ADAPTIVE CONSENSUS: A NETWORK PRUNING APPROACH FOR DECENTRALIZED OPTIMIZATION

SUHAIL M. SHAH[†], ALBERT S. BERAHAS[‡], AND RAGHU BOLLAPRAGADA[†]

Abstract. We consider network-based decentralized optimization problems, where each node in the network possesses a local function and the objective is to collectively attain a consensus solution that minimizes the sum of all the local functions. A major challenge in decentralized optimization is the reliance on communication which remains a considerable bottleneck in many applications. To address this challenge, we propose an adaptive randomized communication-efficient algorithmic framework that reduces the volume of communication by periodically tracking the disagreement error and judiciously selecting the most influential and effective edges at each node for communication. Within this framework, we present two algorithms: Adaptive Consensus (`AC`) to solve the consensus problem and Adaptive Consensus based Gradient Tracking (`AC-GT`) to solve smooth strongly convex decentralized optimization problems. We establish strong theoretical convergence guarantees for the proposed algorithms and quantify their performance in terms of various algorithmic parameters under standard assumptions. Finally, numerical experiments showcase the effectiveness of the framework in significantly reducing the information exchange required to achieve a consensus solution.

Key words. Decentralized Optimization, Gradient Tracking Methods, Adaptive Consensus

MSC codes. 49M37, 65K05, 90C06, 90C25, 90C30, 90C35

1. Introduction.

The problem of network-based decentralized optimization can be formally stated as,

$$
\begin{aligned}
(1.1)\qquad & \min_{x_i \in \mathbb{R}^d} \quad \frac{1}{n}\sum_{i=1}^{n} f_i(x_i) \\
& \text{s.t.} \quad x_i = x_j,\ \forall\, i, j \in [n] := \{1, 2, \cdots, n\},
\end{aligned}
$$

where $f_i(\cdot) : \mathbb{R}^d \to \mathbb{R}$ is a component of the objective function located at node $i \in [n]$, and $x_i \in \mathbb{R}^d$ is a copy of the optimization variable at node $i \in [n]$. A closely related yet simplified version of this problem, whose goal is to reach consensus among the nodes, i.e., $x_i = x_j$ for all $i \in [n]$, without minimizing an objective function, is referred to as the consensus problem [43]. Problems of these types arise in several applications including wireless sensor networks [38, 46], power systems design [21, 31], parallel computing [8, 15], and robotics [3, 11]. More recently, decentralized optimization has experienced renewed interest owing to the abundance of decentralized data and privacy-preserving machine learning [23, 44], where f_i is a function of the data held by node $i \in [n]$. Several classes of decentralized optimization algorithms have been proposed to solve (1.1), where the main components consist of local computations at every node and information exchange (communication) between nodes in order to achieve consensus [8]. The communication requirement in many applications remains a major bottleneck in the performance of decentralized optimization methods [27, 32, 35, 41, 42, 51].

In this work, we propose and develop a novel approach to reduce the communication requirements in decentralized optimization without significantly impacting the convergence properties of the underlying algorithm. The core principle of our

[†]Operations Research and Industrial Engineering Program, UT Austin, Austin, TX, USA; E-mail: suhailshah2005@gmail.com, raghu.bollapragada@utexas.edu.

[‡]Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI, USA; E-mail: albertberahas@gmail.com.

approach involves judiciously selecting a subset of the edges of the network (instead of all the edges) along which communication is performed at each iteration, thereby reducing the communication efforts. A key observation motivating this approach is that selectively pruning the edges of the network has marginal impact on the spectral properties of the mixing matrix associated with any graph topology. This matrix plays a crucial role in determining the rate of information diffusion through the network [51], which subsequently affects the rate of achieving consensus amongst nodes. In fact, for many network structures, the spectral properties remain virtually unchanged even after selectively pruning up to 50-60% of the edges (see Section 4.1), thus retaining a consensus rate akin to that of an unpruned network while reducing the communication volume.

However, to fully leverage the potential of such pruning approaches, one requires information about the most influential edges, i.e., the edges that achieve consensus with minimal communication cost, information that is typically unknown. For example, the bridge edge that connects two fully connected components in a barbell graph [22, Figure 2] has a significantly more influential role in the consensus process than other edges. Therefore, it is beneficial to communicate along the bridge edge as compared to other edges. Unfortunately, due to the decentralized nature of the network, nodes cannot a priori determine these influential edges. Moreover, the relative influence of different edges in achieving consensus can vary significantly depending on the network state and structure, and the application. To overcome this challenge, our work proposes a cyclic adaptive randomized procedure that can be implemented in a decentralized manner to identify such edges and reduce the communication costs. Specifically, we periodically track the *disagreement error* along edges during the consensus process to estimate the relative importance of edges in achieving consensus and maintain a network with only the most influential edges.

1.1. Contributions. A concise summary of the contributions is as follows:

- We propose an adaptive communication-efficient algorithmic framework. Within this framework, we introduce two new algorithms: Adaptive Consensus (`AC`) to solve the consensus problem and Adaptive Consensus based Gradient Tracking (`AC-GT`) to solve the decentralized optimization problem[1]. The novelty in our approach lies in the ability to exploit the underlying structure of the network to reduce the volume of communication. This is accomplished via an adaptive consensus scheme that selects the most influential and effective edges for communication at each node based on the graph topology. The proposed framework has broad applicability and can be integrated with other existing decentralized optimization algorithms or adapted to other settings including directed graphs, time-varying topologies, and asynchronous updates.
- We provide theoretical convergence guarantees for smooth strongly convex problems for both `AC` and `AC-GT`, demonstrating that they retain the linear convergence properties of their base counterparts, i.e., methods that do not utilize the adaptive consensus framework, while requiring reduced communication. The analysis utilizes the inhomogeneous matrix product theory to prove linear convergence by showing that the pruned matrix products remain contractive. In contrast to prevalent analytical approaches in decentralized optimization with time-varying graphs, the rate constant in our results is obtained using the coefficient of ergodicity which effectively highlights the dependence of the convergence

[1]For better exposition of the consensus framework, the consensus and decentralized optimization problems are treated separately even though the former is a simplified version of the latter.

rate on the network pruning procedure parameters.

- We illustrate the empirical performance of AC in solving the standard consensus problem and of AC-GT in solving linear regression and binary classification logistic regression problems. Our numerical results highlight that the proposed methods achieve significant communication savings while maintaining solution quality, compared to the contemporary state-of-the-art techniques.

1.2. Literature Review. The proposed idea of exploiting the relative significance of edges to improve algorithmic efficiency is not exclusive to decentralized optimization and has been studied in other fields that use graphical modeling on networks [17, 18, 30, 50]. In the context of traffic modeling, a converse analogue falls under the category of "Braess's paradox", which suggests that adding one or more roads to a road network can actually slow down the overall traffic flow [17, 50]. Another example, although somewhat tangential, is found in neural networks where the "lottery ticket hypothesis" states that within dense, feed-forward networks, there are smaller pruned sub-networks that, when trained in isolation, can achieve test accuracy comparable to the original network in a similar number of iterations [18, 30].

Within decentralized optimization, several recent works have proposed communication-efficient algorithms that balance the communication and computation costs to achieve overall efficiency [4–7, 10, 45, 57]. Our proposed approaches are complementary to and can be integrated with these existing works. Furthermore, the proposed framework (adaptive consensus) adds to the list of techniques that reduce the communication costs. One such approach is gossip communication protocols where nodes selectively communicate with neighbors asynchronously [9, 12, 53, 54]. It is worth noting that in gossip protocols a convex optimization problem is often solved to optimize the spectral gap of the expected consensus matrix [9]. Another class of approaches leverage quantized communication where only quantized (reduced size) information is communicated to reduce the communication costs. However, these techniques typically lack convergence guarantees to the solution [8, 48]. Moreover, quantization techniques can also be incorporated into our framework to further reduce the communication overhead. We emphasize that our approach differs significantly from the aforementioned approaches in several ways including the focus on enhancing communication efficiency by adaptively modifying the graph structure in a decentralized manner, and achieving convergence guarantees to the solution.

While several classes of algorithms have been proposed for solving decentralized optimization, gradient tracking methods have emerged as popular alternatives due to their simplicity, optimal theoretical convergence properties and empirical performance [4, 13, 26, 34, 49, 56]. We incorporate the proposed communication-efficient technique into the gradient tracking algorithmic framework with the goal of reducing the communication costs while retaining optimal convergence guarantees. Furthermore, we note that the setting of time-varying graphs, which also arises in our work, has been explored previously in [1, 33, 34, 52], among others.

1.3. Organization. The paper is organized as follows. In the remainder of this section, we define the notation employed in the paper. In Section 2, we describe the network model, introduce the Adaptive Consensus (AC) algorithm, and establish convergence guarantees under standard assumptions. Building upon the adaptive consensus procedure and gradient tracking algorithms, we propose the Adaptive Consensus based Gradient Tracking (AC-GT) algorithm and study its convergence properties in Section 3. Section 4 presents numerical results that illustrate the performance of the proposed algorithms. Finally, concluding remarks are provided in Section 5.

1.4. Notation. We use $\mathbb{R}$ to denote the set of real numbers and $\mathbb{N}$ to denote the set of all strictly positive integers. The ℓ_2-inner product between two vectors is denoted by $\langle \cdot, \cdot \rangle$ and $\otimes$ denotes the Kronecker product between two matrices. All norms, unless otherwise specified, can be assumed to be ℓ_2-norms of a vector or matrix depending on the argument. Let $\lfloor x \rfloor$ ($\lceil x \rceil$) denote the nearest integer less (greater) than or equal to x. We use $a|b$ to denote integer division between any two $a, b \in \mathbb{N}$, i.e., $a|b = \lfloor a/b \rfloor$. We use $\mathbf{1}_n := \frac{1}{n} 1_n \otimes I_d \in \mathbb{R}^{nd \times d}$, where $1_n \in \mathbb{R}^n$ is the column vector of all ones and $I_d \in \mathbb{R}^{d \times d}$ is the $d \times d$ identity matrix. For any matrix Q with eigenvalues $-1 < \lambda_n \leq \cdots \leq \lambda_2 < \lambda_1 = 1$, the *spectral gap* is defined as $\sigma(Q) := 1 - \max\{|\lambda_n|, |\lambda_2|\}$. The set $A \setminus B$ consists of the elements of A which are not elements of B. We use x^* denotes the optimal solution of (1.1). We use the column vector $x_{i,k} \in \mathbb{R}^d$ to denote the value of the objective variable held by node i at iteration k. The vector $\mathbf{x}_k \in \mathbb{R}^{nd}$ denotes the column-stacked version of $x_{i,k}$ and $\nabla \mathbf{f}(\mathbf{x}_k)$ denotes the column-stacked gradients, i.e.,

$$\mathbf{x}_k := [x_{1,k}, \cdots, x_{n,k}] \in \mathbb{R}^{nd} \quad \text{and} \quad \nabla \mathbf{f}(\mathbf{x}_k) := \left[\nabla f_1(x_{1,k}), \cdots \nabla f_n(x_{n,k})\right] \in \mathbb{R}^{nd},$$

where $\nabla f_i : \mathbb{R}^d \to \mathbb{R}^d$ is the gradient of the local function f_i. The following quantities are used in the presentation and analysis of the algorithms,

$$\bar{x}_k := \frac{1}{n} \sum_{i=1}^{n} x_{i,k} \in \mathbb{R}^d, \;\; \bar{\mathbf{x}}_k = [\bar{x}_k, \cdots, \bar{x}_k] \in \mathbb{R}^{nd}, \;\; \nabla f(\bar{x}_k) := \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(\bar{x}_k) \in \mathbb{R}^d.$$

2. Adaptive Consensus.

This section provides a description of the pruning protocol which serves as the basic building block for the proposed consensus scheme referred to as the Adaptive Consensus algorithm (Algorithm 2.2, `ADAPTIVE CONSENSUS (AC)`). We describe the network model we assume in the paper, discuss the pruning protocol, and present the algorithm and its associated convergence guarantees.

2.1. Network Model. The underlying network is assumed to be modeled by a undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of edges. We use the matrix $Q = [q_{ij}]_{i \in [n], j \in [n]}$ to denote the mixing matrix. The mixing matrix has the following properties: the entry $q_{ij} > 0$ (assumed to be equal to q_{ji}) if there is a link between any two nodes i, $j \in \mathcal{V}$. We use $\mathcal{E}_i$ to denote the set of all edges (i,j) such that $j \in \mathcal{V}$ is a neighbor of $i \in \mathcal{V}$, i.e., the set of all $j \in \mathcal{V}$ with $j \neq i$ for which $q_{ij} > 0$. Note that the neighbors of i for any $i \in [n]$ is the set of all j such that $(i,j) \in \mathcal{E}_i$. Since we assume that the graph is undirected, $(i,j) \in \mathcal{E}_i$ if and only if $(j,i) \in \mathcal{E}_j$. We make the following assumption on the network.

ASSUMPTION 2.1 (Graph Connectivity). *$\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ is static and connected.*

2.2. Pruning Protocol. The main goal of the pruning protocol is to provide a systematic approach for selecting the (subset of) edges within a graph along which to communicate in order to achieve consensus with reduced communication efforts. To be more precise, given the reference graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ and a set of node estimates a_i for all $i \in [n]$, the pruning protocol generates a modified graph $\mathcal{G}(\mathcal{V}, \bar{\mathcal{E}})$ by selectively removing edges from the reference graph. The edges to be pruned are determined by a function of the node estimates. The function assigns a probability to each edge in $\mathcal{E}$ based on its likelihood of being least effective and influential with respect to achieving consensus. The pseudo-code for the pruning protocol is given in Algorithm 2.1.

Algorithm 2.1 has three free (user-defined) parameters ($\bar{\kappa}_i$, $\underline{\kappa}_i$ and β). Broadly speaking, $\bar{\kappa}_i \in [0, 1]$ represents the fraction of edges to be pruned at node $i \in [n]$ and

Algorithm 2.1 PRUNING PROTOCOL$(\mathcal{G}(\mathcal{V},\mathcal{E}), a_i, (\bar{\kappa}_i, \underline{\kappa}_i), \beta)$.

Inputs: Graph $\mathcal{G}(\mathcal{V},\mathcal{E})$; Node estimates a_i for all $i \in [n]$; Softmax parameter $\beta \in [0,\infty]$; Thresholding factors $(\bar{\kappa}_i, \underline{\kappa}_i) \in [0,1]^2$ for all $i \in [n]$.

Set $\mathcal{E}_i^{\text{prune}} := \{\}$ for all $i \in [n]$.
for all $i \in [n]$ in parallel **do**
  Receive estimates a_j from all neighbors j.
  Compute a dissimilarity measure $\Delta(a_i, a_j)$ for all edges $(i,j) \in \mathcal{E}_i$.
  while $|\mathcal{E}_i^{\text{prune}}| \le \lfloor \bar{\kappa}_i \times |\mathcal{E}_i| \rfloor$ **do**
    Draw a sample edge (i,j') from $\mathcal{E}_i \setminus \mathcal{E}_i^{\text{prune}}$ according to:

$$p_{i,j} \sim \frac{\exp(-\beta\Delta(a_i,a_j))}{\sum_{(i,j')\in\mathcal{E}_i\setminus\mathcal{E}_i^{\text{prune}}} \exp(-\beta\Delta(a_i,a_{j'}))}, \qquad ((i,j) \in \mathcal{E}_i \setminus \mathcal{E}_i^{\text{prune}}).$$

    Update set $\mathcal{E}_i^{\text{prune}} \to \mathcal{E}_i^{\text{prune}} \cup (i,j')$ for all $i \in [n]$.
  end while
end for
Set $\bar{\mathcal{E}}_i := \mathcal{E}_i$, for all $i \in [n]$.
for all all $i \in [n]$ **do**
  Send requests to all neighbors j such that $(i,j) \in \mathcal{E}_i^{\text{prune}}$ to prune edge $(j,i) \in \mathcal{E}_j$.
  Receive request from all neighbors j' such that $(j',i) \in \mathcal{E}_{j'}^{\text{prune}}$ to prune edge $(i,j') \in \mathcal{E}_i$.
  for all (i,j') such that $(i,j') \in \mathcal{E}_i^{\text{prune}}$ **do**
    Remove edge (i,j') from $\bar{\mathcal{E}}_i$.
  end for
  for all requests (i,j') such that $(i,j') \notin \mathcal{E}_i^{\text{prune}}$ **do**
    if $|\bar{\mathcal{E}}_i| > \lceil \underline{\kappa}_i |\mathcal{E}_i| \rceil$ **then**
      Remove edge (i,j') from $\bar{\mathcal{E}}_i$.
    end if
  end for
end for
if *Graph='Undirected'* **then**
  for all $(i,j) \in \bar{\mathcal{E}}_i$ and $(j,i) \notin \bar{\mathcal{E}}_j$ **do**
    Update set $\bar{\mathcal{E}}_j \to \bar{\mathcal{E}}_j \cup (j,i)$.
  end for
end if

Output: $\mathcal{G}(\mathcal{V}, \bar{\boldsymbol{\mathcal{E}}})$, where $\bar{\boldsymbol{\mathcal{E}}} := \cup_{i=1}^{n} \bar{\mathcal{E}}_i$.

$\underline{\kappa}_i \in [0,1]$ is a lower bound on the minimum number of edges retained at node i. The parameter $\beta \in [0,\infty]$ determines the level of influence of the dissimilarity measure in assigning the pruning probabilities. The role and significance of these parameters becomes evident by examining the main steps of the protocol, which we discuss next.

Selecting Candidate Edges for Pruning. To select the edges to be pruned, each node $i \in [n]$ constructs a set $\mathcal{E}_i^{\text{prune}}$ by iteratively drawing a sample edge from the set $\mathcal{E}_i \setminus \mathcal{E}_i^{\text{prune}}$, $\lfloor \bar{\kappa}_i \times |\mathcal{E}_i| \rfloor$ times, where $\bar{\kappa}_i$ represents the fraction of the total number of edges to be removed at node i during pruning. The probability of selecting an edge (i,j) is determined by the softmax of a dissimilarity measure (denoted by $\Delta(a_i,a_j)$) between the estimates at i and j. A possible candidate for $\Delta(a_i,a_j)$ is the ℓ_1-norm difference between a_i and a_j, i.e., $\|a_i - a_j\|_1$. For large values of the parameter β (the argument of the softmax) edges exhibiting small dissimilarity (small $\Delta(a_i,a_j)$), where a_i and a_j are in similar, have an increased likelihood of being pruned.

More formally, for the kth draw at node $i \in [n]$, where $1 \le k \le \lfloor \bar{\kappa}_i |\mathcal{E}_i| \rfloor$, the probability distribution over the set of edges $(i,j) \in \mathcal{E}_i \setminus \mathcal{E}_i^{\text{prune}}$ is given by

$$p_{i,j} \sim \frac{\exp(-\beta\Delta(a_i,a_j))}{\sum_{(i,j')\in\mathcal{E}_i/\mathcal{E}_i^{\text{prune}}} \exp(-\beta\Delta(a_i,a_{j'}))}, \qquad \text{for all } (i,j) \in \mathcal{E}_i \setminus \mathcal{E}_i^{\text{prune}},$$

where $\beta \in [0,\infty]$ is the softmax parameter that controls the influence of the dissimilarity measure. Note that $\beta = \infty$ represents the greedy case, where each node $i \in [n]$ selects the top $\lfloor \bar{\kappa}_i |\mathcal{E}_i| \rfloor$ edges with least dissimilarity measure. At the other extreme, $\beta = 0$ represents the case of random pruning independent of the dissimilarity measure. The

Pruning Mechanism. To perform the actual pruning, each node $i \in [n]$ sends a request to neighboring nodes j, where $(i, j) \in \mathcal{E}_i^{\text{prune}}$, to prune edge (j, i). At the same time, node $i \in [n]$ receives and catalogues the requests from all its neighboring nodes j' with $(j', i) \in \mathcal{E}_{j'}^{\text{prune}}$ to prune edges (i, j'). It is worth noting that the request for (i, j') does not necessarily require (i, j') to be in $\mathcal{E}_i^{\text{prune}}$. Initially, each node creates a copy $\bar{\mathcal{E}}_i$ of the original set of edges $\mathcal{E}_i$. The following steps are then performed in order by each node:

(i) For each (i, j') such that $(i, j') \in \mathcal{E}_i^{\text{prune}}$, edge (i, j') is removed from $\bar{\mathcal{E}}_i$. This covers the ideal case where both nodes i and j' want to remove the edge (i, j') and (j', i) from their respective edge sets $\mathcal{E}_i$ and $\mathcal{E}_{j'}$.

(ii) If $(i, j') \notin \mathcal{E}_i^{\text{prune}}$, then the edge is pruned if $|\bar{\mathcal{E}}_i| > \lceil \underline{\kappa}_i |\mathcal{E}_i| \rceil$. So, node $i \in [n]$ prunes an edge not included in $\mathcal{E}_i^{\text{prune}}$ only if the number of edges remaining in $\bar{\mathcal{E}}_i$ *is greater than a certain fraction* $\underline{\kappa}_i$ of $|\mathcal{E}_i|$. An implicit assumption here is that $\underline{\kappa}_i \leq 1 - \bar{\kappa}_i$ so that $\lceil \underline{\kappa}_i |\mathcal{E}_i| \rceil \leq \lceil (1 - \bar{\kappa}_i) |\mathcal{E}_i| \rceil$. It should be noted that for the algorithm to be well-defined, pruning requests of this type are processed in the order in which they are received.

The output of Algorithm 2.1 is $\mathcal{G}(\mathcal{V}, \bar{\boldsymbol{\mathcal{E}}})$, where $\bar{\boldsymbol{\mathcal{E}}} := \cup_i \bar{\mathcal{E}}_i$. An important point worth noting here is that the resulting set $\bar{\mathcal{E}}_i$ for $i \in [n]$ may contain edges (i, j) for which $(j, i) \notin \bar{\mathcal{E}}_j$. To make the pruned graph undirected, there are two possible approaches; either node j adds (j, i) to $\bar{\mathcal{E}}_j$, or alternatively, node i removes (i, j) from $\bar{\mathcal{E}}_i$. These approaches can be implemented by performing one additional round of communication among the nodes with negligible overhead.

2.3. Adaptive Consensus. Building upon the pruning protocol presented in the previous subsection, we introduce an algorithm to solve the consensus problem [37, Section 1], which requires the convergence of all the node estimates to the average of their initial estimates. The pseudo-code is provided in Algorithm 2.2.

Algorithm 2.2 `ADAPTIVE CONSENSUS (AC)`

Inputs: Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$; Cycle length $\tau \in \mathbb{N}$; Softmax parameter $\beta \in [0, \infty]$; Thresholding factors $(\bar{\kappa}_i, \underline{\kappa}_i) \in [0, 1]^2$ for all $i \in [n]$; Initial estimates $x_{i,0} \in \mathbb{R}^d$ for all $i \in [n]$; Total number of iterations $T \in \mathbb{N}$.

for $k = 0, \dots, T$ **do**
    for all $i \in [n]$ in parallel **do**
        if $k \in \mathcal{I}$, **then**
            Generate $\mathcal{G}(\mathcal{V}, \bar{\boldsymbol{\mathcal{E}}}_{k|\tau}) \sim$ `PRUNING PROTOCOL`$(\mathcal{G}(\mathcal{V}, \mathcal{E}), x_{i,k}, (\bar{\kappa}_i, \underline{\kappa}_i), \beta)$.
            Get new weights $\bar{q}_{ij}[k|\tau] \sim$ `GENERATE WEIGHTS` $(\mathcal{G}(\mathcal{V}, \bar{\boldsymbol{\mathcal{E}}}_{k|\tau}))$.
        end if
        Update estimate at node i according to: $x_{i,k+1} = \sum_{j=1}^{n} \bar{q}_{ij}[k|\tau] x_{j,k}$.
    end for
end for

Output: $x_{i,T}$ for all $i \in [n]$.

We discuss the main steps of the algorithm and how to select the parameters $\bar{\kappa}_i$ and $\underline{\kappa}_i$. Algorithm 2.2 has a cyclic structure with cycle length $\tau \in \mathbb{N}$. The set of indices where the pruning protocol is executed is denoted by $\mathcal{I} := [\tau, 2\tau, \dots)$. For any $k \in \mathcal{I}$, the iterations $t \in [k, k + \tau)$ are said to constitute a *consensus cycle*.

Pruning Step. At the start of the $k|\tau$ consensus cycle, the pruning protocol is executed to obtain the pruned graph $\mathcal{G}(\mathcal{V}, \bar{\boldsymbol{\mathcal{E}}}_{k|\tau})$, where $\bar{\boldsymbol{\mathcal{E}}}_{k|\tau} := \cup_i \bar{\mathcal{E}}_{i,k|\tau}$, using the current local estimates $x_{i,k}$ for all $i \in [n]$. Subsequently, the mixing matrix, denoted by $Q_{k|\tau} := [q_{ij}[k|\tau]]_{i\in[n],j\in[n]}$, of the pruned graph $\mathcal{G}(\mathcal{V}, \bar{\boldsymbol{\mathcal{E}}}_{k|\tau})$ is constructed in a decentralized manner. As an example, we can consider the Metropolis-Hastings scheme [34], which generates the weights via the following prescribed rule:

$$q_{ij}[k|\tau] := \begin{cases} \frac{1}{(1+\max\{|\bar{\mathcal{E}}_{i,k|\tau}|, |\bar{\mathcal{E}}_{j,k|\tau}|\})} & \text{if } (i,j) \in \bar{\boldsymbol{\mathcal{E}}}_{k|\tau} \\ 1 - \sum_{p=1}^{n} \bar{q}_{ip}[k|\tau] & \text{if } i = j \\ 0 & \text{otherwise,} \end{cases} \tag{2.1}$$

where $\bar{\mathcal{E}}_{i,k|\tau}$ denotes the (pruned) edge set at node $i \in [n]$.

Pruned Graph based Averaging. For all iterations $t \in [k, k+\tau)$ with $k \in \mathcal{I}$, the algorithm performs decentralized averaging using the pruned weights, $\bar{q}_{ij}[k|\tau]$. Subsequent to this, the pruning step (Line 4, Algorithm 2.2) is performed again with the updated node estimates.

Remark 2.1. *We make the following remarks about Algorithm 2.2.*

- *It is worth noting that the ideal choice of values for $\bar{\kappa}_i$ and $\underline{\kappa}_i$ can be problem-specific and depends on the network structure. For instance, preserving connectivity might be crucial in some cases, while in others, optimizing for low communication overhead may take precedence. Broadly speaking, a higher value of $\bar{\kappa}_i$ results in aggressive pruning more suited to graphs with high edge density. Conversely, $\underline{\kappa}_i$ acts as a lower bound on the edges to be retained post pruning, and a higher value of $\underline{\kappa}_i$ corresponds to a more conservative pruning approach, which is beneficial if maintaining connectivity is important. For β, lower values lead to increased randomness in edge selection, resembling approaches such as the gossip protocol [9], while higher values promote a more deterministic and greedy approach to edge selection.*
- *If directed edges are permitted in the output of the pruning protocol, the application of the push-sum protocol [25] offers an alternative to simple distributed averaging that alleviates the requirement for doubly stochastic mixing matrices.*

2.4. Convergence Analysis. To provide convergence guarantees, we begin by writing the key step of `AC` (Line 7, Algorithm 2.2) in matrix form by employing the stacked vector notation,

$$\mathbf{x}_{k+1} = \mathbf{Q}_k \mathbf{x}_k, \tag{2.2}$$

where $\mathbf{Q}_k = Q_k \otimes I_d = Q_{k|\tau} \otimes I_d \in \mathbb{R}^{nd\times nd}$, where $Q_{k|\tau} := [q_{ij}[k|\tau]]_{i\in[n],j\in[n]} \in \mathbb{R}^{n\times n}$ denotes the mixing matrix of the pruned graph $\mathcal{G}(\mathcal{V}, \bar{\boldsymbol{\mathcal{E}}}_{k|\tau})$ for the $k|\tau$ cycle. We use $\mathbf{Q}[r : s] \in \mathbb{R}^{nd\times nd}$ to denote the product of $s - r$ consecutive matrices indexed by $\{\mathbf{Q}_k\}_{k=r}^{s-1}$, i.e., $\mathbf{Q}[r : s] := \mathbf{Q}_{s-1} \times \cdots \times \mathbf{Q}_r$, with the convention that $Q[s : s] := I_n \otimes I_d \in \mathbb{R}^{nd\times nd}$. Using the above notation, we can express $\mathbf{x}_{(k+1)\tau}$ for any $k \geq 0$ in terms of $\mathbf{x}_0$ as follows

$$\begin{aligned} \mathbf{x}_{(k+1)\tau} = \mathbf{Q}_{k|\tau}^{\tau} \mathbf{x}_{k\tau} &= \mathbf{Q}[k\tau : (k+1)\tau] \mathbf{x}_{k\tau} \\ &= \mathbf{Q}[k\tau : (k+1)\tau] \times \cdots \times \mathbf{Q}[0 : \tau] \mathbf{x}_0. \end{aligned} \tag{2.3}$$

We establish convergence under the following assumption.

Assumption 2.2 ($\bar{\tau}$-*Connectivity*). *There exists a constant $\bar{\tau} \in \mathbb{N}$, such that for all $k \in \mathcal{I}_{\bar{\tau}} := \{\bar{\tau}, \bar{\tau}+\tau, \bar{\tau}+2\tau \cdots\} \subset \mathcal{I}$, the graph $\mathcal{G}(\mathcal{V}, \bar{\boldsymbol{\mathcal{E}}}_{(k|\tau-\bar{\tau}+1)}) \cup \cdots \cup \mathcal{G}(\mathcal{V}, \bar{\boldsymbol{\mathcal{E}}}_{k|\tau})$ is connected.*

Remark 2.2. *Assumption 2.2 plays a key role in the analysis. In words, it implies the existence of a constant $\bar{\tau}$, such that within $\bar{\tau}$ pruning cycles, the union of the resulting undirected (directed) pruned graphs is connected (strongly connected). For the special case where the pruned graph is connected for all cycles, $\bar{\tau} = 1$. It is possible to guarantee this assumption by imposing a consensus iteration with the reference graph every $\bar{\tau}$ iterations of the algorithm for some finite $\bar{\tau} \in \mathbb{N}$. Additionally, it is worth noting that it suffices to assume this property only for indices $\mathcal{I}_{\bar{\tau}}$ rather than for all $k \in \mathbb{N}$. Another important point to note is that the assumption can be replaced by a stochastic version which takes into account the utilization of softmax based sampling in the pruning protocol. Specifically, the assumption of connectedness can either be assumed to hold almost surely or replaced by an assumption that ensures a reduction in the consensus error in expectation (with respect to $\mathbf{Q}_k$) over a period of $\bar{\tau}$ iterations.*

Remark 2.3. *We note that Assumption 2.2 is equivalent to assuming that every edge $(i, j) \in \mathcal{E}$ gets activated every $\bar{\tau}$ iterations for some finite $\bar{\tau} > 0$. Let $\mathcal{A}_k$ denote a random subset of the edge set $\mathcal{E}$, composed of the subset of edges updated at time k, and let $\nu_{(i,j),k} := \sum_{m=0}^{k} \mathcal{I}\{(i, j) \in \mathcal{A}_m\}$, where $\mathcal{I}(\cdot)$ denotes the indicator function, representing the number of times (i, j) is activated up until time k. Assumption 2.2 can be satisfied if the following condition holds:*

$$\liminf_{k\to\infty} \frac{\nu_{(i,j),k}}{k} > 0 \;\; \forall \;\; (i, j) \in \mathcal{E}. \tag{2.4}$$

That is, all edges are updated comparably often. To ensure this, we can mix the softmax policy with a uniformly random policy with an arbitrarily small θ. More formally, for node $i \in [n]$ the probability distribution over the set of edges (i, j) can be written as

$$p_{i,j} \sim (1-\theta) \frac{\exp(-\beta\Delta(a_i, a_j))}{\sum_{(i,j')\in\mathcal{E}_i/\mathcal{E}_i^{prune}} \exp(-\beta\Delta(a_i, a_{j'}))} + \frac{\theta}{|\mathcal{E}_i \setminus \mathcal{E}_i^{prune}|}, \qquad \forall \;\; (i, j) \in \mathcal{E}_i \setminus \mathcal{E}_i^{prune},$$

where $\theta > 0$ is an arbitrarily small parameter. Since $p_{i,j} \geq \frac{\theta}{|\mathcal{E}_i \setminus \mathcal{E}_i^{prune}|} > 0$ for any edge (i, j), using Borel-Cantelli Lemma, we have that each edge (i, j) is activated infinitely often.

To prove convergence of the algorithm, we need to establish convergence of the following product sequence to the $\frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T$ rank-one matrix, i.e.,

$$\prod_{j=0}^{k} \mathbf{Q}[j\tau : (j+1)\tau] \to \tfrac{1}{n}\mathbf{1}_n\mathbf{1}_n^T, \qquad \text{as } k \to \infty.$$

To show this, we use the notion of coefficient of ergodicity [47], denoted by $\rho(Q)$ for any row-stochastic matrix Q, defined as,

$$\rho(Q) := 1 - \min_{i_1, i_2} \sum_{j=1}^{n} \min\left(q_{i_1 j}, q_{i_2 j}\right). \tag{2.5}$$

Using the coefficient of ergodicity instead of directly bounding the spectral gap offers several advantages, particularly in scenarios involving time-varying topologies. First, it allows us to clearly characterize the influence of different graph parameters, such as maximum node degree and diameter, on convergence. This characterization helps us establish an explicit relationship between pruning and convergence. Second, it allows for extensions to directed graphs (with push-sum protocols) where the condition of double stochasticity may not be satisfied.

There are two key properties of (2.5) that will be useful in establishing convergence. The first property is that $\rho(\cdot)$ is sub-multiplicative, i.e., for any two matrices Q_1, Q_2,

$$\rho(Q_1Q_2) \leq \rho(Q_1)\rho(Q_2). \tag{2.6}$$

The second property is that it can serve as an upper bound on the dissimilarity between the rows of matrix Q. More formally, we have (cf. [55, Lemma 2], [20, Lemma 4])

$$\delta(Q) := \max_{j} \max_{i_1,i_2} |q_{i_1 j} - q_{i_2 j}| \leq \rho(Q), \tag{2.7}$$

for any matrix Q which is ergodic, i.e., it is row stochastic, aperiodic and irreducible (cf. [55] or, [24, Chapter 8]).

Next, we state and prove the main theoretical result of this section.

Theorem 2.1. *Suppose that: (i) Assumptions 2.1 and 2.2 hold, (ii) the matrices $Q_k := [q_{ij}[k]]_{i\in[n],j\in[n]}$ are doubly stochastic for all $k \geq 0$, (iii) $q_{ii}[k] > 0$ for all $k \geq 0$ for at least one $i \in [n]$, and, (iv) if $q_{ij}[k] > 0$ for any $(i,j) \in \mathcal{E}$ and $k \geq 0$, then $q_{ij}[k] > q$ for some strictly positive constant $q > 0$ independent of k and (i,j). Then, for any $k \geq 0$,*

$$\|\mathbf{x}_k - \bar{\mathbf{x}}_k\| \leq \min\left\{ n^{\frac{3}{2}} \gamma^{\left\lfloor \frac{k}{\bar{\tau} d_{\mathcal{G}}} \right\rfloor}, n\left(1 - \frac{q}{2n^2}\right)^{\frac{k}{\bar{\tau}}} \right\} \|\mathbf{x}_0 - \bar{\mathbf{x}}_0\|, \tag{2.8}$$

where $\gamma := \left(1 - q^{\bar{\tau} d_{\mathcal{G}}}\right) < 1$ with $q < 1$ and $d_{\mathcal{G}}$ is the diameter of a graph $\mathcal{G}(\mathcal{V},\mathcal{E})$ defined as $d_{\mathcal{G}} := \max_{u,v\in\mathcal{V}}\{dist(u,v)\}$, where $dist(u,v)$ denotes the shortest path distance between any two vertices u, $v \in \mathcal{V}$.

Proof. We first establish the ergodicity of the product sequence $\mathbf{Q}[m\bar{\tau} : (m+1)\bar{\tau}]$ for any $m \geq 0$ with $\bar{\tau} \in \mathbb{N}$ as in Assumption 2.2. The stochasticity of $\mathbf{Q}[m\bar{\tau} : (m+1)\bar{\tau}]$ follows from that the fact that the product of stochastic matrices is also stochastic. Furthermore, a matrix is considered irreducible if its zero/non-zero structure corresponds to a connected graph. By Assumption 2.2, the structure of $\mathbf{Q}[m\bar{\tau} : (m+1)\bar{\tau}]$ also exhibits this property [19, Section 1-C]. Finally, an irreducible matrix is aperiodic if it has at least one self-loop which is satisfied by $\mathbf{Q}[m\bar{\tau} : (m+1)\bar{\tau}]$ by condition (ii) in the theorem statement [19, Section 1-C].

Next, we establish a useful upper bound on $\delta\left(\mathbf{Q}[0 : k+1]\right)$. To do this, we consider the following decomposition of $\mathbf{Q}[0 : k+1]$

$$\begin{aligned} &\mathbf{Q}[0 : k+1] \\ &= \underbrace{\mathbf{Q}[0 : \bar{k}\bar{\tau}] \times \cdots \times \mathbf{Q}[m\bar{k}\bar{\tau} : (m+1)\bar{k}\bar{\tau}] \times \cdots \times \mathbf{Q}[(K-1)\bar{k}\bar{\tau} : K\bar{k}\bar{\tau}]}_{\mathbf{Q}[0:K\bar{k}\bar{\tau}]} \times \mathbf{Q}[K\bar{k}\bar{\tau} : k+1] \end{aligned}$$

where $K := \lfloor k/\bar{k}\bar{\tau} \rfloor$, $\bar{k} \geq 1$ is a constant to be specified later. Let $\tau' := \bar{k}\bar{\tau}$. We bound $\delta\left(\mathbf{Q}[0 : K\tau']\right)$ by individually bounding $\rho\left(\mathbf{Q}[m\tau' : (m+1)\tau']\right)$ in the above product.

By (2.5), it follows that,

$$\rho\left(\mathbf{Q}[m\tau' : (m+1)\tau']\right) = 1 - \min_{i_1, i_2} \sum_j \min\left(q_{i_1 j}[m\tau' : (m+1)\tau'], q_{i_2 j}[m\tau' : (m+1)\tau']\right), \tag{2.9}$$

where $\mathbf{Q}[m\tau' : (m+1)\tau'] := [q_{ij}[m\tau' : (m+1)\tau']]_{i,j\in[n]}$. By (2.9), we note that $\rho\left(\mathbf{Q}[m\tau' : (m+1)\tau']\right)$ is guaranteed to satisfy $\rho\left(\mathbf{Q}[m\tau' : (m+1)\tau']\right) < 1$, if for every pair of rows i_1 and i_2, there exists some j^* such that $q_{i_1 j^*}[m\tau' : (m+1)\tau']$, $q_{i_2 j^*}[m\tau' : (m+1)\tau'] > 0$, i.e., if there is a path from some j^* to both i_1 and i_2. This, in turn, is always satisfied if for some $\bar{k} > 0$, $q_{ij}[m\tau' : (m+1)\tau'] > 0$ for every i, $j \in [n]$, i.e., all the entries are strictly positive.

To find such a candidate $\bar{k}$, we make the following observation: $\mathbf{Q}[m\bar{\tau} : (m+1)\bar{\tau}]$ is ergodic, so there exists a path from i to j for every i, $j \in [n]$. Setting $\bar{k} = d_{\mathcal{G}}$ in the definition of τ', we have $\tau' = \bar{k}\bar{\tau} = d_{\mathcal{G}}\bar{\tau}$. It follows that for the matrix $\mathbf{Q}[m\tau' : (m+1)\tau']$, $q_{ij}[m\tau' : (m+1)\tau'] > 0$ for all i, $j \in [n]$ since we can reach any node i from any other node j in at most $\tau' = d_{\mathcal{G}}\bar{\tau}$ steps.

For the remainder of the proof, let $\tau' = d_{\mathcal{G}}\bar{\tau}$. To lower bound $q_{ij}[m\tau' : (m+1)\tau'] > 0$, $m \geq 0$, we note that by the definition of q and Assumption 2.2, it follows that $q_{ij}[p\bar{\tau} : (p+1)\bar{\tau}] \geq q^{\bar{\tau}}$ for any $p \geq 0$ and any $(i,j) \in \mathcal{E}_{p\bar{\tau}} \cup \cdots \cup \mathcal{E}_{(p+1)\bar{\tau}-1}$. Since $\mathbf{Q}[m\tau' : (m+1)\tau'] = \mathbf{Q}[m\tau' : m\tau' + \bar{\tau}] \cdots \mathbf{Q}[m\tau' + (d_{\mathcal{G}} - 1)\bar{\tau} : m\tau' + d_{\mathcal{G}}\bar{\tau}]$, for any $i', j' \in [n]$,

$$q_{i'j'}[m\tau' : (m+1)\tau'] \geq q^{\bar{\tau} d_{\mathcal{G}}}. \tag{2.10}$$

By (2.9) and (2.10),

$$\rho\left(\mathbf{Q}[m\tau' : (m+1)\tau']\right) \leq 1 - q^{\bar{\tau} d_{\mathcal{G}}}. \tag{2.11}$$

Thus, it follows that,

$$\begin{aligned}
\delta(\mathbf{Q}[(0 : K\bar{\tau}]) &\leq \rho\left(\mathbf{Q}[0 : K\bar{\tau}]\right) \\
&\leq \rho(\mathbf{Q}[0 : \bar{\tau}] \cdots \mathbf{Q}[(K-1)\bar{\tau} : K\bar{\tau}]) \\
&\leq \rho(\mathbf{Q}[0 : \bar{\tau}]) \times \cdots \times \rho(\mathbf{Q}[(K-1)\bar{\tau} : K\bar{\tau}]) \\
&\leq \left(1 - q^{\bar{\tau} d_{\mathcal{G}}}\right)^K, \qquad (2.12)
\end{aligned}$$

where the first inequality follows by (2.7), the second inequality by the the sub-multiplicative property of $\rho(\cdot)$ (2.6), and the final inequality follows by (2.11). By (2.2) and (2.3), it follows that,

$$\begin{aligned}
\mathbf{x}_k &= \mathbf{Q}_{k-1}\mathbf{x}_{k-1} \\
&= \mathbf{Q}[K\bar{\tau} + 1 : k]\mathbf{Q}[(K-1)\bar{\tau} : K\bar{\tau}] \times \cdots \times \mathbf{Q}[0 : \bar{\tau}]\mathbf{x}_0 \\
&= \mathbf{Q}[K\bar{\tau} + 1 : k]\mathbf{Q}[0 : K\bar{\tau}]\mathbf{x}_0. \qquad (2.13)
\end{aligned}$$

Multiplying both sides of (2.13) by $\mathbf{1}_n$, by the the double stochasticity of $\mathbf{Q}[K\bar{\tau}+1 : k]$ and $\mathbf{Q}[0 : K\bar{\tau}]$, it follows that,

$$\bar{\mathbf{x}}_k = \bar{\mathbf{x}}_0 = \tfrac{1}{n}\mathbf{1}_n\mathbf{1}_n^k\mathbf{x}_0 = \tfrac{1}{n}\mathbf{1}_n\mathbf{1}_n^k\mathbf{Q}[0 : K\bar{\tau}]\mathbf{x}_0. \tag{2.14}$$

Subtracting (2.14) from (2.13),

$$\begin{aligned}
\mathbf{x}_k - \bar{\mathbf{x}}_k &= \mathbf{Q}[K\bar{\tau} + 1 : k]\mathbf{Q}[0 : K\bar{\tau}]\mathbf{x}_0 - \tfrac{1}{n}\mathbf{1}_n\mathbf{1}_n^k\mathbf{Q}[0 : K\bar{\tau}]\mathbf{x}_0 \\
&= \mathbf{Q}[K\bar{\tau} + 1 : k]\left(\mathbf{Q}[0 : K\bar{\tau}] - \tfrac{1}{n}\mathbf{1}_n\mathbf{1}_n^k\mathbf{Q}[0 : K\bar{\tau}]\right)(\mathbf{x}_0 - \bar{\mathbf{x}}_0),
\end{aligned}$$

where the second equality holds due to $\mathbf{Q}[K\bar{\tau}+1:k]\mathbf{1}=\mathbf{1}$ and the fact that $\mathbf{A}\bar{\mathbf{x}}_0=\bar{\mathbf{x}}_0$ for any doubly stochastic matrix $\mathbf{A}$. Taking norms of the above, it follows that,

$$\|\mathbf{x}_k-\bar{\mathbf{x}}_k\| \le \|\mathbf{Q}[K\bar{\tau}+1:k]\| \left\|\mathbf{Q}[0:K\bar{\tau}]-\tfrac{1}{n}\mathbf{1}_n\mathbf{1}_n^k\mathbf{Q}[0:K\bar{\tau}]\right\| \|\mathbf{x}_0-\bar{\mathbf{x}}_0\|$$

$$\le \sqrt{n}\left\|\mathbf{Q}[0:K\bar{\tau}]-\tfrac{1}{n}\mathbf{1}_n\mathbf{1}_n^k\mathbf{Q}[0:K\bar{\tau}]\right\|_1 \|\mathbf{x}_0-\bar{\mathbf{x}}_0\|_1 \tag{2.15}$$

where the first inequality is due to the Cauchy–Schwarz inequality and the second inequality follows due to the facts that $\|A\| \le \sqrt{n}\|A\|_1$ for any $A \in \mathbb{R}^{n\times n}$ and $\|\mathbf{Q}[K\bar{\tau}+1:k]\| \le 1$. We have by definition of the ℓ_1-norm for matrices,

$$\left\|\mathbf{Q}[0:K\bar{\tau}]-\tfrac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\mathbf{Q}[0:K\bar{\tau}]\right\|_1 = \max_{1\le j\le n}\sum_{i=1}^n \left| q_{ij}[0:K\bar{\tau}]-\frac{1}{n}\sum_{i'=1}^n q_{i'j}[0:K\bar{\tau}]\right|$$

$$\le \max_{1\le j\le n}\sum_{i=1}^n \frac{1}{n}\sum_{i'=1}^n \underbrace{\left|q_{ij}[0:K\bar{\tau}]-q_{i'j}[0:K\bar{\tau}]\right|}_{\le \delta(\bar{\mathbf{Q}}[0:K\bar{\tau}])}$$

$$\le n\delta(\bar{\mathbf{Q}}[0:K\bar{\tau}]) \tag{2.16}$$

$$\le n\left(1-q^{d_{\mathcal{G}}\bar{\tau}}\right)^K, \tag{2.17}$$

where the last inequality follows by (2.12).

We can also bound $\left\|\mathbf{Q}[0:K\bar{\tau}]-\frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right\|_1$ using [35, Theorem 3.1], as

$$\left\|\mathbf{Q}[0:K\bar{\tau}]-\tfrac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right\|_1 \le \sqrt{n}\left(1-\frac{q}{2n^2}\right)^{\frac{1}{\bar{\tau}}}. \tag{2.18}$$

Combining (2.15), (2.17) and (2.18) with $K=\lfloor k/\bar{k}\bar{\tau}\rfloor$ gives the required bound. ◻

We note that the convergence rate in Theorem 2.1 is primarily dependent of the diameter of the graph, $d_{\mathcal{G}}$, and the lower bound on the nonzero entries of the mixing matrix, q. The form of the convergence rate factor γ confirms the empirical observation that compact graphs with shorter diameters generally fare better with pruning since multiple information pathways can potentially exist between two nodes. The dependence on q can be illustrated by considering the Metropolis-Hastings scheme as described in (2.1). Let $n_{\mathcal{G}_{k|\tau}}$ denote the maximum node degree of graph $\mathcal{G}(\mathcal{V},\boldsymbol{\mathcal{E}}_{k|\tau})$. If $n_{\max} := \max_{k\in\mathcal{I}} n_{\mathcal{G}_{k|\tau}}$, denotes the maximum node degree amongst all the pruned graphs (assumed to be connected) obtained during the algorithm, then $q=\frac{1}{1+n_{\max}}$. Since $n_{\max}$ can be smaller than the maximum node degree of the underlying reference graph, q can potentially be larger for `AC`.

Another point to note here is that either term on the right-hand side of the minimum in (2.8) can be active. For graphs, where $d_{\mathcal{G}}$ is large, the second term is active. Conversely, in the case of small $d_{\mathcal{G}}$ and large n, the first term is active. As a concrete example of the latter case, we can consider a dumbbell graph with $d_{\mathcal{G}}=3$, $n \gg 1$, and $\bar{\tau}=1$. The first term in (2.8) will be active provided, $\left(1-q^3\right)^{\lfloor \frac{t}{3}\rfloor} \le \left(1-\frac{q}{2n^2}\right)^t$, $t\ge 1$.

REMARK 2.4. *We make the following additional remarks about Theorem 2.1.*

- *It should be noted that the convergence factor γ in (2.8) may be a conservative estimate in general. Nevertheless, the analysis provided here remains applicable in a broad range of scenarios, even when tighter estimates for specific cases may not hold. In particular, the extension of Theorem 2.1 to a directed graph setting,* *where only column stochasticity is satisfied (as in the push-sum protocol), can be derived relatively easily. This is due to the fact that the definition of the coefficient of ergodicity and the associated bounds, e.g.,* (2.7)*, do not necessitate a double-stochasticity assumption on the matrix* Q_k*.*

- *The assumptions on the matrix entries of* Q_k *in Theorem 2.1 are typical in ergodic matrix literature [24] and multi-agent coordination and optimization problems [35]. For undirected graphs, the assumptions are satisfied if the weights are generated according to* (2.1)*.*
- *To understand (and quantify) the impact of pruning on distributed averaging within a simplified context, let us consider a scenario where there is a total communication budget of* B *bits, and each node utilizes* D *bits to transmit the quantized objective variable to its neighboring nodes. The maximum number of iterations that can be executed under these settings is given by* $T = \frac{B}{2D|\mathcal{E}|}$*. Let* $\sigma(Q)$ *denote the spectral gap of the mixing matrix* Q*, assumed to be generated in accordance to* (2.1)*. Under Assumption 2.1, for* x_k *generated via* (2.2) *with* $\mathbf{Q}_k = Q \otimes I_d$, $\forall\, k$,

$$\|\mathbf{x}_T - \bar{\mathbf{x}}_T\| \leq (1-\sigma(Q))^T \|\mathbf{x}_0 - \bar{\mathbf{x}}_0\|. \tag{2.19}$$

If we consider the same scenario with a fraction $\kappa < 1$ *of the edges pruned (where the pruned mixing matrix is denoted by* Q^{prune}*) and assume the pruned graph satisfies Assumption 2.1, we have*[2],

$$\|\mathbf{x}_{T^{prune}} - \bar{\mathbf{x}}_{T^{prune}}\| \leq (1-\sigma(Q^{prune}))^{T^{prune}} \|\mathbf{x}_0 - \bar{\mathbf{x}}_0\|. \tag{2.20}$$

Since $T^{prune} = \frac{B}{2(1-\kappa)D|\mathcal{E}|} = \frac{T}{1-\kappa} > T$*, the upper bound for the consensus error with the pruned network, where* $\sigma(Q^{prune}) \approx \sigma(Q)$*, is potentially tighter since* $(1-\sigma(Q^{prune}))^{T^{prune}} \lessapprox (1-\sigma(Q))^T$*. In Section 4.1 (Figure 1(c)), we empirically observe that* $\sigma(Q^{prune})$ *for small to medium values of* κ *does not significantly deviate from* $\sigma(Q)$*, suggesting that there are instances for which the inequality is likely to hold.*

3. Adaptive Consensus based Decentralized Optimization. In this section, we describe the proposed Adaptive Consensus based Gradient Tracking algorithm (Algorithm 3.1, `AC-GT`) for decentralized optimization. The problem under consideration can be expressed as,

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^{nd}} \quad & f(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^{n} f_i(x_i) \\ \text{s.t.} \quad & \mathbf{Q}\mathbf{x} = \mathbf{x}, \end{aligned} \tag{3.1}$$

where $f : \mathbb{R}^{nd} \to \mathbb{R}$ and $\mathbf{Q} := Q \otimes I_d \in \mathbb{R}^{nd\times nd}$. Under Assumption 2.1, the constraint is equivalent to the condition that $x_i = x_j$, for all $i, j \in [n]$, and thus problems (3.1) and (1.1) are equivalent. We make the following assumption with regards to the component functions (f_i).

Assumption 3.1 (Regularity and convexity of f_i). *Each* f_i *is* L*-smooth and* μ*-strongly convex.*

[2] To keep the presentation clear, we assume $T, T^{\text{prune}} \in \mathbb{N}$.

The general idea of AC-GT is to leverage the adaptive consensus protocol of the previous section and combine it with a gradient tracking algorithm [34] in a manner that preserves the strong convergence guarantees of the latter while harnessing the communication savings of the former. The pseudo-code for the algorithm is provided in Algorithm 3.1.

Algorithm 3.1 ADAPTIVE CONSENSUS BASED GRADIENT TRACKING (AC-GT)

Inputs: Graph $\mathcal{G}(\mathcal{V},\mathcal{E})$; Cycle Length $\tau \in \mathbb{N}$; Softmax parameter $\beta \in [0,\infty]$; Thresholding factors $(\bar{\kappa}_i, \underline{\kappa}_i) \in [0,1]^2$ for all $i \in [n]$; Step size $\alpha > 0$; Initial iterates $x_{i,0} \in \mathbb{R}^d$, $y_{i,0} = \nabla f_i(x_{i,0})$ for all $i \in [n]$; Total number of iterations $T \in \mathbb{N}$.

- **for** $k = 0, \ldots, T$ **do**
  - **for** all $i \in [n]$ in parallel **do**
    - **if** $k \in \mathcal{I}$, **then**
      - Generate $\mathcal{G}(\mathcal{V}, \bar{\boldsymbol{\mathcal{E}}}_{k|\tau}) \sim$ PRUNING PROTOCOL($\mathcal{G}(\mathcal{V},\mathcal{E}), x_{i,k}, (\bar{\kappa}_i, \underline{\kappa}_i), \beta$).
      - Get new weights $\bar{q}_{ij}[k|\tau] \sim$ GENERATE WEIGHTS ($\mathcal{G}(\mathcal{V}, \bar{\boldsymbol{\mathcal{E}}}_{k|\tau})$).
      - Generate $\mathcal{G}(\mathcal{V}, \hat{\boldsymbol{\mathcal{E}}}_{k|\tau}) \sim$ PRUNING PROTOCOL($\mathcal{G}(\mathcal{V},\mathcal{E}), y_{i,k}, (\bar{\kappa}_i, \underline{\kappa}_i), \beta$).
      - Get new weights $\hat{q}_{ij}[k|\tau] \sim$ GENERATE WEIGHTS ($\mathcal{G}(\mathcal{V}, \hat{\boldsymbol{\mathcal{E}}}_{k|\tau})$).
    - **end if**
    - Update estimate at node i according to: $x_{i,k+1} = \sum_{j=1}^n \bar{q}_{ij}[k|\tau] \left(x_{j,k} - \alpha y_{j,k}\right)$.
    - Update gradient estimate at node i according to: $y_{i,k+1} = \sum_{j=1}^n \hat{q}_{ij}[k|\tau] y_{j,k} + \nabla f_i(x_{i,k+1}) - \nabla f_i(x_{i,k})$.
  - **end for**
- **end for**

Output: $x_{i,T}$ for all $i \in [n]$.

To provide intuition for the algorithm, we review the main steps of the gradient tracking algorithm (GTA), as it serves as a foundational component of AC-GT. The main iterations of the gradient tracking algorithm can be expressed as,

$$x_{i,k+1} = \sum_{j=1}^{n} q_{ij}\left(x_{j,k} - \alpha y_{j,k}\right), \qquad y_{i,k+1} = \sum_{j=1}^{n} q_{ij} y_{j,k} + \nabla f_i(x_{i,k+1}) - \nabla f_i(x_{i,k}),$$

where $\alpha > 0$ is a constant referred to as the step size.

The underlying computational principles of AC-GT are similar to those of GTA. However, the communication structure of AC-GT is based on AC. Similar to AC, AC-GT operates in a cyclical manner. In the $k|\tau$ cycle, if k belongs to the set $\mathcal{I}$, the pruning protocol is executed twice. The first instance employs the $\mathbf{x}$ estimates to get the pruned graph (Q_k) and the associated mixing matrix, which are subsequently utilized to update the $\mathbf{x}$ estimate,

$$\mathbf{x}_{k+1} = \mathbf{Q}_k\left(\mathbf{x}_k - \alpha \mathbf{y}_k\right), \text{ where } \mathbf{Q}_k = \mathbf{Q}_{k|\tau}, \ \forall k \in [(k|\tau)\tau, \, (k|\tau+1)\tau). \tag{3.2}$$

The second instance of the protocol obtains a different pruned graph ($\hat{Q}_k$) using the $\mathbf{y}$ estimates. The mixing matrix corresponding to this graph is then used to update the $\mathbf{y}$ estimate as follows,

$$\mathbf{y}_{k+1} = \hat{\mathbf{Q}}_k \mathbf{y}_k + \nabla \mathbf{f}(\mathbf{x}_{k+1}) - \nabla \mathbf{f}(\mathbf{x}_k), \text{ where } \hat{\mathbf{Q}}_k = \hat{\mathbf{Q}}_{k|\tau}, \ \forall k \in [(k|\tau)\tau, \, (k|\tau+1)\tau). \tag{3.3}$$

The pruning protocol is executed twice because the dissimilarity between the $\mathbf{y}$ estimates is expected to be different from the dissimilarity between the $\mathbf{x}$ estimates. AC-GT employs a constant step size $\alpha > 0$ which depends on both the properties of the function and the structure of the pruned network as shown in the next subsection.

Remark 3.1. *We make a couple of remarks about* `AC-GT` *(Algorithm 3.1).*

- *For $\tau = 1$ and $\mathbf{Q}_k = Q \otimes I_d$ for all $k \in \mathbb{N}$, where Q is the mixing matrix corresponding to the reference graph,* `AC-GT` *reduces to a standard gradient tracking algorithm (*`DIGing`*) [34].*
- *The extension of* `AC-GT` *to a directed graph setting is feasible by leveraging the push-pull gradient algorithm (*`Push-DIGing`*) [39]. Similar to* `AC`*, the principles and theory of* `AC-GT` *for the directed graph setting can be derived from the current framework, with appropriate adjustments.*

3.1. Convergence Analysis. We provide theoretical convergence guarantees for `AC-GT`. For simplicity, we assume that $\mathbf{Q}_k = \hat{\mathbf{Q}}_k$ for all $k \geq 0$ in (3.2) and (3.3) and note that one can derive the same results verbatim for the case where $\mathbf{Q}_k \neq \hat{\mathbf{Q}}_k$, with additional notation required. We build up to our main result through a series of technical lemmas which we state next. We begin by proving a descent relation for the consensus error Ψ_k, defined as,

$$\Psi_k := \begin{bmatrix} \mathbf{x}_k - \bar{\mathbf{x}}_k \\ \alpha(\mathbf{y}_k - \bar{\mathbf{y}}_k) \end{bmatrix} \in \mathbb{R}^{2nd}. \tag{3.4}$$

Lemma 3.1. *Suppose that the matrices $\mathbf{Q}_k$, for all k, are doubly stochastic and $\hat{\mathbf{Q}}_k = \mathbf{Q}_k$. For Ψ_k given in* (3.4) *and $\hat{\tau} \in \mathbb{N}$,*

$$\|\Psi_k\|^2 \leq \rho' \|\Psi_{k-\hat{\tau}}\|^2 + b \sum_{j=k-\hat{\tau}}^{k-1} \|\Psi_j\|^2 + c \sum_{j=k-\hat{\tau}}^{k-1} \left(f(\bar{x}_j) - f(x^*) \right), \quad \text{if} \quad k \geq \hat{\tau}, \tag{3.5}$$

$$\|\Psi_k\|^2 \leq 5(1+\hat{\tau}^2) \|\Psi_0\|^2 + b \sum_{j=0}^{k-1} \|\Psi_j\|^2 + c \sum_{j=0}^{k-1} \left(f(\bar{x}_j) - f(x^*) \right), \quad \text{if} \quad 0 < k < \hat{\tau}, \tag{3.6}$$

where $\rho' := 2(1+\hat{\tau}^2) \max_{\hat{\tau} \leq j \leq t} \left\| \mathbf{Q}[j-\hat{\tau} : j] - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^T \right\|^2$, $b := 180\alpha^2 L^2 (1+\hat{\tau}^2)\hat{\tau}$, and $c := 320 n \alpha^4 L^3 (1+\hat{\tau}^2)\hat{\tau}$.

Proof. We start by considering the expression $\mathbf{x}_k - \bar{\mathbf{x}}_k$. By (3.2) and the double stochasticity of $\mathbf{Q}_k$,

$$\mathbf{x}_k - \bar{\mathbf{x}}_k = \left(\mathbf{Q}_{k-1} - \tfrac{\mathbf{1}_n \mathbf{1}_n^T}{n} \right) \left(\mathbf{x}_{k-1} - \bar{\mathbf{x}}_{k-1} - \alpha(\mathbf{y}_{k-1} - \bar{\mathbf{y}}_{k-1}) \right). \tag{3.7}$$

Using (3.3), a similar expression for $\mathbf{y}_k - \bar{\mathbf{y}}_k$ is given as,

$$\mathbf{y}_k - \bar{\mathbf{y}}_k = \left(\mathbf{Q}_{k-1} - \tfrac{\mathbf{1}_n \mathbf{1}_n^T}{n} \right) \left(\mathbf{y}_{k-1} - \bar{\mathbf{y}}_{k-1} \right) - \left(\mathbf{I}_n - \tfrac{\mathbf{1}_n \mathbf{1}_n^T}{n} \right) \left(\nabla \mathbf{f}(\mathbf{x}_k) - \nabla \mathbf{f}(\mathbf{x}_{k-1}) \right), \tag{3.8}$$

where $\mathbf{I}_n := I_n \otimes I_d \in \mathbb{R}^{nd \times nd}$. The expressions in (3.7) and (3.8) can be compactly represented in matrix form as follows,

$$\begin{aligned} \Psi_k &= \mathbf{J}_{k-1} \Psi_{k-1} + \alpha \mathbf{E}_{k-1} \\ &= \mathbf{J}[k-\hat{\tau} : k] \Psi_{k-\hat{\tau}} + \alpha \sum_{j=1}^{\hat{\tau}} \mathbf{J}[k-j+1 : k] \mathbf{E}_{k-j}, \end{aligned} \tag{3.9}$$

where

(3.10)
$$\mathbf{J}_k := \begin{bmatrix} \mathbf{Q}_k - \frac{\mathbf{1}_n\mathbf{1}_n^T}{n} & -\left(\mathbf{Q}_k - \frac{\mathbf{1}_n\mathbf{1}_n^T}{n}\right) \\ 0 & \mathbf{Q}_k - \frac{\mathbf{1}_n\mathbf{1}_n^T}{n} \end{bmatrix}, \mathbf{E}_{k-1} := \begin{bmatrix} 0 \\ \left(\mathbf{I}_n - \frac{\mathbf{1}_n\mathbf{1}_n^T}{n}\right)(\nabla\mathbf{f}(x_{k-1}) - \nabla\mathbf{f}(x_k)) \end{bmatrix}$$

and $\mathbf{J}[k-j:k] := \mathbf{J}_{k-1}\cdots\mathbf{J}_{k-j}$, for any $j \leq \hat{\tau} \leq k$. The matrix $\mathbf{J}[k-j:k]$ can be expressed as,

$$\text{(3.11)} \qquad \mathbf{J}[k-j:k] = \begin{bmatrix} \mathbf{Q}[k-j:k] - \frac{\mathbf{1}_n\mathbf{1}_n^T}{n} & -j\left(\mathbf{Q}[k-j:k] - \frac{\mathbf{1}_n\mathbf{1}_n^T}{n}\right) \\ 0 & \mathbf{Q}[k-j:k] - \frac{\mathbf{1}_n\mathbf{1}_n^T}{n} \end{bmatrix}.$$

The above equation can be derived by a straightforward induction argument using the facts that

$$\begin{bmatrix} A_1 & -A_1 \\ 0 & A_1 \end{bmatrix} \times \begin{bmatrix} A_2 & -A_2 \\ 0 & A_2 \end{bmatrix} = \begin{bmatrix} A_1A_2 & -2A_1A_2 \\ 0 & A_1A_2 \end{bmatrix},$$

and, for any two doubly stochastic matrices $\mathbf{Q}$ and $\mathbf{Q}'$,

$$\left(\mathbf{Q} - \frac{\mathbf{1}_n\mathbf{1}_n^T}{n}\right)\left(\mathbf{Q}' - \frac{\mathbf{1}_n\mathbf{1}_n^T}{n}\right) = \left(\mathbf{Q}\mathbf{Q}' - \frac{\mathbf{1}_n\mathbf{1}_n^T}{n}\right).$$

By (3.11), it follows that

$$\text{(3.12)} \qquad \|\mathbf{J}[k-j:k]\|^2 \leq (1+j^2)\left\|\mathbf{Q}[k-j:k] - \tfrac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right\|^2,$$

and, since $\left\|\mathbf{Q}[k-j:k-1] - n^{-1}\mathbf{1}_n\mathbf{1}_n^T\right\|^2 \leq 4$,

$$\text{(3.13)} \qquad \|\mathbf{J}[k-j:k]\|^2 \leq 4(1+j^2) \leq 4(1+\hat{\tau}^2), \quad \forall\, j < \hat{\tau}.$$

Taking the norm square of (3.9),

$$\begin{aligned} \|\Psi_k\|^2 &= \left\|\mathbf{J}[k-\hat{\tau}:k]\Psi_{k-\hat{\tau}} + \alpha\sum_{j=1}^{\hat{\tau}} |\mathbf{J}[k-j+1:k]\mathbf{E}_{k-j}\right\|^2 \\ &\leq \left(1+\tfrac{1}{4}\right)\|\mathbf{J}[k-\hat{\tau}:k]\Psi_{k-\hat{\tau}}\|^2 + 5\alpha^2\left\|\sum_{j=1}^{\hat{\tau}}\mathbf{J}[k-j+1:k]\mathbf{E}_{k-j}\right\|^2 \\ \text{(3.14)} \quad &\leq \tfrac{5}{4}(1+\hat{\tau}^2)\left\|\mathbf{Q}[k-\hat{\tau}:k] - \tfrac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right\|^2\|\Psi_{k-\hat{\tau}}\|^2 + 20\alpha^2(1+\hat{\tau}^2)\hat{\tau}\sum_{j=1}^{\hat{\tau}}\|\mathbf{E}_{k-j}\|^2, \end{aligned}$$

where the first inequality is due to the fact that $\|a+b\|^2 \leq (1+\xi)\|a\|^2 + (1+\xi^{-1})\|b\|^2$ for any constant $\xi > 0$, and the second inequality follows by (3.12) with $j = \hat{\tau}$, (3.13), and the fact that $\left\|\sum_{j=1}^{\hat{\tau}} a_j\right\|^2 \leq \hat{\tau}\sum_{j=1}^{\hat{\tau}}\|a_j\|^2$. We next bound $\|\mathbf{E}_{p-1}\|$ for any $p \geq 1$. By the definition of $\mathbf{E}_k$ (3.10) with $k = p$,

$$\text{(3.15)} \quad \|\mathbf{E}_{p-1}\|^2 \leq \left\|\left(\mathbf{I}_n - \frac{\mathbf{1}_n\mathbf{1}_n^T}{n}\right)(\nabla\mathbf{f}(\mathbf{x}_p) - \nabla\mathbf{f}(\mathbf{x}_{p-1}))\right\|^2 \leq \|\nabla\mathbf{f}(\mathbf{x}_p) - \nabla\mathbf{f}(\mathbf{x}_{p-1}))\|^2.$$

The term on the right-hand-side of (3.15) can be bounded as follows

$$\begin{aligned}
&\|\nabla \mathbf{f}(\mathbf{x}_p) - \nabla \mathbf{f}(\mathbf{x}_{p-1})\|^2 \\
\leq& L^2\|\mathbf{x}_p - \mathbf{x}_{p-1}\|^2 \\
=& L^2\|(\mathbf{Q}_{p-1} - \mathbf{I}_n)(\mathbf{x}_{p-1} - \bar{\mathbf{x}}_{p-1}) - \alpha\mathbf{Q}_{p-1}\mathbf{y}_{p-1})\|^2 \\
\leq& 2L^2\|(\mathbf{Q}_{p-1} - \mathbf{I}_n)(\mathbf{x}_{p-1} - \bar{\mathbf{x}}_{p-1})\|^2 + 2\alpha^2L^2\|\mathbf{y}_{p-1}\|^2 \\
\leq& 8L^2\|\mathbf{x}_{p-1} - \bar{\mathbf{x}}_{p-1}\|^2 + 4\alpha^2L^2\|\mathbf{y}_{p-1} - \bar{\mathbf{y}}_{p-1}\|^2 + 4\alpha^2L^2\|\bar{\mathbf{y}}_{p-1}\|^2, \qquad (3.16)
\end{aligned}$$

where we have used Assumption 3.1 to get the first inequality, (3.2) with $k = p - 1$ to substitute for $\mathbf{x}_p$ and the fact that $(\mathbf{Q}_{p-1} - \mathbf{I}_n)\bar{\mathbf{x}}_{p-1} = 0$ to get the equality, and $\|\mathbf{Q}_{p-1} - \mathbf{I}_n\| \leq 2$ to obtain the first term in the last inequality. By Assumption 3.1,

$$\begin{aligned}
\|\bar{\mathbf{y}}_{p-1}\|^2 =& n\|\bar{y}_{p-1}\|^2 \\
=& n\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(x_{i,p-1})\right\|^2 \\
\leq& 2n\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(x_{i,p-1}) - \frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\bar{x}_{p-1})\right\|^2 \\
&+ 2n\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\bar{x}_{p-1}) - \frac{1}{n}\sum_{i=1}^{n}\nabla f_i(x^*)\right\|^2 \\
\leq& 2L^2\|\mathbf{x}_{p-1} - \bar{\mathbf{x}}_{p-1}\|^2 + 4L\sum_{i=1}^{n}\left(f_i(\bar{x}_{p-1}) - f_i(x^*)\right). \qquad (3.17)
\end{aligned}$$

Combining (3.15), (3.16) and (3.17), and using the fact that $\alpha < 1/3L$, it follows that for any $p \geq 1$,

$$\begin{aligned}
\|\mathbf{E}_{p-1}\|^2 \leq& \|\nabla \mathbf{f}(\mathbf{x}_p) - \nabla \mathbf{f}(\mathbf{x}_{p-1})\|^2 \\
\leq& 9L^2\left(\|\mathbf{x}_{p-1} - \bar{\mathbf{x}}_{p-1}\|^2 + \alpha^2\|\mathbf{y}_{p-1} - \bar{\mathbf{y}}_{p-1}\|^2\right) \qquad (3.18) \\
&+ 16\alpha^2L^3\sum_{i=1}^{n}\left(f_i(\bar{x}_{p-1}) - f_i(x^*)\right).
\end{aligned}$$

Using (3.18) with $p = k - j + 1$ to bound $\|E_{k-j}\|$, $1 \leq j \leq \hat{\tau}$ in (3.14), we get,

$$\begin{aligned}
\|\Psi_k\|^2 \leq& 2(1+\hat{\tau}^2)\left\|\mathbf{Q}[k-\hat{\tau}:k] - \tfrac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right\|^2\|\Psi_{k-\hat{\tau}}\|^2 \\
&+ 180\alpha^2L^2(1+\hat{\tau}^2)\hat{\tau}\sum_{j=1}^{\hat{\tau}}\|\Psi_{k-j}\|^2 \\
&+ 320n\alpha^4L^3(1+\hat{\tau}^2)\hat{\tau}\sum_{j=1}^{\hat{\tau}}\left(f(\bar{x}_{k-j}) - f(x^*)\right),
\end{aligned}$$

which proves (3.5). To prove (3.6), we note that for $k < \hat{\tau}$, we can write (3.9) as,

$$\Psi_k = \mathbf{J}_{k-1}\Psi_{k-1} + \alpha\mathbf{E}_{k-1} = \mathbf{J}[0:k]\Psi_0 + \alpha\sum_{j=0}^{k-1}\mathbf{J}[k-j:k]\mathbf{E}_j. \qquad (3.19)$$

Taking the norm square of (3.19),

$$\|\Psi_k\|^2 \leq \left(1+\tfrac{1}{4}\right)\|\mathbf{J}[0:k]\Psi_0\|^2 + 5\alpha^2 \left\|\sum_{j=0}^{k-1}\mathbf{J}[k-j:k]\mathbf{E}_j\right\|^2$$

$$\leq 5(1+\hat{\tau}^2)\|\Psi_0\|^2 + 20\alpha^2(1+\hat{\tau}^2)\hat{\tau}\sum_{j=0}^{k-1}\|\mathbf{E}_j\|^2, \tag{3.20}$$

where we have used $\|a+b\|^2 \leq (1+\xi)\|a\|^2 + (1+\xi^{-1})\|b\|^2$ for any constant $\xi > 0$ in the first inequality and (3.13) to obtain the second inequality. The final result (3.6) can be derived using (3.18) with $p = j+1$ for $1 \leq j \leq k-1$ in (3.20). ∎

Next, we state an auxiliary lemma whose proof can be found in [48, Lemma 4].

LEMMA 3.2. *Suppose the non-negative scalar sequences $\{a_t\}_{t\geq 0}$ and $\{e_t\}_{t\geq 0}$ satisfy the following recursive relation for a fixed $\hat{\tau} \in \mathbb{N}$*

$$a_t \leq \rho' a_{t-\hat{\tau}} + \frac{b}{\hat{\tau}}\sum_{i=t-\hat{\tau}}^{t-1} a_i + c\sum_{i=t-\hat{\tau}}^{t-1} e_i + r, \qquad \text{if} \quad t \geq \hat{\tau}, \tag{3.21}$$

$$a_t \leq \rho'' a_0 + \frac{b}{\hat{\tau}}\sum_{i=0}^{t-1} a_i + c\sum_{i=0}^{t-1} e_i + r, \qquad \text{if} \quad t < \hat{\tau}, \tag{3.22}$$

where b, c, r , ρ'' are non-negative constants, $b \leq \rho'/4$ and $\rho' \in (0, 1/4)$. Then, for any $t \in \mathbb{N}$,

$$a_t \leq 20\rho''\Big(1-\frac{3\rho}{4\hat{\tau}}\Big)^t a_0 + 60c\sum_{i=0}^{t-1}\Big(1-\frac{3\rho}{4\hat{\tau}}\Big)^{t-i} e_i + \frac{26r}{\rho}, \tag{3.23}$$

where $\rho := 1 - 2\rho'$.

We are ready to state and prove the main theorem.

THEOREM 3.1. *Suppose that: (i) Assumptions 2.1 and 3.1 hold, and, (ii) $\mathbf{Q}_k$ are doubly stochastic matrices and $\hat{\mathbf{Q}}_k = \mathbf{Q}_k$ for $k \geq 0$. Let $x_{i,k}$ denote the iterates generated via the recursions* (3.2)-(3.3) *and $\bar{x}_k := n^{-1}\sum_{i=1}^n x_{i,k}$. Then, for all $k \geq 0$,*

$$\begin{aligned}&\|\bar{x}_k - x^*\|^2 \\ &\leq \left(1-\tfrac{\alpha\mu}{4}\right)^k\left(\|\bar{x}_0 - x^*\|^2 + \tfrac{1000L(1+\hat{\tau}^2)}{\mu n\left(1-\frac{\alpha\mu}{4}\right)}\left(\|\mathbf{x}_0 - \bar{\mathbf{x}}_0\|^2 + \alpha^2\|\mathbf{y}_0 - \bar{\mathbf{y}}_0\|^2\right)\right)\end{aligned} \tag{3.24}$$

where $\hat{\tau} \in \mathbb{N}$ with $\rho' := 2(1+\hat{\tau}^2)\max_{\hat{\tau}\leq t\leq k}\left\|\mathbf{Q}[t-\hat{\tau}:t] - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right\|^2 < 1/4$ and

$$\alpha < \min\left\{1, \tfrac{\sqrt{\rho'}}{58L\hat{\tau}^2}\right\}. \tag{3.25}$$

Proof. By (3.2), the optimization error of the average iterates for any $t \in \mathbb{N}$ is

$$\|\bar{x}_{t+1} - x^*\|^2 = \|\bar{x}_t - \alpha\bar{y}_t - x^*\|^2$$

$$= \left\|\bar{x}_t - \frac{\alpha}{n}\sum_{i=1}^n \nabla f_i(x_{i,t}) - x^*\right\|^2$$

$$= \|\bar{x}_t - x^*\|^2 - \frac{2\alpha}{n}\left\langle\sum_{i=1}^n \nabla f_i(x_{i,t}), \bar{x}_t - x^*\right\rangle + \alpha^2\left\|\frac{1}{n}\sum_{i=1}^n \nabla f_i(x_{i,t})\right\|^2, \tag{3.26}$$

where $\bar{y}_t = n^{-1}\sum_{i=1}^n \nabla f_i(x_{i,t})$. (This can be proven by an induction argument using (3.3).) The second term in (3.26) can be bounded as,

$$\begin{aligned}
&\left\langle \sum_{i=1}^n \nabla f_i(x_{i,t}), \bar{x}_t - x^* \right\rangle \\
&= \left\langle \sum_{i=1}^n \nabla f_i(x_{i,t}), \bar{x}_t - x_{i,t} \right\rangle + \left\langle \sum_{i=1}^n \nabla f_i(x_{i,t}), x_{i,t} - x^* \right\rangle \\
&\geq \sum_{i=1}^n \left[f_i(\bar{x}_t) - f_i(x_{i,t}) - \tfrac{L}{2}\|\bar{x}_t - x_{i,t}\|^2 + f_i(x_{i,t}) - f_i(x^*) + \tfrac{\mu}{2}\|x_{i,t} - x^*\|^2 \right] \\
&\geq \sum_{i=1}^n \left[f_i(\bar{x}_t) - f_i(x^*) - \tfrac{L+\mu}{2}\|\bar{x}_t - x_{i,t}\|^2 + \tfrac{\mu}{4}\|\bar{x}_t - x^*\|^2 \right], \qquad (3.27)
\end{aligned}$$

where Assumption 3.1 is used in the first inequality and the bound $\|\bar{x}_t - x^*\|^2 \leq 2\|\bar{x}_t - x_{i,t}\|^2 + 2\|x_{i,t} - x^*\|^2$ is used to derive the last inequality. The last term in (3.26) can be bounded as,

$$\begin{aligned}
&\left\| \frac{1}{n}\sum_{i=1}^n \nabla f_i(x_{i,t}) \right\|^2 \\
&= \left\| \frac{1}{n}\sum_{i=1}^n \nabla f_i(x_{i,t}) - \frac{1}{n}\sum_{i=1}^n \nabla f_i(\bar{x}_t) + \frac{1}{n}\sum_{i=1}^n \nabla f_i(\bar{x}_t) - \frac{1}{n}\sum_{i=1}^n \nabla f_i(x^*) \right\|^2 \\
&\leq \frac{2L^2}{n}\sum_{i=1}^n \|x_{i,t} - \bar{x}_t\|^2 + \frac{4L}{n}\sum_{i=1}^n (f_i(\bar{x}_t) - f_i(x^*)), \qquad (3.28)
\end{aligned}$$

where in the second summation we have used the fact that $\|\nabla f_i(\bar{x}_t) - \nabla f_i(x^*)\|^2 \leq 2L(f_i(\bar{x}_t) - f_i(x^*))$ by Assumption 3.1 [36, Theorem 2.1.5]. Using (3.27) and (3.28) in (3.26) along with $\alpha < 1/4L$, it follows that,

$$\begin{aligned}
\|\bar{x}_{t+1} - x^*\|^2 &\leq \left(1 - \tfrac{\alpha\mu}{2}\right)\|\bar{x}_t - x^*\|^2 - \frac{\alpha}{n}\left(\sum_{i=1}^n f_i(\bar{x}_t) - f_i(x^*)\right) \\
&\quad + \frac{(3L/2 + \mu)\alpha}{n}\sum_{i=1}^n \|\bar{x}_t - x_{i,t}\|^2 \\
&\leq \left(1 - \tfrac{\alpha\mu}{2}\right)\|\bar{x}_t - x^*\|^2 - \frac{\alpha}{n}\left(\sum_{i=1}^n f_i(\bar{x}_t) - f_i(x^*)\right) + \tfrac{5\alpha L}{2n}\|\Psi_t\|^2, \qquad (3.29)
\end{aligned}$$

where the last inequality follows due to $\|\bar{\mathbf{x}}_t - \mathbf{x}_t\|^2 \leq \|\Psi_t\|^2$. Let $r_t := \|\bar{x}_t - x^*\|^2$. Multiplying both sides of (3.29) by $w_{t+1} = (1 - \alpha\mu/4)^{-(t+1)}$, it follows that,

$$w_{t+1} r_{t+1} \leq w_t r_t - w_{t+1}\alpha\left(f(\bar{x}_t) - f(x^*)\right) + w_{t+1}\tfrac{5\alpha L}{2n}\|\Psi_t\|^2, \qquad (3.30)$$

where $w_{t+1}(1 - \alpha\mu/2) \leq w_t$.

Next, we express (3.5) (and (3.6)) in the form of (3.21) (and (3.22)) with $a_t = \|\Psi_t\|^2$, $b = 180\alpha^2 L^2(1 + \hat{\tau}^2)\hat{\tau}^2$, $c = 320n\alpha^4 L^3(1 + \hat{\tau}^2)\hat{\tau}$, $e_t = f(\bar{x}_t) - f(x^*)$ and $r = 0$. By Lemma 3.2, it follows that,

$$\|\Psi_t\|^2 \leq 100(1 + \hat{\tau}^2)\left(1 - \tfrac{3\rho}{4\hat{\tau}}\right)^t \|\Psi_0\|^2 + 19200n\alpha^4 L^3(1 + \hat{\tau}^2)\hat{\tau}^2 \sum_{j=0}^{t-1}\left(1 - \tfrac{3\rho}{4\hat{\tau}}\right)^{t-j} e_j. \qquad (3.31)$$

Note that the condition on the step size (3.25) ensures that $b < \rho'/4$. Multiplying both sides of (3.31) by $w_{t+1} := (1-\alpha\mu/4)^{-(t+1)}$ and summing from $t = 0$ to $k-1$

$$\begin{aligned}
&\sum_{t=0}^{k-1} \left(1 - \tfrac{\alpha\mu}{4}\right)^{-(t+1)} \|\Psi_t\|^2 \\
&\le 100(1+\hat{\tau}^2)\|\Psi_0\|^2 \sum_{t=0}^{k-1} \left(1 - \tfrac{\alpha\mu}{4}\right)^{-(t+1)} \left(1 - \tfrac{3\rho}{4\hat{\tau}}\right)^{t} \\
&\quad + 19200 n\alpha^4 L^3 (1+\hat{\tau}^2)\hat{\tau}^2 \sum_{t=0}^{k-1} \left(1 - \tfrac{\alpha\mu}{4}\right)^{-(k+1)} \sum_{j=0}^{t-1} \left(1 - \tfrac{3\rho}{4\hat{\tau}}\right)^{t-j} e_j.
\end{aligned} \tag{3.32}$$

By (3.25), we have $\alpha \le \frac{\sqrt{\rho'}}{L\hat{\tau}^2} \le \frac{1}{2L\hat{\tau}^2} \le \frac{\rho}{L\hat{\tau}} \le \frac{3\rho}{2\mu\hat{\tau}}$, where the second inequality is due to $\sqrt{\rho'} \le 1/2$, the third inequality follows by $\hat{\tau} \ge 1$ and $\rho = 1-2\rho' \ge 1/2$ for $\rho' < 1/4$, and the last inequality is due to the fact that $\mu < L$. Thus, it follows that

$$\tfrac{\alpha\mu}{2} \le \tfrac{3\rho}{4\hat{\tau}} \implies \tfrac{\alpha\mu}{2}\left(1 - \tfrac{\alpha\mu}{8}\right) \le \tfrac{3\rho}{4\hat{\tau}} \implies 1 - \tfrac{3\rho}{4\hat{\tau}} \le \left(1 - \tfrac{\alpha\mu}{4}\right)^2. \tag{3.33}$$

We use (3.33) to bound the two summations on the right-hand-side of (3.32) as follows

$$\sum_{t=0}^{k-1} \left(1 - \tfrac{\alpha\mu}{4}\right)^{-(t+1)} \left(1 - \tfrac{3\rho}{4\hat{\tau}}\right)^{t} \le \sum_{t=0}^{k-1} \left(1 - \tfrac{\alpha\mu}{4}\right)^{t-1} \le \tfrac{4w_1}{\alpha\mu}, \tag{3.34}$$

and

$$\begin{aligned}
&\sum_{t=0}^{k-1} \left(1 - \tfrac{\alpha\mu}{4}\right)^{-(t+1)} \sum_{j=0}^{t-1} \left(1 - \tfrac{3\rho}{4\hat{\tau}}\right)^{t-j} e_j \\
&= \sum_{t=0}^{k-1}\sum_{j=0}^{t-1} \left(1 - \tfrac{\alpha\mu}{4}\right)^{-(t+1)+j+1} \left(1 - \tfrac{3\rho}{4\hat{\tau}}\right)^{t-j} w_{j+1} e_j \\
&= \sum_{t=0}^{k-1}\sum_{j=0}^{t-1} \left(\tfrac{1-3\rho/4\hat{\tau}}{1-\alpha\mu/4}\right)^{t-j} w_{j+1} e_j \\
&\le \sum_{t=0}^{k-1}\sum_{j=0}^{t-1} \left(1 - \tfrac{\alpha\mu}{4}\right)^{t-j} w_{j+1} e_j \\
&\le \sum_{t=0}^{k-1} \left(1 - \tfrac{\alpha\mu}{4}\right)^{t} \sum_{t=0}^{k-1} w_{t+1} e_t \le \tfrac{4}{\alpha\mu} \sum_{t=0}^{k-1} w_{t+1} e_t,
\end{aligned} \tag{3.35}$$

where the second inequality is due to (3.33) and the relation $\sum_{t=0}^{k-1}\sum_{j=0}^{t-1} a_{t-j} b_j \le \sum_{t=0}^{k-1} a_t \sum_{t=0}^{k-1} b_t$ for any two non-negative scalar sequences a_t, b_t, $t \in \mathbb{N}$. By (3.34), (3.35) and (3.32), it follows that,

$$\begin{aligned}
&\sum_{t=0}^{k-1} w_{t+1} \|\Psi_t\|^2 \\
&\le \tfrac{400 w_1 (1+\hat{\tau}^2)}{\mu\alpha} \|\Psi_0\|^2 + \tfrac{76800 n\alpha^3 L^3 (1+\hat{\tau}^2)\hat{\tau}}{\mu} \sum_{t=0}^{k-1} w_{t+1} \left(f(\bar{x}_t) - f(x^*)\right).
\end{aligned} \tag{3.36}$$

Finally, summing (3.30) from $t = 0$ to $k-1$, and dividing by w_t, it follows that,

$$
\begin{aligned}
r_k \leq & \frac{1}{w_k}\Bigg(w_0 r_0 + \frac{1000 w_1 (1+\hat{\tau}^2) L}{n\mu} \|\Psi_0\|^2 \\
& + \left(\frac{192000\alpha^4 L^4 (1+\hat{\tau}^2)\hat{\tau}}{\mu} - \alpha\right) \sum_{t=0}^{k-1} w_{t+1} \left(f(\bar{x}_t) - f(x^*)\right) \Bigg),
\end{aligned}
$$

where we have used (3.36) to bound $\sum_t w_{t+1}\|\Psi_t\|^2$. To prove (3.24), we note that $w_k^{-1} = (1-\alpha\mu)^k$ by definition, and the last term in the above inequality is non-positive since $\alpha^3 \leq \frac{1}{2\times 307200 L^3 \hat{\tau}^3}$. ∎

Broadly, Theorem 3.1 establishes the decay of the optimization error for a gradient tracking method with time inhomgeneous weight matrices. The convergence rate of the algorithm remains linear even when using time-varying matrices, and the form of the convergence factor remains remarkably consistent. However, it is worth mentioning that this convergence factor can potentially be smaller due to the possibility of using smaller step sizes, which depend on the value of $\hat{\tau}$. In this context, the constant $\hat{\tau}$ determines the effect of the network on the step size via (3.25). More precisely, $\hat{\tau}$ is a constant chosen to ensure that $(1+\hat{\tau}^2)\left\|\mathbf{Q}[k-\hat{\tau}:k] - \frac{1}{n}\mathbf{1}\mathbf{1}^T\right\|^2$ is less than one. This implies that for better connected graphs, i.e., smaller $\left\|\mathbf{Q}[k-\hat{\tau}:k] - \frac{1}{n}\mathbf{1}\mathbf{1}^T\right\|^2$, $\hat{\tau}$ can be smaller so that α can be larger (cf. (3.25)). For time-inhomogeneous matrices satisfying Assumption 2.2, we can establish precise upper bounds on the value of $\hat{\tau}$ using the coefficient of ergodicity (cf. Corollary 3.1). We note that our final convergence bound has a better dependency in terms of the condition number by a factor of $\sqrt{\frac{L}{\mu}}$ as compared to [34]. We believe the reason for this improvement is due to the utilization of small-gain theorem in [33] compared to our more standard approach of bounding the spectral norm of the associated matrices (cf. Lemma 3.1).

Remark 3.2. *For fixed graphs, one can recover the optimal convergence rate of the* `DIGing` *algorithm [34], up to logarithmic factors, from Theorem 3.1. For* `GTA`*, we have $\mathbf{Q}_k = Q \otimes I_d$, for all $k \geq 0$. Then, for $\hat{\tau} < k$, if $\hat{\tau} > \mathcal{O}\left(\frac{1}{\sigma(Q)}\log\frac{1}{\sigma(Q)}\right)$,*

$$
\begin{aligned}
\rho' = 2(1+\hat{\tau}^2)\left\|\mathbf{Q}[k-\hat{\tau}:k] - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right\|^2 &\leq 2(1+\hat{\tau}^2)\prod_{j=k-\hat{\tau}}^{k-1}\left\|\mathbf{Q} - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right\|^2 \\
&\leq 4\hat{\tau}^2(1-\sigma(Q))^{2\hat{\tau}} < 1/4.
\end{aligned}
$$

which implies $\alpha = \tilde{\mathcal{O}}(\frac{\sigma^2(Q)}{L})$, where $\tilde{\mathcal{O}}(\cdot)$ hides logarithmic factors. Thus, from Theorem 3.1 we have $\|\bar{x}_T - x^*\|^2 \leq \epsilon$, if $T \geq \tilde{\mathcal{O}}\left(\frac{L}{\mu\sigma^2(Q)}\log\frac{1}{\epsilon}\right)$.

We have the following corollary to Theorem 3.1.

Corollary 3.1. *Suppose that: (i) Assumptions 2.1, 2.2 and 3.1 hold, (ii) the matrices $Q_k := [q_{ij}[k]]_{i\in[n],j\in[n]}$ are doubly stochastic and $\hat{\mathbf{Q}}_k = \mathbf{Q}_k$ for all $k \geq 0$, (iii) $q_{ii}[k] > 0$ for all $k \geq 0$ for at least one $i \in [n]$, and, (iv) if $q_{ij}[k] > 0$ for any $(i,j) \in \mathcal{E}$ and $k \geq 0$, then $q_{ij}[k] > q$ for some strictly positive constant $q > 0$ independent of k and (i,j). Let $\tau_\eta := \eta\bar{\tau}d_{\mathcal{G}}$, where $\bar{\tau}$ is defined in Assumption 2.2 and $\eta \in \mathbb{N}$ satisfies*

$$
\eta \geq \left\lceil \frac{\max\{\ln 16 n^3\bar{\tau}^2 d_{\mathcal{G}}^2, 16\ln 4/\gamma\}}{\gamma} \right\rceil \tag{3.37}
$$

where $\gamma := q^{d_{\mathcal{G}}\bar{\tau}}$. Then, if $\alpha = \mathcal{O}\left(\frac{1}{L\tau_\eta^2}\right)$, (3.24) is satisfied for $\bar{x}_k$ generated via the recursions (3.2)-(3.3).

Proof. To prove the corollary, we need to show that there exists a constant $\eta \in \mathbb{N}$ such that for $\tau_\eta = \eta\bar{\tau}d_{\mathcal{G}}$, we have $\rho' := 2(1+\tau_\eta^2)\left\|\mathbf{Q}[j-\tau_\eta : j] - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right\| < 1/4$ for any $\tau_\eta \le j \le t$ to ensure the results of Theorem 3.1 hold with $\hat{\tau} = \tau_\eta$. It follows that

$$\begin{aligned}
\delta(\mathbf{Q}[(j-\tau_\eta : j]) &\le \rho\left(\mathbf{Q}[(j-\tau_\eta : j]\right) \\
&\le \rho(\mathbf{Q}[j-\eta\bar{\tau}d_{\mathcal{G}} : j-(\eta-1)\bar{\tau}d_{\mathcal{G}}]) \cdots \rho(\mathbf{Q}[(j-\bar{\tau}d_{\mathcal{G}} : j])) \\
&\le (1-\gamma)^\eta,
\end{aligned}$$

where $\gamma := q^{\bar{\tau}d_{\mathcal{G}}}$, and the first, second and third inequalities are due to (2.7), (2.6) and (2.11), respectively. Following the same logic as in (2.16), it follows that,

$$\left\|\mathbf{Q}[j-\tau_\eta : j] - \tfrac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right\|_1 \le n\delta(\mathbf{Q}[j-\tau_\eta : j]) \le n\,(1-\gamma)^\eta \le n\exp(-\gamma\eta). \tag{3.38}$$

Consequently, this implies,

$$\begin{aligned}
2(1+\tau_\eta^2)\left\|\mathbf{Q}[j-\tau_\eta : j] - \tfrac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right\|^2 &\le 2(1+\tau_\eta^2)n\left\|\mathbf{Q}[j-\tau_\eta : j] - \tfrac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right\|_1^2 \\
&\le 2n^3(1+\eta^2\bar{\tau}^2d_{\mathcal{G}}^2)\exp(-2\gamma\eta) \\
&\le \underbrace{4n^3\bar{\tau}^2d_{\mathcal{G}}^2}_{:=A}\eta^2\exp(-2\gamma\eta), \qquad (3.39)
\end{aligned}$$

where the second inequality is due to (3.38) and the last inequality follows since $\eta\bar{\tau}d_{\mathcal{G}} \ge 1$. We next prove the following claim for any scalars $\eta, A \ge 1$ and $0 < \gamma < 1$:

$$\eta^2\exp(-2\gamma\eta) < \tfrac{1}{4A} \qquad \text{if} \qquad \eta > \left\lceil \max\left\{\tfrac{\ln 4A, 16\ln 4/\gamma}{\gamma}\right\}\right\rceil. \tag{3.40}$$

To prove the claim, we note that the assumed inequality implies $\left(1 - \frac{\ln\eta}{\gamma\eta}\right)\eta > \frac{\ln 4A}{2\gamma}$. Let $\tilde{\eta} \in \mathbb{R}$ be such that $0 < \ln\tilde{\eta}/\gamma\tilde{\eta} < 1/4$. Then, for any $\eta > \tilde{\eta}$,

$$\eta \ge \tfrac{2\ln 4A}{3\gamma}. \tag{3.41}$$

To prove the existence of a $\tilde{\eta}$ satisfying $\ln\tilde{\eta}/\tilde{\eta} \le \gamma/4 := \epsilon$, we consider $\tilde{\eta} = \frac{4\ln 1/\epsilon}{\epsilon}$, $\epsilon < \frac{1}{4}$. For such a $\tilde{\eta}$, we have, $\ln\tilde{\eta}/\tilde{\eta} = \epsilon\frac{\ln\frac{4}{\epsilon} + \ln\ln\frac{1}{\epsilon}}{4\ln 1/\epsilon} < \epsilon$. Combining (3.41) with $A = 4n^3\bar{\tau}^2d_{\mathcal{G}}^2$ and $\eta \ge \tilde{\eta} = 16\ln\left(4/\gamma\right)/\gamma$ gives the lower bound on η in (3.40). Finally, by (3.40), (3.39) can be bounded as, $2(1+\tau_\eta^2)\left\|\mathbf{Q}[j-\tau_\eta : j] - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right\|^2 \le A\eta^2\exp(-2\gamma\eta) < \frac{1}{4}$, which completes the proof. ∎

The exact convergence rate of `AC-GT` can be derived from Corollary 3.1. By (3.24), the number of iterations required to reach ϵ-accuracy, denoted by T, is of the order of $\mathcal{O}\left(\frac{L\tau_\eta^2}{\mu}\log\frac{1}{\epsilon}\right)$ since $\alpha = \mathcal{O}\left(\frac{1}{L\tau_\eta^2}\right)$. Using (3.37) to bound η in $\tau_\eta = \eta\bar{\tau}d_{\mathcal{G}}$, it follows

$$T = \mathcal{O}\left(\frac{L\eta^2\bar{\tau}^2d_{\mathcal{G}}^2}{\mu}\log\frac{1}{\epsilon}\right) = \tilde{\mathcal{O}}\left(\left(\frac{\bar{\tau}^2d_{\mathcal{G}}^2}{\gamma^2}\right)\frac{L}{\mu}\log\frac{1}{\epsilon}\right).$$

where $\tilde{\mathcal{O}}(\cdot)$ hides logarithmic factors. Compared to the iteration complexity of `GTA` (see Remark 3.2) under the connected graph assumption, we note that the number of iterations can potentially increase by a factor of $\tilde{\mathcal{O}}\left(\frac{\bar{\tau}^2d_{\mathcal{G}}^2}{\gamma^2}\right)$. This is expected given the weaker assumptions made, i.e., the underlying graph is static and connected. Despite the increased iteration complexity, one can potentially have savings in overall communication volume for `AC-GT` (cf. Section 4.2) analogous to those for `AC` (cf. Remark 2.4).

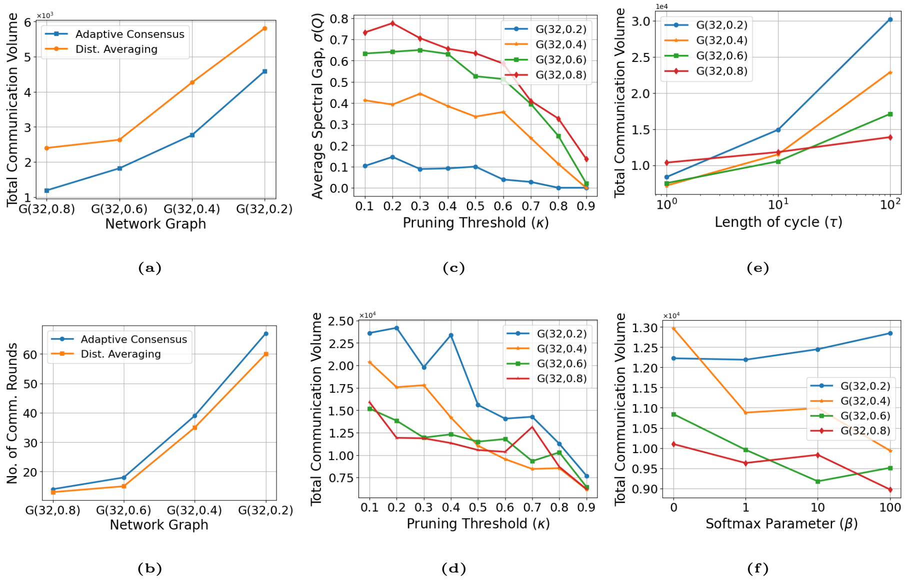


Fig. 1: **(a)-(b)** Total communication *volume*/*rounds* required to achieve a consensus error of 10^{-10}. **(c)** Variation of spectral gap with respect to pruning threshold, $\kappa \in \{0.1, 0.2, \ldots, 0.9\}$. **(d)-(f)** Total communication volume required to achieve a consensus error of 10^{-10} for different $\kappa \in \{0.1, 0.2, \ldots, 0.9\}$, $\tau \in \{1, 10^1, 10^2\}$ and $\beta \in \{0, 1, 10^1, 10^2\}$, respectively.

4. Numerical Experiments. In this section, we illustrate the empirical performance of `AC` and `AC-GT` via two sets of experiments. The first set of experiments demonstrates the benefits of `AC` compared to the distributed averaging algorithm in achieving consensus and illustrates the effect of the parameters of the pruning protocol on the performance of `AC`. The second set of experiments show the merits of `AC-GT` compared to popular methods on a linear regression problem with synthetic data [28], and a logistic regression problem with real datasets [29, 40] from the UCI repository [2]. All methods are implemented in Python, with a dedicated CPU core functioning as a node.

4.1. Performance of `AC`. We first showcase the effectiveness of `AC` in achieving consensus, where the goal is for all nodes to attain the average value of the initial estimates of the nodes [9, Section 1]. The network topologies (graphs) are generated randomly using the Erdös-Rényi graph model [16] and are represented as $G(n, p)$, where n represents the number of nodes, and $p \in \{0.2, 0.4, 0.6, 0.8\}$ denotes the probability with which each possible edge is independently included in the pruned graph. The performance metric used is the average consensus error, defined as $\frac{1}{|\mathcal{E}|}\sum_{(i,j)\in\mathcal{E}} \|x_i - x_j\|$, where $\mathcal{E}$ represents the set of all edges and $x_i \in \mathbb{R}^d$ for all $i \in [n]$ with $d = 10$. The total communication volume is measured as the total number of vectors exchanged amongst all the nodes in the network. The initial values $\{x_{i,0}\}_{i\in[n]}$ at each node are generated following a standard normal distribution.

Comparison to distributed averaging. Figs. 1**(a)-(b)** compare the performance of `AC` to distributed averaging [43]. The latter can be considered a specific case of `AC` with $\bar{\kappa} = 0$ and $\tau = \infty$. For the pruning protocol part of `AC`, we have set $\bar{\kappa}_i = \kappa = 0.75$ for all $i \in [n]$ and choose $\underline{\kappa}_i$ to ensure that $|\mathcal{E}_i| \geq 1$, so that each node has at least one neighbor. The softmax parameter is set to $\beta = 1$ and the cycle length is set to $\tau = 10$. The mixing matrix is generated using the Metropolis Hastings rule (cf. (2.1)). Fig. 1**(a)** shows a significant reduction in the total communication volume required to reach a consensus error of 10^{-10} as compared to distributed averaging across all graph topologies. Fig. 1**(b)** demonstrates that the number of communication rounds for `AC` undergoes only a modest increase as compared to distributed averaging.

Variation of pruning threshold (κ). In Fig. 1**(c)**, we plot the average spectral gap of the mixing matrices as a function of $\kappa \in \{0.1, 0.2, \dots, 0.9\}$. The average spectral gap is defined as the average of the spectral gaps of all the weight matrices obtained throughout the pruning cycles in a run of the algorithm. The plot reveals an important observation: pruning up to 50-60% of the edges does not significantly affect the spectral properties of the mixing matrix. Moreover, increasing the value of κ leads to a decrease in communication volume across all graphs, see Fig. 1**(d)**.

Variation of consensus cycle length (τ). Intuitively, one expects `AC` to perform better with shorter cycles since more frequent pruning of the graph can potentially allow `AC` to adapt more effectively to varying consensus errors. Fig. 1**(e)** confirms this intuition, where we consider $\tau \in \{1, 10, 100\}$ with $\kappa = 0.75$. While a value of $\tau = 1$ yields optimal performance in terms of communication volume, it necessitates executing the pruning protocol at every iteration.

Variation of softmax parameter (β). Fig. 1**(f)** plots the total communication volume required to achieve a consensus error of 10^{-10} as a function of the softmax parameter $\beta \in \{0, 1, 10, 100\}$ with $\tau = 10$ and $\kappa = 0.75$. The total communication volume is obtained by averaging over 100 independent trials. Fig. 1**(f)** shows that higher values of β tend to show a modest improvement in the performance.

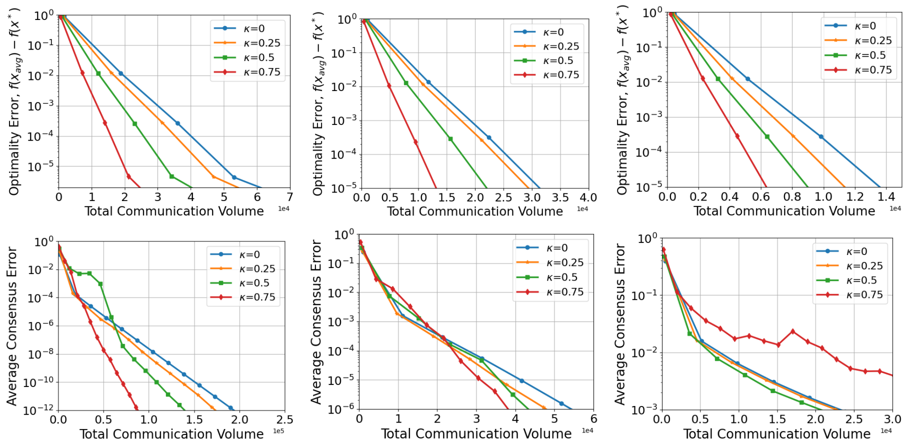
4.2. Performance of `AC-GT`. This subsection considers the evaluation of the performance of `AC-GT` on linear and logistic regression problems.

4.2.1. Linear Regression. We first consider a linear least-squares regression problem with synthetic data, formally defined as,

$$\min_{x \in \mathbb{R}^d} \quad f(x) := \tfrac{1}{N} \sum_{i=1}^{N} (a_i^T x - b_i)^2,$$

where $a_i \in \mathbb{R}^d$ denotes the ith feature vector and $b_i \in \mathbb{R}$ denotes the corresponding label. The data is generated using the technique proposed in [28] with $N = 32000$ and $d = 10$. The network topologies considered are $G(n, p)$, where $n = 32$ and $p \in \{0.2, 0.5, 0.8\}$. The data is partitioned uniformly in a disjoint manner amongst the nodes. We tuned the step size parameter in `AC-GT` using a grid-search over the range $\alpha \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}\}$ and present the results for the best step size. The softmax parameter is set to $\beta = 1$ and the cycle length is set to $\tau = 10$. The mixing matrix is generated using the Metropolis Hastings rule (cf. (2.1)).

Fig. 2 illustrates the performance of `AC-GT` in terms of two metrics, optimality error, defined as $f(x_{\text{avg}}) - f(x^*)$, where $x_{\text{avg}} = \frac{1}{n}\sum_{i=1}^{n} x_i$, and average consensus error described in Section 4.1, with respect to the total communication volume. The results suggest that, in terms of optimality error, it is preferable to use a higher value of κ, the pruning threshold. This observation is consistent across graph topologies. That said, there is a slight degradation in the decay of the consensus error as κ increases. This degradation becomes more noticeable in sparser topologies, as seen in Fig. 2**(c)**.



(a) G(32,0.8) **(b) G(32,0.5)** **(c) G(32,0.2)**

Fig. 2: Performance of `AC-GT` on linear regression problems for three different graphs, **(a)** G(32,0.8) **(b)** G(32,0.5) **(c)** G(32,0.2). **Top**: Optimality Error versus Total Communication Volume. **Bottom**: Average Consensus Error versus Total Communication Volume.

4.2.2. Logistic Regression. We consider ℓ_2-regularized logistic regression problems with real datasets of the form,

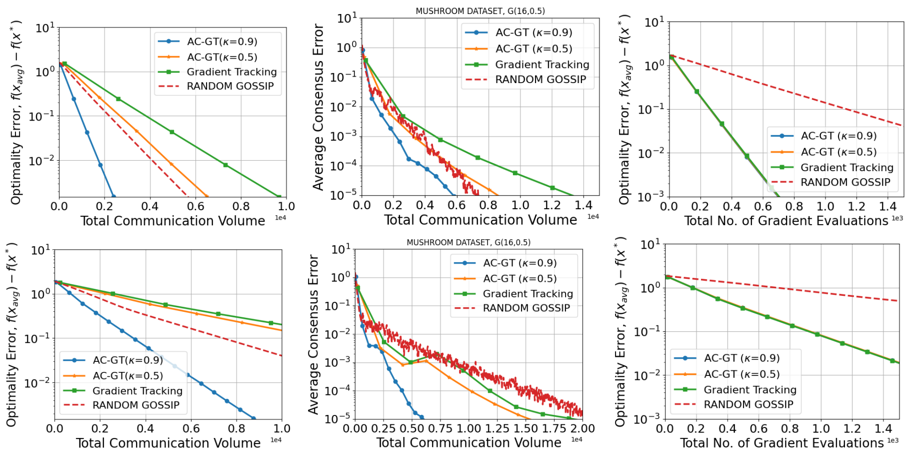
$$\min_{x\in\mathbb{R}^d} \quad f(x) := -\tfrac{1}{N}\sum_{i=1}^{N}\left\{b_i \log\sigma(a_i^T x) + (1-b_i)\log\left(1-\sigma(a_i^T x)\right)\right\} + \tfrac{\lambda}{2}\|x\|^2$$

where $\{a_i, b_i\}_{i=1}^N$ represent the training samples with label $b_i \in \{0,1\}$, $\lambda > 0$ is the regularization parameter and $\sigma(z) = \frac{1}{1+\exp(-z)}$, $\forall z \in \mathbb{R}$ is the sigmoid function.

We consider the Statlog [40] and the Mushroom [29] datasets from the UCI repository [2]. The Statlog dataset consists of $N = 690$ samples and $d = 14$ features whereas the Mushroom dataset consists of $N = 8124$ samples and $d = 22$ features. For these experiments, we consider $G(n,p)$ with $n = 16$ and $p = 0.5$. The data partition and the algorithm parameters for `AC-GT` are set in the same manner as Section 4.2.1. The step size is tuned using a grid-search over the range $\alpha \in \{10^{-4}, 10^{-3}, 10^{-3}, 10^{-1}, 10^{0}\}$ for all the algorithms. The regularization parameter is set to $\lambda = 10^{-4}$. The optimal solution x^* is computed using the L-BFGS algorithm from the SciPy library in Python and solving the problems to high accuracy.

The performance of `AC-GT` is compared to EXTRA [49] a popular gradient tracking algorithm (denoted by "Gradient Tracking" in the plots) and the random gossip algorithm [9][3]. In addition to the previous metrics, we also report the optimality error versus the total number of gradient evaluations of $f(\cdot)$. From the optimality error plots shown in Figs. 3**(a)** and **(c)**, it is evident that `AC-GT` with a parameter value of $\kappa = 0.9$ exhibits the best performance. Note that the curves of the algorithms AC-GT and Gradient Tracking are overlapping here. While the optimality error of random gossip is comparable to `AC-GT` with $\kappa = 0.5$ in terms of total communication volume, `AC-GT` outperforms the former with respect to total gradient evaluations. As

[3]To solve the semi-definite problem required for implementing the random gossip algorithm from [9], we utilize the CVXPY library [14].



(a) (b) (c)

Fig. 3: Performance of `AC-GT` on logistic regression problems: **(a)** Optimality Error versus Total Communication Volume **(b)** Consensus Error versus Total Communication Volume **(c)** Optimality Error versus Total number of Gradient Evaluations. **Top:** Statlog Dataset, G(16,0.5). **Bottom:** Mushroom Dataset, G(16,0.5).

for the consensus error, there is no notable difference in algorithm performance for the Statlog dataset. However, for the Mushroom dataset, random gossip and gradient tracking appear to exhibit inferior performance.

5. Conclusion. In this paper, we have developed an adaptive randomized algorithmic framework aimed at enhancing the communication efficiency of decentralized algorithms. Based on this framework, we have proposed the `AC` algorithm to solve the consensus problem and the `AC-GT` algorithm to solve the decentralized optimization problem. The distinguishing feature of the framework is the ability to reduce the volume of communication by making use of the inherent network structure and local information. We have established theoretical convergence guarantees and have analyzed the impact of various algorithmic parameters on the performance of the algorithms. Numerical results on the consensus problem, and linear and logisitc regression problems, demonstrate that proposed algorithms achieve significant communication savings as compared to existing methodologies.

Finally, several interesting extensions of the proposed algorithmic framework can be considered. From a communication perspective, one could consider directed graphs. Most of the groundwork for this setting has already been laid out in this work and as mentioned earlier, the theory can be extended to accommodate push-pull gradient methods [39], where either row or column stochasticity is satisfied. Additionally, asynchronous updating within each consensus cycle can also be incorporated to alleviate the constraints imposed by slower (straggler) nodes. Other interesting directions include nonconvex problems, stochastic local information and inexact communication.

References.

[1] Mahmoud S Assran and Michael G Rabbat. Asynchronous gradient push. *IEEE Transactions on Automatic Control*, 66(1):168–183, 2020.
[2] Arthur Asuncion and David Newman. Uci machine learning repository, 2007.
[3] Randal W Beard and Vahram Stepanyan. Information consensus in distributed multiple

vehicle coordinated control. In *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475)*, volume 2, pages 2029–2034. IEEE, 2003.

[4] Albert S Berahas, Raghu Bollapragada, and Shagun Gupta. Balancing communication and computation in gradient tracking algorithms for decentralized optimization. *arXiv preprint arXiv:2303.14289*, 2023.

[5] Albert S Berahas, Raghu Bollapragada, Nitish Shirish Keskar, and Ermin Wei. Balancing communication and computation in distributed optimization. *IEEE Transactions on Automatic Control*, 64(8):3141–3155, 2018.

[6] Albert S Berahas, Raghu Bollapragada, and Ermin Wei. On the convergence of nested decentralized gradient methods with multiple consensus and gradient steps. *IEEE Transactions on Signal Processing*, 69:4192–4203, 2021.

[7] Albert S Berahas, Charikleia Iakovidou, and Ermin Wei. Nested distributed gradient methods with adaptive quantized communication. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 1519–1525. IEEE, 2019.

[8] Dimitri Bertsekas and John Tsitsiklis. *Parallel and distributed computation: numerical methods*. Athena Scientific, 2015.

[9] Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. Randomized gossip algorithms. *IEEE transactions on information theory*, 52(6):2508–2530, 2006.

[10] Annie I Chen and Asuman Ozdaglar. A fast distributed proximal-gradient method. In *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 601–608. IEEE, 2012.

[11] Titus Cieslewski, Siddharth Choudhary, and Davide Scaramuzza. Data-efficient decentralized visual slam. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 2466–2473. IEEE, 2018.

[12] George Cybenko. Dynamic load balancing for distributed memory multiprocessors. *Journal of parallel and distributed computing*, 7(2):279–301, 1989.

[13] Paolo Di Lorenzo and Gesualdo Scutari. Next: In-network nonconvex optimization. *IEEE Transactions on Signal and Information Processing over Networks*, 2(2):120–136, 2016.

[14] Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

[15] Alexandros G Dimakis, Soummya Kar, José MF Moura, Michael G Rabbat, and Anna Scaglione. Gossip algorithms for distributed signal processing. *Proceedings of the IEEE*, 98(11):1847–1864, 2010.

[16] Paul Erdős and Alfréd Rényi. On random graphs. *Publ. math. debrecen*, 6(290-297):18, 1959.

[17] Marguerite Frank. The braess paradox. *Mathematical Programming*, 20(1):283–302, 1981.

[18] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

[19] Christoforos N Hadjicostis and Themistoklis Charalambous. Average consensus in the presence of delays in directed graph topologies. *IEEE Transactions on Automatic Control*, 59(3):763–768, 2013.

[20] John Hajnal and Maurice S Bartlett. Weak ergodicity in non-homogeneous markov chains. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 54(2), pages 233–246. Cambridge University Press, 1958.

[21] Yubin He, Mingyu Yan, Mohammad Shahidehpour, Zhiyi Li, Chuangxin Guo, Lei Wu, and Yi Ding. Decentralized optimization of multi-area electricity-natural gas flows based on cone reformulation. *IEEE Transactions on Power Systems*, 33(4):4531–4542, 2017.

[22] Mark Herbster and Massimiliano Pontil. Prediction on a graph with a perceptron. *Advances in neural information processing systems*, 19, 2006.

[23] Mingyi Hong, Meisam Razaviyayn, Zhi-Quan Luo, and Jong-Shi Pang. A unified algorithmic framework for block-structured optimization involving big data: With applications in machine learning and signal processing. *IEEE Signal Processing Magazine*,

33(1):57–77, 2015.
[24] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.
[25] David Kempe, Alin Dobra, and Johannes Gehrke. Gossip-based computation of aggregate information. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 482–491. IEEE, 2003.
[26] Anastasiia Koloskova, Tao Lin, and Sebastian U Stich. An improved analysis of gradient tracking for decentralized machine learning. *Advances in Neural Information Processing Systems*, 34:11422–11435, 2021.
[27] Guanghui Lan, Soomin Lee, and Yi Zhou. Communication-efficient algorithms for decentralized and stochastic optimization. *Mathematical Programming*, 180(1):237–284, 2020.
[28] Melanie L Lenard and Michael Minkoff. Randomly generated test problems for positive definite quadratic programming. *ACM Transactions on Mathematical Software (TOMS)*, 10(1):86–96, 1984.
[29] Gary H Lincoff. *Field guide to North American mushrooms*. Knopf National Audubon Society, 1997.
[30] Eran Malach, Gilad Yehudai, Shai Shalev-Schwartz, and Ohad Shamir. Proving the lottery ticket hypothesis: Pruning is all you need. In *International Conference on Machine Learning*, pages 6682–6691. PMLR, 2020.
[31] Daniel K Molzahn, Florian Dörfler, Henrik Sandberg, Steven H Low, Sambuddha Chakrabarti, Ross Baldick, and Javad Lavaei. A survey of distributed optimization and control algorithms for electric power systems. *IEEE Transactions on Smart Grid*, 8(6):2941–2962, 2017.
[32] Angelia Nedic, Alex Olshevsky, Asuman Ozdaglar, and John N Tsitsiklis. Distributed subgradient methods and quantization effects. In *2008 47th IEEE conference on decision and control*, pages 4177–4184. IEEE, 2008.
[33] Angelia Nedic, Alex Olshevsky, Asuman Ozdaglar, and John N Tsitsiklis. On distributed averaging algorithms and quantization effects. *IEEE Transactions on automatic control*, 54(11):2506–2517, 2009.
[34] Angelia Nedic, Alex Olshevsky, and Wei Shi. Achieving geometric convergence for distributed optimization over time-varying graphs. *SIAM Journal on Optimization*, 27(4):2597–2633, 2017.
[35] Angelia Nedich. Convergence rate of distributed averaging dynamics and optimization in networks. *Foundations and Trends® in Systems and Control*, 2(1):1–100, 2015.
[36] Yurii Nesterov. Introductory lectures on convex programming volume i: Basic course. *Lecture notes*, 3(4):5, 1998.
[37] Alex Olshevsky and John N Tsitsiklis. Convergence speed in distributed consensus and averaging. *SIAM journal on control and optimization*, 48(1):33–55, 2009.
[38] Joel B Predd, Sanjeev R Kulkarni, and H Vincent Poor. A collaborative training algorithm for distributed learning. *IEEE Transactions on Information Theory*, 55(4):1856–1871, 2009.
[39] Shi Pu, Wei Shi, Jinming Xu, and Angelia Nedić. Push–pull gradient methods for distributed optimization in networks. *IEEE Transactions on Automatic Control*, 66(1):1–16, 2020.
[40] J. Ross Quinlan. Simplifying decision trees. *International journal of man-machine studies*, 27(3):221–234, 1987.
[41] Michael G Rabbat and Robert D Nowak. Quantized incremental algorithms for distributed optimization. *IEEE Journal on Selected Areas in Communications*, 23(4):798–808, 2005.
[42] Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, and Ramtin Pedarsani. An exact quantized decentralized gradient descent algorithm. *IEEE Transactions on Signal Processing*, 67(19):4934–4947, 2019.
[43] Wei Ren, Randal W Beard, and Ella M Atkins. A survey of consensus problems in

multi-agent coordination. In *Proceedings of the 2005, American Control Conference, 2005.*, pages 1859–1864. IEEE, 2005.
[44] Peter Richtárik and Martin Takáč. Parallel coordinate descent methods for big data optimization. *Mathematical Programming*, 156(1):433–484, 2016.
[45] Ali H Sayed. Diffusion adaptation over networks. In *Academic Press Library in Signal Processing*, volume 3, pages 323–453. Elsevier, 2014.
[46] Ioannis D Schizas, Alejandro Ribeiro, and Georgios B Giannakis. Consensus in ad hoc wsns with noisy links—part i: Distributed estimation of deterministic signals. *IEEE Transactions on Signal Processing*, 56(1):350–364, 2007.
[47] Eugene Seneta. Coefficients of ergodicity: structure and applications. *Advances in applied probability*, 11(3):576–590, 1979.
[48] Suhail M. Shah and Raghu Bollapragada. A stochastic gradient tracking algorithm for decentralized optimization with inexact communication. *arXiv preprint arXiv:2307.14942*, 2020.
[49] Wei Shi, Qing Ling, Gang Wu, and Wotao Yin. Extra: An exact first-order algorithm for decentralized consensus optimization. *SIAM Journal on Optimization*, 25(2):944–966, 2015.
[50] Richard Steinberg and Willard I Zangwill. The prevalence of braess' paradox. *Transportation Science*, 17(3):301–318, 1983.
[51] John Tsitsiklis, Dimitri Bertsekas, and Michael Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE Transactions on Automatic Control*, 31(9):803–812, 1986.
[52] John N Tsitsiklis. Problems in decentralized decision making and computation. Technical report, Massachusetts Inst of Tech Cambridge Lab for Information and Decision Systems, 1984.
[53] Deniz Ustebay, Boris N Oreshkin, Mark J Coates, and Michael G Rabbat. Greedy gossip with eavesdropping. *IEEE Transactions on Signal Processing*, 58(7):3765–3776, 2010.
[54] Ashwin Verma, Marcos M Vasconcelos, Urbashi Mitra, and Behrouz Touri. Maximal dissent: a state-dependent way to agree in distributed convex optimization. *IEEE Transactions on Control of Network Systems*, 2023.
[55] Jacob Wolfowitz. Products of indecomposable, aperiodic, stochastic matrices. *Proceedings of the American Mathematical Society*, 14(5):733–737, 1963.
[56] Jinming Xu, Shanying Zhu, Yeng Chai Soh, and Lihua Xie. Augmented distributed gradient methods for multi-agent optimization under uncoordinated constant stepsizes. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 2055–2060. IEEE, 2015.
[57] Yuchen Zhang and Lin Xiao. Communication-efficient distributed optimization of self-concordant empirical loss. *Large-Scale and Distributed Optimization*, pages 289–341, 2018.