2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Communication-efficient distributed optimization with adaptability to system heterogeneity

Ziyi Yu and Nikolaos M. Freris

***Abstract*— We consider the setting of agents cooperatively minimizing the sum of local objectives plus a regularizer on a graph. This paper proposes a primal-dual method in consideration of three distinctive attributes of real-life multi-agent systems, namely: (i) expensive communication, (ii) lack of synchronization, and (iii) system heterogeneity. In specific, we propose a distributed asynchronous algorithm with minimal communication cost, in which users commit variable amounts of local work on their respective sub-problems. We illustrate this both theoretically and experimentally in the machine learning setting, where the agents hold private data and use a stochastic Newton method as the local solver. Under standard assumptions on Lipschitz continuous gradients and strong convexity, our analysis establishes linear convergence in expectation and characterizes the dependency of the rate on the number of local iterations. We proceed a step further to propose a simple means for tuning agents' hyperparameters locally, so as to adjust to heterogeneity and accelerate the overall convergence. Last, we validate our proposed method on a benchmark machine learning dataset to illustrate the merits in terms of computation, communication, and run-time saving as well as adaptability to heterogeneity.**

I. INTRODUCTION

The resurgence of distributed optimization [1] is underscored by the vast numbers of smart devices (phones, sensors, etc.) that collect massive volumes of data in a distributed fashion. It has concise advantages compared to its centralized counterpart in terms of balancing computation and communication costs as well as minimizing delays, and has thus found applications in a broad range of fields, such as in machine learning and data mining [2], [3], signal processing [4], wireless sensor networks [5], and control [6].

The archetypal problem that we consider is to:

$$\underset{\hat{\mathbf{x}} \in \mathbb{R}^d}{\text{minimize}} \quad \frac{1}{n}\sum_{i=1}^{n} f_i(\hat{\mathbf{x}}) + r(\hat{\mathbf{x}}), \tag{1}$$

where n is the number of agents in the network, $f_i(\cdot) : \mathbb{R}^d \to \mathbb{R}$ is a strongly convex and smooth cost function pertaining to agent $i \in [n]$ ($[n] := \{1, \ldots, n\}$), and $r(\cdot)$ is a convex but possibly non-smooth function that serves to impose regularity on the solution. We further pose additional structure on the cost function as motivated by the machine learning framework. In specific, the i-th agent possesses a batch of data samples $\{\mathbf{s}_i^j\}_{j=1}^{D_i}$ with volume D_i, whence:

$$f_i(\hat{\mathbf{x}}) := \frac{1}{D_i}\sum_{j=1}^{D_i} l(\mathbf{s}_i^j; \hat{\mathbf{x}}), \tag{2}$$

where $l(\cdot; \hat{\mathbf{x}})$ is the loss function associated with one data sample for a model parameter $\hat{\mathbf{x}}$.

There has been extensive work on the *consensus* framework, especially first-order methods [7]–[12], which are prone to slow convergence in ill-conditioned problems. Second-order information can be used to remedy this [13]–[15], nonetheless, this comes with caveats in terms of extensive message-passing and stringent synchronization requirements.

Our proposed solution falls under the taxonomy of primal-dual methods, in particular, the celebrated *Alternating Direction Method of Multipliers* (ADMM) [16] which has given rise to a multitude of distributed methods [17]–[21] that feature fast convergence and robustness to hyperparameter selection. The computationally challenging component of ADMM lies in solving optimization sub-problems associated with the local objectives. To this end, there has been extensive work on inexact methods, that typically involves a gradient step for the local problems [19], [20]. In contrast, the use of second-order information for obtaining inexact solutions has been unattractive as it requires inner communication loops to approximate the Hessian [22], [23]. This issue was resolved by the DRUID framework [24], which reformulated the problem by introducing intermediate variables, thus allowing for a decomposition that supports (quasi-)Newton local solvers without an increase in communication costs. Nevertheless, the methods in [24] only considered a single local iteration and deterministic local solvers, while the analysis heavily relies on this setting; they are thus unsuitable to capture computational heterogeneity in a real system.

The proposed solution in this paper is motivated by three main features/requirements in large-scale smart systems, namely:

1) *communication is costly* (in view of battery drainage, limited bandwidths, and stringent delay considerations),
2) *synchronous methods are unattractive* (due to agent unavailability caused by varying operating condi-

The authors are with the School of Computer Science, University of Science and Technology of China, Hefei, China
Emails: `yuziyi@mail.ustc.edu.cn, nfr@ustc.edu.cn`
This research was supported by the USTC Research Department under grant WK2150110025. Correspondence to N. Freris.

tions as well as the difficulty of the synchronization problem [25]), and

3) *heterogeneity in computing capabilities* (this allows for variable work loads to be committed based on the device hardware and battery levels).

We propose a distributed asynchronous communication-efficient protocol tailored for variability to system heterogeneity. The contributions are summarized as follows.

Contributions:

1) The proposed method, DRUID-VL, allows an arbitrary subset of agents to participate in each round and features minimal communication costs (a single broadcasting step of the local variable to neighboring agents).
2) It specifically addresses heterogeneity in terms of local work. We illustrate this in the machine learning setting by applying an agent-dependent number of stochastic Newton steps for the local sub-problem.
3) We establish linear convergence in expectation (for arbitrary levels of heterogeneity) and characterize the convergence rate (*Theorem* 1). In particular, our analysis captures the effect of local work on the convergence of the algorithm and allows for a simple method to tune hyperparameters locally to further accelerate the convergence.
4) We experimentally demonstrate on a real-life dataset the communication and computation benefits of our method compared to prior art, and also study the algorithm's gains by adapting to heterogeneity.

II. PRELIMINARIES

A. Problem Formulation

The network topology is captured by an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \ldots, n\}$ is the vertex set and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set, i.e. $(i,j) \in \mathcal{E}$ if agent i can exchange packets with agent j ($\mathcal{N}_i := \{j : (i,j) \in \mathcal{E}\}$ denotes the neighborhood of i). Provided that $\mathcal{G}$ is connected, (1) can be directly recast in the following consensus setting:

$$\begin{aligned} \underset{\mathbf{x}_i, \boldsymbol{\theta}, \mathbf{z}_{ij}}{\text{minimize}} \quad & \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x}_i) + r(\boldsymbol{\theta}), \\ \text{subject to} \quad & \mathbf{x}_i = \mathbf{z}_{ij} = \mathbf{x}_j, \quad \forall (i,j) \in \mathcal{E}, \\ & \mathbf{x}_q = \boldsymbol{\theta}, \quad \text{for one arbitrary } q \in [n], \end{aligned} \tag{3}$$

where we introduce θ to separate the smooth and non-smooth functions in (1), and select (arbitrarily) the q-th agent to enforce the equality constraint as $\mathbf{x}_q = \boldsymbol{\theta}$. By defining the source and destination matrices $\hat{A}_s, \hat{A}_d \in \mathbb{R}^{|\mathcal{E}| \times n}$: $[\hat{A}_s]_{ki} = [\hat{A}_d]_{kj} = 1$ if (i,j) is the k-th element in $\mathcal{E}$ and all other entries are zero, stacking $\mathbf{x}_i, \mathbf{z}_{ij} \in \mathbb{R}^d$ into column vectors $\mathbf{x} \in \mathbb{R}^{nd}, \mathbf{z} \in \mathbb{R}^{|\mathcal{E}|d}$, respectively, denoting $f(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x}_i)$, and defining $S := \mathbf{e}_q \otimes I_d \in \mathbb{R}^{nd \times d}$ ($\mathbf{e}_q \in \mathbb{R}^n$ has one at the q-th entry and zero elsewhere), we obtain a compact representation as:

$$\begin{aligned} \underset{\mathbf{x}, \mathbf{z}, \boldsymbol{\theta}}{\text{minimize}} \quad & f(\mathbf{x}) + r(\boldsymbol{\theta}), \\ \text{subject to} \quad & A\mathbf{x} = \begin{bmatrix} \hat{A}_s \otimes I_d \\ \hat{A}_d \otimes I_d \end{bmatrix} \mathbf{x} = \begin{bmatrix} I_{|\mathcal{E}|d} \\ I_{|\mathcal{E}|d} \end{bmatrix} \mathbf{z} = B\mathbf{z}, \\ & S^\top \mathbf{x} = \boldsymbol{\theta}. \end{aligned} \tag{4}$$

We further make the definitions $\hat{E}_u = \hat{A}_s + \hat{A}_d, \hat{E}_s = \hat{A}_s - \hat{A}_d \in \mathbb{R}^{|\mathcal{E}| \times n}$ (unsigned and signed graph incidence matrix), $\hat{L}_u = \hat{E}_u^\top \hat{E}_u, \hat{L}_s = \hat{E}_s^\top \hat{E}_s \in \mathbb{R}^{n \times n}$ (unsigned and signed graph Laplacian matrix), and $\hat{D} = \frac{1}{2}(\hat{L}_u + \hat{L}_s) \in \mathbb{R}^{n \times n}$ (graph degree matrix with entries $D_{ii} = |\mathcal{N}_i|$). Their block extensions are given by $E_u = \hat{E}_u \otimes I_d$, $E_s = \hat{E}_s \otimes I_d$, $L_u = \hat{L}_u \otimes I_d$, $L_s = \hat{L}_s \otimes I_d$, and $D = \hat{D} \otimes I_d$ (where $\otimes$ denotes the Kronecker product and I_d the identity matrix).

***Remark** 1:* It is worth noting that the use of the intermediate variables $\{z_{ij}\}$ is essential for each agent to estimate curvature without information exchange in the neighborhood, leading to a communication-efficient protocol. For more details, we refer the reader to [24, *Remark* 2].

B. ADMM Formulation

The augmented Lagrangian (AL) for (4) is

$$\begin{aligned} \mathcal{L}(\mathbf{x}, \boldsymbol{\theta}, \mathbf{z}; \mathbf{y}, \boldsymbol{\lambda}) := f(\mathbf{x}) + r(\boldsymbol{\theta}) + \mathbf{y}^\top (A\mathbf{x} - B\mathbf{z}) \\ + \boldsymbol{\lambda}^\top (S^\top \mathbf{x} - \boldsymbol{\theta}) + \frac{\mu_z}{2}\|A\mathbf{x} - B\mathbf{z}\|^2 + \frac{\mu_\theta}{2}\|S^\top \mathbf{x} - \boldsymbol{\theta}\|^2. \end{aligned} \tag{5}$$

We apply (3-step) ADMM to (5):

$$\mathbf{x}^{t+1} = \underset{\mathbf{x}}{\operatorname{argmin}}\, \mathcal{L}_\Gamma(\mathbf{x}, \boldsymbol{\theta}^t, \mathbf{z}^t; \mathbf{y}^t, \boldsymbol{\lambda}^t), \tag{6a}$$

$$\boldsymbol{\theta}^{t+1} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, \mathcal{L}(\mathbf{x}^{t+1}, \boldsymbol{\theta}, \mathbf{z}^t; \mathbf{y}^t, \boldsymbol{\lambda}^t), \tag{6b}$$

$$\mathbf{z}^{t+1} = \underset{\mathbf{z}}{\operatorname{argmin}}\, \mathcal{L}(\mathbf{x}^{t+1}, \boldsymbol{\theta}^{t+1}, \mathbf{z}; \mathbf{y}^t, \boldsymbol{\lambda}^t), \tag{6c}$$

$$\mathbf{y}^{t+1} = \mathbf{y}^t + \mu_z(A\mathbf{x}^{t+1} - B\mathbf{z}^{t+1}), \tag{6d}$$

$$\boldsymbol{\lambda}^{t+1} = \boldsymbol{\lambda}^t + \mu_\theta(S^\top \mathbf{x}^{t+1} - \boldsymbol{\theta}^{t+1}). \tag{6e}$$

Note that minimization in (6a) is carried over the perturbed AL:

$$\mathcal{L}_\Gamma(\mathbf{x}, \boldsymbol{\theta}^t, \mathbf{z}^t; \mathbf{y}^t, \boldsymbol{\lambda}^t) := \mathcal{L}(\mathbf{x}, \boldsymbol{\theta}^t, \mathbf{z}^t; \mathbf{y}^t, \boldsymbol{\lambda}^t) + \frac{1}{2}\|\mathbf{x} - \mathbf{x}^t\|_\Gamma^2, \tag{7}$$

where we define the coefficient matrix $\Gamma := \mathbf{diag}\,(\epsilon_1, \ldots, \epsilon_n) \otimes I_d \in \mathbb{R}^{nd \times nd}$, and $\epsilon_i > 0, \forall i \in [n]$. This serves to add robustness by means of *personalized* hyperparameters (ϵ_i for each agent i). The main computational burden resides in the sub-optimization problem (6a), for which obtaining an exact solution would be time-consuming [18]; to remedy this issue, vanilla DRUID [24] adopts a single step of the same deterministic algorithm (GD, Newton, or BFGS) across all agents to solve (6a) inexactly.

Our work serves to bridge the gap between distributed inexact [24] and exact ADMM [18] by deviating from single-step updates and permitting a variable, agent-dependent number of updates (E_i for agent i). This is necessary in order to further exploit local computation resources to attain a better trade-off between computation (more work) and communication (fewer global rounds). For the local solver, we opt for a stochastic Newton method [26] that employs sub-sampled Hessian inverse and stochastic gradient. This is in view of the special structure as in (2), for which stochastic methods have proliferated in problems with *Big Data* (these methods operate by sampling mini-batches of data, and use tools such as gradient tracking and variance reduction to recover linear convergence [26]–[28]).

The following extends [24, *Lemma* 1] to the case of personalized hyperparameters and variable local iterations of stochastic Newton, and is key for an efficient implementation of (6) under agent-specific stochastic Newton steps for sub-problem (6a). Note that below we use a generic local iteration number E for notational simplification but would like to emphasize that the implementation of the algorithm supports freely choosing the local work as E_i.

Lemma 1: Consider (6) with (6a) replaced by E steps of stochastic Newton ($\hat{\nabla}, \hat{\nabla}^2$ denote stochastic gradient/Hessian). Let $\mathbf{y}^t = [\boldsymbol{\alpha}^t; \boldsymbol{\beta}^t]$, $\boldsymbol{\alpha}^t, \boldsymbol{\beta}^t \in \mathbb{R}^{|\mathcal{E}|d}$. If $\mathbf{y}^0$ and $\mathbf{z}^0$ are initialized so that $\boldsymbol{\alpha}^0 + \boldsymbol{\beta}^0 = 0$ and $\mathbf{z}^0 = \frac{1}{2}E_u\mathbf{x}^0$, then $\boldsymbol{\alpha}^t + \boldsymbol{\beta}^t = 0$ and $\mathbf{z}^t = \frac{1}{2}E_u\mathbf{x}^t$ for all $t \geq 0$. Moreover, defining $\boldsymbol{\phi}^t := E_s^\top \boldsymbol{\alpha}^t$, $\mathbf{x}^{t,0} := \mathbf{x}^t$ and $\mathbf{x}^{t,E} := \mathbf{x}^{t+1}$, the updates can be written as:

$$\mathbf{x}^{t,e} = \mathbf{x}^{t,e-1} - \left(\hat{\nabla}^2 f(\mathbf{x}^{t,e-1}) + \mu_z D + \mu_\theta SS^\top + \Gamma\right)^{-1} \Big[\hat{\nabla} f(\mathbf{x}^{t,e-1}) + \boldsymbol{\phi}^t + S\boldsymbol{\lambda}^t + \frac{1}{2}\mu_z L_s \mathbf{x}^t + \Gamma(\mathbf{x}^{t,e-1} - \mathbf{x}^t) + \mu_\theta S(S^\top \mathbf{x}^t - \boldsymbol{\theta}^t)\Big],\ e = 1, \dots, E, \tag{8a}$$

$$\boldsymbol{\theta}^{t+1} = \mathbf{prox}_{r/\mu_\theta}(S^\top \mathbf{x}^{t+1} + \frac{1}{\mu_\theta}\boldsymbol{\lambda}^t), \tag{8b}$$

$$\boldsymbol{\phi}^{t+1} = \boldsymbol{\phi}^t + \frac{1}{2}\mu_z L_s \mathbf{x}^{t+1}, \tag{8c}$$

$$\boldsymbol{\lambda}^{t+1} = \boldsymbol{\lambda}^t + \mu_\theta(S^\top \mathbf{x}^{t+1} - \boldsymbol{\theta}^{t+1}), \tag{8d}$$

where **prox** denotes the proximal mapping [29].

III. ALGORITHM

The updates in (8) can be directly written as a distributed protocol (9), that is amenable to implementation with solely *agent-based* variables. In specific, agent i holds $(\mathbf{x}_i, \boldsymbol{\phi}_i)$ (the q-th agent associated with the regularizer $r(\cdot)$ additionally holds $(\boldsymbol{\theta}, \boldsymbol{\lambda})$) which are updated as:

$$\mathbf{x}_i^{t,e} = \mathbf{x}_i^{t,e-1} - \mathbf{d}_i^{t,e-1},\ e = 1, \dots, E_i, \tag{9a}$$

$$\boldsymbol{\theta}^{t+1} = \mathbf{prox}_{r/\mu_\theta}(\mathbf{x}_q^{t+1} + \frac{1}{\mu_\theta}\boldsymbol{\lambda}^t), \tag{9b}$$

$$\boldsymbol{\phi}_i^{t+1} = \boldsymbol{\phi}_i^t + \frac{\mu_z}{2}\sum_{j\in\mathcal{N}_i}(\mathbf{x}_i^{t+1} - \mathbf{x}_j^{t+1}), \tag{9c}$$

$$\boldsymbol{\lambda}^{t+1} = \boldsymbol{\lambda}^t + \mu_\theta(\mathbf{x}_q^{t+1} - \boldsymbol{\theta}^{t+1}). \tag{9d}$$

Agent i conducts E_i local iterations as in (9a), where $\mathbf{d}_i^{t,e-1}$ is an approximated Newton step that is obtained

Algorithm 1 DRUID-VL

```
    Input: global rounds $T$, work loads $E_i$
 1: for global round $t = 1, \dots, T$ do
 2:     for each active agent $i$ in parallel do
   Primal update:
 3:         $\mathbf{x}_i^{t,0} \leftarrow \mathbf{x}_i^t$
 4:         for $e = 1, \dots, E_i$ do
 5:             sample batches $b_g$ and $b_H$
 6:             compute $\mathbf{g}_{i,b_g}^{t,e-1}, H_{i,b_H}^{t,e-1}$ as in (10)
 7:             solve $H_{i,b_H}^{t,e-1}\mathbf{d}_i^{t,e-1} = \mathbf{g}_{i,b_g}^{t,e-1}$
 8:             $\mathbf{x}_i^{t,e} \leftarrow \mathbf{x}_i^{t,e-1} - \mathbf{d}_i^{t,e-1}$
 9:         end for
10:         $\mathbf{x}_i^{t+1} \leftarrow \mathbf{x}_i^{t,E_i}$
   Communication:
11:         broadcast $\mathbf{x}_i^{t+1}$ to neighbors
   Dual update:
12:         $\boldsymbol{\phi}_i^{t+1} \leftarrow \boldsymbol{\phi}_i^t + \frac{\mu_z}{2}\sum_{j\in\mathcal{N}_i}(\mathbf{x}_i^{t+1} - \mathbf{x}_j^{t+1})$
   Updates pertaining to the regularizer:
13:         if $i = q$ then
14:             $\boldsymbol{\theta}^{t+1} \leftarrow \mathbf{prox}_{r/\mu_\theta}(\mathbf{x}_q^{t+1} + \frac{1}{\mu_\theta}\boldsymbol{\lambda}^t)$
15:             $\boldsymbol{\lambda}^{t+1} \leftarrow \boldsymbol{\lambda}^t + \mu_\theta(\mathbf{x}_q^{t+1} - \boldsymbol{\theta}^{t+1})$
16:         end if
17:     end for
18: end for
```

as follows. In local iteration e, we sample mini-batches b_g and b_H from the local dataset of the i-th agent $\{\mathbf{s}_i^j\}_{j=1}^{D_i}$. We then compute stochastic gradient $\mathbf{g}_{i,b_g}^{t,e-1}$ and sub-sampled Hessian $H_{i,b_H}^{t,e-1}$ for the local sub-problem with respect to $\mathbf{x}_i^{t,e-1}$ as follows:

$$\begin{aligned}\mathbf{g}_{i,b_g}^{t,e-1} &:= \tfrac{1}{|b_g|}\sum_{j\in b_g}\nabla l(\mathbf{s}_i^j; \mathbf{x}_i^{t,e-1}) + \delta_{iq}\mu_\theta\left(\mathbf{x}_i^t - \boldsymbol{\theta}^t + \mu_\theta^{-1}\boldsymbol{\lambda}^t\right) \\ &\quad + \boldsymbol{\phi}_i^t + \tfrac{\mu_z}{2}\sum_{j\in\mathcal{N}_i}\left(\mathbf{x}_i^t - \mathbf{x}_j^t\right) + \epsilon_i\left(\mathbf{x}_i^{t,e-1} - \mathbf{x}_i^t\right), \\ H_{i,b_H}^{t,e-1} &:= \tfrac{1}{|b_H|}\sum_{j\in b_H}\nabla^2 l(\mathbf{s}_i^j; \mathbf{x}_i^{t,e-1}) \\ &\quad + (\mu_z|\mathcal{N}_i| + \delta_{iq}\mu_\theta + \epsilon_i)I_d,\end{aligned} \tag{10}$$

where $\delta_{iq} = 1$ if $i = q$ and 0 otherwise. By solving the linear system $H_{i,b_H}^{t,e-1}\mathbf{d}_i^{t,e-1} = \mathbf{g}_{i,b_g}^{t,e-1}$, we obtain the desired stochastic Newton step $\mathbf{d}_i^{t,e-1}$.

The details of our proposed method, DRUID-VL (VL: variable loads), are given in Algorithm 1. It admits asynchronous implementation by allowing variable number of participating agents in each global round (step 2). Each active agent executes four stages sequentially: (i) *Primal update* (steps 3-10, using E_i iterations of stochastic Newton), (ii) *Communication* (broadcasting of the local variable $\mathbf{x}_i$ as in step 11), (iii) *Dual update* (step 12) and (iv) *Updates pertaining to the regularizer* (steps 13-16, only for the $q-$th agent).

IV. CONVERGENCE ANALYSIS

We proceed to establish our main convergence theorem (*Theorem* 1) under the following assumptions. The full proofs of all theoretical results are deferred to [30].

***Assumption** 1:* The graph $\mathcal{G}$ is connected.

***Assumption** 2:* The mini-batches $\frac{1}{|b|}\sum_{j\in b} l(\mathbf{s}_i^j;\cdot)$ and the regularizer function $r(\cdot)$ satisfy the following conditions:

(i) For each agent $i \in [n]$, $\frac{1}{|b|}\sum_{j\in b} l(\mathbf{s}_i^j;\cdot)$ are twice continuously differentiable, m_f-strongly convex with M_f-Lipschitz continuous gradient, i.e., $\forall i \in [n], \mathbf{x}_i \in \mathbb{R}^d$,

$$m_f I_d \preceq \frac{1}{|b|}\sum_{j\in b} \nabla^2 l(\mathbf{s}_i^j;\mathbf{x}_i) \preceq M_f I_d, \tag{11}$$

where $0 < m_f \le M_f < \infty$.

(ii) The Hessians of $\frac{1}{|b|}\sum_{j\in b} l(\mathbf{s}_i^j;\cdot)$ are Lipschitz continuous with constant L, i.e., $\forall i \in [n]$, $\mathbf{x}_i, \mathbf{y}_i \in \mathbb{R}^d$,

$$\left\|\frac{1}{|b|}\sum_{j\in b}\nabla^2 l(\mathbf{s}_i^j;\mathbf{x}_i) - \frac{1}{|b|}\sum_{j\in b}\nabla^2 l(\mathbf{s}_i^j;\mathbf{y}_i)\right\| \le L\|\mathbf{x}_i - \mathbf{y}_i\|. \tag{12}$$

(iii) $r(\cdot): \mathbb{R}^d \to \mathbb{R}$ is proper, closed, and convex, i.e., $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^d$,

$$(\mathbf{x}-\mathbf{y})^\top(\partial r(\mathbf{x}) - \partial r(\mathbf{y})) \ge 0, \tag{13}$$

where the inequality is meant for arbitrary elements in the subdifferential.

***Assumption** 3:* The agent participation over rounds is independent with probability p_i for agent i and $p_{\min} := \min_{i\in[n]} p_i > 0$.

In the following (*Lemmas* 2, 4, and *Theorem* 1), expectation $\mathbb{E}$ denotes conditional expectation upon past mini-batch selections and agent activations (assuming independence of the two processes). It is taken over the mini-batch selection at the current step (*Lemmas* 2, 4) and the user activation (*Theorem* 1).

The following lemma is a direct consequence of the analysis of stochastic Newton method [26, *Theorem* 11] applied to our setting (9a).

***Lemma** 2:* Let $\mathbf{x}^{t,\star}$ be solution to (6a). For sufficiently large batch sizes $|b_g|$ and $|b_H|$, there exists $c_i \in (0,1)$ for $i \in [n]$. For $C := \mathbf{diag}\left(c_1^{E_1},\ldots,c_n^{E_n}\right) \otimes I_{nd} \in \mathbb{R}^{nd\times nd}$, it holds that

$$\mathbb{E}\Big[\|\mathbf{x}^{t,E} - \mathbf{x}^{t,\star}\|^2\Big] \le \|\mathbf{x}^{t,0} - \mathbf{x}^{t,\star}\|_C^2. \tag{14}$$

The next lemma characterizes the residual error of primal sub-problem (6a) induced by inexact updates as in (9a). Recall that $\mathbf{y}^t = [\boldsymbol{\alpha}^t; -\boldsymbol{\alpha}^t]$ and $\boldsymbol{\phi}^t := E_s^\top \boldsymbol{\alpha}^t$.

***Lemma** 3:* The residual error is given by:

$$\begin{aligned}\mathbf{e}^t &:= \nabla\mathcal{L}_\Gamma(\mathbf{x}^{t+1},\boldsymbol{\theta}^t,\mathbf{z}^t;\mathbf{y}^t,\boldsymbol{\lambda}^t) &(15)\\ &= \nabla f(\mathbf{x}^{t+1}) + E_s^\top\boldsymbol{\alpha}^{t+1} + \mu_z E_u^\top(\mathbf{z}^{t+1} - \mathbf{z}^t)\\ &\quad + S(\boldsymbol{\lambda}^{t+1} + \mu_\theta(\boldsymbol{\theta}^{t+1} - \boldsymbol{\theta}^t)) + \Gamma(\mathbf{x}^{t+1} - \mathbf{x}^t). &(16)\end{aligned}$$

Note that from *Assumption* 2 it follows that $\mathcal{L}_\Gamma(\mathbf{x},\boldsymbol{\theta}^t,\mathbf{z}^t;\mathbf{y}^t,\boldsymbol{\lambda}^t)$ in (7) has Lipschitz continuous gradient with parameter

$$M := M_f + \mu_z \max_{i\in[n]}|\mathcal{N}_i| + \mu_\theta + \max_{i\in[n]}\epsilon_i. \tag{17}$$

The following lemma provides an upper bound for the residual error given in *Lemma* 3 in relation to the difference of two consecutive primal iterates $\mathbf{x}^{t+1} - \mathbf{x}^t$.

***Lemma** 4:* Recall $C := \mathbf{diag}\left(c_1^{E_1},\ldots,c_n^{E_n}\right)\otimes I_{nd}$. For any $i \in [n]$, under sufficiently large batch sizes there exists $c_i \in (0,1)$ so that for any $\xi_i \in \left(0, c_i^{-1} - 1\right)$ it holds that:

$$\mathbb{E}\left[\|\mathbf{e}^t\|^2\right] \le \mathbb{E}\left[\|\mathbf{x}^{t+1} - \mathbf{x}^t\|_{\mathcal{T}(E)}^2\right], \tag{18}$$

where $\mathcal{T}(E) := \mathbf{diag}\left(\tau_1(E_1),\ldots,\tau_n(E_n)\right)\otimes I_{nd} \in \mathbb{R}^{nd\times nd}$ with

$$\tau_i(E_i) := \frac{\left(1+\xi_i^{-1}\right)M^2 c_i^{E_i}}{1-(1+\xi_i)c_i^{E_i}},\ \ i \in [n]. \tag{19}$$

In what follows, we present the main convergence theorem that establishes linear convergence in expectation of DRUID-VL.

***Theorem** 1:* Define $P := \mathbf{diag}(p_i)$ and

$$\begin{aligned}\mathcal{H} &:= \mathbf{diag}\left(\Gamma, 2\mu_z I_{|\mathcal{E}|d}, 2\mu_z^{-1} I_{|\mathcal{E}|d}, \mu_\theta I_d, \mu_\theta^{-1} I_d\right), &(20)\\ \mathbf{v} &:= [\mathbf{x};\mathbf{z};\boldsymbol{\alpha};\boldsymbol{\theta};\boldsymbol{\lambda}] \in \mathbb{R}^{(n+2|\mathcal{E}|+2)d}. &(21)\end{aligned}$$

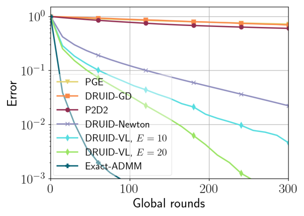
Let

$$\begin{aligned}\eta = \min\Bigg\{&\min_{i\in[n]}\frac{\mu_\theta\sigma_{\min}^+(\epsilon_i - \zeta\tau_i(E_i))}{5(\tau_i(E_i)+\epsilon_i^2)},\\ &\left(\frac{2m_f M_f}{m_f+M_f} - \frac{1}{\zeta}\right)\frac{1}{\max_{i\in[n]}\epsilon_i + \mu_\theta(\sigma_{\max}^{L_u}+2)},\\ &\frac{2}{5}\frac{\mu_\theta\sigma_{\min}^+}{m_f+M_f}, \frac{\sigma_{\min}^+}{5\max\left\{1,\sigma_{\max}^{L_u}\right\}}, \frac{1}{2}\Bigg\}, \end{aligned}\tag{22}$$

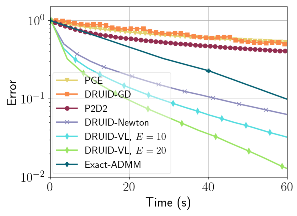
where $\sigma_{\min}^+$ denotes the smallest positive eigenvalue of $\begin{bmatrix}E_s\\ S^\top\end{bmatrix}\begin{bmatrix}E_s\ S^\top\end{bmatrix}$, $\sigma_{\max}^{L_u}$ denotes the maximum eigenvalue of L_u, $\zeta \in \left(\frac{m_f+M_f}{2m_fM_f}, \min_{i\in[n]}\frac{\epsilon_i}{\tau_i(E_i)}\right)$, and $\epsilon_i > \frac{\tau_i(E_i)(m_f+M_f)}{2m_fM_f}$, $i \in [n]$, then it holds that:

$$\mathbb{E}\left[\|\mathbf{v}^{t+1} - \mathbf{v}^\star\|_{\mathcal{H}P^{-1}}^2\right] \le \left(1 - \frac{\eta p_{\min}}{1+\eta}\right)\|\mathbf{v}^t - \mathbf{v}^\star\|_{\mathcal{H}P^{-1}}^2. \tag{23}$$

***Remark** 2:* ADMM is a first-order method and thus converges linearly (the best possible rate [18]). Our goal is to accelerate convergence inside the ADMM framework by performing variable work loads on agents and therefore the result in *Theorem* 1 falls into the taxonomy of linear convergence. On the other hand, superlinear convergence to the global optimum in distributed networks requires heavy communication costs among agents (usually multiple message exchanges in one round [13]–[15], and does not admit asynchronous implementation). In contrast, our proposed method requires a single communication step of size d and is asynchronous, suitable for real-life multi-agent networks where communication is the main bottleneck.



(a) Relative error versus global rounds.



(b) Relative error versus time.

Fig. 1: Comparisons of DRUID-VL (synchronous setting and common E for all agents) with baseline algorithms in terms of relative error $\frac{\|\mathbf{x}^t-\mathbf{x}^\star\|^2}{\|\mathbf{x}^0-\mathbf{x}^\star\|^2}$; convergence versus (a) global rounds and (b) run-time.

In the following corollary, we show that the personalized coefficient ϵ_i (the diagonal entries of Γ in (7)) can be further optimized to adjust to the heterogeneity of variable work loads, so as to achieve faster convergence (i.e., increase η in (22)).

***Corollary** 1:* By selecting

$$\epsilon_i = \zeta\tau_i(E_i) + \sqrt{\zeta^2\tau_i^2(E_i) + \tau_i(E_i)}, \tag{24}$$

for $\zeta > \frac{m_f+M_f}{2m_fM_f}$, (23) holds with

$$\eta = \min\Bigg\{ \frac{\mu_\theta\sigma_{\min}^+}{10\zeta} \min_{i\in[n]} \left(\sqrt{\tau_i^2(E_i) + \zeta^{-2}\tau_i(E_i)} + \tau_i(E_i)\right)^{-1}, \left(\frac{2m_fM_f}{m_f+M_f} - \frac{1}{\zeta}\right) \frac{1}{\max_{i\in[n]}\epsilon_i + \mu_\theta(\sigma_{\max}^{L_u}+2)}, \frac{2}{5}\frac{\mu_\theta\sigma_{\min}^+}{m_f+M_f}, \frac{\sigma_{\min}^+}{5\max\left\{1,\sigma_{\max}^{L_u}\right\}}, \frac{1}{2}\Bigg\}. \tag{25}$$

V. EXPERIMENTS

We consider distributed sparse logistic regression, i.e,

$$l(\mathbf{s};\hat{\mathbf{x}}) := \log\left(1 + e^{\hat{\mathbf{x}}^\top\mathbf{w}}\right) + (1-y)\hat{\mathbf{x}}^\top\mathbf{w},$$

for $\mathbf{s} = (\mathbf{w}, y) \in \mathbb{R}^d \times \{0,1\}$ and $r(\hat{x}) = \gamma\|\hat{x}\|_1$, with $\gamma = 2\times 10^{-6}$. We take $4,000$ samples of dimension $d = 22$ from *ijcnn1*[1] and evenly distribute them to 10 agents, whose network topology is generated by the Erdős-Rényi model with $p = 0.2$ (each edge is generated independently with probability 0.2). We first evaluate our proposed method, DRUID-VL (with the same work load $E = 10$ and 20 across all agents), against five baselines, namely PGE [8], P2D2 [12], DRUID-GD/-Newton [24], and Exact-ADMM (i.e., solving (6a) 'exactly', which means assuring a primal-dual gap of 10^{-5}, in *Primal update* of Algorithm 1, while other stages are unchanged), considering only synchronous updates as all but DRUID do not support asynchronous implementation. We set $\epsilon = \mu_\theta = 2\mu_z = 10^{-4}$ (the selection follows from our analysis in [30, (36)]), and sampling batch size $|b_g| = |b_H| = 100$ for DRUID-VL.

The convergence of each algorithm is plotted in Fig. 1a and 1b, in terms of global rounds and time, respectively. Fig. 1a demonstrates that for a fixed target error, DRUID-VL with $E = 10/20$ requires 41%/59% less global rounds compared to DRUID-Newton, while the benefits are much higher for the other (gradient-based) baselines. Fig. 1b illustrates the progress of each algorithm per actual run-time (this is done since the cost of one round using a second-order local solver is higher than for gradient-based methods). We deduce that for a given time, DRUID-VL with $E = 10/20$ decreases the relative error by $1.96\times/4.76\times$ of DRUID-Newton, and $12.5\times/33.3\times$ of PGE, DRUID-GD, and P2D2. We also emphasize that although Exact-ADMM attains the fastest convergence in terms of global rounds (Fig. 1a), solving the sub-problem exactly brings about heavy computational overhead. As a result, Exact-ADMM has the slowest convergence speed among all DRUID variants in real-time scale (Fig. 1b). These findings corroborate that our proposed method better exploits local computation resources to reduce communication costs and run-time.

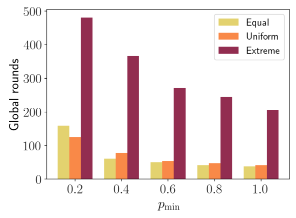


Fig. 2: Average communication cost per agent to reach an error of 10^{-2} under variable heterogeneity and participation rate.

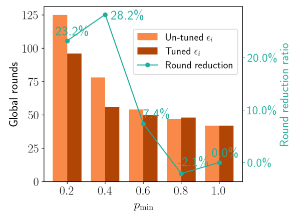


Fig. 3: Global rounds required to reach an error of 10^{-2} with tuned and untuned ϵ_i under *Uniform* selection of $\{E_i\}_{i=1}^n$.

We further study the impact of heterogeneity and participation rate in Fig. 2, in terms of the communication cost to reach a given accuracy (10^{-2} in all cases). The former is achieved by three alternatives for work load selection (for fair comparison we set the average load fixed): *Equal* - $E_i = 10$ for all i; *Uniform* - $\{E_i\}_{i=1}^n$ are sampled at uniformly random from $\sim\mathcal{U}\{1,19\}$; *Extreme* - half of agents are assigned $E_i = 1$ while the others have $E_i = 19$. We use the same parameter setting as in Fig. 1 and fix $\epsilon_i = 10^{-4}$ for all agents. Fig. 2 shows that the participation rate $p_{\min}$ does not have a significant impact on the communication burden per agent, in full accord with (23). More interestingly, heterogeneity is shown to play a major role in exacerbating communication costs: higher variance of work load $\{E_i\}_{i=1}^n$ (e.g., in our case, *Extreme* selection) yields significant rise in communication cost.

Based on the analysis in *Corollary* 1, we further tune the local hyperparameters ϵ_i so as to adjust to heterogeneity. Given that the expression in (24) cannot be

[1]Available at https://www.csie.ntu.edu.tw/~cjlin/libsvm/.

analytically evaluated in a real scenario (where constants M_f, m_f, c_i are unknown), we choose a surrogate rule based on (19) and (24) as:

$$\epsilon_i = \bar{\epsilon} c^{E_i - \bar{E}}[1-(1+\zeta)c^{\bar{E}}]/[1-(1+\zeta)c^{E_i}],$$

where we select $\bar{\epsilon} = 10^{-4}$, $\bar{E} = 10$, $c = 0.98$, $\zeta = 5\times10^{-3}$. The results are plotted in Fig. 3, for the *Uniform* heterogeneity setting (as in Fig. 2). We observe a noticeable reduction of the number of global rounds needed to reach a common accuracy (10^{-2}). Maximum gain from hyperparameter tuning is achieved at $p_{\min} = 0.4$ (28.2% less rounds). Last but not least, the gain increases when the participation rate $p_{\min}$ decreases, indicating that our proposed method is well-suited to adapt to heterogeneity in the most challenging (albeit common in real-systems) scenario of 'high' asynchrony.

VI. Conclusion

We proposed a distributed asynchronous primal-dual method tailored for heterogeneous multi-agent systems (in terms of variability of local work). The algorithm allows for a variable number of participating agents and requires a single broadcast of the local vector. Linear convergence was established for arbitrary degree of heterogeneity (*Theorem* 1). Additionally, a simple method of personalizing hyperparameters locally (*Corollary* 1) was shown to accelerate the algorithm both theoretically and experimentally. The full proofs are available in [30].

References

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

[2] Y. Li, P. Voulgaris, and N. Freris, "Distributed primal-dual optimization for heterogeneous multi-agent systems," in *IEEE Conference on Decision and Control (CDC)*, 2022, pp. 5870–5875.

[3] M. Vlachos, N. Freris, and A. Kyrillidis, "Compressive mining: fast and optimal data mining in the compressed domain," *International Journal on Very Large Data Bases*, vol. 24, no. 1, pp. 1–24, 2015.

[4] P. Sopasakis, N. Freris, and P. Patrinos, "Accelerated reconstruction of a compressively sampled data stream," in *IEEE European Signal Processing Conference (EUSIPCO)*, 2016, pp. 1078–1082.

[5] N. Freris, H. Kowshik, and P. R. Kumar, "Fundamentals of large sensor networks: Connectivity, capacity, clocks, and computation," *Proceedings of the IEEE*, vol. 98, no. 11, pp. 1828–1846, 2010.

[6] C. Conte, T. Summers, M. Zeilinger, M. Morari, and C. Jones, "Computational aspects of distributed optimization in model predictive control," in *IEEE Conference on Decision and Control (CDC)*, 2012, pp. 6819–6824.

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

[8] 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.

[9] W. Shi, Q. Ling, G. Wu, and W. Yin, "A proximal gradient algorithm for decentralized composite optimization," *IEEE Transactions on Signal Processing*, vol. 63, no. 22, pp. 6013–6023, 2015.

[10] 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.

[11] A. Nedić, 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.

[12] S. Alghunaim, K. Yuan, and A. Sayed, "A linearly convergent proximal gradient algorithm for decentralized optimization," *Advances in Neural Information Processing Systems*, 2019.

[13] E. Wei, A. Ozdaglar, and A. Jadbabaie, "A distributed Newton method for network utility maximization–i: Algorithm," *IEEE Transactions on Automatic Control*, vol. 58, no. 9, pp. 2162–2175, 2013.

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

[15] R. Crane and F. Roosta, "DINGO: Distributed Newton-type method for gradient-norm optimization," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[16] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[17] E. Wei and A. Ozdaglar, "Distributed alternating direction method of multipliers," in *IEEE Conference on Decision and Control (CDC)*, 2012, pp. 5445–5450.

[18] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "On the linear convergence of the ADMM in decentralized consensus optimization," *IEEE Transactions on Signal Processing*, vol. 62, no. 7, pp. 1750–1761, 2014.

[19] Q. Ling, W. Shi, G. Wu, and A. Ribeiro, "DLM: Decentralized linearized alternating direction method of multipliers," *IEEE Transactions on Signal Processing*, vol. 63, no. 15, pp. 4051–4064, 2015.

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

[21] T. Chang, M. Hong, W. Liao, and X. Wang, "Asynchronous distributed ADMM for large-scale optimization—Part I: Algorithm and convergence analysis," *IEEE Transactions on Signal Processing*, vol. 64, no. 12, pp. 3118–3130, 2016.

[22] 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.

[23] 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.

[24] Y. Li, P. Voulgaris, D. Stipanovic, and N. Freris, "Communication efficient curvature aided primal-dual algorithms for decentralized optimization," *IEEE Transactions on Automatic Control*, 2023.

[25] N. Freris, S. Graham, and P. R. Kumar, "Fundamental limits on synchronizing clocks over networks," *IEEE Transactions on Automatic Control*, vol. 56, no. 6, pp. 1352–1364, 2011.

[26] F. Roosta-Khorasani and M. Mahoney, "Sub-sampled Newton methods," *Mathematical Programming*, vol. 174, no. 1, pp. 293–326, 2019.

[27] R. Johnson and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," *Advances in Neural Information Processing Systems*, vol. 26, 2013.

[28] N. Agarwal, B. Bullins, and E. Hazan, "Second-order stochastic optimization for machine learning in linear time," *Journal of Machine Learning Research*, vol. 18, no. 116, pp. 1–40, 2017.

[29] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends® in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[30] Z. Yu and N. Freris, "Communication-efficient distributed optimization with adaptability to system heterogeneity," *arXiv preprint arXiv:2308.05395*, 2023.