mathematics

MDPI

Article

Distributed Optimization Algorithm for Composite Optimization Problems with Non-Smooth Function

Yawei Shi, Liang Ran *, Jialong Tang and Xiangzhao Wu

Chongqing Key Laboratory of Nonlinear Circuits and Intelligent Information Processing, College of Electronic and Information Engineering, Southwest University, Chongqing 400715, China
* Correspondence: ranliang_rl@163.com

Abstract: This paper mainly studies the distributed optimization problems in a class of undirected networks. The objective function of the problem consists of a smooth convex function and a non-smooth convex function. Each agent in the network needs to optimize the sum of the two objective functions. For this kind of problem, based on the operator splitting method, this paper uses the proximal operator to deal with the non-smooth term and further designs a distributed algorithm that allows the use of uncoordinated step-sizes. At the same time, by introducing the random-block coordinate mechanism, this paper develops an asynchronous iterative version of the synchronous algorithm. Finally, the convergence of the algorithms is proven, and the effectiveness is verified through numerical simulations.

Keywords: distributed optimization; non-smooth convex function; proximal operator; random-block coordinate

MSC: 68W15

Citation: Shi, Y.; Ran, L.; Tang, J.; Wu, X. Distributed Optimization Algorithm for Composite Optimization Problems with Non-Smooth Function. *Mathematics* **2022**, *10*, 3135. https://doi.org/10.3390/math10173135

Academic Editors: Honglei Xu and Lingyun Wang

Received: 22 July 2022
Accepted: 24 August 2022
Published: 1 September 2022

1. Introduction

In this paper, we study a class of distributed multi-agent problems on networks. Each agent in the network system has the following private objective function to be solved

$$F_i(\bar{x}) = f_i(\bar{x}) + g_i(\bar{x}), \tag{1}$$

where $\bar{x} \in \mathbb{R}^n$ is the decision variable, f_i is a Lipschitz-differentiable convex function, and g_i is a non-smooth convex function. Examples of f_i include quadratic functions and logistic functions [1], and applications of function g_i include the elastic-net norm, L1-norm, and indicator functions [2].

For the network system, we consider that each agent in the system is only allowed to interact with neighbor agents, and there is no central agent to process data; then we can obtain

$$\begin{aligned} &\min_{x_1,\cdots,x_m} \sum_{i=1}^{m} F_i(x_i) \\ &\text{s. t. } x_i = x_i,\ (i,j) \in \mathcal{E} \end{aligned} \tag{2}$$

where $x_i \in \mathbb{R}^n$ is the local estimation for $\bar{x}$ and $\mathcal{E}$ represents a collection of edges in the network. This distributed computing architecture captures various areas containing distributed information processing and decision making, networked multi-vehicle coordination, distributed estimation, etc. Typical applications include power systems control [3], model predictive control [4], statistical inference and learning [5], and distributed average consensus [6].

In recent years, most of the literature has mainly focused on the case that the optimization objective function contains only one smooth convex function. At the same time, many centralized algorithms with excellent performance, such as proximal gradient descent, sub-gradient algorithm, Newton method, and so on, solve these problems by extending to a distributed form. The sub-gradient algorithm is the most commonly used method. In [7], Nedić and Ozdaglar apply this method to the distributed optimization problem on time-varying networks and creatively propose the distributed sub-gradient method (DGD). Shi et al. [8] propose an exact first-order algorithm (EXTRA) and prove the linear convergence of the algorithm. The algorithm makes use of the error between adjacent iterations of the DGD algorithm. Then, [9] designs a distributed first-order algorithm by combining DGD and the gradient tracking method. In order to further accelerate the convergence of the algorithm, researchers successively propose the distributed ADMM algorithm in [10–13]. However, these algorithms can only solve the optimization problem of a single function.

For (2) this composite distributed optimization problem with a non-smooth term, many research results have emerged. The authors of [14] design a proximal gradient method by combining Nesterov acceleration mechanisms. However, each iteration will lead to the consumption of more computing resources because more internal iteration steps are required. In undirected networks, Shi et al. design a proximal gradient exact first-order algorithm (PG-EXTRA) for composite optimization problems based on the classical first-order distributed optimization algorithm (EXTRA) [8] in [15]. The algorithm can accurately converge to the optimal solution of the problem by using a fixed step-size, so it is different from most algorithms that must use attenuation step-size. The authors of [16] propose a communication-efficient random walk named Walkman by using a Markov chain. By analyzing the relationship between optimization accuracy and network connectivity, this method obtains the explicit expression of communication complexity and the communication efficiency of the system. Further, considering that the complex situation of the real scene causes most agents in the network to transmit data in a directed way, ref. [17] uses the push sum mechanism to eliminate the information imbalance caused by the directed network and proposes the PG-ExtraPush algorithm on the basis of [8] and maintains the same convergence property.

Recently developed, the operator splitting technology has become the mainstream method to deal with this kind of complex optimization problem. Operator splitting technology is applied for the first time to composite optimization since Combettes and Pesquet designed a fully splitting algorithm, refs. [18–21] and others successively propose various algorithms for composite optimization. However, operator splitting technology is rarely applied to distributed composite optimization. Based on this, this paper aims to design a distributed algorithm with excellent performance by using the operator splitting method and based on the theory of operator monotonicity.

Contributions: Compared with most existing distributed optimization algorithms, the main contributions of this paper are summarized as follows:

1. To solve problem (2), this paper develops a novel, fully distributed algorithm based on the operator splitting method, which has superiorities in flexibility and efficiency compared with relatively centralized counterparts [18–21].
2. Based on a class of randomized block-coordinate methods, an asynchronous iterative version of the proposed algorithm is also derived, wherein only a subset of agents that are independently activated participate in the updates. Note that such an activation scheme is more flexible compared with the single coordinate activation [22].
3. Both proposed algorithms allow not only local information interaction among neighboring agents but also the use of uncoordinated step-sizes, without any requirement of coordinated or dynamic ones considered in [7–9,14,23]. Additionally, the convergence of both algorithms is ensured under the same mild assumptions. In particular, the consideration of the local Lipschitz assumption avoids the conservative selections of step-sizes, unlike the global one assumed in [8,14,15,17].

Organization: The contents of the remaining sections of the paper are as follows. Section 2 provides the symbols, lemmas, definitions, and assumptions that will be used in the paper. We give the specific process of algorithm derivation in Section 3. In Section 4, we show the convergence analysis of the proposed algorithms. Section 5 presents the simulation experiment to verify the algorithms. Finally, Section 6 gives the conclusion of the paper.

2. Preliminaries

In this section, we give the notations and display the definitions and lemmas that will be used in the paper. Then, we give two important assumptions.

Above all, we introduce some knowledge about graph theory. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ represent an undirected network composed of n agents, where $\mathcal{V}$ denotes the set of agents and $\mathcal{E}$ denotes the set of edges. The neighborhood of the i-th agent is recorded as $\mathcal{N}_i = \{j|(i,j) \in \mathcal{E}\}$. Specifically, when there is at least one path in any two agents in an undirected network $\mathcal{G}$, the network is connected.

Let $\mathbb{R}^n$ denote the n-dimensional Euclidean space and $\|\cdot\|$ denote the Euclidean norm of a vector $x \in \mathbb{R}^n$. The notation $\rho_{\max}(\cdot)$ is the spectral radius of a matrix, and $\mathbb{N}$ represents the set of positive integers. Then let $\mathrm{X}_0(\mathbb{R}^n)$ denote the collection of all proper lower semi-continuous convex functions from $\mathbb{R}^n$ to $(-\infty, +\infty]$. When W_i denotes a positive definite matrix, using W_i as the diagonal element can form a positive definite diagonal matrix $\mathrm{blkdiag}\{W_i\}_{i\in\mathcal{V}}$. Let $\mathrm{ri}(\cdot)$ denote the interior of a convex subset and dom_f denote the effective domain of f. The subdifferential of function f_i is expressed as $\partial f(x_1) = \{v \in \mathbb{R}^n | (x_2 - x_1)^{\mathrm{T}} v \leq f(x_2) - f(x_1),\ \forall x_2 \in \mathbb{R}^n\}$. The proximity operator of a function $f \in \mathrm{X}_0(\mathbb{R}^n)$ related to $\|\cdot\|_P$ is defined by $prox_{P^{-1}f}(x) = \arg\min_{y\in\mathbb{R}^n}\{f(y) + (1/2)||x - y||_P^2\}$. The convex conjugate function of f is written as $f^{\otimes}$.

At the same time, we give the following lemmas and assumptions.

Lemma 1 ([24]). *Let $f \in \mathrm{X}_0(\mathbb{R}^n)$, then for vectors $x_1, x_2 \in \mathbb{R}^n$, the following relation holds:*

$$\begin{aligned} x_2 \in \partial f(x_1) &\Leftrightarrow x_1 = prox_f(x_1 + x_2) \\ &\Leftrightarrow x_2 = \Big(I - prox_f\Big)(x_1 + x_2). \end{aligned} \tag{3}$$

Lemma 2 ([25]). *Let $f \in \mathrm{X}_0(\mathbb{R}^n)$, then both $prox_f$ and $I - prox_f$ satisfy a firmly nonexpansive relationship.*

Lemma 3 ([19]). *For a fixed point iteration $u_{k+1} = T(u_k)$, $\{u_k\}$ will converge to the fixed point of T when it satisfies the following conditions:*

1. *T is continuous,*
2. *$\{||u_k - u^*||^2\}$ is non-increasing,*
3. *$\lim_{k\to\infty} \|u_{k+1} - u_k\|^2 = 0$.*

Definition 1. *For all $x_1, x_2 \in \mathbb{R}^n$, if an operator T satisfies $\|Tx_1 - Tx_2\| \leq \|x_1 - x_2\|$, then T is a nonexpansive operator. Further, if T satisfies $\|Tx_1 - Tx_2\|^2 \leq (Tx_1 - Tx_2)^{\mathrm{T}}(x_1 - x_2)$, then T is firmly a nonexpansive operator.*

Definition 2. *When $(Tx_1 - Tx_2)^{\mathrm{T}}(x_1 - x_2) \geq \sigma_T\|x_1 - x_2\|^2$, $x_1, x_2 \in \mathbb{R}^n$ exist for a constant $\sigma_T > 0$ and operator T, then operator T satisfies σ_T-strongly monotone.*

The following assumptions will also be used.

Assumption 1. *Graph $\mathcal{G}$ satisfies undirected and connected operators.*

Assumption 2. *The following three points are satisfied:*

1. $f_i : \mathbb{R}^n \to \mathbb{R}$ *is a smooth convex function, let* $1/\beta_i$ *be Lipschitz constant, then* f_i *satisfies*

$$\beta_i \|\nabla f_i(x_1) - \nabla f_i(x_2)\| \leq \|x_1 - x_2\|, \forall x_1, x_2 \in \mathbb{R}^n,$$

2. $g_i : \mathbb{R}^n \to \mathbb{R}$ *is a convex non-smooth function,*
3. *Problem* (2) *has at least one solution.*

3. Algorithm Development

In this section, we design and derive the synchronous algorithm and asynchronous algorithm.

We next carry on the equivalent transformation to problem (2) to facilitate the subsequent algorithm design. The constraint, $x_i = x_j$ in (2) can be written as the edge-based form

$$E_{ij}x_i + E_{ji}x_j = 0, \tag{4}$$

where $E_{ij} = I \in \mathbb{R}^{n\times n}$ for $i < j$, and $E_{ij} = -I \in \mathbb{R}^{n\times n}$ otherwise. Then, define the following linear operator:

$$M_{(i,j)} : x \to (E_{ij}x_i, E_{ji}x_j) \in \mathbb{R}^{2n\times mn}, \tag{5}$$

with the compact variable $x = [x_1^{\mathrm{T}}, \cdots, x_m^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{mn}$. We stack all $M_{(i,j)}$ to get the following operator:

$$M : x \to \left(M_{(i,j)}x\right)_{(i,j)\in\mathcal{E}}, \tag{6}$$

with the dimension $2n|\mathcal{E}| \times mn$, where $|\mathcal{E}|$ is the number of edges of the network $\mathcal{E}$. Considering the set

$$C_{(i,j)} = \{(e_1, e_2) \in \mathbb{R}^n \times \mathbb{R}^n | e_1 + e_2 = 0\}.$$

Then, constraint (4) can be further reformulated in the following form:

$$M_{(i,j)}x \in C_{(i,j)}. \tag{7}$$

Based on the above analysis, problem (2) can be transformed into

$$\min_{x_i\in\mathbb{R}^n} \sum_{i=1}^{m} f_i(x_i) + g_i(x_i) + \sum_{i=1}^{m} \sum_{(i,j)\in\mathcal{E}} \delta_{C_{(i,j)}}\left(M_{(i,j)}x\right), \tag{8}$$

where δ_C represents the indicator function, i.e.,

$$\delta_{C_{(i,j)}}\left(M_{(i,j)}x\right) = \begin{cases} 0, & M_{(i,j)}x \in C_{(i,j)}, \\ +\infty, & M_{(i,j)}x \notin C_{(i,j)}. \end{cases}$$

Then, let

$$f(x) = \sum_{i=1}^{m} f_i(x_i),$$
$$g(x) = \sum_{i=1}^{m} g_i(x_i),$$
$$\delta_C(Mx) = \sum_{i=1}^{m} \sum_{j\in\mathcal{N}_i} \delta_{C_{(i,j)}}\left(M_{(i,j)}x\right),$$

and $C = \prod_{(i,j)\in\mathcal{E}} C_{(i,j)}$ ($\prod$ denotes the Cartesian product). Hence, the compact form of problem (8) can be expressed by

$$\min_{x\in\mathbb{R}^{mn}} f(x) + g(x) + \delta_C(Mx). \tag{9}$$

3.1. Synchronous Algorithm 1

According to the fixed point theory, we design the distributed optimization algorithm of problem (2) from (9). We define the step-size matrices $\Gamma = \text{blkdiag}\{\gamma_i I_n\}_{i\in\mathcal{V}}$, $\tilde{\Lambda} = \text{blkdiag}\left\{\tilde{\lambda}_{(i,j)}\tilde{\gamma}^{-1}_{(i,j)}\right\}_{(i,j)\in\mathcal{E}}$, and $\tilde{\mathrm{H}} = \text{blkdiag}\left\{\lambda_{(i,j)} I_{2n}\right\}_{(i,j)\in\mathcal{E}}$, where we let $\tilde{\gamma}_{(i,j)} = \text{blkdiag}\{\gamma_i I_n, \gamma_j I_n\}$, and then introduce the following operators:

$$T_0(s^*, x^*) = prox_{\Gamma g}(\ x^* - \Gamma\nabla f(x^*) - (\tilde{\mathrm{H}}M)^{\mathrm{T}} s^*\ ), \tag{10}$$

$$\tilde{T}_1(s^*, x^*) = \left(I - prox_{\tilde{\Lambda}^{-1}\delta_C}\right)(MT_0(s^*, x^*) + s^*), \tag{11}$$

$$T_2(s^*, x^*) = prox_{\Gamma g}\left(x^* - \Gamma\nabla f(x^*) - (\tilde{\mathrm{H}}M)^{\mathrm{T}}\tilde{T}_1(s^*, x^*)\right), \tag{12}$$

$$T(s^*, x^*) = \left(\tilde{T}_1(s^*, x^*), T_2(s^*, x^*)\right), \tag{13}$$

where $x^* = \text{col}\{x_i^*\}_{i=1}^m$ and $s^* = \text{col}\{s^*_{(i,j)}\}_{(i,j)\in\mathcal{E}}$ with $s^*_{(i,j)} = \text{col}\{s^*_{(i,j),i}, s^*_{(i,j),i}\}$ are the fixed points of T. In particular, $s^*_{(i,j),i}$ and $s^*_{(i,j),j}$ are maintained by i and j, respectively. Considering the update variables $y_{k+1} = \text{col}\{y_i^{k+1}\}_{i=1}^m$, $x_{k+1} = \text{col}\{x_i^{k+1}\}_{i=1}^m$, and $s_{k+1} = \text{col}\{s^{k+1}_{(i,j)}\}_{(i,j)\in\mathcal{E}}$ with the edge-based variable $s^{k+1}_{(i,j)} = \text{col}\{s^{k+1}_{(i,j),i}, s^{k+1}_{(i,j),j}\}$, we give the Picard sequence of T and obtain the following update rules:

$$\begin{aligned} y_{k+1} &= prox_{\Gamma g}\left(x_k - \Gamma\nabla f(x_k) - \left(\tilde{H}M\right)^{\mathrm{T}} s_k\right) \\ s_{k+1} &= \left(I - prox_{\tilde{\Lambda}^{-1}\delta_C}\right)(My_{k+1} + s_k) \\ x_{k+1} &= prox_{\Gamma g}\left(x_k - \Gamma\nabla f(x_k) - \left(\tilde{H}M\right)^{\mathrm{T}} s_{k+1}\right) \end{aligned} \tag{14}$$

Let $\bar{w}_{k+1} = \text{col}\{\bar{w}^{k+1}_{(i,j)}\}_{(i,j)\in\mathcal{E}} = \text{col}\{\tilde{\lambda}_{(i,j)}\tilde{\gamma}^{-1}_{(i,j)} \cdot s^{k+1}_{(i,j)}\}_{(i,j)\in\mathcal{E}}$. Using Lemma 2, (14) can be rewritten as

$$y_{k+1} = prox_{\Gamma g}\left(x_k - \Gamma\nabla f(x_k) - \Gamma M^{\mathrm{T}}\tilde{w}_k\right), \tag{15a}$$

$$s_{k+1} = \left(I - prox_{\tilde{\Lambda}^{-1}\delta_C}\right)(My_{k+1} + s_k), \tag{15b}$$

$$x_{k+1} = prox_{\Gamma g}\left(x_k - \Gamma\nabla f(x_k) - \Gamma M^{\mathrm{T}}\tilde{w}_{k+1}\right). \tag{15c}$$

Next, we split (15a)–(15c) in a distributed manner. It follows from (5) and (6) that the i-th component of $M^{\mathrm{T}}\tilde{w}_{k+1}$ is $E_{ij}\tilde{w}^{k+1}_{(i,j),i}$. Note that (15a) can be decomposed into

$$\begin{pmatrix} y_1^{k+1} \\ \vdots \\ y_m^{k+1} \end{pmatrix} = \begin{pmatrix} prox_{\gamma_1 g_1}\left(x_1^k - \gamma_1\nabla f_1\left(x_1^k\right) - \gamma_1\sum_{j\in\mathcal{N}_1} E_{1j}\tilde{w}^k_{(1,j),1}\right) \\ \vdots \\ prox_{\gamma_m g_m}\left(x_m^k - \gamma_m\nabla f_m\left(x_m^k\right) - \gamma_m\sum_{j\in\mathcal{N}_m} E_{mj}\tilde{w}^k_{(m,j),m}\right) \end{pmatrix}. \tag{16}$$

For (15b), multiply both sides of the equality by $\tilde{\Lambda}$. As done in (16), we also split (15b) and (15c) and use the result $prox_{\delta_{C_{(i,j)}}}=proj_{C_{(i,j)}}$ to get the semi-distributed form:

$$y_i^{k+1} = prox_{\gamma_i g_i}\left(x_i^k - \gamma_i \nabla f_i\left(x_i^k\right) - \gamma_i \sum_{j\in\mathcal{N}_i} E_{ij}\tilde{w}_{(i,j),i}^k\right), \tag{17a}$$

$$\tilde{w}_{(i,j)}^{k+1} = \tilde{w}_{(i,j)}^k + \tilde{\lambda}_{(i,j)}\tilde{\gamma}_{(i,j)}^{-1}\left(M_{(i,j)}y^{k+1} - proj_{C_{(i,j)}}\left(\frac{\tilde{\gamma}_{(i,j)}}{\tilde{\lambda}_{(i,j)}}\tilde{w}_{(i,j)}^k + M_{(i,j)}y^{k+1}\right)\right), \tag{17b}$$

$$x_i^{k+1} = prox_{\gamma_i g_i}\left(x_i^k - \gamma_i \nabla f_i\left(x_i^k\right) - \gamma_i \sum_{j\in\mathcal{N}_i} E_{ij}\tilde{w}_{(i,j),i}^{k+1}\right). \tag{17c}$$

Note that (17b) is not fully distributed due to the structure $w_{(i,j)}^{k+1} = \mathrm{col}\{w_{(i,j),i}^{k+1}, w_{(i,j),j}^{k+1}\}$. By using (4) and (5), we can derive that the projection of vectors $e_1, e_2 \in \mathbb{R}^n$ to $C_{(i,j)}$ is expressed as

$$proj_{C_{(i,j)}}(e_1, e_2) = \frac{1}{2}(e_1 - e_2, e_2 - e_1),$$

which contributes to the local update of (17b), i.e.,

$$\tilde{w}_{(i,j),i}^{k+1} = \tilde{w}_{(i,j),i}^k + \frac{\tilde{\lambda}_{(i,j)}}{\gamma_i}\left(y_i^{k+1} - \tfrac{1}{2}\left(\left(\tfrac{\gamma_i}{\tilde{\lambda}_{(i,j)}}\tilde{w}_{(i,j),i}^k + y_i^{k+1}\right) - \left(\tfrac{\gamma_i}{\tilde{\lambda}_{(i,j)}}\tilde{w}_{(i,j),j}^k - y_j^{k+1}\right)\right)\right),$$

$$\tilde{w}_{(i,j),j}^{k+1} = \tilde{w}_{(i,j),j}^k + \frac{\tilde{\lambda}_{(i,j)}}{\gamma_j}\left(-y_j^{k+1} - \tfrac{1}{2}\left(\left(\tfrac{\gamma_j}{\tilde{\lambda}_{(i,j)}}\tilde{w}_{(i,j),j}^k - y_j^{k+1}\right) - \left(\tfrac{\gamma_j}{\tilde{\lambda}_{(i,j)}}\tilde{w}_{(i,j),i}^k + y_i^{k+1}\right)\right)\right).$$

Therefore, according to (17a), (17c), and the update of $w_{(i,j),i}^{k+1}$, we can summarize the synchronous distributed algorithm as follows:

Remark 1. *Notice that Algorithm 1 is completely distributed without involving any global parameters. For example, each agent individually maintains the private primal variable x_i^k, auxiliary variable y_i^k, and edge-based variables $\tilde{w}_{(i,j),i}^{k+1}$. For each edge $(i,j) \in \mathcal{E}$ in the network, $\tilde{w}_{(i,j)}^k = col\{\tilde{w}_{(i,j),i}^k, \tilde{w}_{(i,j),i}^k\}$ as an auxiliary profile contains two components, i.e., $\tilde{w}_{(i,j),i}^k$ and $\tilde{w}_{(i,j),i}^k$, which are respectively kept by i and j. Meanwhile, the information exchange is locally conducted; that is, agent i shares its updated data y_i^{k+1} and $\tilde{w}_{(i,j),i}^{k+1}$ with its all neighbors $j \in \mathcal{N}_i$. On the other hand, the proposed algorithm takes uncoordinated constant positive step-sizes, γ_i, essentially distinguished from the global and dynamic ones in [7–9,14,23]. It is also worth noting that the edge-based step-size $\tilde{\lambda}_{(i,j)}$, held by agents i and j linked by the edge $(i,j) \in \mathcal{E}$, can be seen as inherent parameters of the communication network, revealing the quality of the communication.*

Algorithm 1 Distributed algorithm based on proximal operators

Input: For all agents $i \in \mathcal{V}$, $x_i^0 \in \mathbb{R}^n$, and $\tilde{w}_{(i,j),i}^0 \in \mathbb{R}^n$, where $j \in \mathcal{N}_i$. And select proper positive step-sizes or parameters, γ_i and $\tilde{\lambda}_{(i,j)}$.
For $k = 0, 1, \ldots,$ **do**:

1. $y_i^{k+1} = prox_{\gamma_i g_i}\left(\ x_i^k - \gamma_i \nabla f_i\left(x_i^k\right) - \gamma_i \sum_{j\in\mathcal{N}_i} E_{ij}\tilde{w}_{(i,j),i}^k\ \right)$,
2. $\tilde{w}_{(i,j),i}^{k+1} = \frac{1}{2}\frac{\tilde{\lambda}_{(i,j)}}{\gamma_i}\left(y_i^{k+1} - y_j^{k+1}\right) + \frac{1}{2}\left(\tilde{w}_{(i,j),i}^k + \tilde{w}_{(i,j),j}^k\right), \forall j \in \mathcal{N}_i$,
3. $x_i^{k+1} = prox_{\gamma_i g_i}\left(\ x_i^k - \gamma_i \nabla f_i\left(x_i^k\right) - \gamma_i \sum_{j\in\mathcal{N}_i} E_{ij}\tilde{w}_{(i,j),i}^{k+1}\ \right)$,
4. Send y_i^{k+1}, $\tilde{w}_{(i,j),i}^{k+1}$ to j for $j \in \mathcal{N}_i$,
5. Until the $\left\|x_i^{k+1} - x_i^k\right\|$ approaches zero.

End
Output: The primal variable x_i^{k+1} as the optimal solution x_i^*.

3.2. Asynchronous Algorithm 2

Here, we extend the synchronous Algorithm 1 to the asynchronous iterative version based on the random-block coordinate mechanism in [2]. Combining with the principle of this mechanism, we define the diagonal matrix $P_i \in \mathbb{R}^{(2|\mathcal{E}|+m)n} \times \mathbb{R}^{(2|\mathcal{E}|+m)n}$ (where $|\mathcal{E}|$ denotes the number of edges of the graph $\mathcal{E}$) diagonal elements of 0 or 1 to represent the coordinate matrix, and then divide the vector (s, x) into m blocks. At the same time, we define the activation vector $\xi^k \in \mathbb{R}^m$ of ϕ-valued, where $\phi = 0, 1$ is a binary string with length m. When $\xi_i^k = 1$, it means that the agent i is activated at the k-th iteration; otherwise it is not activated.

In order to describe the activation state of different coordinate blocks and ensure random activation, we give the following assumption.

Assumption 3. *The following two points are satisfied:*

1. *The sum of P_i satisfies $\sum_i^m P_i = I$,*
2. *$\left(\xi^k\right)_{k\geq 0}$ is a ϕ-valued vector satisfying identical independent distributionsand its probability is $p_i = \mathbb{P}\left(\xi_i^k = 1\right) > 0, k \geq 0$.*

Then, based on the given assumption, we can develop the asynchronous algorithm as follows:

It can be seen that Algorithm 2 allows each agent to awaken with an independent probability, which means that a subset of randomly activated agents will participate in the updates while inactivated ones stay in previous states. Such a scheme is more flexible than the single waking-up scheme [22] or other activated block coordinates that are uniformly selected [26]. In addition, the probability is completely independent of the others, which does not meet some strict conditions, such as $\sum_{i=1}^m p_i = 1$.

Algorithm 2 Asynchronous distributed version

Input: For all agents $i \in \mathcal{V}$, $x_i^0 \in \mathbb{R}^{n_i}$, and $\tilde{w}_{(i,j),i}^0 \in \mathbb{R}^n$, where $j \in \mathcal{N}_i$. And select proper positive step-sizes or parameters, γ_i and $\tilde{\lambda}_{(i,j)}$.
For $k = 0, 1, \ldots,$ **do** :

For $j \in \mathcal{N}_i$, each agent i is activated independently with probability p_i, and further performs the update steps 1-5 in Algorithm 1. While agents that are not activated, the last values keep unchanged.

End
Output: The primal variable x_i^{k+1} as the optimal solution x_i^*.

In order to facilitate the subsequent derivation of convergence, we need to give a compact form of Algorithm 2. By making $u = (s, x)$, we get

$$u_{k+1} = u_k + E_{k+1}(Tu_k - u_k), \tag{18}$$

where $E_{k+1} = \sum_{i=1}^m \xi_i^{k+1} P_i$ and operator T can be seen in Equation (11).

4. Convergence Analysis

The convergence proof of the algorithms is provided in this section. The following assumption is the condition to be met for the convergence of the algorithms.

Assumption 4. *Recall the local Lipschitz constant β_i in Assumption 2. It is assumed that the step-sizes satisfy the following conditions:*

$$0 < \gamma_i < 2\beta_i, 0 < \tilde{\lambda}_{(i,j)} < 1.$$

Lemma 4. *Let x^* be a solution to (9), then there are*

$$s^* = \tilde{T}_1(s^*, x^*),$$
$$x^* = T_2(s^*, x^*),$$

which means $u^* = (s^*, x^*)$ is a fixed point of T. On the contrary, x^* is the solution to (9) when u^* is the fixed point of T.

Proof. Use the first-order optimal condition of (9) to obtain $0 \in \Gamma\nabla f(x^*) + \Gamma\partial g(x^*) + \Gamma M^{\mathrm{T}}\partial\delta_C(Mx^*)$, where x^* is the optimal solution. According to the definition of matrix step-sizes, we further obtain

$$0 \in \Gamma\nabla f(x^*) + \Gamma\partial g(x^*) + (\tilde{\mathrm{H}}M)^{\mathrm{T}}\tilde{\Lambda}^{-1}\partial\delta_C(Mx^*).$$

Use Lemma 1 and let $s^* \in \tilde{\Lambda}^{-1}\partial\delta_C(Mx^*)$ to get

$$s^* = \left(I - prox_{\tilde{\Lambda}^{-1}\delta_C}\right)(Mx^* + s^*), \tag{19}$$

$$x^* = prox_{\Gamma g}\left(x^* - \Gamma\nabla f(x^*) - (\tilde{\mathrm{H}}M)^{\mathrm{T}}s^*\right). \tag{20}$$

Then according to (19) and (20), we can get

$$s^* = \left(I - prox_{\tilde{\Lambda}^{-1}\delta_C}\right)\left(Mprox_{\Gamma g}(x^* - \Gamma\nabla f(x^*) - (\tilde{\mathrm{H}}M)^{\mathrm{T}}s^*) + s^*\right).$$

Therefore, we have $x^* = T_2(s^*, x^*)$ and $s^* = \tilde{T}_1(s^*, x^*)$. Meanwhile, $u^* = Tu^*$, where $u^* = (s^*, x^*)$. Accordingly, if there is $u^* = Tu^*$, it can also be deduced that x^* satisfies the first-order optimality condition of problem (9). Thus x^* is an optimal solution of problem (9). □

Lemma 5. *Let Assumptions 1 and 2 hold, then there are*

$$\|s_{k+1} - s^*\|^2_{\tilde{\Lambda}} \le \|s_k - s^*\|^2_{\tilde{\Lambda}} - \|s_{k+1} - s_k\|^2_{\tilde{\Lambda}} + 2(s_{k+1} - s^*)^{\mathrm{T}}\tilde{\Lambda}M(y_{k+1} - x^*), \tag{21}$$

$$\begin{aligned}
&\|x_{k+1} - x^*\|^2_{\Gamma^{-1}} \\
&\quad\le \|x_k - x^*\|^2_{\Gamma^{-1}} - \|x_{k+1} - y_{k+1}\|^2_{\Gamma^{-1}} - \|x_k - y_{k+1}\|^2_{\Gamma^{-1}} \\
&\qquad + 2\left(\Gamma^{-1}x_{k+1} - \Gamma^{-1}y_{k+1}\right)^{\mathrm{T}}\left(\Gamma\nabla f(x_k) + (\tilde{\mathrm{H}}M)^{\mathrm{T}}s_k\right) \\
&\qquad + 2\left(\Gamma^{-1}x^* - \Gamma^{-1}x_{k+1}\right)^{\mathrm{T}}\left(\Gamma\nabla f(x_k) + (\tilde{\mathrm{H}}M)^{\mathrm{T}}s_{k+1}\right) \\
&\qquad + 2\left((g\circ\Gamma)\left(\Gamma^{-1}x^*\right) - (g\circ\Gamma)\left(\Gamma^{-1}y_{k+1}\right)\right).
\end{aligned} \tag{22}$$

Proof. Combining (14), (19), and Lemma 2, we get

$$\begin{aligned}
&\|s_{k+1} - s^*\|^2_{\tilde{\Lambda}} \\
&\quad= \left\|\left(I - prox_{\tilde{\Lambda}^{-1}\delta_C}\right)(My_{k+1} + s_k) - \left(I - prox_{\tilde{\Lambda}^{-1}\delta_C}\right)(Mx^* + s^*)\right\|^2_{\tilde{\Lambda}} \\
&\qquad\le (s_{k+1} - s^*)^{\mathrm{T}}\tilde{\Lambda}((My_{k+1} + s_k) - (Mx^* + s^*)).
\end{aligned}$$

It is further concluded that

$$\begin{aligned}&(s_{k+1}-s^*)^{\mathrm{T}}\tilde{\Lambda}(s_{k+1}-s_k+s_k-s^*)\\&\quad\le(s_{k+1}-s^*)^{\mathrm{T}}\tilde{\Lambda}M(y_{k+1}-x^*)+(s_{k+1}-s^*)^{\mathrm{T}}\tilde{\Lambda}(s_k-s^*).\end{aligned}$$

Here we introduce an equality. For a positive definite matrix K and $x_1,\ x_2,\ x_3\in\mathbb{R}^n$, we have

$$2(x_1-x_2)^{\mathrm{T}}\mathrm{K}(x_3-x_2)=\|x_3-x_2\|_{\mathrm{K}}^2+\|x_1-x_2\|_{\mathrm{K}}^2-\|x_1-x_3\|_{\mathrm{K}}^2. \tag{23}$$

Combining the above two results, we derive

$$\begin{aligned}\|s_{k+1}-s^*\|_{\tilde{\Lambda}}^2=&\|s_k-s^*\|_{\tilde{\Lambda}}^2-\|s_{k+1}-s_k\|_{\tilde{\Lambda}}^2+2(s_{k+1}-s^*)^{\mathrm{T}}\tilde{\Lambda}(s_{k+1}-s_k)\\\le&\|s_k-s^*\|_{\tilde{\Lambda}}^2-\|s_{k+1}-s_k\|_{\tilde{\mathrm{H}}}^2+2(s_{k+1}-s^*)^{\mathrm{T}}\tilde{\Lambda}M(y_{k+1}-x^*).\end{aligned} \tag{24}$$

In order to prove the validity of (22), (3) is used for (14)

$$\Gamma^{-1}\Big(x_k-\Gamma\nabla f(x_k)-(\tilde{\mathrm{H}}M)^{\mathrm{T}}s_{k+1}-x_{k+1}\Big)\in\partial g(x_{k+1}).$$

Using subdifferential properties to obtain

$$(x^*-x_{k+1})^{\mathrm{T}}\Gamma^{-1}\Big(x_k-\Gamma\nabla f(x_k)-(\tilde{\mathrm{H}}M)^{\mathrm{T}}s_{k+1}-x_{k+1}\Big)\le g(x^*)-g(x_{k+1})$$

and equivalent

$$\begin{aligned}&\Big(\Gamma^{-1}x_{k+1}-\Gamma^{-1}x^*\Big)^{\mathrm{T}}(x_{k+1}-x_k)\\&\quad\le-\Big(\Gamma^{-1}x_{k+1}-\Gamma^{-1}x^*\Big)^{\mathrm{T}}\Gamma^{-1}(\Gamma\nabla f(x_k)+(\tilde{\mathrm{H}}M)^{\mathrm{T}}s_{k+1})\\&\qquad+(g\circ\Gamma)\Big(\Gamma^{-1}x^*\Big)-(g\circ\Gamma)\Big(\Gamma^{-1}x_{k+1}\Big).\end{aligned} \tag{25}$$

Moreover, there is

$$\|x_{k+1}-x^*\|_{\Gamma^{-1}}^2=\|x_k-x^*\|_{\Gamma^{-1}}^2-\|x_{k+1}-x_k\|_{\Gamma^{-1}}^2+2(x_{k+1}-x^*)^{\mathrm{T}}\Gamma^{-1}(x_{k+1}-x_k). \tag{26}$$

A derivation similar to (25) is obtained for (14)

$$\begin{aligned}&\Big(\Gamma^{-1}x_{k+1}-\Gamma^{-1}y_{k+1}\Big)^{\mathrm{T}}(x_k-\Gamma\nabla f(x_k)-(\tilde{\mathrm{H}}M)^{\mathrm{T}}s_k-y_{k+1})\\&\quad\le(g\circ\Gamma)\Big(\Gamma^{-1}x_{k+1}\Big)-(g\circ\Gamma)\Big(\Gamma^{-1}y_{k+1}\Big)\\&\quad\Leftrightarrow\Big(\Gamma^{-1}x_{k+1}-\Gamma^{-1}y_{k+1}\Big)^{\mathrm{T}}(x_k-y_{k+1})\\&\quad\le\Big(\Gamma^{-1}x_{k+1}-\Gamma^{-1}y_{k+1}\Big)^{\mathrm{T}}\Big(\Gamma\nabla f(x_k)+(\tilde{\mathrm{H}}M)^{\mathrm{T}}s_k\Big)\\&\qquad+(g\circ\Gamma)\Big(\Gamma^{-1}x_{k+1}\Big)-(g\circ\Gamma)\Big(\Gamma^{-1}y_{k+1}\Big).\end{aligned}$$

Therefore, we deduce

$$\begin{aligned}&-\|x_{k+1}-x_k\|_{\Gamma^{-1}}^2\\&\quad=-\|x_k-y_{k+1}\|_{\Gamma^{-1}}^2-\|x_{k+1}-y_{k+1}\|_{\Gamma^{-1}}^2+2(x_k-y_{k+1})^{\mathrm{T}}\Gamma^{-1}(x_{k+1}-y_{k+1})\end{aligned}$$

$$\begin{aligned}
&\leq -\|x_k - y_{k+1}\|^2_{\Gamma^{-1}} - \|x_{k+1} - y_{k+1}\|^2_{\Gamma^{-1}} \\
&\quad + 2\Big(\Gamma^{-1}x_{k+1} - \Gamma^{-1}y_{k+1}\Big)^{\mathrm{T}}\Big(\Gamma\nabla f(x_k) + (\tilde{\mathrm{H}}M)^{\mathrm{T}}s_k\Big) \\
&\quad + 2\Big((g\circ\Gamma)\Big(\Gamma^{-1}x_{k+1}\Big) - (g\circ\Gamma)\Big(\Gamma^{-1}y_{k+1}\Big)\Big).
\end{aligned}$$

Combining the above two equalities and (26), we can get (22). □

Lemma 6. *Let Assumptions 1 and 2 hold. Set* $\beta = \mathrm{blkdiag}\{\beta_i I_n\}_{i\in\mathcal{V}}$. *For matrix* $P = \mathrm{blkdiag}\{\tilde{\Lambda}, \Gamma^{-1}\}$ *and* $u = (s, x)$, *there is*

$$\begin{aligned}
&\|u_{k+1} - u^*\|^2_P \\
&\quad \leq \|u_{k+1} - u^*\|^2_P - \|s_{k+1} - s_k\|^2_{\tilde{\Lambda}(I - M\Gamma M^{\mathrm{T}}\tilde{\Lambda})} \\
&\qquad - \Big\|y_{k+1} - x_{k+1} + (\tilde{\mathrm{H}}M)^{\mathrm{T}}(s_{k+1} - s_k)\Big\|^2_{\Gamma^{-1}} \\
&\qquad - \|x_k - y_{k+1} - (\Gamma\nabla f(x_k) - \Gamma\nabla f(x^*))\|^2_{\Gamma^{-1}} \\
&\qquad - \|\nabla f(x_k) - \nabla f(x^*)\|^2_{2\beta-\Gamma}.
\end{aligned} \tag{27}$$

Proof. Adding (21) and (22), then rearranging to get

$$\begin{aligned}
&\|x_{k+1} - x^*\|^2_{\Gamma^{-1}} + \|s_{k+1} - s^*\|^2_{\tilde{\Lambda}} \\
&\quad \leq \|x_k - x^*\|^2_{\Gamma^{-1}} + \|s_k - s^*\|^2_{\tilde{\Lambda}} - \|x_k - y_{k+1}\|^2_{\Gamma^{-1}} \\
&\qquad - \|s_{k+1} - s_k\|^2_{\tilde{\Lambda}} - \|x_{k+1} - y_{k+1}\|^2_{\Gamma^{-1}} \\
&\qquad + 2\Big((\tilde{\mathrm{H}}M)^{\mathrm{T}}(s_{k+1} - s_k)\Big)^{\mathrm{T}}\Gamma^{-1}(y_{k+1} - x_{k+1}) \\
&\qquad + 2(\Gamma\nabla f(x_k) - \Gamma\nabla f(x^*))^{\mathrm{T}}\Gamma^{-1}(x_k - y_{k+1}) \\
&\qquad - 2(\Gamma\nabla f(x_k) - \Gamma\nabla f(x^*))^{\mathrm{T}}\Gamma^{-1}(x_k - x^*) \\
&\qquad + 2\begin{pmatrix} -\Gamma\nabla f(x^*) - (\tilde{\mathrm{H}}M)^{\mathrm{T}}s^* \\ +(g\circ\Gamma)(\Gamma^{-1}x^*) - (g\circ\Gamma)(\Gamma^{-1}y_{k+1}) \end{pmatrix}^{\mathrm{T}}\Gamma^{-1}(y_{k+1} - x^*).
\end{aligned}$$

Further, we have

$$\begin{aligned}
&\|x_{k+1} - x^*\|^2_{\Gamma^{-1}} + \|s_{k+1} - s^*\|^2_{\tilde{\Lambda}} \\
&\quad \leq \|x_k - x^*\|^2_{\Gamma^{-1}} + \|s_k - s^*\|^2_{\tilde{\Lambda}} - \|x_k - y_{k+1}\|^2_{\Gamma^{-1}} \\
&\qquad - \|s_{k+1} - s_k\|^2_{\tilde{\Lambda}(I - M\Gamma M^{\mathrm{T}}\tilde{\Lambda})} - \|x_{k+1} - y_{k+1}\|^2_{\Gamma^{-1}} \\
&\qquad + \Big\|(\tilde{\mathrm{H}}M)^{\mathrm{T}}(s_{k+1} - s_k)\Big\|^2_{\Gamma^{-1}} + \|y_{k+1} - x_{k+1}\|^2_{\Gamma^{-1}} \\
&\qquad - \Big\|y_{k+1} - x_{k+1} + (\tilde{\mathrm{H}}M)^{\mathrm{T}}(s_{k+1} - s_k)\Big\|^2_{\Gamma^{-1}} \\
&\qquad + \|\Gamma\nabla f(x_k) - \Gamma\nabla f(x^*)\|^2_{\Gamma^{-1}} + \|x_k - y_{k+1}\|^2_{\Gamma^{-1}} \\
&\qquad - \|x_k - y_{k+1} - (\Gamma\nabla f(x_k) - \Gamma\nabla f(x^*))\|^2_{\Gamma^{-1}} \\
&\qquad + 2\begin{pmatrix} \Big(-\Gamma\nabla f(x^*) - (\tilde{\mathrm{H}}M)^{\mathrm{T}}s^*\Big)^{\mathrm{T}}(y_{k+1} - x^*) \\ +(g\circ\Gamma)(\Gamma^{-1}x^*) - (g\circ\Gamma)(\Gamma^{-1}y_{k+1}) \end{pmatrix}.
\end{aligned} \tag{28}$$

Then we deal with some terms in the above inequality. For (20) combined with Lemma 1 can deduce

$$-\Gamma\nabla f(x^*) - (\tilde{\mathrm{H}}M)^{\mathrm{T}}s^* \in \Gamma\partial g(x^*) = \partial(g\circ\Gamma)\Big(\Gamma^{-1}x^*\Big).$$

Further using subdifferential properties, we have

$$\begin{aligned}&\left(\Gamma^{-1}y_{k+1}-\Gamma^{-1}x^*\right)^{\mathrm{T}}\left(-\Gamma\nabla f(x^*)-\left(\tilde{\mathrm{H}}M\right)^{\mathrm{T}}s^*\right)\\&\quad+(g\circ\Gamma)\left(\Gamma^{-1}x^*\right)-(g\circ\Gamma)\left(\Gamma^{-1}y_{k+1}\right)\le 0.\end{aligned}$$

Meanwhile, because ∇f_i is $1/\beta_i$-strongly monotone, there is

$$-(\nabla f(x_k)-\nabla f(x^*))^{\mathrm{T}}(x_k-x^*)\le-\|\nabla f(x_k)-\nabla f(x^*)\|_\beta^2. \tag{29}$$

Bring the above results back to (28) and get (27). □

Lemma 7. *Under Assumptions 1–4,* $\{||u_k-u^*||_P^2\}$ *is non-increasing and* $\lim_{k\to\infty}\|u_{k+1}-u_k\|_P^2=0$.

Proof. If Assumption 4 holds, we can deduce that $\{||u_k-u^*||_P^2\}$ satisfies non-increasing operators.

Sum (27) over k from 0 to N to obtain

$$\begin{aligned}&\|u_{N+1}-u^*\|_P^2\\&\le\|u_0-u^*\|_P^2-\sum_{k=0}^{n}\|s_{k+1}-s_k\|^2_{\tilde{\Lambda}\left(I-M\Gamma M^{\mathrm{T}}\tilde{\Lambda}\right)}\\&\quad-\sum_{k=0}^{n}\left\|(x_{k+1}-y_{k+1})+(\tilde{\mathrm{H}}M\Gamma)^{\mathrm{T}}(s_{k+1}-s_k)\right\|^2_{\Gamma^{-1}}\\&\quad-\sum_{k=0}^{n}\|x_k-y_{k+1}-(\Gamma\nabla f(x_k)-\Gamma\nabla f(x^*))\|^2_{\Gamma^{-1}}\\&\quad-\sum_{k=0}^{n}\|\nabla f(x_k)-\nabla f(x^*)\|^2_{2\beta-\Gamma}.\end{aligned}$$

When N tends to infinity, we can get

$$\begin{aligned}&\sum_{k=0}^{\infty}\|s_{k+1}-s_k\|_{\tilde{\Lambda}\left(I-M\Gamma M^{\mathrm{T}}\tilde{\Lambda}\right)}<\infty,\\&\sum_{k=0}^{\infty}\left\|(x_{k+1}-y_{k+1})+(\tilde{\mathrm{H}}M\Gamma)^{\mathrm{T}}(s_{k+1}-s_k)\right\|<\infty,\\&\sum_{k=0}^{\infty}\|x_k-y_{k+1}-(\Gamma\nabla f(x_k)-\Gamma\nabla f(x^*))\|<\infty,\\&\sum_{k=0}^{\infty}\|\nabla f(x_k)-\nabla f(x^*)\|<\infty.\end{aligned}$$

This means

$$\lim_{k\to\infty}\|s_{k+1}-s_k\|_{\tilde{\Lambda}\left(I-M\Gamma M^{\mathrm{T}}\tilde{\Lambda}\right)}=0, \tag{30}$$

$$\lim_{k\to\infty}\left\|(x_{k+1}-y_{k+1})+(\tilde{\mathrm{H}}M\Gamma)^{\mathrm{T}}(s_{k+1}-s_k)\right\|=0, \tag{31}$$

$$\lim_{k\to\infty}\|x_k-y_{k+1}-(\Gamma\nabla f(x_k)-\Gamma\nabla f(x^*))\|=0, \tag{32}$$

$$\lim_{k\to\infty}\|\nabla f(x^*)-\nabla f(x_k)\|=0. \tag{33}$$

Next, according to (32) and (33), we obtain

$$\lim_{k\to\infty}\|x_k-y_{k+1}\|=0. \tag{34}$$

Meanwhile, if Assumption 4 holds, $I - M\Gamma M^{\mathrm{T}}\tilde{\Lambda}$ is a symmetric positive definite. Therefore, we can get

$$\lim_{k\to\infty} \|s_{k+1} - s_k\| = 0. \tag{35}$$

According to (31) and (35) we obtain $\lim_{k\to\infty}\|x_{k+1} - y_{k+1}\| = 0$. Combining with (34), we get

$$\lim_{k\to\infty} \|x_{k+1} - x_k\|^2 = 0. \tag{36}$$

Then according to (35) and (36), we get $\lim_{k\to\infty}\|u_{k+1} - u_k\|^2 = 0$. □

Next, we give the following theorem to prove the convergence of Algorithm 1.

Theorem 1. *Under Assumptions 1–4, $\{x_k\}$ and $\{u_k\}$ converge to the optimal solution of (2) and the fixed points of T, respectively.*

Proof. Because $prox_f$ and $I - prox_f$ are firmly nonexpansive, T is continuous. Then, $\lim_{k\to\infty}\|u_{k+1} - u_k\|_P^2 = 0$ and the sequence $\left\{||u_k - u^*||_P^2\right\}$ satisfies non-increasing are obtained from Lemma 7. Based on Lemma 3, the sequence $\{u_k\}$ converges to a fixed point of T. According to Lemma 4, it can be concluded that $\{x_k\}$ converges to a solution to (2). □

At the same time, we also give the following theorem to prove the convergence of Algorithm 2.

Theorem 2. *Under Assumptions 1–4, relative to the solution set $\mathcal{S}$, the sequence $\{u_k\}_{k\geq k_0}, k_0 \in \mathbb{N}$ satisfies $\Pi^{-1}P$ stochastic Fejér monotonicity [27]:*

$$\begin{aligned}&\mathbb{E}\left[\|u_{k+1} - u^*\|_{\Pi^{-1}P}^2\right]\\ &\quad\leq \|u_k - u^*\|_{\Pi^{-1}P}^2 - \|s_{k+1} - s_k\|_{\tilde{\Lambda}\left(I - M\Gamma M^{\mathrm{T}}\tilde{\Lambda}\right)}^2 - \|\nabla f(x_k) - \nabla f(x^*)\|_{2\beta-\Gamma}^2.\end{aligned} \tag{37}$$

Further, the sequence $\{u_k\}_{k\geq k_0}$ converges almost surely to some $u^* \in \mathcal{S}$.

Proof. Before proving, we give some definitions. Here $\Pi = \sum_{i=1}^m p_i P_i$ denotes the probability matrix, and $\mathbb{E}[\cdot|\mathcal{F}_k]$ is ca onditional expectation, and its abbreviation is $\mathbb{E}_k[\cdot]$, where $\mathcal{F}_k$ represents the filtration generated by $\left(\xi^1, \ldots, \xi^k\right)$. We use $E_k = \sum_{i=1}^m \xi_i^k P_i$ to map the components of $\left(\mathbb{R}^{(2|\mathcal{E}|+m)n}, \mathcal{F}_{k-1}\right)$ to $\left(\mathbb{R}^{(2|\mathcal{E}|+m)n}, \mathcal{F}_k\right)$.

Based on the definition of ξ^k, we have $\mathbb{E} \circ (E_{k+1}) = \Pi$.

Using the idempotent property of E_k, we have

$$\begin{aligned}&\mathbb{E}\left[\|u_{k+1} - u^*\|_{\Pi^{-1}P}^2\right]\\ &\quad= \mathbb{E}\left[\|u_k + E_{k+1}(Tu_k - u_k) - u^*\|_{\Pi^{-1}P}^2\right]\\ &\quad= \mathbb{E}\begin{bmatrix}\|u_k - u^*\|_{\Pi^{-1}P}^2 + \|E_{k+1}(Tu_k - u_k)\|_{\Pi^{-1}P}^2\\ +2(u_k - u^*)^{\mathrm{T}}\Pi^{-1}P(E_{k+1}(Tu_k - u_k))\end{bmatrix}\\ &\quad= \|u_k - u^*\|_{\Pi^{-1}P}^2 + \|Tu_k - u_k\|_P^2 + 2(u_k - u^*)^{\mathrm{T}}P(Tu_k - u_k).\end{aligned}$$

Then according to Lemma 6 and (23), we get

$$\begin{aligned}
&\mathbb{E}\left[\|u_{k+1}-u^*\|^2_{\Pi^{-1}P}\right]\\
&\quad=\|u_k-u^*\|^2_{\Pi^{-1}P}+\|Tu_k-u^*\|^2_P-\|u_k-u^*\|^2_P\\
&\quad\le\|u_k-u^*\|^2_{\Pi^{-1}P}-\|s_{k+1}-s_k\|^2_{\tilde{\Lambda}\left(I-M\Gamma M^{\mathrm{T}}\tilde{\Lambda}\right)}-\|\nabla f(x_k)-\nabla f(x^*)\|^2_{2\beta-\Gamma}.
\end{aligned}$$

Therefore, if Assumption 4 holds, we can obtain the convergence of (37) according to [28] Th. 3, [27] Prop. 2.3, and the Robbins–Siegmund lemma in [29]. □

5. Numerical Experiments

5.1. Case Study I: Performance Examination

We present the effectiveness of the algorithms in this section by solving a class of quadratic programming problems on undirected networks. The network topology is shown in Figure 1.

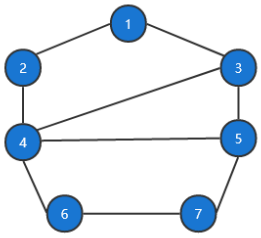


Figure 1. Graph topology.

The quadratic programming problem model is as follows:

$$\begin{aligned}
&\min_{x_1,\cdots,x_m}\sum_{i=1}^{m}f_i(x_i)=x_i^{\mathrm{T}}V_ix_i+b_i^{\mathrm{T}}x_i\\
&\text{s. t. } x_i^{\min}\le x_i\le x_i^{\max},\ i=1,\cdots,m,\\
&\qquad x_i=x_j,\ i=1,\cdots,m,\ (i,j)\in\mathcal{E},
\end{aligned}\tag{38}$$

where x_i is the decision variable of each agent. Matrix V_i in the objective function is a diagonal matrix, and its elements are randomly selected in $[-8, 8]$, and the elements of vector b_i are randomly selected in $[-10, -5]$. For the box constraint of x_i, the range of x_i^{min} is $[-10, -5]$, and the range of x_i^{max} is $[5, 10]$.

To solve problem (38), we need to convert the problem into the form of problem (8). Defining the set $X_i=\left\{e\in\mathbb{R}^2\middle|x_i^m\le e\le x_i^M\right\}$ and defining the indicator function $\delta_{X_i}(x_i)$, then we can get the following problem:

$$\min_{x_i\in\mathbb{R}^2}\sum_{i=1}^{m}f_i(x_i)+\delta_{X_i}(x_i)+\sum_{i=1}^{m}\sum_{(i,j)\in\mathcal{E}}\delta_{C_{(i,j)}}\left(M_{(i,j)}x\right).$$

Figure 2a shows that the agent finally converges to a consistent state through synchronous Algorithm 1. In Figure 2b, we use asynchronous Algorithm 2 with activation probability $p_i = 0.2$ to describe the state of the agent under the same parameter conditions.

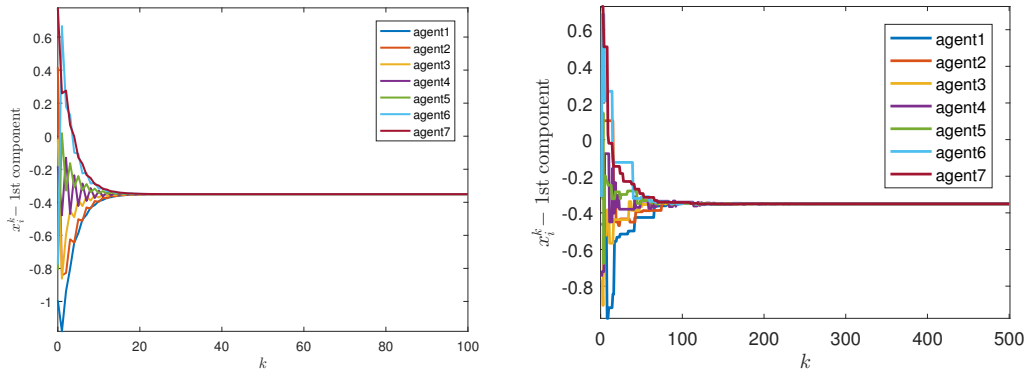


Figure 2. Convergence results of the two algorithms. (**a**) Algorithm 1. (**b**) Algorithm 2.

In Figure 3, the performance of both proposed algorithms is depicted through a comparison with existing algorithms, i.e., an ADMM-based method [30], TriPD-Dist, and its asynchronous version [2]. It can be shown that Algorithm 1 outperforms the ADMM-based method and TriPD-Dist, and the proposed asynchronous algorithm (Algorithm 2) also has a faster convergence speed than asynchronous TriPD-Dist, mainly by estimating the logarithmic values of $1/m \cdot \sum_{i=1}^{m} \|x_i^k - \tilde{x}^*\|$.

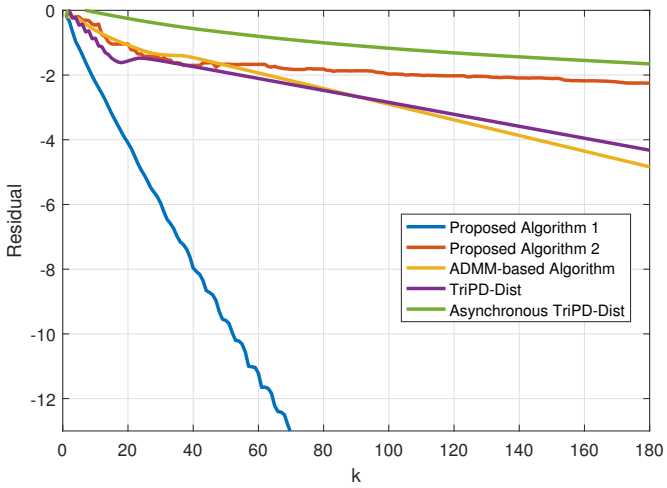


Figure 3. Performance comparison.

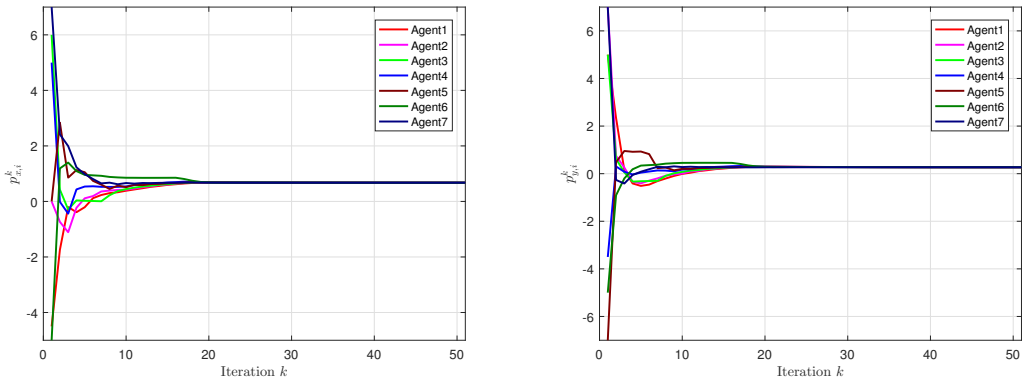
5.2. Case Study II: First-Order Dynamics System

In this subsection, we apply the proposed synchronous algorithm to solve a first-order dynamics system problem in a 2-D space [31], where each agent has its own cost function $f_i(\tilde{p}) = \|\tilde{p} - \tilde{p}_{x,i}\|^2 + \|\tilde{p} - \tilde{p}_{y,i}\|^2$, with the action response $\tilde{p} = [\tilde{p}_x, \tilde{p}_y]^{\mathrm{T}}$, and the private reference positions $\bar{p}_{x,i} = [i - 3.5, 0]^{\mathrm{T}}$ and $\bar{p}_{y,i} = [0, i - 3.5]^{\mathrm{T}}$. The goal of the considered problem is that all agents cooperatively find the optimal position $\tilde{p}$ under the local constraints $\Omega_i = \left\{ \tilde{p} \in \mathbb{R}^2 \middle| \left\| \tilde{p} - \bar{p}_i^0 \right\|^2 \leq 64 \right\}$, where $\bar{p}_i^0$ is the initial position of agent $i \in \{1, 2, 3, 4, 5, 6, 7\}$. Let $\bar{p}_1^0 = [-4, 5.5]^{\mathrm{T}}$, $\bar{p}_2^0 = [0, 7]^{\mathrm{T}}$, $\bar{p}_3^0 = [6, 5]^{\mathrm{T}}$, $\bar{p}_4^0 = [5, -3.5]^{\mathrm{T}}$, $\bar{p}_5^0 = [0, -7]^{\mathrm{T}}$, $\bar{p}_6^0 = [-5, -5]^{\mathrm{T}}$, $\bar{p}_7^0 = [7, 7]^{\mathrm{T}}$, then the distributed problem can be formulated as

$$\begin{aligned} \min_{p_1, \ldots, p_m} \quad & \sum_{i=1}^{m} \|p_i - \bar{p}_{x,i}\|^2 + \left\|p_i - \bar{p}_{y,i}\right\|^2 + \delta_{\Omega_i}(p_i), \\ \text{s.t.} \quad & p_i = p_j, \ (i,j) \in \mathcal{E}, \end{aligned}$$

where $p_i \in \mathbb{R}^2$ is the local estimation action for $\tilde{p}$. In light of (1), we can set $g_i(p_i) = \delta_{\Omega_i}(p_i)$. The selections of step-sizes are the same as that of Case Study I.

The results are described in Figures 4 and 5. To be specific, Figure 4a,b reflect the trajectories of $p_i = [p_{x,i}, p_{y,i}]^{\mathrm{T}}$. Figure 5 depicts the motions of the entire system over iterations, where the optimal position $\tilde{p}^* = [0.6743, 0.2711]^{\mathrm{T}}$ is marked by a cross at the intersection of two star lines, the circles with a dotted line are the corresponding motion areas of agents, and the solid ones are the initial positions.



(**a**) (**b**)

Figure 4. Evaluations of positions. (**a**) Evaluations of $p^k_{x,i}$. (**b**) Evaluations of $p^k_{y,i}$.

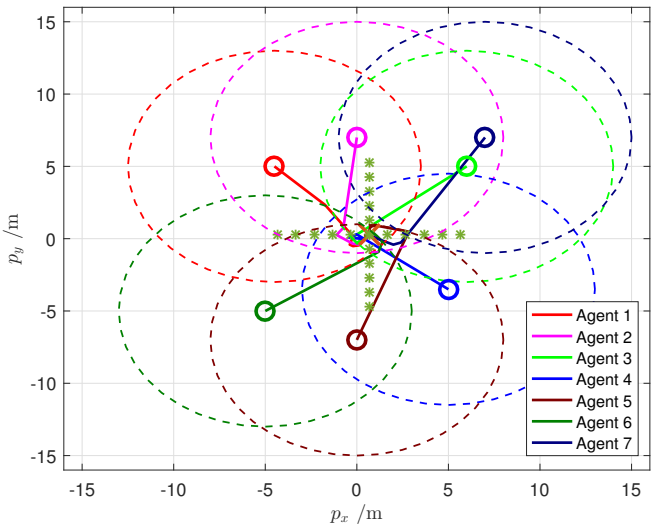


Figure 5. Motions of all agents in the 2-D space.

6. Conclusions

This paper mainly studies a class of distributed composite optimization problems with non-smooth convex functions. To solve this kind of problem, this paper proposes two completely distributed algorithms. At the same time, the algorithms are verified in theory and simulation. However, there are still some aspects worthy of improvement in this paper. For example, in the network structure, we can consider expanding from an undirected graph to a directed graph, and we can also combine it with more practical application scenarios, such as resource allocation.

Author Contributions: Y.S.: Data curation, Formal analysis, Investigation, Methodology, Resources, Software, Supervision, Writing—original draft. L.R.: Data curation, Formal analysis, Software. J.T.: Formal analysis, Software, Writing—original draft. X.W.: Methodology, Software, Formal analysis. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Li, Z.; Shi, W.; Yan, M. A decentralized proximal-gradient method with network independent step-sizes and separated convergence rates. *IEEE Trans. Signal Process.* **2019**, *67*, 4494–4506. [CrossRef]
2. Latafat, P.; Freris, N.M.; Patrinos, P. A new randomized block-coordinate primal-dual proximal algorithm for distributed optimization. *IEEE Trans. Autom. Control* **2019**, *64*, 4050–4065. [CrossRef]
3. Bai, L.; Ye, M.; Sun, C.; Hu, G. Distributed economic dispatch control via saddle point dynamics and consensus algorithms. *IEEE Trans. Control Syst. Technol.* **2019**, *27*, 898–905. [CrossRef]
4. Jin, B.; Li, H.; Yan, W.; Cao, M. Distributed model predictive control and optimization for linear systems with global constraints and time-varying communication. *IEEE Trans. Autom. Control* **2020**, *66*, 3393–3400. [CrossRef]
5. Boyd, S.; Parikh, N.; Chu, E.; Peleato, B.; Eckstein, J. Distributed optimization and statistical learning via alternating direction method of multipliers. *Found. Trends Mach. Learn.* **2011**, *3*, 1–122. [CrossRef]
6. Olshevsky, A. Linear time average consensus and distributed optimization on fixed graphs. *SIAM J. Control. Optim.* **2017**, *55*, 3990–4014. [CrossRef]
7. Nedić, A.; Ozdaglar, A. Distributed subgradient methods for multi-agent optimization. *IEEE Trans. Autom. Control* **2009**, *54*, 48–61. [CrossRef]
8. Shi, W.; Ling, Q.; Wu, G.; Yin, W. Extra: An exact first-order algorithm for decentralized consensus optimization. *SIAM J. Optim.* **2015**, *25*, 944–966. [CrossRef]
9. Qu, G.; Li, N. Harnessing smoothness to accelerate distributed optimization. *IEEE Trans. Control. Netw. Syst.* **2018**, *5*, 1245–1260. [CrossRef]
10. Chang, T.H.; Hong, M.Y.; Wang, X.F. Multi-agent distributed optimization via inexact consensus ADMM. *IEEE Trans. Signal Process.* **2015**, *63*, 482–497. [CrossRef]
11. Iutzeler, F.; Bianchi, P.; Ciblat, P.; Hachem, W. Explicit convergence rate of a distributed alternating direction method of multipliers. *IEEE Trans. Autom. Control* **2016**, *61*, 892–904. [CrossRef]
12. Shi, W.; Ling, Q.; Yuan, K.; Wu, G.; Yin, W.T. On the linear convergence of the ADMM in decentralized consensus optimization. *IEEE Trans. Signal Process.* **2014**, *62*, 1750–1761. [CrossRef]
13. Wei, E.; Ozdaglar, A. Distributed alternating direction method of multipliers. In *The 51st IEEE Conference on Decision and Control*; IEEE: Maui, HI, USA, 2012; pp. 5445–5450.
14. Chen, A.I.; Ozdaglar, A. A fast distributed proximal-gradient method. In Proceedings of the Annual Allerton Conference on Communication, Control, and Computing, Monticello, IL, USA, 1–5 October 2012; pp. 601–608.
15. Shi, W.; Ling, Q.; Wu, G.; Yin, W. A proximal gradient algorithm for decentralized composite optimization. *IEEE Trans. Signal Process.* **2015**, *63*, 6013–6023. [CrossRef]
16. Mao, X.; Yuan, K.; Hu, Y.; Gu, Y.; Sayed, A.; Walkman, W.Y. A communication-efficient random-walk algorithm for decentralized optimization. *IEEE Trans. Signal Process.* **2020**, *68*, 2513–2528. [CrossRef]
17. Zeng, J.; He, T.; Wang, M. A fast proximal gradient algorithm for decentralized composite optimization over directed networks. *Syst. Control. Lett.* **2017**, *107*, 36–43. [CrossRef]
18. Condat, L. A primal-dual splitting method for convex optimization involving lipschitzian, proximable and linear composite terms. *J. Optim. Theory Appl.* **2013**, *158*, 460–479. [CrossRef]
19. Chen, P.; Huang, J.; Zhang, X. A primal-dual fixed point algorithm for minimization of the sum of three convex separable functions. *Fixed Point Theory Appl.* **2016**. [CrossRef]
20. Latafat, P.; Patrinos, P. Asymmetric forward-backward-adjoint splitting for solving monotone inclusions involving three operators. *Comput. Optim. Appl.* **2017**, *68*, 57–93. [CrossRef]
21. Yan, M. A new primal-dual algorithm for minimizing the sum of three functions with a linear operator. *J. Sci. Comput.* **2018**, *76*, 1698–1717. [CrossRef]
22. Boyd, S.; Ghosh, A.; Prabhakar, B.; Shah, D. Randomized gossip algorithms. *IEEE Trans. Inf. Theory* **2006**, *52*, 2508–2530. [CrossRef]
23. Ren, X.; Li, D.; Xi, Y.; Shao, H. Distributed subgradient algorithm for multi-agent optimization with dynamic stepsize. *IEEE/CAA J. Autom. Sin.* **2021**, *8*, 1451–1464. [CrossRef]
24. Micchelli, C.A.; Shen, L.; Xu, Y. Proximity algorithms for image models: Denoising. *Inverse Probl.* **2011**, *27*, 45009. [CrossRef]

25. Bauschke, H.H.; Combettes, P.L. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*; Springer: New York, NY, USA, 2011.
26. Hong, M.; Chang, T.-H. Stochastic proximal gradient consensus over random networks. *IEEE Trans. Signal Process.* **2017**, *65*, 2933–2948. [CrossRef]
27. Combettes, P.L.; Pesquet, J.C. Stochastic quasi-Fejér block- coordinate fixed point iterations with random sweeping. *SIAM J. Optim.* **2015**, *25*, 1221–1248. [CrossRef]
28. Bianchi, P.; Hachem, W.; Iutzeler, F. A coordinate descent primal-dual algorithm and application to distributed asynchronous optimization. *IEEE Trans. Autom. Control* **2016**, *61*, 2947–2957. [CrossRef]
29. Robbins, H.; Siegmund, D. A convergence theorem for non negative almost supermartingales and some applications. In *Herbert Robbins Selected Papers*; Lai, T.L., Siegmund, D., Eds.; Springer: New York, NY, USA, 1985; pp. 111–135.
30. Aybat, N.S.; Wang, Z.; Lin, T.; Ma, S. Distributed linearized alternating direction method of multipliers for composite convex consensus optimization. *IEEE Trans. Autom. Control* **2018**, *63*, 5–20. [CrossRef]
31. Li, H.; Su, E.; Wang, C.; Liu, J.; Xia, D. A primal-dual forward-backward splitting algorithm for distributed convex optimization. *IEEE Trans. Emerg. Top. Comput. Intell.* **2021**. [CrossRef]