
Decentralized Constrained Optimization: Double Averaging and Gradient Projection

Firooz Shahriari-Mehr, David Bosch and Ashkan Panahi

Department of Computer Science and Engineering
Chalmers University of Technology
Gothenburg, Sweden

Firooz, Davidbos, Ashkan.Panahi@Chalmers.se

Abstract

In this paper, we consider the convex, finite-sum minimization problem with explicit convex constraints over strongly connected directed graphs. The constraint is an intersection of several convex sets each being known to only one node. To solve this problem, we propose a novel decentralized projected gradient scheme based on local averaging and prove its convergence using only local functions' smoothness. Experimental studies demonstrate the effectiveness of the proposed method in both constrained and unconstrained problems.

1 Introduction

In the past decade, decentralized optimization techniques have attracted significant interest [18, 36]. In this setting, multiple computing nodes are involved, and there is no coordinator (central) node with which all nodes communicate. A fairly general framework for decentralized optimization problems is given by

$$\min_{\mathbf{x} \in \mathbb{R}^m} f(\mathbf{x}) \triangleq \sum_{v=1}^M f_v(\mathbf{x}), \quad (1)$$

where M is the total number of the nodes in the network, $\mathbf{x} \in \mathbb{R}^m$ is called the global optimization variable, and $f(\cdot)$ is the global objective function which has a finite-sum structure. Here, each node v has access to its local function $f_v : \mathbb{R}^m \rightarrow \mathbb{R}$ and communicates to its neighbors \mathcal{N}_v to achieve an *optimal consensus solution*. A natural extension to this setup is when \mathbf{x} in problem (1) is required to lie in an intersection of several convex sets, i.e. $\mathbf{x} \in \bigcap_{v=1}^N S_v$, and each constraint S_v is known only to one node. Applications of this setup, that we refer to as the Decentralized Constrained Optimization Problem (DCOP), are ubiquitous, e.g. smart grid control [1, 7], optimal energy management [23], sensor networks [2], and support vector machines [4]. However, practical approaches to solving it have not been extensively discussed in the literature. This paper responds to this shortcoming by providing a numerical method to solve DCOP with guaranteed convergence properties.

Node communication is a crucial factor in the design of decentralized optimization techniques, which is represented by either a directed or undirected communication graph. Earlier studies on decentralized techniques considered static undirected graphs, which indicate that each communication link between two nodes is time invariant and bi-directional, meaning that both nodes can send and receive information. This assumption is not compatible with many practical applications, such as broadcast channels with no return link, or communication failures leading to uni-directional links [34]. These problems intrigue researchers to propose decentralized methods considering directed graphs as the underlying communication network, where each communication link in the

network is uni-directional. For simplicity, decentralized methods in which a directed graph represents their node communication are called *decentralized directed methods*, throughout this paper. In the same way, we refer to *decentralized undirected methods*. We address the more general case of decentralized directed scenarios, while the undirected cases follow as a special case.

For undirected communication graphs, efficient optimization techniques with provable convergence properties exist based on suitable iterative averaging over neighbor nodes [5, 21, 28]. The averaging procedure is mathematically represented by the so-called gossip matrices, which are compatible with the network structure, double stochastic, and symmetric [20, 28]. The required gossip matrices can be constructed using Laplacian or Metropolis matrices for undirected graphs. Such gossip matrices are not compatible with directed graphs, as they require asymmetry and finding doubly stochastic matrices is not straightforward, often requiring distributed and iterative numerical procedures such as iterative weight balancing [6]. For this reason, practical schemes utilize row stochastic or column stochastic matrices, instead of using doubly stochastic matrices. In this case, convergence bounds comparable to the undirected scenarios, even in the absence of constraints, are lacking to the best of our knowledge. We further address this issue by proposing a novel double-averaging scheme, similar to the so-called push-pull approach [24], which takes both row and column stochastic matrices into account, and at the same time enjoys superior convergence guarantees.

1.1 Contributions

The main contributions of the paper are summarized as follows:

- We propose a novel algorithm, called DAGP, to solve the problem of decentralized constraint optimization. Our scheme employs double averaging and projection onto convex sets. It extends the tracking approach, which has been proposed for the first time in [20, 28], to constrained problems and can benefit a fixed step size and fast convergence.
- In contrast to the previously proposed methods in the literature, our method simultaneously considers a directed communication graph and individual constraints at each node.
- We show that our technique is applicable to generic constrained convex problems, lacking strong convexity, while maintaining the convergence rate of order $\mathcal{O}(1/\sqrt{n})$, under mild conditions. We are not aware of any decentralized unconstrained method over directed communication graphs with similar established convergence properties.
- We present experiments for constrained decentralized optimization problems on directed graphs, where DAGP outperforms the existing algorithms. We also conduct experiments on unconstrained problems, where DAGP performs similarly to the state of the art, decentralized optimization algorithms.

1.2 Literature Review

In this section, we review the decentralized methods in the existing literature. We organize our review into three parts: techniques on directed and undirected graphs, and decentralized constrained methods. Several classes of methods exist in the literature that are not in the scope of this paper, *e.g.*, methods considering time-varying graphs [19, 20], local functions with a finite-sum structure [8, 9, 17, 33], or compressed communication [3, 12].

1.2.1 Decentralized optimization over undirected graphs

The algorithms for undirected communication graphs can be divided into several categories. First, the decentralized gradient decent methods including [15, 21] use diminishing step size for convergence to the exact solution of the problem. The diminishing step size leads to practical difficulties with step tuning but establish the convergence rate of $\mathcal{O}(\log n/\sqrt{n})$ in a convex and smooth setting and $\mathcal{O}(\log n/n)$ in a strongly convex and smooth setting. The second category refers to the methods that use gradient tracking technique and leverage the gradient information at all nodes to estimate the gradient of the global function [20, 25, 28]. These methods use fixed step sizes and achieve linear convergence rate, *i.e.* $\mathcal{O}(\mu^n)$ $\mu < 1$, in a strongly convex and smooth setting. [25] has also shown a sublinear rate of convergence, *i.e.* $\mathcal{O}(1/n)$, when the functions are not strongly convex. The dual-based methods [10, 27, 30] are included in the third group. Although these methods are optimal and

have linear convergence rate, they need to compute some computationally costly oracles, *e.g.* the gradient of a conjugate function, which is not practical in some applications.

1.2.2 Decentralized optimization over directed graphs

Earlier methods for directed problems apply the so-called push-sum protocol [11] to decentralized gradient descent methods to tackle the problem of computing a doubly stochastic gossip matrix for directed graphs [19, 29]. These methods utilize a column stochastic matrix, but a diminishing step size is still vital for their convergence. The methods based on the push-sum protocol converge with order of $\mathcal{O}(\log n/n)$ for smooth and strongly convex functions. To achieve fixed step size, [20, 32] have put push-sum protocol and gradient tracking technique together and have respectively proposed the DEXTRA and Push-DIGing algorithms. These algorithms achieved a linear rate of convergence in a smooth and strongly convex setting. DEXTRA suffers theoretical limitations on the step size, namely a feasible step size might not exist in some cases. [32] has proposed the ADD-OPT algorithm to solve this problem. This algorithm also enjoys linear convergence for strongly convex functions. Recently, methods based on two gossip matrices, one column stochastic and the other one row stochastic, have been proposed in the literature [24, 35], called Push-Pull methods. These methods also have linear convergence in a smooth and strongly convex setting. Our algorithm is similar to push-pull methods as it can be applied with similar underlying matrices.

1.2.3 Decentralized Constrained optimization

Despite extensive studies on decentralized optimization, there exist few papers that consider constraints explicitly. It is worth mentioning that a straightforward approach to solving constrained problems is to add the indicator functions of the constraint sets to the problem, then apply the methods proposed for the unconstrained problem. This approach requires methods that are applicable to non-smooth and non-strongly convex functions with unbounded (sub)gradients due to indicator function characteristics. For this reason, we note that utilizing the previously mentioned methods does not guarantee convergence. [26] is among the first papers incorporating the projection and averaging approaches, but it assumes that the constraint set is identical at all nodes. This leads to a problem when the projection onto the constraint set is not computationally efficient. In response, the projected subgradient algorithm has been proposed, which assumes that the constraint set is different and distributed among all nodes [22]. This paper is similar in setup to ours, but does not provide a precise convergence rate. Moreover, convergence of local variables to a consensus stopping point is proven only in two special cases: when the constraints are identical, or when the graph is fully connected. As the constraint at each node might be an intersection of several constraints, or in some applications, the nodes do not have access to all of their local constraints at each iteration, [14] has proposed a randomized projection scheme. This algorithm suffers from the same limitations as [22], *i.e.* the proof is only reliable for fully connected networks or a setting with identical constraints at each node. All the above-mentioned methods use a diminishing step size as they do not leverage any gradient tracking technique. Moreover, they assume that the underlying communication graph is undirected. [31] has proposed the DDPS algorithm, which is applicable when the communication graph is directed. However, this algorithm uses diminishing step size as well, and its convergence rate is of order $\mathcal{O}(\log n/\sqrt{n})$. Moreover, it is subject to the restrictive assumption that the constraints are identical. There are also methods with different problem description, such as composite constrained optimization [16]. These methods consider undirected communication graphs, and they are different from our problem, in nature.

1.3 Paper Outline

The rest of the paper is organized as follows. In the following, some preliminary definitions and notations are introduced. The DAGP algorithm is proposed in section 2, along with theoretical convergence analysis. The proofs of all Lemmas and Theorems are provided in the Appendix. Finally, section 3 is devoted to the numerical studies.

Definition 1 (Normal cone and Projection Operator). For a closed convex set $S \subset \mathbb{R}^n$, the normal cone of S is given by

$$\partial I_S(\mathbf{x}) = \begin{cases} \emptyset & \mathbf{x} \notin S \\ \{\mathbf{g} \in \mathbb{R}^n \mid \forall \mathbf{z} \in S, \mathbf{g}^T(\mathbf{z} - \mathbf{x}) \leq 0\} & \mathbf{x} \in S \end{cases}.$$

Moreover, the projection of a vector $\mathbf{x} \in \mathbb{R}^n$ onto S is computed by

$$P_S(\mathbf{x}) = \arg \min_{\mathbf{y} \in S} \|\mathbf{y} - \mathbf{x}\|_2^2.$$

Definition 2 (Graph Theory). A directed graph is shown by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \dots, N\}$ is a set of all nodes, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is a set of ordered pairs of distinct nodes, called edges. A directed path between two distinct nodes $u, v \in \mathcal{V}$ is a sequence of nodes $(u = v_0, v_1, \dots, v_k = v)$ such that each pair (v_i, v_{i+1}) is an edge in \mathcal{E} . A graph \mathcal{G} is strongly connected if for any two distinct nodes $u, v \in \mathcal{V}$, there exist a directed path between u and v . The adjacency matrix denoted by $\mathbf{A} = [a_{ij}]$ is an asymmetric matrix, where a_{ij} is +1 if $(i, j) \in \mathcal{E}$, and 0 otherwise.

In this paper, each pair shows a communication link between two distinct nodes, and (i, j) is a pair in \mathcal{E} , if there is a link from node j to node i . With this intuition, the i th row of an adjacency matrix shows from which nodes it can receive information, and constitute the incoming neighbors of node i called $\mathcal{N}_i^{\text{in}} = \{j | (i, j) \in \mathcal{E}\}$. On the other hand, the i th column of an adjacency matrix shows to which nodes it can send information, and constitute the outgoing neighbors of node i called $\mathcal{N}_i^{\text{out}} = \{j | (j, i) \in \mathcal{E}\}$. The in-degree and out-degree of node i are defined as the cardinality of $\mathcal{N}_i^{\text{in}}$ and $\mathcal{N}_i^{\text{out}}$, respectively. Consequently, two Laplacian matrices can be defined as

$$\mathbf{L}^{\text{in}} = \mathbf{D}^{\text{in}} - \mathbf{A}, \quad \mathbf{L}^{\text{out}} = \mathbf{D}^{\text{out}} - \mathbf{A},$$

where \mathbf{D}^{in} is the in-degree diagonal matrix; that is $d_{ii}^{\text{in}} = |\mathcal{N}_i^{\text{in}}|$, and \mathbf{D}^{out} is defined in a similar way. \mathbf{L}^{in} and \mathbf{L}^{out} have zero row-sum and column-sum characteristics and their scaled versions are used in this paper.

1.4 Mathematical Notation

In this paper, bold lowercase and uppercase letters are used to respectively represent vectors and matrices. w_{vu} denotes the element at the v^{th} row and the u^{th} column of the matrix \mathbf{W} . \mathbf{W}^T shows the transpose of \mathbf{W} , and $\ker(\mathbf{W})$ is its right null space, meaning that $\mathbf{x} \in \ker(\mathbf{W})$, if and only if $\mathbf{W}\mathbf{x} = \mathbf{0}$. $\mathbf{1}_n$ and $\mathbf{0}_n$ respectively denote the n -dimensional vectors of all ones and zeros. The index n may be dropped if there is no risk of confusion. Furthermore, \mathbf{O} denotes a matrix with all zero elements. The Euclidean inner product of vectors is denoted by $\langle \cdot, \cdot \rangle$. Matrix inner product is denoted by $\langle \mathbf{A}, \mathbf{C} \rangle = \text{Tr}(\mathbf{A}\mathbf{C}^T)$. In this paper, subscript generally defines the iteration number, and superscript defines the node number, e.g., $\nabla f_v(\mathbf{x}_n^v)$ indicates the gradient of the node v 's local function at its local variable at iteration n . Finally, $\delta_{n,0}$ is the Kronecker delta function.

2 Problem setting and Proposed Algorithm

In this section, we propose a new algorithm to solve DCOP, considering directed graphs as a communication network between the nodes. The proposed algorithm is called *DAGP* due to Double Averaging and Gradient Projection approaches used in the iterative equations, which are introduced in the subsection 2.2.

2.1 Problem Formulation

Decentralized Constraint Optimization Problem (DCOP) is formulated as

$$\min_{\mathbf{x} \in \mathbb{R}^m} f(\mathbf{x}) \triangleq \sum_{v=1}^M f_v(\mathbf{x}) \quad \text{s.t.} \quad \mathbf{x} \in \bigcap_{v=1}^M S_v, \quad (2)$$

where S_v is a closed convex set, and the intersection of all these constraint sets is called the feasible set. Note that without loss of generality, the number of constraints is equal to the number of functions. This allows us to assume M nodes in our setting, each having access to one function and one constraint. Note that a different number of constraints can still be considered in our setup, as each node may further access a composite constraint (i.e. an intersection of simpler constraints), or multiple nodes may share identical constraints (i.e. $S_v = S_u$) or have trivial constraints $S_v = \mathbb{R}^m$. Nevertheless, we merely require the projection operator to the constraint set S_v of each node to be available, and neglect further possible structures in them.

In decentralized optimization, each node stores and updates a local variable \mathbf{x}^v as its solution. The nodes should achieve consensus, i.e. \mathbf{x}^v 's must converge to an equal stopping point \mathbf{x}^* , which is further required to be a feasible and optimal solution of (2).

2.2 DAGP Algorithm

The DAGP algorithm does the following updates in each iteration, $\forall v \in \mathcal{V}$.

$$\mathbf{z}^v = \mathbf{x}_n^v - \sum_{u \in \mathcal{N}_v^{\text{in}}} w_{vu} \mathbf{x}_n^u - \mu (\nabla f_v(\mathbf{x}_n^v) - \mathbf{g}_n^v) \quad (3)$$

$$\mathbf{x}_{n+1}^v = P_{S_v}(\mathbf{z}^v) \quad (4)$$

$$\begin{aligned} \mathbf{g}_{n+1}^v = \mathbf{g}_n^v + \rho \left[\nabla f_v(\mathbf{x}_n^v) - \mathbf{g}_n^v + \frac{1}{\mu} (\mathbf{z}^v - \mathbf{x}_{n+1}^v) \right] \\ + \alpha (\mathbf{h}_n^v - \mathbf{g}_n^v) \end{aligned} \quad (5)$$

$$\mathbf{h}_{n+1}^v = \mathbf{h}_n^v - \sum_{u \in \mathcal{N}_v^{\text{in}}} q_{vu} (\mathbf{h}_n^u - \mathbf{g}_n^u) \quad (6)$$

To interpret the algorithm, consider the above update equations. Take into account that the node v has access to $(\mathbf{x}_n^u, \mathbf{h}_n^u - \mathbf{g}_n^u)$, $\forall u \in \mathcal{N}_v^{\text{in}}$, as each node u broadcasts this pair of messages to its out-neighbors. First weighted averaging happens in (3), where each node computes a weighted average of its local variable and its in-neighbors' local variables. This averaging is a basis for achieving consensus, and $\mathbf{W} = [w_{vu}]$ must have zero row-sum structure to achieve this objective. Then, the resulting averaged vector is aligned with the negative of the augmented local descent direction $(\nabla f_v(\mathbf{x}_n^v) - \mathbf{g}_n^v)$ scaled by a fixed step size μ . The resulting solution \mathbf{z}^v is projected onto the local constraint set in (4). Therefore from the second iteration, the local variables at each iteration lie in their own local constraint set, but not necessarily in the feasible set of the problem in (2). One of the novelties of this paper is the definition of additional variables \mathbf{h}_n^v together with \mathbf{g}_n^v to push the algorithm towards an optimal consensus solution.

Vectors \mathbf{g}^v act as the memory of the algorithm, which preserve and track the previous information of local functions' gradients and *feasible directions*, i.e. ∇f_v and $\mathbf{z}^v - \mathbf{x}^v$, respectively. To reach an optimal solution, the gradients and feasible directions of all nodes must be aggregated. This is achieved by adding the term $\alpha(\mathbf{h}^v - \mathbf{g}^v)$ to (5), where \mathbf{h}^v is updated using (6). \mathbf{h}^v propagates the information of the gradients and feasible directions of other nodes through the second weighted averaging using $\mathbf{Q} = [q_{vu}]$. In (6), we further require \mathbf{Q} to be a matrix with zero column-sum structure. Then, $\sum_{v \in \mathcal{V}} \mathbf{h}^v$ will not change over time. We also require \mathbf{h}_0^v 's be initialized in a way that satisfy $\sum_{v \in \mathcal{V}} \mathbf{h}_0^v = \mathbf{0}$. The easiest way is to initialize them with zero vectors. In this way, when \mathbf{g}^v 's converges to \mathbf{h}^v 's, their summation over all nodes will be equal to zero. This in turn, leads to satisfying the optimality condition of the problem as \mathbf{g}^v 's contain gradients and feasible directions of the problem. This is further elaborated in Theorem 1, and Appendix A.

2.3 Convergence Analysis

In this section, we discuss the convergence properties of DAGP. We present two results. First in Theorem 1, we prove that if the iterates of DAGP converge, any stopping point is an optimal and consensus solution of the problem in (2). Then, in Theorem 2, we prove the convergence rate of our proposed algorithm in a smooth and convex setting.

2.3.1 Assumptions

We proceed by formalizing the adopted assumptions as follows.

Assumption 1. The nodes will communicate across a strongly connected directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. This assumption guarantees the sufficient information flow between the nodes as there exists a directed path between every two nodes in the graph. As a result, the nodes can achieve consensus.

Assumption 2. The optimization problem (2) is feasible and attains a finite optimal value $f^* = f(\mathbf{x}^*)$ at an optimal feasible solution \mathbf{x}^* satisfying the optimality condition:

$$\mathbf{0} \in \sum_{v=1}^M (\partial I_{S_v}(\mathbf{x}^*) + \nabla f_v(\mathbf{x}^*)).$$

Assumption 3. There are two weight matrices \mathbf{W} and \mathbf{Q} with a similar sparsity pattern to the adjacency matrix \mathbf{A} of \mathcal{G} . They further satisfy the zero row-sum and zero column-sum structure, respectively. The first one is required for achieving consensus, and the second one is required for attaining the optimality of the solution. Moreover, we assume that $\ker(\mathbf{Q}) = \ker(\mathbf{W}^T)$ and $\ker(\mathbf{W}) = \text{span}\{\mathbf{1}\}$.

Assumption 4. The functions $f_v(\cdot)$ are convex, differentiable and L -smooth.

Now, we define the matrices \mathbf{R} and \mathbf{P} as

$$\mathbf{R} = \begin{bmatrix} \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{O} \\ \mathbf{I} & \mathbf{O} & \mathbf{O} & \mathbf{O} \\ -\frac{\rho}{\mu}\mathbf{I} & \frac{\rho}{\mu}(\mathbf{I} - \mathbf{W}) & \mathbf{I} & \alpha\mathbf{I} \\ \frac{\rho}{\mu}\mathbf{I} & -\frac{\rho}{\mu}(\mathbf{I} - \mathbf{W}) & \mathbf{O} & (1 - \alpha)\mathbf{I} - \mathbf{Q} \end{bmatrix}, \quad \mathbf{P} = \begin{bmatrix} \mathbf{I} \\ \mathbf{O} \\ \mathbf{O} \\ \mathbf{O} \end{bmatrix}. \quad (7)$$

Moreover, for an arbitrary positive value of η , matrix \mathbf{S} is computed as

$$\mathbf{S} = \begin{bmatrix} \left(1 - \frac{L\mu}{2}\right)\mathbf{I} - M\eta\left(\mathbf{I} - \frac{1}{M}\mathbf{1}\mathbf{1}^T\right) & -\frac{1}{2}(\mathbf{I} - \mathbf{W}) + \frac{L\mu}{2}\mathbf{I} & -\frac{\mu}{2}\mathbf{I} & \mathbf{O} \\ -\frac{1}{2}(\mathbf{I} - \mathbf{W}^T) + \frac{L\mu}{2}\mathbf{I} & -\frac{L\mu}{2}\mathbf{I} & \mathbf{O} & \mathbf{O} \\ -\frac{\mu}{2}\mathbf{I} & \mathbf{O} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{O} \end{bmatrix}. \quad (8)$$

Please check Appendix B for understanding the evolution of these matrices.

Assumption 5. There exists a strictly positive constant C such that for every value of $\beta > 0$ and in a small neighborhood of $z = 0$ on the complex plane, 1 is not an eigenvalue of the matrix

$$[\mathbf{I} \ \mathbf{O} \ \mathbf{O}] \mathbf{F}^{-1}(z, \beta) \begin{bmatrix} -(C + \beta)\mathbf{I} \\ \mathbf{I} \\ \mathbf{O} \end{bmatrix}, \quad (9)$$

where \mathbf{F} is defined as

$$\mathbf{F}(z, \beta) = \begin{bmatrix} \mathbf{S} & z^{-1}\mathbf{I} - \mathbf{R}^T & \mathbf{O} \\ z\mathbf{I} - \mathbf{R} & \mathbf{O} & -\mathbf{P} \\ \mathbf{O} & -\mathbf{P}^T & -\beta\mathbf{I} \end{bmatrix}. \quad (10)$$

2.3.2 Main Results

Here, we present the main theoretical results and postpone the details and proofs to the Appendix.

Theorem 1. Let Assumptions 3 and 4 hold. If the iterates of DAGP algorithm converge, any stopping point is an optimal and consensus solution of the decentralized constrained optimization problem in (2), i.e. $\mathbf{x}^v = \mathbf{x}^*$, $\forall v \in \mathcal{V}$, and \mathbf{x}^* satisfies the sufficient optimality conditions.

We also present guarantees for the rate of convergence.

Theorem 2. Let all the assumptions hold. Define $\bar{\mathbf{x}}_N^v = \frac{1}{N} \sum_{n=0}^{N-1} \mathbf{x}_n^v$ and $\bar{\mathbf{x}}_N = \frac{1}{M} \sum_v \bar{\mathbf{x}}_N^v$. Then, $\|\bar{\mathbf{x}}_N - \bar{\mathbf{x}}_N^v\|^2 = O\left(\frac{1}{N}\right)$, $\text{dist}^2(\bar{\mathbf{x}}_N, S_v) = O\left(\frac{1}{N}\right)$ and

$$\left| \sum_v f_v(\bar{\mathbf{x}}_N^v) - \sum_v f_v(\mathbf{x}^*) \right| = O\left(\frac{1}{\sqrt{N}}\right). \quad (11)$$

3 Experimental Results

We evaluate and compare the performance of the DAGP algorithm in two scenarios; decentralized constraint and unconstrained problems. In the first experiment, which contains examples with synthetic data, we consider constraints and examine the convergence and feasibility gap of the DAGP compared to the DDPS algorithm. In the second experiment, we solve the classical logistic regression problem, which is unconstrained. In the latter, we compare our algorithm to the state of the art decentralized unconstrained optimization algorithms over directed graphs, namely the ADD-OPT and Push-Pull algorithms.

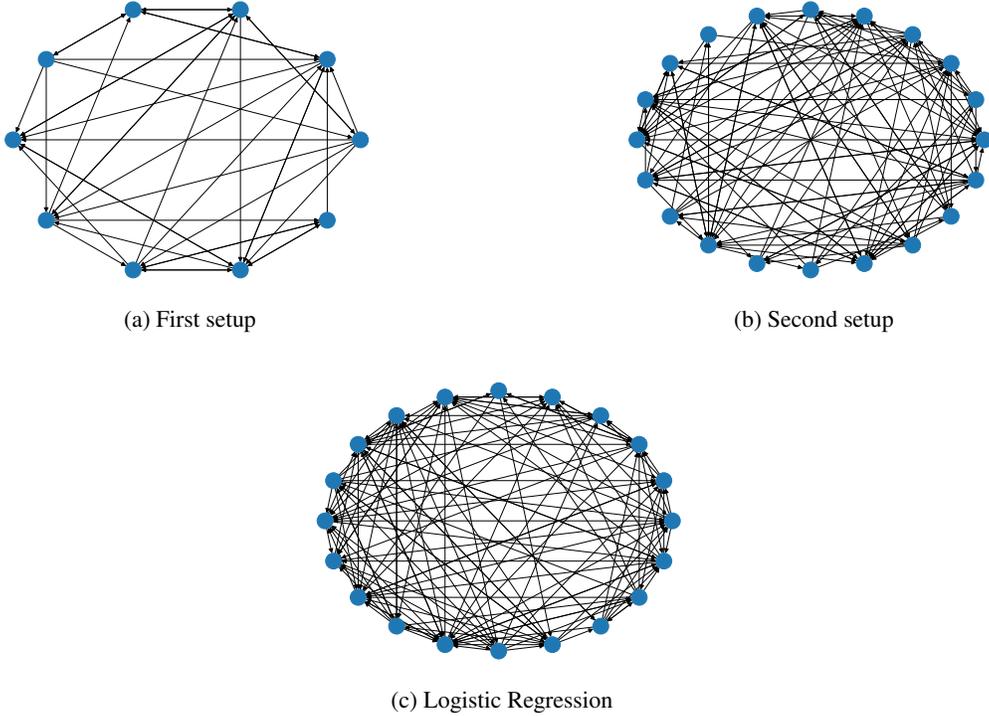


Figure 1: Directed random graphs used in our experiments.

In all algorithms used in the first experiment, the parameters are hand-tuned in a way that the algorithms achieve their best performance, leading to a fair comparison. In the real-world logistic regression problem, hand-tuning parameters is not computationally feasible due to the size of the experiment. Therefore, an appropriate step size is selected for all algorithms in the second experiment. Different algorithms use different matrices for averaging. In this paper, we respectively use $\mathbf{L}^{\text{in}}/2\mathbf{d}_{\text{max}}^{\text{in}}$ and $\mathbf{L}^{\text{out}}/2\mathbf{d}_{\text{max}}^{\text{out}}$ as zero row-sum and zero column-sum matrices, where $\mathbf{d}_{\text{max}}^{\text{in}}$ and $\mathbf{d}_{\text{max}}^{\text{out}}$ are the largest diagonal elements of \mathbf{L}^{in} and \mathbf{L}^{out} . By subtracting these matrices from the identity matrix, row stochastic and column stochastic matrices used in this paper are computed. Moreover, random directed and strongly connected graphs are used in our experiments, which are shown in Fig 1. The numerical experiments are described next. We repeated each experiment multiple times, but only one instance from each experiment is presented as the difference between individual runs was minimal.

3.1 Numerical Results

In this experiment, which contains two setups, we consider synthetic functions and constraints. In our setups, there are M nodes, each having access to one function and one constraint. The nodes communicate over a randomly generated graph and their local variables \mathbf{x}^v s are of size m , which their initial vectors are generated randomly from zero mean and unit variance normal distribution. The functions are selected to be smooth, but not strongly convex as follows:

$$f_v(\mathbf{x}) = \log(\cosh(\mathbf{a}_v^T \mathbf{x} - b_v)), \quad (12)$$

where \mathbf{a}_v s and b_v s are randomly generated from a zero mean and unit variance normal distribution. Moreover, we choose randomly generated linear constraints $\mathbf{c}_v^T \mathbf{x} - d_v \leq 0$ since their orthogonal projection operator is simple to compute.¹

In the first setup, $m = 20$ and $M = 10$, while in the second one, $m = 10$ and $M = 20$. These parameters are chosen since the feasible set in the second experiment is significantly smaller than

¹In all simulations, \mathbf{c}_v and d_v are selected such that their intersection not being an empty set, *e.g.*, \mathbf{c}_v s are generated randomly, then d_v s are selected such that $\mathbf{c}_v^T \mathbf{x} \leq d_v$ for one arbitrary vector \mathbf{x} .

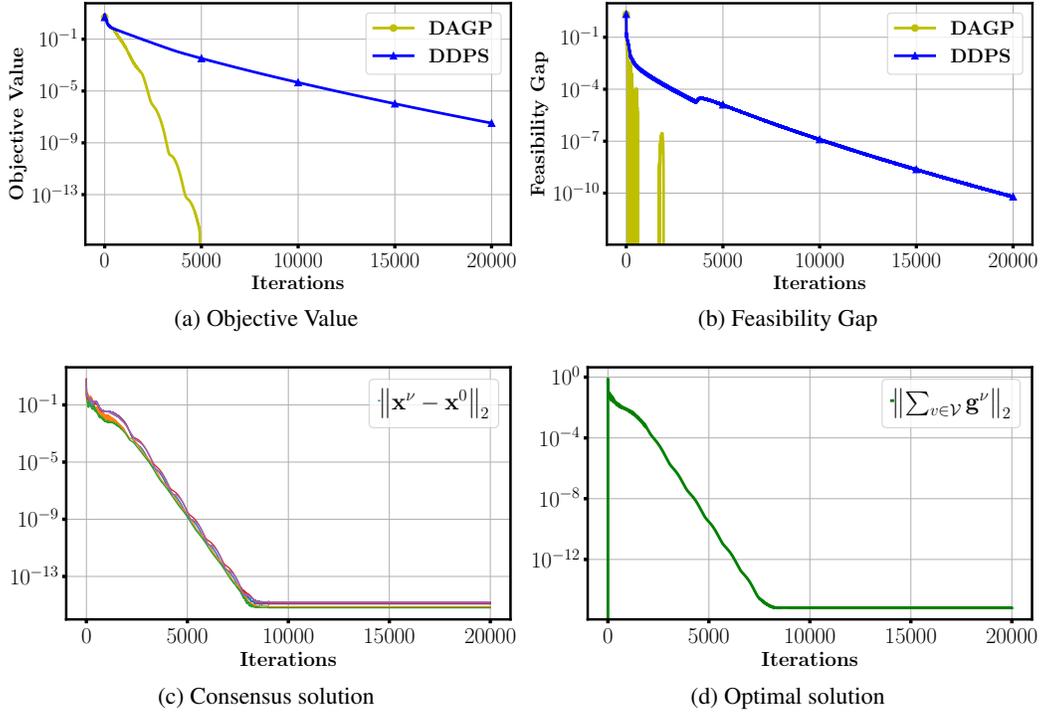


Figure 2: First setup results with $m = 20$ and $M = 10$. Local variables move to a consensus and optimal stopping point in DAGP, while they move to a sub-optimal point in DDPS.

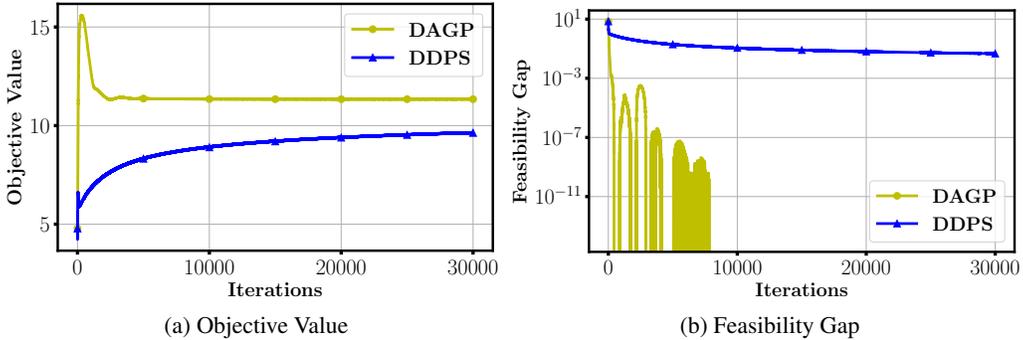


Figure 3: Second setup results with $m = 10$ and $M = 20$. As DDPS has not converged to a point in the feasible set, it can achieve less function value.

the feasible set in the first experiment. In both setups, the objective value and the distance to the feasible set, called feasibility gap, are reported, both computed at $\bar{\mathbf{x}} = \sum_{v \in \mathcal{V}} \mathbf{x}^v$.

The results of these setups are shown respectively in Figures 2 and 3. We observe that $\bar{\mathbf{x}}$ moves completely to the feasible set in our algorithm unlike DDPS in which $\bar{\mathbf{x}}$ becomes only close to the feasible set. Moreover, our algorithm converges faster to the optimal consensus solution in comparison with DDPS algorithm since DDPS needs a diminishing step size. To show that all nodes achieve consensus, the squared norm of error between \mathbf{x}^v at five random nodes and \mathbf{x}^0 is plotted in Fig 2c, where all nodes converge to one stopping point. As described in section 2.3, $\sum_{v \in \mathcal{V}} \mathbf{g}^v$ should become equal to zero to have optimal solution. For this reason, the norm of this variable is computed and shown in Fig 2d, which approaches zero as the algorithm proceeds.

In the first setup, $\mathbf{C} = [c_i^T] \in \mathbb{R}^{M \times m}$ has a null space, and the feasible set of its corresponding optimization problem is larger than the feasible set of the second experiment. The linear convergence

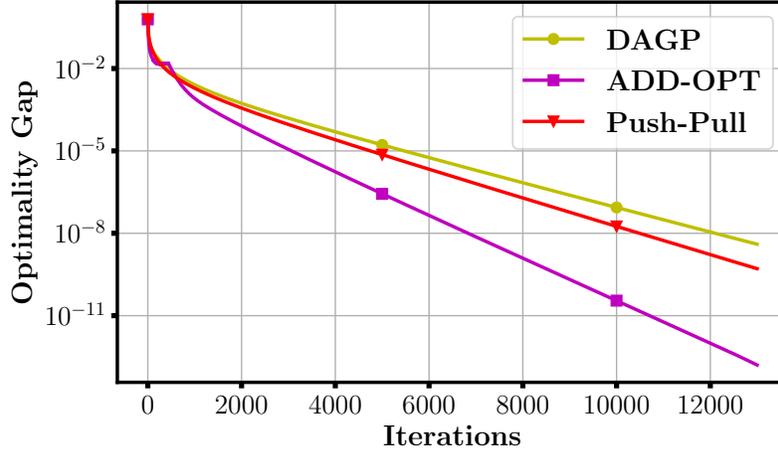


Figure 4: Convergence rate comparison of decentralized unconstrained algorithms over directed graphs. A fixed step size is used in all algorithms.

rate of DAGP in Fig 2a, can be due to a situation where the constraints are not active at the solution, and the algorithm attains the optimal value of the unconstrained version of the problem. Then, the overall unconstrained objective function may be strongly convex, explaining faster convergence. On the other hand, in the second experiment, some constraints are active, and the algorithm slows down in converging to a consensus and an optimal solution. In Fig 3a, DDPS achieves a smaller objective value as its local variables remain infeasible.

3.2 Logistic Regression

For the sake of comparison with other algorithms, we examine an unconstrained logistic regression problem. We consider the MNIST [13] dataset restricted to two digits to form a binary classification problem. We once again consider a random directed graph with $M = 20$ nodes as shown in Fig 1c. Total number of $N_s = 10000$ images are used for training the model, i.e. for minimizing the logistic loss function defined as

$$\min_{\mathbf{w} \in \mathbb{R}^{784}} \sum_{i=1}^{N_s} \log(1 + \exp(-y_i \mathbf{x}_i^T \mathbf{w})) + \frac{\lambda}{2} \|\mathbf{w}\|_2, \quad (13)$$

where $\{\mathbf{x}_i, y_i\}_{i=1}^{N_s} \subseteq \mathbb{R}^{784} \times \{+1, -1\}$ is the set of training samples, and λ is the regularization parameter, which is chosen to be $1/N_s$. We assume that the training samples are distributed in a balanced way between 20 nodes; as a result, each node has 500 training samples. The loss function at each node f_v is the collection of terms in (13) associated with the samples of the node v . The regularization term ensures that this problem is strongly convex, leading to linear rate of convergence for all algorithms.

We compare DAGP with Push-Pull and ADD-OPT. For all algorithms, fixed, similar, and appropriate step size is used. Centralized gradient descent is used to determine the optimal value f^* and compare all algorithms objective values with respect to it. The results for optimality gap, defined as $\sum_{v \in \mathcal{V}} f_v(\bar{\mathbf{x}}) - f^*$, are shown in Fig 4.

As observed, the difference between the convergence rate of these three graphs is minimal. As discussed earlier, optimal step sizes are not used in this experiment due to the size of the problem. Nevertheless, for smaller experiments, we observe that DAGP and Push-Pull can utilize larger step sizes, while ADD-OPT fails to converge with a similar step size. We conclude that by choosing the optimal values of the step size, DAGP and Push-Pull behave similarly, and both outperform ADD-OPT. The practical applicability of our algorithm is immediate, as it is competitive with respect to constrained problems, but also provably capable of solving unconstrained optimization problems, without any modifications to the algorithmic structure.

4 Conclusion

We introduce the Double Averaging Gradient Projection (DAGP) algorithm designed for solving decentralized optimization problems over directed graphs for problems with and without constraints. In contrast to the existing literature, DAGP allows for different constraints at each node in a directed graph. Previous algorithms such as DDPS [31], require a decreasing step size to solve similar constrained problems, while DAGP requires constant step size by employing a gradient tracking technique. By taking advantage of the projection operator onto convex sets and double averaging, we prove the $\mathcal{O}(1/\sqrt{n})$ rate of convergence on constrained problems in a convex and smooth setting. Comparing to the previously proposed algorithms, DAGP is competitive with other decentralized unconstrained directed optimization algorithms, such as ADP-OPT [32] and Push-Pull [24], and greatly outcompetes the previous decentralized constrained optimization algorithms such as DDPS.

5 acknowledgement

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

References

- [1] Mahnoosh Alizadeh, Xiao Li, Zhifang Wang, Anna Scaglione, and Ronald Melton. Demand-side management in the smart grid: Information processing for the power switch. *IEEE Signal Processing Magazine*, 29(5):55–67, 2012.
- [2] Juan Andrés Bazerque and Georgios B Giannakis. Distributed spectrum sensing for cognitive radio networks by exploiting sparsity. *IEEE Transactions on Signal Processing*, 58(3):1847–1862, 2009.
- [3] Aleksandr Beznosikov, Samuel Horváth, Peter Richtárik, and Mher Safaryan. On biased compression for distributed learning. *arXiv preprint arXiv:2002.12410*, 2020.
- [4] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.
- [5] Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. Randomized gossip algorithms. *IEEE transactions on information theory*, 52(6):2508–2530, 2006.
- [6] Bahman Ghahesifard and Jorge Cortés. Distributed strategies for generating weight-balanced and doubly stochastic digraphs. *European Journal of Control*, 18(6):539–557, 2012.
- [7] Xiaohong Guan, Zhanbo Xu, and Qing-Shan Jia. Energy-efficient buildings facilitated by microgrid. *IEEE Transactions on smart grid*, 1(3):243–252, 2010.
- [8] Hadrien Hendrikx, Francis Bach, and Laurent Massoulié. An accelerated decentralized stochastic proximal algorithm for finite sums. *arXiv preprint arXiv:1905.11394*, 2019.
- [9] Hadrien Hendrikx, Francis Bach, and Laurent Massoulié. Dual-free stochastic decentralized optimization with variance reduction. *arXiv*, (NeurIPS):1–12, 2020.
- [10] Hadrien Hendrikx, Francis Bach, and Laurent Massoulié. An optimal algorithm for decentralized finite sum optimization. *arXiv preprint arXiv:2005.10675*, 2020.
- [11] David Kempe, Alin Dobra, and Johannes Gehrke. Gossip-based computation of aggregate information. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 482–491. IEEE, 2003.
- [12] Anastasia Koloskova, Sebastian U. Stich, and Martin Jaggi. Decentralized stochastic optimization and gossip algorithms with compressed communication. *36th International Conference on Machine Learning, ICML 2019, 2019-June:6088–6111*, 2019.
- [13] Yann LeCun and Corinna Cortes. Mnist handwritten digit database. 2010.
- [14] Soomin Lee and Angelia Nedic. Distributed random projection algorithm for convex optimization. *IEEE Journal of Selected Topics in Signal Processing*, 7(2):221–229, 2013.
- [15] Ilan Lobel and Asuman Ozdaglar. Distributed subgradient methods for convex optimization over random networks. *IEEE Transactions on Automatic Control*, 56(6):1291–1306, 2010.

- [16] Qingguo Lü, Xiaofeng Liao, Huaqing Li, and Tingwen Huang. A computation-efficient decentralized algorithm for composite constrained optimization. *IEEE Transactions on Signal and Information Processing over Networks*, 6:774–789, 2020.
- [17] Aryan Mokhtari and Alejandro Ribeiro. DSA: Decentralized double stochastic averaging gradient algorithm. *Journal of Machine Learning Research*, 17:1–35, 2016.
- [18] Angelia Nedic. Distributed Gradient Methods for Convex Machine Learning Problems in Networks: Distributed Optimization. *IEEE Signal Processing Magazine*, 37(3):92–101, 2020.
- [19] Angelia Nedić and Alex Olshevsky. Distributed optimization over time-varying directed graphs. *IEEE Transactions on Automatic Control*, 60(3):601–615, 2014.
- [20] 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.
- [21] Angelia Nedic and Asuman Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, 2009.
- [22] Angelia Nedic, Asuman Ozdaglar, and Pablo A Parrilo. Constrained consensus and optimization in multi-agent networks. *IEEE Transactions on Automatic Control*, 55(4):922–938, 2010.
- [23] HNT Dinh Hoa Nguyen, Tatsuo Narikiyo, and Michihiro Kawanishi. A distributed optimization method for optimal energy management in smart grid. In *Research Trends and Challenges in Smart Grids*. IntechOpen, 2019.
- [24] Shi Pu, Wei Shi, Jinming Xu, and Angelia Nedic. Push-pull gradient methods for distributed optimization in networks. *IEEE Transactions on Automatic Control*, 2020.
- [25] Guannan Qu and Na Li. Harnessing smoothness to accelerate distributed optimization. *IEEE Transactions on Control of Network Systems*, 5(3):1245–1260, 2018.
- [26] S Sundhar Ram, Angelia Nedić, and Venugopal V Veeravalli. Distributed stochastic subgradient projection algorithms for convex optimization. *Journal of optimization theory and applications*, 147(3):516–545, 2010.
- [27] Kevin Seaman, Francis Bach, Sebastien Bubeck, Yin Tat Lee, and Laurent Massoulié. Optimal algorithms for smooth and strongly convex distributed optimization in networks. *34th International Conference on Machine Learning, ICML 2017*, 6:4630–4642, 2017.
- [28] 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.
- [29] Konstantinos I Tsianos, Sean Lawlor, and Michael G Rabbat. Push-sum distributed dual averaging for convex optimization. In *2012 IEEE 51st IEEE conference on decision and control (cdc)*, pages 5453–5458. IEEE, 2012.
- [30] César A Uribe, Soomin Lee, Alexander Gasnikov, and Angelia Nedić. A dual approach for optimal algorithms in distributed optimization over networks. *Optimization Methods and Software*, pages 1–40, 2020.
- [31] Chenguang Xi and Usman A Khan. Distributed subgradient projection algorithm over directed graphs. *IEEE Transactions on Automatic Control*, 62(8):3986–3992, 2016.
- [32] Chenguang Xi, Ran Xin, and Usman A. Khan. ADD-OPT: Accelerated Distributed Directed Optimization. *IEEE Transactions on Automatic Control*, 63(5):1329–1339, 2018.
- [33] Ran Xin, Soumya Kar, and Usman A Khan. Decentralized stochastic optimization and machine learning: A unified variance-reduction framework for robust performance and fast convergence. *IEEE Signal Processing Magazine*, 37(3):102–113, 2020.
- [34] Ran Xin and Usman A. Khan. A Linear Algorithm for Optimization over Directed Graphs with Geometric Convergence. *IEEE Control Systems Letters*, 2(3):313–318, 2018.
- [35] Ran Xin and Usman A Khan. A linear algorithm for optimization over directed graphs with geometric convergence. *IEEE Control Systems Letters*, 2(3):315–320, 2018.
- [36] Ran Xin, Shi Pu, Angelia Nedić, and Usman A Khan. A general framework for decentralized optimization with first-order methods. *Proceedings of the IEEE*, 108(11):1869–1889, 2020.

The notations used throughout the paper are used similarly here, except for the \mathbf{z}^v variable, to which the iteration number is added.

Appendix A: Proof of Theorem 1

We start by presenting the following lemma.

Lemma 1. Let $\ker(\mathbf{W}^T) = \ker(\mathbf{Q})$. Then, $\mathbf{Q}\mathbf{W}\mathbf{x} = \mathbf{0}$ if and only if $\mathbf{x} \in \ker(\mathbf{W})$.

Proof. The forward proof is trivial. For the backward proof, we can write

$$\begin{aligned} \mathbf{Q}\mathbf{W}\mathbf{x} = \mathbf{0} &\Rightarrow \mathbf{W}\mathbf{x} \in \ker(\mathbf{Q}) \\ &\Rightarrow \mathbf{W}\mathbf{x} \in \ker(\mathbf{W}^T) \\ &\Rightarrow \mathbf{W}^T\mathbf{W}\mathbf{x} = \mathbf{0} \\ &\Rightarrow \mathbf{x}^T\mathbf{W}^T\mathbf{W}\mathbf{x} = 0 \\ &\Rightarrow \|\mathbf{W}\mathbf{x}\|_2^2 = 0 \\ &\Rightarrow \mathbf{x} \in \ker(\mathbf{W}) \end{aligned}$$

■

Now, consider an arbitrary stopping point of the algorithm, that is $\mathbf{x}_{n+1}^v = \mathbf{x}_n^v = \mathbf{x}^v$, $\nabla\mathbf{f}_{n+1} = \nabla\mathbf{f}_n = \nabla\mathbf{f}$, $\mathbf{h}_{n+1}^v = \mathbf{h}_n^v = \mathbf{h}^v$ and $\mathbf{g}_{n+1}^v = \mathbf{g}_n^v = \mathbf{g}^v$. We have

$$\mathbf{Z} = \mathbf{X} - \mathbf{W}\mathbf{X} - \mu(\nabla\mathbf{f} - \mathbf{G}) \quad (14)$$

$$\mathbf{X} = \mathcal{P}_S(\mathbf{Z}) \quad (15)$$

$$\rho \left[\nabla\mathbf{f} - \mathbf{G} + \frac{1}{\mu}(\mathbf{Z} - \mathbf{X}) \right] + \alpha(\mathbf{H} - \mathbf{G}) = \mathbf{0} \quad (16)$$

$$\mathbf{Q}(\mathbf{H} - \mathbf{G}) = \mathbf{0}, \quad (17)$$

where $\mathbf{Z}, \mathbf{G}, \mathbf{H}, \nabla\mathbf{f}, \mathcal{P}_S$ are matrices with $\mathbf{z}^v, \mathbf{g}^v, \mathbf{h}^v, \nabla f^v(\mathbf{x}^v), P_{S_v}(\mathbf{z}^v)$ as their rows. Left multiplying (16) by \mathbf{Q} , considering (17), we have

$$\mathbf{Q}(\mathbf{G} - \nabla\mathbf{f}) = \frac{1}{\mu}\mathbf{Q}(\mathbf{Z} - \mathbf{X}). \quad (18)$$

Left multiplying (14) by \mathbf{Q} , and applying (18) leads to $\mathbf{Q}\mathbf{W}\mathbf{X} = \mathbf{0}$. Therefore, $\mathbf{X} \in \ker(\mathbf{W}) = \text{span}\{\mathbf{1}\}$ based on the result of Lemma 1, which means that $\mathbf{x}^v = \mathbf{x}^*$, $\forall v \in \mathcal{V}$. As $\mathbf{X} \in \ker(\mathbf{W})$, (14) reduces to $\mathbf{Z} - \mathbf{X} = \mu(\mathbf{G} - \nabla\mathbf{f})$, which leads to $\mathbf{H} = \mathbf{G}$ by incorporating it into (16). Since (6) is designed to preserve the summation of \mathbf{h}^v 's, and each element of \mathbf{H} is initialized with zero vector, we have

$$\mathbf{1}^T\mathbf{G} = \mathbf{1}^T\mathbf{H} = \sum_{v \in \mathcal{V}} (\mathbf{h}^v)^T = \mathbf{0}^T. \quad (19)$$

From (15), we have $\mathbf{Z} - \mathbf{X} \in \partial\mathbf{I}_S$, consequently, $\mu(\mathbf{G} - \nabla\mathbf{f}) \in \partial\mathbf{I}_S$. As $\partial\mathbf{I}_{S_v}$ is a cone, and therefore invariant to scaling, we can write $(\mathbf{G} - \nabla\mathbf{f}) \in \partial\mathbf{I}_S$. Left multiplying by $\mathbf{1}^T$, and moving all the terms to one side, considering (19), we have

$$\mathbf{0} \in \sum_{v \in \mathcal{V}} (\partial\mathbf{I}_{S_v}(\mathbf{x}^*) + \nabla f_v(\mathbf{x}^*)), \quad (20)$$

which shows that \mathbf{x}^* is the optimal solution. ■

Appendix B: Proof of Theorem 2

We start by defining

$$F^v(\mathbf{x}) = f_v(\mathbf{x}) - f_v(\mathbf{x}^*) - \langle \nabla f_v(\mathbf{x}^*), \mathbf{x} - \mathbf{x}^* \rangle \quad (21)$$

and

$$F_n^v = F^v(\mathbf{x}_n^v). \quad (22)$$

Note that from the convexity of f_v , the values of $F^v(\mathbf{x})$, particularly F_n^v are non-negative. From convexity, we also conclude that

$$\begin{aligned} F_n^v + \langle \nabla f_v(\mathbf{x}^*) - \nabla f_v(\mathbf{x}_n^v), \mathbf{x}_n^v - \mathbf{x}^* \rangle = \\ f_v(\mathbf{x}_n^v) - f_v(\mathbf{x}^*) + \langle \nabla f_v(\mathbf{x}_n^v), \mathbf{x}^* - \mathbf{x}_n^v \rangle \leq 0. \end{aligned} \quad (23)$$

From the L -smoothness property of f_v , we also obtain

$$\begin{aligned} F_{n+1}^v - F_n^v - \langle \nabla f_v(\mathbf{x}_n^v) - \nabla f_v(\mathbf{x}^*), \mathbf{x}_{n+1}^v - \mathbf{x}_n^v \rangle = \\ f_v(\mathbf{x}_{n+1}^v) - f_v(\mathbf{x}_n^v) - \langle \nabla f_v(\mathbf{x}_n^v), \mathbf{x}_{n+1}^v - \mathbf{x}_n^v \rangle \leq \\ \frac{L}{2} \|\mathbf{x}_{n+1}^v - \mathbf{x}_n^v\|^2. \end{aligned} \quad (24)$$

Adding (23) to (24) yields:

$$F_{n+1}^v + \langle \nabla f_v(\mathbf{x}^*) - \nabla f_v(\mathbf{x}_n^v), \mathbf{x}_{n+1}^v - \mathbf{x}^* \rangle - \frac{L}{2} \|\mathbf{x}_{n+1}^v - \mathbf{x}_n^v\|^2 \leq 0. \quad (25)$$

Now, we define

$$T^v(\mathbf{x}) = -\langle \mathbf{n}^v, \mathbf{x} - \mathbf{x}^* \rangle, \quad (26)$$

and $T_n^v = T^v(\mathbf{x}_n^v)$, where $\mathbf{n}^v \in \partial I_{S_v}(\mathbf{x}^*)$. The fact that $\mathbf{x}_{n+1}^v \in S_v$ yields $T_{n+1}^v \geq 0$. Note that from $\mathbf{x}_{n+1}^v = P_{S_v}(\mathbf{z}_n^v)$ and the fact that $\mathbf{x}^* \in S_v$, we have

$$\langle \mathbf{x}^* - \mathbf{x}_{n+1}^v, \mathbf{z}_n^v - \mathbf{x}_{n+1}^v \rangle \leq 0, \quad (27)$$

which can also be written as

$$\mu T_{n+1}^v + \langle \mathbf{x}^* - \mathbf{x}_{n+1}^v, \mathbf{z}_n^v - \mathbf{x}_{n+1}^v - \mu \mathbf{n}^v \rangle \leq 0. \quad (28)$$

Multiplying (25) by μ , adding to (28), plugging the definition of \mathbf{z}_n^v and summing over $v \in \mathcal{V}$ and $n = 0, 1, \dots, N-1$ we obtain

$$\begin{aligned} \mu \sum_{n \in [N], v} (F_{n+1}^v + T_{n+1}^v) - \frac{L\mu}{2} \sum_{n \in [N], v} \|\mathbf{x}_{n+1}^v - \mathbf{x}_n^v\|^2 \\ + \sum_{n \in [N], v} \left\langle \mathbf{x}^* - \mathbf{x}_{n+1}^v, \mathbf{x}_n^v - \sum_u w_{vu} \mathbf{x}_n^u - \mathbf{x}_{n+1}^v + \mu(\mathbf{g}_n^v - \nabla f_v(\mathbf{x}^*) - \mathbf{n}^v) \right\rangle \leq 0. \end{aligned} \quad (29)$$

We also replace the expression of \mathbf{z}_n^v in the dynamics of \mathbf{g}_n^v leading to

$$\mathbf{g}_{n+1}^v = \mathbf{g}_n^v + \frac{\rho}{\mu} \left(\mathbf{x}_n^v - \sum_u w_{vu} \mathbf{x}_n^u - \mathbf{x}_{n+1}^v \right) + \alpha \delta_n^v, \quad (30)$$

where $\delta_n^v = \mathbf{h}_n^v - \mathbf{g}_n^v$ that follows the following dynamics

$$\delta_{n+1}^v = (1 - \alpha) \delta_n^v - \sum_u q_{vu} \delta_n^u - \frac{\rho}{\mu} \left(\mathbf{x}_n^v - \sum_u w_{vu} \mathbf{x}_n^u - \mathbf{x}_{n+1}^v \right). \quad (31)$$

For simplicity, we define $\tilde{\mathbf{x}}_n^v = \mathbf{x}_n^v - \mathbf{x}^*$ and $\tilde{\mathbf{g}}_n^v = \mathbf{g}_n^v - \nabla f_v(\mathbf{x}^*) - \mathbf{n}^v$ and rewrite (29), (30) and (31) as

$$\begin{aligned}
& \sum_{n=0}^{N-1} \left(\mu \sum_v (F_{n+1}^v + T_{n+1}^v) + \frac{\eta}{2} \sum_{u,v} \|\mathbf{x}_{n+1}^u - \mathbf{x}_{n+1}^v\|^2 \right) \\
& - \sum_{n=0}^{N-1} \sum_v \left\langle \tilde{\mathbf{x}}_{n+1}^v, \tilde{\mathbf{x}}_n^v - \sum_u w_{vu} \tilde{\mathbf{x}}_n^u - \tilde{\mathbf{x}}_{n+1}^v + \mu \tilde{\mathbf{g}}_n^v \right\rangle \\
& - \frac{L\mu}{2} \sum_{n=0}^{N-1} \sum_v \|\tilde{\mathbf{x}}_{n+1}^v - \tilde{\mathbf{x}}_n^v\|^2 \\
& - \frac{\eta}{2} \sum_{n=0}^{N-1} \sum_{u,v} \|\tilde{\mathbf{x}}_{n+1}^u - \tilde{\mathbf{x}}_{n+1}^v\|^2 \leq 0, \tag{32}
\end{aligned}$$

$$\tilde{\mathbf{g}}_{n+1}^v = \tilde{\mathbf{g}}_n^v + \frac{\rho}{\mu} \left(\tilde{\mathbf{x}}_n^v - \sum_u w_{vu} \tilde{\mathbf{x}}_n^u - \tilde{\mathbf{x}}_{n+1}^v \right) + \alpha \delta_n^v, \tag{33}$$

$$\delta_{n+1}^v = (1 - \alpha) \delta_n^v - \sum_u q_{vu} \delta_n^u - \frac{\rho}{\mu} \left(\tilde{\mathbf{x}}_n^v - \sum_u w_{vu} \tilde{\mathbf{x}}_n^u - \tilde{\mathbf{x}}_{n+1}^v \right), \tag{34}$$

where in the first inequality we also add and remove the term $\frac{\eta}{2} \sum_{u,v} \|\mathbf{x}_{n+1}^u - \mathbf{x}_{n+1}^v\|^2$.

In the following, we will consider the last three summations in (32) as A_N . We show an asymptotic lower bound for A_N , i.e. show that there exists a constant C only depending on the initial values such that for sufficiently large N , $A_N \geq -C$. Note that since $A_N \leq 0$, we must have $C \geq 0$. Then, we conclude from (32) that

$$\sum_{n=0}^{N-1} \left(\mu \sum_v (F_{n+1}^v + T_{n+1}^v) + \frac{\eta}{2} \sum_{u,v} \|\mathbf{x}_{n+1}^u - \mathbf{x}_{n+1}^v\|^2 \right) \leq C. \tag{35}$$

Defining $\bar{\mathbf{x}}_N^v = \frac{1}{N} \sum_{n=0}^{N-1} \mathbf{x}_n^v$ and noting that each term in the summation over n is a fixed convex function of $\{\mathbf{x}_{n+1}^v\}_v$, we may recall Jensen's inequality to conclude

$$\mu \sum_v F^v(\bar{\mathbf{x}}_N^v) + T^v(\bar{\mathbf{x}}_N^v) + \frac{\eta}{2} \sum_{u,v} \|\bar{\mathbf{x}}_N^u - \bar{\mathbf{x}}_N^v\|^2 \leq \frac{C}{N}. \tag{36}$$

Defining $\bar{\mathbf{x}}_N = \frac{1}{M} \sum_v \bar{\mathbf{x}}_N^v$, we conclude that

$$\|\bar{\mathbf{x}}_N - \bar{\mathbf{x}}_N^u\|^2 = O\left(\frac{1}{N}\right).$$

Since $\bar{\mathbf{x}}_N^u \in S_u$, we also conclude that

$$\text{dist}^2(\bar{\mathbf{x}}_N, S_u) = O\left(\frac{1}{N}\right).$$

Finally,

$$\begin{aligned}
\left| \sum_v f_v(\bar{\mathbf{x}}_N^v) - \sum_v f_v(\mathbf{x}^*) \right| & \leq \frac{C}{\mu N} + \sum_v |\langle \mathbf{n}^v + \nabla f_v(\mathbf{x}^*), \bar{\mathbf{x}}_N^v - \bar{\mathbf{x}}_N \rangle| \\
& \leq \frac{C}{\mu N} + \sqrt{\sum_v \|\mathbf{n}^v + \nabla f_v(\mathbf{x}^*)\|^2} \sqrt{\sum_v \|\bar{\mathbf{x}}_N^v - \bar{\mathbf{x}}_N\|^2} \\
& = O\left(\frac{1}{\sqrt{N}}\right)
\end{aligned}$$

which completes the proof. ■

Bound on A_N

To find the bound C , we start by simplifying the notation in (32), (33) and (34). Let us introduce

$$\Psi_n = [\tilde{\mathbf{X}}_{n+1} \quad \tilde{\mathbf{X}}_n \quad \tilde{\mathbf{G}}_n \quad \Delta_n]^T, \quad (37)$$

where $\tilde{\mathbf{X}}_n, \tilde{\mathbf{G}}_n, \Delta_n$ are matrices with $\tilde{\mathbf{x}}_n^v, \tilde{\mathbf{g}}_n^v, \delta_n^v$ as their v^{th} row, respectively. We may write (33) and (34) as

$$\Psi_{n+1} = \mathbf{R}\Psi_n + \mathbf{P}\tilde{\mathbf{X}}_{n+2}, \quad n = 0, \dots, N-2 \quad (38)$$

where \mathbf{P} and \mathbf{R} are defined as

$$\mathbf{R} = \begin{bmatrix} \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{O} \\ \mathbf{I} & \mathbf{O} & \mathbf{O} & \mathbf{O} \\ -\frac{\rho}{\mu}\mathbf{I} & \frac{\rho}{\mu}(\mathbf{I} - \mathbf{W}) & \mathbf{I} & \alpha\mathbf{I} \\ \frac{\rho}{\mu}\mathbf{I} & -\frac{\rho}{\mu}(\mathbf{I} - \mathbf{W}) & \mathbf{O} & (1 - \alpha)\mathbf{I} - \mathbf{Q} \end{bmatrix}, \quad \mathbf{P} = \begin{bmatrix} \mathbf{I} \\ \mathbf{O} \\ \mathbf{O} \\ \mathbf{O} \end{bmatrix}. \quad (39)$$

We also have

$$A_N = \sum_{n=0}^{N-1} \langle \Psi_n, \mathbf{S}\Psi_n \rangle, \quad (40)$$

where \mathbf{S} is computed as

$$\mathbf{S} = \begin{bmatrix} \left(1 - \frac{L\mu}{2}\right)\mathbf{I} - M\eta\left(\mathbf{I} - \frac{1}{M}\mathbf{1}\mathbf{1}^T\right) & -\frac{1}{2}(\mathbf{I} - \mathbf{W}) + \frac{L\mu}{2}\mathbf{I} & -\frac{\mu}{2}\mathbf{I} & \mathbf{O} \\ -\frac{1}{2}(\mathbf{I} - \mathbf{W}^T) + \frac{L\mu}{2}\mathbf{I} & -\frac{L\mu}{2}\mathbf{I} & \mathbf{O} & \mathbf{O} \\ & -\frac{\mu}{2}\mathbf{I} & \mathbf{O} & \mathbf{O} \\ & \mathbf{O} & \mathbf{O} & \mathbf{O} \end{bmatrix}. \quad (41)$$

The following Lemmas provide a guarantee that A_N is bounded.

Lemma 2. Consider matrices \mathbf{R}, \mathbf{P} and \mathbf{S} , defined in (39, 41). Define a "dual" sequence $\{\Lambda_n\}_{n=-1}^{N-1}$ such that $\Lambda_{N-1} = \Lambda_{-1} = \mathbf{O}$. Suppose that there exists a $C \geq 0$ such that for every $\beta > 0$, the system of equations in (38) together with

$$\Lambda_{n-1} - \mathbf{R}^T \Lambda_n + (\mathbf{S} + (C + \beta)\delta_{n,0}\mathbf{I})\Psi_n = \mathbf{O} \quad n = 0, 1, \dots, N-1 \quad (42)$$

$$\mathbf{P}^T \Lambda_n + \beta\tilde{\mathbf{X}}_{n+2} = \mathbf{O} \quad n = 0, 1, \dots, N-2 \quad (43)$$

has no non-zero solution for $\{\Psi_n, \Lambda_n, \tilde{\mathbf{X}}_{n+2}\}$. Then, $\mathbf{A}_n \geq -C\|\Psi_0\|_{\mathbb{F}}^2$ always holds true.

Proof. Note that the claim is equivalent to the statement that zero is the optimal value for the optimization problem

$$\begin{aligned} & \min_{\{\Psi_n\}_{n=0}^{N-1}, \{\tilde{\mathbf{X}}_{n+2}\}_{n=0}^{N-2}} \frac{1}{2} \sum_{n=0}^{N-1} \langle \Psi_n, \mathbf{S}\Psi_n \rangle + \frac{C}{2} \|\Psi_0\|_{\mathbb{F}}^2 \\ & \text{subject to} \quad \Psi_{n+1} = \mathbf{R}\Psi_n + \mathbf{P}\tilde{\mathbf{X}}_{n+2}, \quad n = 0, 1, \dots, N-2 \end{aligned} \quad (44)$$

If the claim does not hold, the optimization is unbounded and the following restricted optimization will achieve a strictly negative optimal value at a non-zero solution:

$$\begin{aligned} & \min_{\{\Psi_n\}_{n=0}^{N-1}, \{\tilde{\mathbf{X}}_{n+2}\}_{n=0}^{N-2}} \frac{1}{2} \sum_{n=0}^{N-1} \langle \Psi_n, \mathbf{S}\Psi_n \rangle + \frac{C}{2} \|\Psi_0\|_{\mathbb{F}}^2 \\ & \text{subject to} \quad \Psi_{n+1} = \mathbf{R}\Psi_n + \mathbf{P}\tilde{\mathbf{X}}_{n+2}, \quad n = 0, 1, \dots, N-2 \\ & \quad \quad \quad \frac{1}{2} \|\Psi_0\|_{\mathbb{F}}^2 + \frac{1}{2} \sum_{n=0}^{N-2} \|\tilde{\mathbf{X}}_{n+2}\|_{\mathbb{F}}^2 \leq \frac{1}{2} \end{aligned} \quad (45)$$

Such a solution satisfies the KKT condition, which coincides with (42), (43) where $\{\Lambda_n\}, \beta \geq 0$ are dual (Lagrangian) multipliers corresponding to the constraints. We also observe that the optimal value at this point is given by $-\beta \left(\|\Psi_0\|_{\mathbb{F}}^2 + \sum_{n=0}^{N-2} \|\tilde{\mathbf{X}}_{n+2}\|_{\mathbb{F}}^2 \right)$. This shows that $\beta > 0$. This contradicts the assumption that such a point does not exist and completes the proof. \blacksquare

We may further simplify the conditions in Lemma 2 by the following result:

Lemma 3. For a complex value z and a real value β , define

$$\mathbf{F}(z, \beta) = \begin{bmatrix} \mathbf{S} & z^{-1}\mathbf{I} - \mathbf{R}^T & \mathbf{O} \\ z\mathbf{I} - \mathbf{R} & \mathbf{O} & -\mathbf{P} \\ \mathbf{O} & -\mathbf{P}^T & -\beta\mathbf{I} \end{bmatrix}. \quad (46)$$

The condition of Lemma 2 is satisfied if the matrix

$$\lim_{z \rightarrow 0} \begin{bmatrix} \mathbf{I} & \mathbf{O} & \mathbf{O} \end{bmatrix} \mathbf{F}^{-1} \begin{bmatrix} -(C + \beta)\mathbf{I} \\ \mathbf{I} \\ \mathbf{O} \end{bmatrix} \quad (47)$$

doesn't have an eigenvalue of 1.

Proof. With an abuse of notation, define the z -transforms

$$\Psi(z) = \sum_{n=0}^{N-1} \Psi_n z^{-n}, \quad \Lambda(z) = \sum_{n=0}^{N-2} \Lambda_n z^{-n}, \quad \mathbf{U}(z) = \sum_{n=0}^{N-2} \tilde{\mathbf{X}}_{n+2} z^{-n}$$

Then, for the sequences defined in (42, 43, 38), we have

$$(z^{-1}\mathbf{I} - \mathbf{R}^T)\Lambda(z) + \mathbf{S}\Psi(z) + (C + \beta)\Psi_0 = \mathbf{O} \quad (48)$$

$$(z\mathbf{I} - \mathbf{R})\Psi(z) - \Psi_0 + \mathbf{R}\Psi_{N-1}z^{N-1} - \mathbf{P}\mathbf{U}(z) = \mathbf{O} \quad (49)$$

$$\mathbf{P}^T \Lambda(z) + \beta\mathbf{U}(z) = \mathbf{O} \quad (50)$$

which can also be written as

$$\mathbf{F}(z, \beta) \begin{bmatrix} \Psi(z) \\ \Lambda(z) \\ \mathbf{U}(z) \end{bmatrix} = \begin{bmatrix} -(C + \beta)\Psi_0 \\ \Psi_0 - \mathbf{R}\Psi_{N-1}z^{N-1} \\ \mathbf{O} \end{bmatrix} \quad (51)$$

Note that $\mathbf{F}(z, \beta)$ may be rank-deficient at a finite number of points. Hence, there exists a sufficiently small simple loop \mathcal{C} around $z = 0$ such that \mathbf{F} is invertible on and inside it except at $z = 0$. In this region we have

$$\Psi(z) = \begin{bmatrix} \mathbf{I} & \mathbf{O} & \mathbf{O} \end{bmatrix} \mathbf{F}^{-1} \mathbf{A}, \quad (52)$$

where \mathbf{A} is

$$\begin{bmatrix} -(C + \beta)\mathbf{I} \\ \mathbf{I} \\ \mathbf{O} \end{bmatrix} \Psi_0 + \begin{bmatrix} \mathbf{O} \\ -\mathbf{R} \\ \mathbf{O} \end{bmatrix} z^{N-1} \Psi_{N-1}$$

On the other hand,

$$2\pi j \Psi_0 = \oint_{\mathcal{C}} \frac{1}{z} \Psi(z) dz \quad (53)$$

Further from the Cauchy integral formula for sufficiently large N , we have

$$\oint_{\mathcal{C}} \frac{1}{z} z^{N-1} \mathbf{F}^{-1}(z, \beta) dz = 2\pi j \lim_{z \rightarrow 0} z^{N-1} \mathbf{F}^{-1}(z, \beta) = \mathbf{O}. \quad (54)$$

By applying (52) to (53), considering relation in (54), we can conclude that

$$\Psi_0 = \lim_{z \rightarrow 0} \begin{bmatrix} \mathbf{I} & \mathbf{O} & \mathbf{O} \end{bmatrix} \mathbf{F}^{-1} \begin{bmatrix} -(C + \beta)\mathbf{I} \\ \mathbf{I} \\ \mathbf{O} \end{bmatrix} \Psi_0 \quad (55)$$

Note that by the assumption we can conclude that $\Psi_0 = \mathbf{O}$, which completes the proof. \blacksquare