Parallel Algorithm for Non-Monotone DR-Submodular Maximization

Alina Ene [*1] **Huy L. Nguyễn** [*2]

Abstract

In this work, we give a new parallel algorithm for the problem of maximizing a non-monotone diminishing returns submodular function subject to a cardinality constraint. For any desired accuracy ϵ, our algorithm achieves a $1/e - \epsilon$ approximation using $O(\log n \log(1/\epsilon)/\epsilon^3)$ parallel rounds of function evaluations. The approximation guarantee nearly matches the best approximation guarantee known for the problem in the sequential setting and the number of parallel rounds is nearly-optimal for any constant ϵ. Previous algorithms achieve worse approximation guarantees using $\Omega(\log^2 n)$ parallel rounds. Our experimental evaluation suggests that our algorithm obtains solutions whose objective value nearly matches the value obtained by the state of the art sequential algorithms, and it outperforms previous parallel algorithms in number of parallel rounds, iterations, and solution quality.

1. Introduction

In this paper, we study parallel algorithms for the problem of maximizing a non-monotone DR-submodular function subject to a single cardinality constraint. The problem is a generalization of submodular maximization subject to a cardinality constraint. Many recent works have shown that DR-submodular maximization has a wide-range of applications beyond submodular maximization. These applications include maximum a-posteriori (MAP) inference for determinantal point processes (DPP), mean-field inference in log-submodular models, quadratic programming, and revenue maximization in social networks (Kulesza et al., 2012; Gillenwater et al., 2012; Bian et al., 2016; Ito & Fujimaki, 2016; Soma & Yoshida, 2017; Bian et al., 2017; 2018).

The problem of maximizing a DR-submodular function subject to a convex constraint is a notable example of a *non-convex* optimization problem that can be solved with *provable* approximation guarantees. The continuous Greedy algorithm (Vondrák, 2008) developed in the context of the multilinear relaxation framework applies more generally to maximizing DR-submodular functions that are monotone increasing (if $\vec{x} \leq \vec{y}$ coordinate-wise then $f(\vec{x}) \leq f(\vec{y})$). Feldman *et al.* (Feldman et al., 2011) developed a variant of continuous Greedy for non-monotone objectives. Chekuri *et al.* (Chekuri et al., 2015) developed algorithms for both monotone and non-monotone DR-submodular maximization subject to packing constraints that are based on the continuous Greedy and multiplicative weights update framework. The work (Bian et al., 2017) generalized continuous Greedy for submodular functions to the DR-submodular case and developed Frank-Wolfe-style algorithms for maximimizing non-monotone DR-submodular function subject to general convex constraints.

A significant drawback of these algorithms is that they are inherently sequential and adaptive. In fact the highly adaptive nature of these algorithms goes back to the classical greedy algorithm for submodular functions: the algorithm sequentially selects the next element based on the marginal gain on top of previous elements. In certain settings such as feature selection (Khanna et al., 2017) evaluating the objective function is a time-consuming procedure and the main bottleneck of the optimization algorithm and therefore, parallelization is a must. Recent lines of work have focused on addressing these shortcomings and understanding the trade-offs between approximation guarantee, parallelization, and adaptivity. Starting with the work of Balkanski and Singer (Balkanski & Singer, 2018), there have been very recent efforts to understand the tradeoff between approximation guarantee and adaptivity for submodular maximization (Balkanski & Singer, 2018; Ene & Nguyen, 2019; Balkanski et al., 2019; Fahrbach et al., 2019; Chekuri & Quanrud, 2019; Balkanski et al., 2018). The *adaptivity* of an algorithm is the number of sequential rounds of queries it makes to the evaluation oracle of the function, where in every round the algorithm is allowed to make polynomially-many parallel queries. Recently, the work (Fahrbach et al., 2018) gave an algorithm for maximizing a submodular function subject to a cardinality constraint

[*]Equal contribution [1]Department of Computer Science, Boston University [2]Khoury College of Computer and Information Science, Northeastern University. Correspondence to: Alina Ene <aene@bu.edu>, Huy L. Nguyễn <hu.nguyen@northeastern.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

in $O(\log n/\epsilon)$ rounds and $0.031-\epsilon$ approximation. For the general setting of DR-submodular functions with m packing constraints, the work (Ene et al., 2019) gave an algorithm with $O(\log(n/\epsilon)\log(1/\epsilon)\log(m+n)/\epsilon^2)$ rounds and $1/e-\epsilon$ approximation. In the special case of $m=1$ constraint, this algorithm uses $O(\log^2(n)\log(1/\epsilon)/\epsilon^2)$ rounds.

In this work, we develop a new algorithm for DR-submodular maximization subject to a single cardinality constraint using $O(\log n\log(1/\epsilon)/\epsilon^3)$ rounds of adaptivity and obtaining $1/e-\epsilon$ approximation. For small n compared to $1/\epsilon$, this is not an improvement over previous work. However, for large n, the number of rounds is almost a quadratic improvement from $O(\log^2 n)$ in the previous work to the nearly optimal $O(\log n)$ rounds. Our experimental evaluation shows that our algorithm outperforms the state of the art *even in the small n regime* that is advantageous for the existing algorithms.

Theorem 1. *Let $f:[0,1]^n\to\mathbb{R}_+$ be a DR-submodular function and $k\in\mathbb{R}_+$. For every $\epsilon>0$, there is an algorithm for the problem $\max_{\vec{x}\in[0,1]^n:\ \|\vec{x}\|_1\le k} f(\vec{x})$ with the following guarantees:*

- *The algorithm is deterministic if provided oracle access for evaluating f and its gradient ∇f;*
- *The algorithm achieves an approximation guarantee of $\frac{1}{e}-\epsilon$;*
- *The number of rounds of adaptivity is $O\left(\frac{\log n\log(1/\epsilon)}{\epsilon^3}\right)$.*

2. Preliminaries

Let $f:[0,1]^n\to\mathbb{R}_+$ be a non-negative function. The function is *diminishing returns submodular* (DR-submodular) if $\forall\vec{x}\le\vec{y}\in[0,1]^n$ (where $\le$ is coordinate-wise), $\forall i\in[n]$, $\forall\delta\in[0,1]$ such that $\vec{x}+\delta\vec{1}_i$ and $\vec{y}+\delta\vec{1}_i$ are still in $[0,1]^n$, it holds

$$f(\vec{x}+\delta\vec{1}_i)-f(\vec{x})\ge f(\vec{y}+\delta\vec{1}_i)-f(\vec{y}),$$

where $\vec{1}_i$ is the i-th basis vector, i.e., the vector whose i-th entry is 1 and all other entries are 0.

If f is differentiable, f is DR-submodular if and only if $\nabla f(\vec{x})\ge\nabla f(\vec{y})$ for all $\vec{x}\le\vec{y}\in[0,1]^n$. If f is twice-differentiable, f is DR-submodular if and only if all the entries of the Hessian are *non-positive*, i.e., $\frac{\partial^2 f}{\partial x_i\partial x_j}(\vec{x})\le 0$ for all $i,j\in[n]$.

Throughout the paper, we assume that f is differentiable. We assume that we are given black-box access to an oracle for evaluating f and its gradient ∇f. We extend the function f to $\mathbb{R}^n_+$ as follows: $f(\vec{x})=f(\vec{x}\wedge\vec{1})$, where $(\vec{x}\wedge\vec{1})_i=\min\{x_i,1\}$.

An example of a DR-submodular function is the multilinear extension of a submodular function g. The multilinear extension is defined as

$$G(\vec{x})=\mathbb{E}[g(R(\vec{x}))]=\sum_{S\subseteq V}g(S)\prod_{i\in S}\vec{x}_i\prod_{i\in V\setminus S}(1-\vec{x}_i),$$

where $R(\vec{x})$ is a random subset of V where each $i\in V$ is included independently at random with probability $\vec{x}_i$.

Basic notation. We use e.g. $\vec{x}=(\vec{x}_1,\ldots,\vec{x}_n)$ to denote a vector in $\mathbb{R}^n$. We use the following vector operations: $\vec{x}\vee\vec{y}$ is the vector whose i-th coordinate is $\max\{\vec{x}_i,\vec{y}_i\}$; $\vec{x}\wedge\vec{y}$ is the vector whose i-th coordinate is $\min\{\vec{x}_i,\vec{y}_i\}$; $\vec{x}\circ\vec{y}$ is the vector whose i-th coordinate is $\vec{x}_i\cdot\vec{y}_i$. We write $\vec{x}\le\vec{y}$ to denote that $\vec{x}_i\le\vec{y}_i$ for all $i\in[n]$. Let $\vec{0}$ (resp. $\vec{1}$) be the n-dimensional all-zeros (resp. all-ones) vector. Let $\vec{1}_S\in\{0,1\}^V$ denote the indicator vector of $S\subseteq V$, i.e., the vector that has a 1 in entry i if and only if $i\in S$.

We will use the following result that was shown in previous work (Chekuri et al., 2015).

Lemma 2 ((Chekuri et al., 2015), Lemma 7)**.** *Let $f:[0,1]^n\to\mathbb{R}_+$ be a DR-submodular function. For all $\vec{x}^*\in[0,1]^n$ and $\vec{x}\in[0,1]^n$, $f(\vec{x}^*\vee\vec{x})\ge(1-\|\vec{x}\|_\infty)f(\vec{x}^*)$.*

3. The Algorithm

In this section, we present an idealized version of our algorithm where we assume that we can compute exactly the step size on line 16. The idealized algorithm is given in Algorithm 1. In the supplement (Section B), we show how to implement that step efficiently and incur only $O(\epsilon)$ additive error in the approximation.

The algorithm takes as input a target value M and it achieves the desired $1/e-O(\epsilon)$ approximation if M is an $(1+\epsilon)$ approximation of the optimal function value $f(\vec{x}^*)$, i.e., we have $f(\vec{x}^*)\le M\le(1+\epsilon)f(\vec{x}^*)$. As noted in previous work (Ene et al., 2019), it is straightforward to approximately guess such a value M using a single parallel round.

The algorithm maintains two solutions: the solution $\vec{x}$ that will be returned, and a second solution $\vec{z}\ge\vec{x}$. We use $\vec{z}$ to handle the non-monotonicity of the objective. Throughout the algorithm, we have $\vec{z}\ge\vec{x}$ and thus $\nabla f(\vec{z})\le\nabla f(\vec{x})$ by DR-submodularity. Instead of using the gradient at $\vec{x}$ to decide which coordinates to update, we use the gradient at $\vec{z}$. By doing so, we may under-estimate the potential gain of some of the coordinates, but it allows us to deal with the non-monotonicity of the function. This strategy is reminiscent of the gradient lookahead technique of (Ene et al., 2019), although we use it in a very different way.

The algorithm iteratively increases the two solutions over $1/\epsilon$ phases (iterations of the outer for loop). During a single phase, the algorithm uses a descending thresholds approach.

Algorithm 1 Algorithm for $\max_{\vec{x}\in[0,1]^n:\ \|\vec{x}\|_1\le k} f(\vec{x})$, where f is a non-negative DR-submodular function. The algorithm takes as input a target value M such that $f(\vec{x}^*) \le M \le (1+\epsilon)f(\vec{x}^*)$.

1: $\vec{x} \leftarrow \vec{0}$
2: $\vec{z} \leftarrow \vec{0}$
3: **for** $j = 1$ to $1/\epsilon$ **do**
4: $\langle\langle$ Start of phase j $\rangle\rangle$
5: $\vec{x}_{\text{start}} \leftarrow \vec{x}$
6: $\vec{z}_{\text{start}} \leftarrow \vec{z}$
7: $v_{\text{start}} \leftarrow \frac{1}{k}(((1-\epsilon)^j - 2\epsilon)M - f(\vec{x}))$
8: $v \leftarrow v_{\text{start}}$
9: **while** $v > \epsilon v_{\text{start}}$ and $\|\vec{z}\|_1 < \epsilon jk$ **do**
10: $\vec{g} = (\vec{1} - \vec{z}) \circ \nabla f(\vec{z})$
11: $S = \{i \in [n]\colon \vec{g}_i \ge v$ and $\vec{z}_i \le 1 - (1-\epsilon)^j$ and $\vec{z}_i - (\vec{z}_{\text{start}})_i < \epsilon(1 - (\vec{z}_{\text{start}})_i)\}$
12: **if** $S = \emptyset$ **then**
13: $v \leftarrow (1-\epsilon)v$
14: **else**
15: For a given $\eta \in [0, \epsilon^2]$, we define:

$$\vec{z}(\eta) = \vec{z} + \eta(\vec{1} - \vec{z}) \circ \vec{1}_S$$
$$\vec{g}(\eta) = (\vec{1} - \vec{z}(\eta)) \circ \nabla f(\vec{z}(\eta))$$
$$S(\eta) = \{i \in S\colon \vec{g}(\eta)_i \ge v\}$$
$$T(\eta) = \{i \in S\colon \vec{g}(\eta)_i > 0\}$$

16: Let η_1 be the maximum $\eta \in [0, \epsilon^2]$ such that $|S(\eta)| \ge (1-\epsilon)|S|$
17: $\langle\langle\ \eta_2 = \min\left\{\epsilon^2, \frac{\epsilon jk - \|\vec{z}\|_1}{|S| - \|\vec{z} \circ \vec{1}_S\|_1}\right\}\ \rangle\rangle$
18: Let η_2 be the maximum $\eta \in [0, \epsilon^2]$ such that $\|\vec{z}(\eta)\|_1 \le \epsilon jk$
19: $\eta \leftarrow \min\{\eta_1, \eta_2\}$
20: $\vec{x} \leftarrow \vec{x} + \eta(\vec{1} - \vec{x}) \circ \vec{1}_{T(\eta)}$
21: $\vec{z} \leftarrow \vec{z} + \eta(\vec{1} - \vec{z}) \circ \vec{1}_S$
22: **if** $f(\vec{z}) > f(\vec{x})$ **then**
23: $\vec{x} \leftarrow \vec{z}$
24: **end if**
25: **end if**
26: **end while**
27: **end for**
28: **return** $\vec{x}$

Starting with an initial threshold v_{start}, the algorithm only updates coordinates whose gradient is above the threshold. The algorithm only updates a carefully chosen subset of these high-gain coordinates: the set S on line 11. The definition of S imposes two further restrictions on the fractional value of each coordinate in S. These restrictions allow us to control the ℓ_∞ norm of the solution and cope with the non-monotonicity of the objective function (cf. Lemma 2). If the set S is empty, the algorithm lowers the threshold by an $1-\epsilon$ factor on line 13. Otherwise, the algorithm simultaneously increases a large fraction of the coordinates in S using an appropriately chosen step size η. The step size is carefully chosen to ensure several competing desiderata. On one hand, we want to take large steps for fast convergence/high parallelism. On the other, we want to take small steps to ensure high function value gain, since the gradient approximates the function value only locally and we need to be conservative on how much we increase due to the non-monotonicity of the objective. We choose our step size so that, after the update, the size of S decreases by an $1-\epsilon$ factor or we reach the budget of ϵjk allocated to the phase or we reach our cap of ϵ^2 on the step size. Thus we converge fast by either shrinking S or filling up the budget, while simultaneously achieving high gains in function value.

4. Analysis of the Approximation Guarantee

In this section, we show that Algorithm 1 achieves a $\frac{1}{e} - O(\epsilon)$ approximation. Recall that we assume that η_1 is computed exactly on line 16. In Section B of the supplement, we show how to extend the algorithm and the analysis so that the algorithm efficiently computes a suitable approximation to η_1 that suffices for obtaining a $\frac{1}{e} - O(\epsilon)$ approximation. The omitted proofs can be found in the supplement (Section A).

In the following, we refer to each iteration of the outer for loop as a *phase*. We refer to each iteration of the inner while loop as an *iteration*. Note that the update vectors are non-negative in each iteration of the algorithm, and thus the vectors $\vec{x}, \vec{z}$ remain non-negative throughout the algorithm and they can only increase. Additionally, since $S(\eta) \subseteq T(\eta) \subseteq S$, we have $\vec{x} \le \vec{z}$ throughout the algorithm. We will also use the following observations repeatedly, whose straightforward proofs are deferred to Section A of the supplement. By DR-submodularity, since the relevant vectors can only increase in each coordinate, the relevant gradients can only decrease in each coordinate. This implies that, for every $\eta \le \eta'$, we have $S(\eta) \supseteq S(\eta')$. Additionally, for every $i \in T(\eta)$, we have $\nabla_i f(\vec{x}) \ge \nabla_i f(\vec{z}) \ge \nabla_i f(\vec{z}(\eta)) > 0$.

We will need an upper bound on the ℓ_1 and ℓ_∞ norms of $\vec{x}$ and $\vec{z}$. Since $\vec{x} \le \vec{z}$, it suffices to upper bound the norms of $\vec{z}$ (the ℓ_1 norm bound will be used to show that the final solution is feasible, and the ℓ_∞ norm bound will be used to

derive the approximation guarantee). The omitted proofs can be found in the supplement.

Lemma 3. *Consider phase j of the algorithm (the j-th iteration of the outer for loop). Throughout the phase, the algorithm maintains the invariant that $\|\vec{z}\|_\infty \leq 1 - (1-\epsilon)^j + \epsilon^2$ and $\|\vec{z}\|_1 \leq \epsilon jk$.*

The following theorem is the heart of the analysis and it shows that each phase makes sufficient gain in function value. This is reminiscent of the gain made by the measured continuous greedy algorithm in the sequential setting and it implies the claimed approximation guarantee via induction. As discussed in Section 3, in order to achieve high parallelism, our algorithm proceeds very differently from the sequential continuous greedy algorithm. As a result, we need a very careful analysis to show the claimed gain.

Theorem 4. *Consider phase j of the algorithm (the j-th iteration of the outer for loop). Let $\vec{x}_{\text{start}}$ and $\vec{x}_{\text{end}}$ be the vector $\vec{x}$ at the beginning and end of the phase. We have*

$$f(\vec{x}_{\text{end}}) - f(\vec{x}_{\text{start}}) \geq (1-5\epsilon)\epsilon((1-\epsilon)^j f(\vec{x}^*) - f(\vec{x}_{\text{end}}) - 3\epsilon f(\vec{x}^*))$$

Proof. Recall that, within a single phase, the algorithm uses a descending thresholds approach and updates a carefully selected subset of the coordinates whose gradient/marginal gain is above the current threshold (line 11). The subset is chosen so that no coordinate increases too much. This allows us to control the ℓ_∞ norm of the solution throughout the phase, which in turn allows us to deal with the non-monotonicity of the function. The main technical challenge is to show that we gain enough throughout the phase despite that: (1) we are restricting the set of high-gain coordinates that are updated to ensure that no coordinate increases too much, and (2) we terminate phase j when we fill up ϵjk budget to ensure that we get a feasible solution after $1/\epsilon$ phases. We divide the analysis into two cases, depending on whether the threshold is ever updated on line 13.

Case 1: we have $v_{\text{end}} = v_{\text{start}}$. Note that the phase terminates with $\|\vec{z}_{\text{end}}\|_1 = \epsilon jk$ in this case.

We start with an overview of the analysis. All of the coordinates that were updated throughout the phase had gradient at least v_{start}, and v_{start} is chosen so that the gain is high. The step size η is chosen so that we update an $1-\epsilon$ fraction of the coordinates in S. Thus, ignoring $1-\epsilon$ factors, an iteration of the phase increases the function value by v_{start} times the increase in the ℓ_1-norm of the solution. Since the phase fills up at least ϵk of the budget, the increase in value over all iterations is at least $\epsilon k v_{\text{start}}$. Finally, the choice of v_{start} ensures that we reached the desired gain.

We now give the formal argument. We fix an iteration of the phase that updates $\vec{x}$ and $\vec{z}$ on lines 20–23, and analyze the gain in function value in the current iteration. We let $\vec{x}, \vec{z}$ denote the vectors right before the update on lines 20–23. Let $\vec{x}'$ be the vector $\vec{x}$ right after the update on line 20, and let $\vec{z}'$ be the vector $\vec{z}$ right after the update on line 21.

We have:

$$\begin{aligned} f(\vec{x}') - f(\vec{x}) &\overset{(a)}{\geq} \langle \nabla f(\vec{x}'), \eta(\vec{1} - \vec{x}) \circ \vec{1}_{T(\eta)} \rangle \\ &= \langle (\vec{1} - \vec{x}) \circ \nabla f(\vec{x}'), \eta \vec{1}_{T(\eta)} \rangle \\ &\overset{(b)}{\geq} \langle \vec{g}(\eta), \eta \vec{1}_{T(\eta)} \rangle \\ &\overset{(c)}{\geq} \eta v_{\text{start}} \underbrace{|S(\eta)|}_{\geq (1-\epsilon)|S|} \\ &\overset{(d)}{\geq} (1-\epsilon)\eta v_{\text{start}} |S| \end{aligned}$$

In (a), we used the fact that $\vec{x}' - \vec{x} \geq 0$ and f is concave in non-negative directions.

We can show (b) as follows. We have $\vec{x}' \leq \vec{z}' = \vec{z}(\eta)$ and thus $\nabla f(\vec{x}') \geq \nabla f(\vec{z}(\eta))$ by DR-submodularity. Additionally, for every coordinate $i \in T(\eta)$, we have $\nabla_i f(\vec{z}(\eta)) > 0$. Therefore, for every $i \in T(\eta)$, we have $(1 - \vec{x}_i)\nabla_i f(\vec{x}') \geq (1 - \vec{z}(\eta)_i)\nabla_i f(\vec{z}(\eta)) = \vec{g}(\eta)_i > 0$.

In (c), we have used that $S(\eta) \subseteq T(\eta)$, $\vec{g}(\eta)_i > 0$ for all $i \in T(\eta)$, and $\vec{g}(\eta)_i \geq v = v_{\text{start}}$ for all $i \in S(\eta)$.

We can show (d) as follows. Since $\eta \leq \eta_1$, we have $|S(\eta)| \geq |S(\eta_1)| \geq (1-\epsilon)|S|$, where the second inequality is by the choice of η_1.

Let η_t and S_t denote η and S in iteration t of the phase (note that we are momentarily overloading η_1 and η_2 here, and they temporarily stand for the step size η in iterations 1 and 2, and not for the step sizes on lines 16 and 18). By summing up the above inequality over all iterations, we obtain:

$$\begin{aligned} & f(\vec{x}_{\text{end}}) - f(\vec{x}_{\text{start}}) \\ &\geq (1-\epsilon) v_{\text{start}} \sum_t \eta_t |S_t| \\ &\geq (1-\epsilon) v_{\text{start}} \underbrace{\|\vec{z}_{\text{end}} - \vec{z}_{\text{start}}\|_1}_{\geq \epsilon k} \\ &\overset{(a)}{\geq} (1-\epsilon) v_{\text{start}} \epsilon k \\ &\overset{(b)}{=} \epsilon(1-\epsilon)(((1-\epsilon)^j - 2\epsilon)M - f(\vec{x}_{\text{start}})) \\ &\overset{(c)}{\geq} \epsilon(1-\epsilon)((1-\epsilon)^j f(\vec{x}^*) - f(\vec{x}_{\text{start}}) - 3\epsilon M) \end{aligned}$$

We can show (a) as follows. Recall that we have $\|\vec{z}_{\text{end}}\|_1 = \epsilon jk$. Since $\|\vec{z}_{\text{start}}\|_1 \leq \epsilon(j-1)k$, we have $\|\vec{z}_{\text{end}} - \vec{z}_{\text{start}}\|_1 \geq \epsilon k$. In (b), we used the definition of v_{start} on line 7. In (c), we used that $f(\vec{x}^*) \leq (1+\epsilon)M$.

Case 2: we have $v_{\text{end}} \neq v_{\text{start}}$. Note that this implies that $v_{\text{end}} \leq (1-\epsilon)v_{\text{start}}$, since line 13 was executed at least once during the phase.

The analysis for this case is more involved. The difficulty stems from the fact that the algorithm updates only a subset of the high-gain coordinates to ensure that no coordinate increases too much. This means that, at the end of the phase, there could still be coordinates whose gain is above the threshold but we cannot increase them further during the current phase. We now formally define this problematic set of coordinates:

$$A := \left\{ i \in [n] \colon (1 - (\vec{z}_{\text{end}})_i) \nabla_i f(\vec{z}_{\text{end}}) \geq \frac{v_{\text{end}}}{1-\epsilon} \right\}$$

The coordinates in $\overline{A}$ are allowed to increase, and thus we can account for their gain similarly to the previous case: their contribution to the overall gain is at least v_{end} times the increase in ℓ_1-norm of the coordinates in $\overline{A}$. For the coordinates in A, we lower bound their contribution to the gain by the gradient times the increase in those coordinates. The following lemma precisely states the lower bound that we obtain. Note that, for the coordinates in A, we use the gradient value and not v_{end}, to get a tighter bound.

Lemma 5. *We have*

$$\begin{aligned}
&f(\vec{x}_{\text{end}}) - f(\vec{x}_{\text{start}}) \\
&\geq (1-\epsilon)\Big(v_{\text{end}} \|(\vec{z}_{\text{end}} - \vec{z}_{\text{start}}) \circ \vec{1}_{\overline{A}}\|_1 \\
&\quad + \langle \nabla f(\vec{z}_{\text{end}}), (\vec{z}_{\text{end}} - \vec{z}_{\text{start}}) \circ \vec{1}_A \rangle \Big)
\end{aligned}$$

Proof. Fix an iteration of the phase that updates $\vec{x}$ and $\vec{z}$ on lines 20–23. Let $\vec{x}, \vec{z}$ denote the vectors right before the update on lines 20–23. Let $\vec{x}'$ be the vector $\vec{x}$ right after the update on line 20, and let $\vec{z}'$ be the vector $\vec{z}$ right after the update on line 21.

We have:

$$\begin{aligned}
&f(\vec{x}') - f(\vec{x}) \\
&\overset{(a)}{\geq} \langle \nabla f(\vec{x}'), \eta(\vec{1} - \vec{x}) \circ \vec{1}_{T(\eta)} \rangle \\
&= \langle (\vec{1} - \vec{x}) \circ \nabla f(\vec{x}'), \eta \vec{1}_{T(\eta)} \rangle \\
&\overset{(b)}{\geq} \langle \vec{g}(\eta), \eta \vec{1}_{T(\eta)} \rangle \\
&= \langle \vec{g}(\eta), \eta \vec{1}_{S(\eta)} \rangle + \langle \vec{g}(\eta), \eta \vec{1}_{T(\eta) \setminus S(\eta)} \rangle \\
&= \eta v_{\text{end}} \underbrace{|S(\eta)|}_{\geq (1-\epsilon)|S|} + \underbrace{\langle \vec{g}(\eta) - v_{\text{end}} \vec{1}, \eta \vec{1}_{S(\eta)} \rangle}_{\geq 0} \\
&\quad + \underbrace{\langle \vec{g}(\eta), \eta \vec{1}_{T(\eta) \setminus S(\eta)} \rangle}_{\geq 0} \\
&\overset{(c)}{\geq} (1-\epsilon)\eta \Big(v_{\text{end}} |S| + \langle \vec{g}(\eta) - v_{\text{end}} \vec{1}, \vec{1}_{S(\eta)} \rangle \\
&\quad + \langle \vec{g}(\eta), \vec{1}_{T(\eta) \setminus S(\eta)} \rangle \Big) \\
&= (1-\epsilon)\eta \Big(v_{\text{end}} |S| - v_{\text{end}} |S(\eta)| + \langle \vec{g}(\eta), \vec{1}_{T(\eta)} \rangle \Big) \\
&= (1-\epsilon)\eta \Big(v_{\text{end}} |S| - v_{\text{end}} |S(\eta)| + \langle \vec{g}(\eta), \vec{1}_{T(\eta) \setminus A} \rangle \\
&\quad + \langle \vec{g}(\eta), \vec{1}_{T(\eta) \cap A} \rangle \Big) \\
&\overset{(d)}{=} (1-\epsilon)\eta \Big(v_{\text{end}} |S| - v_{\text{end}} |S(\eta)| + \langle \vec{g}(\eta), \vec{1}_{T(\eta) \setminus A} \rangle \\
&\quad + \langle \vec{g}(\eta), \vec{1}_{S \cap A} \rangle \Big) \\
&\overset{(e)}{\geq} (1-\epsilon)\eta \Big(v_{\text{end}} |S| - v_{\text{end}} |S(\eta)| + \langle \vec{g}(\eta), \vec{1}_{S(\eta) \setminus A} \rangle \\
&\quad + \langle \vec{g}(\eta), \vec{1}_{S \cap A} \rangle \Big) \\
&\overset{(f)}{\geq} (1-\epsilon)\eta \Big(v_{\text{end}} |S| - v_{\text{end}} |S(\eta)| + v_{\text{end}} |S(\eta) \setminus A| \\
&\quad + \langle \vec{g}(\eta), \vec{1}_{S \cap A} \rangle \Big) \\
&= (1-\epsilon)\eta \Big(v_{\text{end}} (|S| - |S(\eta) \cap A|) + \langle \vec{g}(\eta), \vec{1}_{S \cap A} \rangle \Big) \\
&\geq (1-\epsilon)\eta \Big(v_{\text{end}} |S \setminus A| + \langle \vec{g}(\eta), \vec{1}_{S \cap A} \rangle \Big) \\
&= (1-\epsilon)\eta \Big(v_{\text{end}} |S \setminus A| \\
&\quad + \langle \nabla f(\vec{z}(\eta)), (\vec{1} - \vec{z}(\eta)) \circ \vec{1}_{S \cap A} \rangle \Big) \\
&\overset{(g)}{\geq} (1-\epsilon)\eta \Big(v_{\text{end}} |S \setminus A| \\
&\quad + \langle \nabla f(\vec{z}_{\text{end}}), (\vec{1} - \vec{z}(\eta)) \circ \vec{1}_{S \cap A} \rangle \Big)
\end{aligned}$$

In (a), we used the fact that $\vec{x}' - \vec{x} \geq 0$ and f is concave in non-negative directions.

We can show (b) as follows. We have $\vec{x}' \leq \vec{z}' = \vec{z}(\eta)$ and thus $\nabla f(\vec{x}') \geq \nabla f(\vec{z}(\eta))$ by DR-submodularity. Additionally, for every coordinate $i \in T(\eta)$, we have $\nabla_i f(\vec{z}(\eta)) > 0$. Therefore, for every $i \in T(\eta)$, we have $(1 - \vec{x}_i)\nabla_i f(\vec{x}') \geq (1 - \vec{z}(\eta)_i)\nabla_i f(\vec{z}(\eta)) = \vec{g}(\eta)_i > 0$.

We can show (c) as follows. Since $\eta \leq \eta_1$, we have $|S(\eta)| \geq |S(\eta_1)| \geq (1-\epsilon)|S|$, where the second inequality is by the choice of η_1. By the definition of $S(\eta)$, we have $\vec{g}(\eta)_i \geq v \geq v_{\text{end}}$ for every $i \in S(\eta)$. By the definition of $T(\eta)$, we have $\vec{g}(\eta)_i > 0$ for every $i \in T(\eta)$.

Equality (d) follows from the fact that $S \cap A = T(\eta) \cap A$, which we can show as follows. We have $T(\eta) \subseteq S$, and $S \setminus T(\eta)$ is the set of all coordinates with negative gradient $\vec{g}(\eta)$. For every $i \in A$, we have $\nabla_i f(\vec{z}(\eta)) \geq \nabla_i f(\vec{z}_{\text{end}}) > 0$, where the first inequality is by DR-submodularity (since $\vec{z}(\eta) \leq \vec{z}_{\text{end}}$) and the second inequality is by the definition of A and the fact that $(\vec{z}_{\text{end}})_i < 1$ for all $i \in [n]$ (Lemma 3). Moreover, we have $\vec{z}(\eta)_i < 1$ for all $i \in [n]$ (Lemma 3). Thus $\vec{g}(\eta)_i > 0$ for all $i \in A$, and hence $S \cap A = T(\eta) \cap A$.

In (e), we have used that $S(\eta) \subseteq T(\eta)$ and $\vec{g}(\eta)$ is non-negative on the coordinates of $T(\eta)$. In (f), we have used that $\vec{g}(\eta)_i \geq v \geq v_{\text{end}}$ for all $i \in S(\eta)$. In (g), we have used that $\vec{z}(\eta) \leq \vec{z}_{\text{end}}$ and thus $\nabla f(\vec{z}(\eta)) \geq \nabla f(\vec{z}_{\text{end}})$ by DR-submodularity.

Let $\eta_t, S_t, T_t(\eta), \vec{z}_t(\eta), \vec{g}_t(\eta)$ denote $\eta, S, T(\eta), \vec{z}(\eta), \vec{g}(\eta)$ in iteration t of the phase (note that we are overloading η_1 and η_2 in this proof only, and they temporarily stand for the step size η in iterations 1 and 2, and not for the step sizes on lines 16 and 18). By summing up the above inequality over all iterations, we obtain:

$$\begin{aligned}
&f(\vec{x}_{\text{end}}) - f(\vec{x}_{\text{start}}) \\
&\geq (1-\epsilon)\Big(\sum_t v_{\text{end}}\eta_t |S_t \setminus A| \\
&+ \sum_t \langle \nabla f(\vec{z}_{\text{end}}), \eta_t(\vec{1} - \vec{z}(\eta)) \circ \vec{1}_{S_t \cap A}\rangle\Big) \\
&\geq (1-\epsilon)\Big(v_{\text{end}}\|(\vec{z}_{\text{end}} - \vec{z}_{\text{start}}) \circ \vec{1}_{\overline{A}}\|_1 \\
&+ \langle \nabla f(\vec{z}_{\text{end}}), (\vec{z}_{\text{end}} - \vec{z}_{\text{start}}) \circ \vec{1}_A\rangle\Big)
\end{aligned}$$

□

With the lower bound on the gain given by Lemma 5 in hand, the remaining work is to relate it to the optimal solution. Before doing so, we show that each coordinate in A increases sufficiently. Recall that the coordinates in A were prevented from increasing further by the two conditions on $\vec{z}_i$ in the definition of S: $\vec{z}_i \leq 1 - (1-\epsilon)^j$ and $\vec{z}_i - (\vec{z}_{\text{start}})_i < \epsilon(1 - (\vec{z}_{\text{start}})_i)$. In the latter case, the lemma follows immediately. In the former case, we use that each iteration increases the coordinates by at most ϵ^2.

Lemma 6. *For every $i \in A$, we have $(\vec{z}_{\text{end}})_i - (\vec{z}_{\text{start}})_i \geq (1-3\epsilon)\epsilon(1 - (\vec{z}_{\text{end}})_i)$.*

Proof of Lemma 6. Since S was empty at the previous threshold $v_{\text{end}}/(1-\epsilon)$, we have $(\vec{z}_{\text{end}})_i \geq 1 - (1-\epsilon)^j$ or $(\vec{z}_{\text{end}})_i - (\vec{z}_{\text{start}})_i \geq \epsilon(1 - (\vec{z}_{\text{start}})_i)$. If it is the latter, the claim follows, since $1 - (\vec{z}_{\text{start}})_i \geq 1 - (\vec{z}_{\text{end}})_i$. Therefore we may assume it is the former. By Lemma 3, $(\vec{z}_{\text{start}})_i \leq 1 - (1-\epsilon)^{j-1} + \epsilon^2$. Therefore

$$\begin{aligned}
(\vec{z}_{\text{end}})_i - (\vec{z}_{\text{start}})_i &\geq \epsilon(1-\epsilon)^{j-1} - \epsilon^2 \\
&\geq (1-3\epsilon)\epsilon(1-\epsilon)^{j-1} \\
&\geq (1-3\epsilon)\epsilon(1 - (\vec{z}_{\text{end}})_i)
\end{aligned}$$

where in the second inequality we used that $(1-\epsilon)^{j-1} \geq 1/3$ for sufficiently small ϵ (since $(1-\epsilon)^{j-1} \geq (1-\epsilon)^{1/\epsilon} \approx 1/e$). □

Recall that the goal of the remainder of the analysis is to relate the lower bound on the gain given in Lemma 5 to the optimal solution $\vec{x}^*$. To this end, we use the inner product between the gradient at the end of the phase and the optimal solution as our potential. To complete the analysis, we sandwich this potential between the gain guaranteed by Lemma 5 and our target gain. We show that the potential is at least our target gain in Lemma 7. In Lemmas 8 and 9, we relate the gain from Lemma 5 to the potential.

As noted above, the following lemma shows that the potential is lower bounded by our target gain. The argument is similar to the analysis of the measured continuous greedy. In this part of the analysis, we use the fact that we carefully controlled the ℓ_∞ norm of the solution, which is crucial when working with non-monotone objectives.

Lemma 7. *We have $\langle \nabla f(\vec{z}_{\text{end}}) \vee \vec{0}, (\vec{1} - \vec{z}_{\text{end}}) \circ \vec{x}^* \rangle \geq ((1-\epsilon)^j - \epsilon^2) f(\vec{x}^*) - f(\vec{x}_{\text{end}})$.*

Proof. We have

$$\begin{aligned}
&\langle \nabla f(\vec{z}_{\text{end}}) \vee \vec{0}, (\vec{1} - \vec{z}_{\text{end}}) \circ \vec{x}^* \rangle \\
&\overset{(a)}{\geq} \langle \nabla f(\vec{z}_{\text{end}}) \vee \vec{0}, \vec{x}^* \vee \vec{z}_{\text{end}} - \vec{z}_{\text{end}} \rangle \\
&\overset{(b)}{\geq} f(\vec{z}_{\text{end}} \vee \vec{x}^*) - f(\vec{z}_{\text{end}}) \\
&\overset{(c)}{\geq} f(\vec{z}_{\text{end}} \vee \vec{x}^*) - f(\vec{x}_{\text{end}}) \\
&\overset{(d)}{\geq} (1 - \|\vec{z}_{\text{end}}\|_\infty) f(\vec{x}^*) - f(\vec{x}_{\text{end}}) \\
&\overset{(e)}{\geq} ((1-\epsilon)^j - \epsilon^2) f(\vec{x}^*) - f(\vec{x}_{\text{end}})
\end{aligned}$$

In (a), we used that $(1-a)b \geq \max\{a,b\} - a$ for all $a, b \in [0,1]$.

In (b), we used the fact that f is concave in non-negative directions.

In (c), we used the fact that the algorithm maintains the invariant that $f(\vec{x}) \geq f(\vec{z})$ via the update on line 23.

In (d), we used Lemma 2.

In (e), we used Lemma 3. □

The next two lemmas relate the gain guaranteed by Lemma 5 to our potential and use Lemma 7 to complete the analysis. Recall that the phase terminates with either $v_{\text{end}} \leq \epsilon v_{\text{start}}$ or $\|\vec{z}_{\text{end}}\|_1 = \epsilon j k$. We consider each of these cases in turn.

Lemma 8. *Suppose that $v_{\text{end}} \leq \epsilon v_{\text{start}}$. We have:*

$$\begin{aligned}
&f(\vec{x}_{\text{end}}) - f(\vec{x}_{\text{start}}) \\
&\geq (1-5\epsilon)\epsilon((1-\epsilon)^j f(\vec{x}^*) - f(\vec{x}_{\text{end}}) - 2\epsilon f(\vec{x}^*))
\end{aligned}$$

Proof. By Lemma 5, we have:

$$f(\vec{x}_{\text{end}}) - f(\vec{x}_{\text{start}})$$

$$
\begin{aligned}
&\geq (1-\epsilon)\langle \nabla f(\vec{z}_{\text{end}}), (\vec{z}_{\text{end}} - \vec{z}_{\text{start}}) \circ \vec{1}_A \rangle \\
&= (1-\epsilon)\Big(\langle \nabla f(\vec{z}_{\text{end}}), \epsilon(1-3\epsilon)(\vec{1} - \vec{z}_{\text{end}}) \circ \vec{x}^* \circ \vec{1}_A \rangle \\
&+ \langle \nabla f(\vec{z}_{\text{end}}), (\vec{z}_{\text{end}} - \vec{z}_{\text{start}}) \circ \vec{1}_A \rangle \\
&- \langle \nabla f(\vec{z}_{\text{end}}), \epsilon(1-3\epsilon)(\vec{1} - \vec{z}_{\text{end}}) \circ \vec{x}^* \circ \vec{1}_A \rangle \Big) \\
&\overset{(a)}{\geq} (1-\epsilon)\langle \nabla f(\vec{z}_{\text{end}}), \epsilon(1-3\epsilon)(\vec{1} - \vec{z}_{\text{end}}) \circ \vec{x}^* \circ \vec{1}_A \rangle \\
&\geq (1-4\epsilon)\epsilon \langle \nabla f(\vec{z}_{\text{end}}), (\vec{1} - \vec{z}_{\text{end}}) \circ \vec{x}^* \circ \vec{1}_A \rangle \\
&= (1-4\epsilon)\epsilon \langle (\vec{1} - \vec{z}_{\text{end}}) \circ \nabla f(\vec{z}_{\text{end}}), \vec{x}^* \circ \vec{1}_A \rangle \\
&= (1-4\epsilon)\epsilon \Big(\langle (\vec{1} - \vec{z}_{\text{end}}) \circ \nabla f(\vec{z}_{\text{end}}) \vee \vec{0}, \vec{x}^* \rangle \\
&- \langle (\vec{1} - \vec{z}_{\text{end}}) \circ \nabla f(\vec{z}_{\text{end}}) \vee \vec{0}, \vec{x}^* \circ \vec{1}_{\overline{A}} \rangle \Big) \\
&\overset{(b)}{\geq} (1-4\epsilon)\epsilon \Big(\langle (\vec{1} - \vec{z}_{\text{end}}) \circ \nabla f(\vec{z}_{\text{end}}) \vee \vec{0}, \vec{x}^* \rangle - \frac{\epsilon v_{\text{start}}}{1-\epsilon} k \Big) \\
&\geq (1-5\epsilon)\epsilon \Big(\langle (\vec{1} - \vec{z}_{\text{end}}) \circ \nabla f(\vec{z}_{\text{end}}) \vee \vec{0}, \vec{x}^* \rangle - \epsilon v_{\text{start}} k \Big) \\
&\overset{(c)}{\geq} (1-5\epsilon)\epsilon (((1-\epsilon)^j - \epsilon^2) f(\vec{x}^*) - f(\vec{x}_{\text{end}}) - \epsilon v_{\text{start}} k) \\
&\overset{(d)}{\geq} (1-5\epsilon)\epsilon (((1-\epsilon)^j - \epsilon^2) f(\vec{x}^*) - f(\vec{x}_{\text{end}}) - \epsilon f(\vec{x}^*)) \\
&\geq (1-5\epsilon)\epsilon ((1-\epsilon)^j f(\vec{x}^*) - f(\vec{x}_{\text{end}}) - 2\epsilon f(\vec{x}^*))
\end{aligned}
$$

In (a), we used Lemma 6 and the fact that $\vec{0} \leq \vec{x}^* \leq \vec{1}$ and the fact that $\nabla_i f(\vec{z}_{\text{end}}) > 0$ for all $i \in A$.

In (b), we used that $\|\vec{x}^*\|_1 \leq k$ and the definition of A.

In (c), we used Lemma 7.

Inequality (d) follows from the definition of v_{start} and the fact that $f(\vec{x}^*) \geq M$. □

The analysis of the second case uses a similar but more involved argument.

Lemma 9. *Suppose that $\|\vec{z}_{\text{end}}\|_1 = \epsilon j k$. We have:*

$$
\begin{aligned}
&f(\vec{x}_{\text{end}}) - f(\vec{x}_{\text{start}}) \\
&\geq \epsilon(1-4\epsilon)(((1-\epsilon)^j - \epsilon^2) f(\vec{x}^*) - f(\vec{x}_{\text{end}}))
\end{aligned}
$$

Proof. By Lemma 5, we have:

$$
\begin{aligned}
&f(\vec{x}_{\text{end}}) - f(\vec{x}_{\text{start}}) \\
&\geq (1-\epsilon)\Big(v_{\text{end}} \|(\vec{z}_{\text{end}} - \vec{z}_{\text{start}}) \circ \vec{1}_{\overline{A}}\|_1 \\
&+ \langle \nabla f(\vec{z}_{\text{end}}), (\vec{z}_{\text{end}} - \vec{z}_{\text{start}}) \circ \vec{1}_A \rangle \Big) \\
&= (1-\epsilon)\Big(v_{\text{end}} \|(\vec{z}_{\text{end}} - \vec{z}_{\text{start}}) \circ \vec{1}_{\overline{A}}\|_1 \\
&+ \langle \nabla f(\vec{z}_{\text{end}}), (\vec{z}_{\text{end}} - \vec{z}_{\text{start}}) \circ \vec{1}_A \rangle \\
&- \langle \nabla f(\vec{z}_{\text{end}}), (1-3\epsilon)\epsilon(\vec{1} - \vec{z}_{\text{end}}) \circ \vec{x}^* \circ \vec{1}_A \rangle \\
&+ \langle \nabla f(\vec{z}_{\text{end}}), (1-3\epsilon)\epsilon(\vec{1} - \vec{z}_{\text{end}}) \circ \vec{x}^* \circ \vec{1}_A \rangle \Big) \\
&\overset{(a)}{\geq} (1-\epsilon)\Big(v_{\text{end}} \|(\vec{z}_{\text{end}} - \vec{z}_{\text{start}}) \circ \vec{1}_{\overline{A}}\|_1 \\
&+ v_{\text{end}} \|(\vec{z}_{\text{end}} - \vec{z}_{\text{start}} - (1-3\epsilon)\epsilon(\vec{1} - \vec{z}_{\text{end}}) \circ \vec{x}^*) \circ \vec{1}_A\|_1 \\
&+ (1-3\epsilon)\epsilon \langle \nabla f(\vec{z}_{\text{end}}), (\vec{1} - \vec{z}_{\text{end}}) \circ \vec{x}^* \circ \vec{1}_A \rangle \Big) \\
&= (1-\epsilon)\Big(v_{\text{end}} \underbrace{\|\vec{z}_{\text{end}} - \vec{z}_{\text{start}}\|}_{\geq \epsilon k \geq \epsilon \|\vec{x}^*\|_1} \\
&- v_{\text{end}} \underbrace{\|(1-3\epsilon)\epsilon(\vec{1} - \vec{z}_{\text{end}}) \circ \vec{x}^* \circ \vec{1}_A\|_1}_{\leq \epsilon \|\vec{x}^* \circ \vec{1}_A\|_1} \\
&+ (1-3\epsilon)\epsilon \langle \nabla f(\vec{z}_{\text{end}}), (\vec{1} - \vec{z}_{\text{end}}) \circ \vec{x}^* \circ \vec{1}_A \rangle \Big) \\
&\overset{(b)}{\geq} (1-\epsilon)\Big(v_{\text{end}} \epsilon \|\vec{x}^* \circ \vec{1}_{\overline{A}}\|_1 \\
&+ (1-3\epsilon)\epsilon \langle \nabla f(\vec{z}_{\text{end}}), (\vec{1} - \vec{z}_{\text{end}}) \circ \vec{x}^* \circ \vec{1}_A \rangle \Big) \\
&\overset{(c)}{\geq} (1-\epsilon)\Big((1-\epsilon)\epsilon \langle \nabla f(\vec{z}_{\text{end}}) \vee \vec{0}, (\vec{1} - \vec{z}_{\text{end}}) \circ \vec{x}^* \circ \vec{1}_{\overline{A}} \rangle \\
&+ (1-3\epsilon)\epsilon \langle \nabla f(\vec{z}_{\text{end}}), (\vec{1} - \vec{z}_{\text{end}}) \circ \vec{x}^* \circ \vec{1}_A \rangle \Big) \\
&\geq (1-4\epsilon)\epsilon \langle \nabla f(\vec{z}_{\text{end}}) \vee \vec{0}, (\vec{1} - \vec{z}_{\text{end}}) \circ \vec{x}^* \rangle \\
&\overset{(d)}{\geq} (1-4\epsilon)\epsilon (((1-\epsilon)^j - \epsilon^2) f(\vec{x}^*) - f(\vec{x}_{\text{end}}))
\end{aligned}
$$

In (a), we used Lemma 6 and the fact that $\vec{0} \leq \vec{x}^* \leq \vec{1}$ and the fact that $\nabla_i f(\vec{z}_{\text{end}}) \geq v_{\text{end}}$ for all $i \in A$.

We can show (b) as follows. Recall that we are in the case $\|\vec{z}_{\text{end}}\|_1 = \epsilon j k$. Since $\|\vec{z}_{\text{start}}\|_1 \leq \epsilon(j-1)k$, we have $\|\vec{z}_{\text{end}} - \vec{z}_{\text{start}}\|_1 \geq \epsilon k$. Additionally, $\|\vec{x}^*\|_1 \leq k$.

In (c), we used that, for all $i \notin A$, we have $v_{\text{end}} \geq (1-\epsilon)(1 - (\vec{z}_{\text{end}})_i)\nabla_i f(\vec{z}_{\text{end}})$.

In (d), we used Lemma 7. □

Thus in both cases we achieve the desired gain, and the proof is complete. □

Using induction and Theorem 4, we can show that the final solution returned by the algorithm is a $1/e - O(\epsilon)$ approximation. By construction, the final solution satisfies $\|\vec{x}\|_1 \leq k$, and thus it also satisfies the constraint.

Theorem 10. *Let $\vec{x}$ be the final solution returned by Algorithm 2. We have $\|\vec{x}\|_1 \leq k$ and $f(\vec{x}) \geq \left(\frac{1}{e} - O(\epsilon)\right) f(\vec{x}^*)$.*

5. Analysis of the Number of Iterations

Recall that we refer to each iteration of the outer for loop as a *phase*. We refer to each iteration of the inner while loop as an *iteration*.

Theorem 11. *The total number of iterations of the algorithm is $O(\log(n)\log(1/\epsilon)/\epsilon^3)$.*

Proof. There are $O(1/\epsilon)$ phases. In each phase, there are $O(\log(1/\epsilon)/\epsilon)$ different thresholds v: the initial threshold is v_{start}, the threshold right before the final one is at least $\epsilon v_{\text{start}}$, and each update on line 13 decreases the threshold by a $(1-\epsilon)$ factor. Thus it only remains to bound the number of iterations with the same threshold.

In the following, we fix a single threshold and we consider only the iterations of the phase at that threshold. Over these iterations, $\vec{z}$ is non-decreasing in every coordinate, $\vec{g}$ is non-increasing in every coordinate by DR-submodularity and $\vec{1} - \vec{z} \geq \vec{0}$, and the set S can only lose coordinates and thus $|S|$ is non-increasing. Additionally, for each coordinate $i \in [n]$, the increase $(\vec{z}_{\text{end}})_i - (\vec{z}_{\text{start}})_i$ over the entire phase is at most $\epsilon + \epsilon^2$: the increase in each iteration is η if $i \in S$ and 0 otherwise; since $\eta \leq \epsilon^2$ and $\vec{z}_i - (\vec{z}_{\text{start}})_i < \epsilon(1 - (\vec{z}_{\text{start}})_i)$ for every $i \in S$, the claim follows.

We say that an iteration is a *large-step iteration* if $\eta = \epsilon^2$ and it is a *smaller-step iteration* if $\eta < \epsilon^2$.

We first consider the large-step iterations. Let t be the last large-step iteration, and let S_t be the set S in that iteration. Let $i \in S_t$. Note that $i \in S_{t'}$ for all iterations $t' \leq t$, since $S_t \subseteq S_{t'}$. Thus every large-step iteration increases $\vec{z}_i$ by $\epsilon^2(1 - \vec{z}_i) \geq \epsilon^2(1-\epsilon)^j \geq \epsilon^2(1-\epsilon)^{1/\epsilon} = \Theta(\epsilon^2)$. Since $\vec{z}_i$ increases by at most $\epsilon + \epsilon^2$ over the entire phase, it follows that the number of large-step iterations is $O(1/\epsilon)$.

Next, we consider the smaller-step iterations. Note that, in every smaller-step iteration except possibly the last one, we have $|S(\eta)| \leq (1-\epsilon)|S|$ (if $\eta = \eta_2 < \epsilon^2$, at the end of the iteration we have $\|\vec{z}\|_1 = \epsilon jk$ and thus the phase ends; if $\eta = \eta_1$, our choice of η_1 ensures that $|S(\eta_1)| \leq (1-\epsilon)|S|$). Thus every smaller-step iteration decreases $|S|$ by at least an $(1-\epsilon)$ factor. Now note that $|S| \leq n$ in the first iteration, $|S| \geq 1$ in the last iteration, and $|S|$ can only decrease with each iteration. Thus the number of smaller-step iterations is $O(\log n/\epsilon)$.

In summary, there are $O(1/\epsilon)$ phases, $O(\log(1/\epsilon)/\epsilon)$ different thresholds per phase, and $O(\log(n)/\epsilon)$ iterations per threshold. Thus the total number of iterations of the algorithm is $O(\log(n)\log(1/\epsilon)/\epsilon^3)$. □

6. Experimental Results

We experimentally evaluate our parallel algorithm on instances of non-concave quadratic programming (NQP) and softmax extension of determinantal point processes (DPP) that were randomly generated similarly to previous work (Bian et al., 2017).

NQP instances are functions of the form $f(\vec{x}) = \frac{1}{2}\vec{x}^\top H\vec{x} + \vec{h}^\top\vec{x}$, where $H \in \mathrm{R}^{n\times n}$ is a matrix with non-positive entries, $\vec{h} \in \mathbb{R}^n$. We randomly generated such instances as follows: we sampled each entry of H uniformly at random from $[-10, 0]$, and we set $\vec{h} = -0.2H^\top\vec{1}$.

DPP instances are functions of the form $f(\vec{x}) = \log\det(\mathrm{diag}(\vec{x})(L - I) + I)$, where $L \in \mathbb{R}^{n\times n}$ is a psd matrix and I is the identity matrix. We randomly generated such instances as follows. We sampled the eigenvalues of L as follows: the i-th eigenvalue is $\ell_i = e^{r_i}$, where r_i was sampled uniformly from $[-0.5, 1]$. We sampled a random orthogonal matrix V. We set $L = V\mathrm{diag}(\ell_1, \ldots, \ell_n)V^\top$.

Algorithms, implementation details, and parameter choices. We empirically compared our parallel algorithm with the sequential continuous greedy algorithm (Chekuri et al., 2015; Bian et al., 2017) and the parallel multiplicative weights update algorithm (Ene et al., 2019) (see Section C in the supplement for pseudocode descriptions). We implemented the sequential continuous greedy algorithm using a step size of ϵ/n, leading to $O(n/\epsilon)$ iterations and adaptive evaluations. We implemented our algorithm with a more aggressive update of the thresholds on line 13: instead of the update $v \leftarrow (1-\epsilon)v$, we performed the update $v \leftarrow 0.75 \cdot v$. We used error $\epsilon = 0.05$ and budget $k = 10$ in all of the experiments.

Computing infrastructure. We implemented the algorithms in C++ and ran the experiments on an iMac with a 3.3 GHz Intel Core i5 processor and 8 GB of memory. The code used for generating the instances and evaluating the algorithms can be found in the supplement.

Results. The experimental results are shown in Figure 1. Each value is the average value for 5 independently sampled instances and the error bar is ± 1 standard deviation. The sequential continuous greedy algorithm achieved the highest solution value in all of the runs, and we report the value obtained by the parallel algorithm as the fraction of the continuous greedy solution value.

In all of the runs, our parallel algorithm achieves higher function value than the parallel multiplicative weights update algorithm, while the number of evaluations is significantly lower. The running time of the MWU algorithm is prohibitive on larger instances, and thus we were only able to compare the algorithms in the small n regime, which is advantageous for the MWU algorithm.

Acknowledgements

We thank the reviewers for their comments and suggestions for improving the presentation. We thank Adrian Vladu for helpful conversations about this work. The work of Alina Ene was supported in part by NSF CAREER grant CCF-1750333, NSF grant CCF-1718342, and NSF grant III-1908510. The work of Huy L. Nguyễn was supported in part by NSF CAREER grant CCF-1750716 and NSF grant CCF-1909314.

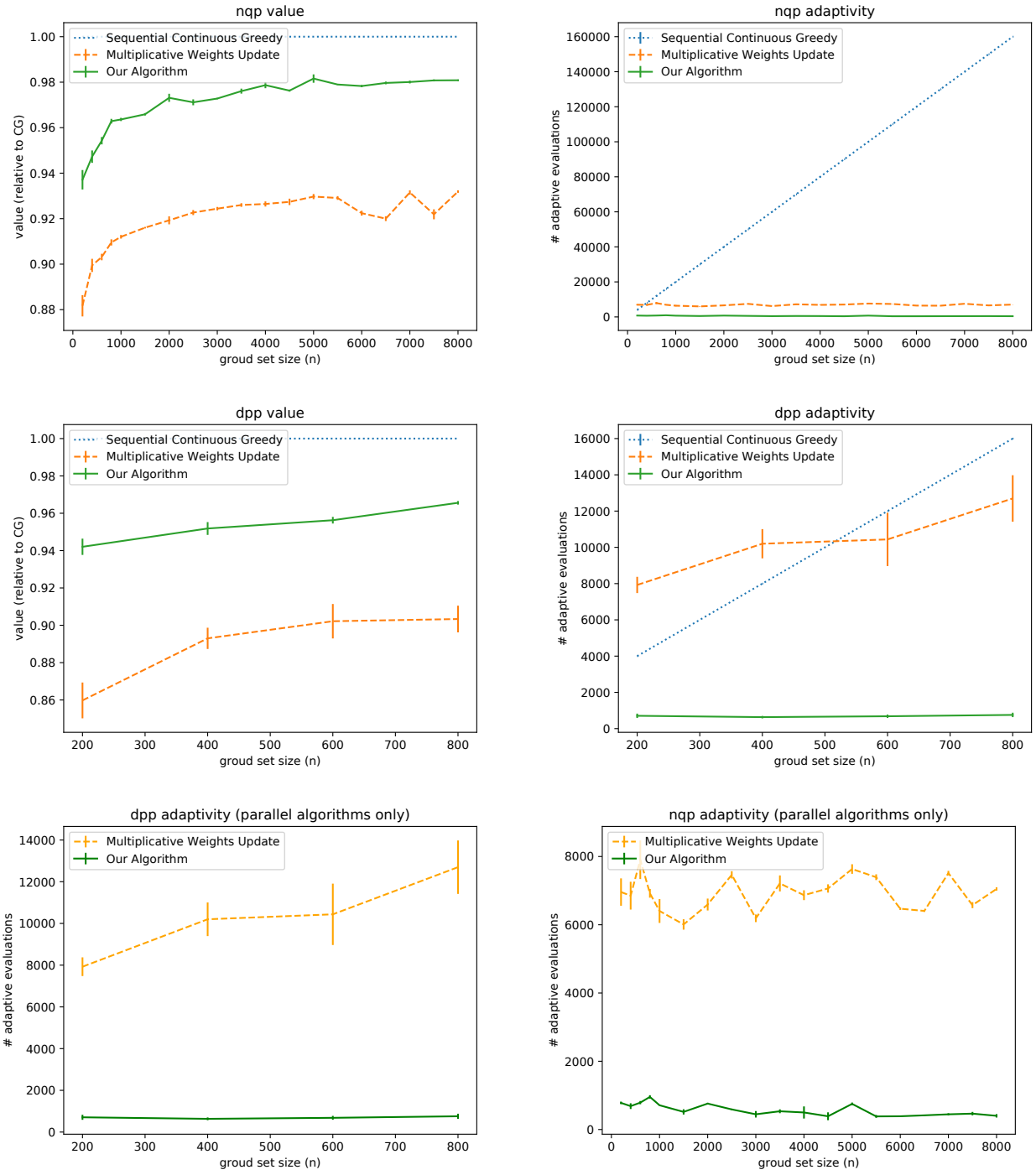

Figure 1. Experimental results.

References

Balkanski, E. and Singer, Y. The adaptive complexity of maximizing a submodular function. In *ACM Symposium on Theory of Computing (STOC)*, 2018.

Balkanski, E., Breuer, A., and Singer, Y. Non-monotone submodular maximization in exponentially fewer iterations. In *Advances in Neural Information Processing Systems (NIPS)*, 2018.

Balkanski, E., Rubinstein, A., and Singer, Y. An exponential speedup in parallel running time for submodular maximization without loss in approximation. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2019.

Bian, A., Levy, K., Krause, A., and Buhmann, J. M. Continuous dr-submodular maximization: Structure and algorithms. In *Advances in Neural Information Processing Systems*, pp. 486–496, 2017.

Bian, A., Buhmann, J. M., and Krause, A. Optimal dr-submodular maximization and applications to provable mean field inference. *arXiv preprint arXiv:1805.07482*, 2018.

Bian, A. A., Mirzasoleiman, B., Buhmann, J. M., and Krause, A. Guaranteed non-convex optimization: Submodular maximization over continuous domains. *arXiv preprint arXiv:1606.05615*, 2016.

Chekuri, C. and Quanrud, K. Submodular function maximization in parallel via the multilinear relaxation. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2019.

Chekuri, C., Jayram, T. S., and Vondrák, J. On multiplicative weight updates for concave and submodular function maximization. In *Conference on Innovations in Theoretical Computer Science (ITCS)*, 2015. doi: 10.1145/2688073.2688086.

Ene, A. and Nguyen, H. L. Submodular maximization with nearly-optimal approximation and adaptivity in nearly-linear time. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2019.

Ene, A., Nguyen, H. L., and Vladu, A. Submodular maximization with matroid and packing constraints in parallel. In *ACM Symposium on Theory of Computing (STOC)*, 2019.

Fahrbach, M., Mirrokni, V., and Zadimoghaddam, M. Non-monotone submodular maximization with nearly optimal adaptivity complexity. *arXiv preprint arXiv:1808.06932*, 2018.

Fahrbach, M., Mirrokni, V., and Zadimoghaddam, M. Submodular maximization with optimal approximation, adaptivity and query complexity. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2019.

Feldman, M., Naor, J., and Schwartz, R. A unified continuous greedy algorithm for submodular maximization. In *IEEE Foundations of Computer Science (FOCS)*, 2011. doi: 10.1109/FOCS.2011.46.

Gillenwater, J., Kulesza, A., and Taskar, B. Near-optimal map inference for determinantal point processes. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 2735–2743, 2012.

Ito, S. and Fujimaki, R. Large-scale price optimization via network flow. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 3855–3863, 2016.

Khanna, R., Elenberg, E. R., Dimakis, A. G., Negahban, S. N., and Ghosh, J. Scalable greedy feature selection via weak submodularity. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*, pp. 1560–1568, 2017.

Kulesza, A., Taskar, B., et al. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2–3):123–286, 2012.

Soma, T. and Yoshida, Y. Non-monotone dr-submodular function maximization. In *AAAI*, volume 17, pp. 898–904, 2017.

Vondrák, J. Optimal approximation for the submodular welfare problem in the value oracle model. In *ACM Symposium on Theory of Computing (STOC)*, 2008.