A conditional gradient algorithm for distributed online optimization in networks

Xiuyu Shen | Dequan Li | Runyue Fang | Qiao Dong

School of Mathematics and Big Data, Anhui University of Science and Technology, Huainan 232001, China

Correspondence
Dequan Li, School of Mathematics and Big Data, Anhui University of Science and Technology, Huainan, 232001, China.
Email: leedqcpp@126.com

Funding information
National Natural Science Foundation of China, Grant/Award Number: 61472003

Abstract
This paper addresses a network of computing nodes aiming to solve an online convex optimisation problem in a distributed manner, that is, by means of the local estimation and communication, without any central coordinator. An online distributed conditional gradient algorithm based on the conditional gradient is developed, which can effectively tackle the problem of high time complexity of the distributed online optimisation. The proposed algorithm allows the global objective function to be decomposed into the sum of the local objective functions, and nodes collectively minimise the sum of local time-varying objective functions while the communication pattern among nodes is captured as a connected undirected graph. By adding a regularisation term to the local objective function of each node, the proposed algorithm constructs a new time-varying objective function. The proposed algorithm also utilises the local linear optimisation oracle to replace the projection operation such that the regret bound of the algorithm can be effectively improved. By introducing the nominal regret and the global regret, the convergence properties of the proposed algorithm are also theoretically analysed. It is shown that, if the objective function of each agent is strongly convex and smooth, these two types of regrets grow sublinearly with the order of $O(\log T)$, where $T$ is the time horizon. Numerical experiments also demonstrate the advantages of the proposed algorithm over existing distributed optimisation algorithms.

1 | INTRODUCTION

Distributed convex optimisation has been greatly concerned by researchers in many fields [1–7]. Classical problems such as distributed tracking, estimation and detection are also essentially optimisation problems [8]. The distributed optimisation problem is mainly to perform global optimisation tasks assigned to each node in the network. As each node has limited resources or partial information about the task, the nodes, therefore, collaborate to perform data collection and update the local estimation via sharing the collected information. Distributed optimisation imposes a lower computational burden on nodes and the networked system remains robust even if a node encounters a local failure, thus it can effectively overcome the defect of relying on a single information processing unit in a centralised scenario.

Distributed optimisation has been widely applied to the case of time-invariant cost functions [9, 10]. However, in practical applications, distributed network systems are often in a dynamic and uncertain environment. For example, consider the issue of tracking moving objects, the purpose of which is to track the position, velocity and acceleration of the objects. Such issues have always been the main focus of online learning in the field of machine learning. Therefore, integrating online optimisation with distributed optimisation and employing any variable cost function to represent the uncertainty of a multi-agent network system can be effective for real-time processing of dynamic data streams of network nodes.

With the rapid development of the distributed online optimisation technology, many traditional optimisation algorithms have been extended to distributed online cases. Some traditional optimisation algorithms such as gradient descent method and dual averaging method have been widely applied to the distributed online optimisation in recent years. For instance, [1, 13–15] introduce four classical methods: distributed online...
gradient descent (DOGD), distributed online alternative direction multiplier method, distributed online dual average (DODA) and distributed online conditional gradient (DOCG).

The conditional gradient method (also known as Frank–Wolfe, FW) proposed in [16] is essentially a first-order optimisation method, which can theoretically achieve a lower convergence rate than other effective optimisation algorithms (such as the Interior-point polynomial algorithm [19]). Compared with the classic projection gradient algorithms [17, 18], the FW algorithm is more attractive due to its projection-free property and ability to handle structural constraints. To be specific, the projection gradient algorithm can make the current iteration point far away from the feasible set of the optimisation problem when performing an optimisation operation and thus obtain an infeasible point. For such a case, it is necessary to regain the feasibility of the gradient algorithm via projection operation. However, the projection means that a point must be located in the feasible set, and the feasible point is usually required to be the shortest distance from the current infeasible point, which is essentially equivalent to solving a convex quadratic programming problem. However, solving such a convex quadratic programming is a hard nut to crack, especially for the high-dimensional constrained optimisation problem, which makes it necessary to carry out effective linear optimisation. Moreover, the FW method has been proved to be a powerful tool for solving large-scale optimisation problems since it can effectively avoid critical issues such as the difficulty of calculating orthogonal projections in the first-order optimisation method.

Therefore, by introducing local linear optimisation oracle into the original conditional gradient algorithm, we present an online distributed conditional gradient algorithm (for short, ODCG), which also extends the conditional gradient online optimisation algorithm to the distributed setting. The regret bound seminally proposed in [20] is now proven to be an effective tool for characterising the performance of an online learning algorithm, but it is noted that the pros and cons of online algorithms mainly depend on the time complexity of regret bound.

The main contributions of this paper are summarised briefly in the following: (1) Firstly, in comparison with the algorithms in [1] and [14], our proposed algorithm constructs a new time-varying objective function via adding a regularisation term to the local cost function of each node; (2) secondly, different from the algorithm proposed in [15], our proposed algorithm considers to find the next iteration direction by using local linear optimisation oracle to improve the regret of the algorithm; (3) thirdly, we also establish convergence properties of the proposed algorithm via introducing the nominal regret and the global regret, which can not only effectively reveal all the local regrets of the entire network but also characterise the quality of local estimations from the perspective of the entire network.

The remainder of this paper is organised as follows. In Section 2, we describe the problem framework and propose an online distributed conditional gradient algorithm. In Section 3, we provide a theoretical analysis of the regret bound of the proposed algorithm in the distributed online convex programming. Empirically, our algorithm outperforms other existing algorithms on the L2 regularised linear support vector machine problem, as shown in Section 4. Finally, Section 5 concludes the paper.

Notations and Terminologies: Throughout the whole paper, we use $\mathbb{R}^d$ to denote the $d$-dimensional real space, and $\mathbb{R}^+\times \mathbb{R}^+$ to represent positive real number, and $\mathcal{V}$ to denote gradient operator. We also employ $[n]$ to denote the set $\{1, 2, \ldots, n\}$ for any integer $n$ and $\| \cdot \|$ to represent the Euclidean norm. Given a pair of vectors $x, y \in \mathbb{R}^d$, $(x, y)$ denotes the standard Euclidean inner product. In addition, $\mathcal{K} \subseteq \mathbb{R}^d$ denotes a closed convex subset of $\mathbb{R}^d$, and all the vectors are in column format.

## 2 PROBLEM STATEMENT AND ALGORITHM

### 2.1 Distributed online convex optimisation

We begin by stating the problem of online convex optimisation for one learner and then display the networked version, which is the focus of the paper. Suppose we have a sequence of convex objective functions $\{f_i\}_{i=1}^T$, where $f_i : \mathcal{K} \rightarrow \mathbb{R}$ for each $i \in \{1, \ldots, T\}$. At each round $t \in \{1, \ldots, T\}$, a decision maker chooses an action $x_t \in \mathcal{K}$ and after committing to this decision, a convex objective function $f_t : \mathcal{K} \rightarrow \mathbb{R}$ is revealed. Consequently, the decision maker is faced with a loss of $f_t(x_t)$. In this scenario, we consider the centralised online optimisation problem as follows

$$\min_{x \in \mathcal{K}} \sum_{t=1}^T f_t(x_t),$$

where $f_t : \mathcal{K} \rightarrow \mathbb{R}$ is a time-varying global objective function.

We now review the setup for a distributed version of the online optimisation problem [11, 12, 14]. Consider a network of $n$ agents communicating with each other over a connected undirected graph denoted by $\mathcal{G} = (V, E)$, with vertex set $V = [n]$ and edge set $E \subseteq V \times V$. At each round $t \in \{1, \ldots, T\}$, each agent $i \in V$ chooses its state $x_{it} \in \mathcal{K}$. After this, a locally convex objective function $f_{it} : \mathcal{K} \rightarrow \mathbb{R}$ is revealed, and the agent suffers the loss $f_{it}(x_{it})$. In this scenario, the whole network at time $t$ aims to minimise the objective function

$$f_t(x) = \sum_{i=1}^n f_{it}(x),$$

which is distributed among agents and is revealed when agents have chosen their states. It is worth noting that each agent $i$ only knows its own objective function $f_{it}(x)$ and cannot have access to the function $f_t(x)$. Therefore, across the time period from $t = 1$ to $T$, the goal of the whole network is to minimise the objective function $f(x) = \sum_{t=1}^T f_t(x)$. In this scenario, we consider solving the following distributed online optimisation
problem

\[
\text{minimise} \sum_{i \in \mathcal{K}} \sum_{t=1}^{T} \sum_{i=1}^{n} f_{ij}(x),
\]

where \( f_{ij} : \mathcal{K} \to \mathbb{R} \) is a time-varying and continuously differentiable convex objective function.

2.2 | Regret

An online algorithm to optimise (1) should mimic the performance of its offline counterpart, and the gap between them is called the regret. If the offline problem is to minimise \( \sum_{t=1}^{T} \sum_{i=1}^{n} f_{ij}(x) \), then the regret is called a static regret [11]. The following two types of static regret bound [13] are adopted in the case of the distributed online learning.

The first category is the nominal regret bound, that is,

\[
R(T) = \sum_{i=1}^{n} \sum_{t=1}^{T} f_{ij}(x_i,t) - \sum_{i=1}^{n} \sum_{t=1}^{T} f_{ij}(x^*).
\]

The second type is the global regret bound, that is,

\[
R^i(T) = \sum_{i=1}^{T} \sum_{t=1}^{n} f_{ij}(x_{ij,t}) - \sum_{i=1}^{T} \sum_{t=1}^{n} f_{ij}(x^*),
\]

where \( x^* \in \arg \min_{x \in \mathcal{K}} \sum_{i=1}^{n} \sum_{t=1}^{T} f_{ij}(x) \).

Remark 1. Note that \( R(T) \) corresponds to all local regrets of the entire network. However, it may not effectively reflect the similarity among all local estimations, which implies that the difference among local estimations may be very large even if \( R(T) \) is small. Hence, it is necessary to introduce global regret, that is, \( R^i(T) \), which characterises the quality of local estimations from the perspective of the entire network. Moreover, in the case of the centralised online learning (\( n = 1 \)), the global regret and the nominal regret are equivalent, but they are utterly different for the case of the distributed online learning. Therefore, it is more significant to analyse the global regret than the nominal regret in the case of the distributed online learning.

2.3 | An online distributed conditional gradient algorithm

We are now in a position to state the proposed ODCG in Algorithm 1 to solve the optimisation problem (1).

For the sake of necessity, we introduce the following definition, which is a primary ingredient of our work.

Definition 1. (Local Linear Optimisation Oracle)[22]. We formally state a procedure \( \text{LLOO}(x, r, g) \), where \( x \in \mathcal{K}, r \in \mathbb{R}^+ \).

**ALGORITHM 1** Online Distributed Conditional Gradient Algorithm (ODCG)

1: Input: convex set \( \mathcal{K} \) and its diameter \( D \), maximum round number \( T \), upper bound of the gradient \( G \), parameter \( \rho \), step-sizes \( \{\alpha_t\}_{t=1}^{T} \) and the weighted adjacency matrix \( W \).
2: Initialise: \( x_1 \in \mathcal{K}, \forall i \in V \).
3: for \( t = 1, ..., T \) do
4: The adversary reveals: \( f_{ij}, \forall i \in V \).
5: Compute subgradients: \( \delta f_{ij}(x_{ij,t}) \), \( \forall i \in V \).
6: for Each node \( i \in V \) do
7: \( \hat{g}_i = \delta f_{ij} + g_i \).
8: \( \hat{g}_i = \sum_{j=1}^{n} W_{ij} \hat{g}_j \).
9: \( F_i(x) = \langle \hat{g}_i, x \rangle + \frac{1}{2} \| x - x_i \|^2 + \frac{1}{2} \| x_i - x \|^2 \).
10: \( p_i = \text{LLOO}(x_i, r_i, r, F_i(x_{ij,t})) \) with \( r_i = \frac{2(G + 2mD)}{\rho(\rho + D)} \).
11: \( x_{i+1} = x_i + \alpha_i(p_i - x_i) \).
12: end for
13: end for

\( \mathbb{R}^{+}, g \in \mathbb{R}^{d} \). It is called the Local Linear Optimisation Oracle parameter \( \rho \geq 1 \) for the feasible set \( \mathcal{K} \), if \( \text{LLOO}(x, r, g) \) returns a feasible point \( p \in \mathcal{K} \) such that:

1. \( \forall y \in B(x, r) \cap \mathcal{K} \), there holds \( y \cdot g \geq p \cdot g \)
2. \( \| x - p \| \leq \rho \cdot r \),

where \( B(x, r) \) denotes the Euclidean ball of radius \( r \) centred at \( x \).

The specific application of the local linear optimisation oracle to be extended to the distributed case is shown in detail in Algorithm 1.

The implementation of the ODCG algorithm is simple and as follows. Each node \( i \) firstly observes its loss function \( f_{ij} \) and suffers its loss \( f_{ij}(x_{ij,t}) \) at time \( t \). Subsequently, each node \( i \) computes its subgradient \( \hat{g}_i = \delta f_{ij}(x_{ij,t}) \) at time \( t \) (cf. Step 5 in Algorithm 1). To ensure full utilisation of the important information of nodes, the information interaction among nodes is carried out by weighted average. Here, we design the update expression of the original gradient to obtain the estimation of the gradient \( \hat{g}_i \) instead of directly performing the weighted average of the original gradient information \( g_i \) (cf. Steps 7 and 8 in Algorithm 1), which can ensure the security of the original information. Moreover, compared with the original standard online conditional gradient algorithm [21] and the algorithms in [1, 14], we define a new time-varying objective function \( F_i(x) \) via adding a new time-varying regularisation term to regularise the time-varying cost function (cf. Step 9 in Algorithm 1), and find next iteration direction by local linear optimisation oracle step given by Step 10 in Algorithm 1. Finally, each node computes its new update after interacting with its neighbouring nodes (cf. Steps 11 and 12 in Algorithm 1).

Remark 2. In Algorithm 1, the learning rate \( \alpha_i \) is positive and non-increasing. For a centralised case, a local linear
optimisation oracle is considered in [22] to find a minimiser of a linear problem over feasible set $\mathcal{K}$. Similarly, in the distributed case, our algorithm also introduces a local linear optimisation oracle (cf. line 10 in Algorithm 1), the purpose of which is to achieve a better regret bound. The efficient linear optimisation oracle is exploited to find a minimiser of a linear problem as in [22] over $\mathcal{K}$ such that our proposed algorithm can thus effectively improve the regret bound.

3 | REGRET ANALYSIS

In this section, we aim to establish convergence properties of the proposed ODCG algorithm under the following necessary assumptions.

**Assumption 1.** (a) The function $f(\lambda) : \mathcal{K} \rightarrow \mathbb{R}$ is $\beta$-smooth and $\mu$-strongly convex for the convex set $\mathcal{K}$. If $\forall x, y \in \mathcal{K}$, we have

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\beta}{2} \|y - x\|^2,$$

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\mu}{2} \|y - x\|^2.$$

Then, for $\forall x^* \in \mathcal{K}$, if $x^* = \arg\min_{x \in \mathcal{K}} f(x)$, we have

$$f(x) - f(x^*) \leq \frac{\mu}{2} \|x - x^*\|^2.$$

(b) Each function $f_i$ is Lipschitz continuous on $\mathcal{K}$ with a uniform constant $L$. That is, $\|f_i(x) - f_i(y)\| \leq L \|x - y\|$.

**Assumption 2.** (a) For $\forall x, y \in \mathcal{K}$, the diameter $\text{diam}(\mathcal{K}) = \text{snip} \|x - y\|$ of $\mathcal{K}$ is bounded by $D < \infty$.

(b) The subgradient of the convex function $f : \mathcal{K} \rightarrow \mathbb{R}$ is bounded, that is, $\|\nabla f\| \leq G$, where $G > 0$.

Assumptions 1 and 2 are common assumptions in distributed optimisation (e.g. [1, 15, 24]). In addition, many objective functions $f_i$ satisfy this type of Lipschitz condition. For instance, Assumption 1(b) holds for any convex function on a compact domain, or for a polyhedral function on an arbitrary domain [27].

**Assumption 3.** (a) The graph $G$ is connected and undirected.

(b) The weighted adjacency matrix $W$ of graph $G$ is symmetric and doubly stochastic, which satisfies $\sum_{i=1}^n W_{ij} = \sum_{j=1}^n W_{ij} = 1$, $W_{ii} > 0$ for all $i \in V$.

**Remark 3.** Assumption 3 about the weighted adjacency matrix $W$ is a common assumption for the distributed setting [1, 14, 15]. It can be seen that the aforementioned weight matrix $W$ has a simple eigenvalue one and all the other eigenvalues less than 1. Moreover, $W$ should be designed properly such that it is symmetric and satisfies $W_{ij} = W_{ji} > 0$ if $(i, j) \in E$, otherwise, $W_{ij} = W_{ji} = 0$.

3.1 | Auxiliary lemmas

In this part, we will give some lemmas that are useful for the regret analysis in the sequel.

Line 9 of Algorithm 1 defines a new regularisation function $F_{ij}(\lambda)$, assuming that the optimal solution of $F_{ij}(\lambda)$ is $x^*_{ij} = \arg\min_{x \in \mathcal{K}} F_{ij}(\lambda)$. Under Assumption 1 and the fact that $F_{ij}(\lambda)$ is $\mu(t + T_0)$-smooth and $\mu(t + T_0)$-strongly convex for $t = \{1, ..., T\}$, we can have

$$\|x_{ij} - x^*_{ij}\| \leq \sqrt{\frac{2}{\mu(t + T_0)} (F_{ij}(x_{ij}) - F_{ij}(x^*_{ij}))}.$$  (4)

The following Lemma 1 will illustrate that the function $F_{ij}(\lambda)$ is Lipschitz continuous on the convex set $\mathcal{K}$.

**Lemma 1.** Suppose Assumption 2 holds. For any $t = \{1, ..., T\}$, the function $F_{ij}(\lambda)$ is $G + 2\mu D$ Lipschitz continuous on the feasible set $\mathcal{K}$.

**Proof.** For $\forall y, z \in \mathcal{K}$, we have

$$|F_{ij}(y) - F_{ij}(z)| \leq \|\nabla F_{ij}(y) + \mu(y - x_{ij}) + \mu(y - x_{ij})\| (y - z) \leq \|\nabla F_{ij}(y) + \mu(y - x_{ij}) + \mu(y - x_{ij})\| (y - z) \leq (G + 2\mu D) \|y - z\|,$$

where the first inequality follows from the convexity of $F_{ij}(\lambda)$, the second inequality holds due to the property of the norm, and the last inequality uses Assumption 2.

The result in Lemma 1 turns out that $F_{ij}(\lambda)$ is Lipschitz continuous with constant of $G + 2\mu D$, which provides a mathematical condition for the analysis of the regret.

In addition, let $\tilde{F}_{ij}(\lambda) = \langle \tilde{g}_{ij}, x \rangle + \frac{\mu}{2} \|x - x_{ij}\|^2$ for $t = \{1, ..., T\}$, the following inequality can be derived by Lemma 1.

$$|\tilde{F}_{ij}(\lambda) - \tilde{F}_{ij}(\lambda)| \leq (G + \mu D) \|y - z\|, \quad \text{for} \forall y, z \in \mathcal{K}.$$  (5)

The above inequality shows that the function $\tilde{F}_{ij}(\lambda)$ is Lipschitz continuous, which will play an important role in the subsequent proof of Theorem 1.

Under the result of Lemma 1, we will provide the upper bound of $\|x_{ij} - x^*_{ij}\|$ in Lemma 2.

**Lemma 2.** Suppose Assumptions 1 and 2 hold. Let the local estimations $\{x_{1j}, x_{2j}, ..., x_{nj}\}_{n=1}^T$ be the sequence generated by Algorithm 1, then

$$\|x_{ij} - x^*_{ij}\| \leq \frac{2(\tilde{g}_{ij} + 2\mu D)}{\mu(t + T_0)}.$$  (2)
Proof. Due to the fact that \( F_{ij}(x) \) is Lipschitz continuous in Lemma 1, we obtain

\[
\left| F_{ij}(x_{ij}) - F_{ij}(\tilde{x}_{ij}) \right| \leq (G + 2\mu D) \left\| x_{ij} - \tilde{x}_{ij} \right\|.
\]

Hence, under Lemma 1 and (4), it yields

\[
\left\| x_{ij} - \tilde{x}_{ij} \right\| \leq \sqrt{\frac{2(G + 2\mu D)}{\mu(\nu + T_0)}} \left\| x_{ij} - \tilde{x}_{ij} \right\|.
\]

By the fact that \( \left\| x_{ij} - \tilde{x}_{ij} \right\| \geq 0 \), then we have

\[
\left\| x_{ij} - \tilde{x}_{ij} \right\| \leq \frac{2(G + 2\mu D)}{\mu(\nu + T_0)}.
\]

Lemma 2 shows the upper bound of \( \left\| x_{ij} - \tilde{x}_{ij} \right\| \), which provides a supporting condition for the derivation of the network error in the sequel.

Lemma 3 provides an upper bound on the deviation of the local estimation of each node at each iteration from their average value and thus reveals the network error.

**Lemma 3. (Network Error).** Suppose Assumption 3 holds. Let the local estimations \( \{x_{11}, x_{22}, \ldots, x_{nn}\} \) be the sequence generated by Algorithm 1 with the learning rate \( \alpha_i = \frac{a_i}{\sqrt{\nu}}, a > 0 \). Then, we have

\[
\left\| x_{ij} - \bar{x}_{ij} \right\| \leq \frac{2\sqrt{n\alpha(G + 2\mu D)}}{(1 - \sigma_2(W))\sqrt{\mu(\nu + T_0)}},
\]

where \( \bar{x}_{ij} = \frac{1}{n} \sum_{i=1}^{n} x_{ij}, \forall i \in V \), \( \sigma_2(W) \) denotes the second largest eigenvalue of matrix \( W \) and \( 1 - \sigma_2(W) \) denotes the corresponding spectral gap value.

**Proof.** Suppose that \( x_{ij+1} = \sum_{j=1}^{n} W_{ij}x_{ij} + \delta_{ij+1} \) and \( x_{ij+1} = x_{ij} + \alpha_i(p_{ij} - x_{ij}) \) are obtained from Algorithm 1, then we have

\[
\delta_{ij+1} = (x_{ij} + \alpha_i(p_{ij} - x_{ij})) - \sum_{j=1}^{n} W_{ij}x_{ij}.
\]

Therefore, the upper bound of \( \left\| \delta_{ij+1} \right\| \) can be given as follows

\[
\left\| \delta_{ij+1} \right\| = \left\| (x_{ij} + \alpha_i(p_{ij} - x_{ij})) - \sum_{j=1}^{n} W_{ij}x_{ij}\right\|
\]

\[
= \left\| (x_{ij} + \alpha_i(p_{ij} - x_{ij})) - x_{ij}\right\|
\]

\[
= \left\| \alpha_i(p_{ij} - x_{ij})\right\|
\]

\[
\leq \alpha_i \left\| x_{ij} - x_{ij}\right\|
\]

\[
\leq \frac{2\alpha(G + 2\mu D)}{\sqrt{\mu(\nu + T_0)}},
\]

where the first inequality holds due to the fact that \( p_{ij} \) is the vector with the smallest value on the feasible set and the second inequality follows from Lemma 2.

Using \( W^{r} \) to denote the rth power of \( W \), and \( W_{ij}^{r} \) to denote the rth row and jth column of the matrix \( W^{r} \), we can acquire the following recursive relation by simply algebraic operations, that is,

\[
x_{ij+1} = \sum_{r=1}^{i+1} \sum_{j=1}^{n} [W_{ij}^{r-1}]_{11} \delta_{ijr},
\]

\[
\bar{x}_{ij+1} = \sum_{r=1}^{i+1} \delta_{ir},
\]

where \( \delta_{ir} = \frac{1}{i} \sum_{j=1}^{i} \delta_{ijr}, \forall i \in V \).

Hence, the following bound of network error can be easily derived by (7)–(10)

\[
\left\| x_{ij+1} - \bar{x}_{ij+1} \right\| \leq \left\| \sum_{r=1}^{i+1} \sum_{j=1}^{n} [W_{ij}^{r-1}]_{11} \delta_{ijr} \right\|
\]

\[
\leq \left\| \sum_{r=1}^{i+1} \sum_{j=1}^{n} [W_{ij}^{r-1}]_{11} \right\| \frac{1}{n} \left\| \delta_{ir} \right\|
\]

\[
\leq \frac{2\alpha(G + 2\mu D)}{\sqrt{\mu(\nu + T_0)} \sqrt{n}} \sum_{r=1}^{i+1} \sigma_{2}^{r-1}(W),
\]

\[
\leq \frac{2\sqrt{n\alpha(G + 2\mu D)}}{(1 - \sigma_2(W))\sqrt{\mu(\nu + T_0)}},
\]

where the first inequality holds due to the property of the norm, the second inequality follows from the following property of mixing matrix [25],

\[
\sum_{j=1}^{n} [W_{ij}^{r}] - \frac{1}{n} \leq \sqrt{n}\sigma_{2}^{r}(W),
\]

and the last inequality holds due to \( \sum_{r=1}^{i+1} \sigma_{2}^{r}(W) < \frac{1}{1 - \sigma_2(W)} \).

So far, we complete the proof of the network error. Lemma 3 turns out that the error bound is related to the network parameter \( \sigma_2(W) \). According to the proof of Lemma 3, we can clearly recognise the network error of the proposed algorithm and further characterise the robustness of the proposed algorithm.

### 3.2 Regret bound

In this part, we will analyse the nominal regret and global regret of the ODCG algorithm under Lemmas 1–3, which are shown in Theorems 1 and 2.
Suppose Assumptions 1–3 hold. If local cost functions \( \{f_1, f_2, \ldots, f_n\} \) are \( \mu \)-strongly convex in Algorithm 1 with the learning rate \( \alpha_i = \frac{\mu}{\sqrt{T}}, \alpha > 0 \), then for all \( i \in V \) and \( t = \{1, \ldots, T\} \), the nominal regret satisfies

\[
R(T) \leq C_1 + C_2 \log T,
\]

where \( C_1 = \frac{nT_0\mu D^2}{2} \) and \( C_2 = \frac{2n(G+\mu D)(G+2\mu D)}{\mu} \).

**Proof.** Noting that \( F_i = T_0\mu \|x - x_{1,1}\|^2 \) and by the definition of \( \bar{f}_i(x) \), we obtain

\[
R(T) = \sum_i \sum_{t=1}^T (f_i(x_{i,t}) - f_i(x^*))
\]

\[
\leq \sum_i \sum_{t=1}^T \bar{g}_i(x_{i,t}) - \bar{g}_i(x^*) \mu \|x_{i,t} - x^*\|^2 - \|x_{i,t} - x_{i,1}\|^2)
\]

\[
= \sum_i \sum_{t=1}^T (\bar{f}_i(x_{i,t}) - \bar{f}_i(x^*)) \mu \|x_{i,t} - x^*\|^2 - \|x_{i,t} - x_{i,1}\|^2)
\]

\[
= \sum_i \sum_{t=1}^T (\bar{f}_i(x_{i,t}) - \bar{f}_i(x^*)) \mu \|x_{i,t} - x^*\|^2 - \|x_{i,t} - x_{i,1}\|^2)
\]

In addition, denote \( x_{i,t}^* = \arg \min_{x \in \mathcal{X}} F_i(x) \). Due to the fact that \( F_i(x) \) is strongly convex, then we have \( F_i(x_{i,t}^*) < F_i(x^*) \), that is, \( F_i(x_{i,t}^*) - F_i(x^*) < 0 \). Hence, we arrive at

\[
\sum_{t=1}^T F_i(x_{i,t}^*) - F_i(x^*)
\]

\[
= F_1(x_{i,1}^*) - F_1(x^*) + F_2(x_{i,2}^*) - F_2(x^*)
\]

\[
+ \cdots + F_{i,T}(x_{i,T}^*) - F_{i,T}(x^*) \leq 0.
\]

Now, by the definition of \( \bar{f}_i(x) \) and \( \bar{f}_i(x) \), together with (14), then we can get

\[
\sum_{t=1}^T \bar{f}_i(x_{i,t}^*) - \bar{f}_i(x^*)
\]

\[
= \sum_{t=1}^T \bar{f}_i(x_{i,t}^*) - \bar{f}_i(x^*)
\]

\[
+ T_0\mu \|x_{i,1} - x_{i,1}\|^2 - \|x^* - x_{i,1}\|^2) \leq 0.
\]

Next, the upper bound of \( \sum_{t=1}^T (\bar{f}_i(x_{i,t}^*) - \bar{f}_i(x^*)) \) is obtained by

\[
\sum_{t=1}^T \bar{f}_i(x_{i,t}^*) - \bar{f}_i(x^*)
\]

\[
\leq T_0\mu \|x^* - x_{i,1}\|^2 - \|x^* - x_{i,1}\|^2) \leq \frac{T_0\mu D^2}{2}.
\]

Now, summing (16) over \( i \in V \) gives

\[
\sum_{i=1}^n \sum_{t=1}^T (\bar{f}_i(x_{i,t}^*) - \bar{f}_i(x^*)) \leq \sum_{i=1}^n \frac{T_0\mu D^2}{2} = \frac{nT_0\mu D^2}{2}.
\]

Next, we turn our attention to the bound of \( \sum_{i=1}^n \sum_{t=1}^T (f_i(x_{i,t}) - f_i(x^*)) \). Since \( \bar{f}_i(x) \) is Lipschitz continuous and under Lemma 2, then we obtain

\[
|\bar{f}_i(x_{i,t}) - \bar{f}_i(x_{i,t}^*)| \leq (G + \mu D)\|x_{i,t} - x_{i,t}^*\| \leq \frac{2(G + \mu D)[G + 2\mu D]}{\mu(t + T_0)}.
\]

Now, summing (18) over \( i \in V \) and \( t = \{1, \ldots, T\} \), we have

\[
\sum_{i=1}^n \sum_{t=1}^T (\bar{f}_i(x_{i,t}) - \bar{f}_i(x_{i,t}^*)) \leq \sum_{i=1}^n \frac{2n(G + \mu D)[G + 2\mu D]}{\mu}\log T.
\]

Plugging (17) and (19) into (13) gives

\[
R(T) = \sum_{i=1}^n \sum_{t=1}^T (f_i(x_{i,t}) - f_i(x^*)) \leq \frac{nT_0\mu D^2}{2} + \frac{2n(G + \mu D)(G + 2\mu D)}{\mu}\log T.
\]

**Remark 4.** Theorem 1 implies that, under the condition of strongly convex, the nominal regret for Algorithm 1 has a sublinear growth of order \( O(\log T) \). The result in Theorem 1 also represents the sum of all local regrets of the entire network. However, it does not effectively reflect the similarity among all local parameter estimations, because there may exist a case that \( R(T) \) is small but the difference among local parameter estimations may be very large. On the other hand, the nominal regret fails to incentivise the collaboration among agents since agents are effectively independent and only need to learn strategies related to their local costs. Hence, it is necessary to introduce the global regret, that is, \( R'(T) \), which considers games where agents have an incentive to collaborate and also characterises the quality of local estimations from the perspective of the entire network. Theorem 2 will reveal the global regret bound.
Theorem 2. Suppose Assumptions 1–3 hold. If local cost functions \( \{f_1, f_2, \ldots, f_n\} \) are \( \mu \)-strongly convex in Algorithm 1 with the learning rate \( \alpha_t = \frac{2}{t}, \alpha > 0 \), then for \( \forall i \in V \) and \( t = \{1, \ldots, T\} \), the global regret satisfies

\[
R^i(T) \leq C_1 + C_2 \log T + C_3 \log T,
\]

where \( C_1 = \frac{n^2 \mu D^2}{2} \), \( C_2 = \frac{2\mu(G+\mu D)(G+2\mu D)}{\mu} \) and \( C_3 = \frac{4\alpha \sqrt{\alpha}(G+2\mu D)}{(1-\sigma_2(W)) \mu} \).

Proof. The difference between two types of regret defined in (2) and (3) is as follows

\[
R^i(T) - R(T) = \sum_{i=1}^{n} \sum_{t=1}^{T} \left( f_i(x_{i,t}) - f_i(x_{i,t}) \right)
\]

\[
\leq \sum_{i=1}^{n} \sum_{t=1}^{T} \langle g_{i,t}, x_{i,t} - x_{i,t} \rangle
\]

\[
\leq G \sum_{i=1}^{n} \sum_{t=1}^{T} \|x_{i,t} - x_{i,t}\|. \tag{21}
\]

Using Lemma 3, the last inequality in (21) can be further derived as follows

\[
\|x_{i,t} - x_{i,t}\| \leq \|x_{i,t} - x_{i,t}\| + \|x_{i,t} - x_{i,t}\|
\]

\[
\leq \frac{4\sqrt{\alpha}(G+2\mu D)}{(1-\sigma_2(W)) \sqrt{t}(t+T_0)}. \tag{22}
\]

Now, substituting (20) and (22) into (21), we have

\[
R^i(T) \leq R(T) + G \sum_{i=1}^{n} \sum_{t=1}^{T} \frac{4\sqrt{\alpha}(G+2\mu D)}{(1-\sigma_2(W)) \sqrt{t}(t+T_0)}
\]

\[
\leq \frac{nT_0 \mu D^2}{2} + \frac{2\mu(G+\mu D)(G+2\mu D)}{\mu} \log T
\]

\[
+ \frac{4\alpha \sqrt{\alpha}(G+2\mu D)}{(1-\sigma_2(W)) \mu} \log T. \tag{23}
\]

So far, we complete the proof of Theorem 2.

Remark 5. Theorem 2 indicates that the global regret for strongly convex functions also has a sublinear growth with order of \( O(\log T) \), which measures the quality of the entire network. It can also be seen that, under the case of the centralised online learning, the global regret and the nominal regret are equivalent, but they are absolutely different for the case of the distributed online learning. In addition, \( 1 - \sigma_2(W) \) in Theorem 2 represents the spectral gap, which is quite natural and, for many families of undirected graph, measures how fast the information can propagate within the network (the larger the spectral gap, the faster the convergence is). Our algorithm also extends the delicate local linear optimisation oracle in the centralised FW algorithm [22] to the distributed setting.

4 | NUMERICAL EXPERIMENTS

In this section, we numerically illustrate the performance of the ODCG algorithm on L2 regularised linear support vector machine problem (linear-SVM) using the synthetic data set and Mushroom data set, which are available from the SGDlibrary [26].

The objective is to consider solving the following online optimisation loss function

\[
f_i(w) = \frac{1}{n} \sum_{j=1}^{n} \left( \max \left( 0, 1 - y_{ij} \langle w^T, x_{ij} \rangle \right) \right)^2 + \frac{\lambda}{2} \|w\|^2,
\]

where \( n \) is the number of nodes, \( y_{ij} \) represents the label of the \( i \)th node at time \( t \), \( x_{ij} \in \mathbb{R}^d \) denotes a data example of the node \( i \) at time \( t \), \( w = [w_1^T, w_2^T, \ldots, w_l^T] \in \mathbb{R}^{l \times d} \) (\( l \) is the number of classes) is the weighting matrix to be optimised and \( \lambda \) is a regularisation parameter \( (\lambda \text{ is set as } 0.1) \).

In our experiments, we consider the \( \ell_1 \) ball \( K_{\ell_1} = \{x \in \mathbb{R}^d : \|x\|_1 \leq \rho\} \), where \( \|x\|_1 \) is the \( \ell_1 \) norm of \( x \), and use the Metropolis constant edge weight matrix \( W \) [28]. For other parameters, setting as follows: the diminishing step-sizes \( \alpha_t = \frac{1}{\sqrt{t}} \) and the parameter \( \mu = 0.1 \).

4.1 | Comparison with classical algorithms

To evaluate the performance of ODCG over its competing algorithms, we compare the ODCG with two classical algorithms over 10 nodes, that is, DOGD [1] and DODA [14]. The parameters we set for both DOGD and DODA are the same. The communication graph is set as described in Assumption 3.

To test the performance of the proposed algorithm in different data sets over its competing algorithms, we conduct our experiments on the synthetic data set and the Mushroom data set, respectively. The results are shown in Figures 1 and 2. It can be clearly observed from Figures 1 and 2 that, no matter in the synthetic data set or real data set, the algorithm we proposed performs better than the other two algorithms, which illustrates the necessity and usefulness of the proposed algorithm in distributed online learning frameworks.

4.2 | Comparison of different nodes

Besides, we compare the performance of the ODCG algorithm with different nodes. The ODCG is tested with \( n = 6, 26, 54 \). Each node computes its new update by running lines 7–12 in Algorithm 1.
Figure 3 depicts the variation of the costs or losses $f_i(w)$ of the proposed algorithm running on different nodes. In Figure 3, we can clearly observe that the costs decrease more slowly on larger graphs than on smaller graphs, which nicely confirms our theoretical results.

### 4.3 Comparison of different network topologies

Finally, we compare the performance of the ODCG under different choices of network topologies. Graphs are set as follows, which represent different levels of connectivity.

- Complete graph [15]. This graph shows the highest level of connectivity: all nodes are connected to each other.
- Cycle graph [15]. This graph represents the lowest level of connectivity: each node has only two close neighbours.
- Watts–Strogatz (WS). The generation of this graph depends on two tunable parameters: the average degree of the network $K$ and the rewiring probability $p$ [29]. Usually, the higher the rewiring probability, the better the connectivity of the graph [30]. We tune the parameters $K = 3$ and $p = 0.4$ to achieve an intermediate level of connectivity in our simulation.

We run experiments on the aforementioned three types of graphs with fixed 10 nodes. Figure 4 depicts the performance of the proposed algorithm under different network topologies.
It is interesting to notice that graphs with better connectivity result in faster convergence.

5 | CONCLUSION

This paper investigated the online optimisation problem over decentralised networks and proposed an online distributed conditional gradient algorithm. Then we detailed and analysed the convergence properties of the proposed algorithm and illustrated that both the global regret and the nominal regret have a sublinear growth with order of $O(\log T)$. Finally, the effectiveness of the proposed algorithm was also demonstrated by tackling a 1.2 regularised linear support vector machine problem, with the advantages over the existing algorithms being confirmed through numerical experiments.

It is worth noting that by assuming that the objective function of each agent is strongly convex and smooth (Assumption 1), we conducted the analysis of the regret and established the main results. However, in many practical applications, the objective function may be non-smooth. For the case of non-smooth objective functions, the authors of [31] proposed a distributed subgradient-free stochastic optimisation algorithm over time-varying network and proved that the consensus of estimations and the global minimisation can be achieved with probability one. The authors of [32] presented a distributed quasi-monotone subgradient algorithm for non-smooth convex optimisation over directed graphs. For the distributed non-smooth convex optimisation problems in second-order multiagent systems, the authors of [33] recently developed a novel continuous-time distributed proximal-gradient algorithms with derivative feedback. Motivated by these excellent efforts [31–33], it would be of interest to focus on investigating distributed online learning algorithms to obtain dynamic regret bounds for non-smooth or non-convex objective functions over time-varying networks in the future work.

ACKNOWLEDGMENTS

This work is supported in part by the Natural Science Foundation of China under Grant No. 61472003, National Key R&D Program of China under Grant No. 2018YFF0301000, Academic and Technical Leaders and Backup Candidates of Anhui Province under Grant No. 2019h211, Innovation team of ‘50 Star of Science and Technology’ of Huainan, Anhui Province.

REFERENCES

1. Yan, F., et al.: Distributed autonomous online learning regrets and intrinsic privacy-preserving properties. IEEE Trans. Knowl. Data Eng. 25, 2483–2493 (2013)
2. Huang, B.: Distributed quadratic optimisation for linear multi-agent systems over jointly connected networks. IET Control Theory Appl. 13, 2811–2816 (2019)
3. Li, K.: Distributed optimisation based on multi-agent system for resource allocation with communication time-delay. IET Control Theory Appl. 14, 549–557 (2020)
4. Koppel, A., et al.: Decentralized online learning with kernels. IEEE Trans. Signal Process. 66, 3240–3255 (2018)
5. Qu, G., Li, N.: Accelerated distributed Nesterov gradient descent. IEEE Trans. Autom. Control 65, 2566–2581 (2020)
6. Zhang, J., You, K.: Assyp: an exact asynchronous algorithm for convex optimization over digraphs. IEEE Trans. Autom. Control 65, 2494–2509 (2020)
7. Tran, N.T., et al.: Distributed optimisation of second-order multi-agent systems by control algorithm using position-only interaction with timevarying delay. IET Control Theory Appl. 11, 2549–2558 (2017)
8. Al-Sayed, S., et al.: Node-specific diffusion LMS-based distributed detection over adaptive networks. IEEE Trans. Signal Process. 66, 682–697 (2017)
9. Nedic, A., Ozdaglar, A.: Distributed subgradient methods for multi-agent optimization. IEEE Trans. Autom. Control 54, 48–61 (2009)
10. Jakovetić, D.: Fast distributed gradient methods. IEEE Trans. Autom. Control 59, 1131–1146 (2014)
11. Akbari, M.: Distributed online convex optimization on time-varying directed graphs. IEEE Trans. Control Netw. Syst. 4, 417–428 (2017)
12. Mateos-Náñez, D., Cortés, J.: Distributed online convex optimization over jointly connected digraphs. IEEE Trans. Netw. Sci. Eng. 1, 23–37 (2014)
13. Xu, H., Ling, Q.: Decentralized online alternating direction method of multipliers. J. Comput. Appl. 6, 1595–1599 (2005)
14. Hosseini, S.: Online distributed optimization via dual averaging. 52nd IEEE Conference on Decision and Control, pp. 484–1489. IEEE, Piscataway, NJ (2013)
15. Zhang, W., et al.: Projection-free distributed online learning in networks. Proceedings of the 34th International Conference on Machine Learning, Sidney, August, pp. 4054–4062. Curan Associates, Red Hook, NY (2017)
16. Frank, M., Wolfe, P.: An algorithm for quadratic programming. Nav. Res. Logist. Q. 3, 95–110 (1956)
17. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J. Imaging Sci. 2, 183–202 (2009)
18. Ram, S.S., Nedic, A., Veeravalli, V.V.: Distributed stochastic subgradient projection algorithms for convex optimization. J. Optim. Theory Appl. 147, 516–545 (2010)
19. Nesterov, Y., Nemirovskii, A.: Interior-point polynomial algorithms in convex programming. Soc. Ind. Appl. Math. 13, (1994). https://doi.org/10.1137/1.9781611970791
20. Hazan, E.: Logarithmic regret algorithms for online convex optimization. Mach. Learn. 69, 169–192 (2007)
21. Hazan, E., Kale, S.: Projection-free online learning. International Conference on Machine Learning, Edinburg, pp. 521–528. Association for Computational Linguistics, Stroudsburg, PA (2012)
22. Garber, D., Hazan, E.: A Linearly Convergent Variant of the Conditional Gradient Algorithm under Strong Convexity, with Applications to Online and Stochastic Optimization. SIAM Journal on Optimization. 26, 1493–1528 (2016)
23. Nedic, A., Ozdaglar, A., Parrilo, P.A.: Constrained consensus and optimization in multi-agent networks. IEEE Trans. Autom. Control 55, 922–938 (2010)
24. Duchi, J.C.: Dual averaging for distributed optimization: convergence analysis and network scaling. IEEE Trans. Autom. Control 57, 592–606 (2012)
25. Horn, R.A., Johnson, C.R.: Matrix Analysis. Cambridge University Press, Cambridge (1990)
26. Kasai, H.: SGDLibrary: A MATLAB library for stochastic optimization algorithms. J. Mach. Learn. Res. 18, 7942–7946 (2017)
27. Hiriart-Urruty, J.B., Lemaréchal, C.: Convex Analysis and Minimization Algorithms I'. Springer, New York, NY (1996).
28. Boyd, S., Diaconis, P., Xiao, L.: Fastest mixing Markov chain on a graph. SIAM Rev. 46, 667–689 (2004)
29. Watts, D.J., Strogatz, S.H.: Collective dynamics of ‘small-world’ networks. Nature 393, 440 (1998)
30. Colín, I., et al.: Gossip dual averaging for decentralized optimization of pairwise functions. In Proceedings of the 33rd International Conference on International Conference on Machine Learning (ICML 16). JMLR.org, 48, 1388–1396 (2016)
31. Wang, Y., et al.: Distributed subgradient-free stochastic optimization algorithm for nonsmooth convex functions over time-varying networks. SIAM J. Control Optim. 57, 2821–2842 (2019)
32. Liang, S.: Distributed quasi-monotone subgradient algorithm for nonsmooth convex optimization over directed graphs. Automatica 101, 175–181 (2019)

33. Wang, Q., et al.: Distributed proximal-gradient algorithms for nonsmooth convex optimization of second-order multiagent systems. Int. J. Robust Nonlinear Control 30, 7574–7592 (2020)

How to cite this article: Shen X, Li D, Fang R, Dong Q. A conditional gradient algorithm for distributed online optimization in networks. IET Control Theory Appl. 2021;15:570–579.
https://doi.org/10.1049/cth2.12062