JOTA manuscript No.
(will be inserted by the editor)

Balancing Communication and Computation in Gradient Tracking Algorithms for Decentralized Optimization

Albert S. Berahas, Raghu Bollapragada, Shagun Gupta

Received: date / Accepted: date

Abstract Gradient tracking methods have emerged as one of the most popular approaches for solving decentralized optimization problems over networks. In this setting, each node in the network has a portion of the global objective function, and the goal is to collectively optimize this function. At every iteration, gradient tracking methods perform two operations (steps): (1) compute local gradients, and (2) communicate information with local neighbors in the network. The complexity of these two steps varies across different applications. In this paper, we present a framework that unifies gradient tracking methods and is endowed with flexibility with respect to the number of communication and computation steps. We establish unified theoretical convergence results for the algorithmic framework with any composition of communication and computation steps, and quantify the improvements achieved as a result of this flexibility. The framework recovers the results of popular gradient tracking methods as special cases, and allows for a direct comparison of these methods. Finally, we illustrate the performance of the proposed methods on quadratic functions and binary classification problems.

Keywords Decentralized Optimization · Gradient Tracking Methods · Network Optimization · Communication · Computation

Mathematics Subject Classification (2000) 49M05 · 49M37 · 65K05 · 90C06 · 90C25 · 90C30 · 90C35

Albert S. Berahas
University of Michigan
Ann Arbor, MI 48109, USA
albertberahas@gmail.com

Raghu Bollapragada, Corresponding author
University of Texas at Austin
Austin, TX 78712, USA
raghu.bollapragada@utexas.edu

Shagun Gupta
University of Texas at Austin
Austin, TX 78712, USA
shagungupta@utexas.edu

1 Introduction

We consider the problem of minimizing a function over a network. In this setting, each node of the network has a portion of the global objective function and the edges represent neighbor nodes that can exchange information, i.e., communicate. The goal is to collectively minimize a finite sum of functions where each component is only known to one of the n nodes (or agents) of the network. Such problems arise in many application areas such as machine learning [17, 51], sensor networks [4, 41], multi-agent coordination [10, 62] and signal processing [13]. The problem, known as a *decentralized optimization* problem, can be represented as follows:

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

where $f : \mathbb{R}^d \to \mathbb{R}$ is the global objective function, $f_i : \mathbb{R}^d \to \mathbb{R}$ for each $i \in \{1, 2, ..., n\}$ is the local objective function known only to node i and $x \in \mathbb{R}^d$ is the decision variable.

To decouple the computation across different nodes, (1) is often reformulated as

$$\begin{aligned} \min_{x_i\in\mathbb{R}^d} \quad & \frac{1}{n}\sum_{i=1}^{n} f_i(x_i) \\ \text{s.t.} \quad & x_i = x_j, \quad \forall\ (i,j) \in \mathcal{E}, \end{aligned} \tag{2}$$

where $x_i \in \mathbb{R}^d$ for each node $i \in \{1, 2, ..., n\}$ is a local copy of the decision variable, and $\mathcal{E}$ denotes the set of edges of the network; see e.g., [9, 38]. If the underlying network is connected, the *consensus* constraint ensures that all local copies are equal, and, thus, problems (1) and (2) are equivalent. For compactness, we express problem (2) as

$$\begin{aligned} \min_{x_i\in\mathbb{R}^d} \quad & \mathbf{f}(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} f_i(x_i) \\ \text{s.t.} \quad & (\mathbf{W} \otimes I_d)\mathbf{x} = \mathbf{x}, \end{aligned} \tag{3}$$

where $\mathbf{x} \in \mathbb{R}^{nd}$ is a concatenation of local copies x_i, $\mathbf{W} \in \mathbb{R}^{n\times n}$ is a matrix that captures the connectivity of the underlying network, $I_d \in \mathbb{R}^{d\times d}$ is the identity matrix of dimension d, and the operator $\otimes$ denotes the Kronecker product, $\mathbf{W} \otimes I_d \in \mathbb{R}^{nd\times nd}$. The matrix $\mathbf{W}$, known as the *mixing* matrix, is a symmetric, doubly-stochastic matrix with $w_{ii} > 0$ and $w_{ij} > 0$ $(i \neq j)$ if and only if $(i, j) \in \mathcal{E}$ in the underlying network. This matrix ensures that $(\mathbf{W} \otimes I_d)\mathbf{x} = \mathbf{x}$ if and only if $x_i = x_j\ \forall\ (i, j) \in \mathcal{E}$ in the connected network, thus, (2) and (3) are equivalent.

In this paper, we focus on gradient tracking methods. These first-order methods update and communicate the local decision variables, and also maintain, update and communicate an additional auxiliary variable that estimates (tracks) the gradient of the global objective function. We refer to the information shared by the methods as the communication strategy. When applied to the same decentralized setting, the theoretical convergence guarantees and practical implementations of gradient tracking methods with different communication strategies can vary significantly. We propose an algorithmic framework that unifies communication strategies in gradient tracking methods and that allows for a direct theoretical and empirical comparison. The framework recovers popular gradient tracking methods as special cases.

The update form of gradient tracking methods can be generalized and decomposed as: (1) one *computation step* of calculating the local gradients, and (2) one *communication step* of sharing information based on the communication strategy. The complexity (cost) of these two steps can vary significantly across applications. For example, a large-scale machine learning problem solved on a cluster of computers with shared memory access has a higher cost of computation than communication [51]. On the other hand, optimally allocating channels over a wireless sensor network requires economic usage of communications due to limited battery power [28]. The subject of developing algorithms (and convergence guarantees) that balance these costs has received significant attention in recent years; see e.g., [6–8, 11, 45, 61] and the references therein. In this paper, we follow the approach used in [6] and explicitly decompose the two steps. As a result, our algorithms are endowed with flexibility in terms of the number of communication and computation steps performed at each iteration. We show the benefits of this flexibility theoretically and empirically.

1.1 Literature Review

Decentralized Gradient Descent (DGD) [9, 38], a primal first-order method, is considered the prototypical method for solving (1). At each iteration nodes perform local computations and communicate local decision variable to neighbors. Gradient tracking methods, e.g., EXTRA [46], SONATA [48], NEXT [15], DIGing [36], Aug-DGM [57], have emerged as popular alternatives due to their superior theoretical guarantees and empirical performance. They maintain, update and communicate an additional auxiliary variable that tracks the average gradient (additional communication cost compared to DGD). These methods are usually applied to smooth convex functions over undirected networks; however, they are also applicable to various other settings such as time varying networks [36], uncoordinated step sizes [37, 57], directed networks [36, 43], nonconvex functions [15, 48] and stochastic gradients [42]. Our algorithmic framework generalizes and extends current gradient tracking methodologies, allowing for a unified analysis and direct comparison of popular methods. Notably, our framework differs significantly from existing works that aim to unify gradient tracking methods. In [49] and [60], semi-definite programming is used for this purpose. In [1] and [56], the authors introduce unifying frameworks, similar to those proposed in this paper, for comparing different communication strategies. However, our framework is simpler and allows for the exact specification of communication and computation steps at each iteration within the network. Furthermore, our proposed framework can accommodate a wider range of communication strategies than those discussed in [1, 49, 56, 60]. As a result of this increased algorithmic flexibility, our framework makes it possible to perform comprehensive comparisons among popular gradient tracking methods.

Another class of popular methods is primal-dual methods [2, 21, 26, 29, 30, 47, 52]. Of these methods, Flex-PD [30] and ADAPD [29] allow for flexibility with respect to the number of communication and computation steps. That said, Flex-PD [30] does not show improved performance with the employment of the flexibility and ADAPD [29] does not allow for a balance between communication and computation. In [40], the authors propose LU-GT, an algorithm that has similarities to our framework in terms of executing multiple local computation steps within gradient tracking methods. Despite the common motivation and similarities, there are several distinct and notable differences. The LU-GT algorithm has two step size hyper-parameters, whereas our approach has only one. Furthermore, our analysis results in less pessimistic step size conditions. It is worth noting that modifying LU-GT to align with our framework by setting the second step size to one is not possible due to the required conditions imposed in [40]. Additionally, our framework also provides a unifying foundation encompassing several popular gradient tracking methods.

Federated learning presents another class of algorithms for distributed optimization that operate in a client-server setup [31]. In this setup, there is a central server that coordinates the computations of all client nodes. This is equivalent to having a fully connected network with equal weights in the decentralized setting, which ensures all agents are always in consensus after a communication. As a result several algorithms have been proposed that perform multiple local updates to reduce communication load. Examples of such methods include, but are not limited to, Scaffnew [32], FedAvg [25], Scaffold [22], Local-SGD [19] and FedLin [33]. Moreover, specialized algorithms have been proposed in federated learning that leverage the additional control and network topology to improve performance via acceleration and momentum, and other problem-specific customization [12, 14, 23, 24, 27, 58]. In the fully decentralized setting considered in this work, where the network could be sparsely connected, the deviation of consensus error becomes highly critical when performing multiple local updates. Therefore, the steps taken in the proposed algorithm tend to be more conservative and are based on the connectivity of the network.

Several other settings beyond those considered in this work exist in decentralized optimization. These include directed networks for broadcast applications; proposed to be solved via gradient tracking methods and assuming row and column stochastic mixing matrices in [35, 36, 43, 53, 54], using only row stochastic mixing matrices in [50, 55, 59], and via implicit gradient tracking and using momentum in [18]. Proximal gradient-type methods have been proposed for tackling problems involving compositions of convex and non-convex functions [1, 3, 56]. While our work focuses on undirected networks and strongly convex problems, the proposed framework can be extended to these settings.

1.2 Contributions

We summarize our main contributions as follows:

1. We propose a gradient tracking algorithmic framework (`GTA`) that unifies communication strategies in gradient tracking methods and provides flexibility in the number of communication and computation steps performed at each iteration. The framework recovers as special cases popular gradient tracking methods, i.e., `GTA-1` [36, 46], `GTA-2` [15, 48] and `GTA-3` [37, 57]; see Table 1.
2. We establish the conditions required, on the communication strategy and the step size parameter, that guarantee a global linear rate of convergence for `GTA` with multiple communication and multiple computation steps. We also compare the relative performance of the special case gradient tracking algorithms, and illustrate the theoretical advantages of `GTA-3` over `GTA-2` (and `GTA-2` over `GTA-1`), a direct comparison not established in prior literature.
3. We show that the rate of convergence improves with increasing the number of communication steps, and the extent of improvement depends on the communication strategy. The improvements are much more profound in `GTA-3` as compared to `GTA-2` and `GTA-1`.
4. We illustrate the empirical performance of the proposed `GTA` framework on quadratic and binary classification logistic regression problems. We show the effect and benefits of multiple communication and/or computation steps per iteration on the performance of the algorithms.

1.3 Notation

Our proposed algorithmic framework is iterative and works with inner and outer loops. The variables $x_{i,k,j} \in \mathbb{R}^d$ and $y_{i,k,j} \in \mathbb{R}^d$ denote the local copies of the decision variable and the auxiliary variable, respectively, of node i, in outer iteration k and inner iteration j. The averages of all local decision variables and local auxiliary variables are denoted by $\bar{x}_{k,j} = \frac{1}{n}\sum_{i=1}^{n} x_{i,k,j}$ and $\bar{y}_{k,j} = \frac{1}{n}\sum_{i=1}^{n} y_{i,k,j}$, respectively. Boldface lowercase letters represent concatenated vectors of local copies

$$\mathbf{x}_{k,j} = \begin{bmatrix} x_{1,k,j} \\ x_{2,k,j} \\ \vdots \\ x_{n,k,j} \end{bmatrix} \in \mathbb{R}^{nd}, \quad \mathbf{y}_{k,j} = \begin{bmatrix} y_{1,k,j} \\ y_{2,k,j} \\ \vdots \\ y_{n,k,j} \end{bmatrix} \in \mathbb{R}^{nd}, \quad \nabla \mathbf{f}(\mathbf{x}_{k,j}) = \begin{bmatrix} \nabla f_1(x_{1,k,j}) \\ \nabla f_2(x_{2,k,j}) \\ \vdots \\ \nabla f_n(x_{n,k,j}) \end{bmatrix} \in \mathbb{R}^{nd}.$$

The concatenated vector of the average of decision variables ($\bar{x}_{k,j}$) and auxiliary variables ($\bar{y}_{k,j}$) repeated n times is denoted by $\bar{\mathbf{x}}_{k,j}$ and $\bar{\mathbf{y}}_{k,j}$, respectively. The n dimensional vector of all ones is denoted by 1_n and the identity matrix of dimension n is denoted by I_n. The spectral radius of square matrix A is $\rho(A)$. Matrix inequalities are defined component wise. The Kronecker product of any two matrices $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{d \times d}$ is represented using the operator $\otimes$ and denoted as $A \otimes B \in \mathbb{R}^{nd \times nd}$.

1.4 Paper Organization

In Section 2, we describe our proposed gradient tracking algorithmic framework (GTA). In Section 3, we provide theoretical convergence guarantees for the proposed algorithmic framework for multiple communication steps and a single computation step at each iteration (Subsection 3.1) and multiple communication and computation steps at each iteration (Subsection 3.2). In Subsection 3.3, we consider the special case of fully connected networks. Numerical experiments on quadratic and binary classification logistic regression problems are presented in Section 4. Finally, we provide concluding remarks in Section 5.

2 Gradient Tracking Algorithmic Framework

In this section, we describe our algorithmic framework (GTA) that unifies gradient tracking methods. We then extend the framework to allow for flexibility in the number of communication and computation steps performed at every iteration. Finally, we make remarks about the algorithmic framework and implementation, and then discuss popular gradient tracking methods as special cases of our proposed framework.

The iterate update form (for all $k \geq 0$) for the decision variable $\mathbf{x} \in \mathbb{R}^{nd}$ and the auxiliary variable $\mathbf{y} \in \mathbb{R}^{nd}$ that we propose to unify gradient tracking methods is

$$\begin{aligned} \mathbf{x}_{k+1,1} &= \mathbf{Z}_1 \mathbf{x}_{k,1} - \alpha \mathbf{Z}_2 \mathbf{y}_{k,1} \\ \mathbf{y}_{k+1,1} &= \mathbf{Z}_3 \mathbf{y}_{k,1} + \mathbf{Z}_4 (\nabla \mathbf{f}(\mathbf{x}_{k+1,1}) - \nabla \mathbf{f}(\mathbf{x}_{k,1})), \end{aligned} \tag{4}$$

where $\alpha > 0$ is the constant step size, $\mathbf{Z}_i = \mathbf{W}_i \otimes I_d \in \mathbb{R}^{nd \times nd}$ for $i = 1, 2, 3, 4$ and $\mathbf{W}_i \in \mathbb{R}^{n \times n}$ are communication matrices. A communication matrix $\mathbf{U} \in \mathbb{R}^{n \times n}$ is a symmetric, doubly stochastic matrix that respects the connectivity of the network, i.e., $u_{ii} > 0$ and $u_{ij} \geq 0$ $(i \neq j)$ if $(i, j) \in \mathcal{E}$ and $u_{ij} = 0$ $(i \neq j)$ if $(i, j) \notin \mathcal{E}$. The communication matrices, $\mathbf{W}_i$ for $i = 1, 2, 3, 4$, represent four (possibly different) network topologies consisting of all the nodes and (possibly different) subsets of the edges of the network over which the corresponding vectors are communicated. The update form given in (4) generalizes many popular gradient tracking methods for different choices of the communication matrices; see Table 1. While the methodology has similarities to [1, 56], our framework allows for the exact specification of the communication quantities within the network and does not impose any interdependent conditions among the communication matrices $\mathbf{W}_i$ for $i = 1, 2, 3, 4$. In (4) one communication and one computation step is performed at every iteration and so the inner iteration index is always 1. We include this subscript for consistency with the presentation of the algorithm and analysis with multiple communication and computation steps.

Table 1: Special cases of Gradient Tracking Algorithm (`GTA`).

Method	Communication Matrices				Algorithms in literature
	$\mathbf{W}_1$	$\mathbf{W}_2$	$\mathbf{W}_3$	$\mathbf{W}_4$	$(n_c = n_g = 1)$
`GTA-1`	$\mathbf{W}$	I_n	$\mathbf{W}$	I_n	DIGing [36], EXTRA [46],
`GTA-2`	$\mathbf{W}$	$\mathbf{W}$	$\mathbf{W}$	I_n	SONATA [48], NEXT [15, 43]
`GTA-3`	$\mathbf{W}$	$\mathbf{W}$	$\mathbf{W}$	$\mathbf{W}$	Aug-DGM [57], ATC-DIGing [37]

Note: $\mathbf{W}$ is a mixing matrix.

We incorporate multiple communications in (4) by replacing $\mathbf{Z}_i$ with $\mathbf{Z}_i^{n_c} = \mathbf{W}_i^{n_c} \otimes I_d$ for $i = 1, 2, 3, 4$, where $n_c \geq 1$ is the number of communication steps at each iteration. Taking the communication matrices to the n_c power represents performing n_c communication (consensus) steps at every iteration. We further extend (4) to incorporate multiple computation steps at each iteration. That is, the algorithm performs multiple local updates before communicating information with local neighbors. Our full algorithmic framework with flexibility in the number of communication and computation steps, i.e., $n_c \geq 1$ and $n_g \geq 1$, is given in Algorithm 1. A balance between the number of communication and computation steps is required to achieve overall efficiency for different applications, and `GTA` allows for such flexibility in these steps via the parameters n_g and n_c.

Algorithm 1 `GTA`: Gradient Tracking Algorithm

Inputs: initial point $\mathbf{x}_{0,1} \in \mathbb{R}^{nd}$, step size $\alpha > 0$, computations $n_g \geq 1$, communications $n_c \geq 1$.

```
1: $\mathbf{y}_{0,1} \leftarrow \nabla \mathbf{f}(\mathbf{x}_{0,1})$
2: for $k \leftarrow 0, 1, 2 \dots$ do
3:     if $n_g > 1$ then
4:         for $j \leftarrow 1, 2 \dots, n_g - 1$ do
5:             $\mathbf{x}_{k,j+1} \leftarrow \mathbf{x}_{k,j} - \alpha\, \mathbf{y}_{k,j}$
6:             $\mathbf{y}_{k,j+1} \leftarrow \mathbf{y}_{k,j} + \nabla \mathbf{f}(\mathbf{x}_{k,j+1}) - \nabla \mathbf{f}(\mathbf{x}_{k,j})$
7:         end for
8:     end if
9:     $\mathbf{x}_{k+1,1} \leftarrow \mathbf{Z}_1^{n_c} \mathbf{x}_{k,n_g} - \alpha\, \mathbf{Z}_2^{n_c} \mathbf{y}_{k,n_g}$
10:    $\mathbf{y}_{k+1,1} \leftarrow \mathbf{Z}_3^{n_c} \mathbf{y}_{k,n_g} + \mathbf{Z}_4^{n_c} (\nabla \mathbf{f}(\mathbf{x}_{k+1,1}) - \nabla \mathbf{f}(\mathbf{x}_{k,n_g}))$
11: end for
```

Remark 2.1 We make the following remarks about Algorithm 1.

- **Communications and Computations:** The number of communication and computation steps are dictated by n_c and n_g, respectively. By performing multiple communication steps, the goal is to improve consensus across the local decision variables. By performing multiple computation steps, the goal is for individual nodes to make more progress on their local objective functions.
- **Inner and Outer Loops:** Lines 2–8 form the outer loop and Lines 4–6 form the inner loop. The algorithm performs n_c communication steps each outer iteration (Lines 7 and 8). The algorithm performs n_g local (gradient) computations at each outer iteration; $n_g - 1$ computations in the inner loop (Line 6, $\nabla \mathbf{f}(\mathbf{x}_{k,j+1})$) and one computation in the outer loop

(Line 8, $\nabla \mathbf{f}(\mathbf{x}_{k+1,1})$). The inner loop is only executed if more than one computation, i.e., $n_g > 1$, is to be performed every outer iteration (Line 3). By default, we refer to outer iterations when we say iterations unless otherwise specified.

– **Step size ($\alpha > 0$):** The algorithm employs a constant step size that depends on the problem parameters, the choices of n_c and n_g, and the communication strategy, i.e., $\mathbf{W}_i$ for $i = 1, 2, 3, 4$.

We analyze `GTA` and provide results for several popular communication strategies as special cases; summarized in Table 1. The choice of the communication matrices ($\mathbf{W}_i$ for $i = 1, 2, 3, 4$), or equivalently the communication strategy, impact both the convergence of the algorithm and practical implementation. Notice that all methods in Table 1 require that $\mathbf{W}_1$ and $\mathbf{W}_3$ are mixing matrices. Our theoretical results recover this for the general framework. Consider `GTA-1`, `GTA-2` and `GTA-3` defined in Table 1 with $n_g = 1$. In `GTA-1` and `GTA-2`, computing local gradients and communications can be performed in parallel because the local gradients need not be communicated ($\mathbf{W}_4 = I_n$). On the other hand, in `GTA-3`, these steps need to be performed sequentially. Such trade-offs can create significant impact depending on the problem setting and system.

As mentioned above, the communication matrices ($\mathbf{W}_i$ for $i = 1, 2, 3, 4$) in `GTA` need not be the same. That is, different information can be exchanged on subsets of the edges of the network. This allows for a flexibility in the communication strategy that current gradient tracking methodologies do not possess. Such strategies can be useful in applying gradient tracking methods to decentralized settings with networks with bandwidth limitations, e.g., optimization problems in cyberphysical systems with battery powered wireless sensors [28].

3 Convergence Analysis

In this section, we present theoretical convergence guarantees for our proposed algorithmic framework (`GTA`). The analysis is divided into three subsections. In Subsection 3.1, we analyze the effect of multiple communications, i.e., $n_c \geq 1$ (and $n_g = 1$), on `GTA` and the three special cases `GTA-1`, `GTA-2` and `GTA-3`. While these results are a special case of the results presented in Subsection 3.2, we present these results first as they are simpler to derive, easier to follow and allow us to gain intuition about the effect of the number of communications. We then look at the effect of multiple computations in conjunction with multiple communications, i.e., $n_c \geq 1$ and $n_g \geq 1$, in Subsection 3.2 by extending the analysis from Subsection 3.1. In Subsection 3.3, we analyze `GTA-2` and `GTA-3` for fully connected networks; this special case is not captured by the analysis in the previous subsections.

We make the following assumption on the functions.

Assumption 3.1 *The global objective function $f : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex. Each component function $f_i : \mathbb{R}^d \to \mathbb{R}$ (for $i \in \{1, 2, \ldots, n\}$) has L-Lipschitz continuous gradients. That is, for all $z, z' \in \mathbb{R}^d$*

$$\begin{aligned} &f(z') \geq f(z) + \langle \nabla f(z), z' - z \rangle + \tfrac{\mu}{2}\|z' - z\|_2^2, \\ &\|\nabla f_i(z) - \nabla f_i(z')\|_2 \leq L\|z - z'\|_2, \qquad \forall\, i = 1, \ldots, n. \end{aligned}$$

By Assumption 3.1, the global minimizer of (1) is unique, and we denote it by x^*. For notational convenience, we define

$$\beta^{n_c} = \left\|\mathbf{W}^{n_c} - \tfrac{1_n 1_n^T}{n}\right\|_2, \quad \beta_i^{n_c} = \left\|\mathbf{W}_i^{n_c} - \tfrac{1_n 1_n^T}{n}\right\|_2, \qquad \forall\, i = 1, 2, 3, 4,$$

where $\beta \in [0,1)$ because $\mathbf{W}$ is a mixing matrix for a connected network and $\beta_i \in [0,1]$ because $\mathbf{W}_i$ for $i = 1,2,3,4$ are symmetric, doubly stochastic matrices. Using the definitions of $\mathbf{Z}^{n_c} = \mathbf{W}^{n_c} \otimes I_d$ and $\mathbf{Z}_i^{n_c} = \mathbf{W}_i^{n_c} \otimes I_d$ for $i = 1,2,3,4$, it follows that

$$\|\mathbf{Z}^{n_c} - \mathbf{I}\|_2 = \beta^{n_c}, \quad \|\mathbf{Z}_i^{n_c} - \mathbf{I}\|_2 = \beta_i^{n_c}, \qquad \forall\, i = 1,2,3,4. \tag{5}$$

We also define,

$$h_{k,j} = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(x_{i,k,j}), \quad \bar{h}_{k,j} = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\bar{x}_{k,j}), \quad \text{and} \quad \mathbf{I} = \frac{1_n 1_n^T}{n} \otimes I_d. \tag{6}$$

where $x_{i,k,j}$, denotes the local copy of the ith node, at outer iteration k and inner iteration j. In the analysis, for all $k \geq 0$, we consider the following error vector

$$r_k = \begin{bmatrix} \|\bar{x}_{k,1} - x^*\|_2 \\ \|\mathbf{x}_{k,1} - \bar{\mathbf{x}}_{k,1}\|_2 \\ \|\mathbf{y}_{k,1} - \bar{\mathbf{y}}_{k,1}\|_2 \end{bmatrix}.$$

The error vector r_k combines the optimization error, $\|\bar{x}_{k,1} - x^*\|_2$, and consensus errors, $\|\mathbf{x}_{k,1} - \bar{\mathbf{x}}_{k,1}\|_2$ and $\|\mathbf{y}_{k,1} - \bar{\mathbf{y}}_{k,1}\|_2$ where $\mathbf{x}_{k,1}$ and $\mathbf{y}_{k,1}$ are the first iterates of outer iteration k. We establish general technical lemmas that quantify the relation between r_{k+1} and r_k for each case of the presented algorithm.

3.1 `GTA` with multiple communication ($n_c \geq 1, n_g = 1$)

In this section, we analyze `GTA` when only one computation step is performed per iteration. In this setting ($n_g = 1$), the inner loop (Lines 4–6 in Algorithm 1) is never executed. Thus, the inner iteration counter in `GTA` can be ignored and the iteration simplifies to

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{Z}_1^{n_c}\mathbf{x}_k - \alpha \mathbf{Z}_2^{n_c}\mathbf{y}_k, \\ \mathbf{y}_{k+1} &= \mathbf{Z}_3^{n_c}\mathbf{y}_k + \mathbf{Z}_4^{n_c}(\nabla \mathbf{f}(\mathbf{x}_{k+1}) - \nabla \mathbf{f}(\mathbf{x}_k)). \end{aligned} \tag{7}$$

We note that throughout this subsection we drop the subscript related to the inner iteration j, i.e., $x_{i,k,j}$ is denoted as $x_{i,k}$ (since $j = 1$), and similarly with other quantities. We first establish the progression of the error vector r_k for iterates generated by (7) as a linear system in Lemma 3.1, introducing the matrix $A(n_c)$ in (8). Next, we analyse the spectral properties of the matrix $A(n_c)$ to establish qualitative trends for the convergence of iterates generated by (7) with respect to n_c, and provide a qualitative comparison between the methods described in Table 1 in Theorem 3.2. We then provide sufficient conditions on the step size in Theorem 3.3 using results from [42] and the Perron-Forbenius Theorem [20, Theorem 8.4.4]. We also provide a convergence rate for iterates generated by (7) in Theorem 3.4 using the approach from [44]. Finally, we provide the step size condition and convergence rate of the special cases from Table 1 in Corollary 3.2 and Corollary 3.3, respectively, and perform a theoretical comparison among the methods.

Lemma 3.1 *Suppose Assumption 3.1 holds and the number of gradient steps in each outer iteration of Algorithm 1 is set to one (i.e., $n_g = 1$). If $\alpha \leq \frac{1}{L}$, then for all $k \geq 0$,*

$$r_{k+1} \leq A(n_c) r_k,$$

$$\text{where} \quad A(n_c) = \begin{bmatrix} 1 - \alpha\mu & \frac{\alpha L}{\sqrt{n}} & 0 \\ 0 & \beta_1^{n_c} & \alpha\beta_2^{n_c} \\ \sqrt{n}\alpha\beta_4^{n_c}L^2 & \beta_4^{n_c}L(\|\mathbf{Z}_1^{n_c} - I_{nd}\|_2 + \alpha L) & \beta_3^{n_c} + \alpha\beta_4^{n_c}L \end{bmatrix}. \tag{8}$$

Proof If $n_g = 1$, using (7), the average iterates can be expressed as

$$\bar{x}_{k+1} = \bar{x}_k - \alpha \bar{y}_k,$$
$$\bar{y}_{k+1} = \bar{y}_k + h_{k+1} - h_k,$$

where h_k is defined in (6). Taking the telescopic sum of $\bar{y}_{i+1}$ from $i = 0$ to $k-1$ with $\bar{y}_0 = h_0$, it follows that

$$\bar{y}_k = h_k. \tag{9}$$

We first consider the optimization error on the average iterates. That is,

$$\begin{aligned}
\|\bar{x}_{k+1} - x^*\|_2 &= \|\bar{x}_k - \alpha \bar{y}_k + \alpha \hbar_k - \alpha \hbar_k - x^*\|_2 \\
&\leq \|\bar{x}_k - \alpha \hbar_k - x^*\|_2 + \alpha \|\bar{y}_k - \hbar_k\|_2 \\
&\leq (1 - \alpha\mu)\|\bar{x}_k - x^*\|_2 + \alpha \|h_k - \hbar_k\|_2 \\
&= (1 - \alpha\mu)\|\bar{x}_k - x^*\|_2 + \tfrac{\alpha}{n} \left\| \sum_{i=1}^n \nabla f_i(x_{i,k}) - \nabla f_i(\bar{x}_k) \right\|_2 \\
&\leq (1 - \alpha\mu)\|\bar{x}_k - x^*\|_2 + \tfrac{\alpha L}{n} \sum_{i=1}^n \|x_{i,k} - \bar{x}_k\|_2 \\
&\leq (1 - \alpha\mu)\|\bar{x}_k - x^*\|_2 + \tfrac{\alpha L}{\sqrt{n}} \|\mathbf{x}_k - \bar{\mathbf{x}}_k\|_2
\end{aligned} \tag{10}$$

where the first inequality is due to the triangle inequality, the second inequality is obtained by performing one gradient descent iteration on function f under Assumption 3.1 at the average iterate $\bar{x}_k$ with $\alpha \leq \frac{1}{L}$ [39, Theorem 2.1.14] and substituting using (9), the equality is due to (6), the second to last inequality follows by Assumption 3.1, and the last inequality is due to $\sum_{i=1}^n \|x_{i,k} - \bar{x}_k\|_2 \leq \sqrt{n}\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|_2$.

Next, we consider the consensus error in $\mathbf{x}_k$,

$$\begin{aligned}
\mathbf{x}_{k+1} - \bar{\mathbf{x}}_{k+1} &= \mathbf{Z}_1^{n_c}\mathbf{x}_k - \bar{\mathbf{x}}_k - \alpha \mathbf{Z}_2^{n_c}\mathbf{y}_k + \alpha \bar{\mathbf{y}}_k \\
&= \mathbf{Z}_1^{n_c}\mathbf{x}_k - \mathbf{Z}_1^{n_c}\bar{\mathbf{x}}_k - \alpha \mathbf{Z}_2^{n_c}\mathbf{y}_k + \alpha \mathbf{Z}_2^{n_c}\bar{\mathbf{y}}_k - \mathbf{I}(\mathbf{x}_k - \bar{\mathbf{x}}_k) + \mathbf{I}(\mathbf{y}_k - \bar{\mathbf{y}}_k) \\
&= (\mathbf{Z}_1^{n_c} - \mathbf{I})(\mathbf{x}_k - \bar{\mathbf{x}}_k) - \alpha (\mathbf{Z}_2^{n_c} - \mathbf{I})(\mathbf{y}_k - \bar{\mathbf{y}}_k).
\end{aligned}$$

where the second equality follows from adding $-\mathbf{I}(\mathbf{x}_k - \bar{\mathbf{x}}_k) = 0$ and $\mathbf{I}(\mathbf{y}_k - \bar{\mathbf{y}}_k) = 0$. By the triangle inequality and (5),

$$\begin{aligned}
\|\mathbf{x}_{k+1} - \bar{\mathbf{x}}_{k+1}\|_2 &\leq \|\mathbf{Z}_1^{n_c} - \mathbf{I}\|_2 \|\mathbf{x}_k - \bar{\mathbf{x}}_k\|_2 + \alpha \|\mathbf{Z}_2^{n_c} - \mathbf{I}\|_2 \|\mathbf{y}_k - \bar{\mathbf{y}}_k\|_2 \\
&= \beta_1^{n_c}\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|_2 + \alpha \beta_2^{n_c}\|\mathbf{y}_k - \bar{\mathbf{y}}_k\|_2.
\end{aligned} \tag{11}$$

Finally, we consider the consensus error in $\mathbf{y}_k$. By the triangle inequality and (5),

$$\begin{aligned}
&\|\mathbf{y}_{k+1} - \bar{\mathbf{y}}_{k+1}\|_2 \\
=& \|\mathbf{Z}_3^{n_c}\mathbf{y}_k - \bar{\mathbf{y}}_k + \mathbf{Z}_4^{n_c}(\nabla \mathbf{f}(\mathbf{x}_{k+1}) - \nabla \mathbf{f}(\mathbf{x}_k)) - \mathbf{I}(\nabla \mathbf{f}(\mathbf{x}_{k+1}) - \nabla \mathbf{f}(\mathbf{x}_k))\|_2 \\
\leq& \|(\mathbf{Z}_3^{n_c} - \mathbf{I})(\mathbf{y}_k - \bar{\mathbf{y}}_k)\|_2 + \|(\mathbf{Z}_4^{n_c} - \mathbf{I})(\nabla \mathbf{f}(\mathbf{x}_{k+1}) - \nabla \mathbf{f}(\mathbf{x}_k))\|_2 \\
\leq& \beta_3^{n_c}\|\mathbf{y}_k - \bar{\mathbf{y}}_k\|_2 + \beta_4^{n_c}\|\nabla \mathbf{f}(\mathbf{x}_{k+1}) - \nabla \mathbf{f}(\mathbf{x}_k)\|_2.
\end{aligned} \tag{12}$$

The last term in (12) can be bounded as follows,

$$\begin{aligned}\|\nabla \mathbf{f}(\mathbf{x}_{k+1}) - \nabla \mathbf{f}(\mathbf{x}_k)\|_2 &\leq L\|\mathbf{x}_{k+1} - \mathbf{x}_k\|_2 \\ &= L\|\mathbf{Z}_1^{n_c}\mathbf{x}_k - \alpha\mathbf{Z}_2^{n_c}\mathbf{y}_k - \mathbf{x}_k\|_2 \\ &= L\|(\mathbf{Z}_1^{n_c} - I_{nd})(\mathbf{x}_k - \bar{\mathbf{x}}_k) - \alpha\mathbf{Z}_2^{n_c}\mathbf{y}_k\|_2 \\ &\leq L\|\mathbf{Z}_1^{n_c} - I_{nd}\|_2\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|_2 + \alpha L\|\mathbf{Z}_2^{n_c}\|_2\|\mathbf{y}_k + \bar{\mathbf{y}}_k - \bar{\mathbf{y}}_k\|_2 \\ &\leq L\|\mathbf{Z}_1^{n_c} - I_{nd}\|_2\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|_2 + \alpha L\|\mathbf{y}_k - \bar{\mathbf{y}}_k\|_2 + \alpha L\left\|\bar{\mathbf{y}}_k\right\|_2, \end{aligned} \tag{13}$$

where the first inequality is due to Assumption 3.1, the first equality is due to iterate update form (7), the second equality is by adding $-(\mathbf{Z}_1^{n_c} - I_{nd})\bar{\mathbf{x}}_k = 0$ and the last two inequalities are applications of the triangle inequality. Next we bound the term $\|\bar{\mathbf{y}}_k\|_2$. By (9), Assumption 3.1 and $\sum_{i=1}^n \|x_{i,k} - \bar{x}_k\|_2 \leq \sqrt{n}\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|_2$,

$$\begin{aligned}\|\bar{\mathbf{y}}_k\|_2 &\leq \sqrt{n}\|\bar{y}_k\|_2 \\ &= \sqrt{n}\|h_k\|_2 \\ &\leq \sqrt{n}\left\|\frac{1}{n}\sum_{i=1}^n \nabla f_i(x_{i,k}) - \frac{1}{n}\sum_{i=1}^n \nabla f_i(\bar{x}_k)\right\|_2 + \sqrt{n}\left\|\frac{1}{n}\sum_{i=1}^n \nabla f_i(\bar{x}_k)\right\|_2 \\ &= \frac{1}{\sqrt{n}}\left\|\sum_{i=1}^n \nabla f_i(x_{i,k}) - \sum_{i=1}^n \nabla f_i(\bar{x}_k)\right\|_2 + \frac{1}{\sqrt{n}}\left\|\sum_{i=1}^n \nabla f_i(\bar{x}_k) - \sum_{i=1}^n \nabla f_i(x^*)\right\|_2 \\ &\leq L\left\|\mathbf{x}_k - \bar{\mathbf{x}}_k\right\|_2 + \sqrt{n}L\|\bar{x}_k - x^*\|_2. \end{aligned} \tag{14}$$

Thus, by (12), (13) and (14), it follows that

$$\begin{aligned}\|\mathbf{y}_{k+1} - \bar{\mathbf{y}}_{k+1}\|_2 \leq \beta_4^{n_c}\sqrt{n}\alpha L^2\|\bar{x}_k - x^*\|_2 + \beta_4^{n_c}L\left(\|\mathbf{Z}_1^{n_c} - I_{nd}\|_2 + \alpha L\right)\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|_2 \\ + \left(\beta_3^{n_c} + \beta_4^{n_c}\alpha L\right)\|\mathbf{y}_k - \bar{\mathbf{y}}_k\|_2. \end{aligned} \tag{15}$$

Combining (10), (11) and (15) concludes the proof. □

Using Lemma 3.1, we now provide the explicit form for $A(n_c)$ in order to establish the progression of the error vector r_k for the special cases defined in Table 1.

Corollary 3.1 *Suppose the conditions of Lemma 3.1 are satisfied. Then, the matrices $A(n_c)$ for the methods described in Table 1 are defined as:*

$$\begin{aligned} \texttt{GTA-1:} \quad A_1(n_c) &= \begin{bmatrix} 1-\alpha\mu & \frac{\alpha L}{\sqrt{n}} & 0 \\ 0 & \beta^{n_c} & \alpha \\ \sqrt{n}\alpha L^2 & L(2+\alpha L) & \beta^{n_c} + \alpha L \end{bmatrix}, \\ \texttt{GTA-2:} \quad A_2(n_c) &= \begin{bmatrix} 1-\alpha\mu & \frac{\alpha L}{\sqrt{n}} & 0 \\ 0 & \beta^{n_c} & \alpha\beta^{n_c} \\ \sqrt{n}\alpha L^2 & L(2+\alpha L) & \beta^{n_c} + \alpha L \end{bmatrix}, \\ \texttt{GTA-3:} \quad A_3(n_c) &= \begin{bmatrix} 1-\alpha\mu & \frac{\alpha L}{\sqrt{n}} & 0 \\ 0 & \beta^{n_c} & \alpha\beta^{n_c} \\ \beta^{n_c}\sqrt{n}\alpha L^2 & \beta^{n_c}L(2+\alpha L) & \beta^{n_c}(1+\alpha L) \end{bmatrix}. \end{aligned} \tag{16}$$

Proof Substituting the matrix values for each method in (8) and using $\|\mathbf{Z}_1^{n_c} - I_{nd}\|_2 \leq 2$ gives the desired result. □

The convergence properties of GTA when $n_g = 1$ can be analyzed using the spectral radius of the matrix $A(n_c)$. We now qualitatively establish the effect of n_c on $\rho(A(n_c))$, the spectral radius of the matrix $A(n_c)$, and the relative ordering between $\rho(A_1(n_c))$, $\rho(A_2(n_c))$ and $\rho(A_3(n_c))$.

Theorem 3.2 *Suppose Assumption 3.1 holds and the number of gradient steps in each outer iteration of Algorithm 1 is set to one (i.e., $n_g = 1$). If $\alpha \leq \frac{1}{L}$, then as n_c increases, $\rho(A(n_c))$ decreases where $A(n_c)$ is defined in (8). Thus, as n_c increases, $\rho(A_i(n_c))$ decreases, for $i = 1, 2, 3$ defined in (16). Moreover, if all three methods described in Table 1 (GTA-1, GTA-2 and GTA-3) employ the same step size,*

$$\rho(A_1(n_c)) \geq \rho(A_2(n_c)) \geq \rho(A_3(n_c)),$$

where the matrices $A_1(n_c)$, $A_2(n_c)$ and $A_3(n_c)$ are defined in (16).

Proof Note that $A(n_c) \geq 0$ and $A(n_c) \geq A(n_c + 1) \geq 0$. By [20, Corollary 8.1.19], it follows that $\rho(A(n_c)) \geq \rho(A(n_c+1))$. The same argument is applicable for $A_1(n_c)$, $A_2(n_c)$ and $A_3(n_c)$. Now, observe that $A_1(n_c) \geq A_2(n_c) \geq A_3(n_c) \geq 0$ when the same step size is employed. Thus, again by [20, Corollary 8.1.19], it follows that $\rho(A_1(n_c)) \geq \rho(A_2(n_c)) \geq \rho(A_3(n_c))$. □

We now derive conditions for establishing a linear rate of convergence to the solution for Algorithm 1 when $n_g = 1$ in terms of network parameters $(\beta_1, \beta_2, \beta_3, \beta_4)$ and objective function parameters $(L,\ \mu,\ \kappa = \frac{L}{\mu})$.

Theorem 3.3 *Suppose Assumption 3.1 holds and the number of gradient steps at each outer iteration of Algorithm 1 is set to one (i.e., $n_g = 1$). If the matrix $A(n_c)$ is irreducible, $\beta_1, \beta_3 < 1$ and*

$$\alpha < \min\left\{ \tfrac{1}{L},\ \tfrac{1-\beta_3^{n_c}}{L\beta_4^{n_c}},\ \tfrac{(1-\beta_1^{n_c}+2\beta_2^{n_c})}{2\beta_2^{n_c}\kappa(L+\mu)} \left(\sqrt{1 + \tfrac{4(1-\beta_1^{n_c})(1-\beta_3^{n_c})\beta_2^{n_c}(\kappa+1)}{\beta_4^{n_c}(1-\beta_1^{n_c}+2\beta_2^{n_c})^2}} - 1 \right) \right\}, \tag{17}$$

then, for all $\epsilon > 0$ there exists a constant $C_\epsilon > 0$ such that, for all $k \geq 0$,

$$\|r_k\|_2 \leq C_\epsilon(\rho(A(n_c)) + \epsilon)^k \|r_0\|_2, \quad \textit{where} \;\; \rho(A(n_c)) < 1.$$

Proof Following [42, Lemma 5], derived from the Perron-Forbenius Theorem [20, Theorem 8.4.4] for a 3×3 matrix, when the matrix $A(n_c)$ is non-negative and irreducible, it is sufficient to show that the diagonal elements of $A(n_c)$ are less than one and that $\det(I_3 - A(n_c)) > 0$ in order to guarantee $\rho(A(n_c)) < 1$. We upper bound $\|\mathbf{Z}_1^{n_c} - I_{nd}\|_2 \leq 2$ in $A(n_c)$ for the results.

Let us first consider the diagonal elements of the matrix $A(n_c)$. The first element is, $1 - \alpha\mu \leq 1 - \frac{\mu}{L} < 1$ by (17). The second element is $\beta_1^{n_c} < 1$ as $\beta_1 < 1$. Finally, the third element is $\beta_3^{n_c} + \alpha\beta_4^{n_c} L < \beta_3^{n_c} + \frac{1-\beta_3^{n_c}}{\beta_4^{n_c}L}\beta_4^{n_c} L = 1$ due to (17) and $\beta_3 < 1$.

Next, we consider

$$\begin{aligned}
&\det(I_3 - A(n_c)) \\
&= -\alpha(\alpha^2 L^2 \beta_2^{n_c}\beta_4^{n_c}(L+\mu) + \alpha\mu L\beta_4^{n_c}(1 - \beta_1^{n_c} + 2\beta_2^{n_c}) - \mu(1-\beta_1^{n_c})(1-\beta_3^{n_c})) \\
&= -L^2\beta_2^{n_c}\beta_4^{n_c}(L+\mu)\alpha(\alpha - \alpha_l)(\alpha - \alpha_u),
\end{aligned}$$

where $\alpha_l = \alpha_1 - \alpha_2$, $\alpha_u = \alpha_1 + \alpha_2$, and

$$\alpha_1 = \tfrac{-(1-\beta_1^{n_c}+2\beta_2^{n_c})}{2\beta_2^{n_c}\kappa(L+\mu)} \quad \text{and} \quad \alpha_2 = -\alpha_1\sqrt{1 + \tfrac{4(1-\beta_1^{n_c})(1-\beta_3^{n_c})\beta_2^{n_c}(\kappa+1)}{\beta_4^{n_c}(1-\beta_1^{n_c}+2\beta_2^{n_c})^2}}.$$

Observe that $\alpha_l < 0 < \alpha_u$ and $\alpha_2 > |\alpha_1|$. From (17), we have $0 < \alpha < \alpha_u$. Therefore, $\det(I_3 - A(n_c)) > 0$, which combined with the fact that the diagonal elements of the matrix are less than 1, implies $\rho(A(n_c)) < 1$. Finally, we bound the norm of error vector $\|r_k\|_2$ by telescoping $r_{i+1} \leq A(n_c)r_i$ from $i = 0$ to $k-1$ and the triangle inequality as

$$\|r_k\|_2 \leq \|A(n_c)^k\|_2 \|r_0\|_2.$$

From [20, Corollary 5.6.13], we can bound $\|A(n_c)^k\|_2 \leq C_\epsilon(\rho(A(n_c)) + \epsilon)^k$ where $\epsilon > 0$ and C_ϵ is a positive constant that depends on $A(n_c)$ and ϵ. □

The only constraint Theorem 3.3 imposes on the system (network) is $\beta_1, \beta_3 < 1$. This implies that the communication matrices $\mathbf{W}_1$ and $\mathbf{W}_3$ must represent connected networks (not necessarily the same network). Properties of $\mathbf{W}_2$ and $\mathbf{W}_4$ change the step size requirement but are not part of the sufficient conditions for convergence. Theorem 3.3 also does not require any relation among $\mathbf{W}_1$, $\mathbf{W}_2$, $\mathbf{W}_3$ and $\mathbf{W}_4$. This allows for more flexibility than the structures considered in the literature. The variables can be communicated along different connections within the network. We note that if $A(n_c)$ is a reducible matrix, the analysis for the progression of r_k can be further simplified from Lemma 3.1. For example, when $\mathbf{W} = \frac{1_n 1_n^T}{n}$, i.e., $\beta = 0$, in `GTA-2` and `GTA-3`. The analysis for these cases is presented in Subsection 3.3.

The next result establishes step size conditions that guarantee a linear rate of convergence for the three special cases (`GTA-1`, `GTA-2` and `GTA-3`).

Corollary 3.2 *Suppose Assumption 3.1 holds, $\mathbf{W} \neq \frac{1_n 1_n^T}{n}$, and the number of gradient steps at each outer iteration of Algorithm 1 is set to one (i.e., $n_g = 1$). If the following step size conditions hold for the methods described in Table 1,*

$$\text{GTA-1:} \quad \alpha < \min\left\{\frac{1-\beta^{n_c}}{L}, \frac{(3-\beta^{n_c})}{2\kappa(L+\mu)}\left(\sqrt{1+4(\kappa+1)\left(\frac{1-\beta^{n_c}}{3-\beta^{n_c}}\right)^2} - 1\right)\right\},$$

$$\text{GTA-2:} \quad \alpha < \min\left\{\frac{1-\beta^{n_c}}{L}, \frac{(1+\beta^{n_c})}{2\kappa(L+\mu)\beta^{n_c}}\left[\sqrt{1+4(\kappa+1)\beta^{n_c}\left[\frac{1-\beta^{n_c}}{1+\beta^{n_c}}\right]^2} - 1\right]\right\},$$

$$\text{GTA-3:} \quad \alpha < \min\left\{\frac{1}{L}, \frac{1-\beta^{n_c}}{L\beta^{n_c}}, \frac{(1+\beta^{n_c})}{2\kappa(L+\mu)\beta^{n_c}}\left(\sqrt{1+4(\kappa+1)\left(\frac{1-\beta^{n_c}}{1+\beta^{n_c}}\right)^2} - 1\right)\right\},$$

then, for all $\epsilon > 0$ there exist constants $C_{i,\epsilon} > 0$ such that, for all $k \geq 0$,

$$\|r_k\|_2 \leq C_{i,\epsilon}(\rho(A_i(n_c)) + \epsilon)^k \|r_0\|_2, \ \text{where} \ \rho(A_i(n_c)) < 1, \ \text{for} \ i = 1, 2, 3.$$

Proof The conditions given in Theorem 3.3 are satisfied for all three methods. That is, the matrices are irreducible as $\mathbf{W} \neq \frac{1_n 1_n^T}{n}$, i.e., $\beta > 0$ and $\beta_1, \beta_3 < 1$ in all the three methods as $\beta < 1$ because $\mathbf{W}$ is mixing matrix of a connected network. Thus, we can use (17) to derive the conditions on the step size for each of the methods. Substituting the values for $\beta_1, \beta_2, \beta_3$ and β_4 for each method yields the desired result. We should note that in `GTA-1` and `GTA-2`, we ignore the term $\frac{1}{L}$ since $\frac{1}{L} > \frac{1-\beta^{n_c}}{L}$. □

Corollary 3.2 shows how the communication strategy affects the step size when $n_g = 1$. Among the three methods, `GTA-3` allows for the largest step size, even having the possibility to use the step size $\frac{1}{L}$ if sufficiently large number of communications steps are performed (high n_c) and depending on β. Among `GTA-1` and `GTA-2`, `GTA-2` allows for a larger step size. While these share the same first term in the bound, the presence of the β^{n_c} factor in the denominator of the second term in `GTA-2` makes the bound larger than `GTA-1`, possibly allowing for a larger step size. Theorem 3.3 states that there exists a step size such that `GTA` converges at a linear rate when $n_g = 1$. We now proceed to analyze the convergence rate `GTA` when $n_g = 1$. Before that, we provide a technical lemma that shows that the largest eigenvalue of the matrix $A(n_c)$ is a positive real number.

Lemma 3.2 *Suppose Assumption 3.1 holds, the number of gradient steps at each outer iteration of Algorithm 1 is set to one (i.e., $n_g = 1$) and $\alpha \le \frac{1}{L}$. If the matrix $A(n_c)$ defined in (8) is irreducible, then, the spectral radius of $A(n_c)$ is the largest eigenvalue of $A(n_c)$ and is a positive real number. Consequently, if $\mathbf{W} \neq \frac{1_n 1_n^T}{n}$, the spectral radius of matrices $A_1(n_c)$, $A_2(n_c)$, $A_3(n_c)$ defined in (16) are also positive real numbers and equal to their largest eigenvalues, respectively.*

Proof The statement about the matrix $A(n_c)$ follows from the Perron-Forbenius Theorem [20, Theorem 8.4.4], and the fact that the matrix is non-negative and irreducible. Using similar arguments, the statement about the matrices $A_1(n_c)$, $A_2(n_c)$ and $A_3(n_c)$ follows as these matrices are irreducible when $\mathbf{W} \neq \frac{1_n 1_n^T}{n}$, i.e., $\beta > 0$. $\square$

The next theorem provides an upper bound on the convergence rate of `GTA` for sufficiently small constant step sizes.

Theorem 3.4 *Suppose Assumption 3.1 holds and the number of gradient steps at each outer iteration of Algorithm 1 is set to one (i.e., $n_g = 1$). If the matrix $A(n_c)$ is irreducible and $\alpha \le \frac{1}{L}$, then,*

$$\rho(A(n_c)) \le \lambda_u = \max\left\{1 - \frac{\alpha\mu}{2}, \hat{\lambda} + \sqrt{2\alpha L\kappa\beta_2^{n_c}\beta_4^{n_c}}\right\}, \tag{18}$$

where $\hat{\lambda} = \frac{\beta_1^{n_c}+\beta_3^{n_c}+L\alpha\beta_4^{n_c}+\sqrt{\left(\beta_1^{n_c}-\beta_3^{n_c}-L\alpha\beta_4^{n_c}\right)^2+4\beta_2^{n_c}\beta_4^{n_c}L^2\alpha^2+8L\alpha\beta_2^{n_c}\beta_4^{n_c}}}{2}$.

Proof Using Lemma 3.2, we know that the spectral radius of $A(n_c)$ is equal to the largest eigenvalue which is a positive real number. Following a similar approach to [44], we prove λ_u is an upper bound on the largest eigenvalue by showing the characteristic equation is non-negative at λ_u and strictly increasing for all values greater than λ_u. Consider

$$\begin{aligned} g(\lambda) &= \det(\lambda I_3 - A(n_c)) \\ &= (\lambda - 1 + \alpha\mu)\left((\lambda - \beta_1^{n_c})(\lambda - \beta_3^{n_c} - \alpha L\beta_4^{n_c}) - \alpha L(2+\alpha L)\beta_2^{n_c}\beta_4^{n_c}\right) \\ &\quad - \alpha^3 L^3 \beta_2^{n_c}\beta_4^{n_c} \\ &= (\lambda - 1 + \alpha\mu)q(\lambda) - \alpha^3 L^3\beta_2^{n_c}\beta_4^{n_c}, \end{aligned}$$

where $q(\lambda) = \lambda^2 - \lambda(\beta_1^{n_c} + \beta_3^{n_c} + L\alpha\beta_4^{n_c}) + \beta_1^{n_c}\beta_3^{n_c} + L\alpha\beta_4^{n_c}(\beta_1^{n_c} - 2\beta_2^{n_c} - L\alpha\beta_2^{n_c})$. Let the roots of the quadratic function $q(\lambda)$ be denoted as λ_1 and λ_2. Then, we have,

$$\begin{aligned} \max\{\lambda_1, \lambda_2\} &= \frac{\beta_1^{n_c}+\beta_3^{n_c}+L\alpha\beta_4^{n_c}+\sqrt{\left(\beta_1^{n_c}+\beta_3^{n_c}+L\alpha\beta_4^{n_c}\right)^2-4\left(\beta_1^{n_c}\beta_3^{n_c}+L\alpha\beta_4^{n_c}(\beta_1^{n_c}-2\beta_2^{n_c}-L\alpha\beta_2^{n_c})\right)}}{2} \\ &= \frac{\beta_1^{n_c}+\beta_3^{n_c}+L\alpha\beta_4^{n_c}+\sqrt{\left(\beta_1^{n_c}-\beta_3^{n_c}-L\alpha\beta_4^{n_c}\right)^2+4\beta_2^{n_c}\beta_4^{n_c}L^2\alpha^2+8L\alpha\beta_2^{n_c}\beta_4^{n_c}}}{2}. \end{aligned}$$

Thus, for any $\lambda \geq \max\left\{1-\alpha\mu, \hat{\lambda}\right\}$, the function $g(\lambda)$ is increasing and is lower bounded by $(\lambda - 1 + \alpha\mu)(\lambda - \hat{\lambda})^2 - \alpha^3 L^3 \beta_2^{n_c}\beta_4^{n_c}$. By $\lambda_u \geq \max\left\{1-\alpha\mu, \hat{\lambda}\right\}$,

$$\begin{aligned} g(\lambda_u) &\geq (\lambda - 1 + \alpha\mu)(\lambda - \hat{\lambda})^2 - \alpha^3 L^3 \beta_2^{n_c}\beta_4^{n_c} \\ &\geq \left(1 - \tfrac{\alpha\mu}{2} - 1 + \alpha\mu\right)(\lambda - \hat{\lambda})^2 - \alpha^3 L^3 \beta_2^{n_c}\beta_4^{n_c} \\ &\geq \tfrac{\alpha\mu}{2}\left(\tfrac{2\alpha L^2 \beta_2^{n_c}\beta_4^{n_c}}{\mu}\right) - \alpha^3 L^3 \beta_2^{n_c}\beta_4^{n_c} \\ &= \alpha^2 L^2 \beta_2^{n_c}\beta_4^{n_c}(1-\alpha L) \geq 0, \end{aligned}$$

where the second and third inequalities are due to the definition of λ_u and the final quantity is non-negative since $\alpha \leq \frac{1}{L}$. Therefore, by the above arguments, we conclude that $\rho(A(n_c)) \leq \lambda_u$ which completes the proof. □

Theorem 3.4 is derived independent of the conditions in Theorem 3.3. When $\rho(A(n_c)) < 1$ is imposed using Theorem 3.4, $\beta_1, \beta_3 < 1$ is a necessary condition for convergence. We show this by constructing a lower bound on λ_u, $\lambda_u \geq \hat{\lambda} \geq \frac{\beta_1^{n_c}+\beta_3^{n_c}+\left|\beta_1^{n_c}-\beta_3^{n_c}\right|}{2}$. For convergence we require $\lambda_u < 1$, i.e., $\frac{\beta_1+\beta_3+|\beta_1-\beta_3|}{2} < 1$, which implies $\beta_1, \beta_3 < 1$ as $\beta_1, \beta_3 \in [0,1]$. Thus, again we require $\mathbf{W}_1$ and $\mathbf{W}_3$ to represent a connected network. The step size condition in Theorem 3.3 is $\mathcal{O}(L^{-1}\kappa^{-0.5})$ while Theorem 3.4 requires $\mathcal{O}(L^{-1}\kappa^{-1})$, which is more pessimistic. That said, the precise and interpretable characterization of the convergence rate in Theorem 3.4 allows us to better differentiate amongst the communication strategies and the effect of n_c.

Corollary 3.3 *Suppose Assumption 3.1 holds, $\mathbf{W} \neq \frac{1_n 1_n^T}{n}$, and the number of gradient steps at each outer iteration of Algorithm 1 is set to one (i.e., $n_g = 1$). If $\alpha \leq \frac{1}{L}$, then, the spectral radii for the methods described in Table 1 satisfy*

$$\begin{aligned} &\texttt{GTA-1:} \quad \rho(A_1(n_c)) \leq \max\left\{1 - \tfrac{\alpha\mu}{2}, \beta^{n_c} + \sqrt{\alpha L}\left(2.5 + \sqrt{2\kappa}\right)\right\}, \\ &\texttt{GTA-2:} \quad \rho(A_2(n_c)) \leq \max\left\{1 - \tfrac{\alpha\mu}{2}, \beta^{n_c} + \sqrt{\alpha L}\left(2.5 + \sqrt{2\kappa\beta^{n_c}}\right)\right\}, \\ &\texttt{GTA-3:} \quad \rho(A_3(n_c)) \leq \max\left\{1 - \tfrac{\alpha\mu}{2}, \beta^{n_c}\left(1 + \sqrt{\alpha L}\left(2.5 + \sqrt{2\kappa}\right)\right)\right\}. \end{aligned}$$

Proof The conditions in Theorem 3.4 are satisfied due to Lemma 3.2. Thus, we can plug in the values for β_i $(i = 1,2,3,4)$ for each method to get an upper bound on the spectral radii. The upper bound λ_u for `GTA-1` can be simplified as

$$\begin{aligned} \hat{\lambda} + \sqrt{\tfrac{2\alpha L^2 \beta_2^{n_c}\beta_4^{n_c}}{\mu}} &= \tfrac{2\beta^{n_c} + L\alpha + \sqrt{5L^2\alpha^2 + 8L\alpha}}{2} + \sqrt{\tfrac{2\alpha L^2}{\mu}} \\ &= \beta^{n_c} + \tfrac{\sqrt{\alpha L}}{2}\left(\sqrt{\alpha L} + 2\sqrt{2\kappa} + \sqrt{8 + 5L\alpha}\right) \\ &\leq \beta^{n_c} + \sqrt{\alpha L}\left(2.5 + \sqrt{2\kappa}\right) \end{aligned}$$

where the last inequality is due to $\alpha \leq \frac{1}{L}$. Following the same approach, λ_u for `GTA-2` can be simplified as

$$\begin{aligned} \hat{\lambda} + \sqrt{\tfrac{2\alpha L^2 \beta_2^{n_c}\beta_4^{n_c}}{\mu}} &= \tfrac{2\beta^{n_c} + L\alpha + \sqrt{L^2\alpha^2 + 4L^2\alpha^2\beta^{n_c} + 8L\alpha\beta^{n_c}}}{2} + \sqrt{\tfrac{2\alpha L^2 \beta^{n_c}}{\mu}} \\ &= \beta^{n_c} + \tfrac{\sqrt{\alpha L}}{2}\left(\sqrt{\alpha L} + 2\sqrt{2\kappa\beta^{n_c}} + \sqrt{8\beta^{n_c} + 4L\alpha\beta^{n_c} + L\alpha}\right) \\ &\leq \beta^{n_c} + \sqrt{\alpha L}\left(2.5 + \sqrt{2\kappa\beta^{n_c}}\right) \end{aligned}$$

where the last inequality uses $\alpha \le \frac{1}{L}$ and $\beta < 1$. Finally, the upper bound λ_u for `GTA-3` is

$$\begin{aligned}\hat{\lambda} + \sqrt{\tfrac{2\alpha L^2 \beta_2^{n_c} \beta_4^{n_c}}{\mu}} &= \tfrac{2\beta^{n_c} + L\alpha\beta^{n_c} + \sqrt{5L^2\alpha^2(\beta^{n_c})^2 + 8L\alpha(\beta^{n_c})^2}}{2} + \sqrt{\tfrac{2\alpha L^2 (\beta^{n_c})^2}{\mu}} \\ &= \beta^{n_c}\left(1 + \tfrac{\sqrt{\alpha L}}{2}\left(\sqrt{\alpha L} + 2\sqrt{2\kappa} + \sqrt{8 + 5L\alpha}\right)\right) \\ &\le \beta^{n_c}\left(1 + \sqrt{\alpha L}\left(2.5 + \sqrt{2\kappa}\right)\right)\end{aligned}$$

where the last inequality is due to $\alpha \le \frac{1}{L}$ and $\beta < 1$. □

Corollary 3.3 characterizes the effect of multiple communication steps (when $n_g = 1$) on the convergence rates of `GTA-1`, `GTA-2` and `GTA-3`. First, the convergence rate improves with increased communications (increase in n_c) when $n_g = 1$ for all methods. The improvement is most apparent in `GTA-3` as increasing n_c drives the second term in the max bound to zero. Thus, if a sufficient number of communication steps are performed in `GTA-3`, the method can achieve convergence rates similar to those of gradient descent, i.e., $(1 - \frac{\alpha\mu}{2})$. The improvement is less apparent in `GTA-2` and the least in `GTA-1`. With an increase in n_c, the dominating term in the max bound, i.e., $\sqrt{2\alpha L\kappa}$, remains unchanged in `GTA-1` and changes to $\sqrt{2\alpha L\kappa\beta^{n_c}}$ in `GTA-2` which is affected by the number of communication steps n_c.

3.2 `GTA` with multiple communication and computation ($n_c \ge 1, n_g \ge 1$)

In this section, we analyze `GTA` when multiple computation and communication steps are performed every iteration. We extend the analysis from Subsection 3.1; the case $n_g = 1$ is a special case of the analysis in this section. The subscript for the inner iteration counter is re-introduced in this section as we consider cases with $n_g > 1$ and the inner loop (Lines 4–6 in Algorithm 1) is executed. We first present Lemma 3.3, which bounds the errors introduced due to the execution of the inner loop. Following a similar approach to Subsection 3.1, we establish the progression of the error vector r_k for Algorithm 1 as a linear system in Lemma 3.4 using the matrix $B(n_c, n_g)$. We then analyze the spectral properties of the matrix $B(n_c, n_g)$ to establish qualitative trends for the convergence rate of Algorithm 1 with respect to both n_c and n_g, and perform a qualitative comparison between methods described in Table 1 in Theorem 3.5. Finally, we provide sufficient conditions on the step size in Theorem 3.6 for linear convergence for any composition of communication and computation steps.

Lemma 3.3 *Suppose Assumption 3.1 holds and $\alpha \le \frac{1}{n_g L}$ in Algorithm 1. Then, for all $k \ge 0$ and $1 \le j \le n_g$*

$$\|\bar{\mathbf{y}}_{k,1}\|_2 \le \|\mathbf{y}_{k,1} - \bar{\mathbf{y}}_{k,1}\|_2 + L\left\|\mathbf{x}_{k,1} - \bar{\mathbf{x}}_{k,1}\right\|_2 + L\sqrt{n}\|\bar{x}_{k,1} - x^*\|_2 \tag{19}$$

$$\|\mathbf{x}_{k,j} - \mathbf{x}_{k,1}\|_2 \le 2\alpha(j-1)\|\mathbf{y}_{k,1}\|_2, \tag{20}$$

$$\|\mathbf{x}_{k,j} - \bar{\mathbf{x}}_{k,j}\|_2 \le 2\alpha(j-1)\|\mathbf{y}_{k,1}\|_2 + \|\mathbf{x}_{k,1} - \bar{\mathbf{x}}_{k,1}\|_2, \tag{21}$$

Proof Taking a telescopic sum of $\mathbf{y}_{k,i+1} = \mathbf{y}_{k,i} + \nabla\mathbf{f}(\mathbf{x}_{k,i+1}) - \nabla\mathbf{f}(\mathbf{x}_{k,i})$, the inner loop update, from $i = 1$ to $j - 1$ we get

$$\mathbf{y}_{k,j} = \mathbf{y}_{k,1} + \nabla\mathbf{f}(\mathbf{x}_{k,j}) - \nabla\mathbf{f}(\mathbf{x}_{k,1}). \tag{22}$$

Using (22), $\mathbf{y}_{k+1,1}$ can be expressed as

$$\begin{aligned}\mathbf{y}_{k+1,1} &= \mathbf{Z}_3^{n_c}\left(\mathbf{y}_{k,1} + \nabla\mathbf{f}(\mathbf{x}_{k,n_g}) - \nabla\mathbf{f}(\mathbf{x}_{k,1})\right) + \mathbf{Z}_4^{n_c}\left(\nabla\mathbf{f}(\mathbf{x}_{k+1,1}) - \nabla\mathbf{f}(\mathbf{x}_{k,n_g})\right) \\ &= \mathbf{Z}_3^{n_c}\mathbf{y}_{k,1} + \mathbf{Z}_4^{n_c}\nabla\mathbf{f}(\mathbf{x}_{k+1,1}) - \mathbf{Z}_3^{n_c}\nabla\mathbf{f}(\mathbf{x}_{k,1}) + \mathbf{Z}_3^{n_c}\nabla\mathbf{f}(\mathbf{x}_{k,n_g}) - \mathbf{Z}_4^{n_c}\nabla\mathbf{f}(\mathbf{x}_{k,n_g}).\end{aligned} \tag{23}$$

Taking the component-wise average across all nodes in (22) and (23) and using (6), it follows that

$$\bar{y}_{k,j} = \bar{y}_{k,1} + h_{k,j} - h_{k,1}, \tag{24}$$
$$\bar{y}_{k+1,1} = \bar{y}_{k,1} + h_{k+1,1} - h_{k,1}. \tag{25}$$

Performing a similar telescopic sum as (9) with (25), we obtain $\bar{y}_{k,1} = h_{k,1}$. Thus, substituting $\bar{y}_{k,1} = h_{k,1}$ in (24) yields

$$\bar{y}_{k,j} = \bar{y}_{k,1} + h_{k,j} - h_{k,1} = h_{k,j}. \tag{26}$$

By the triangle inequality, $\|\mathbf{y}_{k,1}\|_2 \leq \|\mathbf{y}_{k,1} - \bar{\mathbf{y}}_{k,1}\|_2 + \|\bar{\mathbf{y}}_{k,1}\|_2$, where $\|\bar{\mathbf{y}}_{k,1}\|_2$ can be bounded by a similar procedure to (14) due to $\bar{y}_{k,1} = h_{k,1}$ to yield (19). Now, taking the telescopic sum of the inner loop update $\mathbf{x}_{k,i} = \mathbf{x}_{k,i-1} - \alpha \mathbf{y}_{k,i-1}$ from $i = 2$ to j yields $\mathbf{x}_{k,j} = \mathbf{x}_{k,1} - \alpha \sum_{i=1}^{j-1} \mathbf{y}_{k,i}$. The sum $\sum_{i=1}^{j-1} \mathbf{y}_{k,i}$ is evaluated using (22) as

$$\begin{aligned} \sum_{i=1}^{j-1} \mathbf{y}_{k,j} &= \mathbf{y}_{k,1} + \sum_{i=2}^{j-1} \mathbf{y}_{k,i} + \nabla \mathbf{f}(\mathbf{x}_{k,i}) - \nabla \mathbf{f}(\mathbf{x}_{k,1}) \\ &= (j-1)\mathbf{y}_{k,1} + \sum_{i=2}^{j-1} \nabla \mathbf{f}(\mathbf{x}_{k,i}) - \nabla \mathbf{f}(\mathbf{x}_{k,1}). \end{aligned} \tag{27}$$

By the triangle inequality and Assumption 3.1, it follows that

$$\begin{aligned} \|\mathbf{x}_{k,j} - \mathbf{x}_{k,1}\|_2 &\leq \alpha(j-1)\|\mathbf{y}_{k,1}\|_2 + \alpha \sum_{i=1}^{j-1} \|\nabla \mathbf{f}(\mathbf{x}_{k,i}) - \nabla \mathbf{f}(\mathbf{x}_{k,1})\|_2 \\ &\leq \alpha(j-1)\|\mathbf{y}_{k,1}\|_2 + \alpha L \sum_{i=1}^{j-1} \|\mathbf{x}_{k,i} - \mathbf{x}_{k,1}\|_2. \end{aligned}$$

Now we apply induction to show (20) using the above inequality.

$$\text{For } j = 1, \qquad \|\mathbf{x}_{k,1} - \mathbf{x}_{k,1}\|_2 = 0 = 2\alpha(1-1)\|\mathbf{y}_{k,1}\|_2.$$

$$\begin{aligned} \text{For } j \geq 2, \qquad \|\mathbf{x}_{k,j} - \mathbf{x}_{k,1}\|_2 &\leq \alpha(j-1)\|\mathbf{y}_{k,1}\|_2 + \alpha L \sum_{i=1}^{j-1} \|\mathbf{x}_{k,i} - \mathbf{x}_{k,1}\|_2 \\ &\leq \alpha(j-1)\|\mathbf{y}_{k,1}\|_2 + 2\alpha^2 L \sum_{i=1}^{j-1} (i-1)\|\mathbf{y}_{k,1}\|_2 \\ &= \alpha(j-1)\|\mathbf{y}_{k,1}\|_2 + 2\alpha^2 L \|\mathbf{y}_{k,1}\|_2 \tfrac{(j-2)(j-1)}{2} \\ &= \alpha(j-1)\left(1 + \alpha L(j-2)\right)\|\mathbf{y}_{k,1}\|_2 \\ &\leq 2\alpha(j-1)\|\mathbf{y}_{k,1}\|_2, \end{aligned}$$

where the first equality uses the sum of $j-1$ natural numbers and the second to last inequality is due to $\alpha L \leq \frac{1}{n_g}$ and $j \leq n_g$.

By (20), the triangle inequality and $\|I_{nd} - \mathbf{I}\|_2 = 1$, it follows that

$$\begin{aligned} \|\mathbf{x}_{k,j} - \bar{\mathbf{x}}_{k,j}\|_2 &\leq \|\mathbf{x}_{k,j} - \mathbf{x}_{k,1} + \bar{\mathbf{x}}_{k,1} - \bar{\mathbf{x}}_{k,j}\|_2 + \|\mathbf{x}_{k,1} - \bar{\mathbf{x}}_{k,1}\|_2 \\ &\leq \|(I_{nd} - \mathbf{I})(\mathbf{x}_{k,j} - \mathbf{x}_{k,1})\|_2 + \|\mathbf{x}_{k,1} - \bar{\mathbf{x}}_{k,1}\|_2 \\ &\leq \|\mathbf{x}_{k,j} - \mathbf{x}_{k,1}\|_2 + \|\mathbf{x}_{k,1} - \bar{\mathbf{x}}_{k,1}\|_2. \end{aligned}$$

□

The two bounds in Lemma 3.3 ((20) and (21)) bound the deviation of the local decision variables from the start of the outer iteration, $\|\mathbf{x}_{k,j} - \mathbf{x}_{k,1}\|_2$, and the consensus error, $\|\mathbf{x}_{k,j} - \bar{\mathbf{x}}_{k,j}\|_2$, in inner iteration j, respectively. Combined with (19), these quantities are bounded as an $\mathcal{O}(\alpha j)$ multiple of the components of the error vector r_k. This property has two implications; (1) if one performs more inner iterations, i.e., increases n_g, the constant step size α needs to be reduced to reduce these quantities, (2) if an outer iterate is the optimal solution, the inner loop does not introduce any deviations in the iterates and maintains optimality.

We now establish the progression of error vector r_k under multiple communication and computation steps being performed every iteration in Algorithm 1.

Lemma 3.4 *Suppose Assumption 3.1 holds and $\alpha \leq \frac{1}{n_g L}$ in Algorithm 1. Then, for all $k \geq 0$,*

$$r_{k+1} \leq B(n_c,n_g) r_k, \quad \textit{where} \;\; B(n_c,n_g) = A(n_c,n_g) + \alpha L(n_g-1)E(n_c,n_g),$$

$$A(n_c,n_g) = \begin{bmatrix} (1-\alpha\mu)^{n_g} & \frac{\kappa}{\sqrt{n}}(1-(1-\alpha\mu)^{n_g}) & 0 \\ 0 & \beta_1^{n_c} & \alpha\left((n_g-1)\beta_1^{n_c} + \beta_2^{n_c}\right) \\ \sqrt{n}\alpha\beta_4^{n_c}L^2 & \beta_4^{n_c}L(\|\mathbf{Z}_1^{n_c} - I_{nd}\|_2 + \alpha L) & \beta_3^{n_c} + \alpha\beta_4^{n_c}L \end{bmatrix},$$
$$E(n_c,n_g) = \begin{bmatrix} \alpha L n_g & \frac{\alpha L n_g}{\sqrt{n}} & \frac{\alpha n_g}{\sqrt{n}} \\ \sqrt{n}\alpha L\delta_1(n_c,n_g) & \alpha L\delta_1(n_c,n_g) & \alpha\delta_1(n_c,n_g) \\ \sqrt{n}L\delta_2(n_c,n_g) & L\delta_2(n_c,n_g) & \delta_2(n_c,n_g) \end{bmatrix}, \tag{28}$$

and

$$\delta_1(n_c,n_g) = 2\beta_2^{n_c} + \beta_1^{n_c}(n_g-2),$$
$$\delta_2(n_c,n_g) = 2\left(\beta_4^{n_c}\|\mathbf{Z}_1^{n_c} - I_{nd}\|_2 + \frac{\beta_4^{n_c}}{n_g} + \beta_3^{n_c}\right). \tag{29}$$

Proof We first consider the optimization error of the average iterates $\bar{x}_{k,1}$. Similar to (10), we bound the optimization error as

$$\|\bar{x}_{k,j+1} - x^*\|_2 \leq (1-\alpha\mu)\|\bar{x}_{k,j} - x^*\|_2 + \tfrac{\alpha L}{\sqrt{n}}\|\mathbf{x}_{k,j} - \bar{\mathbf{x}}_{k,j}\|_2 \quad \forall\; 1 \leq j \leq n_g - 1,$$

where the above holds by using (26) (the generalization of (9)) and the error bound of gradient descent from [39, Theorem 2.1.14]. Next, we bound the optimization error in $\bar{x}_{k+1,1}$ with respect to $\mathbf{x}_{k,n_g}$ in a similar manner as,

$$\|\bar{x}_{k+1,1} - x^*\|_2 \leq (1-\alpha\mu)\|\bar{x}_{k,n_g} - x^*\|_2 + \tfrac{\alpha L}{\sqrt{n}}\|\mathbf{x}_{k,n_g} - \bar{\mathbf{x}}_{k,n_g}\|_2.$$

Recursively applying the above two bounds, by (21) it follows that,

$$\begin{aligned} \|\bar{x}_{k+1,1} - x^*\|_2 &\leq (1-\alpha\mu)^{n_g}\|\bar{x}_{k,1} - x^*\|_2 + \tfrac{\alpha L}{\sqrt{n}}\sum_{j=1}^{n_g}(1-\alpha\mu)^{n_g-j}\|\mathbf{x}_{k,1} - \bar{\mathbf{x}}_{k,1}\|_2 \\ &\quad + \tfrac{2\alpha^2 L}{\sqrt{n}}\sum_{j=1}^{n_g}(1-\alpha\mu)^{n_g-j}(j-1)\|\mathbf{y}_{k,1}\|_2 \\ &\leq (1-\alpha\mu)^{n_g}\|\bar{x}_{k,1} - x^*\|_2 + \tfrac{\kappa}{\sqrt{n}}\left[1-(1-\alpha\mu)^{n_g}\right]\|\mathbf{x}_{k,1} - \bar{\mathbf{x}}_{k,1}\|_2 \\ &\quad + \tfrac{\alpha^2 L}{\sqrt{n}} n_g(n_g-1)\|\mathbf{y}_{k,1}\|_2, \end{aligned}$$

where the last inequality is due to the fact that $(1-\alpha\mu)^{n_g-j} \leq 1 \, \forall j = 1,2,...,n_g$ due to $\alpha \leq \frac{1}{Ln_g}$, the coefficient of the second term is the sum of a geometric progression, and the coefficient of the third term is the sum of the first $n_g - 1$ natural numbers. By (19), we obtain the desired bound on the optimization error.

Next, we consider the consensus error in $\mathbf{x}_{k,1}$,

$$\begin{aligned}
&\mathbf{x}_{k+1,1} - \bar{\mathbf{x}}_{k+1,1} \\
&= (I_{nd} - \mathbf{I})\,\mathbf{x}_{k+1,1} = (I_{nd} - \mathbf{I})\left(\mathbf{Z}_1^{n_c}\mathbf{x}_{k,n_g} - \alpha \mathbf{Z}_2^{n_c}\mathbf{y}_{k,n_g}\right) \\
&= (I_{nd} - \mathbf{I})\left(\mathbf{Z}_1^{n_c}\left(\mathbf{x}_{k,1} - \alpha\sum_{j=1}^{n_g-1}\mathbf{y}_{k,j}\right) - \alpha\mathbf{Z}_2^{n_c}\mathbf{y}_{k,n_g}\right) \\
&= (I_{nd} - \mathbf{I})\left(\mathbf{Z}_1^{n_c}\mathbf{x}_{k,1} - \alpha\mathbf{Z}_1^{n_c}\left((n_g-1)\mathbf{y}_{k,1} + \sum_{j=2}^{n_g-1}\nabla\mathbf{f}(\mathbf{x}_{k,j}) - \nabla\mathbf{f}(\mathbf{x}_{k,1})\right)\right) \\
&\quad - \alpha\,(I_{nd} - \mathbf{I})\left(\mathbf{Z}_2^{n_c}(\mathbf{y}_{k,1} + \nabla\mathbf{f}(\mathbf{x}_{k,n_g}) - \nabla\mathbf{f}(\mathbf{x}_{k,1}))\right) \\
&= (\mathbf{Z}_1^{n_c} - \mathbf{I})(\mathbf{x}_{k,1} - \bar{\mathbf{x}}_{k,1}) - \alpha\left((n_g-1)(\mathbf{Z}_1^{n_c} - \mathbf{I}) + (\mathbf{Z}_2^{n_c} - \mathbf{I})\right)(\mathbf{y}_{k,1} - \bar{\mathbf{y}}_{k,1}) \\
&\quad - \alpha(\mathbf{Z}_2^{n_c} - \mathbf{I})\left(\nabla\mathbf{f}(\mathbf{x}_{k,n_g}) - \nabla\mathbf{f}(\mathbf{x}_{k,1})\right) - \alpha(\mathbf{Z}_1^{n_c} - \mathbf{I})\left(\sum_{j=2}^{n_g-1}\nabla\mathbf{f}(\mathbf{x}_{k,j}) - \nabla\mathbf{f}(\mathbf{x}_{k,1})\right)
\end{aligned}$$

where the second equality is a telescopic sum of the inner loop update ($\mathbf{x}_{k,i} = \mathbf{x}_{k,i-1} - \alpha\mathbf{y}_{k,i-1}$) from $i = 2$ to n_g and the third equality is due to (27). By the triangle inequality, Assumption 3.1 and (5),

$$\begin{aligned}
\|\mathbf{x}_{k+1,1} - \bar{\mathbf{x}}_{k+1,1}\|_2 \le \beta_1^{n_c}\|\mathbf{x}_{k,1} - \bar{\mathbf{x}}_{k,1}\|_2 + \alpha\left((n_g-1)\beta_1^{n_c} + \beta_2^{n_c}\right)\|\mathbf{y}_{k,1} - \bar{\mathbf{y}}_{k,1}\|_2 \\
+ \alpha\beta_2^{n_c}L\|\mathbf{x}_{k,n_g} - \mathbf{x}_{k,1}\|_2 + \alpha\beta_1^{n_c}L\sum_{j=2}^{n_g-1}\|\mathbf{x}_{k,j} - \mathbf{x}_{k,1}\|_2.
\end{aligned}$$

Adding $\alpha\beta_1^{n_c}L\|\mathbf{x}_{k,1} - \mathbf{x}_{k,1}\|_2 = 0$ to the right hand side and (20), it follows,

$$\begin{aligned}
\|\mathbf{x}_{k+1,1} - \bar{\mathbf{x}}_{k+1,1}\|_2 &\le \beta_1^{n_c}\|\mathbf{x}_{k,1} - \bar{\mathbf{x}}_{k,1}\|_2 + \alpha\left((n_g-1)\beta_1^{n_c} + \beta_2^{n_c}\right)\|\mathbf{y}_{k,1} - \bar{\mathbf{y}}_{k,1}\|_2 \\
&\quad + \alpha^2\beta_2^{n_c}L(2(n_g-1))\|y_{k,1}\|_2 + \alpha^2\beta_1^{n_c}L\sum_{j=1}^{n_g-1}2(j-1)\|y_{k,1}\|_2 \\
&= \beta_1^{n_c}\|\mathbf{x}_{k,1} - \bar{\mathbf{x}}_{k,1}\|_2 + \alpha\left((n_g-1)\beta_1^{n_c} + \beta_2^{n_c}\right)\|\mathbf{y}_{k,1} - \bar{\mathbf{y}}_{k,1}\|_2 \\
&\quad + 2\alpha^2L(n_g-1)\left(\beta_2^{n_c} + \beta_1^{n_c}\tfrac{(n_g-2)}{2}\right)\|\mathbf{y}_{k,1}\|_2.
\end{aligned}$$

The desired bound for the consensus error in $\mathbf{x}_{k,1}$ follows by using (19).

Finally, we consider the consensus error in $\mathbf{y}_{k,1}$. By (23),

$$\begin{aligned}
\mathbf{y}_{k+1,1} - \bar{\mathbf{y}}_{k+1,1} &= (I_{nd} - \mathbf{I})\,\mathbf{y}_{k+1,1} \\
&= (I_{nd} - \mathbf{I})\left(\mathbf{Z}_3^{n_c}\mathbf{y}_{k,1} + \mathbf{Z}_4^{n_c}\nabla\mathbf{f}(\mathbf{x}_{k+1,1}) - \mathbf{Z}_3^{n_c}\nabla\mathbf{f}(\mathbf{x}_{k,1})\right) \\
&\quad + (I_{nd} - \mathbf{I})\left(\mathbf{Z}_3^{n_c}\nabla\mathbf{f}(\mathbf{x}_{k,n_g}) - \mathbf{Z}_4^{n_c}\nabla\mathbf{f}(\mathbf{x}_{k,n_g})\right) \\
&= (\mathbf{Z}_3^{n_c} - \mathbf{I})(\mathbf{y}_{k,1} - \bar{\mathbf{y}}_{k,1}) + (\mathbf{Z}_4^{n_c} - \mathbf{I})\left(\nabla\mathbf{f}(\mathbf{x}_{k+1,1}) - \nabla\mathbf{f}(\mathbf{x}_{k,n_g})\right) \\
&\quad + (\mathbf{Z}_3^{n_c} - \mathbf{I})\left(\nabla\mathbf{f}(\mathbf{x}_{k,n_g}) - \nabla\mathbf{f}(\mathbf{x}_{k,1})\right).
\end{aligned}$$

By Assumption 3.1 and (5),

$$
\begin{aligned}
&\|\mathbf{y}_{k+1,1} - \bar{\mathbf{y}}_{k+1,1}\|_2 \\
&\le \beta_3^{n_c}\|\mathbf{y}_{k,1} - \bar{\mathbf{y}}_{k,1}\|_2 + \beta_4^{n_c} L\|\mathbf{x}_{k+1,1} - \mathbf{x}_{k,n_g}\|_2 + \beta_3^{n_c} L\|\mathbf{x}_{k,n_g} - \mathbf{x}_{k,1}\|_2 \\
&= \beta_3^{n_c}\|\mathbf{y}_{k,1} - \bar{\mathbf{y}}_{k,1}\|_2 + \beta_4^{n_c} L\|(\mathbf{Z}_1^{n_c} - I_{nd})(\mathbf{x}_{k,n_g} - \bar{\mathbf{x}}_{k,n_g}) - \alpha \mathbf{Z}_2^{n_c}\mathbf{y}_{k,n_g}\|_2 \\
&\quad + \beta_3^{n_c} L\|\mathbf{x}_{k,n_g} - \mathbf{x}_{k,1}\|_2 \\
&\le \beta_3^{n_c}\|\mathbf{y}_{k,1} - \bar{\mathbf{y}}_{k,1}\|_2 + \beta_4^{n_c} L\|\mathbf{Z}_1^{n_c} - I_{nd}\|_2\|\mathbf{x}_{k,n_g} - \bar{\mathbf{x}}_{k,n_g}\|_2 + \alpha\beta_4^{n_c} L\|\mathbf{Z}_2^{n_c}\|_2\|\mathbf{y}_{k,n_g}\|_2 \\
&\quad + \beta_3^{n_c} L\|\mathbf{x}_{k,n_g} - \mathbf{x}_{k,1}\|_2 \\
&= \beta_3^{n_c}\|\mathbf{y}_{k,1} - \bar{\mathbf{y}}_{k,1}\|_2 + \beta_4^{n_c} L\|\mathbf{Z}_1^{n_c} - I_{nd}\|_2\|\mathbf{x}_{k,n_g} - \bar{\mathbf{x}}_{k,n_g}\|_2 \\
&\quad + \alpha\beta_4^{n_c} L\|\mathbf{y}_{k,1} + \nabla \mathbf{f}(\mathbf{x}_{k,n_g}) - \nabla \mathbf{f}(\mathbf{x}_{k,1})\|_2 + \beta_3^{n_c} L\|\mathbf{x}_{k,n_g} - \mathbf{x}_{k,1}\|_2 \\
&\le \beta_3^{n_c}\|\mathbf{y}_{k,1} - \bar{\mathbf{y}}_{k,1}\|_2 + \beta_4^{n_c} L\|\mathbf{Z}_1^{n_c} - I_{nd}\|_2\|\mathbf{x}_{k,n_g} - \bar{\mathbf{x}}_{k,n_g}\|_2 \\
&\quad + \alpha\beta_4^{n_c} L\|\mathbf{y}_{k,1}\|_2 + \alpha\beta_4^{n_c} L^2\|\mathbf{x}_{k,n_g} - \mathbf{x}_{k,1}\|_2 + \beta_3^{n_c} L\|\mathbf{x}_{k,n_g} - \mathbf{x}_{k,1}\|_2
\end{aligned}
$$

where the first equality follows from $\mathbf{x}_{k+1,1} = \mathbf{Z}_1^{n_c}\mathbf{x}_{k,n_g} - \alpha\mathbf{Z}_2^{n_c}\mathbf{y}_{k,n_g}$ and $-(\mathbf{Z}_1^{n_c} - I_{nd})\bar{\mathbf{x}}_{k,n_g} = 0$, the second inequality is by the triangle inequality, the second equality follows by (22), and the last inequality is an application of triangle inequality and Assumption 3.1. By (20), (21) and $\alpha L \le \frac{1}{n_g}$, it follows,

$$
\begin{aligned}
&\|\mathbf{y}_{k+1,1} - \bar{\mathbf{y}}_{k+1,1}\|_2 \\
\le& \beta_3^{n_c}\|\mathbf{y}_{k,1} - \bar{\mathbf{y}}_{k,1}\|_2 + \beta_4^{n_c} L\|\mathbf{Z}_1^{n_c} - I_{nd}\|_2\|\mathbf{x}_{k,1} - \bar{\mathbf{x}}_{k,1}\|_2 \\
&+ \left(\alpha\beta_4^{n_c} L + 2\alpha(n_g - 1)L\left(\beta_4^{n_c}\|\mathbf{Z}_1^{n_c} - I_{nd}\|_2 + \frac{\beta_4^{n_c}}{n_g} + \beta_3^{n_c}\right)\right)\|\mathbf{y}_{k,1}\|_2.
\end{aligned}
$$

Substituting (19) yields the desired bound for the consensus error in $\mathbf{y}_{k,1}$. □

Lemma 3.4 quantifies the progression of error vector r_k using the matrix $B(n_c, n_g)$, similar to Lemma 3.1 but now allowing for multiple computation steps. Notice that when $n_g = 1$, Lemma 3.4 reduces to Lemma 3.1, making it a special case of this analysis. We split the matrix $B(n_c, n_g)$ into the matrices $A(n_c, n_g)$ and $E(n_c, n_g)$. The latter matrix is characterized by the terms $\delta_1(n_c, n_g)$ and $\delta_2(n_c, n_g)$. We now define the explicit form of $B(n_c, n_g)$ for the methods defined in Table 1.

Corollary 3.4 *Suppose the conditions of Lemma 3.4 are satisfied. Then, the matrices $A(n_c, n_g)$ for the methods described in Table 1 are defined as:*

$$
\begin{aligned}
\texttt{GTA-1:}\quad A_1(n_c,n_g) &= \begin{bmatrix} (1-\alpha\mu)^{n_g} & \frac{\kappa}{\sqrt{n}}(1-(1-\alpha\mu)^{n_g}) & 0 \\ 0 & \beta^{n_c} & \alpha\left((n_g-1)\beta^{n_c}+1\right) \\ \sqrt{n}\alpha L^2 & L(2+\alpha L) & \beta^{n_c}+\alpha L \end{bmatrix}, \\
\texttt{GTA-2:}\quad A_2(n_c,n_g) &= \begin{bmatrix} (1-\alpha\mu)^{n_g} & \frac{\kappa}{\sqrt{n}}(1-(1-\alpha\mu)^{n_g}) & 0 \\ 0 & \beta^{n_c} & \alpha\beta^{n_c} n_g \\ \sqrt{n}\alpha L^2 & L(2+\alpha L) & \beta^{n_c}+\alpha L \end{bmatrix}, \\
\texttt{GTA-3:}\quad A_3(n_c,n_g) &= \begin{bmatrix} (1-\alpha\mu)^{n_g} & \frac{\kappa}{\sqrt{n}}(1-(1-\alpha\mu)^{n_g}) & 0 \\ 0 & \beta^{n_c} & \alpha\beta^{n_c} n_g \\ \sqrt{n}\alpha\beta^{n_c} L^2 & \beta^{n_c} L(2+\alpha L) & \beta^{n_c}(1+\alpha L) \end{bmatrix}.
\end{aligned}
\tag{30}
$$

The matrix $E(n_c, n_g)$ for the methods described in Table 1 is defined using the error terms $(\delta_1(n_c, n_g)$ and $\delta_2(n_c, n_g))$. The error terms for the methods described in Table 1 are defined in Table 2.

Table 2: Error terms ($\delta_1(n_c, n_g)$ and $\delta_2(n_c, n_g)$) for `GTA-1`, `GTA-2` and `GTA-3`

Method	$\delta_1(n_c, n_g)$	$\delta_2(n_c, n_g)$
`GTA-1`	$2 + \beta^{n_c}(n_g - 2)$	$2\left(2 + \frac{1}{n_g} + \beta^{n_c}\right)$
`GTA-2`	$n_g \beta^{n_c}$	$2\left(2 + \frac{1}{n_g} + \beta^{n_c}\right)$
`GTA-3`	$n_g \beta^{n_c}$	$2\beta^{n_c}\left(3 + \frac{1}{n_g}\right)$

Proof Substituting the matrix values for each method in (28) and bounding $\|\mathbf{Z}_1^{n_c} - I_{nd}\|_2 \leq 2$ gives the desired result. □

Corollary 3.4 presents the explicit form of the matrices $B_i(n_c, n_g) = A_i(n_c, n_g) + \alpha L(n_g - 1)E_i(n_c, n_g)$ for $i = 1, 2, 3$, for each of the methods in Table 1. The convergence properties of `GTA` can be analyzed using the spectral radius of $B(n_c, n_g)$. We now qualitatively establish the effect of the number of communication steps n_c on $\rho(B(n_c, n_g))$ and a relative ordering for $\rho(B_1(n_c, n_g))$, $\rho(B_2(n_c, n_g))$ and $\rho(B_3(n_c, n_g))$.

Theorem 3.5 *Suppose Assumption 3.1 holds. If $\alpha \leq \frac{1}{Ln_g}$ in Algorithm 1, then as n_c increases, $\rho(B(n_c, n_g))$ decreases where $B(n_c, n_g)$ is defined in Lemma 3.4. Thus, as n_c increases, $\rho(B_i(n_c, n_g))$ decreases for all $i = 1, 2, 3$ defined in Corollary 3.4. Moreover, if all three methods defined in Table 1 (`GTA-1`, `GTA-2` and `GTA-3`) employ the same step size,*

$$\rho(B_1(n_c, n_g)) \geq \rho(B_2(n_c, n_g)) \geq \rho(B_3(n_c, n_g)).$$

Proof Note that $A(n_c, n_g) \geq 0$ and $E(n_c, n_g) \geq 0$, thus $B(n_c, n_g) \geq 0$. Also, $A(n_c, n_g) \geq A(n_c+1, n_g)$, $\delta_1(n_c, n_g) \geq \delta_1(n_c+1, n_g)$, $\delta_2(n_c, n_g) \geq \delta_2(n_c+1, n_g)$, thus $E(n_c, n_g) \geq E(n_c+1, n_g)$ and $B(n_c, n_g) \geq B(n_c+1, n_g)$. By [20, Corollary 8.1.19], it follows that $\rho(A(n_c, n_g)) \geq \rho(A(n_c + 1, n_g))$, $\rho(E(n_c, n_g)) \geq \rho(E(n_c + 1, n_g))$ and $\rho(B(n_c, n_g)) \geq \rho(B(n_c + 1, n_g))$. The same argument is applicable for $B_1(n_c, n_g)$, $B_2(n_c, n_g)$ and $B_3(n_c, n_g)$. Now, observe that $B_1(n_c, n_g) \geq B_2(n_c, n_g) \geq B_3(n_c, n_g) \geq 0$ when the same step size is employed. Thus, again by [20, Corollary 8.1.19], it follows that $\rho(B_1(n_c, n_g)) \geq \rho(B_2(n_c, n_g)) \geq \rho(B_3(n_c, n_g))$. □

The effect of the number of computation steps n_g on $\rho(B(n_c, n_g))$ is not as clear as the effect of the number of communication steps n_c. Increasing n_g increases all elements of the matrix $\alpha L(n_g - 1)E(n_c, n_g)$, while $(1 - \alpha\mu)^{n_g}$ in the matrix $A(n_c, n_g)$ decreases since $\alpha \leq \frac{1}{Ln_g}$. Thus, the effect of n_g on $\rho(B(n_c, n_g))$ is not monotonic.

We now derive conditions for establishing a linear rate of convergence for Algorithm 1 with multiple communication and computation steps every iteration in terms of network parameters $(\beta_1, \beta_2, \beta_3, \beta_4)$ and objective function parameters $(L, \mu, \kappa = \frac{L}{\mu})$.

Theorem 3.6 *Suppose Assumption 3.1 holds and a finite number of computation steps are performed at each outer iteration of Algorithm 1 (i.e., $1 \leq n_g < \infty$). If the matrix $B(n_c, n_g)$ is irreducible, $\beta_1, \beta_3 < 1$ and*

$$\alpha < \min\left\{\frac{1}{n_g L}, \frac{\mu}{(2L^2+\mu^2)(n_g-1)}, \frac{1}{2L}\sqrt{\frac{3(1-\beta_1^{n_c})}{\delta_1(n_c,n_g)(n_g-1)}}, \frac{3(1-\beta_3^{n_c})}{4L(\beta_4^{n_c}+\delta_2(n_c,n_g)(n_g-1))}, \frac{-b_2+\sqrt{b_2^2+4b_1b_3}}{2b_1}\right\} \tag{31}$$

where

$$\begin{aligned} b_1 =&\tfrac{\mu L^2 n_g}{2}\left[(n_g-1)\left(\beta_1^{n_c}+\delta_1(n_c,n_g)\right)+\beta_2^{n_c}\right]\left[\beta_4^{n_c}+(n_g-1)\delta_2(n_c,n_g)\right] \\ &+L^3 n_g(n_g-1)\left[\delta_1(n_c,n_g)\left(\tfrac{1-\beta_3^{n_c}}{4}\right)+\left(\beta_4^{n_c}+(n_g-1)\delta_2(n_c,n_g)\right)\left(\tfrac{1-\beta_1^{n_c}}{4}\right)\right] \\ &+L^2(n_g-1)^2\left[L\delta_1(n_c,n_g)\left(3\beta_4^{n_c}+(n_g-1)\delta_2(n_c,n_g)\right)+\delta_1(n_c,n_g)\left(\tfrac{1-\beta_3^{n_c}}{4}\right)\right] \\ &+L^2[\beta_4^{n_c}+(n_g-1)\delta_2(n_c,n_g)]\left[Ln_g+(n_g-1)\right]\left[(n_g-1)\left(\beta_1^{n_c}+\delta_1(n_c,n_g)\right)+\beta_2^{n_c}\right] \\ b_2 =&\mu n_g\beta_4^{n_c}L\left((n_g-1)(\beta_1^{n_c}+\delta_1(n_c,n_g))+\beta_2^{n_c}\right), \text{ and } b_3=\tfrac{\mu n_g}{2}\left(\tfrac{1-\beta_1^{n_c}}{4}\right)\left(\tfrac{1-\beta_3^{n_c}}{4}\right) \end{aligned}$$

and $\delta_1(n_c,n_g)$ *and* $\delta_2(n_c,n_g)$ *are defined in* (29)*, then, for all* $\epsilon>0$ *there exists a constant* $C_\epsilon>0$ *such that, for all* $k\geq 0$,

$$\|r_k\|_2 \leq C_\epsilon(\rho(B(n_c,n_g))+\epsilon)^k\|r_0\|_2, \quad \text{where } \rho(B(n_c,n_g))<1.$$

Proof By the binomial expansion of $(1-\alpha\mu)^{n_g}$ and the condition that $\alpha\leq\frac{1}{Ln_g}$, it follows that $1-\alpha\mu n_g\leq(1-\alpha\mu)^{n_g}\leq 1-\alpha\mu n_g+\alpha^2\mu^2\frac{n_g(n_g-1)}{2}$. Following a similar approach to [43, Theorem 2], since the step size satisfies (31), the first, second and third diagonal terms of $B(n_c,n_g)$ can be upper bounded as

$$\begin{aligned} (1-\alpha\mu)^{n_g}+\alpha^2L^2n_g(n_g-1)\leq 1-\alpha\mu n_g+\alpha^2(L^2+\tfrac{\mu^2}{2})n_g(n_g-1)&<1-\tfrac{\alpha\mu n_g}{2}, \\ \beta_1^{n_c}+\alpha^2L^2(n_g-1)\delta_1(n_c,n_g)&<\tfrac{3+\beta_1^{n_c}}{4}, \\ \beta_3^{n_c}+\alpha\beta_4^{n_c}L+\alpha L(n_g-1)\delta_2(n_c,n_g)&<\tfrac{3+\beta_3^{n_c}}{4}. \end{aligned}$$

With the above bounds, $(1-\alpha\mu)^{n_g}\geq 1-\alpha\mu n_g$ and $\|\mathbf{Z}_1^{n_c}-I_{nd}\|\leq 2$, we construct the 3×3 matrix $\tilde{B}(n_c,n_g)$ that has entries $\tilde{b}_{ij}$ defined as follows:

$$\begin{aligned} &\tilde{b}_{11}=1-\tfrac{\alpha\mu n_g}{2}, \quad \tilde{b}_{12}=\tfrac{\alpha Ln_g}{\sqrt{n}}\left(1+\alpha L(n_g-1)\right), \quad \tilde{b}_{13}=\tfrac{\alpha^2Ln_g(n_g-1)}{\sqrt{n}} \\ &\tilde{b}_{21}=\sqrt{n}\alpha^2L^2(n_g-1)\delta_1(n_c,n_g), \quad \tilde{b}_{22}=\tfrac{3+\beta_1^{n_c}}{4}, \\ &\tilde{b}_{23}=\alpha\left((n_g-1)(\beta_1^{n_c}+\alpha L\delta_1(n_c,n_g))+\beta_2^{n_c}\right), \\ &\tilde{b}_{31}=\sqrt{n}\alpha L^2\left(\beta_4^{n_c}+(n_g-1)\delta_2(n_c,n_g)\right), \\ &\tilde{b}_{32}=\beta_4^{n_c}L(2+\alpha L)+\alpha L^2(n_g-1)\delta_2(n_c,n_g), \quad \tilde{b}_{33}=\tfrac{3+\beta_3^{n_c}}{4}, \end{aligned}$$

such that $0\leq B(n_c,n_g)\leq\tilde{B}(n_c,n_g)$ and by [20, Corollary 8.1.19], $\rho(B(n_c,n_g))\leq\rho(\tilde{B}(n_c,n_g))$. Following [42, Lemma 5] derived from the Perron-Forbenius Theorem [20, Theorem 8.4.4] for a 3×3 matrix, when the matrix $\tilde{B}(n_c,n_g)$ is nonnegative and irreducible, it is sufficient to show that the diagonal elements of $\tilde{B}(n_c,n_g)$ are less than one and $\det(I_3-\tilde{B}(n_c,n_g))>0$ in order to guarantee $\rho(\tilde{B}(n_c,n_g))<1$ which suffices to show $\rho(B(n_c,n_g))<1$.

Consider the diagonal elements of the matrix $\tilde{B}(n_c,n_g)$. The first element is $1-\frac{\alpha\mu n_g}{2}\leq 1-\frac{\mu}{2L}<1$ by (31). The second element is $\frac{3+\beta_1^{n_c}}{4}<1$ as $\beta_1<1$. Finally the third element is

$\frac{3+\beta_3^{n_c}}{4} < 1$ as $\beta_3 < 1$. Next, let us consider,

$$\begin{aligned}
&\det(I_3 - \tilde{B}(n_c, n_g)) \\
=&\tfrac{\alpha\mu n_g}{2}\left(\tfrac{1-\beta_1^{n_c}}{4}\right)\left(\tfrac{1-\beta_3^{n_c}}{4}\right) - \alpha^3 L^3 n_g(n_g-1)\left[\beta_4^{n_c}\left(\tfrac{1-\beta_1^{n_c}}{4}\right) + \delta_1(n_c,n_g)\left(\tfrac{1-\beta_3^{n_c}}{4}\right)\right] \\
&-\tfrac{\alpha^2\mu L n_g}{2}\left[(n_g-1)(\beta_1^{n_c} + \alpha L\delta_1(n_c,n_g)) + \beta_2^{n_c}\right]\left[\beta_4^{n_c}(2+\alpha L) + \alpha L(n_g-1)\delta_2(n_c,n_g)\right] \\
&-\alpha^4 L^4 n_g(n_g-1)^2\delta_1(n_c,n_g)\left[2\beta_4^{n_c} + \alpha L\left(\beta_4^{n_c} + (n_g-1)\delta_2(n_c,n_g)\right)\right] \\
&-\alpha^3 L^3 n_g\left(1+\alpha(n_g-1)\right)\left[\beta_4^{n_c} + (n_g-1)\delta_2(n_c,n_g)\right]\left[(n_g-1)(\beta_1^{n_c} + \alpha L\delta_1(n_c,n_g)) + \beta_2^{n_c}\right] \\
&-\alpha^3 L^3 n_g(n_g-1)^2\left[\delta_2(n_c,n_g)\left(\tfrac{1-\beta_1^{n_c}}{4}\right) + \alpha\delta_1(n_c,n_g)\left(\tfrac{1-\beta_3^{n_c}}{4}\right)\right] \\
\geq&\alpha(-b_1\alpha^2 - b_2\alpha + b_3) = -b_1\alpha(\alpha-\alpha_l)(\alpha-\alpha_u)
\end{aligned}$$

where the inequality is due to $\alpha L n_g \leq 1$ and thus $\alpha L \leq 1$ as $n_g \geq 1$, and

$$\alpha_l = \tfrac{-b_2-\sqrt{b_2^2+4b_1b_3}}{2b_1} \quad \text{and} \quad \alpha_u = \tfrac{-b_2+\sqrt{b_2^2+4b_1b_3}}{2b_1}.$$

Observe that $\alpha_l < 0 < \alpha_u$ since $b_1, b_2, b_3 \geq 0$. From (31), we have $0 < \alpha < \alpha_u$. Therefore, $\det(I_3 - \tilde{B}(n_c, n_g)) > 0$, which combined with the fact that the diagonal elements of the matrix are less than 1, implies $\rho(B(n_c, n_g)) \leq \rho(\tilde{B}(n_c, n_g)) < 1$.

Finally, we bound the norm of error vector $\|r_k\|_2$ by telescoping $r_{i+1} \leq B(n_c, n_g) r_i$ from $i = 0$ to $k-1$ and triangle inequality as

$$\|r_k\|_2 \leq \|B(n_c, n_g)^k\|_2 \|r_0\|_2.$$

From [20, Corollary 5.6.13], we can bound $\|B(n_c, n_g)^k\|_2 \leq C_\epsilon(\rho(B(n_c, n_g)) + \epsilon)^k$ where $\epsilon > 0$ and C_ϵ is a positive constant depending on $B(n_c, n_g)$ and ϵ. □

Remark 3.1 Linear convergence can be established for the iterates generated by Algorithm 1 by treating inner iterations as a special case of of time-varying networks and following the analysis techniques in [36, 37, 40]. Such analysis would ensure descent in each inner iteration and require that the step size satisfy a pessimistic $\mathcal{O}(\frac{1}{n_g^2})$ condition. Our analysis takes a different approach; we quantify the error across outer iterations, and as a result, the condition on the step size is less pessimistic, i.e., $\mathcal{O}(\frac{1}{n_g})$. That said, the analysis involves a complicated error recursion that makes it difficult to explicitly quantify the rate of convergence of Algorithm 1. In our follow-up work [5], we present an analogous randomized algorithm that explicitly quantifies the complexity in terms of number of communication and computation steps.

Similar to Theorem 3.4, the only constraint Theorem 3.6 imposes on the system is $\beta_1, \beta_3 < 1$. This implies the communication matrices $\mathbf{W}_1$ and $\mathbf{W}_3$ must represent connected networks (not necessarily the same network) even when multiple communication and multiple computation steps are performed. Theorem 3.6 does not impose any restrictions on the relation among $\mathbf{W}_1$, $\mathbf{W}_2$, $\mathbf{W}_3$ and $\mathbf{W}_4$. Thus, it allows for more flexibility than the structures considered in the literature even when multiple communication and multiple computation steps are performed. Theorem 3.6 uses a relaxation of the original matrix $B(n_c, n_g)$ to provide a more pessimistic step size condition than required. But observe, when $n_g = 1$, (31) recovers the $\mathcal{O}(L^{-1}\kappa^{-0.5})$ step size condition of Theorem 3.3, suggesting it might not be very pessimistic.

Based on Theorem 3.6, the step size conditions for methods described in Table 1 can be derived. We omit these conditions as they are complex and do not offer any additional insights. We also omit the counterpart to Theorem 3.4 as the matrix $B(n_c, n_g)$ is now a dense matrix, thus any such bounds are again highly complex and do not offer strong insights into the effects of communication and computation on the convergence rate. If $B(n_c, n_g)$ is a reducible matrix, the analysis for the progression of r_k can be further simplified from Lemma 3.4. The analysis for this case is presented in Subsection 3.3 with the examples of `GTA-2` and `GTA-3` when $\mathbf{W} = \frac{1_n 1_n^T}{n}$, i.e., $\beta = 0$.

3.3 Fully connected network

In this section, we analyze the methods defined in Table 1 under a fully connected network. While showing linear convergence of `GTA` in Theorem 3.6, we assume $B(n_c, n_g)$ is an irreducible matrix. When the network is fully connected, i.e., $\mathbf{W} = \frac{1_n 1_n^T}{n}$ and $\beta = 0$, the assumption does not hold for `GTA-2` and `GTA-3` as the matrices $B_2(n_c, n_g)$ and $B_3(n_c, n_g)$ defined by Corollary 3.4 are reducible. For `GTA-1`, such an issue does not arise as $A_1(n_c, n_g)$ defined in Corollary 3.4 is irreducible for all $\beta \in [0, 1]$. Thus, we now present sufficient conditions for linear rate of convergence and the convergence rate for `GTA-2` and `GTA-3` for the special case of fully connected networks.

Theorem 3.7 *Suppose Assumption Theorem 3.1 holds, $\mathbf{W} = \frac{1_n 1_n^T}{n}$ and a finite number of computation steps are performed each outer iteration of `GTA-3` defined in Table 1 (i.e., $1 \leq n_g < \infty$). If $\alpha < \min\left\{\frac{\mu}{(2L^2+\mu^2)(n_g-1)}, \frac{1}{Ln_g}\right\}$, then for all $k \geq 0$,*

$$\|\bar{x}_{k+1,1} - x^*\|_2 \leq \left((1-\alpha\mu)^{n_g} + \alpha^2 L^2 n_g (n_g - 1)\right) \|\bar{x}_{k,1} - x^*\|_2.$$

Moreover, suppose the number of computation steps performed each outer iteration of `GTA-2` and `GTA-3` defined in Table 1 is set to one (i.e., $n_g = 1$). If $\alpha \leq \frac{1}{L}$, then for both the methods, for all $k \geq 0$,

$$\|\bar{x}_{k+1,1} - x^*\|_2 \leq (1-\alpha\mu)\|\bar{x}_{k,1} - x^*\|_2.$$

Proof When we substitute $\beta = 0$ in Corollary 3.4 as $\alpha < \frac{1}{n_g L}$, the matrices $B_2(n_c, n_g)$ and $B_3(n_c, n_g)$ now have rows of zeros that make them reducible. Thus, we reduce these matrices by ignoring the error terms corresponding to the row of zeros. This yields the following systems for the progression of errors in these methods,

$$\texttt{GTA-2:}\ \tilde{r}_{k+1} \leq \begin{bmatrix} (1-\alpha\mu)^{n_g} + \alpha^2 L^2 n_g (n_g-1) & \frac{\alpha^2 L n_g (n_g-1)}{\sqrt{n}} \\ \sqrt{n}\alpha L^2 \tilde{\delta}(n_c, n_g) & \alpha L \tilde{\delta}(n_c, n_g) \end{bmatrix} \tilde{r}_k, \tag{32}$$

$$\texttt{GTA-3:}\ \|\bar{x}_{k+1,1} - x^*\|_2 \leq \left((1-\alpha\mu)^{n_g} + \alpha^2 L^2 n_g (n_g-1)\right) \|\bar{x}_{k,1} - x^*\|_2, \tag{33}$$

where $\tilde{\delta}(n_c, n_g) = 1 + 2(n_g - 1)\left(2 + \frac{1}{n_g}\right)$ and $\tilde{r}_k = \begin{bmatrix} \|\bar{x}_{k,1} - x^*\|_2 \\ \|\mathbf{y}_{k,1} - \bar{\mathbf{y}}_{k,1}\|_2 \end{bmatrix}$.

By $\alpha < \frac{\mu}{(2L^2+\mu^2)(n_g-1)}$ and $(1-\alpha\mu)^{n_g} \leq 1 - \alpha\mu n_g + \alpha^2\mu^2 \frac{n_g(n_g-1)}{2}$ from Theorem 3.6,

$$(1-\alpha\mu)^{n_g} + \alpha^2 L^2 n_g(n_g-1) \leq 1 - \alpha\mu n_g + \alpha^2\left(L^2 + \frac{\mu^2}{2}\right) n_g (n_g-1) < 1,$$

and thus the result for `GTA-3` follows. When the number of computation steps performed each outer iteration is set to one, i.e., $n_g = 1$, the result for `GTA-3` follows by substituting $n_g = 1$ in (33), where $1 - \alpha\mu < 1$ as $\alpha \leq \frac{1}{L}$. Substituting $n_g = 1$ in (32) for `GTA-2` yields, $\tilde{r}_{k+1} \leq \begin{bmatrix} 1-\alpha\mu & 0 \\ \sqrt{n}\alpha L^2 & \alpha L \end{bmatrix} \tilde{r}_k$, where the bound on optimization error is independent of the consensus error in $\mathbf{y}_{k,1}$. Thus, we obtain $\|\bar{x}_{k+1,1} - x^*\|_2 \leq (1-\alpha\mu)\|\bar{x}_{k,1} - x^*\|_2$ for `GTA-2`. □

By Theorem 3.7 if the network is fully connected and a single computation step is performed, i.e., $n_g = 1$, `GTA-2` and `GTA-3` display gradient descent performance. For `GTA-2`, when the network is fully connected and $n_g > 1$, the convergence rate can be expressed as the spectral radius of the 2×2 matrix in (32).

4 Numerical Experiments

In this section, we illustrate the empirical performance of the methods defined in Table 1 using Python implementations[2]. The aim of this section is to show, over multiple problems, that different communication strategies and the balance between communication and computation steps can substantially effect the algorithm's performance. Specifically, we establish the relative performance of the methods defined in Table 1 and illustrate the benefits of the flexibility in terms of communication and computation steps.

We present results on two problems: (1) a synthetic strongly convex quadratic problem (Subsection 4.1); and, (2) binary classification logistic regression problems over the mushroom and australian datasets [16] (Subsection 4.2). We investigated two network structures (different mixing matrix $\mathbf{W}$) with $n = 16$ nodes: (1) a connected cyclic network ($\beta = 0.992$) where all nodes have two neighbours; and, (2) a connected star network ($\beta = 0.95$) where all nodes are connected to a single central node. Both networks have low connectivity (i.e., high β). We should note that the performance of Algorithm 1 with multiple communication steps is equivalent to the performance over a network with higher connectivity (i.e., lower β).

The methods defined in Table 1 are denoted as $\texttt{GTA}-i(n_c, n_g)$, $i = 1, 2, 3$, where n_c and n_g are the number of communication and computation steps, respectively. We tested 5 values of n_c and n_g for each of the methods; $n_c \in \{1, 5, 10, 50, 100\}$ and $n_g \in \{1, 5, 20, 50, 100\}$. We compared the performance of popular gradient tracking methods, which are special cases of our generalized framework. The step sizes were tuned over the set $\{2^{-t} | t = 0, 1, 2, .., 20\}$ for all algorithms and problems, and the initial iterates for all algorithms, problems and nodes were set to the zero vector (i.e., $\mathbf{x}_k = \mathbf{0}$). The performance of the methods was measured in terms of the optimization error ($\|\bar{x}_k - x^*\|_2$) and the consensus error ($\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|_2$). We do not report the consensus error in the auxiliary variable $\mathbf{y}_k$ ($\|\mathbf{y}_k - \bar{\mathbf{y}}_k\|_2$) as this measure does not provide any significant additional insights about the performance of the algorithms. The optimal solution x^* for quadratic problem was obtained analytically and for the logistic regression problems was obtained by running gradient descent in the centralized setting to high accuracy, i.e., $\|\nabla f(x^*)\|_2 \leq 10^{-12}$.

4.1 Quadratic Problems

We first consider quadratic problems

$$f(x) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{2} x^T Q_i x + b_i^T x,$$

where $Q_i \in \mathbb{R}^{10 \times 10}$, $Q_i \succ 0$ and $b_i \in \mathbb{R}^{10}$ is the local information at each node $i \in \{1, 2, .., n\}$, and $n = 16$. Each local problem is strongly convex and was generated using the procedure described in [34], with global condition number $\kappa \approx 10^4$.

[2] Our code will be made publicly available upon publication of the manuscript. Github repository: `https://github.com/SANDOPT/Gradient-Tracking-Algorithmic-Framework.git`. Moreover, additional extensive numerical results can be found in the same repository.

Figs. 1 and 2 show the performance of `GTA-1`, `GTA-2` and `GTA-3` over a cyclic network and a star network, respectively. Our first observation, from the iteration plots in both the figures, is that the optimization error and consensus error converge at a linear rate for all methods, matching the theoretical results of Section 3. Moreover, improvements in the rates of convergence of all methods are observed as a result of the flexibility in terms of the number of communication and computation steps. Specifically, the consensus error is improved (and on par optimization error) when multiple communication steps with single computation step are performed (see `GTA-i`(1, 1) vs. `GTA-i`(n_c, 1) lines), and the optimization error is improved (and on par consensus error) when multiple computation steps with same number of communication steps are performed (see `GTA-i`(n_c, 1) vs. `GTA-i`(n_c, n_g) lines). These observations match the theory presented in Section 3.2. That being said, these improvements come at a higher cost in terms of total communication or computation steps, respectively, and an optimal choice of (n_c, n_g) depends on the exact cost structure that combines the complexity of both these steps; see e.g., [6]. Finally, we also observe that `GTA-2` and `GTA-3` outperform `GTA-1` in terms of optimization error and achieve similar consensus error. The performance of `GTA-2` and `GTA-3` is very similar for this problem, we suspect the reason for this behavior is due to the large β and the high condition number ($\kappa \approx 10^4$) that dominate the rate constant; see Corollary 3.3.

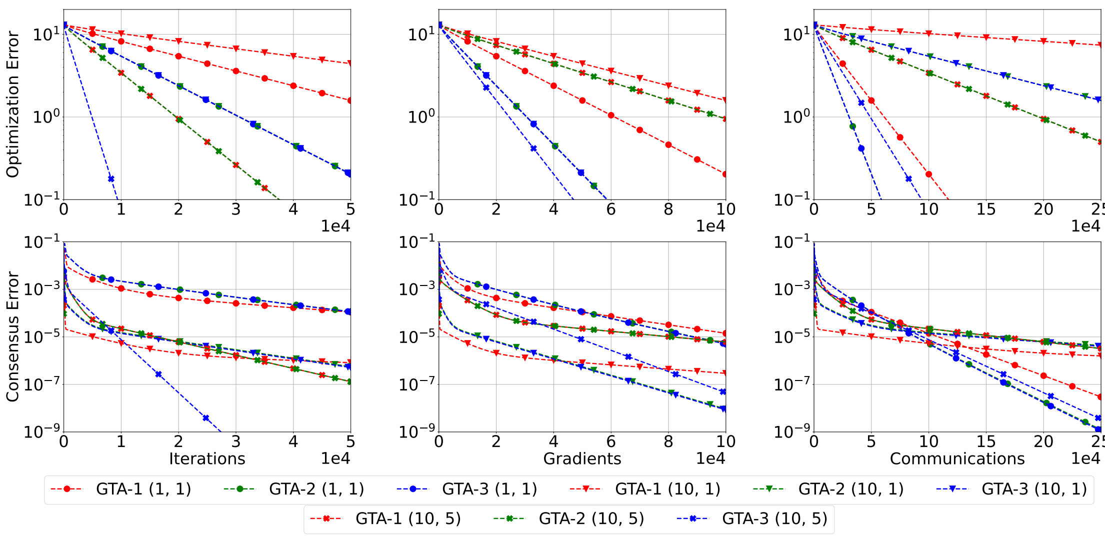

Fig. 1: Optimization Error ($\|\bar{x}_k - x^*\|_2$) and Consensus Error ($\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|_2$) of `GTA-1`, `GTA-2` and `GTA-3` with respect to number of iterations, communications and gradient evaluations for a synthetic quadratic problem ($n = 16$, $d = 10$, $\kappa = 10^4$) over a cyclic network ($\beta = 0.992$).

4.2 Binary Classification Logistic Regression

Next, we consider ℓ_2-regularized binary classification logistic regression problems of the form

$$f(x) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{n_i} \log(1 + e^{-b_i^T A_i x}) + \frac{1}{n_i}\|x\|_2^2,$$

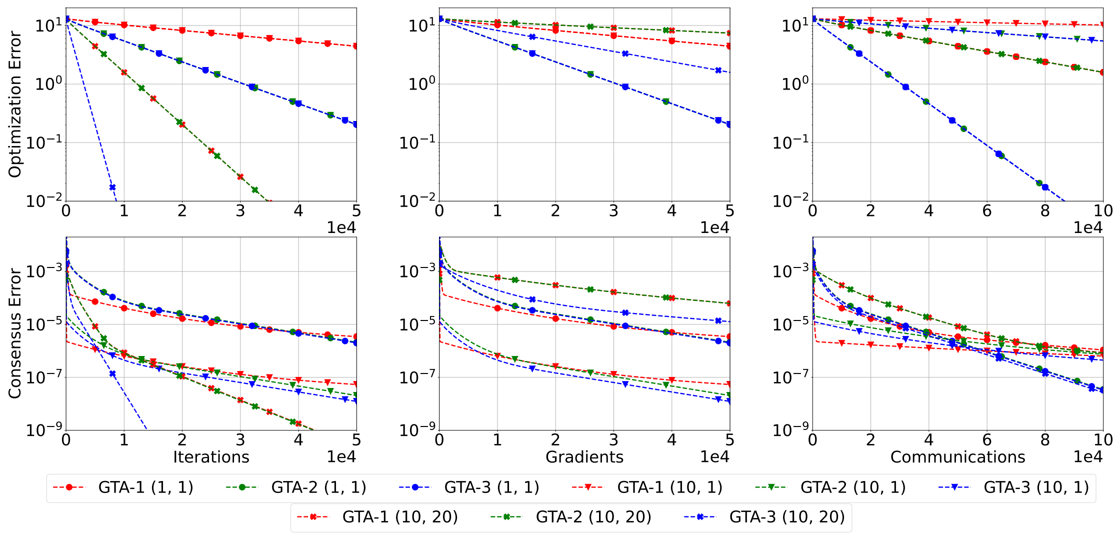

Fig. 2: Optimization Error ($\|\bar{x}_k - x^*\|_2$) and Consensus Error ($\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|_2$) of `GTA-1`, `GTA-2` and `GTA-3` with respect to number of iterations, communications and gradient evaluations for a synthetic quadratic problem ($n = 16$, $d = 10$, $\kappa = 10^4$) over star network($\beta = 0.95$).

where each node $i \in \{1, 2, .., n\}$ has a portion of data samples $A_i \in \mathbb{R}^{n_i \times d}$ and corresponding labels $b_i \in \{0, 1\}^{n_i}$. Experiments were performed over the mushroom dataset ($n = 16$, $d = 117$, $\sum_{i=1}^{n} n_i = 8124$) and the australian dataset ($n = 16$, $d = 41$, $\sum_{i=1}^{n} n_i = 690$) [16].

Figs. 3 and 4 show the performance of `GTA-1`, `GTA-2` and `GTA-3` over a cyclic network ($\beta = 0.992$) for the mushroom dataset and a star network for the australian dataset ($\beta = 0.95$), respectively. Similar observations to those made for the quadratic problem with respect to the effect of performing multiple communication and computation steps can also be made for these problems. Additionally, we observe that `GTA-3` outperforms `GTA-2` on these problems. We should note that although `GTA-3` performs the best within these experiments, it also brings certain implementation constraints; see Section 2.

5 Final Remarks

In this paper, we have proposed a framework that unifies and generalizes communication strategies in gradient tracking methods with flexibility in the number of communication and computation steps performed at every iteration. We have established convergence guarantees for the proposed gradient tracking framework. Specifically, we have shown linear convergence for the general framework and the special cases of gradient tracking methods. Moreover, we have shown the positive influence of performing multiple communication steps at every iteration on the convergence rate and provide results that allow for the direct comparison of popular gradient tracking methods. Our experiments on quadratic and logistic regression problems illustrate the effects of different communication strategies and the benefits of the flexibility in terms of iterations and number of communication and computation steps. The advantages of the proposed framework can be further realized when the actual cost, i.e., a combination of the complexity of both communication and computation steps that is application specific, is considered. The framework is extendable to diverse decentralized optimization settings, such as non-convex problems,

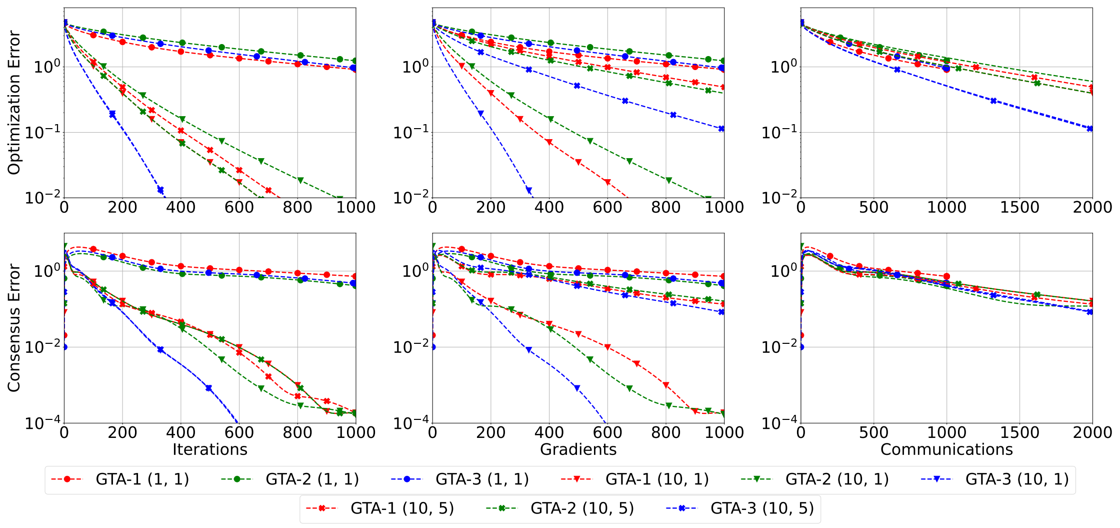

Fig. 3: Optimization Error ($\|\bar{x}_k - x^*\|_2$) and Consensus Error ($\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|_2$) of `GTA-1`, `GTA-2` and `GTA-3` with respect to number of iterations, communications and gradient evaluations for binary logistic regression on Mushroom dataset ($n = 16$, $d = 117$, $\sum_{i=1}^{n} n_i = 8124$) over cyclic network ($\beta = 0.992$).

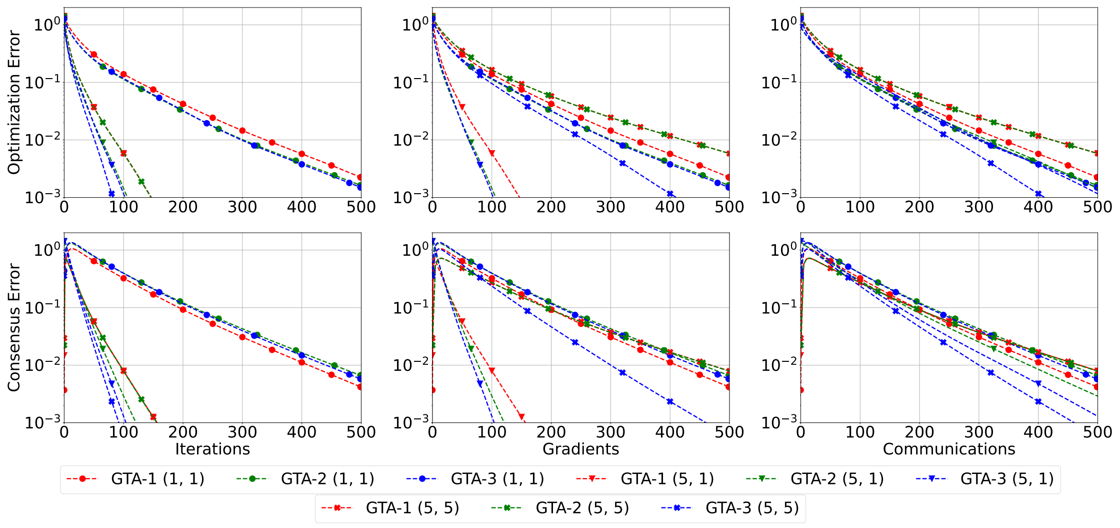

Fig. 4: Optimization Error ($\|\bar{x}_k - x^*\|_2$) and Consensus Error ($\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|_2$) of `GTA-1`, `GTA-2` and `GTA-3` with respect to number of iterations, communications and gradient evaluations for binary logistic regression on Australian dataset ($n = 16$, $d = 41$, $\sum_{i=1}^{n} n_i = 690$) over star network ($\beta = 0.95$).

non-smooth structured problems, stochastic local objective functions, and directed communication graphs. It is also valuable to adapt the framework to asynchronous communication setups. This allows for adaptive schemes in which agents determine the number of communication and computation steps independently based on local information.

Data Availability Statement
The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

References

1. Sulaiman A Alghunaim, Ernest K Ryu, Kun Yuan, and Ali H Sayed. Decentralized proximal gradient algorithms with linear convergence rates. *IEEE Transactions on Automatic Control*, 66(6):2787–2794, 2020.
2. Yossi Arjevani, Joan Bruna, Bugra Can, Mert Gurbuzbalaban, Stefanie Jegelka, and Hongzhou Lin. Ideal: Inexact decentralized accelerated augmented lagrangian method. *Advances in Neural Information Processing Systems*, 33:20648–20659, 2020.
3. Necdet Serhat Aybat, Zi Wang, Tianyi Lin, and Shiqian Ma. Distributed linearized alternating direction method of multipliers for composite convex consensus optimization. *IEEE Transactions on Automatic Control*, 63(1):5–20, 2017.
4. Brian Baingana, Gonzalo Mateos, and Georgios B Giannakis. Proximal-gradient algorithms for tracking cascades over social networks. *IEEE Journal of Selected Topics in Signal Processing*, 8(4):563–575, 2014.
5. Albert S Berahas, Raghu Bollapragada, and Shagun Gupta. A flexible gradient tracking algorithmic framework for decentralized optimization. *arXiv preprint arXiv:2312.06814*, 2023.
6. 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.
7. 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.
8. 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.
9. Dimitri Bertsekas and John Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 2015.
10. Yongcan Cao, Wenwu Yu, Wei Ren, and Guanrong Chen. An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial Informatics*, 9(1):427–438, 2012.
11. 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.
12. Fengwen Chen, Guodong Long, Zonghan Wu, Tianyi Zhou, and Jing Jiang. Personalized federated learning with graph. *arXiv preprint arXiv:2203.00829*, 2022.
13. Patrick L Combettes and Jean-Christophe Pesquet. Proximal splitting methods in signal processing. In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pages 185–212. Springer, 2011.

14. Rudrajit Das, Anish Acharya, Abolfazl Hashemi, Sujay Sanghavi, Inderjit S Dhillon, and Ufuk Topcu. Faster non-convex federated learning via global and local momentum. In *Uncertainty in Artificial Intelligence*, pages 496–506. PMLR, 2022.
15. 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.
16. Dheeru Dua and Casey Graff. UCI Machine Learning Repository, 2017.
17. Pedro A Forero, Alfonso Cano, and Georgios B Giannakis. Consensus-based distributed support vector machines. *Journal of Machine Learning Research*, 11(5), 2010.
18. Diyako Ghaderyan, Necdet Serhat Aybat, A Pedro Aguiar, and Fernando Lobo Pereira. A fast row-stochastic decentralized method for distributed optimization over directed graphs. *IEEE Transactions on Automatic Control*, 2023.
19. Eduard Gorbunov, Filip Hanzely, and Peter Richtárik. Local sgd: Unified theory and new efficient methods. In *International Conference on Artificial Intelligence and Statistics*, pages 3556–3564. PMLR, 2021.
20. Roger A Horn and Charles R Johnson. *Matrix Analysis*. Cambridge university press, 2012.
21. Dušan Jakovetić, José MF Moura, and Joao Xavier. Linear convergence rate of a class of distributed augmented lagrangian algorithms. *IEEE Transactions on Automatic Control*, 60(4):922–936, 2014.
22. Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pages 5132–5143. PMLR, 2020.
23. Anusha Lalitha, Osman Cihan Kilinc, Tara Javidi, and Farinaz Koushanfar. Peer-to-peer federated learning on graphs. *arXiv preprint arXiv:1901.11173*, 2019.
24. Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated Learning: Challenges, Methods, and Future Directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.
25. Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*, 2019.
26. Qing Ling, Wei Shi, Gang Wu, and Alejandro Ribeiro. Dlm: Decentralized linearized alternating direction method of multipliers. *IEEE Transactions on Signal Processing*, 63(15):4051–4064, 2015.
27. Wei Liu, Li Chen, Yunfei Chen, and Wenyi Zhang. Accelerating federated learning via momentum gradient descent. *IEEE Transactions on Parallel and Distributed Systems*, 31(8):1754–1766, 2020.
28. Sindri Magnússon. *Bandwidth Limited Distributed Optimization with Applications to Networked Cyberphysical Systems*. PhD thesis, KTH Royal Institute of Technology, 2017.
29. Gabriel Mancino-Ball, Yangyang Xu, and Jie Chen. A decentralized primal-dual framework for non-convex smooth consensus optimization. *arXiv preprint arXiv:2107.11321*, 2021.
30. Fatemeh Mansoori and Ermin Wei. Flexpd: A flexible framework of first-order primal-dual algorithms for distributed optimization. *IEEE Transactions on Signal Processing*, 69:3500–3512, 2021.
31. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017.
32. Konstantin Mishchenko, Grigory Malinovsky, Sebastian Stich, and Peter Richtárik. Proxskip: Yes! local gradient steps provably lead to communication acceleration! finally! *arXiv preprint arXiv:2202.09357*, 2022.
33. Aritra Mitra, Rayana Jaafar, George J Pappas, and Hamed Hassani. Linear convergence in federated learning: Tackling client heterogeneity and sparse gradients. *Advances in Neural*

Information Processing Systems, 34:14606–14619, 2021.
34. Aryan Mokhtari, Qing Ling, and Alejandro Ribeiro. Network newton distributed optimization methods. *IEEE Transactions on Signal Processing*, 65(1):146–161, 2016.
35. Angelia Nedić and Alex Olshevsky. Distributed optimization over time-varying directed graphs. *IEEE Transactions on Automatic Control*, 60(3):601–615, 2014.
36. 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.
37. Angelia Nedić, Alex Olshevsky, Wei Shi, and César A Uribe. Geometrically convergent distributed optimization with uncoordinated step-sizes. In *2017 American Control Conference (ACC)*, pages 3950–3955. IEEE, 2017.
38. Angelia Nedic and Asuman Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, 2009.
39. Yurii Nesterov. Introductory lectures on convex programming volume i: Basic Course. *Lecture Notes*, 3(4):5, 1998.
40. Edward Duc Hien Nguyen, Sulaiman A Alghunaim, Kun Yuan, and César A Uribe. On the performance of gradient tracking with local updates. *arXiv preprint arXiv:2210.04757*, 2022.
41. Joel B Predd, Sanjeev R Kulkarni, and H Vincent Poor. *Distributed Learning in Wireless Sensor Networks*. John Wiley & Sons: Chichester, UK, 2007.
42. Shi Pu and Angelia Nedić. Distributed stochastic gradient tracking methods. *Mathematical Programming*, 187(1):409–457, 2021.
43. 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.
44. Guannan Qu and Na Li. Harnessing smoothness to accelerate distributed optimization. *IEEE Transactions on Control of Network Systems*, 5(3):1245–1260, 2017.
45. Ali H Sayed. Diffusion adaptation over networks. In *Academic Press Library in Signal Processing*, volume 3, pages 323–453. Elsevier, 2014.
46. 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.
47. Wei Shi, Qing Ling, Kun Yuan, Gang Wu, and Wotao Yin. On the linear convergence of the admm in decentralized consensus optimization. *IEEE Transactions on Signal Processing*, 62(7):1750–1761, 2014.
48. Ying Sun, Gesualdo Scutari, and Amir Daneshmand. Distributed optimization based on gradient tracking revisited: Enhancing convergence rate via surrogation. *SIAM Journal on Optimization*, 32(2):354–385, 2022.
49. Akhil Sundararajan, Bin Hu, and Laurent Lessard. Robust convergence analysis of distributed optimization algorithms. In *2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1206–1212. IEEE, 2017.
50. Ye Tian, Ying Sun, and Gesualdo Scutari. Asy-sonata: Achieving linear convergence in distributed asynchronous multiagent optimization. In *2018 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 543–551. IEEE, 2018.
51. Konstantinos I Tsianos, Sean Lawlor, and Michael G Rabbat. Consensus-based distributed optimization: Practical issues and applications in large-scale machine learning. In *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1543–1550. IEEE, 2012.
52. Ermin Wei and Asuman Ozdaglar. On the o (1= k) convergence of asynchronous distributed alternating direction method of multipliers. In *2013 IEEE Global Conference on Signal and Information Processing*, pages 551–554. IEEE, 2013.

53. Chenguang Xi and Usman A Khan. Dextra: A fast algorithm for optimization over directed graphs. *IEEE Transactions on Automatic Control*, 62(10):4980–4993, 2017.
54. Chenguang Xi, Ran Xin, and Usman A. Khan. Add-opt: Accelerated distributed directed optimization. *IEEE Transactions on Automatic Control*, 63(5):1329–1339, 2018.
55. Ran Xin, Chenguang Xi, and Usman A Khan. Frost—fast row-stochastic optimization with uncoordinated step-sizes. *EURASIP Journal on Advances in Signal Processing*, 2019:1–14, 2019.
56. Jinming Xu, Ye Tian, Ying Sun, and Gesualdo Scutari. Distributed algorithms for composite optimization: Unified framework and convergence analysis. *IEEE Transactions on Signal Processing*, 69:3555–3570, 2021.
57. 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.
58. Honglin Yuan and Tengyu Ma. Federated accelerated stochastic gradient descent. *Advances in Neural Information Processing Systems*, 33:5332–5344, 2020.
59. Jiaqi Zhang and Keyou You. Asyspa: An exact asynchronous algorithm for convex optimization over digraphs. *IEEE Transactions on Automatic Control*, 65(6):2494–2509, 2020.
60. Shengjun Zhang, Xinlei Yi, Jemin George, and Tao Yang. Computational convergence analysis of distributed optimization algorithms for directed graphs. In *2019 IEEE 15th International Conference on Control and Automation (ICCA)*, pages 1096–1101. IEEE, 2019.
61. Yuchen Zhang and Lin Xiao. Communication-efficient distributed optimization of self-concordant empirical loss. *Large-Scale and Distributed Optimization*, pages 289–341, 2018.
62. Ke Zhou and Stergios I Roumeliotis. Multirobot active target tracking with combinations of relative observations. *IEEE Transactions on Robotics*, 27(4):678–695, 2011.