Decentralized Submodular Maximization: Bridging Discrete and Continuous Settings

Aryan Mokhtari [1] **Hamed Hassani** [2] **Amin Karbasi** [3]

Abstract

In this paper, we showcase the interplay between discrete and continuous optimization in network-structured settings. We propose the first fully decentralized optimization method for a wide class of non-convex objective functions that possess a diminishing returns property. More specifically, given an arbitrary connected network and a global *continuous* submodular function, formed by a sum of local functions, we develop **`Decentralized Continuous Greedy`** (DCG), a message passing algorithm that converges to the tight $(1-1/e)$ approximation factor of the optimum global solution using only local computation and communication. We also provide strong convergence bounds as a function of network size and spectral characteristics of the underlying topology. Interestingly, DCG readily provides a simple recipe for decentralized *discrete* submodular maximization through the means of continuous relaxations. Formally, we demonstrate that by lifting the local discrete functions to continuous domains and using DCG as an interface we can develop a consensus algorithm that also achieves the tight $(1-1/e)$ approximation guarantee of the global discrete solution once a proper rounding scheme is applied.

1. Introduction

In recent years, we have reached unprecedented data volumes that are high dimensional and sit over (clouds of) networked machines. As a result, *decentralized* collection of these data sets along with accompanying distributed optimization methods are not only desirable but very often necessary (Boyd et al., 2011).

The focus of this paper is on decentralized optimization, the goal of which is to maximize/minimize a global objective function –distributed over a network of computing units– through local computation and communications among nodes. A canonical example in machine learning is fitting models using M-estimators where given a set of data points the parameters of the model are estimated through an empirical risk minimization (Vapnik, 1998). Here, the global objective function is defined as an average of local loss functions associated with each data point. Such local loss functions can be convex (e.g., logistic regression, SVM, etc) or non-convex (e.g., non-linear square loss, robust regression, mixture of Gaussians, deep neural nets, etc) (Mei et al., 2016). Due to the sheer volume of data points, these optimization tasks cannot be fulfilled on a single computing cluster node. Instead, we need to opt for decentralized solutions that can efficiently exploit dispersed (and often distant) computational resources linked through a tightly connected network. Furthermore, local computations should be light so that they can be done on single machines. In particular, when the data is high dimensional, extra care should be given to any optimization procedure that relies on projections over the feasibility domain.

In addition to large scale machine learning applications, decentralized optimization is a method of choice in many other domains such as Internet of Things (IoT) (Abu-Elkheir et al., 2013), remote sensing (Ma et al., 2015), multi-robot systems (Tanner & Kumar, 2005), and sensor networks (Rabbat & Nowak, 2004). In such scenarios, individual entities can communicate over a network and interact with the environment by exchanging the data generated through sensing. At the same time they can react to events and trigger actions to control the physical world. These applications highlight another important aspect of decentralized optimization where *private* data is collected by different sensing units (Yang et al., 2017). Here again, we aim to optimize a global objective function while avoiding to share the private data among computing units. Thus, by design, one cannot solve such private optimization problems in a centralized manner and should rely on decentralized solutions where local private

[1]Laboratory for Information and Decision Systems, Massachusetts Institute of Technology [2]Department of Electrical and Systems Engineering, University of Pennsylvania [3]Department of Electrical Engineering and Computer Science, Yale University. Correspondence to: Aryan Mokhtari <aryanm@mit.edu>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

computation is done where the data is collected.

Continuous submodular functions, a broad subclass of non-convex functions with diminishing returns property, have recently received considerable attention (Bach, 2015; Bian et al., 2017). Due to their interesting structures that allow strong approximation guarantees (Mokhtari et al., 2018a; Bian et al., 2017), they have found various applications, including the design of online experiments (Chen et al., 2018), budget and resource allocations (Eghbali & Fazel, 2016; Staib & Jegelka, 2017), and learning assignments (Golovin et al., 2014). However, all the existing work suffer from centralized computing. Given that many information gathering, data summarization, and non-parametric learning problems are inherently related to large-scale submodular maximization, the demand for a fully decentralized solution is immediate. In this paper, we develop the first decentralized framework for both continuous and discrete submodular functions. Our contributions are as follows:

- *Continuous submodular maximization:* For any global objective function that is monotone and continuous DR-submodular and subject to any down-closed and bounded convex body, we develop **`Decentralized Continuous Greedy`**, a decentralized and *projection-free* algorithm that achieves the tight $(1 - 1/e - \epsilon)$ approximation guarantee in $O(1/\epsilon^2)$ rounds of local communication.

- *Discrete submodular maximization:* For any global objective function that is monotone and submodular and subject to any matroid constraint, we develop a discrete variant of **`Decentralized Continuous Greedy`** that achieves the tight $(1 - 1/e - \epsilon)$ approximation ratio in $O(1/\epsilon^3)$ rounds of communication.

All proofs are provided in the supplementary material.

2. Related Work

Decentralized optimization is a challenging problem as nodes only have access to separate components of the global objective function, while they aim to collectively reach the global optimum point. Indeed, one naive approach to tackle this problem is to broadcast local objective functions to all the nodes in the network and then solve the problem locally. However, this scheme requires high communication overhead and disregards the privacy associated with the data of each node. An alternative approach is the master-slave setting (Bekkerman et al., 2011; Shamir et al., 2014; Zhang & Lin, 2015) where at each iteration, nodes use their local data to compute the information needed by the master node. Once the master node receives all the local information, it updates its decision and broadcasts the decision to all the nodes. Although this scheme protects the privacy of nodes it is not robust to machine failures and is prone to high overall communication time. In decentralized methods, these issues are overcame by removing the master node and considering each node as an independent unit that is allowed to exchange information with its neighbors.

Convex decentralized consensus optimization is a relatively mature area with a myriad of primal and dual algorithms (Bertsekas & Tsitsiklis, 1989). Among primal methods, decentralized (sub)gradient descent is perhaps the most well known algorithm which is a mix of local gradient descent and successive averaging (Nedic & Ozdaglar, 2009; Yuan et al., 2016). It also can be interpreted as a penalty method that encourages agreement among neighboring nodes. This latter interpretation has been exploited to solve the penalized objective function using accelerated gradient descent (Jakovetić et al., 2014; Qu & Li, 2017), Newton's method (Mokhtari et al., 2017; Bajovic et al., 2017), or quasi-Newton algorithms (Eisen et al., 2017). The methods that operate in the dual domain consider a constraint that enforces equality between nodes' variables and solve the problem by ascending on the dual function to find optimal Lagrange multipliers. A short list of dual methods are the alternating directions method of multipliers (ADMM) (Boyd et al., 2011), dual ascent algorithm (Rabbat et al., 2005), and augmented Lagrangian methods (Jakovetic et al., 2015; Chatzipanagiotis & Zavlanos, 2015). Recently, there have been many attempts to extend the tools in decentralized consensus optimization to the case that the objective function is non-convex (Di Lorenzo & Scutari, 2016; Sun et al., 2016; Hajinezhad et al., 2016; Tatarenko & Touri, 2017). However, such works are mainly concerned with reaching a stationary point and naturally cannot provide any optimality guarantee.

In this paper, our focus is to provide the first decentralized algorithms for both discrete and continuous submodular functions. It is known that the centralized greedy approach of (Nemhauser et al., 1978), and its many variants (Feige et al., 2011; Buchbinder et al., 2015; 2014; Feldman et al., 2017; Mirzasoleiman et al., 2016), reach tight approximation guarantees in various scenarios. As such methods are sequential in nature, they do not scale to massive datasets. To partially resolve this issue, MapReduce style methods, with a master-slave architecture, have been proposed (Mirzasoleiman et al., 2013; Kumar et al., 2015; da Ponte Barbosa et al., 2015; Mirrokni & Zadimoghaddam, 2015; Qu et al., 2015).

One can extend the notion of diminishing returns to continuous domains (Wolsey, 1982; Bach, 2015). Even though continuous submodular functions are not generally convex (nor concave) Hassani et al. (2017) showed that in the monotone setting and subject to a general bounded convex body constraint, stochastic gradient methods can achieve a $1/2$ approximation guarantee. The approximation guarantee can be tightened to $(1 - 1/e)$ by using Frank-Wolfe (Bian et al.,

2017) or stochastic Frank-Wolfe (Mokhtari et al., 2018a).

3. Notation and Background

In this section, we review the notation that we use throughout the paper. We then give the precise definition of submodularity in discrete and continuous domains.

Notation. Lowercase boldface $\mathbf{v}$ denotes a vector and uppercase boldface $\mathbf{W}$ a matrix. The i-th element of $\mathbf{v}$ is written as v_i and the element on the i-th row and j-th column of $\mathbf{W}$ is denoted by $w_{i,j}$. We use $\|\mathbf{v}\|$ to denote the Euclidean norm of vector $\mathbf{v}$ and $\|\mathbf{W}\|$ to denote the spectral norm of matrix $\mathbf{W}$. The null space of matrix $\mathbf{W}$ is denoted by $\text{null}(\mathbf{W})$. The inner product of vectors $\mathbf{x}, \mathbf{y}$ is indicated by $\langle \mathbf{x}, \mathbf{y} \rangle$, and the transpose of a vector $\mathbf{v}$ or matrix $\mathbf{W}$ are denoted by $\mathbf{v}^\dagger$ and $\mathbf{W}^\dagger$, respectively. The vector $\mathbf{1}_n \in \mathbb{R}^n$ is the vector of all ones with n components, and the vector $\mathbf{0}_p \in \mathbb{R}^p$ is the vector of all zeros with p components.

Submodulary. A *set* function $f : 2^V \to \mathbb{R}_+$, defined on the ground set V, is called submodular if for all $A, B \subseteq V$, we have $f(A) + f(B) \geq f(A \cap B) + f(A \cup B)$. We often need to maximize submodular functions subject to a down-closed set family $\mathcal{I}$. In particular, we say $\mathcal{I} \subset 2^V$ is a matroid if 1) for any $A \subset B \subset V$, if $B \in \mathcal{I}$, then $A \in \mathcal{I}$ and 2) for any $A, B \in \mathcal{I}$ if $|A| < |B|$, then there is an element $e \in B$ such that $A \cup \{e\} \in \mathcal{I}$.

The notion of submodularity goes beyond the discrete domain (Wolsey, 1982; Vondrák, 2007; Bach, 2015). Consider a continuous function $F : \mathcal{X} \to \mathbb{R}_+$ where the set $\mathcal{X} \subseteq \mathbb{R}^p$ is of the form $\mathcal{X} = \prod_{i=1}^p \mathcal{X}_i$ and each $\mathcal{X}_i$ is a compact subset of $\mathbb{R}_+$. We call the *continuous* function F submodular if for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}$ we have

$$F(\mathbf{x}) + F(\mathbf{y}) \geq F(\mathbf{x} \vee \mathbf{y}) + F(\mathbf{x} \wedge \mathbf{y}), \tag{1}$$

where $\mathbf{x} \vee \mathbf{y} := \max(\mathbf{x}, \mathbf{y})$ (component-wise) and $\mathbf{x} \wedge \mathbf{y} := \min(\mathbf{x}, \mathbf{y})$ (component-wise). In this paper, our focus is on differentiable continuous submodular functions with two additional properties: monotonicity and diminishing returns. Formally, a submodular function F is monotone if

$$\mathbf{x} \leq \mathbf{y} \quad \Longrightarrow \quad F(\mathbf{x}) \leq F(\mathbf{y}), \tag{2}$$

for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}$. Note that $\mathbf{x} \leq \mathbf{y}$ in (2) means that $x_i \leq y_i$ for all $i = 1, \ldots, p$. Furthermore, a differentiable submodular function F is called *DR-submodular* (i.e., shows diminishing returns) if the gradients are antitone, namely, for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}$ we have

$$\mathbf{x} \leq \mathbf{y} \quad \Longrightarrow \quad \nabla F(\mathbf{x}) \geq \nabla F(\mathbf{y}). \tag{3}$$

When the function F is twice differentiable, submodularity implies that all cross-second-derivatives are non-positive (Bach, 2015), and DR-submodularity implies that all second-derivatives are non-positive (Bian et al., 2017) In this work, we consider the maximization of continuous submodular functions subject to *down-closed convex bodies* $\mathcal{C} \subset \mathbb{R}_+^p$ defined as follows. For any two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}_+^p$, where $\mathbf{x} \leq \mathbf{y}$, down-closedness means that if $\mathbf{y} \in \mathcal{C}$, then so is $\mathbf{x} \in \mathcal{C}$. Note that for a down-closed set we have $\mathbf{0}_p \in \mathcal{C}$.

4. Decentralized Submodular Maximization

In this section, we state the problem of decentralized submodular maximization in continuous and discrete settings.

Continuous Case. We consider a set of n computing machines/sensors that communicate over a graph to maximize a global objective function. Each machine can be viewed as a node $i \in \mathcal{N} \triangleq \{1, \cdots, n\}$. We further assume that the possible communication links among nodes are given by a bidirectional connected *communication graph* $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ where each node can only communicate with its neighbors in $\mathcal{G}$. We formally use $\mathcal{N}_i$ to denote node i's neighbors. In our setting, we assume that each node $i \in \mathcal{N}$ has access to a local function $F_i : \mathcal{X} \to \mathbb{R}_+$. The nodes cooperate in order to maximize the aggregate monotone and continuous DR-submodular function $F : \mathcal{X} \to \mathbb{R}_+$ subject to a down-closed convex body $\mathcal{C} \subset \mathcal{X} \subset \mathbb{R}_+^p$, i.e.,

$$\max_{\mathbf{x} \in \mathcal{C}} F(\mathbf{x}) \;=\; \max_{\mathbf{x} \in \mathcal{C}} \frac{1}{n} \sum_{i=1}^n F_i(\mathbf{x}). \tag{4}$$

The goal is to design a message passing algorithm to solve (4) such that: (i) at each iteration t, the nodes send their messages (and share their information) to their neighbors in $\mathcal{G}$, and (ii) as t grows, all the nodes reach to a point $\mathbf{x} \in \mathcal{C}$ that provides a (near-) optimal solution for (4).

Discrete Case. Let us now consider the discrete counterpart of problem (4). In this setting, each node $i \in \mathcal{N}$ has access to a local *set* function $f_i : 2^V \to \mathbb{R}_+$. The nodes cooperate in maximizing the aggregate monotone submodular function $f : 2^V \to \mathbb{R}_+$ subject to a matroid constraint $\mathcal{I}$, i.e.

$$\max_{S \in \mathcal{I}} f(S) \;=\; \max_{S \in \mathcal{I}} \frac{1}{n} \sum_{i=1}^n f_i(S). \tag{5}$$

Note that even in the centralized case, and under reasonable complexity-theoretic assumptions, the best approximation guarantee we can achieve for Problems (4) and (5) is $(1 - 1/e)$ (Feige, 1998). In the following, we show that it is possible to achieve the same approximation guarantee in a decentralized setting.

5. Decentralized Continuous Greedy Method

In this section, we introduce the **`Decentralized Continuous Greedy`** (DCG) algorithm for solving Problem (4). Recall that in a decentralized setting, the nodes

have to cooperate (i.e., send messages to their neighbors) in order to solve the global optimization problem. We will explain how such messages are designed and communicated in DCG. Each node i in the network keeps track of two local variables $\mathbf{x}_i, \mathbf{d}_i \in \mathbb{R}^p$ which are iteratively updated at each round t using the information gathered from the neighboring nodes. The vector $\mathbf{x}_i^t$ is the local decision variable of node i at step t whose value we expect to eventually converge to the $(1-1/e)$ fraction of the optimal solution of Problem (4). The vector $\mathbf{d}_i^t$ is the estimate of the gradient of the global objective function that node i keeps at step t.

To properly incorporate the received information from their neighbors, nodes should assign nonnegative weights to their neighbors. Define $w_{ij} \geq 0$ to be the weight that node i assigns to node j. These weights indicate the effect of (variable or gradient) information nodes received from their neighbors in order to update their local (variable or gradient) information. Indeed, the weights w_{ij} must fulfill some requirements (later described in Assumption 1), but they are design parameters of DCG and can be properly chosen by the nodes prior to the implementation of the algorithm.

The first step at each round t of DCG is updating the local gradient approximation vectors $\mathbf{d}_i^t$ using local and neighboring gradient information. In particular, node i computes its vector $\mathbf{d}_i^t$ according to the update rule

$$\mathbf{d}_i^t = (1-\alpha) \sum_{j \in \mathcal{N}_i \cup \{i\}} w_{ij} \mathbf{d}_j^{t-1} \; + \; \alpha \nabla F_i(\mathbf{x}_i^t), \qquad (6)$$

where $\alpha \in [0,1]$ is an averaging coefficient. Note that the sum $\sum_{j \in \mathcal{N}_i \cup \{i\}} w_{ij} \mathbf{d}_j^{t-1}$ in (6) is a weighted average of node i's vector $\mathbf{d}_i^{t-1}$ and its neighbors $\mathbf{d}_j^{t-1}$, evaluated at step $t-1$. Hence, node i computes the vector $\mathbf{d}_i^t$ by evaluating a weighted average of its current local gradient $\nabla F_i(\mathbf{x}_i^t)$ and the local and neighboring gradient information at step $t-1$, i.e., $\sum_{j \in \mathcal{N}_i \cup \{i\}} w_{ij} \mathbf{d}_j^{t-1}$. Since the vector $\mathbf{d}_i^t$ is evaluated by aggregating gradient information from neighboring nodes, it is reasonable to expect that $\mathbf{d}_i^t$ becomes a proper approximation for the global objective function gradient $(1/n) \sum_{k=1}^n \nabla f_k(x)$ as time progresses. Note that to implement the update in (6) nodes should exchange their local vectors $\mathbf{d}_i^t$ with their neighbors.

Using the gradient approximation vector $\mathbf{d}_i^t$, each node i evaluates its local ascent direction $\mathbf{v}_i^t$ by solving

$$\mathbf{v}_i^t = \operatorname*{argmax}_{\mathbf{v} \in \mathcal{C}} \langle \mathbf{d}_i^t, \mathbf{v} \rangle. \qquad (7)$$

The update in (7) is also known as *conditional gradient* update. Ideally, in a conditional gradient method, we should choose the feasible direction $\mathbf{v} \in \mathcal{C}$ that maximizes the inner product by the full gradient vector $\frac{1}{n} \sum_{k=1}^n \nabla F_k(\mathbf{x}_i^t)$. However, since in the decentralized setting the exact gradient $\frac{1}{n} \sum_{k=1}^n \nabla F_k(\mathbf{x}_i^t)$ is not available at the i-th node, we replace it by its current approximation $\mathbf{d}_i^t$ and hence we obtain the update rule (7).

Algorithm 1 DCG at node i

Require: Stepsize α and weights w_{ij} for $j \in \mathcal{N}_i \cup \{i\}$

1: Initialize local vectors as $\mathbf{x}_i^0 = \mathbf{d}_i^0 = \mathbf{0}_p$
2: Initialize neighbor's vectors as $\mathbf{x}_j^0 = \mathbf{d}_j^0 = \mathbf{0}_p$ if $j \in \mathcal{N}_i$
3: **for** $t = 1, 2, \ldots, T$ **do**
4: Compute $\mathbf{d}_i^t = (1-\alpha) \sum_{j \in \mathcal{N}_i \cup \{i\}} w_{ij} \mathbf{d}_j^{t-1} + \alpha \nabla F_i(\mathbf{x}_i^t)$;
5: Exchange $\mathbf{d}_i^t$ with neighboring nodes $j \in \mathcal{N}_i$
6: Evaluate $\mathbf{v}_i^t = \operatorname{argmax}_{\mathbf{v} \in \mathcal{C}} \langle \mathbf{d}_i^t, \mathbf{v} \rangle$;
7: Update the variable $\mathbf{x}_i^{t+1} = \sum_{j \in \mathcal{N}_i \cup \{i\}} w_{ij} \mathbf{x}_j^t + \frac{1}{T} \mathbf{v}_i^t$;
8: Exchange $\mathbf{x}_i^{t+1}$ with neighboring nodes $j \in \mathcal{N}_i$
9: **end for**

After computing the local ascent directions $\mathbf{v}_i^t$, the nodes update their local variables x_i^t by averaging their local and neighboring iterates and ascend in the direction $\mathbf{v}_i^t$ with stepsize $1/T$ where T is the total number of iterations, i.e.,

$$\mathbf{x}_i^{t+1} = \sum_{j \in \mathcal{N}_i \cup \{i\}} w_{ij} \mathbf{x}_j^t \; + \; \frac{1}{T} \mathbf{v}_i^t. \qquad (8)$$

The update rule (8) ensures that the neighboring iterates are not far from each other via the averaging term $\sum_{j \in \mathcal{N}_i \cup \{i\}} w_{ij} \mathbf{x}_j^t$, while the iterates approach the optimal maximizer of the global objective function by ascending in the conditional gradient direction $\mathbf{v}_i^t$. The update in (8) requires a round of local communication among neighbors to exchange their local variables $\mathbf{x}_i^t$. The steps of the DCG method are summarized in Algorithm 1.

Indeed, the weights w_{ij} that nodes assign to each other cannot be arbitrary. In the following, we formalize the conditions that they should satisfy (Yuan et al., 2016).

Assumption 1 *The weights that nodes assign to each other are nonegative, i.e., $w_{ij} \geq 0$ for all $i, j \in \mathcal{N}$, and if node j is not a neighbor of node i then the corresponding weight is zero, i.e., $w_{ij} = 0$ if $j \notin \mathcal{N}_i$. Further, the weight matrix $\mathbf{W} \in \mathbb{R}^{n \times n}$ with entries w_{ij} satisfies*

$$\mathbf{W}^\dagger = \mathbf{W}, \; \mathbf{W}\mathbf{1}_n = \mathbf{1}_n, \; \text{null}(\mathbf{I} - \mathbf{W}) = \text{span}(\mathbf{1}_n). \quad (9)$$

The first condition in (9) ensures that the weights are symmetric, i.e., $w_{ij} = w_{ji}$. The second condition guarantees the weights that each node assigns to itself and its neighbors sum up to 1, i.e., $\sum_{j=1}^n w_{ij} = 1$ for all i. Note that the condition $\mathbf{W}\mathbf{1}_n = \mathbf{1}_n$ implies that $\mathbf{I} - \mathbf{W}$ is rank deficient. Hence, the last condition in (9) ensures that the rank of $\mathbf{I} - \mathbf{W}$ is exactly $n-1$. Indeed, it is possible to optimally

design the weight matrix $\mathbf{W}$ to accelerate the averaging process as discussed in (Boyd et al., 2004), but this is not the focus of this paper. We emphasize that $\mathbf{W}$ is not a problem parameter, and we design it prior to running DCG.

Notice that the stepsize $1/T$ and the conditions in Assumption 1 on the weights w_{ij} are needed to ensure that the local variables $\mathbf{x}_i^t$ are in the feasible set $\mathcal{C}$, as stated in the following proposition.

Proposition 1 *Consider the DCG method outlined in Algorithm 1. If Assumption 1 holds and nodes start from $\mathbf{x}_i^0 = \mathbf{0}_p \in \mathcal{C}$, then the local iterates $\mathbf{x}_i^t$ are always in the feasible set $\mathcal{C}$, i.e., $\mathbf{x}_i^t \in \mathcal{C}$ for all $i \in \mathcal{N}$ and $t = 1, \dots, T$.*

Let us now explain how DCG relates to and innovates beyond the exisiting work in submodular maximization as well as decentralized convex optimization. Note that in order to solve Problem (4) in a *centralized* fashion (i.e., when every node has access to *all* the local functions) we can use the continuous greedy algorithm (Vondrák, 2008), a variant of the conditional gradient method. However, in decentralized settings, nodes have only access to their local gradients, and therefore, continuous greedy is not implementable. Similar to the decentralized convex optimization, we can address this issue via local information aggregation. Our proposed DCG method incorporates the idea of choosing the ascent direction according to a conditional gradient update as is done in the continuous greedy algorithm (i.e., the update rule (7)), while it aggregates the global objective function information through local communications with neighboring nodes (i.e., the update rule (8)). Unlike traditional consensus optimization methods that require exchanging nodes' local variables only (Nedic & Ozdaglar, 2009; Nedic et al., 2010), DCG also requires exchanging local gradient vectors to achieve a $(1 - 1/e)$ fraction of the optimal solution at each node (i.e., the update rule (6)). This major difference is due to the fact that in conditional gradient methods, unlike proximal gradient algorithms, the local gradients can not be used instead of the global gradient. In other words, in the update rule (7), we can not use the local gradients $\nabla F_i(\mathbf{x}_i^t)$ in lieu of $\mathbf{d}_i^t$. Indeed, there are settings for which such a replacement provides arbitrarily bad solutions. We formally characterize the convergence of DCG in Theorem 1.

5.1. Extension to the Discrete Setting

In this section we show how DCG can be used for maximizing a decentralized submodular *set* function f, namely Problem (5), through its continuous relaxation. Formally, in lieu of solving Problem (5), we can form the following decentralized continuous optimization problem

$$\max_{\mathbf{x} \in \mathcal{C}} \frac{1}{n} \sum_{i=1}^{n} F_i(\mathbf{x}), \tag{10}$$

Algorithm 2 Discrete DCG at node i

Require: $\alpha, \phi \in [0, 1]$ and weights w_{ij} for $j \in \mathcal{N}_i \cup \{i\}$
1: Initialize local vectors as $\mathbf{x}_i^0 = \mathbf{d}_i^0 = \mathbf{g}_i^0 = \mathbf{0}$
2: Initialize neighbor's vectors as $\mathbf{x}_j^0 = \mathbf{d}_j^0 = \mathbf{0}$ if $j \in \mathcal{N}_i$
3: **for** $t = 1, 2, \dots, T$ **do**
4: Compute $\mathbf{g}_i^t = (1 - \phi)\mathbf{g}_i^{t-1} + \phi \nabla \tilde{F}_i(\mathbf{x}_i^t)$;
5: Compute $\mathbf{d}_i^t = (1 - \alpha) \sum_{j \in \mathcal{N}_i \cup \{i\}} w_{ij} \mathbf{d}_j^{t-1} + \alpha \mathbf{g}_i^t$;
6: Exchange $\mathbf{d}_i^t$ with neighboring nodes $j \in \mathcal{N}_i$
7: Evaluate $\mathbf{v}_i^t = \operatorname{argmax}_{\mathbf{v} \in \mathcal{C}} \langle \mathbf{d}_i^t, \mathbf{v} \rangle$;
8: Update the variable $\mathbf{x}_i^{t+1} = \sum_{j \in \mathcal{N}_i \cup \{i\}} w_{ij} \mathbf{x}_j^t + \frac{1}{T} \mathbf{v}_i^t$;
9: Exchange $\mathbf{x}_i^{t+1}$ with neighboring nodes $j \in \mathcal{N}_i$;
10: **end for**
11: Apply proper rounding to obtain a solution for (5);

where F_i is the multilinear extension of f_i defined as

$$F_i(\mathbf{x}) = \sum_{S \subset V} f_i(S) \prod_{i \in S} x_i \prod_{j \notin S} (1 - x_j), \tag{11}$$

and the down-closed convex set $\mathcal{C} = \text{conv}\{1_I : I \in \mathcal{I}\}$ is the matroid polytope. Note that the discrete and continuous optimization formulations lead to the same optimal value (Calinescu et al., 2011).

Based on the expression in (11), computing the full gradient ∇F_i at each node i will require an exponential computation in terms of $|V|$, since the number of summands in (11) is $2^{|V|}$. As a result, in the discrete setting, we will slightly modify the DCG algorithm and work with *unbiased estimates* of the gradient that can be computed in time $O(|V|)$ (see Appendix 9.7 for one such estimator). More precisely, in the discrete setting, each node $i \in \mathcal{N}$ updates three local variables $\mathbf{x}_i^t, \mathbf{d}_i^t, \mathbf{g}_i^t \in \mathbb{R}^{|V|}$. The variables $\mathbf{x}_i^t, \mathbf{d}_i^t$ play the same role as in DCG and are updated using the messages received from the neighboring nodes. The variable $\mathbf{g}_i^t$ at node i is defined to approximate the local gradient $\nabla F_i(\mathbf{x}_i^t)$. Consider the vector $\nabla \tilde{F}_i(\mathbf{x}_i^t)$ as an unbiased estimator of the local gradient $\nabla F_i(\mathbf{x}_i^t)$ at time t, and define the vector $\mathbf{g}_i^t$ as the outcome of the recursion

$$\mathbf{g}_i^t = (1 - \phi)\mathbf{g}_i^{t-1} + \phi \nabla \tilde{F}_i(\mathbf{x}_i^t), \tag{12}$$

where $\phi \in [0, 1]$ is the averaging parameter. We initialize all vectors as $\mathbf{g}_i^0 = \mathbf{0} \in \mathbb{R}^{|V|}$. It was shown recently (Mokhtari et al., 2018a;b) that the averaging technique in (12) reduces the noise of the gradient approximations. Therefore, the sequence of $\mathbf{g}_i^t$ approaches the true local gradient $\nabla F_i(\mathbf{x}_i^t)$ as time progresses.

The steps of the **`Decentralized Continuous Greedy`** for the discrete setting is summarized in Algo-

rithm 2. Note that the major difference between the Discrete DCG method (Algorithm 2) and the continuous DCG method (Algorithm 1) is in Step 5 in which the exact local gradient $\nabla F_i(\mathbf{x}_i^t)$ is replaced by the stochastic approximation $\mathbf{g}_i^t$ which only requires access to the computationally cheap unbiased gradient estimator $\nabla \tilde{F}_i(\mathbf{x}_i^t)$. The communication complexity of both the discrete and continuous versions of DCG are the same at each round. However, since we are using unbiased estimations of the local gradients $\nabla F_i(\mathbf{x}_i)$, the Discrete DCG takes more rounds to converge to a near-optimal solution compared to continuous DCG. We characterize the convergence of Discrete DCG in Theorem 2. Further, the implementation of Discrete DCG requires rounding the continuous solution to obtain a discrete solution for the original problem without any loss in terms of the objective function value. The provably lossless rounding schemes include the pipage rounding (Calinescu et al., 2011) and contention resolution (Chekuri et al., 2014).

6. Convergence Analysis

In this section, we study the convergence properties of DCG in both continuous and discrete settings. In this regard, we assume that the following conditions hold.

Assumption 2 *Euclidean distance of the elements in the set $\mathcal{C}$ are uniformly bounded, i.e., for all $\mathbf{x}, \mathbf{y} \in \mathcal{C}$ we have*

$$\|\mathbf{x} - \mathbf{y}\| \leq D. \tag{13}$$

Assumption 3 *The local objective functions $F_i(\mathbf{x})$ are monotone and DR-submodular. Further, their gradients are L-Lipschitz continuous over the set $\mathcal{X}$, i.e., for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}$*

$$\|\nabla F_i(\mathbf{x}) - \nabla F_i(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|. \tag{14}$$

Assumption 4 *The norm of gradients $\|\nabla F_i(\mathbf{x})\|$ are bounded over the convex set $\mathcal{C}$, i.e., for all $\mathbf{x} \in \mathcal{C}$, $i \in \mathcal{N}$,*

$$\|\nabla F_i(\mathbf{x})\| \leq G. \tag{15}$$

The condition in Assumption 2 guarantees that the diameter of the convex set $\mathcal{C}$ is bounded. Assumption 3 is needed to ensure that the local objective functions F_i are smooth. Finally, the condition in Assumption 4 enforces the gradients norm to be bounded over the convex set $\mathcal{C}$. All these assumptions are customary and necessary in the analysis of decentralized algorithms. For more details, please check Section VII-B in Jakovetić et al. (2014).

We proceed to derive a constant factor approximation for DCG. Our main result is stated in Theorem 1. However, to better illustrate the main result, we first need to provide several definitions and technical lemmas. Let us begin by defining the average variables $\bar{\mathbf{x}}^t$ as

$$\bar{\mathbf{x}}^t = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i^t. \tag{16}$$

In the following lemma, we establish an upper bound on the variation in the sequence of average variables $\{\bar{\mathbf{x}}^t\}$.

Lemma 1 *Consider the proposed DCG algorithm defined in Algorithm 1. Further, recall the definition of $\bar{\mathbf{x}}^t$ in (16). If Assumptions 1 and 2 hold, then the difference between two consecutive average vectors is upper bounded by*

$$\|\bar{\mathbf{x}}^{t+1} - \bar{\mathbf{x}}^t\| \leq \frac{D}{T}. \tag{17}$$

Recall that at every node i, the messages are mixed using the coefficients w_{ij}, i.e., the i-th row of the matrix $\mathbf{W}$. It is thus not hard to see that the spectral properties of $\mathbf{W}$ (e.g. the spectral gap) play an important role in the the speed of achieving consensus in decentralized methods.

Definition 1 *Consider the eigenvalues of $\mathbf{W}$ which can be sorted in a nonincreasing order as $1 = \lambda_1(\mathbf{W}) \geq \lambda_2(\mathbf{W}) \cdots \geq \lambda_n(\mathbf{W}) > -1$. Define β as the second largest magnitude of the eigenvalues of $\mathbf{W}$, i.e.,*

$$\beta := \max\{|\lambda_2(\mathbf{W})|, |\lambda_n(\mathbf{W})|\}. \tag{18}$$

As we will see, a mixing matrix $\mathbf{W}$ with smaller β has a larger spectral gap $1 - \beta$ which yields faster convergence (Boyd et al., 2004; Duchi et al., 2012). In the following lemma, we derive an upper bound on the sum of the distances between the local iterates $\mathbf{x}_i^t$ and their average $\bar{\mathbf{x}}^t$, where the bound is a function of the graph spectral gap $1-\beta$, size of the network n, and the total number of iterations T.

Lemma 2 *Consider the proposed DCG algorithm defined in Algorithm 1. Further, recall the definition of $\bar{\mathbf{x}}^t$ in (16). If Assumptions 1 and 2 hold, then for all $t \leq T$ we have*

$$\left(\sum_{i=1}^{n} \left\|\mathbf{x}_i^t - \bar{\mathbf{x}}^t\right\|^2\right)^{1/2} \leq \frac{\sqrt{n}D}{T(1-\beta)}. \tag{19}$$

Let us now define $\bar{\mathbf{d}}^t$ as the average of local gradient approximations $\mathbf{d}_i^t$ at step t, i.e., $\bar{\mathbf{d}}^t = \frac{1}{n}\sum_{i=1}^{n} \mathbf{d}_i^t$. We will show in the following that the vectors $\mathbf{d}_i^t$ also become uniformly close to $\bar{\mathbf{d}}^t$.

Lemma 3 *Consider the proposed DCG algorithm defined in Algorithm 1. If Assumptions 1 and 3 hold, then*

$$\left(\sum_{i=1}^{n} \|\mathbf{d}_i^t - \bar{\mathbf{d}}^t\|^2\right)^{1/2} \leq \frac{\alpha\sqrt{n}G}{1 - \beta(1-\alpha)}. \tag{20}$$

Lemma 3 guarantees that the individual local gradient approximation vectors $\mathbf{d}_i^t$ are close to the average vector $\bar{\mathbf{d}}^t$ if the parameter α is small. To show that the gradient vectors $\mathbf{d}_i^t$, generated by DCG, approximate the gradient of the

global objective function, we further need to show that the average vector $\bar{\mathbf{d}}^t$ approaches the global objective function gradient ∇F. We prove this claim in the following lemma.

Lemma 4 *Consider the proposed DCG algorithm defined in Algorithm 1. If Assumptions 1-4 hold, then*

$$\left\| \bar{\mathbf{d}}^t - \frac{1}{n}\sum_{i=1}^{n} \nabla F_i(\bar{\mathbf{x}}^t) \right\| \leq (1-\alpha)^t G + \left(\frac{(1-\alpha)LD}{\alpha T} + \frac{LD}{T(1-\beta)} \right). \quad (21)$$

By combining Lemmas 3 and 4 and setting $\alpha = 1/\sqrt{T}$ we can conclude that the local gradient approximation vector $\mathbf{d}_i^t$ of each node i is within $\mathcal{O}(1/\sqrt{T})$ distance of the global objective gradient $\nabla F(\bar{\mathbf{x}}^t)$ evaluated at $\bar{\mathbf{x}}^t$. We use this observation in the following theorem to show that the sequence of iterates generated by DCG achieves the tight $(1-1/e)$ approximation ratio of the optimum global solution.

Theorem 1 *Consider the proposed DCG method outlined in Algorithm 1. Further, consider $\mathbf{x}^*$ as the global maximizer of Problem (4). If Assumptions 1-4 hold and we set $\alpha = 1/\sqrt{T}$, for all nodes $j \in \mathcal{N}$, the local variable $\mathbf{x}_j^T$ obtained after T iterations satisfies*

$$F(\mathbf{x}_j^T) \geq (1-e^{-1})F(\mathbf{x}^*) - \frac{LD^2+GD}{T^{1/2}} - \frac{GD}{T^{1/2}(1-\beta)} - \frac{LD^2}{2T} - \frac{GD+LD^2}{T(1-\beta)}. \quad (22)$$

Theorem 1 shows that the sequence of the local variables $\mathbf{x}_j^t$, generated by DCG, is able to achieve the optimal approximation ratio $(1-1/e)$, while the error term vanishes at a sublinear rate of $\mathcal{O}(1/T^{1/2})$, i.e.,

$$F(\mathbf{x}_j^T) \geq (1-1/e)F(\mathbf{x}^*) - \mathcal{O}\left(\frac{1}{(1-\beta)T^{1/2}} \right), \quad (23)$$

which implies that the iterate of *each node* reaches an objective value larger than $(1-1/e-\epsilon)OPT$ after $\mathcal{O}(1/\epsilon^2)$ rounds of communication. It is worth mentioning that the result in Theorem 1 is consistent with classical results in decentralized optimization that the error term vanishes faster for the graphs with larger spectral gap $1-\beta$. We proceed to study the convergence properties of Discrete DCG in Algorithm 2. To do so, we first assume that the variance of the stochastic gradients $\nabla \tilde{F}_i(\mathbf{x})$ used in Discrete DCG is bounded. We justify this assumption in Remark 1.

Assumption 5 *The variance of the unbiased estimators $\nabla \tilde{F}(\mathbf{x})$ is bounded above by σ^2 over the convex set $\mathcal{C}$, i.e., for any $i \in \mathcal{N}$ and any vector $\mathbf{x} \in \mathcal{C}$ we can write*

$$\mathbb{E}\left[\|\nabla \tilde{F}_i(\mathbf{x}) - \nabla F_i(\mathbf{x})\|^2 \right] \leq \sigma^2, \quad (24)$$

where the expectation is with respect to the randomness of the unbiased estimator.

In the following theorem, we show that Discrete DCG achieves a $(1-1/e)$ approximation ration for Problem (5).

Theorem 2 *Consider our proposed Discrete DCG algorithm outlined in Algorithm 2. Recall the definition of the multilinear extension function F_i in (11). If Assumptions 1-5 hold and we set $\alpha = T^{-1/2}$ and $\phi = T^{-2/3}$, then for all nodes $j \in \mathcal{N}$ the local variables $\mathbf{x}_j^T$ obtained after running Discrete DCG for T iterations satisfy*

$$\mathbb{E}\left[F(\mathbf{x}_j^T)\right] \geq (1-e^{-1})F(\mathbf{x}^*) - \mathcal{O}\left(\frac{1}{(1-\beta)T^{1/3}} \right), \quad (25)$$

where $\mathbf{x}^*$ is the global maximizer of Problem (10).

Theorem 2 states that the sequence of iterates generated by Discrete DCG achieves the tight $(1-1/e-\epsilon)$ approximation guarantee for Problem (10) after $\mathcal{O}(1/\epsilon^3)$ iterations.

Remark 1 *For any submodular set function $h : 2^V \to \mathbb{R}$ with associated multilinear extension H, it can be shown that its Lipschitz constant L and the gradient norm G are both bounded above by $m_f\sqrt{|V|}$, where m_f is the maximum marginal value of f, i.e., $m_f = \max_{i\in V} f(\{i\})$ (see, Hassani et al. (2017)). Similarly, it can be shown that for the unbiased estimator in Appendix 9.7 we have $\sigma \leq m_f\sqrt{|V|}$.*

7. Numerical Experiments

We will consider a discrete setting for our experiments and use Algorithm 2 to find a decentralized solution. The main objective is to demonstrate how consensus is reached and how the global objective increases depending on the topology of the network and the parameters of the algorithm.

For our experiments, we have used the MovieLens data set. It consists of 1 million ratings (from 1 to 5) by $M = 6000$ users for $p = 4000$ movies. We consider a network of $n = 100$ nodes. The data has been distributed equally between the nodes of the network, i.e., the set of users has been partitioned into 100 equally-sized sets and each node in the network has access to only one chunk (partition) of the data. The global task is to find a set of k movies that are most satisfactory to *all* the users (the precise formulation will appear shortly). However, as each of the nodes in the network has access to the data of a small portion of the users, the nodes have to cooperate to fulfill the global task.

We consider a well motivated objective function for the experiments. Let $r_{\ell,j}$ denote the rating of user ℓ for movie j (if such a rating does not exist in the data we assign $r_{\ell,j}$ to 0). We associate to each user ℓ a "facility location" objective function $g_\ell(S) = \max_{j\in S} r_{\ell,j}$, where S is any subset of

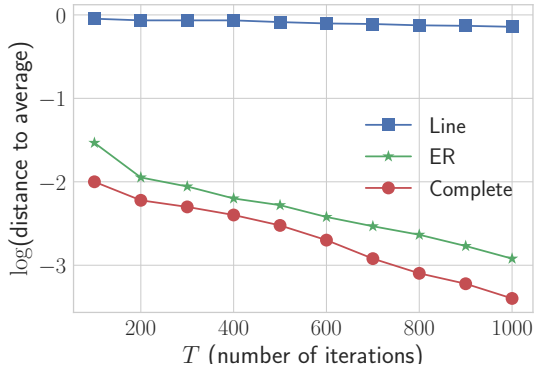

Figure 1. The logarithm of the distance-to-average at final round T is plotted as a function of T. Note that when the underlying graph is complete or Erdos-Renyi (ER) with a good average degree, then consensus will be achieved even for small number of iterations T. However, for poor connected graphs such as the line graph, reaching consensus requires a large number of iterations.

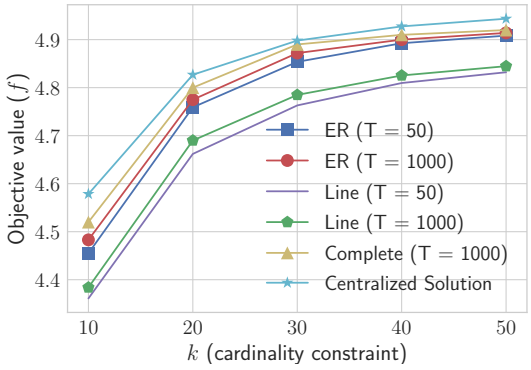

Figure 2. The average objective value is plotted as a function of the cardinality constraint k for different choices of the communication graph as well as number of iterations T. Note that "ER" stands for the Erdos-Reny graph with average degree 5, "Line" stands for the line graph and "Complete" is for the complete graph. We have run Algorithm 2 for $T = 50$ and $T = 1000$.

the movies (i.e. the ground set V is the set of the movies). Such a function shows how much user ℓ will be "satisfied" by a subset S of the movies. Recall that each node i in the network has access to the data of a (small) subset of users which we denote by $\mathcal{U}_i$. The objective function associated with node i is given by $f_i(S) = \sum_{\ell \in \mathcal{U}_i} g_\ell(S)$. With such a choice of the local functions, our global task is hence to solve problem (5) when the matroid $\mathcal{I}$ is the k-uniform matroid (a.k.a. the k-cardinality constraint).

We consider three different choices for the underlying communication graph between the 100 nodes: A line graph (which looks like a simple path from node 1 to node 100), an Erdos-Renyi random graph (with average degree 5), and a complete graph. The matrix $\mathbf{W}$ is chosen as follows (based on each of the three graphs). If (i, j) is and edge of the graph, we let $w_{i,j} = 1/(1 + \max(d_i, d_j))$. If (i, j) is not an edge and i, j are distinct integers, we have $w_{i,j} = 0$. Finally we let $w_{i,i} = 1 - \sum_{j \in \mathcal{N}} w_{i,j}$. It is not hard to show that the above choice for $\mathbf{W}$ satisfies Assumption 1.

Figure 1 shows how consensus is reached w.r.t each of the three underlying networks. To measure consensus, we plot the (logarithm of) distance-to-average value $\frac{1}{n}\sum_{i=1}^{n} ||\mathbf{x}_i^T - \bar{\mathbf{x}}^T||$ as a function of the total number of iterations T averaged over many trials (see (16) for the definition of $\bar{\mathbf{x}}^T$). It is easy to see that the distance to average is small if and only if all the local decisions $\mathbf{x}_i^T$ are close to the average decision $\bar{\mathbf{x}}^T$. As expected, it takes much less time to reach consensus when the underlying graph is fully connected (i.e. complete graph). For the line graph, the convergence is very slow as this graph has the least degree of connectivity.

Figure 2 depicts the obtained objective value of Discrete DCG (Algorithm 2) for the three networks considered above. More precisely, we plot the value $\frac{1}{n}\sum_{i=1}^{n} f(\mathbf{x}_i^T)$ obtained at the end of Algorithm 2 as a function of the cardinality constraint k. We also compare these values with the value obtained by the centralized greedy algorithm (i.e. the centralized solution). A few comments are in order. The performance of Algorithm 2 is close to the centralized solution when the underlying graph is the Erdos-Renyi (with average degree 5) graph or the complete graphs. This is because for both such graphs consensus is achieved from the early stages of the algorithm. By increasing T, we see that the performance becomes closer to the centralized solution. However, when the underlying graph is the line graph, then consensus will not be achieved unless the number of iterations is significantly increased. Consequently, for small number of iterations (e.g. $T \leq 1000$) the performance of the algorithm will not be close to the centralized solution.

8. Conclusion

In this paper, we proposed the first fully decentralized optimization method for maximizing discrete and continuous submodular functions. We developed **`Decentralized Continuous Greedy`** (DCG) that achieves a $(1 - 1/e - \epsilon)$ approximation guarantee with $\mathcal{O}(1/\epsilon^2)$ and $(1/\epsilon^3)$ local rounds of communication in the continuous and discrete settings, respectively.

Acknowledgements

This work was done while A. Mokhtari was visiting the Simons Institute for the Theory of Computing, and his work was partially supported by the DIMACS/Simons Collaboration on Bridging Continuous and Discrete Optimization through NSF grant #CCF-1740425. The work of A. Karbasi was supported by DARPA Young Faculty Award (D16AP00046) and AFOSR YIP (FA9550-18-1-0160).

References

Abu-Elkheir, Mervat, Hayajneh, Mohammad, and Ali, Najah Abu. Data management for the internet of things: Design primitives and solution. *Sensors*, 13(11):15582–15612, 2013.

Bach, F. Submodular functions: from discrete to continuous domains. *arXiv preprint arXiv:1511.00394*, 2015.

Bajovic, Dragana, Jakovetic, Dusan, Krejic, Natasa, and Jerinkic, Natasa Krklec. Newton-like method with diagonal correction for distributed optimization. *SIAM Journal on Optimization*, 27(2):1171–1203, 2017.

Bekkerman, Ron, Bilenko, Mikhail, and Langford, John. *Scaling up machine learning: Parallel and distributed approaches*. Cambridge University Press, 2011.

Bertsekas, Dimitri P and Tsitsiklis, John N. *Parallel and distributed computation: numerical methods*, volume 23. Prentice hall Englewood Cliffs, NJ, 1989.

Bian, Andrew An, Mirzasoleiman, Baharan, Buhmann, Joachim M., and Krause, Andreas. Guaranteed nonconvex optimization: Submodular maximization over continuous domains. In *AISTATS*, 2017.

Boyd, Stephen, Diaconis, Persi, and Xiao, Lin. Fastest mixing markov chain on a graph. *SIAM review*, 46(4): 667–689, 2004.

Boyd, Stephen, Parikh, Neal, Chu, Eric, Peleato, Borja, and Eckstein, Jonathan. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

Buchbinder, Niv, Feldman, Moran, Naor, Joseph, and Schwartz, Roy. Submodular maximization with cardinality constraints. In *SODA 2014*, pp. 1433–1452, 2014.

Buchbinder, Niv, Feldman, Moran, Naor, Joseph, and Schwartz, Roy. A tight linear time (1/2)-approximation for unconstrained submodular maximization. *SIAM Journal on Computing*, 44(5):1384–1402, 2015.

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

Chatzipanagiotis, Nikolaos and Zavlanos, Michael M. On the convergence rate of a distributed augmented Lagrangian optimization algorithm. In *(ACC)*, 2015.

Chekuri, Chandra, Vondrák, Jan, and Zenklusen, Rico. Submodular function maximization via the multilinear relaxation and contention resolution schemes. *SIAM J. Comput.*, 43(6):1831–1879, 2014.

Chen, Lin, Hassani, Hamed, and Karbasi, Amin. Online continuous submodular maximization. In *AISTATS*, 2018.

da Ponte Barbosa, Rafael, Ene, Alina, Nguyen, Huy L., and Ward, Justin. The power of randomization: Distributed submodular maximization on massive datasets. In *ICML*, 2015.

Di Lorenzo, Paolo and Scutari, Gesualdo. Next: In-network nonconvex optimization. *IEEE Trans. on Signal and Information Process. over Networks*, 2(2):120–136, 2016.

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

Eghbali, Reza and Fazel, Maryam. Designing smoothing functions for improved worst-case competitive ratio in online optimization. In *NIPS*, pp. 3279–3287, 2016.

Eisen, Mark, Mokhtari, Aryan, and Ribeiro, Alejandro. Decentralized quasi-Newton methods. *IEEE Transactions on Signal Processing*, 65(10):2613–2628, 2017.

Feige, Uriel. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 1998.

Feige, Uriel, Mirrokni, Vahab S, and Vondrak, Jan. Maximizing non-monotone submodular functions. *SIAM Journal on Computing*, 40(4):1133–1153, 2011.

Feldman, Moran, Harshaw, Christopher, and Karbasi, Amin. Greed is good: Near-optimal submodular maximization via greedy optimization. *arXiv preprint arXiv:1704.01652*, 2017.

Golovin, Daniel, Krause, Andreas, and Streeter, Matthew. Online submodular maximization under a matroid constraint with application to learning assignments. *arXiv preprint arXiv:1407.1082*, 2014.

Hajinezhad, Davood, Hong, Mingyi, Zhao, Tuo, and Wang, Zhaoran. NESTT: A nonconvex primal-dual splitting method for distributed and stochastic optimization. In *NIPS*, 2016.

Hassani, S. Hamed, Soltanolkotabi, Mahdi, and Karbasi, Amin. Gradient methods for submodular maximization. In *NIPS*, pp. 5843–5853, 2017.

Jakovetić, Dušan, Xavier, Joao, and Moura, José MF. Fast distributed gradient methods. *IEEE Transactions on Automatic Control*, 59(5):1131–1146, 2014.

Jakovetic, Dusan, Moura, Jose MF, and Xavier, Joao. Linear convergence rate of a class of distributed augmented Lagrangian algorithms. *Automatic Control, IEEE Transactions on*, 60(4):922–936, 2015.

Kumar, Ravi, Moseley, Benjamin, Vassilvitskii, Sergei, and Vattani, Andrea. Fast greedy algorithms in mapreduce and streaming. *TOPC*, 2015.

Ma, Yan, Wu, Haiping, Wang, Lizhe, Huang, Bormin, Ranjan, Rajiv, Zomaya, Albert, and Jie, Wei. Remote sensing big data computing: Challenges and opportunities. *Future Generation Computer Systems*, 51:47–60, 2015.

Mei, Song, Bai, Yu, and Montanari, Andrea. The landscape of empirical risk for non-convex losses. *arXiv preprint arXiv:1607.06534*, 2016.

Mirrokni, Vahab S. and Zadimoghaddam, Morteza. Randomized composable core-sets for distributed submodular maximization. In *STOC 2015*, pp. 153–162, 2015.

Mirzasoleiman, Baharan, Karbasi, Amin, Sarkar, Rik, and Krause, Andreas. Distributed submodular maximization: Identifying representative elements in massive data. In *NIPS*, 2013.

Mirzasoleiman, Baharan, Badanidiyuru, Ashwinkumar, and Karbasi, Amin. Fast constrained submodular maximization: Personalized data summarization. In *ICML 2016*, pp. 1358–1367, 2016.

Mokhtari, Aryan, Ling, Qing, and Ribeiro, Alejandro. Network Newton distributed optimization methods. *IEEE Trans. on Signal Process.*, 65(1):146–161, 2017.

Mokhtari, Aryan, Hassani, Hamed, and Karbasi, Amin. Conditional gradient method for stochastic submodular maximization: Closing the gap. In *AISTATS*, pp. 1886–1895, 2018a.

Mokhtari, Aryan, Hassani, Hamed, and Karbasi, Amin. Stochastic conditional gradient methods: From convex minimization to submodular maximization. *arXiv preprint arXiv:1804.09554*, 2018b.

Nedic, Angelia and Ozdaglar, Asuman. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, 2009.

Nedic, Angelia, Ozdaglar, Asuman, and Parrilo, Pablo A. Constrained consensus and optimization in multi-agent networks. *IEEE Transactions on Automatic Control*, 55 (4):922–938, 2010.

Nemhauser, George L, Wolsey, Laurence A, and Fisher, Marshall L. An analysis of approximations for maximizing submodular set functions–I. *Mathematical Programming*, 14(1):265–294, 1978.

Qu, Guannan and Li, Na. Accelerated distributed Nesterov gradient descent. *arXiv preprint arXiv:1705.07176*, 2017.

Qu, Guannan, Brown, Dave, and Li, Na. Distributed greedy algorithm for satellite assignment problem with submodular utility function? *IFAC-PapersOnLine*, 2015.

Rabbat, Michael and Nowak, Robert. Distributed optimization in sensor networks. In *Proceedings of the 3rd international symposium on Information processing in sensor networks*, pp. 20–27. ACM, 2004.

Rabbat, Michael G, Nowak, Robert D, et al. Generalized consensus computation in networked systems with erasure links. In *SPAWC*, pp. 1088–1092, 2005.

Shamir, Ohad, Srebro, Nathan, and Zhang, Tong. Communication-efficient distributed optimization using an approximate Newton-type method. In *ICML 2014*, pp. 1000–1008, 2014.

Staib, Matthew and Jegelka, Stefanie. Robust budget allocation via continuous submodular functions. In *Advances in neural information processing systems*, 2017.

Sun, Ying, Scutari, Gesualdo, and Palomar, Daniel. Distributed nonconvex multiagent optimization over time-varying networks. In *Signals, Systems and Computers, 2016 50th Asilomar Conference on*, pp. 788–794, 2016.

Tanner, Herbert G. and Kumar, Amit. Towards decentralization of multi-robot navigation functions. In *ICRA 2005*, pp. 4132–4137, 2005.

Tatarenko, Tatiana and Touri, Behrouz. Non-convex distributed optimization. *IEEE Transactions on Automatic Control*, 62(8):3744–3757, 2017.

Vapnik, Vladimir. *Statistical learning theory*. Wiley, 1998. ISBN 978-0-471-03003-4.

Vondrák, Jan. Submodularity in combinatorial optimization. 2007.

Vondrák, Jan. Optimal approximation for the submodular welfare problem in the value oracle model. In *STOC*, pp. 67–74, 2008.

Wolsey, Laurence A. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2(4):385–393, 1982.

Yang, Yuchen, Wu, Longfei, Yin, Guisheng, Li, Lijie, and Zhao, Hongbin. A survey on security and privacy issues in internet-of-things. *IEEE Internet of Things Journal*, 4 (5):1250–1258, 2017.

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

Zhang, Yuchen and Lin, Xiao. DiSCO: Distributed optimization for self-concordant empirical loss. In *ICML 2015*, pp. 362–370, 2015.