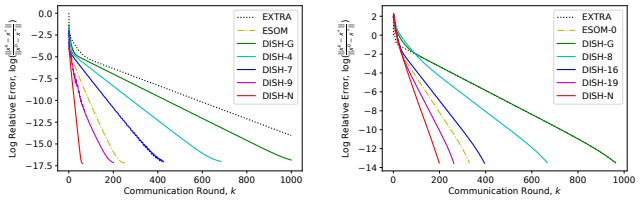
In this section, we present numerical studies of DISH on convex distributed empirical risk minimization problems including linear least squares and binary classifications. All the experiments are conducted on 3.30GHz Intel Core i9 CPUs, Ubuntu 20.04.2, in Python 3.8.5. Our code is publicly available at https://github.com/xiaochunniu/DISH.

Experimental Setups. We evaluate all methods on two setups, both with synthetic data. In each setup, the underlying network is randomly generated by the Erdős-Rényi model with n nodes (agents) and probability p to generate each edge. We denote by δ_i the degree of node i and $\delta_{\max} = \max_{i\in\mathcal{N}}\{\delta_i\}$ the largest degree of the network. We define the elements of the consensus matrix Z as $z_{ij} = 1/(\delta_{\max}+1)$ for $\{i,j\} \in \mathcal{E}$, $z_{ii} = 1 - \delta_i/(\delta_{\max}+1)$ for $i \in \mathcal{N}$, and $z_{ij} = 0$ otherwise. In each setup, the decision variable is d-dimensional and there are total amount $N = \sum_{i\in\mathcal{N}} N_i$ of data in the network with the local dataset size N_i at agent i. Here are more details of the setups.



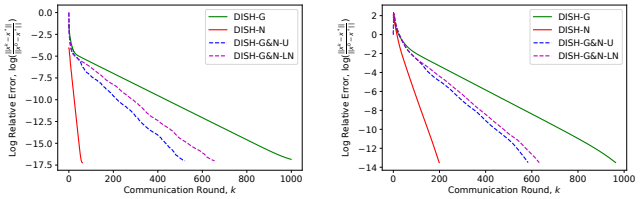
(a) Least Squares in Setup 1 (b) Logistic Regression in Setup 2

Fig. 2. Underlying Networks.



(a) Least Squares in Setup 1 (b) Logistic Regression in Setup 2

Fig. 3. Performance of EXTRA, ESOM-0, and DISH.



(a) Least Squares in Setup 1 (b) Logistic Regression in Setup 2

Fig. 4. Performance of DISH-G&N.

Setup 1: Decentralized Linear Least Squares over a Random Graph. We first consider the decentralized regularized linear least squares problem as follows,

$$\min_{\omega\in\mathbb{R}^d} \frac{1}{2N}\sum_{i=1}^{n} \|A_i\omega - y_i\|^2 + \frac{\rho}{2}\|\omega\|^2,$$

where $A_i \in \mathbb{R}^{N_i\times d}$ and $y_i \in \mathbb{R}^{N_i}$ are the feature matrix and the response vector at agent i, respectively, and $\rho \geq 0$ is the penalty parameter. Specifically, we set $n = 10$, $p = 0.7$, $d = 5$, $N_i = 50$ for all $i \in \mathcal{N}$, and $\rho = 1$. We generate matrices $\widehat{A}_i \in \mathbb{R}^{50\times 5}$, noise vectors $v_i \in \mathbb{R}^{50}$ for $i \in \mathcal{N}$, and a vector $\omega_0 \in \mathbb{R}^5$ from standard Normal distributions. We set feature matrices $A_i = \widehat{A}_i\Theta$, where $\Theta = \mathrm{diag}\{10, 10, 0.1, 0.1, 0.1\} \in \mathbb{R}^{5\times 5}$ is the scaling matrix. We generate the response vector $y_i \in \mathbb{R}^{50}$ by the formula $y_i = A_i\omega_0 + v_i$ for $i \in \mathcal{N}$.

Setup 2: Decentralized Logistic Regression over a Random Graph. The second setup studies the regularized logistic regression model for solving binary classification problems

$$\min_{\omega\in\mathbb{R}^d} \frac{1}{N}\sum_{i=1}^{n} \big[-y_i^\intercal \log h_i - (1-y_i)^\intercal \log(1-h_i)\big] + \frac{\rho}{2}\|\omega\|^2,$$

where $h_i = 1/(1+\exp(-A_i\omega))$, $A_i \in \mathbb{R}^{N_i\times d}$, and $y_i \in \mathbb{R}^{N_i}$ are the known feature matrix and label vector at agent i, respectively, and ρ is the penalty parameter. Specifically, we set $n = 20$, $p = 0.5$, $d = 3$, $N_i = 50$ for all $i \in \mathcal{N}$, and $\rho = 1$. We generate matrices $\widehat{A}_i \in \mathbb{R}^{50\times 3}$, noise vectors $v_i \in \mathbb{R}^{50}$ for $i \in \mathcal{N}$, and a vector $\omega_0 \in \mathbb{R}^3$ from Normal distributions. We scale $\widehat{A}_i$ with matrix $\Theta = \mathrm{diag}\{10, 0.1, 0.1\} \in \mathbb{R}^{3\times 3}$ and set feature matrices to be $A_i = \widehat{A}_i\Theta$. The response vector $y_i \in \mathbb{R}^{50}$ is generated by $y_i = \mathrm{argmax}(\mathrm{softmax}(A_i\omega_0 + v_i))$. The generated underlying networks are shown in Figure 2.

Implemented Methods. We implement EXTRA [12], ESOM-0 [20], and DISH in Algorithm 1 on the introduced two setups. For convenience, we denote by DISH-K DISH with K agents performing Newton-type updates and the other agents performing gradient-type updates all the time. Moreover, we represent DISH-G&N as DISH with all agents switching between gradient-type and Newton-type updates once in a while. In particular, for DISH-G&N-U and DISH-G&N-LN, we generate $t_i \sim U[5, 50]$ and $t_i \sim \mathrm{lognormal}(2, 4) + 30$, respectively. In both cases, we let agent i change its updates type every t_i iterations with the initial updates uniformly sampled from {'gradient-type', 'Newton-type'}. We remark that all these methods require one communication round with the same communication costs for each iteration independent of the update type.

For all setups and methods, we tune stepsizes and parameters by grid search in the range $[2^{-6}, 2^4]$ and select the optimal ones that minimize the number of iterations to reach a predetermined relative error threshold, measured by $\|\mathbf{x}^k - \mathbf{x}^{\mathsf{OPT}}\|/\|\mathbf{x}^0 - \mathbf{x}^{\mathsf{OPT}}\|$, where the optimal point $\mathbf{x}^{\mathsf{OPT}}$ is obtained by a centralized solver for Problem 1. We remark that in DISH we fix $a_i = 1$ when agent i takes Newton-type updates to mimic primal Newton's step.

Results and Conclusions. In both Figures 3 and 4, the x-axis shows the number of communication rounds (iterations) and the y-axis is the logarithm of the relative error. As shown in Figures 3 and 4, it is clear that the DISH framework has a linear convergence performance regardless of agents' choice of gradient-type and Newton-type updates, which validates the theoretical guarantees in Theorem 7.

As shown in Figure 3, primal-dual gradient-type methods, EXTRA and DISH-G, perform similarly due to their similar update formulas. However, when some agents take Newton-type updates, DISH improves the overall training speed and outperforms the baseline method DISH-G consistently. In particular, the second-order DISH-N method outperforms ESOM-0 in many scenarios, implying DISH-N benefits from the dual Hessian approximation.

Moreover, as the number of agents that perform Newton-type updates, K, increases, the numerical convergence of DISH is likely to become faster since the Hessian information can be more fully utilized. This observation suggests that in practical systems, those agents with higher computational capabilities and/or cheaper costs to perform computation can choose to take Newton-type updates locally to help speed up the overall convergence of the whole system.

In traditional distributed optimization algorithms, all agents perform the same type of updates. The complexity of the method is determined by the agents equipped with the worst computation hardware. While in DISH, since efficient Newton-type updates are involved at parts of the network, the overall system enjoys a faster convergence speed compared to systems running gradient-type methods only. Therefore, we can maximally leverage the parallel heterogeneous computation capabilities in this setting.

VI. Final Remarks and Future Work

This paper proposes DISH as a distributed hybrid primal-dual algorithmic framework allowing agents to perform either gradient-type or Newton-type updates based on their computation capacities. We show a linear convergence rate of DISH for strongly convex functions. Numerical studies are provided to demonstrate the efficacy of DISH in practice.

We highlight a few interesting directions for future works on the DISH framework and distributed optimization. First, we expect DISH to be generalized to broader settings like time-varying graphs or systems with non-convex objective functions. Also, we could involve stochastic methods in DISH. For instance, agents could perform stochastic gradient-type or subsampled Newton-type methods locally. Moreover, we could consider asynchronous updates in DISH, where at each communication round, only a randomly selected subset of the agents take computation steps since it is possible that only a few agents are active in practice.

References

[1] G. B. Giannakis, V. Kekatos, N. Gatsis, S.-J. Kim, H. Zhu, and B. F. Wollenberg, "Monitoring and optimization for power grids: A signal processing perspective," *IEEE Signal Processing Magazine*, vol. 30, no. 5, pp. 107–128, 2013.

[2] F. Dörfler, J. W. Simpson-Porco, and F. Bullo, "Breaking the hierarchy: Distributed control and economic optimality in microgrids," *IEEE Transactions on Control of Network Systems*, vol. 3, no. 3, pp. 241–253, 2015.

[3] Q. Ling and Z. Tian, "Decentralized sparse signal recovery for compressive sleeping wireless sensor networks," *IEEE Transactions on Signal Processing*, vol. 58, no. 7, pp. 3816–3827, 2010.

[4] I. D. Schizas, A. Ribeiro, and G. B. Giannakis, "Consensus in ad hoc wsns with noisy links—part i: Distributed estimation of deterministic signals," *IEEE Transactions on Signal Processing*, vol. 56, no. 1, pp. 350–364, 2007.

[5] F. Lamnabhi-Lagarrigue, A. Annaswamy, S. Engell, A. Isaksson, P. Khargonekar, R. M. Murray, H. Nijmeijer, T. Samad, D. Tilbury, and P. Van den Hof, "Systems & control for the future of humanity, research agenda: Current and future roles, impact and grand challenges," *Annual Reviews in Control*, vol. 43, pp. 1–64, 2017.

[6] T. Yang, X. Yi, J. Wu, Y. Yuan, D. Wu, Z. Meng, Y. Hong, H. Wang, Z. Lin, and K. H. Johansson, "A survey of distributed optimization," *Annual Reviews in Control*, vol. 47, pp. 278–305, 2019.

[7] A. H. Sayed, "Adaptation, learning, and optimization over networks," *Foundations and Trends in Machine Learning*, vol. 7, no. ARTICLE, pp. 311–801, 2014.

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

[9] J. Konečnỳ, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," *arXiv preprint arXiv:1610.02527*, 2016.

[10] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and distributed computation: numerical methods*. Prentice hall Englewood Cliffs, NJ, 1989, vol. 23.

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

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

[13] A. Nedic, A. Olshevsky, and W. Shi, "Achieving geometric convergence for distributed optimization over time-varying graphs," *SIAM Journal on Optimization*, vol. 27, no. 4, pp. 2597–2633, 2017.

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

[15] O. Shamir, N. Srebro, and T. Zhang, "Communication-efficient distributed optimization using an approximate newton-type method," in *International conference on machine learning*. PMLR, 2014, pp. 1000–1008.

[16] Y. Zhang and X. Lin, "Disco: Distributed optimization for self-concordant empirical loss," in *International conference on machine learning*. PMLR, 2015, pp. 362–370.

[17] S. Wang, F. Roosta, P. Xu, and M. W. Mahoney, "Giant: Globally improved approximate newton method for distributed optimization," *Advances in Neural Information Processing Systems*, vol. 31, pp. 2332–2342, 2018.

[18] R. Crane and F. Roosta, "Dingo: Distributed newton-type method for gradient-norm optimization," *arXiv preprint arXiv:1901.05134*, 2019.

[19] T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang, and Z. Zhang, "Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems," *arXiv preprint arXiv:1512.01274*, 2015.

[20] A. Mokhtari, W. Shi, Q. Ling, and A. Ribeiro, "A decentralized second-order method with exact linear convergence rate for consensus optimization," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, no. 4, pp. 507–522, 2016.

[21] D. Jakovetić, "A unification and generalization of exact distributed first-order methods," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 5, no. 1, pp. 31–46, 2018.

[22] A. Mokhtari, Q. Ling, and A. Ribeiro, "Network newton," in *2014 48th Asilomar Conference on Signals, Systems and Computers*, 2014, pp. 1621–1625.

[23] R. Tutunov, H. Bou-Ammar, and A. Jadbabaie, "Distributed newton method for large-scale consensus optimization," *IEEE Transactions on Automatic Control*, vol. 64, no. 10, pp. 3983–3994, 2019.

[24] C. Sun, M. Ye, and G. Hu, "Distributed optimization for two types of heterogeneous multiagent systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 3, pp. 1314–1324, 2021.

[25] S. Boyd, N. Parikh, and E. Chu, *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc, 2011.

[26] Y. Wang, W. Yin, and J. Zeng, "Global convergence of admm in nonconvex nonsmooth optimization," *Journal of Scientific Computing*, vol. 78, no. 1, pp. 29–63, 2019.

[27] V. Smith, S. Forte, M. Chenxin, M. Takáč, M. I. Jordan, and M. Jaggi, "Cocoa: A general framework for communication-efficient distributed optimization," *Journal of Machine Learning Research*, vol. 18, p. 230, 2018.

[28] M. Eisen, A. Mokhtari, and A. Ribeiro, "A primal-dual quasi-newton method for exact consensus optimization," *IEEE Transactions on Signal Processing*, vol. 67, no. 23, pp. 5983–5997, 2019.

[29] X. Niu and E. Wei, "Fedhybrid: A hybrid primal-dual algorithm framework for federated optimization," *arXiv preprint arXiv:2106.01279*, 2021.

[30] D. P. Bertsekas, *Constrained optimization and Lagrange multiplier methods*. Academic press, 2014.

[31] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[32] R. A. Tapia, "Diagonalized multiplier methods and quasi-newton methods for constrained optimization," *Journal of Optimization Theory and Applications*, vol. 22, no. 2, pp. 135–194, 1977.

[33] J. Nocedal and S. J. Wright, *Numerical optimization*. Springer, 1999.

[34] K. Arrow and L. Hurwicz, "H. uzawa—studies in nonlinear programming," 1958.

[35] H. Karimi, J. Nutini, and M. Schmidt, "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, 2016, pp. 795–811.