One Gradient Frank-Wolfe for Decentralized Online Convex and Submodular Optimization

Tuan-Anh Nguyen TUAN-ANH.NGUYEN@INRIA.FR
LIG, INRIA, University Grenoble-Alpes
38400 Saint-Martin-d'Hères, France

Nguyen Kim Thang KIM-THANG.NGUYEN@UNIV-GRENOBLE-ALPES.FR
LIG, INRIA, CNRS, Grenoble INP, University Grenoble-Alpes
38400 Saint-Martin-d'Hères, France

Denis Trystram DENIS.TRYSTRAM@IMAG.FR
LIG, INRIA, CNRS, Grenoble INP, University Grenoble-Alpes
38400 Saint-Martin-d'Hères, France

Editors: Emtiyaz Khan and Mehmet Gönen

Abstract

Decentralized learning has been studied intensively in recent years motivated by its wide applications in the context of federated learning. The majority of previous research focuses on the offline setting in which the objective function is static. However, the offline setting becomes unrealistic in numerous machine learning applications that witness the change of massive data. In this paper, we propose *decentralized online* algorithm for convex and continuous DR-submodular optimization, two classes of functions that are present in a variety of machine learning problems. Our algorithms achieve performance guarantees comparable to those in the centralized offline setting. Moreover, on average, each participant performs only a *single* gradient computation per time step. Subsequently, we extend our algorithms to the bandit setting. Finally, we illustrate the competitive performance of our algorithms in real-world experiments.

Keywords: Decentralized Learning, Online Optimization, Frank-Wolfe Algorithm, Submodular Function

1. Introduction

Learning over data generated by sensors and mobile devices has gained a high interest in recent years due to the continual interaction with users and the environment on a timely basis. The patterns related to user's behavior, preference, and the surrounding stochastic events become a promising source for machine learning applications to be more and more reliable. However, collecting such data in a centralized location has become problematic due to privacy concerns and the high cost of data transfer over the network. Consequently, the learning methods that can leave the data locally while efficiently exploiting data patterns, such as decentralized learning, are emerging as an alternative to traditional centralized learning.

Under the optimization scheme, learning in a decentralized manner consists of multiple interconnected agents cooperating to optimize a global objective function where each agent

detains partial information of the interested function. Several works Deori et al. (2016); Reisizadeh et al. (2019); Yuan et al. (2016); Duchi et al. (2012); Zheng et al. (2018) have considered this setting for convex and strongly convex functions. Wai et al. (2017) also study the problem when the objective function is generally non-convex whereas Mokhtari et al. (2018a); Xie et al. (2019) proposes a decentralized algorithm to maximize monotone submodular functions for both continuous and discrete domains. However, these works only consider the offline setting which is not realistic since data constantly evolve in many real-world applications. In this paper, we study decentralized online algorithms for optimizing both convex and submodular functions.

Problem definition. Formally, we are given a convex set $\mathcal{K} \subseteq \mathbb{R}^d$ (w.l.o.g one can assume that $\mathcal{K} \subseteq [0,1]^d$) and a set of agents connected over a network represented by a graph $G = (V, E)$ where $n = |V|$ is the number of agents. At every time $1 \leq t \leq T$, each agent $i \in V$ can communicate with (and only with) its immediate neighbors, i.e., adjacent agents in G and takes a decision $\boldsymbol{x}_t^i \in \mathcal{K}$. Subsequently, a cost/reward function $f_t^i : \mathcal{K} \to \mathbb{R}$ is revealed adversarially and locally to agent i. Note that in the *bandit* setting, agent i observes only the value $f_t^i(\boldsymbol{x}_t^i)$ instead of the whole function f_t^i. Although each agent i observes only function f_t^i (or the value $f_t^i(\boldsymbol{x}_t^i)$ in the bandit setting), agent i is interested in the cumulating cost/reward $F_t(\cdot)$ where $F_t(\cdot) := \frac{1}{n}\sum_{j=1}^n f_t^j(\cdot)$. In particular, at time t, the cost/reward of agent i with the its chosen $\boldsymbol{x}_t^i$ is $F_t(\boldsymbol{x}_t^i)$.

In the context of convex minimization, the functions f_t^i's are convex and the goal of each agent i is to minimize the total cumulating cost $\sum_{t=1}^T F_t(\boldsymbol{x}_t^i)$ via local communication with its immediate neighbors. Our objective is to design an algorithm with small regret. An online algorithm is $\mathcal{R}_T$-*regret* if for every agent $1 \leq i \leq n$,

$$\sum_{t=1}^{T} F_t(\boldsymbol{x}_t^i) - \min_{\boldsymbol{x}\in\mathcal{K}} \sum_{t=1}^{T} F_t(\boldsymbol{x}) \leq \mathcal{R}_T$$

In the context of monotone DR-submodular maximization, the functions f_t^i's are monotone DR-submodular. Roughly speaking, a bounded differentiable and non-negative function $F : [0,1]^d \to \mathbb{R}_+$ is *DR-submodular* if for every $\boldsymbol{x}, \boldsymbol{y} \in [0,1]^d$ satisfying $x_i \leq y_i, \forall i \in [d]$, we have $\nabla F(\boldsymbol{x}) \geq \nabla F(\boldsymbol{y})$. The goal of each agent i is to maximize the total cumulating reward $\sum_{t=1}^T F_t(\boldsymbol{x}_t^i)$, again via local communication with its immediate neighbors. Our objective is to design an algorithm with an approximation ratio as close to 1 as possible and together with a small regret. An online algorithm has a ρ-*regret* of $\mathcal{R}_T$ if for every agent $1 \leq i \leq n$,

$$\rho \cdot \max_{\boldsymbol{x}\in\mathcal{K}} \sum_{t=1}^{T} F_t(\boldsymbol{x}) - \sum_{t=1}^{T} F_t(\boldsymbol{x}_t^i) \leq \mathcal{R}_T$$

1.1. Our contribution

The challenge in designing robust and efficient algorithms for these problems is to simultaneously address the following issues:

- Uncertainty (online setting, agents observe their loss functions only after selecting their decisions).

- Partial information (decentralized setting, agents know only its local loss functions while attempting to minimize the cumulated cost).
- Low computation/communication resources of agents (so it is desirable that each agent performs a small number of gradient computations and communications).
- Additionally, in the bandit setting, one has only limited feedback (agents can only observe the function value of their decisions).

We present performance-guaranteed algorithms for solving the constraint convex and continuous DR-submodular optimization problem in the decentralized and online setting with *only* one gradient evaluation and low communications per agent per time step on average. Specifically, our algorithms achieve the regret and the $\left(1-\frac{1}{e}\right)$-regret bounds of $O(T^{4/5})$ for both convex and monotone continuous DR-submodular functions. Using a one-point gradient estimator Flaxman et al. (2005), we extend the algorithms to the bandit setting in which the gradient is unavailable to the agents. We obtain the $\left(1-\frac{1}{e}\right)$-regret bound of $O(T^{8/9})$ for the bandit setting. It should be noted that the $\left(1-\frac{1}{e}\right)$-regret of $O(T^{4/5})$ and $O(T^{8/9})$ matches the regret guarantees in the centralized online settings. Besides, one can convert the algorithm to be projection-free (by selecting suitable oracles). This property allows the algorithm to be implemented in various contexts based on the computing capacity of local devices. We demonstrate the practical application of our algorithm on a Movie Recommendation problem and present a thorough analysis of different aspects of the performance guarantee, the effects of network topology, and decentralization, which are predictably explained by our theoretical results.

Algorithm	Stochastic Gradient	$(1-1/e)$-Regret	Communications	Gradient Evaluations
DMFW	Yes	$\mathcal{O}(T^{1/2})$	$2 \cdot T^{3/2}$	$T^{3/2}$
Monode-FW	Yes	$\mathcal{O}(T^{4/5})$	2· #neighbors	1
Bandit Monode-FW	-	$\mathcal{O}(T^{8/9})$	2· #neighbors	-

Table 1: Comparison of previous work on *adversarial* decentralized online monotone DR-submodular maximization (DMFW Zhu et al. (2021)) and our proposed algorithms (in bold). The communications and gradient evaluations are mesured per agent per time step.

1.2. Related Works

Decentralized Online Optimization. Yan et al. (2013) introduced decentralized online projected subgradient descent and showed vanishing regret for convex and strongly convex functions. In contrast, Hosseini et al. (2013) extended distributed dual averaging technique to the online setting, using a general regularized projection for both unconstrained and constrained optimization. A distributed variant of online conditional gradient Hazan (2016) was designed and analyzed in Zhang et al. (2017) that requires linear minimizers and uses exact gradients. However, computing exact gradients may be prohibitively expensive for

moderately sized data and intractable when a closed-form does not exist. Zhu et al. (2021) proposes a decentralized online algorithm for maximizing monotone submodular function on a time-varying network using stochastic gradient estimate and multiple optimization oracles. This work achieves the optimal regret bound of $O(T^{1/2})$ but requires $T^{3/2}$-gradient evaluation and communication per function. In this work, we advance further in designing a distributed algorithm that uses stochastic gradient estimates and requires only one gradient evaluation.

Monotone DR-submodular Maximization. The maximization of monotone DR-submodular functions has been investigated in both offline and online settings. For the offline case, Bian et al. (2017) examined the problem where the constraint set is a down-closed convex set and demonstrated that the greedy method Calinescu et al. (2011), a variation of the Frank-Wolfe algorithm, ensures a $(1-1/e)$-approximation. Hassani et al. (2017) demonstrated the restriction of the greedy method in a stochastic environment where only unbiased gradient estimates are available. Later, Mokhtari et al. (2018b) introduced an algorithm for maximizing monotone DR-submodular function over the general convex set using new variance reduction techniques to accomplish $(1-1/e)$-approximation in a stochastic setting. Chen et al. (2018) suggested a method that achieves $(1-1/e, O(\sqrt{T}))$-regret for maximizing monotone Dr-submodular over a general convex set in an online setting. Subsequently, Zhang et al. (2019) introduced an approach that reduces the number of per-function gradient evaluations from $T^{3/2}$ to 1, while maintaining the same approximation ratio of $(1-1/e)$. They also presented a bandit approach that achieves an expected $(1-1/e)$-approximation ratio with regret $T^{8/9}$ to tackle the same problem.

2. Preliminaries and Notations

We begin by explaining the notations and concepts that will be used throughout the paper. Given an undirected graph $G = (V, E)$, the set of neighbors of an agent $i \in V$ is denoted $N(i) := \{j \in V : (i,j) \in E\}$. Consider the following symmetric matrix $\mathbf{W} \in \mathbb{R}_+^{n\times n}$ with entry W_{ij} defined as follows.

$$W_{ij} = \begin{cases} \dfrac{1}{1+\max\{d_i, d_j\}} & \text{if } (i,j) \in E \\ 0 & \text{if } (i,j) \notin E, i \neq j \\ 1 - \sum_{j\in N(i)} W_{ij} & \text{if } i = j \end{cases}$$

where $d_i = |N(i)|$, the degree of vertex i. In fact, the matrix $\mathbf{W}$ is doubly stochastic, i.e $\mathbf{W1} = \mathbf{W}^T\mathbf{1} = \mathbf{1}$ and so it inherits several useful properties of doubly stochastic matrices. We use boldface letter e.g $\boldsymbol{x}$ to represent vectors. We denote by $\boldsymbol{x}_{q,k}^i$ the decision vector of agent i at time step k of phase q. If not specified otherwise, we suppose that the constraint set $\mathcal{K}$ is a bounded convex set with diameters $D = \sup_{\boldsymbol{x},\boldsymbol{y}\in\mathcal{K}} \|\boldsymbol{x}-\boldsymbol{y}\|$ and radius $R = \sup_{\boldsymbol{x}\in\mathcal{K}} \|\boldsymbol{x}\|$. For two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$, we note $\boldsymbol{x} \leq \boldsymbol{y}$ if $x_i \leq y_i \ \forall i$. We note $\mathbb{B}_d$ and $\mathbb{S}_d$ the d-dimensional unit ball and the unit sphere, respectively.

A continuous function $F : [0,1]^d \to \mathbb{R}_+$ is *DR-submodular* if for any vectors $\boldsymbol{x}, \boldsymbol{y} \in [0,1]^d$ such that $\boldsymbol{x} \leq \boldsymbol{y}$, for a constant $\alpha > 0$ and any basis vectors $\boldsymbol{e}_i = (0, \ldots, 1, \ldots, 0)$ such that

$\boldsymbol{x} + \alpha \boldsymbol{e}_i \in [0,1]^d$ and $\boldsymbol{y} + \alpha \boldsymbol{e}_i \in [0,1]^d$.

$$F(\boldsymbol{x} + \alpha \boldsymbol{e}_i) - F(\boldsymbol{x}) \geq F(\boldsymbol{y} + \alpha \boldsymbol{e}_i) - F(\boldsymbol{y}) \tag{1}$$

For a differentiable function, the DR-property is equivalent to $\nabla F(\boldsymbol{x}) \geq \nabla F(\boldsymbol{y})$, $\forall \boldsymbol{x} \leq \boldsymbol{y} \in [0,1]^d$. More over, if F is twice-differentiable, the DR-property is equivalent to all entries of the Hessian matrix being non-positive i.e $\forall 1 \leq i, j \leq d$, $\frac{\partial^2 F}{\partial \boldsymbol{x}_i \partial \boldsymbol{x}_j} \leq 0$. A function F is *monotone* if $\forall \boldsymbol{x} \leq \boldsymbol{y} \in [0,1]^d$, we have $F(\boldsymbol{x}) \leq F(\boldsymbol{y})$.

A function F is β-*smooth* if for all $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{K}$:

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

or equivalently $\|\nabla F(\boldsymbol{x}) - \nabla F(\boldsymbol{y})\| \leq \beta \|\boldsymbol{x} - \boldsymbol{y}\|$. Also, we say a function F is G-*Lipschitz* if for all $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{K}$

$$\|F(\boldsymbol{x}) - F(\boldsymbol{y})\| \leq G \|\boldsymbol{x} - \boldsymbol{y}\|$$

In this paper, we employ linear optimization oracles in our algorithm to solve an online linear optimization problem given a feedback function and a constraint set. In particular, in the online linear optimization problem, one must choose $\boldsymbol{u}^t \in \mathcal{K}$ at every time $1 \leq t \leq T$. The adversary then discloses a vector $\boldsymbol{d}^t$ and feedbacks the cost function $\langle \cdot, \boldsymbol{d}^t \rangle$ where the goal is to minimize the regret of the linear objective. Several algorithms Hazan (2016), including the projection-free follow-the-perturbed-leader algorithm offer an optimum regret bound of $\mathcal{R}_T = O(\sqrt{T})$ for the online linear optimization problem. One of these methods can be used as an oracle to solve the online linear optimization problem.

In practice, it may not be possible to use a full gradient due to the vast quantity of data and processing restrictions. To address this issue, our approach utilizes an unbiased stochastic gradient in place of the gradient and proposes a variance reduction technique for distributed optimization based on a rigorous analysis that may be applied to problems of independent interest. We make the following assumptions for the next two sections.

Assumption 1 *We let k_0 to be the smallest integer such that $\lambda_2(\mathbf{W}) \leq \left(\frac{k_0}{k_0+1}\right)^2$.*

Assumption 2 *The function f_t is G-Lipschitz and β-smooth. Its stochastic gradient $\widetilde{\nabla} f_t(\boldsymbol{x})$ is unbiased, uniformly upper-bounded and has a bounded variance, i.e., $\mathbb{E}\left[\widetilde{\nabla} f_t(\boldsymbol{x})\right] = \nabla f_t(\boldsymbol{x})$, $\left\|\widetilde{\nabla} f_t(\boldsymbol{x})\right\| \leq G_0$, and $\mathbb{E}\left[\left\|\widetilde{\nabla} f_t(\boldsymbol{x}) - \nabla f_t(\boldsymbol{x})\right\|^2\right] \leq \sigma_0^2$.*

Assumption 3 *For all $t \in [T]$ and $i \in [n]$, $\sup_{\boldsymbol{x} \in \mathcal{K}} |f_t^i(\boldsymbol{x})| \leq B$*

Assumption 4 *There exist a number $r \geq 0$ such that $r\mathbb{B}_d \subseteq \mathcal{K}$*

3. Full Information Setting

This section thoroughly describes the algorithm for both convex and DR-submodular optimization. Recall that each agent receives a function f_t^i at every time $t \in [T]$. We partition time steps into Q blocks, each of size K so that $T = QK$. For each block $q \in [Q]$, we define f_q^i as the average of the K functions within the block. Additionally, each agent $1 \leq i \leq n$ maintains K online linear optimization oracles $\mathcal{O}_{i,1}, \ldots, \mathcal{O}_{i,K}$. Let $\sigma_q \in \mathfrak{S}_K$ be a random permutation of function indexes for all agents.

At a high level, at each block q, the agent i performs K-steps of Frank-Wolfe algorithm, where the update vector is a combination of the oracles' outputs and the aggregate of its neighbors' current decisions. The final decision $\boldsymbol{x}_q^i$ for the block q is disclosed at the end of K steps, such that at each time step in the block, agent i plays the same decision $\boldsymbol{x}_q^i$.

More specifically, following the Frank-Wolfe steps, agent i performs K gradient updates using the estimators $f^i_{\sigma_q(k)}$. It calculates the stochastic gradient of the permuted function $f^i_{\sigma_q(k)}$ evaluated at the corresponding decision vector $\boldsymbol{x}^i_{q,k}$ and thereafter exchanges information with its neighbors. It then computes a variance reduction version $\widetilde{\boldsymbol{a}}^i_{q,k}$ of the vector $\widetilde{\boldsymbol{d}}^i_{q,k}$ and returns $\langle \widetilde{\boldsymbol{a}}^i_{q,k}, \cdot \rangle$ as the cost function at time $\sigma_q^{-1}(k)$ to the oracle $\mathcal{O}^i_k$. The vectors $\widetilde{\boldsymbol{d}}^i_{q,k}$ are subtly constructed to capture progressively more information on the accumulating cost functions.

Note that the use of random permutation σ_q is crucial here. By that, all the permuted functions $f^i_{\sigma_q(k)}$ become an estimation of f^i_q, i.e., $\mathbb{E}[f^i_{\sigma_q(k)}] = f^i_q$. Therefore the gradient of $f^i_{\sigma_q(k)}$ is likewise an estimation of the gradient of f^i_q. One can think of f^i_q as an artificial objective function for which we have access to its gradient estimates, where each estimation is one gradient evaluation per function within the block. As a result, conducting K gradient updates of f^i_q turns out to be executing one gradient update for each of the K functions. Using this approach, initiated in Zhang et al. (2019), we can effectively reduce the gradient evaluation number to 1 for each arriving function f^i_t.

Since we deal with both convex and submodular, there are modifications to adapt for both kinds of optimization problem. The online optimization oracle's objective function should be minimized for convex optimization and maximized for submodular optimization. The decision update for convex problems is a convex combination of the aggregated neighbors' decisions $\boldsymbol{y}^i_{q,k}$ and the oracle's output $\boldsymbol{v}^i_{q,k}$, i.e.,

$$\boldsymbol{x}^i_{q,k+1} = (1 - \eta_k)\boldsymbol{y}^i_{q,k} + \eta_k \boldsymbol{v}^i_{q,k}, \quad \eta_k \in [0, 1] \tag{2}$$

whereas the update for the submodular optimization problem is achieved by shifting the aggregated decisions towards the direction of the oracle's output by a step-size η_k, i.e.,

$$\boldsymbol{x}^i_{q,k+1} = \boldsymbol{y}^i_{q,k} + \eta_k \boldsymbol{v}^i_{q,k}, \quad \eta_k \in [0, 1] \tag{3}$$

For convex functions, the initialization can be any random point inside the constraint set, however for submodular functions, this value should be set to 0.

The formal description is given in Algorithm 1. The proof of the following lemmas and theorems can be found in **??**.

Algorithm 1 Monode Frank-Wolfe

Input: A convex set $\mathcal{K}$, a time horizon T, a block size K, online linear optimization oracles $\mathcal{O}_{i,1}, \ldots, \mathcal{O}_{i,K}$ for each agent $1 \leq i \leq n$, step sizes $\eta_k \in (0,1)$ for all $1 \leq k \leq K$, number of blocks $Q = T/K$

Initialize linear optimizing oracle $\mathcal{O}_k^i$ for all $1 \leq k \leq K$
for $q = 1$ to Q **do**
  for every agent $1 \leq i \leq n$ **do**
    Initialize $\boldsymbol{x}_{q,1}^i$ and set $\widetilde{\boldsymbol{a}}_{i,0}^t \leftarrow 0$
    for $1 \leq k \leq K$ **do**
      Let $\boldsymbol{v}_{q,k}^i$ be the output of oracle $\mathcal{O}_k^i$ at phase q.
      Send $\boldsymbol{x}_{q,k}^i$ to all neighbours $N(i)$
      Once receiving $\boldsymbol{x}_{q,k}^j$ from all neighbours $j \in N(i)$, set $\boldsymbol{y}_{q,k}^i \leftarrow \sum_j W_{ij} \boldsymbol{x}_{q,k}^j$.
      Update $\boldsymbol{x}_{q,k+1}^i$ as (2) or (3).
    end for
    Choose $\boldsymbol{x}_q^i \leftarrow \boldsymbol{x}_{q,K+1}^i$ and agent i plays the same $\boldsymbol{x}_q^i$ for every time t in phase q.
    Let σ_q be a random permutation of $1, \ldots, K$ — times in phase q.
    for $1 \leq k \leq K$ **do**
      Let $s = \sigma_q^{-1}(k)$
      Query the values of $\widetilde{\nabla} f_k^i(\boldsymbol{x}_{q,s}^i)$
    end for
    Set $\widetilde{\boldsymbol{g}}_{q,1}^i \leftarrow \widetilde{\nabla} f_{\sigma_q(1)}^i(\boldsymbol{x}_{q,1}^i)$
    for $1 \leq k \leq K$ **do**
      Send $\widetilde{\boldsymbol{g}}_{q,k}^i$ to all neighbours $N(i)$.
      After receiving $\widetilde{\boldsymbol{g}}_{q,k}^j$ from all neighbours $j \in N(i)$, compute $\widetilde{\boldsymbol{d}}_{q,k}^i \leftarrow \sum_{j \in N(i)} W_{ij} \widetilde{\boldsymbol{g}}_{q,k}^j$ and $\widetilde{\boldsymbol{g}}_{q,k+1}^i \leftarrow \left(\widetilde{\nabla} f_{\sigma_q(k+1)}^i(\boldsymbol{x}_{q,k+1}^i) - \widetilde{\nabla} f_{\sigma_q(k)}^i(\boldsymbol{x}_{q,k}^i)\right) + \widetilde{\boldsymbol{d}}_{q,k}^i$
      $\widetilde{\boldsymbol{a}}_{q,k}^i \leftarrow (1-\rho_k) \cdot \widetilde{\boldsymbol{a}}_{q,k-1}^i + \rho_k \cdot \widetilde{\boldsymbol{d}}_{q,k}^i$.
      Feedback function $\langle \widetilde{\boldsymbol{a}}_{q,k}^i, \cdot \rangle$ to oracles $\mathcal{O}_k^i$. (The cost of the oracle $\mathcal{O}_k^i$ at block q is $\langle \widetilde{\boldsymbol{a}}_{q,k}^i, \boldsymbol{v}_{q,k}^i \rangle$.)
    end for
  end for
end for

Lemma 1 *Let* $V_d = 2nG\left(\frac{\lambda_2}{1-\lambda_2} + 1\right)$, *the local gradient is uniformly upper-bounded, i.e,* $\forall i \in [n], \forall k \in [K]$. $\left\|\boldsymbol{d}_{q,k}^i\right\| \leq V_d$.

Lemma 2 *Under Assumption 2 and let* $\sigma_1^2 = 4n\left[\left(\frac{G+G_0}{\frac{1}{\lambda_2}-1}\right)^2 + 2\sigma_0^2\right]$. *For* $i \in [n], k \in [K]$, *the variance of the local stochastic gradient is uniformly bounded i.e* $\mathbb{E}\left[\left\|\boldsymbol{d}_{q,k}^i - \widetilde{\boldsymbol{d}}_{q,k}^i\right\|^2\right] \leq \sigma_1^2$

Theorem 3 *Given a convex set $\mathcal{K}$ and assume that F_t is a convex function. Setting $Q = T^{2/5}, K = T^{3/5}, T = QK$ and step-size $\eta_k = \frac{1}{k}$. Let $\rho_k = \frac{2}{(k+3)^{2/3}}$ and $\rho_k = \frac{1.5}{(K-k+2)^{2/3}}$ when $k \in \left[1, \frac{K}{2}\right]$ and $k \in \left[\frac{K}{2}+1, K\right]$ respectively. Then, the expected regret of Algorithm 1 is at most*

$$\mathbb{E}\left[\mathcal{R}_T\right] \leq \left(GD + 2\beta D^2\right) T^{2/5} + \left(C + 6D\left(N + \sqrt{M}\right)\right) T^{4/5} + \frac{3}{5}\beta D^2 T^{2/5} \log(T) \tag{4}$$

where $N = k_0 \cdot nG \max\{\lambda_2\left(1 + \frac{2}{1-\lambda_2}\right), 2\}$ and $M = \max\{M_1, M_2\}$ where $M_0 = 4\left(V_{\boldsymbol{d}}^2 + \sigma_1^2\right) + 128V_{\boldsymbol{d}}^2$, $M_1 = \max\left\{5^{2/3}\left(V_{\boldsymbol{d}} + \frac{2}{4^{2/3}}G_0\right)^2, M_0\right\}$ and $M_2 = 2.55\left(V_{\boldsymbol{d}}^2 + \sigma_1^2\right) + \frac{28V_{\boldsymbol{d}}^2}{3}$

Theorem 4 *Given a convex set $\mathcal{K}$ and assume that the function F_t is monotone continuous DR-submodular. Setting $Q = T^{2/5}, K = T^{3/5}, T = QK$ and step-size $\eta_k = \frac{1}{K}$. Let $\rho_k = \frac{2}{(k+3)^{2/3}}$ and $\rho_k = \frac{1.5}{(K-k+2)^{2/3}}$ when $1 \leq k \leq \frac{K}{2}+1$ and $\frac{K}{2}+1 \leq k \leq K$ respectively. Then, the expected $\left(1 - \frac{1}{e}\right)$-regret is at most*

$$\mathbb{E}\left[\mathcal{R}_T\right] \leq \frac{3}{2}\beta D^2 T^{2/5} + \left(C + 3D(N + \sqrt{M}\right) T^{4/5} \tag{5}$$

where the constants are defined in Theorem 3

As stated in the preceding paragraph, the distinction between convex and submodular optimization can be found in line 9 of Algorithm 1 and in the oracle optimization subroutine. To achieve the regret bound mentioned in Theorems 3 and 4, we use follow-the-perturbed-leader as the oracle with regret $\mathcal{R}_T = C\sqrt{T}$. In the case of convex optimization, one may use online gradient descent to obtain the same outcome, but this method is more computationally intensive because it involves a projection step onto the constraint set.

4. Bandit Setting

This section describes a bandit algorithm for a decentralized submodular maximization. We let $\mathcal{K}$ be a down-closed convex set. A major difference between this algorithm and the previous one is the function's value $f_t^i(\boldsymbol{x}_t^i)$ being the only information provided to the agent. It does not know of the value incurred if it had chosen another decision in the constraint set. As a consequence, this setting makes access to the gradient impossible for the agent. To circumvent this limitation, we use the one-point gradient estimate Flaxman et al. (2005) and adapt the biphasic bandit setting Zhang et al. (2019) to our decentralized algorithm.

We recall that for a function f_t defined on $\mathcal{K} \subset \mathbb{R}^d$, it admits a δ-smoothed version for any $\delta > 0$, given as

$$\hat{f}_{t,\delta}(\boldsymbol{x}_t) = \mathbb{E}_{\boldsymbol{v} \sim \mathbb{B}_d}\left[f_t(\boldsymbol{x}_t + \delta \boldsymbol{v})\right]$$

where $\boldsymbol{v}$ is drawn uniformly random from the d-dimensional unit ball. The value of $\hat{f}_{t,\delta}$ at a point $\boldsymbol{x}$ is the average of f_t evaluated across the d-dimensional ball of radius δ centered at $\boldsymbol{x}$. This function inherits various functional properties from f_t, therefore becomes a suitable approximation for f_t, as shown in the following lemma.

Lemma 5 (Lemma 2 Chen et al. (2020), Lemma 6.6 Hazan (2016)) *Let f be a monotone continuous DR-submodular function. If f is β-smooth, G-Lipschitz, then so is $\hat{f}_\delta$ and we have $\left\|\hat{f}_\delta(\boldsymbol{x}) - f(\boldsymbol{x})\right\| \leq \delta G$. More over, if we choose $\boldsymbol{u}$ uniformly from the unit sphere $\mathbb{S}^{d-1}$, the following equation holds*

$$\nabla \hat{f}_{t,\delta}(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{u}\sim\mathbb{S}_{d-1}}\left[\frac{d}{\delta} f_t(\boldsymbol{x}+\delta\boldsymbol{u})\boldsymbol{u}\right] \tag{6}$$

Lemma 5 shows that a decision that maximizes $\hat{f}_{t,\delta}$ can also maximizes f_t approximately. The δ-smooth version additionally provides a one-point gradient estimate that can be used to estimate the gradient of f_t by evaluating the function at a random point on the $(d-1)$-dimensional sphere of radius δ. It is important to note that the point $\boldsymbol{x}+\delta\boldsymbol{u}$ may be outside the set when $\boldsymbol{x}$ is near to the constraint set's boundary. For this reason, we let $\mathcal{K}' \subset \mathcal{K}$ be the δ-interior of $\mathcal{K}$ that verifies : $\forall \boldsymbol{x} \in \mathcal{K}'$, $\mathbb{B}(\boldsymbol{x}, \delta) \subset \mathcal{K}$, and solve the optimization problem on the new set $\mathcal{K}'$. By shrinking the constraint set down to $\mathcal{K}'$, we assure that the point $\boldsymbol{x}+\delta\boldsymbol{u}$ is in $\mathcal{K}$ for any point $\boldsymbol{x}$ in $\mathcal{K}$'. More over, if the distance $d(\mathcal{K}', \mathcal{K})$ between $\mathcal{K}'$ and $\mathcal{K}$ is small enough, we can approximately get the optimal regret bound on the original constraint set $\mathcal{K}$ by running the bandit algorithm on $\mathcal{K}'$. The detail on the construction of $\mathcal{K}'$ is given is Lemma 6

The biphasic setting consist of partitioning T into Q blocks of size L, with each block consisting of two phases: exploration and exploitation. Each agent i performs $K < L$ steps of exploration by updating the decision vector $\boldsymbol{x}^i_{q,k}$ using equation (3). During the exploration phase, rather than playing the final decision as in Algorithm 1, the agent draws uniformly a random vector $\boldsymbol{u}^i_{q,k}$ from $\mathbb{S}_{d-1}$ and plays $\boldsymbol{x}^i_{q,k} + \delta\boldsymbol{u}^i_{q,k}$ for the function $f^i_{\sigma_q(k)}$, as it can only estimate the gradient at the point it plays. The gradient estimate $\widetilde{\boldsymbol{h}}^i_{q,k}$ is then computed using equation (6), followed by a local aggregation and variance reductions steps, the final step consisting of feeding the variance reduction vector $\widetilde{\boldsymbol{a}}^i_{q,k}$ back to the oracle $\mathcal{O}^i_k$. The remaining $L-K$ iterations are used for exploitation, where each agent plays the final decision x^i_q to obtain a high reward. We give the detail of every step in Algorithm 2. **??** contains the analysis of Theorem 7

Lemma 6 (Lemma 1, Zhang et al. (2019)) *Let $\mathcal{K}$ is down-closed convex set and δ is is sufficiently small such that $\alpha = \frac{\sqrt{d}+1}{r}\delta < 1$. The set $\mathcal{K}' = (1-\alpha)\mathcal{K} + \delta\mathbf{1}$ is convex, compact and down-closed δ-interior of $\mathcal{K}$ satisfies $d(\mathcal{K}, \mathcal{K}') \leq \left(\sqrt{d}\left(\frac{R}{e}+1\right)+\frac{R}{r}\right)\delta$*

Theorem 7 *Let $\mathcal{K}$ be a down-closed convex and compact set. We suppose the δ-interior $\mathcal{K}'$ verify Lemma 6. Let $Q = T^{2/9}, L = T^{7/9}, K = T^{2/3}$, $\delta = \frac{r}{\sqrt{d}+2}T^{-1/9}$ and $\rho_k = \frac{2}{(k+2)^{2/3}}$, $\eta_k = \frac{1}{K}$. Then the expected $\left(1-\frac{1}{e}\right)$-regret is upper bounded*

$$\mathbb{E}\left[\mathcal{R}_T\right] \leq ZT^{8/9} + \frac{\beta D^2}{2}T^{1/9} + \frac{3}{2}D\frac{d\left(\sqrt{d}+2\right)}{r}P_{n,\lambda_2}T^{2/9} + \beta D^2 T^{3/9} \tag{7}$$

where we note $Z = \left(1-\frac{1}{e}\right)\left(\sqrt{d}\left(\frac{R}{e}+1\right)+\frac{R}{r}\right)G\frac{r}{\sqrt{d}+2} + \left(2-\frac{1}{e}\right)G\frac{r}{\sqrt{d}+2} + 2\beta + C$ and $P_{n,\lambda_2} = k_0 \cdot nB\max\left\{\lambda_2\left(1+\frac{2}{1-\lambda_2}\right), 2\right\} + 4^{1/3}\left(24n^2\left(\frac{1}{\frac{1}{\lambda_2}-1}+1\right)^2 + 8n\left(\frac{1}{\left(\frac{1}{\lambda_2}-1\right)^2}+2\right)\right)^{1/2}$

Algorithm 2 Bandit Monode Frank-Wolfe

Input: Smoothing radius δ, δ-interior $\mathcal{K}'$ with lower bound $\underline{u}$, a time horizon T, a block size L, number of exploration step K. Online linear optimization oracles $\mathcal{O}_{i,1}, \ldots, \mathcal{O}_{i,K}$ for each player $1 \leq i \leq n$, step sizes $\eta_k, \rho_k \in (0,1)$ for all $1 \leq k \leq K$, number of blocks $Q = T/L$

1: Initialize linear optimizing oracle $\mathcal{O}_k^i$ for all $1 \leq k \leq K$
2: **for** $q = 1$ to Q **do**
3: **for** every agent $1 \leq i \leq n$ **do**
4: Initialize $\boldsymbol{x}_{q,1}^i \leftarrow \underline{u}$ and set $\widetilde{\boldsymbol{a}}_{i,0}^t \leftarrow 0$
5: Update $\boldsymbol{x}_{q,k}^i$ using line 5 to 10 of Algorithm 1. Choose $\boldsymbol{x}_q^i \leftarrow \boldsymbol{x}_{q,K+1}^i$
6: Let σ_q be a random permutation of $1, \ldots, L$ — times in phase q.
7: **for** $1 \leq \ell \leq L$ **do**
8: Let $s = \sigma_q^{-1}(\ell)$
9: **if** $\ell \leq K$ **then**
10: play $f_{q,\ell}^i\left(\boldsymbol{x}_{q,s}^i + \delta \boldsymbol{u}_{q,s}^i\right)$ where $\boldsymbol{u}_{q,s}^i \in \mathbb{S}^{d-1}$. - *Exploration*
11: **else**
12: play $f_{q,\ell}^i\left(\boldsymbol{x}_q^i\right)$. - *Exploitation*
13: **end if**
14: **end for**
15: Set $\widetilde{\boldsymbol{g}}_{q,1}^i \leftarrow \frac{d}{\delta} f_{\sigma_q(1)}^i\left(\boldsymbol{x}_{q,1}^i + \delta \boldsymbol{u}_{q,1}^i\right) \boldsymbol{u}_{q,1}^i$
16: **for** $1 \leq k \leq K$ **do**
17: Let $\widetilde{\boldsymbol{h}}_{q,k}^i = \frac{d}{\delta} f_{\sigma_q(k)}^i\left(\boldsymbol{x}_{q,k}^i + \delta \boldsymbol{u}_{q,k}^i\right) \boldsymbol{u}_{q,k}^i$
18: Send $\widetilde{\boldsymbol{g}}_{q,k}^i$ to all neighbours $N(i)$.
19: After receiving $\widetilde{\boldsymbol{g}}_{q,k}^j$ from all neighbours $j \in N(i)$, compute $\widetilde{\boldsymbol{d}}_{q,k}^i \leftarrow \sum_{j \in N(i)} W_{ij} \widetilde{\boldsymbol{g}}_{q,k}^j$
20: $\widetilde{\boldsymbol{g}}_{q,k+1}^i \leftarrow \widetilde{\boldsymbol{h}}_{q,k+1}^i - \widetilde{\boldsymbol{h}}_{q,k}^i + \widetilde{\boldsymbol{d}}_{q,k}^i$
21: $\widetilde{\boldsymbol{a}}_{q,k}^i \leftarrow (1-\rho_k) \cdot \widetilde{\boldsymbol{a}}_{q,k-1}^i + \rho_k \cdot \widetilde{\boldsymbol{d}}_{q,k}^i$.
22: Feedback function $\langle \widetilde{\boldsymbol{a}}_{q,k}^i, \cdot \rangle$ to oracles $\mathcal{O}_k^i$. (The cost of the oracle $\mathcal{O}_k^i$ at block q is $\langle \widetilde{\boldsymbol{a}}_{q,k}^i, \boldsymbol{v}_{q,k}^i \rangle$.)
23: **end for**
24: **end for**
25: **end for**

5. Experiments

We run the algorithm on a movie recommendation problem, with the goal of identifying a set of k movies that satisfy all users. Our setting is closely related to the one in Mokhtari et al. (2018a) and Xie et al. (2019). We use the MovieLens dataset, which contains one million ratings ranging from 1 to 5 from 6000 users on 3883 movies. We divided the data set into T batches $B_1, \ldots, B_T$, with each batch B_t containing ratings from 50 users. We chose Complete, Line, Grid, and Erdos-Renyi graphs with linked probability 0.2. We set the number of nodes/agents equals to 10, 25, and 50. At each iteration t, the agent i receives

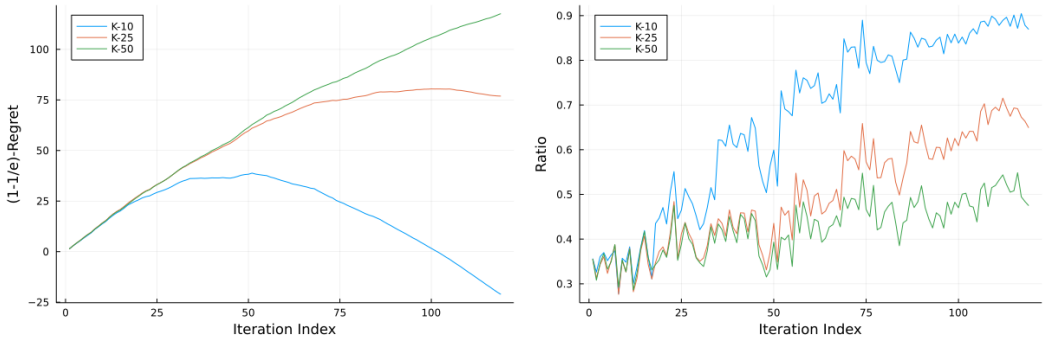
a subset of ratings $B_t^i \subset B_t$. Let $\mathcal{M}$ be the set of movies and $\mathcal{U}$ the set of users; we note $r(u, m)$ the rating of user $u \in \mathcal{U}$ for movies $m \in \mathcal{M}$. Let $S \subset \mathcal{M}$ a collection of movies such that $|S| = k$, the facility location function associated to each agent i denoted,

$$f(B_t^i, S) = f_t^i(S) = \frac{1}{|B_t^i|} \sum_{u \in B_t^i} \max_{m \in S} r(u, m) \tag{8}$$

We denote by $\mathcal{K} = \left\{ \boldsymbol{x} \in [0,1]^d \,|\, \sum_{j=1}^d \boldsymbol{x}_j = k \right\}$. The multilinear extension of f_i^t is defined as,

$$F_t^i(\boldsymbol{x}) = \sum_{S \subset \mathcal{M}} f_t^i(S) \prod_{j \in S} \boldsymbol{x}_j \prod_{\ell \notin S} (1 - \boldsymbol{x}_\ell), \quad \forall \boldsymbol{x} \in \mathcal{K} \tag{9}$$

The goal is to maximize the global objective function $F_t(\boldsymbol{x}) = \frac{1}{n} \sum_{i=1}^n F_t^i(\boldsymbol{x})$, subject to $\boldsymbol{x} \in \mathcal{K}$ while using only local communication and partial information for each local functions.



(a) $(1 - 1/e)$-Regret on Complete Graph (b) Ratio on Complete graph

Figure 1: Algorithm performance on complete graph. The number of nodes is 10, 25 and 50.

Figure 1a shows the $\left(1 - \frac{1}{e}\right)$-regret of the algorithm for $k = 20$ on a complete graph with different node's configuration. We observe that increasing network size leads to a decrease in regret value, which is expected in a decentralized setting because information distributed across a larger set of nodes makes reaching consensus more difficult. Recall that the algorithm uses the same value for each function f_t in block q. If we set $K = 17$ and $Q = 6$, we can expect a stepwise-like curve since the objective function's value changes significantly at each round $t \mod 17$. In a small graph configuration, this value change is more pronounced, bringing the cumulative sum of the objective function closer to the $\left(1 - \frac{1}{e}\right)$-optimal value. Figure 1b depicts the ratio of our algorithm's objective value on a complete graph to an offline centralized Frank-Wolfe. As t increases, the ratio approaches one, demonstrating that our algorithm's performance is comparable to that of an offline setting if we run the algorithm for many rounds, particularly in a 10-nodes configuration. Thus, the results validate our theoretical analysis in the previous section.

Figure 2 shows the average value of the objective function over T rounds for all graph types when the number of movie suggestions k is varied in a 50-node configuration. The

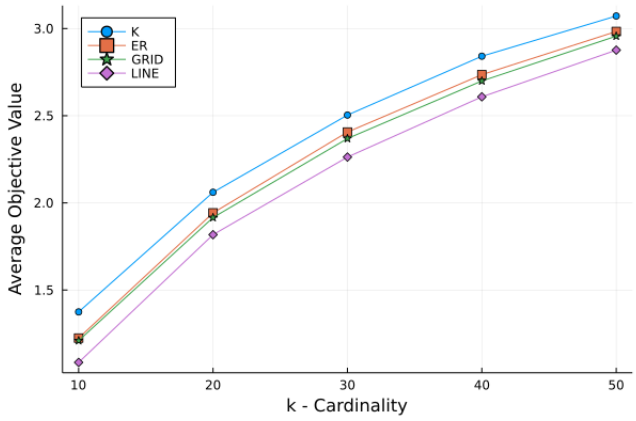


Figure 2: Average objective value over T rounds in function of the cardinality constraint.

average degree for Erdos-Renyi, Complete, Grid, and Line is $5.8, 51, 5.4$, and 4, respectively. As a result, we observe lower performance on less connected graphs when compared to other graph settings. We also notice that increasing the value of k is equivalent to relaxing the cardinality constraint, which results in better performance on the objective function.

6. Concluding remarks

In this paper, we propose a decentralized online algorithm for optimizing convex and monotone continuous DR-submodular functions with regret and $\left(1-\frac{1}{e}\right)$-regret bound of $O(T^{4/5})$. The extension of the algorithm to the bandit setting ensures a $\left(1-\frac{1}{e}\right)$-regret bound of $O(T^{8/9})$. A detailed analysis is given when the constraint set is either a general convex set or a downward-closed convex set under full information and bandit settings, respectively. In addition, the experiment results on a real-life movie recommendation problem assess the interest of the proposed algorithm for learning in decentralized settings.

References

Andrew An Bian, Baharan Mirzasoleiman, Joachim Buhmann, and Andreas Krause. Guaranteed non-convex optimization: Submodular maximization over continuous domains. In *Artificial Intelligence and Statistics*, pages 111–120, 2017.

Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40 (6):1740–1766, 2011.

Lin Chen, Christopher Harshaw, Hamed Hassani, and Amin Karbasi. Projection-free online optimization with stochastic gradient: From convexity to submodularity. In *Proceedings of the 35th International Conference on Machine Learning*, pages 814–823, 2018.

Lin Chen, Mingrui Zhang, Hamed Hassani, and Amin Karbasi. Black box submodular maximization: Discrete and continuous settings. In *The 23rd International Conference on Ar-*

tificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy], Proceedings of Machine Learning Research, pages 1058–1070. PMLR, 2020.

L. Deori, K. Margellos, and M. Prandini. On decentralized convex optimization in a multi-agent setting with separable constraints and its application to optimal charging of electric vehicles. In *IEEE Conference on Decision and Control (CDC)*, pages 6044–6049, 2016.

J. C. Duchi, A. Agarwal, and M. J. Wainwright. Dual averaging for distributed optimization: Convergence analysis and network scaling. *IEEE Transactions on Automatic Control*, 57 (3):592–606, 2012.

Abraham D Flaxman, Adam Tauman Kalai, and H Brendan McMahan. Online convex optimization in the bandit setting: gradient descent without a gradient. In *Proc. 16th Symposium on Discrete Algorithms*, pages 385–394, 2005.

Hamed Hassani, Mahdi Soltanolkotabi, and Amin Karbasi. Gradient methods for submodular maximization. In *Advances in Neural Information Processing Systems*, pages 5841–5851, 2017.

Elad Hazan. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

S. Hosseini, A. Chapman, and M. Mesbahi. Online distributed optimization via dual averaging. In *52nd IEEE Conference on Decision and Control*, pages 1484–1489, 2013.

Aryan Mokhtari, Hamed Hassani, and Amin Karbasi. Decentralized submodular maximization: Bridging discrete and continuous settings. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, Proceedings of Machine Learning Research, pages 3613–3622. PMLR, 2018a.

Aryan Mokhtari, Hamed Hassani, and Amin Karbasi. Conditional gradient method for stochastic submodular maximization: Closing the gap. In *Procs of the International Conference on Artificial Intelligence and Statistics*, volume 84, pages 1886–1895, 2018b.

A. Reisizadeh, A. Mokhtari, H. Hassani, and R. Pedarsani. An exact quantized decentralized gradient descent algorithm. *IEEE Transactions on Signal Processing*, 67(19):4934–4947, 2019.

H. Wai, J. Lafond, A. Scaglione, and E. Moulines. Decentralized frank–wolfe algorithm for convex and nonconvex problems. *IEEE Transactions on Automatic Control*, 62(11): 5522–5537, 2017.

Jiahao Xie, Chao Zhang, Zebang Shen, Chao Mi, and Hui Qian. Decentralized gradient tracking for continuous dr-submodular maximization. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, Proceedings of Machine Learning Research, pages 2897–2906. PMLR, 2019.

F. Yan, S. Sundaram, S. V. N. Vishwanathan, and Y. Qi. Distributed autonomous online learning: Regrets and intrinsic privacy-preserving properties. *IEEE Transactions on Knowledge and Data Engineering*, 25(11):2483–2493, 2013.

K. Yuan, Q. Ling, and W. Yin. On the convergence of decentralized gradient descent. *SIAM Journal on Optimization*, 26(3):1835–1854, 2016.

Mingrui Zhang, Lin Chen, Hamed Hassani, and Amin Karbasi. Online continuous submodular maximization: From full-information to bandit feedback. In *Advances in Neural Information Processing Systems*, pages 9206–9217, 2019.

W. Zhang, P. Zhao, W. Zhu, S.C.V. Hoi, and T. Zhang. Projection-free distributed online learning in networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 4054–4062, 2017.

W. Zheng, A. Bellet, and P. Gallinari. A distributed frank—wolfe framework for learning low-rank matrices with the trace norm. *Machine Learning*, 107(810):1457–1475, 2018.

Junlong Zhu, Qingtao Wu, Mingchuan Zhang, Ruijuan Zheng, and Keqin Li. Projection-free decentralized online learning for submodular maximization over time-varying networks. *Journal of Machine Learning Research*, 2021.