Achieving Linear Convergence with Parameter-Free Algorithms in Decentralized Optimization

Anonymous Author(s)
Affiliation
Address
email

Abstract

This paper addresses the minimization of (locally strongly) convex, locally smooth functions over a network of agents without a centralized server. Existing decentralized algorithms require knowledge of problem and network parameters, such as the Lipschitz constant of the global gradient and/or network connectivity, for hyperparameter tuning. Agents usually cannot access this information, leading to conservative selections and slow convergence or divergence. This paper introduces a decentralized algorithm that eliminates the need for specific parameter tuning. Our approach employs an operator splitting technique with a novel variable metric, enabling a local backtracking line-search to adaptively select the stepsize without global information or extensive communications. This results in favorable convergence guarantees and dependence on optimization and network parameters compared to existing nonadaptive methods. Notably, our method is the first *adaptive* decentralized algorithm that achieves linear convergence for (locally) strongly convex (locally) smooth functions. Numerical experiments on machine learning problems demonstrate superior performance in convergence speed and scalability.

1 Introduction

We study optimization across a network of $m > 1$ agents, modeled as an undirected, static graph, possibly with no centralized server. The agents cooperatively solve the following problem:

$$\min_{x\in\mathbb{R}^d} \sum_{i=1}^{m} f_i(x), \tag{P}$$

where $f_i : \mathbb{R}^d \to \mathbb{R}$ is the loss function of agent i, assumed to be (locally strongly) convex and locally smooth (i.e., with gradient being locally Lipschitz continuous), and accessible only to agent i.

This formulation applies to various fields, particularly emphasizing decentralized machine learning problems where datasets are produced and collected at different locations. Traditionally, statistical and computational methods in this domain have relied on a centralized paradigm, aggregating computational resources at a single, central location. However, this approach is increasingly unsuitable for modern applications with many machines, leading to server congestion, inefficient communication, and high energy consumption [25, 21]. This has motivated the surge of learning algorithms that target *decentralized* networks with *no servers*, a.k.a. *mesh* networks, which is the setting of this paper.

Decentralized convex optimization has a long history, with numerous proposals applicable to Problem (P), particularly when the loss functions are *globally* smooth. Recent tutorials include [31, 38, 7, 30, 42]. **Lack of adaptivity:** While these methods are different in their updates, they share the hurdle of relying sensibly on the tuning of hyperparameters, such as the stepsize (a.k.a. learning rate), for both theoretical and practical convergence. Existing theories ensure convergence under generally conservative bounds on the stepsize, which depend on parameters like the Lipschitz constant of the global gradient, the spectral gap of the graph adjacency matrix, or other topological properties. Acquiring such information is challenging in practice, due to physical or privacy limitations and computational/communication constraints. This often leads to manual tuning, which is not only tedious but also results in less predictable, problem-dependent, and non-reproducible performance.

Parameter-free centralized methods: On the the hand, significant progress has been made in the *centralized* setting to automate the selection of the stepsize across various optimization and learning problem classes. **(i)** Traditional approaches in optimization–such as line-search methods [33], Barzilai-Borwein's stepsize [3], and Polyak's stepsize [34]–have been supplemented by recent adaptive stepsize rules based on estimates of local curvature [27] and subsequent techniques [28, 17, 18, 20, 46]. **(ii)** In the ML community, adaptive gradient methods such as AdaGrad [12], Adam [16], AMSGrad [37], NSGD-M [10], and variants [23, 41, 26] have gained significant attention for training large-scale learning models. These methods apply to stochastic, nonconvex optimization problems. **(iii)** Further advancements extend adaptivity to stochastic/online convex optimization problems, e.g., [5, 13].

Distributed adaptive methods: While variant of these centralized algorithms have been adapted to federated architectures (server-client systems), e.g., in [36, 22, 9], their application to mesh networks is *not feasible*. In federated learning, a central server aggregates local model updates, a process integral to its hierarchical structure. However, mesh networks, which lack a centralized coordinating node, do not support such a direct aggregation of large-scale vectors. Recent attempts to implement some form of stepsize adaptivity for *stochastic (non)convex/online* optimization problems over mesh networks are [29, 8, 19]. These methods generally achieve adaptivity by properly normalizing agents' gradients using past information. However, with the exception of [19], they rely on the strong assumption that the (population) losses are *globally* Lipschitz continuous (i.e., their gradients are bounded). In fact, Lipschitz continuity in convex optimization readily unlocks parameter-free convergence by using stepsize tuning of $\mathcal{O}(1/\sqrt{k})$ (here, k is the iteration index). Moreover, [29, 8] still require knowledge of some optimization parameters for the stepsize tuning, to guarantee convergence.

Open questions and challenges: To our knowledge, no deterministic, parameter-free decentralized algorithms exist that solve Problem (P) over mesh networks, particularly achieving linear convergence when agents' functions are (locally) strongly convex and smooth. The current decentralized adaptive stochastic methods [29, 8, 19] discussed earlier do not adequately bridge this gap. Tailored for stochastic environments, these methods merely ensure that cumulative consensus errors along the iterations remain bounded, *not necessarily decreasing*. This typically involves either diminishing stepsizes or adjustments based on the final horizon to manage the bias-variance trade-off. These strategies fall short in deterministic scenarios like Problem (P), failing to ensure convergence to *exact* solutions, and achieve faster $\mathcal{O}(1/k)$ convergence rates in convex cases or *linear* rates in strongly convex scenarios. Furthermore, none of these methods effectively handle losses that are *locally* (rather than globally) smooth and strongly convex.

Major contributions: This paper addresses this open problem. Our contributions are the following:

1. A new parameter-free decentralized algorithm: We propose a decentralized algorithm that eliminates the need for specific tuning of the step size. Our approach leverages a Forward-Backward operator splitting technique combined with a novel variable metric, enabling a local backtracking line-search procedure to adaptively select the step size at each iteration without requiring global information on optimization and network parameters or extensive communications. We are not aware of any other decentralized line-search methods over mesh networks.

Designing decentralized line-search procedures that are well-defined (terminating in a finite number of steps), locally implementable, and ensure algorithm convergence through satisfactory descent on an appropriate merit function presents significant challenges. A major issue is that line-search procedures merely based on the local curvature of agents' functions often fail to ensure convergence, producing *excessively large*, heterogeneous stepsizes that, e.g., poorly connected networks cannot support. This necessitates the identification of line-search *directions* and *surrogate functions* that encapsulate *both* optimization and network influences, aspects that have not yet formalized. Our design guidelines (cf., Sec. 3) are of independent interest; hopefully they will provide valuable insights for the development of other decentralized adaptive schemes, such as those based on alternative operator splittings.

2. Convergence guarantees: We have established convergence for the proposed decentralized adaptive method. **(i)** For agents' losses that are strongly convex, linear convergence rates are achieved, while typical $\mathcal{O}(1/k)$ sublinear rates are confirmed for the convex (non-strongly convex) setting. Our analysis crucially identifies key quantities capturing the interplay between optimization and network conditions and governing the rate expressions. Specifically, (a) In relatively "well-connected" networks, the convergence rate is influenced primarily by the optimization parameters, showing a *linear* dependence on the condition number of the local losses; (b) in contrast, in poorly connected networks, the rates suffer from network degradation terms and exhibit *quadratic* (instead of linear) dependence on the condition number, indicative of expected performance degradation. **(ii)** Unlike most existing results in distributed optimization, the optimization parameters in our rate expressions, such as the smooth and strong convexity constants, are localized to the *convex hull* of the traveled iterates. This results from the stepsize tuning based on the line-search procedure that adapts to local geometries, leading to more favorable dependencies on optimization parameters and thus enhanced convergence guarantees. **(iii)** Our analysis also extends to functions that are only locally smooth (and strongly convex), significantly broadening the class of functions to which the proposed algorithm can be applied to. This advancement distinguishes our work from the existing literature on decentralized (including nonadaptive) optimization algorithms, which generally focus on globally smooth functions (when differentiable). **(iv)** Numerical experiments demonstrate superior performance of the proposed adaptive algorithm in convergence speed and scalability compared to existing non-adaptive methods.

1.1 Notation and paper organization

Capital letters denote matrices. Bold capital letters represent matrices where each row is an agent's variable, e.g., $\mathbf{X} = [x_1, \ldots, x_m]^\top$. For such matrices, the i-th row is denoted by the corresponding lowercase letter with the subscript i; e.g., for $\mathbf{X}$, we write x_i (as column vector). Let $\mathbb{S}^m$, $\mathbb{S}^m_+$, and $\mathbb{S}^m_{++}$ be the set of $m \times m$ (real) symmetric, symmetric positive semidefinite, and symmetric positive definite matrices, respectively; $A^\dagger$ denotes the Moore-Penrose pseudoinverse of A. The eigenvalues of $W \in \mathbb{S}^m$ are ordered in nonincreasing order, and denoted by $\lambda_1(W) \geq \cdots \geq \lambda_m(W)$. For two operators A and B of appropriate size, $(A \circ B)(\bullet)$ stands for $A(B(\bullet))$. We denote: $[m] = \{1, \ldots, m\}$; $[x]_+ := \max(x, 0)$, $x \in \mathbb{R}$; $1_m \in \mathbb{R}^m$ is the vector of all ones; I_m (resp. 0_m) is the $m \times m$ identity (resp. the $m \times m$ zero) matrix; $\texttt{null}(A)$ (resp. $\texttt{span}(A)$) is the nullspace (resp. range space) of the matrix A. Let $\langle X, Y \rangle := \texttt{tr}(X^\top Y)$, for any X and Y of suitable size ($\texttt{tr}(\bullet)$) is the trace operator; and $\|X\|_M := \langle MX, X \rangle$, for any symmetric, positive definite M and X of suitable dimensions. We still use $\|X\|_M$ when M is positive semidefinite and $X \in \texttt{span}(M)$. We set $1/0 = \infty$.

2 Problem Setup

We investigate Problem (P) over a network of $[m]$ agents, modeled as an undirected, static, connected graph $\mathcal{G} = ([m], \mathcal{E})$, where $(i, j) \in \mathcal{E}$ if there is communication link (edge) between i and j. We consider either convex or strongly convex instances of (P), as stated below.

Assumption 1. *(i) Each function f_i in (P) is L-smooth and μ-strong convex on $\mathbb{R}^d$, for some $L \in (0, \infty)$ and $\mu \in [0, \infty)$. When $\mu > 0$, we define $\kappa := L/\mu$. When $\mu = 0$, (P) is assumed to have a solution. Furthermore, (ii) each agent i has access only to its own function f_i.*

Note that the case $\mu_i = 0$ merely corresponds to convexity. For readability, our convergence results are presented under Assumption 1, while the proofs in the appendix tackle the more general case of *local* smoothness (and strong convexity). We refer to the appendix for these more general statements.

The following matrices are commonly utilized in the design of gossip-based algorithms.

Definition 2 (Gossip matrices)**.** *Let $\mathcal{W}_\mathcal{G}$ denote the set of matrices $\widetilde{W} = [\widetilde{W}_{ij}]_{i,j=1}^m$ that satisfy the following properties:* **(i)** *(compliance with $\mathcal{G}$) $\widetilde{W}_{ij} > 0$ if $(i, j) \in \mathcal{E}$; otherwise $\widetilde{W}_{ij} = 0$. Furthermore, $\widetilde{W}_{ii} > 0$, for all $i \in [m]$; and* **(ii)** *(doubly stochastic) $\widetilde{W} \in \mathbb{S}^m$ and $\widetilde{W} 1_m = 1_m$.*

These matrices are standard in the literature on decentralized optimization algorithms, and several instances have been employed in practice; see [31, 38, 30] for some representative examples. Notice that for any $\widetilde{W} \in \mathcal{W}_\mathcal{G}$ (assuming $\mathcal{G}$ connected) it hold: **(i)** (null space condition) $\texttt{null}(I_m - W) = \texttt{span}(1_m)$; and **(ii)** (eigen-spectrum distribution) $2I \succeq \widetilde{W} + I \succ 0_m$.

3 Algorithm Design

Our approach to solving Problem (P) involves a saddle-point reformulation tackled via a variable metric operator splitting, implementable across the graph $\mathcal{G}$. The innovative aspect of the proposed method lies in the selection of the variable metric that, coupled with a Forward Backward Splitting (FBS), enable adaptive stepsize selections through a decentralized line-search procedures.

Introducing local copies $x_i \in \mathbb{R}^d$ of the shared variable x (the i-th one is controlled by agent i), and the stack matrix $\mathbf{X} := [x_1, \ldots, x_m]^\top \in \mathbb{R}^{m \times d}$, let us consider the following auxiliary problem:

$$\min_{\mathbf{x},\mathbf{y}\in\mathbb{R}^{m\times d}} \left[F(\mathbf{X}) := \sum_{i=1}^{m} f_i([K\mathbf{X}]_i) \right], \ \text{ s.t. } \ \text{Ł}\,\mathbf{X} = 0. \tag{P$'$}$$

Here, Ł and K are $m \times m$ matrices that meet the following criteria: **(c1)** $\text{Ł} \in \mathbb{S}^m$ and $\texttt{null}(\text{Ł}) = \texttt{span}(1_m)$; **(c2)** $K \in \mathbb{S}^m_{++}$ and $\texttt{null}(I-K) = \texttt{span}(1_m)$; and **(c3)** Ł and K commute. Conditions (c1) and (c2) ensure that (P) and (P$'$) are equivalent. Specifically, any solution $\mathbf{X}^\star$ of (P$'$) has the form of $\mathbf{X}^\star = 1_m(x^\star)^\top$, where $x^\star$ solves (P), and vice versa. While not essential, condition (c3) is postulated to simplify the algorithm derivation.

Primal-dual optimality for (P$'$) reads, with $\mathbf{Y}$ being the dual-variable associated with the constraints,

$$(A+B)\left(\begin{bmatrix}\mathbf{X}^\star\\ \mathbf{Y}^\star\end{bmatrix}\right) = 0, \quad \text{where} \quad A := \begin{bmatrix} K\circ\nabla F\circ K & 0\\ 0 & 0\end{bmatrix} \ \text{and} \ B := \begin{bmatrix} 0 & \text{Ł}\\ -\text{Ł} & 0\end{bmatrix}.$$

Given $\mathbf{X}^k, \mathbf{Y}^k$ at iteration k, the update $\mathbf{X}^{k+1}, \mathbf{Y}^{k+1}$ via FBS with metric $C \in \mathbb{S}^{2m}_{++}$ reads [4]

$$(C+B)\left(\begin{bmatrix}\mathbf{X}^{k+1}\\ \mathbf{Y}^{k+1}\end{bmatrix}\right) = (C-A)\left(\begin{bmatrix}\mathbf{X}^{k}\\ \mathbf{Y}^{k}\end{bmatrix}\right). \tag{1}$$

Monotone operator theory [4] ensures convergence of (1) under the following conditions:

(c4) B is a monotone operator, $C \in \mathbb{S}^{2m}_{++}$, and **(c5)** $I - C^{-1/2}AC^{-1/2}$ is an averaged operator.

Condition (c4) is satisfied by construction; (c5) can be enforced through a suitable selection of $C \in \mathbb{S}^{2m}_{++}$ while leveraging the co-coercivity of A (implied by Assumption 1). Denoting by $\alpha > 0$ the stepsize employed in the algorithm, we seek for C with the following structure:

$$C = \begin{bmatrix}\alpha^{-1}C_1 & 0\\ 0 & C_2\end{bmatrix}, \quad \text{with} \quad C_1, C_2 \in \mathbb{S}^m_{++}$$

to be determined. We proceed solving (1). Taking $(C+B)^{-1}$, we have

$$\begin{aligned}\mathbf{X}^{k+1} &= (I)\,(\mathbf{X}^k) - \alpha\left((II)\,(\mathbf{X}^k) + (III)\,(\mathbf{Y}^k)\right),\\ \mathbf{Y}^{k+1} &= (IV)\,(\mathbf{Y}^k) + (V)\,(\mathbf{X}^k),\end{aligned} \tag{2}$$

where

$$\begin{aligned}
(I) &:= I_m - \alpha\cdot C_1^{-1}\,\text{Ł}\,\left(C_2 + \alpha\cdot \text{Ł}\,C_1^{-1}\,\text{Ł}\right)^{-1}\,\text{Ł},\\
(II) &:= (I)\,C_1^{-1}\,K\,\nabla F\circ K,\\
(III) &:= C_1^{-1}\,\text{Ł}\,\left(C_2 + \alpha\cdot\text{Ł}\,C_1^{-1}\,\text{Ł}\right)^{-1}\,C_2,\\
(IV) &:= \left(C_2 + \alpha\cdot\text{Ł}\,C^{-1}\,\text{Ł}\right)^{-1}\,C_2,\\
(V) &:= \left(C_2 + \alpha\cdot\text{Ł}\,C_1^{-1}\,\text{Ł}\right)^{-1}\text{Ł}\left(I - \alpha\cdot C_1^{-1}K\,\nabla F\circ K\right).
\end{aligned} \tag{3}$$

In addition to satisfying (c5), $C_1, C_2 \in \mathbb{S}^m_{++}$ must be strategically chosen to facilitate the design of a decentralized line-search procedure for α. We propose the following guiding principles:

(c6) The range of admissible stepsize values α ensuring convergence–hence satisfying (c5)–should be independent of the network parameters; and

(c7) the operators (I), (II), and (III) in (2) should be independent of α.

At a high level, (c6) aims to decouple the line-search mechanism from network-dependent constraints. By doing so, it ensures that performing the line-search from the agents' sides requires no mid-process communications during backtracking, relying solely on local computations. Meanwhile, (c7) facilitates the identification of $-((II)(\mathbf{X}^k) + (III)(\mathbf{Y}^k))$ as a potential direction for the line-search. This direction must be paired with an appropriate surrogate function, which we will define shortly.

Among several potential selections, in this paper, we consider the following for C_1 and C_2:

$$C_1 = K \quad \text{and} \quad C_2 = \alpha K^{-1}\left(c^{-1}\,I - \text{Ł}^2\right), \ \text{with } c < 1/2, \tag{4}$$

which satisfy all the specified requirements. Using (4) and (c3), the operators in (3) simplify to

$$(I) = I_m - c\cdot\text{Ł}^2,\ (II) = (I)\nabla F\circ K,\ (III) = (I)\text{Ł}^2K^{-1},\ (IV) = (I),\ (V) = \frac{c}{\alpha}\cdot K\,\text{Ł}\,(I - \nabla F\circ K).$$

Notice that (I), (II), and (III) are independent of the stepsize. Substituting the above expressions in (2) and introducing $\mathbf{D}^k := K^{-1}\text{Ł}\mathbf{Y}^k$, the algorithm can be rewritten as

$$\mathbf{X}^{k+1} = (I - c\text{Ł}^2)\,\mathbf{X}^k - \alpha\cdot(I - c\text{Ł}^2)\left(\mathbf{D}^k + \nabla F(K\mathbf{X}^k)\right),$$
$$\mathbf{D}^{k+1} = (I - c\text{Ł}^2)\,\mathbf{D}^k + \frac{c}{\alpha}\cdot\text{Ł}^2\left(\mathbf{X}^k - \alpha\nabla F(K\mathbf{X}^k)\right).$$

To make the above updates compliant with the graph $\mathcal{G}$ while satisfying (c1)-(c3), we set $\text{Ł}^2 = (I - \widetilde{W})$, with $\widetilde{W} \in \mathcal{W}_{\mathcal{G}}$, and $K = I - c\text{Ł}^2$, where $c \in (0, 1/2)$ is a free universal constant. Introducing $W := (1-c)I_m + c\widetilde{W} \in \mathcal{W}_{\mathcal{G}}$, the final decentralized algorithm can be rewritten as

$$\begin{aligned}\mathbf{X}^{k+1/2} &= W\,\mathbf{X}^k, \quad \mathbf{D}^{k+1/2} = W\left(\mathbf{D}^k + \nabla F(\mathbf{X}^{k+1/2})\right),\\ \mathbf{X}^{k+1} &= \mathbf{X}^{k+1/2} - \alpha\cdot\mathbf{D}^{k+1/2},\\ \mathbf{D}^{k+1} &= \mathbf{D}^{k+1/2} + \frac{1}{\alpha}\cdot\left(\mathbf{X}^k - \mathbf{X}^{k+1} - \alpha\nabla F(\mathbf{X}^{k+1/2})\right).\end{aligned} \tag{5}$$

Finally, it can be verified that (c6) is met if $(\sqrt{\alpha}K^{-1/2})\circ\nabla F\circ(\sqrt{\alpha}K^{-1/2})$ is nonexpansive, which holds if $\alpha < 1/L$, being independent on the network parameters. Next, we introduce a line-search procedure that enables the use of an adaptive stepsize α rather than a more conservative constant one.

Decentralized backtracking: It is not difficult to check that (i) $-\mathbf{D}^{k+1/2}$ is a descent direction of $F^k(\mathbf{X}) := F(\mathbf{X}) + \langle\mathbf{D}^k, \mathbf{X}\rangle$ at $\mathbf{X}^{k+1/2}$, and (ii) F^k and F share the same smooth constant. These suggest the following backtracking procedure for α: at iteration k, find the largest $\alpha^k > 0$ such that

$$F^k(\mathbf{X}^{k+1}) \le F^k(\mathbf{X}^{k+1/2}) + \left\langle\nabla F^k(\mathbf{X}^{k+1/2}), \mathbf{X}^{k+1} - \mathbf{X}^{k+1/2}\right\rangle + \frac{\delta}{2\alpha^k}\|\mathbf{X}^{k+1} - \mathbf{X}^{k+1/2}\|^2, \tag{6}$$

where $\delta \in (0, 1]$ is a tuning parameter. However, this condition would require a communication round for each backtracking step. To reduce the communication burden, we introduce a local stepsize for each agent i, denoted by α_i^k, determined by a backtracking line-search on the local function $f_i^k(x) := f_i(x) + \langle d_i^k, x\rangle$. Specifically, each α_i^k is the largest positive value satisfying

$$f_i^k(x_i^{k+1}) \le f_i^k(x_i^{k+1/2}) + \left\langle\nabla f_i^k(x_i^{k+1/2}), x_i^{k+1} - x_i^{k+1/2}\right\rangle + \frac{\delta}{2\alpha_i^k}\|x_i^{k+1} - x_i^{k+1/2}\|^2. \tag{7}$$

The proposed decentralized algorithm is summarized in Algorithm 1, with the backtracking line-search procedure detailed in Algorithm 2.

3.1 Discussion

Several comments are in order.

On the proposed algorithm: We emphasize that selecting $K \neq I_m$ in (P′) marks a significant departure from the commonly used saddle-point reformulations of Problem (P), where $K = I_m$, e.g., [43, 31, 30, 1]. Choosing $K \neq I_m$, in conjunction with the novel variable metric C in the FBS as specified in (4), is critical to obtain a valid line-search procedure that is also implementable across the network. For instance, popular decentralized algorithms such as EXTRA [39] and NIDS [24] can be interpreted as FBS with suitable metrics associated with the primal-dual reformulation of (P) as (P′) but with $K = I_m$. However, these schemes do not facilitate any suitable line-search, as no stepsize-independent descent direction can be identified in their updates. Hopefully, our approach will provide principled guidelines for the design of other parameter-free decentralized algorithms, stemming from alternative decentralized formulations of (P) and their corresponding operator splittings.

On the backtracking: The following lemma shows that the line-search procedure in Algorithm 2 is well-defined, as long as the function f is locally smooth (the proof can be found in the appendix).

Lemma 3. *Let f in Algorithm 2 be any L_f-smooth and μ_f-strongly convex function on the segment $[x, x + \gamma\alpha d]$, with $L_f \in (0, \infty)$, $\mu_f \in [0, \infty)$, and $\gamma \in [1, \infty)$. The following hold for Algorithm 2:*

1. *The backtracking procedure terminates in no more than* $\max\left(1, \lceil\log_2\frac{2L_i\gamma\alpha}{\delta}\rceil\right)$ *steps;*
2. *The returned α^+ satisfies*
$$\min\left(\gamma\alpha, \frac{\delta}{2L_f}\right) \le \alpha^+ \le \min\left(\gamma\alpha, \frac{\delta}{\mu_f}\right) \le \infty; \tag{8}$$
3. *For any α^+ returned by Algorithm 2, $\bar{\alpha}^+ \in (0, \alpha^+]$ satisfies the backtracking condition as well.*

Algorithm 1

Data: (i) Initialization $\mathbf{X}^0 \in \mathbb{R}^{m\times d}$ and $\mathbf{D}^0 = 0$; (ii) initial value $\alpha_{-1} \in (0, \infty)$; (iii) Backtracking parameters $\delta > 0$; (iv) nondecreasing sequence $\{\gamma^k\}_k \subseteq [1, \infty)$ (v) Gossip matrix $W := (1-c)I_m + c\widetilde{W}$, with $\widetilde{W} \in \mathcal{W}_{\mathcal{G}}$, and $c \in (0, 1/2)$. Set the iteration index $k = 0$.

1: `(S.1) Communication step`: Agents updates primal and dual variables via gossiping:

$$\mathbf{X}^{k+1/2} = W\,\mathbf{X}^k \quad \text{and} \quad \mathbf{D}^{k+1/2} = W\left(\mathbf{D}^k + \nabla F(\mathbf{X}^{k+1/2})\right);$$

2: `(S.2) Decentralized line-search`: Each agent updates α_i^k according to

$$\alpha_i^k = \texttt{Backtracking}\left(\alpha^{k-1}, f_i, x_i^{k+1/2}, d_i^{k+1/2}, \gamma^k, \delta\right);$$

3: `(S.3) Min-consensus`:

$$\alpha^k = \min_{i\in[m]} \alpha_i^k;$$

4: `(S.4) Local updates of the primal and dual variables`:

$$\mathbf{X}^{k+1} = \mathbf{X}^{k+1/2} - \alpha^k \cdot \mathbf{D}^{k+1/2},$$

$$\mathbf{D}^{k+1} = \mathbf{D}^{k+1/2} + \frac{1}{\alpha^k} \cdot \left(\mathbf{X}^k - \mathbf{X}^{k+1} - \alpha^k \nabla F(\mathbf{X}^{k+1/2})\right).$$

5: `(S.5)` If a termination criterion is not met, $k \leftarrow k+1$ and go to step `(S.1)`.

Algorithm 2 Backtracking(α, f, x, d, γ, δ)

1: $\alpha^+ \leftarrow \gamma\alpha$;
2: $x^+ := x - \alpha^+ d$;
3: **while** $f(x^+) > f(x) + \langle \nabla f(x), x^+ - x\rangle + \frac{\delta}{2\alpha^+}\|x^+ - x\|^2$ **do**
4: $\quad \alpha^+ \leftarrow (1/2)\alpha^+$;
5: $\quad x^+ := x - \alpha^+ d$;
return α^+.

Notice that the last statement of the lemma guarantees that the each $\alpha^k = \min_{i\in[m]} \alpha_i^k$ satisfies the descent property (6) on the global loss F^k, as each α_i^k meets the local condition (7).

The sequence $\{\gamma^k\}_{k=1}^{\infty}$ used in line 1 of the backtracking algorithm, with each $\gamma^k \geq 1$, is introduced to favor nonmonotone, and thus potentially larger, stepsize values between two consecutive line-search calls. Any sequence satisfying $\gamma^k \downarrow 1$ and $\prod_{k=1}^{\infty} \gamma_k = \infty$, is advisable. In our experiments, we found the following rule quite effective: $\gamma^k = \left((k+\beta_1)/(k+1)\right)^{\beta_2}$, for some $\beta_2 > 0$ and $\beta_1 \geq 1$. One can opt for $\gamma^k = 1$, for all k, thus eliminating this extra parameter, if simplicity is desired.

On the min-consensus: Step `(S.3)` involves a min-consensus across the network to establish a common stepsize, $\alpha^k = \min_{i\in[m]} \alpha_i^k$, among the agents. This procedure is easily implemented in federated systems, where a server node facilitates information exchange between clients. Interestingly, this min-consensus protocol is also well-suited to current wireless mesh network technologies. Modern networks support multi-interface communications, including WiFi and LoRa (Low-Range) [15, 2, 14]. WiFi allows high-speed, short-range communications, supporting a mesh topology where nodes transmit large data volumes to immediate neighbors. Conversely, LoRa facilitates long-range but low-rate communications, ideal for communication flooding that reaches all network nodes in a single hop but transmits minimal information. Therefore, in multi-interface networks, the proposed algorithm operates by transmitting vector variables in Steps `(S.1)` via WiFi, while LoRa is used for the min-consensus in Step `(S.3)`. Furthermore, the values α_i^k's can be quantized to their nearest lower values using a few bits before transmission. Based on Lemma 3(3), this quantization ensures that the descent condition (6) is still met with the resultant min quantized stepsize. This approach renders the extra communication cost for implementing the min-consensus step negligible.

4 Convergence Results

The strongly convex case: We begin stating convergence under strong convexity of f_i's.

We begin introducing two quantities that help to identify different convergence regimes of the proposed algorithm. Let $(\mathbf{X}^\star, \mathbf{D}^\star)$ be a fixed point of Algorithm 1 (whose existence is ensured by Assumption 1). Define the quantities of interest along the iterates $\{(\mathbf{X}^k, \mathbf{D}^k)\}$ of the algorithm as

$$g^k := \frac{1}{\alpha^k} \frac{\|\mathbf{X}^{k+1/2} - \mathbf{X}^\star\|}{\|c(I - \widetilde{W})\big(\nabla F(\mathbf{X}^{k+1/2}) - \nabla F(\mathbf{X}^\star)\big)\|} \tag{9}$$

and

$$r^k := \frac{\max\Big((\alpha^k)^{-1}\|\mathbf{X}^k\|_{c(I-\widetilde{W})}, \|c(I - \widetilde{W})(\nabla F(\mathbf{X}^{k+1/2}) - \nabla F(\mathbf{X}^\star))\|_M\Big)}{\|c(I - \widetilde{W})(\mathbf{D}^k - \mathbf{D}^\star)\|_M}. \tag{10}$$

where $M := c^{-1}(I - \widetilde{W})^\dagger - I$. Here, g^k assesses the quality of the selected stepsize α^k in approximating the inverse of the Lipschitz constant of $(I - \widetilde{W})\nabla F$ along the direction $\mathbf{X}^{k+1/2} - \mathbf{X}^\star$. It captures network and optimization quantities. It follows from Lemma 3 that $g_k \geq 1/\kappa$ (when $\delta = 1$). The quantity r^k reflects the convergence progress of the dual variables $\mathbf{D}^k$. Rewriting the update for these variables as $\mathbf{D}^{k+1} = \mathbf{D}^k + \frac{c}{\alpha^k}(I - \widetilde{W})\mathbf{X}^k - c(I - \widetilde{W})\left(\nabla F(\mathbf{X}^{k+1/2}) + \mathbf{D}^k\right)$, we claim that small values of $\|\frac{c}{\alpha^k}(I - \widetilde{W})\mathbf{X}^k - c(I - \widetilde{W})\left(\nabla F(\mathbf{X}^{k+1/2}) + \mathbf{D}^k\right)\|$ compared to $\|\mathbf{D}^k - \mathbf{D}^\star\|$ (i.e., small r^k values), indicate slow improvements of the dual variables towards convergence. Conversely, large values of r^k suggest rapid dual convergence. This is made formal in Lemma 9 in the appendix. We remark that neither g^k nor r^k need to be known by the agents; they are instrumental only for analysis and posterior assessment of algorithm convergence.

Linear convergence is established below via contraction of the following merit function along the iterates $\{(\mathbf{X}^k, \mathbf{D}^k)\}$ of the algorithm

$$V^k := \left\|\mathbf{X}^k - \mathbf{X}^\star\right\|^2 + (\alpha^{k-1})^2\|\mathbf{D}^k - \mathbf{D}^\star\|_M^2. \tag{11}$$

Notice that (i) $\mathbf{D}^k, \mathbf{D}^\star \in \mathtt{span}(I - \widetilde{W})$, for all k; hence, $\|\mathbf{D}^k - \mathbf{D}^\star\|_M = 0$ if and only if $\mathbf{D}^k = \mathbf{D}^\star$; and (ii) under Assumption 1, it must be $\mathbf{X}^\star = 1(x^\star)^\top$, where $x^\star$ is the solution of Problem (P).

Theorem 4. *Consider Problem (P) under Assumption 1, with $\mu > 0$. Let $\{(\mathbf{X}^k, \mathbf{D}^k)\}$ be the sequence generated by Algorithm 1, with parameters: $\delta = 1$, $c \leq 1/2$, $\{\gamma^k \geq 1\}$ being arbitrary, and $\widetilde{W} \in \mathcal{W}_{\mathcal{G}}$. Then, the following holds:*

$$V^{k+1} \leq \left(1 - \rho^k\right) \max\left(1, (\alpha^k/\alpha^{k-1})^2\right)\, V^k, \tag{12}$$

where

$$\rho^k := \min\left(\mu\,\alpha^k\, \frac{(1 - c(1 - \lambda_n(\widetilde{W})))^2}{2}, \max\left((r^k)^2, \mu\,\alpha^k\, \frac{(g^k\,[1 - r^k]_+)^2}{2}\right)c^2(1 - \lambda_2(\widetilde{W}))^2\right).$$

The theorem establishes linear convergence of Algorithm 1. As $\max(1, (\alpha^k/\alpha^{k-1})^2)$ is bounded away from zero and uniformly upper bounded (with value depending on the sequence $\{\gamma^k\}$)–see Lemma 3–the convergence rate is predominantly determined by ρ^k. Within the setting of the theorem, $\rho^k \in (0, 1)$. Intriguingly, ρ^k is, in particular, affected by the values of r^k and g^k, which implies that the algorithm may exhibit different operational regimes based on the range of values these parameters take along the trajectory of the algorithm. The following result highlights this distinctive aspect.

Corollary 4.1. *Instate Theorem 4, with $\{\gamma^k\}$ being chosen such that $\gamma_k \leq \left((k + \beta_1)/(k+1)\right)^{\beta_2}$, for all k and some $\beta_1 \geq 1, \beta_2 > 0$. Then $\left\|\mathbf{X}^{N+1} - \mathbf{X}^\star\right\|^2 + \frac{1}{4L^2}\|\mathbf{D}^{N+1} - \mathbf{D}^\star\|_M^2 \leq \varepsilon$, with the number of iterations N bounded as follows:*

1. *If $r^k \geq 1/2$ for all k, then $N = O\left(\max\left(\frac{\kappa}{(1-c(1-\lambda_m(\widetilde{W})))^2}, \frac{1}{c^2(1-\lambda_2(\widetilde{W}))^2}\right)\log(V^0/\varepsilon)\right)$;*
2. *If $r^k \geq (1/4)\sqrt{\kappa}$ or $g^k \geq 1/2$, for all k, then*

$$N = O\left(\frac{\kappa}{\min\left(c(1 - \lambda_2(\widetilde{W})), (1 - c(1 - \lambda_m(\widetilde{W})))\right)^2}\log(V^0/\varepsilon)\right);$$

3. *Otherwise,* $N = O\left(\left(\frac{\kappa}{\min\left(c,(1-c(1-\lambda_m(\widetilde{W})))\right)\cdot(1-\lambda_2(\widetilde{W}))}\right)^2 \log(V^0/\varepsilon)\right)$.

Corollary 4.1 identifies different operational regimes of the algorithm, each resulting in difference performance based upon the network connectivity and optimization condition number. Specifically,

(1) Strong connectivity regime: when $r^k \geq 1/2$ for all k, a fact that numerically has been consistently observed for 'relatively good' network connectivity, the convergence rate exhibits a separation in the dependence on the network and optimization parameters. Noticing $1 - c(1 - \lambda_m(\widetilde{W})) > 1 - 2c$, when $c(1 - \lambda_2(\widetilde{W})) \geq (1 - 2c)/\sqrt{\kappa}$, the rate of the algorithm reduced to $\mathcal{O}(\kappa)$, matching that of the centralized gradient algorithm. This suggests scenarios where the optimization problem is harder than a consensus problem over the network, resulting in the bottleneck between the two. Conversely, the rate is dominated by the consensus algorithm's rate–$\mathcal{O}((1 - \lambda_2(\widetilde{W}))^{-2})$– when the condition number κ is large relative to the network connectivity $1 - \lambda_2(\widetilde{W})$. Quite interestingly, this rate separation property mirrors the convergence behaviour of certain *nonadaptive* primal-dual decentralized schemes including NEXT [11], AugDGM [44], Exact Diffusion [45] (with rate improved in [43]), NIDS [24], and ABC [43].

(2) Intermediate connectivity regime: In networks with 'moderate' connectivity and effective stepsize adaptivity ($g^k \geq 1/2$), generally the algorithm achieves convergence rates of the order $\mathcal{O}(\kappa/(1 - \lambda_2(\widetilde{W}))^2)$, where optimization and network parameters are now mixed. This rate aligns with those of *nonadaptive* decentralized gradient-tracking schemes, such as DGing [32], SONATA [40] (subject to sufficiently small network connectivity), and [35].

(3) Worst-case regime: This regime reflects the algorithm's worst-case performance, with a quadratic scaling of the rate with the condition number κ, typically registered in poorly connected networks. Such performance degradation aligns with the worst-case rates proved in schemes like SONATA [40].

In summary, the proposed algorithm achieves convergence rates of the same order of those of most non-accelerated decentralized algorithms, importantly, *without* requiring knowledge of network and optimization parameters or the specific values of r^k and g^k. To the best of our knowledge this is the first decentralized algorithm of its kind to combine such desirable properties.

Weakly convex functions: We complete the characterization of the proposed algorithm considering weakly convex functions. The main result is summarized next.

Theorem 5. *Consider Problem* (P) *under Assumption 1, with* $\mu = 0$*. Let* $\{(\mathbf{X}^k, \mathbf{D}^k)\}$ *be the sequence generated by Algorithm 1, with parameters:* $\delta < 1$ *and* $\gamma_k \leq \big((k + \beta_1)/(k + 1)\big)^{\beta_2}$*, for all* k *and some* $\beta_1 \geq 1, \beta_2 > 0$ *such that* $r := 2\beta_2\lceil\beta_1\rceil < 1$*,* $c \leq 1/2$*, and* $\widetilde{W} \in \mathcal{G}_{\mathcal{W}}$*. Then, the following holds:*

$$\min_{j\in[k]}\left(\|\mathbf{X}^j - \mathbf{X}^{j+1}\|^2 + \frac{\delta}{2L}\|\mathbf{D}^j - \mathbf{D}^{j+1}\|_M^2\right) \leq \frac{c'V^0}{(k+1)^{1-r}}, \text{ with } c' = \frac{1}{1-\delta}\left(\frac{\lceil\beta_1\rceil^{\lceil\beta_1\rceil}}{(\lceil\beta_1\rceil + 1)!}\right)^{2\beta_2}.$$

Furthermore, one can check that if the sequence $\{\gamma^k\}$ is chosen such that $\prod_k \gamma \leq \ln k$, the merit function above decays at the rate of $(\ln k)/(k + 1)$. This rates are inline with those obtained by certain decentralized primal-dual methods applied to convex optimization problems.

Remark 6. *It is important to note that although the above results are presented under Assumption 1, the same conclusions drawn in Theorem 4 and Theorem 5 also hold under the significantly weaker condition that each* f_i *is locally smooth (and locally strongly convex)–see the appendix for details. Specifically, in the rate expressions mentioned earlier, the global condition number* κ *and the global smooth constant* L *are replaced by are replaced by their local counterparts, which are generally much smaller and defined on the convex hull of the set* $\{\mathbf{X}^\star, \{\mathbf{X}^k, \mathbf{X}^{k+1/2}\}_{k=0}^N\}$*. This adjustment highlights the algorithm's capability to adapt to the local geometry of the optimization problem. Such a nuanced approach offers more favorable rate dependencies compared to those found in the existing decentralized optimization literature.*

5 Numerical Results

In this section, we present some preliminary numerical results. We compare Algorithm 1 with EXTRA [39] and NIDS [24] on a ridge regression problem using synthetic data, and logistic regression on real data from the a3a dataset [6].

Ridge regression: This strongly convex instance of (P) is defined for each agent i by the function $f_i(x) = \|A_i x_i - b_i\|^2 + \sigma\|x_i\|_2^2$, where $A_i \in \mathbb{R}^{20\times300}, b_i \in \mathbb{R}^{20}$, and $\sigma > 0$ is the regularization parameter. The elements of A_i, b_i were independently sampled from the standard normal distribution; the regularization is set to $\sigma = 0.1$. We simulated a network of $m = 20$ agents, and the following three different graph topologies, reflecting varying connectivity levels:: **(i)** $\mathcal{G}_1$: Graph-path with $m-1$ edges and diameter $m-1$, i.e., $\mathcal{G} = \{[m], \{(i, i+1)\}_{i=1}^{m-1}\}$; **(ii)** $\mathcal{G}_2$: Erdős–Rényi graph, sparsely connected; and **(iii)** $\mathcal{G}_3$: Erdős–Rényi graph, well-connected. These setups help to evaluate the performance of the algorithm under low, moderate, and high network connectivity.

The comparison of the three algorithms is summarized in Fig. 1 and Fig. 2. For EXTRA and NIDS we use the nominal stepsize tuning as recommended in their respectively papers, which requires full knowledge of the optimization parameters L, μ and eigen-spectrum of the gossip matrix. Algorithm 1 is simulated under the following choice of the line-search parameters: $\gamma^k = (k+2)/(k+1)$, and $\beta_1 = \beta_2 = 1$. For all the algorithm we used the Metropolis-Hastings weight matrix $W \in \mathcal{G}_{\mathcal{W}}$ [31].

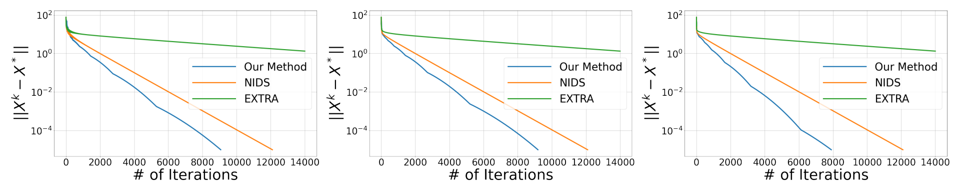

Figure 1: Ridge regression ($\kappa = 2\times10^3$) over $\mathcal{G}_1$ (left panel), $\mathcal{G}_2$ (mid panel), and $\mathcal{G}_3$ (right panel): optimization error $\|\mathbf{X}^k - \mathbf{X}^\star\|$ versus iterations k.

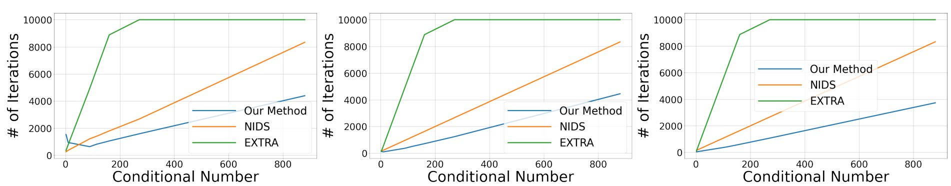

Figure 2: Ridge regression over $\mathcal{G}_1$ (left panel), $\mathcal{G}_2$ (mid panel), and $\mathcal{G}_3$ (right panel): number of iterations N for $\|\mathbf{X}^N - \mathbf{X}^\star\| \leq 10^{-5}$ versus the condition number κ.

The figures clearly demonstrate that the proposed method consistently outperforms both EXTRA and NIDS; the gap becomes quite significant as the condition number κ grows. This performance is particularly noteworthy given that Algorithm 1 operates effectively without requiring tedious tuning or global knowledge of the optimization and network parameters.

Logistic regression: This is an instance of (P), where $f_i(x) = (1/m)\sum_{j=1}^m \log(1 + \exp(-y_{i,j}\langle f_{i,j}, x\rangle))$. Here, $y_{ij} \in \{0,1\}$, $f_{ij} \in \mathbb{R}^{200}$ are data problem, taken from the dataset a3a [6]. We distribute data across $m = 20$ nodes, each owning $n = 159$ samples. We simulated the same three network topologies, $\mathcal{G}_1$, $\mathcal{G}_2$, and $\mathcal{G}_3$, as for the ridge regression problem. Results are summarized in Fig. 3. The tuning of the algorithms is as discussed above for the ridge regression problem. The figures show that our method compare favorably with EXTRA and NIDS also on this class of problems and on real data. All experiments above are run on Acer Swift 5 SF514-55TA-56B6 with processor Intel(R) Core(TM) i5-8250U @ CPU 1.60GHz, 1800 MHz.

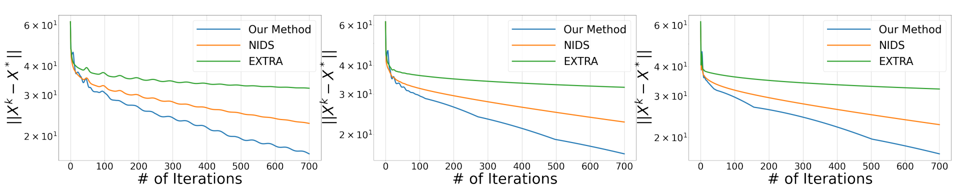

Figure 3: Logistic regression over $\mathcal{G}_1$ (left panel), $\mathcal{G}_2$ (mid panel), and $\mathcal{G}_3$ (right panel): optimization error $\|\mathbf{X}^k - \mathbf{X}^\star\|$ versus iteration k.

References

[1] S. A. Alghunaim, E. K. Ryu, K. Yuan, and A. H. Sayed. Decentralized proximal gradient algorithms with linear convergence rates. *IEEE Transactions on Automatic Control*, 66(6):2787–2794, June 2021.

[2] A. Askhedkar, B. Chaudhari, and M. Zennaro. *Hardware and software platforms for low-power wide-area networks*, page 397–407. Elsevier, 2020.

[3] J. Barzilai and J. M. Borwein. Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, 8(1):141–148, 1988.

[4] H. H. Bauschke and P.L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. CMS Books in Mathematics. Springer New York, New York, NY, 2011.

[5] Y. Carmon and O. Hinder. Making sgd parameter-free. In *Proceedings of Thirty Fifth Conference on Learning Theory*, volume 178 of *Proceedings of Machine Learning Research*, pages 2360–2389. PMLR, 02–05 Jul 2022.

[6] C. Chang and C. Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2, 07 2007.

[7] T. Chang, M. Hong, H. Wai, X. Zhang, and S. Lu. Distributed learning in the nonconvex world: From batch data to streaming and beyond. *IEEE Signal Processing Magazine*, 37(3):26–38, 2020.

[8] X. Chen, B. Karimi, W. Zhao, and P. Li. On the convergence of decentralized adaptive gradient methods. In *Asian Conference on Machine Learning*, pages 217–232. PMLR, 2023.

[9] X. Chen, X. Li, and P. Li. Toward communication efficient adaptive gradient method. In *Proceedings of the 2020 ACM-IMS on Foundations of Data Science Conference*, page 119–128, Virtual Event USA, October 2020. ACM.

[10] A. Cutkosky and H. Mehta. Momentum improves normalized SGD. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 2260–2268. PMLR, 13–18 Jul 2020.

[11] P. Di Lorenzo and G. Scutari. NEXT: In-network nonconvex optimization. *IEEE Trans. Signal Inf. Process. Netw.*, 2(2):120–136, June 2016.

[12] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. In *Proceedings of the 24th International Conference on Neural Information Processing Systems*, pages 257–265, 2011.

[13] M. Ivgi, O. Hinder, and Y. Carmon. DoG is SGD's best friend: A parameter-free dynamic step size schedule. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 14465–14499. PMLR, 23–29 Jul 2023.

[14] T. Janssen, N. BniLam, M. Aernouts, R. Berkvens, and M. Weyn. Lora 2.4 ghz communication link and range. *Sensors*, 20(16):4366, August 2020.

[15] D.H. Kim, J.Y. Lim, and J.D. Kim. Low-power, long-range, high-data transmission using wi-fi and lora. In *2016 6th International Conference on IT Convergence and Security (ICITCS)*, page 1–3, Prague, Czech Republic, September 2016. IEEE.

[16] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[17] P. Latafat, A. Themelis, and P. Patrinos. Adaptive proximal algorithms for convex optimization under local lipschitz continuity of the gradient. *arXiv preprint arXiv:2301.04431*, 2023.

[18] P. Latafat, A. Themelis, and P. Patrinos. On the convergence of adaptive first order methods: proximal gradient and alternating minimization algorithms. *arXiv preprint arXiv:2311.18431*, 2023.

[19] J. Li, X. Chen, S. Ma, and M. Hong. Problem-parameter-free decentralized nonconvex stochastic optimization. *arXiv preprint arXiv:2402.08821*, 2024.

[20] T. Li and G. Lan. A simple uniformly optimal method without line search for convex optimization. *arXiv preprint arXiv:2310.10082*, 2023.

[21] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

[22] X. Li, B. Karimi, and P. Li. On distributed adaptive optimization with gradient compression. In *International Conference on Learning Representations (ICLR)*, 2022.

[23] X. Li and F. Orabona. On the convergence of stochastic gradient descent with adaptive stepsizes. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTAT)*. PMLR, 2019.

[24] Z. Li, W. Shi, and M. Yan. A decentralized proximal-gradient method with network independent step-sizes and separated convergence rates. *IEEE Transactions on Signal Processing*, 67(17):4494–4506, 2019.

[25] X. Lian, C. Zhang, H. Zhang, C. Hsieh, W. Zhang, and J. Liu. Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pages 5330–5340, 2017.

[26] L. Luo, Y. Xiong, Y. Liu, and X. Sun. Adaptive gradient methods with dynamic bound of learning rate. In *Proceedings of the 7th International Conference on Learning Representations*, New Orleans, Louisiana, May 2019.

[27] Y. Malitsky and K. Mishchenko. Adaptive gradient descent without descent. In *International Conference on Machine Learning*, 2019.

[28] Y. Malitsky and K. Mishchenko. Adaptive proximal gradient method for convex optimization. *arXiv preprint arXiv:2308.02261*, 2024.

[29] P. Nazari, D.A. Tarzanagh, and G. Michailidis. Dadam: A consensus-based distributed adaptive gradient method for online optimization. *IEEE Transactions on Signal Processing*, 70:6065–6079, 2022.

[30] A. Nedic. Distributed gradient methods for convex machine learning problems in networks: Distributed optimization. *IEEE Signal Processing Magazine*, 37(3):92–101, 2020.

[31] A. Nedić, A. Olshevsky, and M. Rabbat. Network topology and communication-computation tradeoffs in decentralized optimization. *Proceedings of the IEEE*, 106:953–976, 2018.

[32] A. Nedić, A. Olshevsky, and W. Shi. Achieving geometric convergence for distributed optimization over time-varying graphs. *SIAM Journal on Optimization*, 27:2597–2633, July 2016.

[33] J. Nocedal and S. Wright. *Numerical Optimization*. Springer, 2 edition, 2006.

[34] B.T. Polyak. Minimization of unsmooth functionals. *USSR Computational Mathematics and Mathematical Physics*, 9(3):14–29, 1969.

[35] G. Qu and N. Li. Harnessing smoothness to accelerate distributed optimization. *IEEE Transactions on Control of Network Systems*, 5(3):1245–1260, Sept 2018.

[36] S. Reddi, Z. Burr Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konev$\mathbf{v}cn$, S. Kumar, and B. McMahan. Adaptive federated optimization. In *International Conference on Learning Representations (ICLR)*, 2021.

[37] S. J. Reddi, S. Kale, and S. Kumar. On the convergence of adam and beyond. In *International Conference on Learning Representations (ICLR)*, 2018.

[38] A. H. Sayed. Adaptation, learning, and optimization over networks. *Foundations and Trends in Machine Learning*, 7:311–801, January 2014.

[39] W. Shi, Q. Ling, G. Wu, and W. Yin. EXTRA: An exact first-order algorithm for decentralized consensus optimization. *SIAM J. on Optimization*, 25(2):944–966, November 2015.

[40] Y. Sun, G. Scutari, and A. Daneshmand. Distributed optimization based on gradient tracking revisited: Enhancing convergence rate via surrogation. *SIAM Journal on Optimization*, 32(2):354–385, June 2022.

[41] R. Ward, X. Wu, and L. Bottou. Adagrad stepsizes: Sharp convergence over nonconvex landscapes. *The Journal of Machine Learning Research*, 21:1–30, 2020.

[42] R. Xin, S. Pu, A. Nedic, and U. A. Khan. A general framework for decentralized optimization with first-order methods. *Proceedings of the IEEE*, 108(11):1869–1889, November 2020.

[43] J. Xu, Y. Tian, Y. Sun, and G. Scutari. Distributed algorithms for composite optimization: Unified framework and convergence analysis. *IEEE Transactions on Signal Processing*, 69:3555–3570, 2021.

[44] J. Xu, S. Zhu, Y.-C. Soh, and L. Xie. Augmented distributed gradient methods for multi-agent optimization under uncoordinated constant stepsizes. In *Proceedings of the 54th IEEE Conference on Decision and Control*, pages 2055–2060, 2015.

[45] K. Yuan, B. Ying, X. Zhao, and A. H. Sayed. Exact diffusion for distributed optimization and learning–part i: Algorithm development. *IEEE Transactions on Signal Processing*, 67(3):708–723, 2018.

[46] D. Zhou, S. Ma, and J. Yang. Adabb: Adaptive barzilai-borwein method for convex optimization. *arXiv preprint arXiv:2401.08024*, 2024.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and precede the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS paper checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Abstract gives accurate presentation of our result. Part Major contributions of Introduction contains full description of our work.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer:[Yes]

Justification: The main limitaion of proposed procedure is min-consensus. The technology for its implementation is carefully discribed in part 3.1.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Main assumptions and definitions are presented in Section 2. All main theoretical results presented in Section 4 with all required assumptions. Proofs are placed in Appendix A-F because of their large size.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

Justification: All setup for numerical experiments are described in Section 5. It is enough to reproduce all experiments.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: code in the form of an attached archive.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.

- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Our paper demonstrates performance of optimization algorithm. Because of that, we do not need test some models. But Section 5 contains full information about our experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Numerical experiments demonstrate performance of optimization algorithm on a given problems. Besides, our algorithm is deterministic.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Information is given at the end of Section 5.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: Authors are familiar with NeurIPS Code of Ethics and paper conform it.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: There are different methods of distributed optimization. The paper propose new method of distributed optimization that has no additional societal impact as the authors think.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The proposed method does not require safeguard.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Numerical experiments use one of datasets from LIBSVM. Authors cite corresponding work of owners (see reference [6] in Section 5 and References)

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification:contains contains README file with sufficient description.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.

- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: Paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Paper does not involve crowdsourcing nor research with human subjects

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.