An Optimal Approximation for Submodular Maximization under a Matroid Constraint in the Adaptive Complexity Model

Eric Balkanski
ericbalkanski@g.harvard.edu
Harvard University
School of Engineering and Applied
Sciences
USA

Aviad Rubinstein
aviad@cs.stanford.edu
Stanford University
Department of Computer Science
USA

Yaron Singer
yaron@seas.harvard.edu
Harvard University
School of Engineering and Applied
Sciences
USA

ABSTRACT

In this paper we study submodular maximization under a matroid constraint in the adaptive complexity model. This model was recently introduced in the context of submodular optimization to quantify the information theoretic complexity of black-box optimization in a parallel computation model. Informally, the *adaptivity* of an algorithm is the number of sequential rounds it makes when each round can execute polynomially-many function evaluations in parallel. Since submodular optimization is regularly applied on large datasets we seek algorithms with low adaptivity to enable speedups via parallelization. Consequently, a recent line of work has been devoted to designing constant factor approximation algorithms for maximizing submodular functions under various constraints in the adaptive complexity model.

Despite the burst in work on submodular maximization in the adaptive complexity model, the fundamental problem of maximizing a monotone submodular function under a matroid constraint has remained elusive. In particular, all known techniques fail for this problem and there are no known constant factor approximation algorithms whose adaptivity is sublinear in the rank of the matroid k or in the worst case sublinear in the size of the ground set n.

In this paper we present an approximation algorithm for the problem of maximizing a monotone submodular function under a matroid constraint in the adaptive complexity model. The approximation guarantee of the algorithm is arbitrarily close to the optimal $1 - 1/e$ and it has near optimal adaptivity of $O(\log(n)\log(k))$. This result is obtained using a novel technique of *adaptive sequencing* which departs from previous techniques for submodular maximization in the adaptive complexity model. In addition to our main result we show how to use this technique to design other approximation algorithms with strong approximation guarantees and polylogarithmic adaptivity.

CCS CONCEPTS

• **Theory of computation → Approximation algorithms analysis**.

KEYWORDS

Adaptivity, submodular optimization, matroids, parallel algorithms

1 INTRODUCTION

In this paper we study submodular maximization under matroid constraints in the adaptive complexity model. The adaptive complexity model was recently introduced in the context of submodular optimization in [BS18a] to quantify the information theoretic complexity of black-box optimization in a parallel computation model. Informally, the *adaptivity* of an algorithm is the number of sequential rounds it makes when each round can execute polynomially-many function evaluations in parallel. The concept of adaptivity is heavily studied in computer science and optimization as it provides a measure of efficiency of parallel computation.

Since submodular optimization is regularly applied on very large datasets, we seek algorithms with low adaptivity to enable speedups via parallelization. For the basic problem of maximizing a monotone submodular function under a cardinality constraint k the celebrated greedy algorithm which iteratively adds to the solution the element with largest marginal contribution is $\Omega(k)$ adaptive. Until very recently, even for this basic problem, there was no known constant-factor approximation algorithm whose adaptivity is sublinear in k. In the worst case $k \in \Omega(n)$ and hence greedy and all other algorithms had adaptivity that is *linear* in the size of the ground set.

The main result in [BS18a] is an *adaptive sampling* algorithm for maximizing a monotone submodular function under a cardinality constraint that achieves a constant factor approximation arbitrarily close to $1/3$ in $O(\log n)$ adaptive rounds as well as a lower bound that shows that no algorithm can achieve a constant factor approximation in $\tilde{o}(\log n)$ rounds. Consequently, this algorithm provided a constant factor approximation with an exponential speedup in parallel runtime for monotone submodular maximization under a cardinality constraint.

In [BRS19, EN19], the adaptive sampling technique was extended to achieve an approximation guarantee arbitrarily close to the optimal $1 - 1/e$ in $O(\log n)$ adaptive rounds. This result was then

obtained with a linear number of queries [FMZ19], which is optimal. Functions with bounded curvature have also been studied using adaptive sampling under a cardinality constraint [BS18b]. The more general family of packing constraints, which includes partition and laminar matroids, has been considered in [CQ19]. In particular, under m packing constraints, a $1 - 1/e - \epsilon$ approximation was obtained in $O(\log^2 m \log n)$ rounds using a combination of continuous optimization and multiplicative weight update techniques.

1.1 Submodular Maximization under a Matroid Constraint

For the fundamental problem of maximizing a monotone submodular function under a general matroid constraint it is well known since the late 70s that the greedy algorithm achieves a $1/2$ approximation [NWF78] and that even for the special case of cardinality constraint no algorithm can obtain an approximation guarantee better than $1 - 1/e$ using polynomially-many value queries [NW78]. Thirty years later, in seminal work, Vondrák introduced the continuous greedy algorithm which approximately maximizes the multilinear extension of the submodular function [CCPV07] and showed it obtains the optimal $1 - 1/e$ approximation guarantee [Von08].

Despite the surge of interest in adaptivity of submodular maximization, the problem of maximizing a monotone submodular function under a matroid constraint in the adaptive complexity model has remained elusive. As we discuss in Section 1.4, when it comes to matroid constraints there are fundamental limitations of the techniques developed in this line of work. The best known adaptivity for obtaining a constant factor approximation guarantee for maximizing a monotone submodular function under a matroid constraint is achieved by the greedy algorithm and is linear in the rank of the matroid. The best known adaptivity for obtaining the optimal $1 - 1/e$ guarantee is achieved by the continuous greedy and is linear in the size of the ground set.

Is there an algorithm whose adaptivity is sublinear in the size of the rank of the matroid that obtains a constant factor approximation guarantee?

1.2 Main Result

Our main result is an algorithm for the problem of maximizing a monotone submodular function under a matroid constraint whose approximation guarantee is arbitrarily close to the optimal $1 - 1/e$ and has near optimal adaptivity of $O(\log(n) \log(k))$.

THEOREM. *For any $\epsilon > 0$ there is an $O\left(\log(n) \log\left(\frac{k}{\epsilon^3}\right) \frac{1}{\epsilon^3}\right)$ adaptive algorithm that, with probability $1 - o(1)$, obtains a $1 - 1/e - O(\epsilon)$ approximation for maximizing a monotone submodular function under a matroid constraint.*

Our result provides an exponential improvement in the adaptivity for maximizing a monotone submodular function under a matroid constraint with an arbitrarily small loss in approximation guarantee. As we later discuss, beyond the information theoretic consequences, this implies that a very broad class of combinatorial optimization problems can be solved exponentially faster in standard parallel computation models given appropriate representations of the matroid constraints.

Our main result is largely powered by a new technique developed in this paper which we call *adaptive sequencing*. This technique proves to be extremely powerful and is a departure from all previous techniques for submodular maximization in the adaptive complexity model. In addition to our main result we show that this technique gives us a set of other strong results that include:

- An $O(\log(n) \log(k))$ adaptive combinatorial algorithm that obtains a $\frac{1}{2} - \epsilon$ approximation for monotone submodular maximization under a matroid constraint (Theorem 1);
- An $O(\log(n) \log(k))$ adaptive combinatorial algorithm that obtains a $\frac{1}{P+1} - \epsilon$ approximation for monotone submodular maximization under intersection of P matroids (Theorem 7);
- An $O(\log(n) \log(k))$ adaptive algorithm that obtains an approximation of $1 - 1/e - \epsilon$ for monotone submodular maximization under a partition matroid constraint that can be implemented in the PRAM model with polylogarithmic depth (Appendix A).

In addition to these results the adaptive sequencing technique can be used to design algorithms that achieve the same results as those for cardinality constraint in [BRS19, EN19, FMZ19] and for non-monotone submodular maximization under cardinality constraint as in [BBS18] (Appendix A).

1.3 Technical Overview

The standard approach to obtain an approximation guarantee arbitrarily close to $1 - 1/e$ for maximizing a submodular function under a matroid constraint $\mathcal{M}$ is by the continuous greedy algorithm due to Vondrák [Von08]. This algorithm approximately maximizes the multilinear extension F of the submodular function [CCPV07] in $O(n)$ adaptive steps. In each step the algorithm updates a continuous solution $\mathbf{x} \in [0, 1]$ in the direction of $\mathbf{1}_S$, where S is chosen by maximizing an additive function under a matroid constraint.

In this paper we introduce the *accelerated continuous greedy* algorithm whose approximation is arbitrarily close to the optimal $1 - 1/e$. Similarly to continuous greedy, this algorithm approximately maximizes the multilinear extension by carefully choosing $S \in \mathcal{M}$ and updating the solution in the direction of $\mathbf{1}_S$. In sharp contrast to continuous greedy, however, the choice of S is done in a manner that allows making a *constant* number of updates to the solution, each requiring $O(\log(n) \log(k))$ adaptive rounds. We do this by constructing a feasible set S using $O(\log(n) \log(k))$ adaptive rounds, at each one of the $1/\lambda$ iterations of accelerated continuous greedy, s.t. S approximately maximizes the contribution of taking a step of constant size λ in the direction of $\mathbf{1}_S$. We construct S via a novel combinatorial algorithm introduced in Section 2.

The new combinatorial algorithm achieves by itself a $1/2$ approximation for submodular maximization under a matroid constraint in $O(\log(n) \log(k))$ adaptive rounds. This algorithm is developed using a fundamentally different approach from all previous low adaptivity algorithms for submodular maximization (see discussion in Section 1.4). This new framework uses a single random *sequence* $(a_1, \ldots, a_k)$ of elements. In particular, for each $i \in [k]$, element a_i is chosen uniformly at random among all elements such that $S \cup \{a_1, \ldots, a_i\} \in \mathcal{M}$. This random feasibility of each element is central to the analysis. Informally, this ordering allows the sequence to navigate randomly through the matroid constraint. For

each position i in this sequence, we analyze the number of elements a such that $S \cup \{a_1, \ldots, a_i\} \cup a \in \mathcal{M}$ and $f_{S \cup \{a_1, \ldots, a_i\}}(a)$ is large. The key observation is that if this number is large at a position i, by the randomness of the sequence, $f_{S \cup \{a_1, \ldots, a_i\}}(a_{i+1})$ is large w.h.p., which is important for the approximation. Otherwise, if this number is low we discard a large number of elements, which is important for the adaptivity.

In Section 3 we analyze the approximation of the accelerated continuous greedy algorithm, which is the main result of the paper. We use the algorithm from Section 2 to selects S as the direction and show $F(\mathbf{x} + \lambda \mathbf{1}_S) - F(\mathbf{x}) \geq (1 - \epsilon)\lambda(\text{OPT} - F(\mathbf{x}))$, which implies a $1 - 1/e - \epsilon$ approximation.

Finally, in Section 4 we parallelize the matroid oracle queries. The random sequence generated in each iteration of the combinatorial algorithm in Section 2 is independent of function evaluations and requires zero adaptive rounds, though it sequentially queries the matroid oracle. For practical implementation it is important to parallelize the matroid queries to achieve fast parallel runtime. When given explicit matroid constraints such as for uniform or partition matroids, this parallelization is relatively simple (Section A). For general matroid constraints given via rank or independence oracles we show how to parallelize the matroid queries in Section 4. We give upper and lower bounds by building on the seminal work of Karp, Upfal, and Wigderson on the parallel complexity of finding the base of a matroid [KUW88]. For rank oracles we show how to execute the algorithms with $O(\log(n)\log(k))$ parallel steps that matches the $O(\log(n)\log(k))$ adaptivity. For independence oracles we show how to execute the algorithm using $\tilde{O}(n^{1/2})$ steps of parallel matroid queries and give an $\tilde{\Omega}(n^{1/3})$ lower bound even for additive functions and partition matroids.

1.4 Previous Optimization Techniques in the Adaptive Complexity Model

The random sequencing approach developed in this paper is a fundamental departure from the adaptive sampling approach introduced in [BS18a] and employed in previous combinatorial algorithms that achieve low adaptivity for submodular maximization [BS18b, BBS18, BRS19, EN19, FMZ19, FMZ18]. In adaptive sampling an algorithm samples multiple large feasible sets at every iteration to determine elements which should be added to the solution or discarded. The issue with these uniformly random feasible sets is that, although they have a simple structure for uniform matroids, they are complex objects to generate and analyze for general matroid constraints.

Chekuri and Quanrud recently obtained a $1 - 1/e - \epsilon$ approximation in $O(\log^2 m \log n)$ adaptive rounds for the family of m packing constraints, which includes partition and laminar matroids [CQ19]. This setting was then also considered for non-monotone functions in [ENV18]. Their approach also uses the continuous greedy algorithm, combined with a multiplicative weight update technique to handle the constraints. Since general matroids consist of exponentially many constraints, a multiplicative weight update approach over these constraints is not feasible. More generally packing constraints assume an explicit representation of the matroid. For general matroid constraints, the algorithm is not given such a representation but an oracle. Access to an independence oracle for a

matroid breaks these results as shown in Section 4: any constant factor approximation algorithm with an independence oracle must have $\tilde{\Omega}(n^{1/3})$ sequential steps.

1.5 Preliminaries

Submodularity. A function $f : 2^N \to \mathbb{R}_+$ over ground set $N = [n]$ is *submodular* if the marginal contributions $f_S(a) := f(S \cup a) - f(S)$ of an element $a \in N \setminus S$ to a set $S \subseteq N$ are diminishing, meaning $f_S(a) \geq f_T(a)$ for all $S \subseteq T \subseteq N$ and $a \in N \setminus T$. Throughout the paper, we abuse notation by writing $S \cup a$ instead of $S \cup \{a\}$ and assume f is monotone, so $f(S) \leq f(T)$ for all $S \subseteq T$. The value of the optimal solution O for the problem of maximizing the submodular function under some constraint $\mathcal{M}$ is denoted by OPT, i.e. $O := \text{argmax}_{S \in \mathcal{M}} f(S)$ and $\text{OPT} := f(O)$.

Adaptivity. Given a value oracle f, an algorithm is r-**adaptive** if every query $f(S)$ for the value of a set S occurs at a round $i \in [r]$ s.t. S is independent of the values $f(S')$ of all other queries at round i, with at most poly(n) queries at every round.

Matroids. A set system $\mathcal{M} \subseteq 2^N$ is a *matroid* if it satisfies the *downward closed* and *augmentation* properties. A set system $\mathcal{M}$ is downward closed if for all $S \subseteq T$ such that $T \in \mathcal{M}$, then $S \in \mathcal{M}$. The augmentation property is that if $S, T \in \mathcal{M}$ and $|S| < |T|$, then there exists $a \in T$ such that $S \cup a \in \mathcal{M}$. We call a set $S \in \mathcal{M}$ *feasible* or *independent*. The rank $k = \text{RANK}(\mathcal{M})$ of a matroid is the maximum size of an independent set S. The rank $\text{RANK}(S)$ of a set S is the maximum size of an independent subset $T \subseteq S$. A set $B \in \mathcal{M}$ is called a base of $\mathcal{M}$ if $|B| = \text{RANK}(\mathcal{M})$. The matroid polytope $P(\mathcal{M})$ is the collection of points $\mathbf{x} \in [0, 1]^n$ in the convex hull of the independent sets of $\mathcal{M}$, or equivalently the points $\mathbf{x}$ such that $\sum_{i \in S} x_i \leq \text{RANK}(S)$ for all $S \subseteq [n]$.

The multilinear extension. The multilinear extension $F : [0, 1]^n \to \mathbb{R}_+$ of a function f maps a point $\mathbf{x} \in [0, 1]^n$ to the expected value of a random set $R \sim \mathbf{x}$ containing each element $i \in [n]$ with probability x_i independently, i.e. $F(\mathbf{x}) = \mathbb{E}_{R \sim \mathbf{x}}[f(R)]$. We note that given an oracle for f, one can estimate $F(\mathbf{x})$ arbitrarily well in one round by querying in parallel a sufficiently large number of samples $R_1, \ldots, R_m$ drawn i.i.d. from $\mathbf{x}$ and taking the average value of $f(R_i)$ over $i \in [m]$ [CJV15, CQ19]. For ease of presentation, we assume throughout the paper that we are given access to an exact value oracle for F in addition to f. The results which rely on F then extend to the case where the algorithm is only given an oracle for f with an arbitrarily small loss in the approximation, no loss in the adaptivity, and additional $O(n \log n)$ factor in the query complexity.[1]

2 THE COMBINATORIAL ALGORITHM

In this section we describe a combinatorial algorithm used at every iteration of the accelerated continuous greedy algorithm to find a direction $\mathbf{1}_S$ for an update of a continuous solution. In the next section we will show how to use this algorithm as a subprocedure in the accelerated continuous greedy algorithm to achieve an approximation arbitrarily close to $1 - 1/e$ with $O(\log(n)\log(k))$

---

[1]With $O(\epsilon^{-2} n \log n)$ samples, $F(\mathbf{x})$ is estimated within a $(1 \pm \epsilon)$ multiplicative factor with high probability[CQ19].

adaptivity. The optimization of this direction S is itself an instance of maximizing a monotone submodular function under a matroid constraint. The main result of this section is a $O(\log(n)\log(k))$ adaptive algorithm, which we call ADAPTIVE SEQUENCING, that returns a solution $\{a_i\}_i$ s.t., for all i, the marginal contribution of a_i to $\{a_1, \ldots, a_{i-1}\}$ is near optimal with respect to all elements a s.t. $\{a_1, \ldots, a_{i-1}, a\} \in \mathcal{M}$. We note that this guarantee also implies that ADAPTIVE SEQUENCING itself achieves an approximation that is arbitrarily close to 1/2 with high probability.

As discussed in Section 1.3 unlike all previous low-adaptivity combinatorial algorithms for submodular maximization, the ADAPTIVE SEQUENCING algorithm developed here does not iteratively sample large sets of elements in parallel at every iteration. Instead, it samples a *single* random *sequence* of elements in every iteration. Importantly, this sequence is generated without any function evaluations, and therefore can be executed in zero adaptive rounds. The goal is then to identify a high-valued prefix of the sequence that can be added to the solution and discard a large number of low-valued elements at every iteration. Identifying a high valued prefix enables the approximation guarantee and discarding a large number of elements in every iteration ensures low adaptivity.

2.1 Generating Random Feasible Sequences

The algorithm crucially requires generating a random sequence of elements in zero adaptive rounds.

DEFINITION 1. *Given a matroid $\mathcal{M}$ we say $(a_1, \ldots, a_{RANK(\mathcal{M})})$ is a **random feasible sequence** if for all $i \in [RANK(\mathcal{M})]$, a_i is an element chosen u.a.r. from $\{a : \{a_1, \ldots, a_{i-1}, a\} \in \mathcal{M}\}$.*

A simple way to obtain a random feasible sequence is by sampling feasible elements sequentially.

Algorithm 1 RANDOM SEQUENCE

Input: matroid $\mathcal{M}$
    for $i = 1$ to RANK($\mathcal{M}$) **do**
        $X \leftarrow \{a : \{a_1, \ldots, a_{i-1}, a\} \in \mathcal{M}\}$
        $a_i \sim$ a uniformly random element from X
    return $a_1, \ldots, a_{RANK(\mathcal{M})}$

It is immediate that Algorithm 1 outputs a random feasible sequence. Since Algorithm 1 is independent of f, its adaptivity is zero. For ease of presentation, we describe the algorithm using RANDOM SEQUENCE as a subroutine, despite its sequential calls to the matroid oracle. In Section 4 we show how to efficiently parallelize this procedure using standard matroid oracles.

2.2 The Algorithm

The main idea behind the algorithm is to generate a random feasible sequence in each adaptive round, and use that sequence to determine which elements should be added to the solution and which should be discarded from consideration. Given a position $i \in \{1, \ldots, l\}$ in a sequence $(a_1, a_2, \ldots, a_l)$, a subset S, and threshold t, we say that an element a is *good* if adding it to $S \cup \{a_1, \ldots, a_i\}$ satisfies the matroid constraint and its marginal contribution to $S \cup \{a_1, \ldots, a_i\}$ is at least threshold t. In each adaptive round the

algorithm generates a random feasible sequence and finds the index $i^\star$ which is the minimal index i such that at most a $1 - \epsilon$ fraction of the surviving elements X are good. The algorithm then adds the set $\{a_1, \ldots, a_{i^\star}\}$ to S. A formal description of the algorithm is included below. We use $\mathcal{M}(S, X) := \{T \subseteq X : S \cup T \in \mathcal{M}\}$ to denote the matroid over elements X where a subset is feasible in $\mathcal{M}(X, S)$ if its union with the current solution S is feasible according to $\mathcal{M}$.

Algorithm 2 ADAPTIVE SEQUENCING

Input: function f, feasibility constraint $\mathcal{M}$
  $S \leftarrow \emptyset, t \leftarrow \max_{a \in N} f(a)$
  for Δ iterations **do**
    $X \leftarrow N$
    while $X \neq \emptyset$ **do**
      $a_1, \ldots, a_{RANK(\mathcal{M}(S,X))} \leftarrow$ RANDOM SEQUENCE$(\mathcal{M}(S, X))$
      $X_i \leftarrow \{a \in X : S \cup \{a_1, \ldots, a_i, a\} \in \mathcal{M}$ and $f_{S \cup \{a_1, \ldots, a_i\}}(a) \geq t\}$
      $i^\star \leftarrow \min\{i : |X_i| \leq (1 - \epsilon)|X|\}$
      $S \leftarrow S \cup \{a_1, \ldots, a_{i^\star}\}$
      $X \leftarrow X_{i^\star}$
    $t \leftarrow (1 - \epsilon)t$
  return S

Intuitively, adding $\{a_1, \ldots, a_{i^\star}\}$ to the current solution S is desirable for two important reasons. First, for a random feasible sequence we have that $S \cup \{a_1, \ldots, a_{i^\star}\} \in \mathcal{M}$ and for each element a_i at a position $i \leq i^\star$, there is a high likelihood that the marginal contribution of a_i to the previous elements in the sequence is at least t. Second, by definition of $i^\star$ a constant fraction ϵ of elements are not good at that position, and we discard these elements from X. This discarding guarantees that there are at most logarithmically many iterations until X is empty.

The threshold t maintains the invariant that it is approximately an upper bound on the optimal marginal contribution to the current solution. By submodularity, the optimal marginal contribution to S decreases as S grows. Thus, to maintain the invariant, the algorithm iterates over decreasing values of t. In particular, at each of $\Delta = O\left(\frac{1}{\epsilon}\log\left(\frac{k}{\epsilon}\right)\right)$ iterations, where $k := RANK(\mathcal{M})$, the algorithm decreases t by a $1 - \epsilon$ factor when there are no more elements which can be added to S with marginal contribution at least t, so when X is empty.

2.3 Adaptivity

In each inner-iteration the algorithm makes polynomially-many queries that are independent of each other. Indeed, in each iteration, we generate $X_1, \ldots, X_{k-|S|}$ non-adaptively and make at most n function evaluations for each X_i. The adaptivity immediately follows from the definition of $i^\star$ that ensures an ϵ fraction of surviving elements in X are discarded at every iteration.

LEMMA 1. *With $\Delta = O\left(\frac{1}{\epsilon}\log\left(\frac{k}{\epsilon}\right)\right)$, ADAPTIVE SEQUENCING has adaptivity $O\left(\log(n)\log\left(\frac{k}{\epsilon}\right)\frac{1}{\epsilon^2}\right)$.*

PROOF. The for loop has Δ iterations. The while loop has at most $O(\epsilon^{-1}\log n)$ iterations since, by definition of $i^\star$, an ϵ fraction of

the surviving elements are discarded from X at every iteration. We can find $i^\star$ by computing X_i for each $i \in [k]$ in parallel in one round.　□

We note that the query complexity, $O\left(nk\log(n)\log\left(\frac{k}{\epsilon}\right)\frac{1}{\epsilon^2}\right)$, of the algorithm can be improved to $O\left(n\log(n)\log(k)\log\left(\frac{k}{\epsilon}\right)\frac{1}{\epsilon^2}\right)$ if we allow $O\left(\log(n)\log(k)\log\left(\frac{k}{\epsilon}\right)\frac{1}{\epsilon^2}\right)$ adaptivity by doing a binary search over at most k sets X_i to find $i^\star$. The details can be found in Appendix B.

2.4 Approximation Guarantee

The main result for the approximation guarantee is that the algorithm returns a solution $S = \{a_1, \ldots, a_l\}$ s.t. for all $i \leq l$, the marginal contribution obtained by a_i to $\{a_1, \ldots, a_{i-1}\}$ is near optimal with respect to all elements a such that $\{a_1, \ldots, a_{i-1}, a\} \in \mathcal{M}$. To prove this we show that the threshold t is an approximate upper bound on the maximum marginal contribution.

LEMMA 2. *Assume that f is submodular and that $\mathcal{M}$ is downward closed. Then, at any iteration, $t \geq (1-\epsilon)\max_{a:S\cup a\in\mathcal{M}} f_S(a)$.*

PROOF. The claim initially holds by the initial definitions of $t = \max_{a\in N} f(a)$, $S = \emptyset$ and $X = N$. We show that this invariant is maintained through the algorithm when either S or t are updated.

First, assume that at some iteration of the algorithm we have $t \geq (1-\epsilon)\max_{a:S\cup a\in\mathcal{M}} f_S(a)$ and that S is updated to $S\cup\{a_1,\ldots,a_{i^\star}\}$. Then, for all a such that $S\cup a\in\mathcal{M}$,

$$f_{S\cup\{a_1,\ldots,a_{i^\star}\}}(a) \leq f_S(a) \leq t/(1-\epsilon)$$

where the first inequality is by submodularity and the second by the inductive hypothesis. Since $\{a : S\cup\{a_1,\ldots,a_{i^\star}\}\cup a\in\mathcal{M}\} \subseteq \{a : S\cup a\in\mathcal{M}\}$ by the downward closed property of $\mathcal{M}$,

$$\max_{a:S\cup\{a_1,\ldots,a_{i^\star}\}\cup a\in\mathcal{M}} f_{S\cup\{a_1,\ldots,a_{i^\star}\}}(a)$$
$$\leq \max_{a:S\cup a\in\mathcal{M}} f_{S\cup\{a_1,\ldots,a_{i^\star}\}}(a).$$

Thus, when S is updated to $S\cup\{a_1,\ldots,a_{i^\star}\}$, we have $t \geq (1-\epsilon)\max_{a:S\cup\{a_1,\ldots,a_{i^\star}\}\cup a\in\mathcal{M}} f_S(a)$.

Next, consider an iteration where t is updated to $t' = (1-\epsilon)t$. By the algorithm, $X = \emptyset$ at that iteration with current solution S. Thus, by the algorithm, for all $a\in N$, a was discarded from X at some previous iteration with current solution S' s.t. $S'\cup\{a_1,\ldots,a_{i^\star}\}\subseteq S$. Since a was discarded, it is either the case that $S'\cup\{a_1,\ldots,a_{i^\star}\}\cup a\notin\mathcal{M}$ or $f_{S'\cup\{a_1,\ldots,a_{i^\star}\}}(a) < t$. If $S'\cup\{a_1,\ldots,a_{i^\star}\}\cup a\notin\mathcal{M}$ then $S\cup a\notin\mathcal{M}$ by the downward closed property of $\mathcal{M}$ and since $S'\cup\{a_1,\ldots,a_{i^\star}\}\subseteq S$. Otherwise, $f_{S'\cup\{a_1,\ldots,a_{i^\star}\}}(a) < t$ and by submodularity, $f_S(a) \leq f_{S'\cup\{a_1,\ldots,a_{i^\star}\}}(a) < t = t'/(1-\epsilon)$. Thus, $\forall a\in N$ s.t. $S\cup a\in\mathcal{M}$, $t' \geq (1-\epsilon)f_S(a)$ and the invariant is maintained.　□

By exploiting the definition of $i^\star$ and the random feasible sequence property we show that Lemma 2 implies that every element added to S at some iteration j has near-optimal expected marginal contribution to S. We define $X_i^{\mathcal{M}} := \{a\in X : S\cup\{a_1,\ldots,a_i\}\cup a\in \mathcal{M}\}$.

LEMMA 3. *Assume that $a_1,\ldots,a_{\text{RANK}(\mathcal{M}(S,X))}$ is a random feasible sequence, then for all $i\leq i^\star$,*

$$\mathbb{E}_{a_i}\left[f_{S\cup\{a_1,\ldots,a_{i-1}\}}(a_i)\right]$$
$$\geq (1-\epsilon)^2 \max_{a:S\cup\{a_1,\ldots,a_{i-1}\}\cup a\in\mathcal{M}} f_{S\cup\{a_1,\ldots,a_{i-1}\}}(a_i).$$

PROOF. By the random feasibility condition, we have $a_i \sim \mathcal{U}(X_{i-1}^{\mathcal{M}})$. We get

$$\Pr_{a_i}\left[f_{S\cup\{a_1,\ldots,a_{i-1}\}}(a_i)\geq t\right]\cdot t$$
$$= \frac{|X_{i-1}|}{|X_{i-1}^{\mathcal{M}}|}\cdot t$$
$$\geq \frac{|X_{i-1}|}{|X|}\cdot t$$
$$\geq (1-\epsilon)(1-\epsilon)\max_{a:S\cup\{a_1,\ldots,a_{i-1}\}\cup a\in\mathcal{M}} f_{S_{i-1}}(a_i)$$

where the equality is by definition of X_{i-1}, the first inequality since $X_{i-1}^{\mathcal{M}}\subseteq X$, and the second since $i\leq i^\star$ and by Lemma 2. Finally, note that $\mathbb{E}\left[f_{S\cup\{a_1,\ldots,a_{i-1}\}}(a_i)\right]\geq\Pr\left[f_{S\cup\{a_1,\ldots,a_{i-1}\}}(a_i)\geq t\right]\cdot t$.　□

Next, we show that if every element a_i in a solution $S = \{a_1,\ldots,a_k\}$ of size $k = \text{RANK}(\mathcal{M})$ has near-optimal expected marginal contribution to $S_{i-1} := \{a_1,\ldots,a_{i-1}\}$, then we obtain an approximation arbitrarily close to $1/2$ in expectation.

LEMMA 4. *Assume that $S = \{a_1,\ldots,a_k\}$ s.t. $\mathbb{E}_{a_i}[f_{S_{i-1}}(a_i)] \geq (1-\epsilon)\max_{a:S_{i-1}\cup a\in\mathcal{M}} f_{S_{i-1}}(a)$ where $S_i = \{a_1,\ldots,a_i\}$. Then, for a matroid constraint $\mathcal{M}$, we have $\mathbb{E}[f(S)]\geq(1/2-O(\epsilon))\text{OPT}$.*

PROOF. Let $O = \{o_1,\ldots,o_k\}$ such that $\{a_1,\ldots,a_{i-1},o_i\}$ is feasible for all i, which exists by the augmentation property of matroids. We get,

$$\mathbb{E}[f(S)] = \sum_{i\in[k]}\mathbb{E}[f_{S_{i-1}}(a_i)]$$
$$\geq (1-\epsilon)\sum_{i\in[k]}\mathbb{E}[f_{S_{i-1}}(o_i)]$$
$$\geq (1-\epsilon)f_S(O)\geq(1-\epsilon)(\text{OPT}-f(S)).　□$$

A corollary of the lemmas above is that ADAPTIVE SEQUENCING has $O(\log(n)\log(k))$ adaptive rounds and provides an approximation that is arbitrarily close to $1/2$, in expectation. To obtain this guarantee with high probability we can simply run parallel instances of the while-loop in the algorithm and include the elements obtained from the best instance. We also note that the solution S returned by ADAPTIVE SEQUENCING might have size smaller than $\text{RANK}(\mathcal{M})$, which causes an arbitrarily small loss for sufficiently large Δ. We give the full details in Appendix B.

THEOREM 1. *For any $\epsilon > 0$, there is an $O\left(\log(n)\log\left(\frac{k}{\epsilon}\right)\frac{1}{\epsilon^2}\right)$ adaptive algorithm that obtains a $1/2 - O(\epsilon)$ approximation with probability $1 - o(1)$ for maximizing a monotone submodular function under a matroid constraint.*

In Appendix B, we generalize this result and obtain a $1/(P+1) - O(\epsilon)$ approximation with high probability for the intersection of P matroids.

3 THE ACCELERATED CONTINUOUS GREEDY ALGORITHM

In this section we describe the accelerated continuous greedy algorithm that achieves the main result of the paper. This algorithm employs the combinatorial algorithm from the previous section to construct a continuous solution which approximately maximizes the multilinear relaxation F of the function f. This algorithm requires $O(\log(n)\log(k))$ adaptive rounds and it produces a continuous solution whose approximation to the optimal solution is with high probability arbitrarily close to $1 - 1/e$. Finally, since the solution is continuous and we seek a feasible *discrete* solution, it requires rounding. Fortunately, by using either dependent rounding [CVZ09] or contention resolution schemes [VCZ11] this can be done with an arbitrarily small loss in the approximation guarantee without any function evaluations, and hence without any additional adaptive rounds.

3.1 The Algorithm

The accelerated continuous greedy algorithm follows the same principle as the (standard) continuous greedy algorithm [Von08]: at every iteration, the solution $\mathbf{x} \in [0,1]^n$ moves in the direction of a feasible set $S \in \mathcal{M}$. The crucial difference between the accelerated continuous greedy and the standard continuous greedy is in the choice of this set S guiding the direction in which $\mathbf{x}$ moves. This difference allows the accelerated continuous greedy to terminate after a *constant* number of iterations, each of which has $O(\log(n)\log(k))$ adaptive rounds, in contrast to the continuous greedy which requires a linear number of iterations.

To determine the direction in every iteration, the accelerated continuous greedy applies Adaptive Sequencing on the surrogate function g that measures the marginal contribution to $\mathbf{x}$ when taking a step of size λ in the direction of S. That is, $g(S) := F_{\mathbf{x}}(\lambda S) = F(\mathbf{x} + \lambda S) - F(\mathbf{x})$ where we abuse notation and write λS instead of $\lambda \mathbf{1}_S$ for $\lambda \in [0,1]$ and $S \subseteq N$. Since f is a monotone submodular function it is immediate that g is monotone and submodular as well.

Algorithm 3 Accelerated Continuous Greedy

Input: matroid $\mathcal{M}$, step size λ
  $\mathbf{x} \leftarrow \mathbf{0}$
  for $1/\lambda$ iterations **do**
    define $g : 2^N \rightarrow \mathbb{R}$ to be $g(T) = F_{\mathbf{x}}(\lambda T)$
    $S \leftarrow$ Adaptive Sequencing$(g, \mathcal{M})$
    $\mathbf{x} \leftarrow \mathbf{x} + \lambda S$
  return $\mathbf{x}$

The analysis shows that in every one of the $1/\lambda$ iterations, Adaptive Sequencing finds S such that the contribution of taking a step of size λ in the direction of S is approximately a λ fraction of $\text{OPT} - F(\mathbf{x})$. For any λ this is a sufficient condition for obtaining the $1 - 1/e - \epsilon$ guarantee.

The reason why the standard continuous greedy cannot be implemented with a constant number of rounds $1/\lambda$ is that in every round it moves in the direction of $\mathbf{1}_S$ for $S := \text{argmax}_{T \in \mathcal{M}} \sum_{a \in T} g(a)$. When λ is constant $F_{\mathbf{x}}(\lambda S)$ is arbitrarily low due to the potential overlap between high valued singletons (see Appendix C). Selecting

S using Adaptive Sequencing is the crucial part of the accelerated continuous greedy which allows implementing it in a constant number of iterations.

3.2 Analysis

We start by giving a sufficient condition on Adaptive Sequencing to obtain the $1 - 1/e - O(\epsilon)$ approximation guarantee. The analysis is standard and the proof is deferred to Appendix C.

Lemma 5. *For a given matroid $\mathcal{M}$ assume that Adaptive Sequencing outputs $S \in \mathcal{M}$ s.t.*

$$\mathbb{E}_S[F_{\mathbf{x}}(\lambda S)] \geq (1 - \epsilon)\lambda(\text{OPT} - F(\mathbf{x}))$$

at every iteration of Accelerated Continuous Greedy. Then Accelerated Continuous Greedy outputs $\mathbf{x} \in P(\mathcal{M})$ s.t.

$$\mathbb{E}[F(\mathbf{x})] \geq (1 - 1/e - \epsilon)\text{OPT}.$$

For a set $S = \{a_1, \ldots, a_k\}$ we define $S_i := \{a_1, \ldots, a_i\}$ and $S_{j:k} := \{a_j, \ldots, a_k\}$. We use this notation in the lemma below. The lemma is folklore and proved in Appendix C for completeness.

Lemma 6. *Let $\mathcal{M}$ be a matroid, then for any feasible sets $S = \{a_1, \ldots, a_k\}$ and O of size k, there exists an ordering of $O = \{o_1, \ldots, o_k\}$ where for all $i \in [k]$, $S_i \cup O_{i+1:k} \in \mathcal{M}$ and $S_i \cap O_{i+1:k} = \emptyset$.*

The following lemma is key in our analysis. We argue that unless the algorithm already constructed S of sufficiently large value, the sum of the contributions of the optimal elements to S is arbitrarily close to the desired $\lambda(\text{OPT} - F(\mathbf{x}))$.

Lemma 7. *Assume that $g(S) \leq \lambda(\text{OPT} - F(\mathbf{x}))$, then*

$$\sum_i g_{S \setminus O_{i:k}}(o_i) \geq \lambda(1 - \lambda)(\text{OPT} - F(\mathbf{x})).$$

Proof. We first lower bound this sum of marginal contribution of optimal elements with the contribution of the optimal solution to the current solution $\mathbf{x} + \lambda S$ at the end of the iteration:

$$\begin{aligned}
\sum_{i \in [k]} g_{S \setminus O_{i:k}}(o_i) &= \sum_{i \in [k]} F_{\mathbf{x} + \lambda S \setminus O_{i:k}}(\lambda o_i) \\
&\geq \sum_{i \in [k]} F_{\mathbf{x} + O_{i-1} + \lambda S}(\lambda o_i) \\
&\geq \lambda \sum_{i \in [k]} F_{\mathbf{x} + O_{i-1} + \lambda S}(o_i) \\
&= \lambda F_{\mathbf{x} + \lambda S}(O)
\end{aligned}$$

where the first inequality is by submodularity and the second by the multilinearity of F. In the standard analysis of greedy algorithms the optimal solution O may overlap with the current solution. In the continuous algorithm, since the algorithm takes steps of size λ, we can bound the overlap between the solution at this iteration λS and the optimal solution:

$$\begin{aligned}
F_{\mathbf{x} + \lambda S}(O) &= F_{\mathbf{x}}(O + \lambda S) - F_{\mathbf{x}}(\lambda S) \\
&\geq F_{\mathbf{x}}(O) - \lambda(\text{OPT} - F(\mathbf{x})) \\
&= (1 - \lambda)(\text{OPT} - F(\mathbf{x}))
\end{aligned}$$

the first inequality is by monotonicity and lemma assumption and the second by monotonicity. □

As shown in Lemma 6, ADAPTIVE SEQUENCING picks elements a_i with near-optimal marginal contributions. Together with Lemma 7 we get the desired bound on the contribution of λS to $\mathbf{x}$.

LEMMA 8. *Let* $\Delta = O\left(\frac{1}{\epsilon} \log \left(\frac{k}{\epsilon \lambda}\right)\right)$ *and* $\lambda = O(\epsilon)$. *For any* $\mathbf{x}$ *s.t.* $F(\mathbf{x}) < (1 - 1/e)\mathsf{OPT}$, *the set S returned by* ADAPTIVE SEQUENCING$(g, \mathcal{M})$ *satisfies*

$$\mathbb{E}\left[F_{\mathbf{x}}(\lambda S)\right] \geq (1 - O(\epsilon))\lambda(\mathsf{OPT} - F(\mathbf{x})).$$

PROOF. Initially, we have $t_i < \mathsf{OPT}$. After $\Delta = O\left(\frac{1}{\epsilon} \log \left(\frac{k}{\epsilon \lambda}\right)\right)$ iterations of the outer loop of ADAPTIVE SEQUENCING, we get $t_f = (1 - \epsilon)^{\Delta}\mathsf{OPT} = O\left(\frac{\epsilon \lambda \mathsf{OPT}}{k}\right)$. We begin by adding dummy elements to S so that $|S| = k$, which enables pairwise comparisons between S and O. In particular, we consider S', which is S together with RANK$(\mathcal{M}) - |S|$ dummy elements $a_{|S|+1}, \dots a_k$ such that, for any $\mathbf{y}$ and λ, $F_{\mathbf{y}}(\lambda a) = t_f$, which is the value of t when ADAPTIVE SEQUENCING terminates. Thus, by Lemma 2, for dummy elements a_i, $g_{S_{i-1}}(a_i) = t_f \geq (1 - \epsilon) \max_{a:S_{i-1} \cup a \in \mathcal{M}} g_{S_{i-1}}(a)$.

We will conclude the proof by showing that S is a good approximation to S'. From Lemma 3 that the contribution of a_i to S_{i-1} approximates the optimal contribution to S_{i-1}:

$$\mathbb{E}\left[F_{\mathbf{x}}(\lambda S')\right] = \sum_{i=1}^{k} \mathbb{E}\left[g_{S_{i-1}}(a_i)\right] \geq \sum_{i=1}^{k}(1 - \epsilon)^2 \max_{a:S_{i-1} \cup a \in \mathcal{M}} g_{S_{i-1}}(a_i).$$

By Lemma 6 and submodularity, we have

$$\max_{a:S_{i-1} \cup a \in \mathcal{M}} g_{S_{i-1}}(a_i) \geq g_{S \setminus O_{i:k}}(o_i).$$

By Lemma 7, we also have

$$\sum_{i=1}^{k} g_{S \setminus O_{i:k}}(o_i) \geq \lambda(1 - \lambda)(\mathsf{OPT} - F(\mathbf{x})).$$

Combining the previous pieces, we obtain

$$\mathbb{E}\left[F_{\mathbf{x}}(\lambda S')\right] \geq (1 - \epsilon)^2 \lambda(1 - \lambda)(\mathsf{OPT} - F(\mathbf{x})).$$

We conclude by removing the value of dummy elements,

$$\mathbb{E}\left[F_{\mathbf{x}}(\lambda S)\right] = \mathbb{E}\left[F_{\mathbf{x}}(\lambda S') - F_{\mathbf{x}+\lambda S}(\lambda(S' \setminus S))\right]$$
$$\geq \mathbb{E}\left[F_{\mathbf{x}}(\lambda S')\right] - k t_f \geq \mathbb{E}\left[F_{\mathbf{x}}(\lambda S')\right] - \epsilon \lambda \mathsf{OPT}.$$

The lemma assumes that $F(\mathbf{x}) < (1 - 1/e)\mathsf{OPT}$ and $\lambda = O(\epsilon)$, so $\mathsf{OPT} \leq e(\mathsf{OPT} - F(\mathbf{x}))$ and $\epsilon \lambda \mathsf{OPT} = O(\epsilon)\lambda(\mathsf{OPT} - F(\mathbf{x}))$. We conclude that $\mathbb{E}\left[F_{\mathbf{x}}(\lambda S)\right] \geq (1 - O(\epsilon))\lambda(\mathsf{OPT} - F(\mathbf{x}))$. □

The approximation guarantee of the ACCELERATED CONTINUOUS GREEDY follows from lemmas 8 and 5, and the adaptivity from Lemma 1. We defer the proof to Appendix C.

THEOREM 2. *For any* $\epsilon > 0$ ACCELERATED CONTINUOUS GREEDY *makes* $O\left(\log(n) \log \left(\frac{k}{\epsilon^2}\right) \frac{1}{\epsilon^2}\right)$ *adaptive rounds and obtains a* $1 - 1/e - O(\epsilon)$ *approximation in expectation for maximizing a monotone submodular function under a matroid constraint.*

The final step in our analysis shows that the guarantee of AC-CELERATED CONTINUOUS GREEDY holds not only in expectation but also with high probability. To do so we argue in the lemma below that if over all iterations i, $F_{\mathbf{x}}(\lambda S)$ is close *on average over the rounds* to $\lambda(\mathsf{OPT} - F(\mathbf{x}))$, we obtain an approximation arbitrarily close to $1 - 1/e$ with high probability. The proof is in Appendix C.

LEMMA 9. *Assume that* ADAPTIVE SEQUENCING *outputs* $S \in \mathcal{M}$ *s.t.* $F_{\mathbf{x}}(\lambda S) \geq \alpha_i \lambda(\mathsf{OPT} - F(\mathbf{x}))$ *at every iteration i of* ACCELERATED CONTINUOUS GREEDY *and that* $\lambda \sum_{i=1}^{\lambda^{-1}} \alpha_i \geq 1 - \epsilon$. *Then* ACCELERATED CONTINUOUS GREEDY *outputs* $\mathbf{x} \in P(\mathcal{M})$ *s.t.* $F(\mathbf{x}) \geq (1 - 1/e - \epsilon)\mathsf{OPT}$.

The approximation α_i obtained at iteration i is $1 - O(\epsilon)$ in expectation by Lemma 8. Thus, by a simple concentration bound, w.h.p. it is close to $1 - O(\epsilon)$ in average over all iterations. Together with Lemma 9, this implies the $1 - 1/e - \epsilon$ approximation w.h.p.. The details are in Appendix C.

THEOREM 3. ACCELERATED CONTINUOUS GREEDY *is an algorithm with adaptivity* $O\left(\log(n) \log \left(\frac{k}{\epsilon \lambda}\right) \frac{1}{\epsilon \lambda}\right)$ *that, with probability* $1 - \delta$, *obtains a* $1 - 1/e - O(\epsilon)$ *approximation for maximizing a monotone submodular function under a matroid constraint, with step size* $\lambda = O\left(\epsilon^2 \log^{-1}\left(\frac{1}{\delta}\right)\right)$.

4 PARALLELIZATION OF MATROID ORACLE QUERIES

Throughout the paper we relied on RANDOM SEQUENCE as a simple procedure to generate a random feasible sequence to achieve our $O(\log(n) \log(k))$ adaptive algorithm with an approximation arbitrarily close to $1 - 1/e$. Although RANDOM SEQUENCE has zero adaptivity, it makes RANK$(\mathcal{M})$ sequential steps depending on membership in the matroid to generate the sets $X_1, \dots, X_{\text{RANK}(\mathcal{M})}$. From a practical perspective, we may wish to accelerate this process via parallelization. In this section we show how to do so in the standard *rank* and *independence* oracle models for matroids.

4.1 Matroid Rank Oracles

Given a rank oracle for the matroid, we get an algorithm that only makes $O(\log(n) \log(k))$ steps of matroid oracle queries and has polylogarithmic depth on a PRAM machine. Recall that a rank oracle for $\mathcal{M}$ is given a set S and returns its rank, i.e. the maximum size of an independent subset $T \subseteq S$. The number of steps of matroid queries of an algorithm is the number of sequential steps it makes when polynomially-many queries to a matroid oracle for $\mathcal{M}$ can be executed in parallel in each step [KUW88].[2] We use a parallel algorithm from [KUW88] designed for constructing a base of a matroid with a rank oracle, and show that it satisfies the random feasibility property.

Algorithm 4 Parallel RANDOM SEQUENCE for matroid constraint with rank oracle

---

Input: matroid $\mathcal{M}$, ground set N

$\quad b_1, \dots, b_{|N|} \leftarrow$ random permutation of N

$\quad r_i \leftarrow$ RANK$(\{b_1, \dots, b_i\})$, for all $i \in \{1, \dots, n\}$

$\quad a_i \leftarrow$ ith b_j s.t. $r_j - r_{j-1} = 1$

$\quad$ **return** $a_1, \dots, a_\ell$

With Algorithm 4 as the RANDOM SEQUENCE subroutine for ADAPTIVE SEQUENCING, we obtain the following result for matroid rank oracles (proof in the full version of the paper).

---

[2]More precisely, it allows p queries per step and the results depend on p, we consider the case of $p = \text{poly}(n)$.

THEOREM 4. *For any $\epsilon > 0$, there is an algorithm that obtains, w.p. $1 - o(1)$, a $1/2 - O(\epsilon)$ approximation with $O\left(\log(n)\log\left(\frac{k}{\epsilon}\right)\frac{1}{\epsilon^2}\right)$ adaptivity and steps of matroid rank queries.*

This gives $O(\log(n)\log(k))$ adaptivity and steps of independence queries with $1 - 1/e - \epsilon$ approximation for maximizing the multilinear relaxation and $1/2 - \epsilon$ approximation for maximizing a monotone submodular function under a matroid constraint. In particular, we get polylogarithmic depth on a PRAM machine with a rank oracle.

4.2　Matroid Independence Oracles

Recall that an independence oracle for $\mathcal{M}$ is an oracle which given $S \subseteq N$ answers whether $S \in \mathcal{M}$ or $S \notin \mathcal{M}$. We give a subroutine that requires $\tilde{O}(n^{1/2})$ steps of independence matroid oracle queries and show that $\tilde{\Omega}(n^{1/3})$ steps are necessary. Similar to the case of rank oracles we use a parallel algorithm from [KUW88] for constructing a base of a matroid that can be used as the RANDOM SEQUENCE subroutine while satisfying the random feasibility condition.

$\tilde{O}(\sqrt{n})$ *upper bound.* We use the algorithm from [KUW88] for constructing a base of a matroid.

Algorithm 5 Parallel RANDOM SEQUENCE for matroid constraint with independence oracle

---

Input: matroid $\mathcal{M}$, ground set N

　　$c \leftarrow 0, X \leftarrow N$
　　while $|N| > 0$ **do**
　　　　$b_1, \ldots, b_{|X|} \leftarrow$ random permutation of X
　　　　$i^\star \leftarrow \max\{i : \{a_1, \ldots, a_c\} \cup \{b_1, \ldots, b_i\} \in \mathcal{M}\}$
　　　　$a_{c+1}, \ldots, a_{c+i^\star} \leftarrow b_1, \ldots, b_{i^\star}$
　　　　$c \leftarrow c + i^\star$
　　　　$X \leftarrow \{a \in X : \{a_1, \ldots, a_c, a\} \in \mathcal{M}\}$
　　return $a_1, \ldots, a_c$

With Algorithm 5 as the RANDOM SEQUENCE subroutine for ADAPTIVE SEQUENCING, we obtain the following result with independence oracles. We defer the proof to the full version of the paper.

THEOREM 5. *There is an algorithm that obtains, w.p. $1 - o(1)$, a $1/2 - O(\epsilon)$ approximation with $O\left(\log(n)\log\left(\frac{k}{\epsilon}\right)\frac{1}{\epsilon^2}\right)$ adaptivity and $O\left(\sqrt{n}\log(n)\log\left(\frac{k}{\epsilon}\right)\frac{1}{\epsilon^2}\right)$ steps of independence queries.*

This gives $O(\log(n)\log(k))$ adaptivity and $\sqrt{n}\log(n)\log(k)$ steps of independence queries with $1 - 1/e - \epsilon$ approximation for maximizing the multilinear relaxation and $1/2 - \epsilon$ approximation for maximizing a monotone submodular function under a matroid constraint. In particular, even with independence oracles we get a sublinear algorithm in the PRAM model.

$\tilde{\Omega}(n^{1/3})$ *lower bound.* We show that there is no algorithm which obtains a constant approximation with less than $\tilde{\Omega}(n^{1/3})$ steps of independence queries, even for a cardinality function $f(S) = |S|$. We do so by using the same construction for a hard matroid instance as in [KUW88] used to show an $\tilde{\Omega}(n^{1/3})$ lower bound on the number of steps of independence queries for constructing a base

of a matroid. Although the matroid instance is the same, we use a different approach since the proof technique of [KUW88] does not hold in our case (see proof and discussion in the full version of the paper.).

THEOREM 6. *For any constant α, there is no algorithm with $\frac{n^{1/3}}{4\alpha\log^2 n} - 1$ steps of poly(n) matroid queries which, w.p. strictly greater than $n^{-\Omega(\log n)}$, obtains an α approximation for maximizing a cardinality function under a partition matroid constraint when given an independence oracle.*

To the best of our knowledge, the gap between the lower and upper bounds of $\tilde{Omega}(n^{1/3})$ and $\tilde{O}(n^{1/2})$ parallel steps for constructing a matroid basis given an independence oracle remains open since [KUW88]. Closing this gap for submodular maximization under a matroid constraint given an independence oracle is an interesting open problem that would also close the gap of [KUW88].

A　DISCUSSION ABOUT ADDITIONAL RESULTS

We discuss several cases for which our results and techniques generalize.

Cardinality constraint. We first discuss a generalization of ADAPTIVE SEQUENCING that is a $O(\log(n))$ adaptive algorithm that obtains a $1 - 1/e - O(\epsilon)$ approximation with probability $1 - o(1)$ for monotone submodular maximization under a cardinality constraint, which is the special case of a uniform matroid. Instead of sampling uniformly random subsets of X of size k/r as done in every iteration of the algorithm in [BRS19], it is possible to generate a single sequence and then add elements to S and discard elements from X in the same manner as ADAPTIVE SEQUENCING. We note that generating a random feasible sequence in parallel is trivial for a cardinality constraint k, one can simply pick k elements uniformly at random. Similarly, the elements we add to the solution are approximately locally optimal and we discard a constant fraction of elements at every round. A main difference is that for the case of a cardinality constraint, setting the threshold t to $t = (\mathsf{OPT} - f(S))/k$ is sufficient and, as shown in [BRS19], this threshold only needs a constant number of updates. Thus, for the case of a cardinality constraint, we obtain a $O(\log n)$ adaptive algorithm with a variant of the algorithm. In addition, the continuous greedy algorithm is not needed for a cardinality constraint since adding elements with marginal contribution which approximates $(\mathsf{OPT} - f(S))/k$ at every iteration guarantees a $1 - 1/e - \epsilon$ approximation.

Non-monotone functions. For maximizing a non-monotone submodular function under a cardinality constraint, similarly as for the monotone algorithm discussed above, we can also generate a single sequence instead of multiple random blocks of elements, as done in [BBS18].

Partition matroids with explicit representation. Special families of matroids, such as graphical and partition matroids, have explicit representations. We consider the case where a partition matroid is given as input to the algorithm not as an oracle but with its explicit representation, meaning the algorithm is given the parts $P_1, \ldots, P_m$

of the partition matroid and the number $p_1, \ldots, p_m$ of elements of each parts allowed by the matroid.

For the more general setting of packing constraints given to the algorithm as a collection of m linear constraints, as previously mentioned, [CQ19] develop a $O(\log^2(m)\log(n))$ adaptive algorithm that obtains with high probability a $1 - 1/e - \epsilon$ approximation, and has polylogarithmic depth on a PRAM machine for partition matroids.

For partition matroids, we obtain a $O\left(\log(n)\log\left(\frac{k}{\epsilon\lambda}\right)\frac{1}{\epsilon\lambda}\right)$ adaptive algorithm that, with probability $1 - \delta$, obtains a $1 - 1/e - O(\epsilon)$ approximation with $\lambda = O\left(\epsilon^2 \log^{-1}\left(\frac{1}{\delta}\right)\right)$. This algorithm also has polylogarithmic depth. This algorithm uses ACCELERATED CONTINUOUS GREEDY with the RANDOM SEQUENCE subroutine for rank oracles since a rank oracle for partition matroids can easily be constructed in polylogarithmic depth when given the explicit representation of the matroid. As mentioned in [CQ19], it is also possible to obtain a rounding scheme for partition matroids in polylogarithmic depth.

Intersection of P matroids. We formally analyze the more general constraint consisting of the intersection of P matroids in Appendix B.

B MISSING PROOFS FROM SECTION 2

B.1 Quasi-linear Query Complexity

The query complexity of ADAPTIVE SEQUENCING and ACCELERATED CONTINUOUS GREEDY can be improved from $O(nk\log(n)\log(k))$ to quasi-linear with $O(n\log(n)\log^2(k))$ queries if we allow

$$O(\log(n)\log^2(k))$$

rounds. This is done by finding $i^\star$ at every iteration of ADAPTIVE SEQUENCING by doing binary search of $i \in [\text{RANK}(\mathcal{M}(S, X))]$ instead of computing X_i for all i in parallel. Since there are at most k values of i, this decrease the query complexity of finding $i^\star$ from nk to $n\log k$, but increases the adaptivity by $\log k$.

An important property to be able to perform binary search is to have $|X_i|$ decreasing in i. We show this with the following lemma.

LEMMA 10. *At every iteration of ADAPTIVE SEQUENCING, $X_{i+1} \subseteq X_i$ for all $i < \text{RANK}(\mathcal{M}(S, X))$.*

PROOF. Assume $a \in X_{i+1}$. Thus, $S \cup \{a_1, \ldots, a_i\} + a \in \mathcal{M}$ and $f_{S \cup \{a_1, \ldots, a_i\}}(a) \geq t$. By the downward closed property of matroids, $S \cup \{a_1, \ldots, a_{i-1}\} + a \in \mathcal{M}$. By submodularity, $f_{S \cup \{a_1, \ldots, a_{i-1}\}}(a) \geq f_{S \cup \{a_1, \ldots, a_i\}}(a) \geq t$. We get that $a \in X_i$. □

COROLLARY 1. *If ADAPTIVE SEQUENCING finds $i^\star$ by doing binary search, then its query complexity is $O(n\log(n)\log^2(k))$ and its adaptivity is $O(\log(n)\log^2(k))$.*

B.2 From Expectation to High Probability for the Combinatorial Algorithm

We generalize ADAPTIVE SEQUENCING to obtain an algorithm called ADAPTIVE SEQUENCING++, described below, which achieves a $1/2 - \epsilon$ approximation with high probability, instead of in expectation. We note that this generalization is not needed when ADAPTIVE

SEQUENCING is used as a subroutine of ACCELERATED CONTINUOUS GREEDY for the $1 - 1/e - \epsilon$ result.

Algorithm 6 ADAPTIVE SEQUENCING++, ADAPTIVE SEQUENCING with high probability guarantee

---

Input: function f, feasibility constraint $\mathcal{M}$
  $S \leftarrow \emptyset, t \leftarrow \max_{a \in N} f(a)$
  for Δ iterations **do**
    $X \leftarrow N$
    while $X \neq \emptyset$ **do**
      for $j = 1$ to ρ **do (non-adaptivity and in parallel)**
        $a_1, \ldots, a_{\text{RANK}(\mathcal{M}(S, X))} \leftarrow$ RANDOM SEQUENCE$(\mathcal{M}(S, X))$
        $X_i \leftarrow \{a \in X : S \cup \{a_1, \ldots, a_i, a\} \in \mathcal{M}$ and $f_{S \cup \{a_1, \ldots, a_i\}}(a) \geq t\}$
        $i^\star \leftarrow \min\{i : |X_i| \leq (1 - \epsilon)|X|\}$
        $S^j \leftarrow S \cup \{a_1, \ldots, a_{i^\star}\}$
        $X^j \leftarrow X_{i^\star}$
        $v^j \leftarrow \frac{1}{i^\star}\sum_{\ell=1}^{i^\star} f_{S \cup \{a_1, \ldots, a_{\ell-1}\}}(a_\ell)$
      $j^\star \leftarrow \text{argmax}_{j \in [\rho]} v^j$
      $S \leftarrow S^j$
      $X \leftarrow X^j$
    $t \leftarrow (1 - \epsilon)t$
  return S

THEOREM 1. *For any $\epsilon > 0$, there is an $O\left(\log(n)\log\left(\frac{k}{\epsilon}\right)\frac{1}{\epsilon^2}\right)$ adaptive algorithm that obtains a $1/2 - O(\epsilon)$ approximation with probability $1 - o(1)$ for maximizing a monotone submodular function under a matroid constraint.*

PROOF. We set $\Delta = O\left(\frac{1}{\epsilon}\log\left(\frac{k}{\epsilon}\right)\right)$. Initially we have $t_i \leq \text{OPT}$. After Δ iterations of ADAPTIVE SEQUENCING, the final value of t is $t_f \leq (1 - \epsilon)^\Delta \text{OPT} = O\left(\frac{\epsilon}{k}\right)\text{OPT}$. We begin by adding dummy elements to S so that $|S| = k$, which enables pairwise comparisons between S and O. In particular, we consider S', which is S together with $\text{RANK}(\mathcal{M}) - |S|$ dummy elements $a_{|S|+1}, \ldots a_k$ such that, for any T, $f_T(a) = t_f$. Thus, by Lemma 2, for dummy elements a_i, $f_{S_{i-1}}(a_i) = t_f \geq (1 - \epsilon)\max_{a:S_{i-1}\cup a \in \mathcal{M}} f_{S_{i-1}}(a)$.

By Lemma 1, there are $O(\Delta\log(n)/\epsilon)$ iterations of the while-loop. Since each iteration of the while-loop is non-adaptive, ADAPTIVE SEQUENCING++ is $O(\Delta\log(n)/\epsilon)$ adaptive

Consider an iteration of the while-loop of ADAPTIVE SEQUENCING++. We first argue that for each inner-iteration j,

$$\sum_{i \in [i^\star]} f_{S_{i-1}}(a_i) \geq (1 - \epsilon)^2 i^\star t.$$

We first note that $\text{Pr}_{a_i}\left[f_{S \cup \{a_1, \ldots, a_{i-1}\}}(a_i) \geq t\right] \geq 1 - \epsilon$ by the definition of $i^\star$ and the random feasible sequence property. Let Y be the number of indices $i \leq i^\star$ such that $f_{S \cup \{a_1, \ldots, a_{i-1}\}}(a_i) \geq t$. By Chernoff bound, with $\mu = \mathbb{E}[Y] \geq (1 - \epsilon)i^\star$

$$\text{Pr}\left[Y \leq (1 - \epsilon)(1 - \epsilon)i^\star\right] \leq e^{-\epsilon^2(1-\epsilon)i^\star/2} \leq e^{-\epsilon^2(1-\epsilon)/2}.$$

Let $Z \leq \rho$ be the number of inner-iterations j such that $Y \geq (1 - \epsilon)(1 - \epsilon)i^\star$. By Chernoff bound, with $\mu = \mathbb{E}[Z] \geq (1 - e^{-\epsilon^2(1-\epsilon)/2})\rho$,

$$\text{Pr}\left[Z \leq \frac{1}{2}(1 - e^{-\epsilon^2(1-\epsilon)/2})\rho\right] \leq e^{(1-e^{-\epsilon^2(1-\epsilon)/2})\rho/8}.$$

Thus, with $\rho = O\left(\frac{1}{1-e^{-\epsilon^2}} \log\left(\frac{\Delta \log n}{\epsilon \delta}\right)\right)$, we have that with probability $1 - O\left(\epsilon\delta/(\Delta \log n)\right)$, there is at least one inner-iteration j such that $Y \geq (1-\epsilon)(1-\epsilon)i^\star$. Thus $\sum_{i \in [i^\star]} f_{S_{i-1}}(a_i) \geq (1-\epsilon)^2 i^\star t$. By Lemma 2,

$$\sum_{i \in [i^\star]} f_{S_{i-1}}(a_i) \geq (1-\epsilon)^3 \max_{a:S_{i-1} \cup a \in \mathcal{M}} f_{S_{i-1}}(a).$$

By a union bound, this holds over all iterations of the while-loop of ADAPTIVE SEQUENCING++ with probability $1 - \delta$ and we get that

$$\sum_{i \in [k]} f_{S_{i-1}}(a_i) \geq (1-\epsilon)^3 \max_{a:S_{i-1} \cup a \in \mathcal{M}} f_{S_{i-1}}(a).$$

Let $O = \{o_1, \ldots, o_k\}$ such that $\{a_1, \ldots, a_{i-1}, o_i\}$ is feasible for all i, which exists by the augmentation property of matroids. We conclude that with probability $1 - \delta$,

$$\begin{aligned}
f(S') &= \sum_{i \in [k]} \mathbb{E}[f_{S_{i-1}}(a_i)] \\
&\geq (1-\epsilon)^3 \max_{a:S_{i-1} \cup a \in \mathcal{M}} f_{S_{i-1}}(a) \\
&\geq (1-\epsilon)^3 \sum_{i \in [k]} \mathbb{E}[f_{S_{i-1}}(o_i)] \\
&\geq (1-\epsilon)^3 f_{S'}(O) \\
&\geq (1-\epsilon)^3 (\text{OPT} - f(S'))
\end{aligned}$$

and since

$$f(S) = f(S') - (\text{RANK}(\mathcal{M}) - |S|)t_f \geq f(S') - O(\epsilon)\text{OPT},$$

we conclude that $f(S) \geq (1/2 - O(\epsilon))\text{OPT}$. □

B.3 Intersection of Matroid Constraints

We consider constraint $\mathcal{M} = \cap_{i=1}^P \mathcal{M}_i$ which is the intersection of P matroids $\mathcal{M}_i$, i.e. $S \in \mathcal{M}$ if $S \in \mathcal{M}_i$ for all $i \leq P$. Similarly as for a single matroid constraint, we denote the size of the largest feasible set by k. We denote the rank of a set S with respect to matroid $\mathcal{M}_j$ by $\text{RANK}_j(S)$. We define $\text{SPAN}_j(S)$, called the *span* of S in $\mathcal{M}_j$ by:

$$\text{SPAN}_j(S) = \{a \in N : \text{RANK}_j(S \cup a) = \text{RANK}_j(S)\}$$

We will use the following claim.

CLAIM 1 (PROP. 2.2 IN [NWF78]). *If for* $\forall t \in [k]$ $\sum_{i=0}^{t-1} \sigma_i \leq t$ *and* $p_{i-1} \geq p_i$, *with* $\sigma_i, p_i \geq 0$ *then:*

$$\sum_{i=0}^{k-1} p_i \sigma_i \leq \sum_{i=0}^{k-1} p_i.$$

Similarly as for a single matroid, we give the approximation guaranteed obtained by a solution S with near-optimal marginal contributions for each $a \in S$.

LEMMA 11. *Assume that* $S = \{a_1, \ldots, a_k\}$ *such that*

$$f_{S_{i-1}}(a_i) \geq (1-\epsilon) \max_{a:S_{i-1} \cup a \in \mathcal{M}} f_{S_{i-1}}(a)$$

where $S_i = \{a_1, \ldots, a_i\}$. *Then, if* $\mathcal{M}$ *is the intersection of* P *matroids, we have*

$$f(S) \geq \left(\frac{1}{P+1} - O(\epsilon)\right)\text{OPT}.$$

PROOF. Since S_i and O are independent sets in $\mathcal{M}_j$ we have:

$$\text{rank}_j(\text{SPAN}_j(S_i) \cap O) = |\text{SPAN}_j(S_i) \cap O| \leq |\text{SPAN}_j(S_i)| = |S_i| \leq i$$

Define $U_i = \cup_{j=1}^P \text{SPAN}_j(S_i)$, to be the set of elements which are not part of the maximization at index $i + 1$ of the procedure, and hence cannot give value at that stage. We have:

$$|U_i \cap O| = |(\cup_{j=1}^P \text{SPAN}_j(S_i)) \cap O| \leq \sum_{j=1}^P |\text{SPAN}_j(S_i) \cap O| \leq P \cdot i$$

Let $V_i = (U_i \setminus U_{i-1}) \cap O$ be the elements of O which are not part of the maximization at index i, but were part of the maximization at index $i - 1$. If $a \in V_i$ then it must be that

$$(1-\epsilon)f_{S_k}(a)(1-\epsilon) \leq f_{S_{i-1}}(a) \leq \max_{b:S_{i-1} \cup b \in \mathcal{M}} f_{S_{i-1}}(b)$$

where the first inequality is due to submodularity of f. Hence, we can upper bound:

$$\begin{aligned}
\sum_{o \in O \setminus S_k} f_{S_k}(o) &\leq \sum_{i=1}^k \sum_{o \in V_i} \max_{a:S_{i-1} \cup a \in \mathcal{M}} f_{S_{i-1}}(a) \\
&= \sum_{i=1}^k |V_i| \max_{a:S_{i-1} \cup a \in \mathcal{M}} f_{S_{i-1}}(a) \\
&\leq P \sum_{i=1}^k \max_{a:S_{i-1} \cup a \in \mathcal{M}} f_{S_{i-1}}(a)
\end{aligned}$$

where the last inequality uses $\sum_{t=1}^i |V_t| = |U_i \cap O| \leq Pi$ and the claim due to 1. Together with $\text{OPT} \leq f(O \cup S_k) \leq f(S_k) + \sum_{o \in O \setminus S_k} f_{S_k}(o)$ and $f_{S_{i-1}}(a_i) \geq (1-\epsilon)\max_{a:S_{i-1} \cup a \in \mathcal{M}} f_{S_{i-1}}(a)$ we get:

$$f(S) \geq \left(\frac{1}{P+1} - O(\epsilon)\right)\text{OPT}.$$

as required. □

Since Lemma 2 only uses the downward closed property of $\mathcal{M}$ and since intersections of matroids are downward closed, ADAPTIVE SEQUENCING++ obtains a solution S with near-optimal marginal contributions for each $a_i \in S = \{a_1, \ldots, a_k\}$. Combined with the previous lemma, we obtain the result for intersections of matroids.

THEOREM 7. *For any* $\epsilon > 0$, ADAPTIVE SEQUENCING++ *is an* $O\left(\log(n) \log\left(\frac{k}{\epsilon}\right)\frac{1}{\epsilon^2}\right)$ *adaptive algorithm that obtains a* $1/(P+1) - O(\epsilon)$ *approximation with probability* $1 - o(1)$ *for maximizing a monotone submodular function under the intersection of* P *matroids.*

PROOF. The first part of the of the proof follows similarly as the proof for Theorem 3 by using Lemma 2, which also hold for intersections of matroids, to obtain the near-optimal marginal contributions of each $a_i \in S$ with probability $1 - o(1)$:

$$\sum_{i \in [i^\star]} f_{S_{i-1}}(a_i) \geq (1-\epsilon)^3 \max_{a:S_{i-1} \cup a \in \mathcal{M}} f_{S_{i-1}}(a).$$

We then combine this with Lemma 11 to obtain the $1/(P+1) - O(\epsilon)$ approximation with probability $1 - o(1)$. □

C MISSING PROOFS FROM SECTION 3

Discussion on constant step size λ. In contrast to the continuous greedy, the accelerated continuous greedy uses *constant* steps sizes λ to guarantee low adaptivity. The challenge with using constant λ is that $F_{\mathbf{x}}(\lambda S)$ is arbitrarily low with $S := \operatorname{argmax}_{T \in \mathcal{M}} \sum_{a \in T} g(a)$ due to the overlap in value of elements a with high individual value $g(a)$. For example, consider ground set $N = A \cup B$ with

$$f(S) = \min(\log n, |S \cap A|) + |S \cap B|,$$

$\mathbf{x} = \mathbf{0}$ and $S = A$. With $\lambda = 1/n$, we note that sampling $R \sim \lambda A$ where R independently contains each element in S with probability $1/n$ gives $|R| \leq \log n$ with high probability and we get $F_{\mathbf{x}}(\lambda A) = (1 - o(1))|A|$, which is near-optimal for a set of size $|A|$. However, with constant λ, then sampling $R \sim \lambda A$ gives $|R| > \log n$ with high probability. Thus $F_{\mathbf{x}}(\lambda A) \leq \log(n)$ which is arbitrarily far from optimal for $|A| = |B| >> \log n$ since $F_{\mathbf{x}}(\lambda B) = \lambda |B|$.

Lemma 5. *For a given matroid $\mathcal{M}$ assume that* ADAPTIVE SEQUENCING *outputs $S \in \mathcal{M}$ s.t.*

$$\mathbb{E}_S [F_{\mathbf{x}}(\lambda S)] \geq (1 - \epsilon)\lambda(\text{OPT} - F(\mathbf{x}))$$

at every iteration of ACCELERATED CONTINUOUS GREEDY. *Then* ACCELERATED CONTINUOUS GREEDY *outputs $\mathbf{x} \in P(\mathcal{M})$ s.t.*

$$\mathbb{E}[F(\mathbf{x})] \geq (1 - 1/e - \epsilon)\text{OPT}.$$

PROOF. First, $\mathbf{x} \in P$ since it is a convex combinations of λ^{-1} vectors $\mathbf{1}S$ with $S \in \mathcal{M}$. Next, let $\mathbf{x}_i$ denote the solution $\mathbf{x}$ at the ith iteration of ACCELERATED CONTINUOUS GREEDY. The algorithm increases the value of the solution $\mathbf{x}$ by at least $(1-\epsilon)\cdot\lambda\cdot(\text{OPT} - F(\mathbf{x}))$ at every iteration. Thus,

$$F(\mathbf{x}_i) \geq F(\mathbf{x}_{i-1}) + (1 - \epsilon) \cdot \lambda \cdot (\text{OPT} - F(\mathbf{x}_{i-1})).$$

Next, we show by induction on i that

$$F(\mathbf{x}_i) \geq \left(1 - (1 - (1 - \epsilon)\lambda)^i\right) \text{OPT}.$$

Observe that

$$
\begin{aligned}
F(\mathbf{x}_i) &\geq F(\mathbf{x}_{i-1}) + (1 - \epsilon)\lambda (\text{OPT} - F(\mathbf{x}_{i-1})) \\
&= (1 - \epsilon)\lambda\text{OPT} + (1 - (1 - \epsilon)\lambda) F(\mathbf{x}_{i-1}) \\
&\geq (1 - \epsilon)\lambda\text{OPT} + (1 - (1 - \epsilon)\lambda) \left(1 - (1 - (1 - \epsilon)\lambda)^{i-1}\right) \text{OPT} \\
&= \left(1 - (1 - (1 - \epsilon)\lambda)^i\right) \text{OPT}
\end{aligned}
$$

Thus, with $i = \lambda^{-1}$, we return solution $\mathbf{x} = \mathbf{x}_{\lambda^{-1}}$ such that

$$F(\mathbf{x}) \geq \left(1 - (1 - (1 - \epsilon)\lambda)^{\lambda^{-1}}\right) \text{OPT}.$$

Next, since $1 - x \leq e^{-x}$ for all $x \in \mathbb{R}$,

$$(1 - (1 - \epsilon)\lambda)^{\lambda^{-1}} \leq \left(e^{-(1-\epsilon)\lambda}\right)^{\lambda^{-1}} = e^{-(1-\epsilon)}.$$

We conclude that

$$
\begin{aligned}
F(\mathbf{x}) \geq \left(1 - e^{-(1-\epsilon)}\right) \text{OPT} &= \left(1 - \frac{e^\epsilon}{e}\right) \text{OPT} \\
&\geq \left(1 - \frac{1 + 2\epsilon}{e}\right) \text{OPT} \\
&\geq \left(1 - \frac{1}{e} - \epsilon\right) \text{OPT}
\end{aligned}
$$

where the second inequality is since $e^x \leq 1 + 2x$ for $0 < x < 1$. □

Lemma 6. *Let $\mathcal{M}$ be a matroid, then for any feasible sets $S = \{a_1, \ldots, a_k\}$ and O of size k, there exists an ordering of $O = \{o_1, \ldots, o_k\}$ where for all $i \in [k]$, $S_i \cup O_{i+1:k} \in \mathcal{M}$ and $S_i \cap O_{i+1:k} = \emptyset$.*

PROOF. The proof is by reverse induction. For $i = k$, we have $S_i \cup O_{i+1:k} = S_k = S \in \mathcal{M}$ by Lemma 2. Consider $i < k$ and assume that $S_{i+1} \cup O_{i+2:k} \in \mathcal{M}$ for some ordering $o_{i+2}, \ldots, o_k$ of $O_{i+2:k}$. By the downward closed property of matroids, $S_i \cup O_{i+2:k} \in \mathcal{M}$. By the augmentation property of matroids, there exists $o_{i+1} \in O \setminus (S_i \cup O_{i+2:k})$ such that $S_i \cup O_{i+2:k} + o_{i+1} = S_i \cup O_{i+1:k} \in \mathcal{M}$. □

THEOREM 2. *For any $\epsilon > 0$ ACCELERATED CONTINUOUS GREEDY makes $O\left(\log(n)\log\left(\frac{k}{\epsilon^2}\right)\frac{1}{\epsilon^2}\right)$ adaptive rounds and obtains a $1 - 1/e - O(\epsilon)$ approximation in expectation for maximizing a monotone submodular function under a matroid constraint.*

PROOF. We use step size

$$\lambda = O(\epsilon)$$

for ACCELERATED CONTINUOUS GREEDY and

$$\Delta = O\left(\frac{1}{\epsilon}\log\left(\frac{k}{\epsilon\lambda}\right)\right)$$

outer-iterations for ADAPTIVE SEQUENCING. Thus, by Lemma 1, the adaptivity is

$$O\left(\frac{\Delta \log n}{\lambda\epsilon}\right) = O\left(\log(n)\log\left(\frac{k}{\epsilon^2}\right)\frac{1}{\epsilon^2}\right).$$

By Lemma 8, we have

$$\mathbb{E}[F_{\mathbf{x}}(\delta S)] \geq (1 - O(\epsilon))\lambda(\text{OPT} - F(\mathbf{x}))$$

at every iteration i. Combining with Lemma 5, we obtain that

$$\mathbb{E}[F(\mathbf{x})] \geq (1 - e^{-1} - O(\epsilon))\text{OPT}.$$

It remains to round the solution $\mathbf{x}$. We note that there exist rounding schemes with arbitrarily small loss that are independent of the function f [CVZ09, VCZ11] (so they do not perform any queries to f). The set S we obtain from rounding the solution $\mathbf{x}$ returned by ACCELERATED CONTINUOUS GREEDY with these techniques is thus a $1 - 1/e - O(\epsilon)$ approximation with no additional adaptivity. □

Lemma 9. *Assume that* ADAPTIVE SEQUENCING *outputs $S \in \mathcal{M}$ s.t. $F_{\mathbf{x}}(\lambda S) \geq \alpha_i \lambda(\text{OPT} - F(\mathbf{x}))$ at every iteration i of* ACCELERATED CONTINUOUS GREEDY *and that $\lambda \sum_{i=1}^{\lambda^{-1}} \alpha_i \geq 1 - \epsilon$. Then* ACCELERATED CONTINUOUS GREEDY *outputs $\mathbf{x} \in P(\mathcal{M})$ s.t. $F(\mathbf{x}) \geq (1 - 1/e - \epsilon)\text{OPT}$.*

PROOF. First, $\mathbf{x} \in P$ since it is a convex combinations of λ^{-1} vectors $\mathbf{1}_S \in \mathcal{M}$. Next, let $\mathbf{x}_i$ denote the solution $\mathbf{x}$ at the ith iteration of ACCELERATED CONTINUOUS GREEDY. The algorithm increases the value of the solution $\mathbf{x}$ by at least $\alpha_i \cdot \lambda \cdot (\text{OPT} - F(\mathbf{x}))$ at every iteration. Thus,

$$F(\mathbf{x}_i) \geq F(\mathbf{x}_{i-1}) + \alpha_i \cdot \lambda \cdot (\text{OPT} - F(\mathbf{x}_{i-1})).$$

Next, we show by induction on i that

$$F(\mathbf{x}_i) \geq \left(1 - \prod_{j=1}^{i} (1 - \lambda\alpha_j)\right)\text{OPT}.$$

Observe the following, where the first inequality is by the assumption of the lemma, the second by the inductive hypothesis, and the equalities by rearranging the terms.

$$F(\mathbf{x}_i) \geq F(\mathbf{x}_{i-1}) + \alpha_i \lambda \left(\mathsf{OPT} - F(\mathbf{x}_{i-1})\right)$$
$$= \alpha_i \lambda \mathsf{OPT} + (1 - \alpha_i \lambda) F(\mathbf{x}_{i-1})$$
$$\geq \alpha_i \lambda \mathsf{OPT} + (1 - \alpha_i \lambda) \left(1 - \prod_{j=1}^{i-1} \left(1 - \lambda \alpha_j\right)\right) \mathsf{OPT}$$
$$= \alpha_i \lambda \mathsf{OPT} + \left(1 - \alpha_i \lambda - \prod_{j=1}^{i} \left(1 - \lambda \alpha_j\right)\right) \mathsf{OPT}$$
$$= \left(1 - \prod_{j=1}^{i} \left(1 - \lambda \alpha_j\right)\right) \mathsf{OPT}$$

Thus, with $i = \lambda^{-1}$, we return solution $\mathbf{x} = \mathbf{x}_{\lambda^{-1}}$ such that

$$F(\mathbf{x}) \geq \left(1 - \prod_{j=1}^{\lambda^{-1}} \left(1 - \lambda \alpha_j\right)\right) \mathsf{OPT}.$$

Since $1 - x \leq e^{-x}$ for all $x \in \mathbb{R}$,

$$1 - \prod_{j=1}^{\lambda^{-1}} \left(1 - \lambda \alpha_j\right) \geq 1 - \prod_{j=1}^{\lambda^{-1}} e^{-\lambda \alpha_j} = 1 - e^{-\lambda \sum_{j=1}^{\lambda^{-1}} \alpha_j}$$
$$\geq 1 - e^{-(1-\epsilon)} \geq 1 - e^{-1} - 2\epsilon/e \geq 1 - e^{-1} - \epsilon$$

where the second inequality is since $e^x \leq 1 + 2x$ for $0 < x < 1$. □

Theorem 3. Accelerated Continuous Greedy *is an algorithm with adaptivity* $O\left(\log(n) \log\left(\frac{k}{\epsilon\lambda}\right) \frac{1}{\epsilon\lambda}\right)$ *that, with probability* $1 - \delta$, *obtains a* $1 - 1/e - O(\epsilon)$ *approximation for maximizing a monotone submodular function under a matroid constraint, with step size* $\lambda = O\left(\epsilon^2 \log^{-1}\left(\frac{1}{\delta}\right)\right)$.

Proof. We use $\Delta = O\left(\frac{1}{\epsilon} \log\left(\frac{k}{\epsilon\lambda}\right)\right)$ outer-iterations for Adaptive Sequencing. Thus, by Lemma 1, the adaptivity is $O\left(\frac{\Delta \log n}{\lambda\epsilon}\right) = O\left(\log(n) \log\left(\frac{k}{\epsilon\lambda}\right) \frac{1}{\epsilon\lambda}\right)$.

By Lemma 8, we have $F_{\mathbf{x}}(\delta S) \geq \alpha_i \lambda (\mathsf{OPT} - F(\mathbf{x}))$ at every iteration i with $\mathbb{E}[\alpha_i] \geq 1 - \epsilon'$ where $\epsilon' = O(\epsilon)$. By a Chernoff bound with $\mathbb{E}[\lambda \sum_{i \in \lambda^{-1}} \alpha_i] \geq 1 - \epsilon'$,

$$\Pr\left[\lambda \sum_{i \in [\lambda^{-1}]} \alpha_i < (1 - \epsilon)(1 - \epsilon')\right] \leq e^{-\epsilon^2(1-\epsilon')\lambda^{-1}/2}.$$

Thus, with probability $p = 1 - e^{-\epsilon^2(1-\epsilon')\lambda^{-1}/2}$, $\lambda \sum_{i \in [\lambda^{-1}]} \alpha_i \geq 1 - \epsilon - \epsilon'$. By Lemma 9, we conclude that w.p. p, $F(\mathbf{x}) \geq (1 - e^{-1} - (\epsilon + \epsilon'))\mathsf{OPT}$. With step size $\lambda = O(\epsilon^2/\log(1/\delta))$, we get that with probability $1 - \delta$, $F(\mathbf{x}) \geq (1 - e^{-1} - O(\epsilon))\mathsf{OPT}$.

It remains to round the solution $\mathbf{x}$. We note that there exist rounding schemes with arbitrarily small loss that are independent of the function f [CVZ09, VCZ11] (so they do not perform any queries to f). The set S we obtain from rounding the solution $\mathbf{x}$ returned by Accelerated Continuous Greedy with these techniques is thus a $1 - 1/e - O(\epsilon)$ approximation with no additional adaptivity. □

REFERENCES

[BBS18] Eric Balkanski, Adam Breuer, and Yaron Singer. Non-monotone submodular maximization in exponentially fewer iterations. *NIPS*, 2018.

[BRS19] Eric Balkanski, Aviad Rubinstein, and Yaron Singer. An exponential speedup in parallel running time for submodular maximization without loss in approximation. *SODA*, 2019.

[BS18a] Eric Balkanski and Yaron Singer. The adaptive complexity of maximizing a submodular function. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1138–1151. ACM, 2018.

[BS18b] Eric Balkanski and Yaron Singer. Approximation guarantees for adaptive sampling. In *International Conference on Machine Learning*, pages 393–402, 2018.

[CCPV07] Gruia Călinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a submodular set function subject to a matroid constraint (extended abstract). In *Integer Programming and Combinatorial Optimization, 12th International IPCO Conference, Ithaca, NY, USA, June 25-27, 2007, Proceedings*, pages 182–196, 2007.

[CJV15] Chandra Chekuri, TS Jayram, and Jan Vondrák. On multiplicative weight updates for concave and submodular function maximization. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*, pages 201–210. ACM, 2015.

[CQ19] Chandra Chekuri and Kent Quanrud. Submodular function maximization in parallel via the multilinear relaxation. *SODA*, 2019.

[CVZ09] Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Dependent randomized rounding for matroid polytopes and applications. *arXiv preprint arXiv:0909.4348*, 2009.

[EN19] Alina Ene and Huy L Nguyen. Submodular maximization with nearly-optimal approximation and adaptivity in nearly-linear time. *SODA*, 2019.

[ENV18] Alina Ene, Huy L Nguyen, and Adrian Vladu. Submodular maximization with packing constraints in parallel. *arXiv preprint arXiv:1808.09987*, 2018.

[FMZ18] Matthew Fahrbach, Vahab Mirrokni, and Morteza Zadimoghaddam. Non-monotone submodular maximization with nearly optimal adaptivity complexity. *arXiv preprint arXiv:1808.06932*, 2018.

[FMZ19] Matthew Fahrbach, Vahab Mirrokni, and Morteza Zadimoghaddam. Submodular maximization with optimal approximation, adaptivity and query complexity. *SODA*, 2019.

[KUW88] Richard M. Karp, Eli Upfal, and Avi Wigderson. The complexity of parallel search. *J. Comput. Syst. Sci.*, 36(2):225–253, 1988.

[NW78] George L Nemhauser and Laurence A Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Mathematics of operations research*, 3(3):177–188, 1978.

[NWF78] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical Programming*, 14(1):265–294, 1978.

[VCZ11] Jan Vondrák, Chandra Chekuri, and Rico Zenklusen. Submodular function maximization via the multilinear relaxation and contention resolution schemes. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 783–792. ACM, 2011.

[Von08] Jan Vondrák. Optimal approximation for the submodular welfare problem in the value oracle model. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 67–74, 2008.