Approximation Guarantees for Adaptive Sampling

Eric Balkanski [1] **Yaron Singer** [1]

Abstract

In this paper we analyze an adaptive sampling approach for submodular maximization. Adaptive sampling is a technique that has recently been shown to achieve a constant factor approximation guarantee for submodular maximization under a cardinality constraint with exponentially fewer adaptive rounds than any previously studied constant factor approximation algorithm for this problem. Adaptivity quantifies the number of sequential rounds that an algorithm makes when function evaluations can be executed in parallel and is the parallel running time of an algorithm, up to low order terms. Adaptive sampling achieves its exponential speedup at the expense of approximation. In theory, it is guaranteed to produce a solution that is a $1/3$ approximation to the optimum. Nevertheless, experiments show that adaptive sampling techniques achieve far better values in practice. In this paper we provide theoretical justification for this phenomenon. In particular, we show that under very mild conditions of *curvature* of a function, adaptive sampling techniques achieve an approximation arbitrarily close to $1/2$ while maintaining their low adaptivity. Furthermore, we show that the approximation ratio approaches 1 in direct relationship to a *homogeneity* property of the submodular function. In addition, we conduct experiments on real data sets in which the curvature and homogeneity properties can be easily manipulated and demonstrate the relationship between approximation and curvature, as well as the effectiveness of adaptive sampling in practice.

1. Introduction

In machine learning, many fundamental quantities we care to optimize such as entropy, diversity, coverage, diffusion, and clustering are submodular functions. For the canonical problem of maximizing a non-decreasing submodular function under a cardinality constraint k, the celebrated greedy algorithm which iteratively adds elements whose marginal contribution is largest is known to achieve a $1-1/e$ approximation (Nemhauser et al., 1978) which is tight unless the algorithm uses exponentially-many queries in the size of the ground set n (Nemhauser & Wolsey, 1978).

Although the simple greedy algorithm achieves an optimal approximation guarantee, it is highly *adaptive*. Informally, the adaptivity of an algorithm is the number of sequential rounds it requires when polynomially-many function evaluations can be executed in parallel in each round. The adaptivity of the greedy algorithm is k since it sequentially adds elements in k rounds, making linearly many function evaluations in each round to evaluate the marginal contribution of every element to the set of elements selected in the previous rounds. In general, $k \in \Omega(n)$ and the adaptivity, as well as the parallel runtime, of the greedy algorithm is hence *linear* in the size of the data.

The concept of adaptivity is generally well-studied in multiple areas of computer science as algorithms with low adaptivity lead to algorithms that can be parallelized efficiently (see Section 6 for further discussion). These areas include sorting and selection (Valiant, 1975; Cole, 1988; Braverman et al., 2016), communication complexity (Papadimitriou & Sipser, 1984; Duris et al., 1984; Nisan & Widgerson, 1991), multi-armed bandits (Agarwal et al., 2017), sparse recovery (Haupt et al., 2009a; Indyk et al., 2011; Haupt et al., 2009b), and property testing (Canonne & Gur, 2017; Buhrman et al., 2012; Chen et al., 2017).

Since the greedy algorithm has linear adaptivity and the size of the ground set n can be large, a natural question is whether constant factor approximations with lower adaptivity are achievable. Somewhat surprisingly, until very recently $\Omega(n)$ was the best known adaptivity required for a constant factor approximation to maximizing a monotone submodular maximization under a cardinality constraint.

In recent work (Balkanski & Singer, 2018) introduce an *adaptive sampling* technique for maximizing monotone submodular functions under a cardinality constraint. This technique produces an algorithm that is $\mathcal{O}(\log n)$-adaptive and achieves an approximation arbitrarily close to $1/3$. Further-

[1]Harvard University. Correspondence to: Eric Balkanski <ericbalkanski@g.harvard.edu>, Yaron Singer <yaron@seas.harvard.edu>.

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

more, this is tight in the sense that no algorithm can achieve a constant factor approximation with $\tilde{o}(\log n)$ rounds.

Despite this exponential improvement in adaptivity, the approximation ratio suffers. In experiments however, it seems that adaptive sampling does substantially better than $1/3$ and in some cases comparable to those of the greedy algorithm that uses $\mathcal{O}(n)$ rounds. Ideally, if we can characterize the settings in which approximation guarantees for adaptive sampling are better than $1/3$, these techniques could be implemented and dramatically reduce the parallel running time of applications that rely on large scale computing.

Why does adaptive sampling perform so well in practice?

In this paper we use the standard notion of *curvature* to reason about the strong performance of adaptive sampling. Curvature is a well-studied concept in the context of submodular optimization (Conforti & Cornuéjols, 1984; Vondrák, 2010; Iyer & Bilmes, 2013; Iyer et al., 2013; Sviridenko et al., 2015; Balkanski et al., 2016). Recall that a function $f : 2^N \to \mathbb{R}$ has curvature κ if $f_S(a) \geq (1-\kappa)f(a)$ for all S and $a \notin S$. Our main result in this paper is that even under very mild conditions of curvature on the function, adaptive sampling achieves an approximation guarantee that is arbitrarily close to $1/2$ in $\mathcal{O}(\log n)$ rounds. In particular we show:

- An approximation arbitrarily close to $\max(1-\kappa, 1/2)$ in $\mathcal{O}\left(\frac{\log n}{1-\kappa}\right)$ adaptive rounds if the function has bounded curvature $\kappa < 1$,
- An approximation arbitrarily close to $1 - \frac{\mu}{2\mu+1}$ for a μ-homogeneous function with bounded curvature,
- A tradeoff between the approximation guarantee and the number of adaptive rounds of the algorithm,
- A tight lower bound of $\log n$ adaptive rounds, up to lower order terms, to obtain a $1/2$ approximation for functions with bounded curvature $\kappa < 1$,
- Experiments over two real-world datasets demonstrating the effectiveness of adaptive sampling in practice and the effect of curvature.

The homogeneity condition, which we introduce to further improve the approximation guarantee, resembles the large market assumption in mechanism design, e.g. (Bei et al., 2012; Anari et al., 2014; Balkanski & Hartline, 2016), in the sense that it bounds the impact of a single element on the overall objective.

We consider a simple, yet useful, operation to alter general submodular functions into functions with bounded curvature, which we call curvaturing and which was previously used in Iyer et al. (2013). This technique can be interpreted as an analogue to regularization in convex optimization.

Interestingly, we use curvaturing to obtain both upper and lower bounds. We use curvaturing on general submodular functions to extend the $1/2$ approximation guarantee to functions with unbounded curvature, at the cost of an additional additive term in the approximation. We also give a reduction from lower bounds for functions with bounded curvature to lower bounds for general submodular functions using this same technique of curvaturing. With this reduction, a previous lower bound on the number of rounds needed to obtain a constant approximation implies the new lower bound for functions with bounded curvature.

Paper organization. We first cover preliminary definitions in Section 2. The ADAPTIVE-SAMPLING algorithm is presented and analyzed in Section 3. We then give a lower bound in Section 4. The experiments are in Section 5. Finally, related work is in Section 6.

2. Preliminaries

A function $f : 2^N \to \mathbb{R}_+$ is *submodular* if the marginal contributions $f_S(a) := f(S \cup a) - f(S)$ of an element $a \in N$ to a set $S \subseteq N$ are diminishing, i.e., $f_S(a) \geq f_T(a)$ for all $a \in N \setminus T$ and $S \subseteq T$, and is *monotone* if $f(S) \leq f(T)$ for all $S \subseteq T$. A submodular function f is also *subadditive*, meaning $f(S \cup T) \leq f(S) + f(T)$ for all $S \subseteq T$ .We assume that f is non-negative, i.e., $f(S) \geq 0$ for all $S \subseteq N$, which is standard.

Informally, the *adaptivity* of an algorithm is the number of sequential rounds of queries it makes, where every round allows for polynomially-many parallel queries.

Definition. *Given a function f, an algorithm is r-adaptive if every query $f(S)$ for the value of a set S occurs at a round $i \in [r]$ such that S is independent of the values $f(S')$ of all other queries at round i.*

A submodular function f has *curvature* κ, $0 \leq \kappa \leq 1$, if

$$f_S(a) \geq (1-\kappa)f(a)$$

for all sets S and elements $a \notin S$. A useful corollary is that $f_S(T) \geq (1-\kappa)f(T)$ for all non intersecting sets $|S \cap T| = 0$. Given a function f, the *curvaturing* operation produces $\tilde{f}(S) = \kappa f(S) + (1-\kappa)|S|$. Even though f might have unbounded curvature, $\tilde{f}$ has curvature κ when f is normalized such that $\max_{a \in N} f(a) \leq 1$.

Finally, a function f is μ-homogeneous, $\mu \geq 0$, if

$$f(a) \leq (1+\mu)\frac{\texttt{OPT}}{k}$$

for all $a \in N$, where $\texttt{OPT} := \max_{S:|S|\leq k} f(S)$ is the value of the optimal solution.

3. The Algorithm

In this section, we present and analyze the ADAPTIVE-SAMPLING algorithm. By its design, this algorithm terminates after $\mathcal{O}(\log n)$ rounds and its approximation ratio is arbitrarily close to $1/3$. We show that if the function respects a mild curvature condition such that it has bounded curvature $\kappa < 1$, the approximation ratio of the algorithm is arbitrarily close to $\max(1-\kappa, 1/2)$, in $\mathcal{O}\left(\frac{\log n}{1-\kappa}\right)$ rounds. In addition, if the function is μ-homogeneous then the approximation ratio of the algorithm is further improved to being arbitrarily close to $1 - \frac{\mu}{2\mu+1}$.

Description of the algorithm. The ADAPTIVE-SAMPLING algorithm is a generalization of the algorithm in Balkanski & Singer (2018) designed to achieve superior approximation guarantees for bounded curvature. The algorithm maintains two solutions X and S, initialized to the empty set and the ground set N respectively. At every round, the algorithm either adds $\frac{k}{r}$ elements to X or discards from S a constant fraction of its remaining elements. The algorithm terminates when $|X| = k$ or alternatively when sufficiently many elements have been discarded to get $|X \cup S| \leq k$. Thus, with $r = \mathcal{O}(\log n)$, the algorithm has at most logarithmic many rounds. The algorithm is formally described below.

Algorithm 1 ADAPTIVE-SAMPLING

input threshold Δ, approximation α, samples m, rounds r
  Initialize $X \leftarrow \emptyset, S \leftarrow N$
  while $|X| < k$ **and** $|X \cup S| > k$ **do**
    update $\mathcal{D}$ to be uniform over subsets of S of size $\frac{k}{r}$
    $R \leftarrow \text{argmax}_{R \in \{R_i \sim \mathcal{D}\}_{i=1}^{m}} f_X(R)$
    $M \leftarrow$ top $\frac{k}{r}$ valued elements a with respect to $f_X(a)$
    if $\max\{f_X(R), f_X(M)\} \geq \frac{\alpha}{r}\texttt{OPT}$ **then**
      add $\text{argmax}\{f_X(R), f_X(M)\}$ to X, discard it from S
    else
      discard $\{a : \mathbb{E}_{R \sim \mathcal{D}}\left[f_{X \cup R \setminus \{a\}}(a)\right] < \Delta\}$ from S
  return X if $|X| = k$, or $X \cup S$ otherwise

Algorithm 1 generalizes the adaptive sampling algorithm in Balkanski & Singer (2018) by not only considering the best sample R when adding elements to X, but also the set M of top k/r elements a with largest contribution $f_X(a)$. This generalization is needed to obtain, by a simple argument about curvature, the $1-\kappa$ term in the approximation.

Algorithm 1 is an idealized version of the algorithm since we cannot exactly compute expectations and $\texttt{OPT}$ is unknown. In practice, the expectations can be estimated arbitrarily well by sampling and the algorithm can be executed multiple times with different guesses for $\texttt{OPT}$. The full algorithm is described in Appendix B.1. For readability we present the analysis of the idealized version as above which easily extends to the full algorithm, as shown in Appendix B.3.

Good and bad optimal elements. The key idea in analyzing the approximation ratio of adaptive sampling as a function of curvature requires partitioning the elements in the optimal solution O into *good optimal elements* and *bad optimal elements*, as we now define. The good and bad optimal elements play complementary roles in the analysis. The good optimal elements O^+ allow analyzing the approximation ratio in terms of curvature and bad optimal elements O^- enable bounding the value lost in terms of homogeneity.

Definition 1. *Let X be the set in* ADAPTIVE-SAMPLING *when the algorithm terminates and $\epsilon > 0$. Given some arbitrary ordering on the elements in O s.t. $O = \{o_1, \ldots, o_k\}$, for every $i \in [k]$, let $O_i = \{o_1, \ldots, o_i\}$. The set of good optimal elements O^+ is the set of elements in O whose marginal contribution to $X \cup O_{j-1}$ exceeds $(1+\epsilon)\Delta$, i.e. $O^+ := \{o_j \in O : f_{X \cup O_{j-1}}(o_j) \geq (1+\epsilon)\Delta\}$. The set of bad optimal elements is $O^- = O \setminus O^+$.*

3.1. Curvature

The analysis of the approximation ratio requires bounding the value of the set of elements discarded S^- from the optimal solution in two major steps:

1. We first bound the value $f(S^- \cap O^+)$ of good optimal elements that are discarded by $\frac{1}{1-\kappa} f_X(S^- \cap O^+)$. We then bound $f_X(S^- \cap O^+)$ by $|O^+ \cap S^-| \frac{\alpha \texttt{OPT}}{k}$. It then follows that $f(S^- \cap O^+) \leq \frac{\alpha}{1-\kappa}\texttt{OPT}$, which is arbitrarily bad as the curvature increases;

2. A second important step in the analysis is bounding $|O^+ \cap S^-|$ by $\mathbb{E}_{R \sim \mathcal{D}}\left[f(R)\right]/\Delta$. The partition of O into O^+ and O^- is what makes this step possible as $|O \cap S^-|$ can be arbitrarily close to k in general. The analysis distinguishes between elements in O^+ that must have large value and elements in S^- that must have small value to improve the bound on $|O^+ \cap S^-|$.

Lemma 1. *Let f be a monontone submodular function with curvature κ and r_d be the number of rounds where elements with contribution less than Δ are discarded, then w.h.p.,*

$$f(S^- \cap O^+) \leq \frac{\left(1+\epsilon^{-1}\right) \cdot r_d \cdot (\alpha+\epsilon)}{(1-\kappa) \cdot r} \cdot \texttt{OPT}.$$

Proof. Let X_i and $\mathcal{D}_i$ denote the set X and distribution $\mathcal{D}$ at a round i of the algorithm. An optimal element $o \in O$ is among the elements S_i^- discarded at round i if $\mathbb{E}_{R \sim \mathcal{D}_i}\left[f_{X_i \cup R \setminus \{o\}}(o)\right] < \Delta$. This bound on the value of elements $o \in S^- \cap O$ is with respect to X_i. We use curvature to relate the value of the set $S^- \cap O^+$ to its marginal

contribution to X as follows:

$$\begin{aligned} f\left(S^- \cap O^+\right) &\leq \frac{1}{1-\kappa} \cdot f_X(S^- \cap O^+) \\ &= \frac{1}{1-\kappa} \cdot f_X(\cup_{i=1}^{r_d}(S_i^- \cap O^+)) \\ &\leq \frac{1}{1-\kappa} \sum_{i=1}^{r_d} f_X\left(O^+ \cap S_i^-\right) \end{aligned}$$

where the first inequality is by curvature and the last by subadditivity. Next, using the definitions of O^+ and S_i^-, we both lower and upper bound $f_X(O^+ \cap S_i^-)$ by terms that are dependent on $|O^+ \cap S_i^-|$. First, the value of $O^+ \cap S_i^-$ is upper bounded using the threshold Δ for elements to be in S_i^-:

$$\begin{aligned} & f_X(O^+ \cap S_i^-) \\ & \leq f_{X_i}(O^+ \cap S_i^-) \\ & \leq \mathop{\mathbb{E}}_{R\sim\mathcal{D}_i}\left[f_{X_i \cup R}(O^+ \cap S_i^-)\right] + \mathop{\mathbb{E}}_{R\sim\mathcal{D}_i}\left[f_{X_i}(R)\right] \\ & \leq \mathop{\mathbb{E}}_{R\sim\mathcal{D}_i}\left[\sum_{a\in O^+\cap S_i^-} f_{X_i\cup R}(a)\right] + \mathop{\mathbb{E}}_{R\sim\mathcal{D}_i}\left[f_{X_i}(R)\right] \\ & \leq \sum_{a\in O^+\cap S_i^-} \mathop{\mathbb{E}}_{R\sim\mathcal{D}_i}\left[f_{X_i\cup R\setminus\{a\}}(a)\right] + \mathop{\mathbb{E}}_{R\sim\mathcal{D}_i}\left[f_{X_i}(R)\right] \\ & \leq |O^+ \cap S_i^-| \cdot \Delta + \mathop{\mathbb{E}}_{R\sim\mathcal{D}_i}\left[f_{X_i}(R)\right] \end{aligned}$$

where the first inequality is by submodularity, the second by monotonicity, the third by submodularity, the fourth by linearity of expectation and monotonicity, and the fifth by the definition of S_i^-. Next, we lower bound $f_X(O^+ \cap S_i^-)$ using submodularity and the definition of O^+:

$$\begin{aligned} f_X(O^+ \cap S_i^-) &\geq \sum_{o_j \in O^+ \cap S_i^-} f_{X\cup O_{j-1}}(o_j) \\ &\geq |O^+ \cap S_i^-| \cdot (1+\epsilon)\,\Delta. \end{aligned}$$

Combining these upper and lower bounds on $f(O^+ \cap S_i^-)$, we obtain the following bound on the number of good optimal elements that are discarded,

$$|O^+ \cap S_i^-| \leq (\epsilon\Delta)^{-1} \cdot \mathop{\mathbb{E}}_{R\sim\mathcal{D}_i}\left[f(R)\right].$$

Then, by adding this last bound to the upper bound for $f_X(O^+ \cap S_i^-)$, we get

$$\begin{aligned} f_X(O^+ \cap S_i^-) &\leq |O^+ \cap S_i^-| \cdot \Delta + \mathbb{E}_{R\sim\mathcal{D}_i}\left[f(R)\right] \\ &\leq \left(1+\epsilon^{-1}\right) \cdot \mathbb{E}_{R\sim\mathcal{D}_i}\left[f(R)\right]. \end{aligned}$$

Finally, by standard concentration bounds (Lemma 7 in Appendix B.1), with $m = (r/\epsilon)^2 \log\left(2r_d/\delta\right)$, w.p. $1-\delta/r_d$, $f_{X_i}(R_i) \geq \mathbb{E}_{R\sim\mathcal{D}}\left[f_{X_i}(R)\right] - \epsilon\texttt{OPT}/r$ where R_i is the sample with largest contribution to X_i at round i. By a union bound this holds for all r_d rounds where elements are discarded w.p. $1-\delta$. By the algorithm, we have $f_{X_i}(R_i) < \frac{\alpha}{r}\texttt{OPT}$ at a round where elements are discarded. Thus, $\mathbb{E}_{R\sim\mathcal{D}}\left[f_{X_i}(R)\right] \leq \frac{\alpha+\epsilon}{r}\texttt{OPT}$ at rounds $i \in r_d$, and

$$\begin{aligned} f\left(S^- \cap O^+\right) &\leq \frac{1}{1-\kappa} \sum_{i=1}^{r_d} f_X(O^+ \cap S_i^-) \\ &\leq \frac{1}{1-\kappa} \sum_{i=1}^{r_d} \left(1+\epsilon^{-1}\right) \mathop{\mathbb{E}}_{R\sim\mathcal{D}_i}\left[f_{X_i}(R)\right] \\ &\leq \frac{1}{1-\kappa}\left(1+\epsilon^{-1}\right) r_d \frac{\alpha+\epsilon}{r}\texttt{OPT}. \end{aligned}$$ □

The next lemma shows that the algorithm obtains a $1-\kappa$ approximation in one round with the k elements with largest marginal contribution, the proof is deferred to Appendix A and follows easily from the definition of curvature.

Lemma 2. *Let f be a monotone submodular function with curvature κ, then* ADAPTIVE-SAMPLING *is a non-adaptive algorithm that obtains a $(1-\kappa)$-approximation with $r = 1$ and $\alpha = 1-\kappa$.*

Combining the two previous lemmas, we get the general theorem about the approximation ratio obtained for functions with bounded curvature. For the remaining of this section, the parameters of ADAPTIVE-SAMPLING are sample complexity $m = (r/\epsilon)^2 \log\left(2\log_{1+\epsilon}(n)/\delta\right)$, $\alpha = 1/2 - \epsilon$ and $\Delta = (1+\epsilon)\texttt{OPT}/(2k)$.

Theorem 1. *Let f be a monotone submodular function with curvature κ, then, for any $\epsilon > 0$,* ADAPTIVE-SAMPLING *is a $\log_{1+\epsilon}(n) + r$ adaptive algorithm which obtains w.h.p. the following approximation:*

$$\max\left(1-\kappa, \frac{1}{2} - \frac{3\epsilon}{2} - \frac{\log_{1+\epsilon}(n)}{(1-\kappa)\cdot\epsilon\cdot r}\right).$$

Proof Sketch, full proof in Appendix A. First, we show that $f(S\cup X) \geq f(O^+ \cup X) - f(O^+ \cap S^-)$ by subadditivity and monotonicity. Then, by the definition of O^+ and O^-, we get that $f(O^+ \cup X) \geq \texttt{OPT} - |O^-|(1+\epsilon)\Delta$. By combining these two inequalities with Lemma 1, we obtain $f(S\cup X) \geq \frac{1}{2} - \epsilon - \frac{1}{1-\kappa}\left(1+\epsilon^{-1}\right)\frac{\log_{1+\epsilon}(n)}{r}$. When the algorithm returns X, $f(X) \geq \sum_{i=1}^{r} \frac{\alpha}{r}\texttt{OPT} = \left(\frac{1}{2}-\epsilon\right)\texttt{OPT}$. By Lemma 2, the algorithm obtains a $1-\kappa$ approximation. Finally, we show that a $(1+\epsilon)$ fraction of the remaining elements are discarded at every round, so the number of rounds where elements are discarded is at most $\log_{1+\epsilon}(n)$. □

Tradeoff between approximation and rounds. An interesting characteristic of this result is the tradeoff between the approximation and the number of rounds of the algorithm as a function of κ. In contrast to previous curvature-dependent approximation guarantees that decrease as a function of κ,

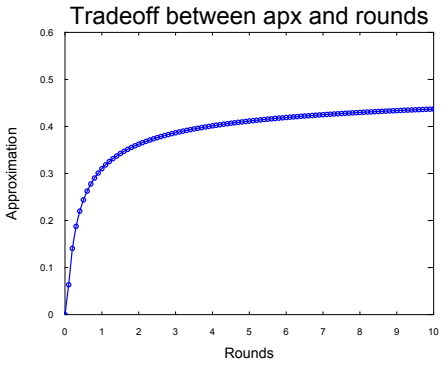

Figure 1. The tradeoff between the approximation guarantee and the number of rounds $x \cdot \frac{1}{1-\kappa} \cdot \log_{1+\epsilon}(n)$, where x is the horizontal axis, from Corollary 1.

Corollary 1 shows that as κ increases, an increase in the number of rounds maintains an approximation arbitrarily close to $1/2$. We illustrate this tradeoff in Figure 1.

Corollary 1. *Let f be a monotone submodular function with curvature κ, then, for any $\epsilon > 0$,* ADAPTIVE-SAMPLING *is a $\left(\frac{1}{1-\kappa}\frac{8}{\epsilon^2}+1\right)\log_{1+\epsilon}(n)$-adaptive algorithm that obtains w.h.p. a $\frac{1}{2}-\epsilon$ approximation.*

In particular, when $\kappa \leq 1 - 1/\operatorname{poly}(\log n)$, this corollary gives a $1/2-\epsilon$ approximation in $\operatorname{poly}(\log n)$ rounds.

Unbounded curvature. In the case where the curvature is unbounded (when $\kappa > 1 - 1/\operatorname{poly}(\log n)$), we obtain approximation guarantees by altering the function via curvaturing. Curvaturing creates a surrogate function with improved curvature such that the previous approximation guarantee holds for the surrogate function. This then implies an approximation guarantee for f with an additional additive loss. We assume that f is normalized. Recall that a function is normalized if $\max_{a \in N} f(a) \leq 1$.

Corollary 2. *Let f be a normalized monotone submodular function and S be the solution obtained by* ADAPTIVE-SAMPLING *over the function $\tilde{f}$ with curvature $\kappa = 1 - 1/\log n$ obtained via curvaturing f. Then,*

$$f(S) \geq \left(\frac{1}{2}-\epsilon\right) OPT - \frac{k}{\log n}\left(\frac{1}{2}+\epsilon\right)\left(1+\frac{1}{\log n}\right).$$

Proof. The function f is curvatured with $\kappa = 1 - 1/\log n$ to obtain the following surrogate function $\tilde{f}$:

$$\tilde{f}(S) = \left(1-\frac{1}{\log n}\right) f(S) + \frac{|S|}{\log n}.$$

Note that the optimal solution O of size k for f is also an optimal solution for $\tilde{f}$. Let S be the solution obtained by ADAPTIVE-SAMPLING on $\tilde{f}$. By Corollary 1, it is a $1/2-\epsilon$ approximation to $\tilde{f}(O)$. We get

$$\begin{aligned} f(S) &= \left(\tilde{f}(S) - \frac{k}{\log n}\right)\left(1-\frac{1}{\log n}\right)^{-1} \\ &\geq \left(\left(\frac{1}{2}-\epsilon\right)\tilde{f}(O) - \frac{k}{\log n}\right)\left(1-\frac{1}{\log n}\right)^{-1} \\ &\geq \left(\frac{1}{2}-\epsilon\right) f(O) - \frac{k}{\log n}\left(\frac{1}{2}+\epsilon\right)\left(1+\frac{1}{\log n}\right). \end{aligned}$$

□

3.2. Homogeneity

The bad optimal elements O^- are used to analyze the approximation in terms of the homogeneity condition. Homogeneity plays a complementary role to curvature which, as previously shown, bounds the loss from good optimal elements O^+. The following lemma improves the bound on the number $|O^-|$ of bad optimal elements as a function of the homogeneity parameter μ, which then implies an improved approximation guarantee (proof deferred to Appendix A).

Lemma 3. *Let f be a μ-homogeneous monotone submodular function. Then,*

$$|O^-| \leq \frac{\mu}{\mu + (1-\epsilon)/2} \cdot k.$$

Combining the bounds on the losses due to both good and bad optimal elements, we obtain an approximation guarantee arbitrarily close to 1 for functions with arbitrarily good homogeneity when the curvature κ is bounded.

Theorem 3. *Let f be a μ-homogeneous monotone submodular function with curvature $\kappa < 1$, then* ADAPTIVE-SAMPLING *is a $\left(\frac{1}{1-\kappa}\frac{1}{\epsilon^2}+1\right)\log_{1+\epsilon}(n)$ adaptive algorithm which obtains w.h.p. the following approximation:*

$$1 - \frac{\mu \cdot (1+\epsilon)^2}{2\mu + 1 - \epsilon} - \epsilon.$$

Proof Sketch, full proof in Appendix A. Similarly as for Theorem 1, we have $f(S \cup X) \geq f(O^+ \cup X) - f(O^+ \cap S^-)$ and $f(O^+ \cup X) \geq \texttt{OPT} - |O^-|(1+\epsilon)\Delta$. By combining these two inequalities with Lemma 1 and Lemma 3, we then get the desired approximation guarantee. The approximation obtained when the algorithm returns X and the number of rounds follow similarly as for Theorem 1. □

4. Lower Bound

In this section, we show that the number of rounds needed to obtain a $1 - \kappa + o(1)$ approximation is $\Omega(\log n / \log\log n)$. Together with Corollary 1 from the previous section, this provides a tight, up to lower order factors, characterization

of the number of rounds needed to obtain a $1/2 - \epsilon$ approximation for functions with bounded curvature. This hardness result is achieved with a general lemma that uses curvaturing to reduce the problem of showing lower bounds for submodular functions with bounded curvature to lower bounds for general submodular functions.

Lemma 4. *Assume $\mathcal{F}$ is a class of normalized monotone submodular functions such that $\texttt{OPT} \geq (1-\epsilon)k$, $\epsilon > 0$, that cannot be α approximated in r rounds. Then there exists a class of monotone submodular functions $\mathcal{F}'$ with curvature κ that cannot be $\alpha\kappa + \frac{1-\kappa}{1-\epsilon}$ approximated in r rounds.*

Proof Sketch, full proof in Appendix C. We consider the class of functions $\mathcal{F}'$ obtained by curvaturing $\mathcal{F}$. We then show that an algorithm that is an $\alpha\kappa + \frac{1-\kappa}{1-\epsilon}$ approximation algorithm for $\mathcal{F}'$ is an algorithm that is an α approximation for $\mathcal{F}$, which does not exist in r rounds by the assumption on the class of functions $\mathcal{F}$. □

With this reduction, the hardness result in Balkanski & Singer (2018) for general monotone submodular functions implies the following lower bound for submodular functions with bounded curvature.

Theorem 4. *There is no $\frac{\log n}{12 \log\log n}$-adaptive algorithm that obtains, with probability $\omega(1/n)$, an approximation of*

$$\frac{1-\kappa}{1-\frac{2}{\log n}} + \frac{\kappa}{\log n}$$

for monotone submodular functions with curvature κ.

Proof Sketch, full proof in Appendix C. The hardness result in Balkanski & Singer (2018) shows that there is no $\frac{\log n}{12 \log\log n}$-adaptive algorithm that obtains, w.p. $\omega(1/n)$, a $\frac{1}{\log n}$-approximation for general monotone submodular functions. After normalizing the hard class of functions, Lemma 4 immediately implies the hardness result. □

5. Experiments

We conduct experiments on two datasets to empirically evaluate the performance of the adaptive sampling algorithm. We observe that it performs almost as well as the standard greedy algorithm, which achieves the optimal $1 - 1/e$ approximation, and outperforms two simple algorithms with low adaptivity. These experiments indicate that in practice, adaptive sampling performs significantly better than its worst-case $1/3$ approximation guarantee.

5.1. Experimental setup

We begin by describing the two datasets and the benchmarks for the experiments.

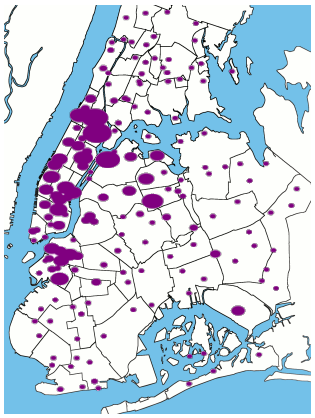

Figure 2. A map of New York City where the purple circles are of size proportional to the number of taxi trips with pick-ups that occurred in the corresponding neighborhood.

5.1.1. Datasets

Movie recommendation system. The goal of a movie recommendation system is to find a personalized and diverse collection of movies to recommend to an individual user, given ratings of movies that this user has already seen. We use the MovieLens 1M dataset (Harper & Konstan., 2015) which contains 1 million ratings from 6000 users on 4000 movies. A standard approach to solve the problem of movie recommendation is low-rank matrix completion. This approach models the problem as an incomplete rating matrix with users as rows and movies as columns and aims to produce a complete matrix which agrees with the incomplete matrix and has low rank. For a given user u_i, the completed matrix then gives a predicted score for each movie m_j which we denote by $v_{i,j}$. A high quality recommendation must also be diverse. We add a diversity term in the objective that is a coverage function C where $C(S)$ is the number of different genres covered by movies in S.[1] We obtain the following objective for user u_i:

$$f_{i,\alpha}(S) = (1-\alpha) \sum_{m_j \in S} v_{i,j} + \alpha C(S)$$

where α is a parameter controlling the weight of the objective on the individual movie scores versus the diversity term. Similar submodular objectives for movie recommendation systems have previously been used, e.g., (Mitrovic et al., 2017; Lindgren et al., 2015; Mirzasoleiman et al., 2016; Feldman et al., 2017). The algorithm used for low-rank matrix completion is an iterative low-rank SVD decomposition algorithm from the python package fancyimpute (Rubinsteyn & Feldman, 2017) corresponding to the SVDimpute algorithm analyzed in Troyanskaya et al. (2001). Unless otherwise specified, we set $k = 100$, $\alpha = 0.6$, and number

[1] Each movie has one genre, for example, "romantic comedy" is one genre, which is different than the "romantic drama" genre.

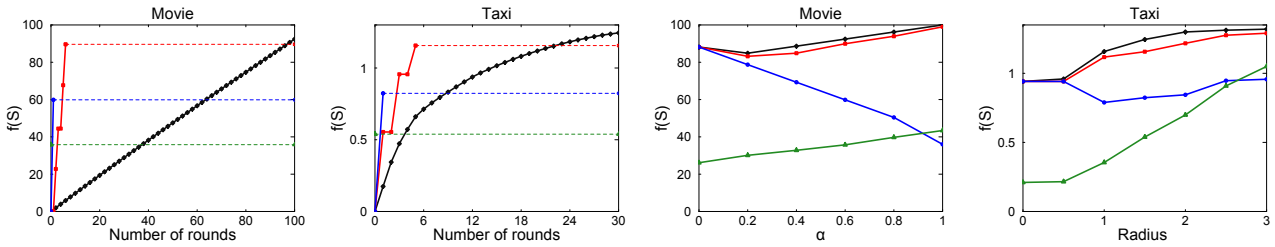

Figure 3. The GREEDY, ADAPTIVE-SAMPLING, TOPK, and RANDOM algorithms correspond to the black, red, blue, and green lines respectively. Figures 3(a) and 3(b) show the evolution of the value of the current solution of each algorithm at every round. The dotted lines indicate that the algorithm terminated at a previous round. Figures 3(c) and 3(d) show the final value obtained by each algorithm as a function of the weight parameter α and radius R for the movie recommendation and taxi applications respectively. The curvature of the functions increases as α and R increase.

of rounds of adding elements $r = 4$ for adaptive sampling.

Taxi dispatch. In the taxi dispatch application, there are k taxis and the goal is to pick the k best locations to cover the maximium number of potential customers. We use 2 millions taxi trips in June 2017 from the New York City taxi and limousine commission trip record dataset (NYC-Taxi-Limousine-Commission, 2017), illustrated in Figure 2. We assign a weight w_i to each neighborhood $n_i \in N$ that is equal to the number of trips where the pick-up was in neighborhood n_i, where N is the collection of all neighborhoods. We then build a coverage function $C_R(S)$ which is equal to the sum of the weights of neighborhoods n_i that are reachable from at least one location in S, where reachable means $n_j \in S$ is at "as the crow flies" distance $d(i,j) \leq R$ from n_i. More precisely,

$$C_R(S) = \sum_{n_i \in N} \mathbb{1}_{\exists n_j \in S : d(i,j) \leq R} \cdot w_i.$$

Unless otherwise specified, the parameters are $k = 30$, radius $R = 1.5km$, and number of rounds of adding elements for adaptive sampling $r = 3$.

5.1.2. BENCHMARKS

We compare the performance of ADAPTIVE-SAMPLING with three algorithms. The GREEDY algorithm, which adds the element with largest marginal contribution at each round, is the standard algorithm for submodular optimization and obtains the optimal $1 - e^{-1}$ approximation (and $(1 - e^{-\kappa})/\kappa$ for functions with curvature κ (Conforti & Cornuéjols, 1984)) in linearly many rounds. It is used as an upper bound to measure the performance cost of obtaining logarithmic adaptivity with ADAPTIVE-SAMPLING. The TOPK algorithm picks the k elements a with largest singleton value $f(a)$. This simple algorithm has one adaptive round and obtains a $1 - \kappa$ approximation for submodular functions with curvature κ. Its low adaptivity and its approximation guarantee make it a natural benchmark. Finally, RANDOM simply returns a random subset of size k and has 0 rounds of adaptivity.

5.2. Experimental results

General performance. We first analyze how the value of the solutions maintained by each algorithm evolves at every round. In Figures 3(a) and 3(b), we observe that ADAPTIVE-SAMPLING achieves a final value that is close to the one obtained by GREEDY, but in a much smaller number of rounds. ADAPTIVE-SAMPLING also significantly outperforms the two simple algorithms. There are rounds where the value of the ADAPTIVE-SAMPLING solution does not increase, these correspond to rounds where elements are discarded, and which allow to then pick better elements in future rounds. For the movie recommender application, the value of the solution obtained by GREEDY increases linearly but we emphasize that this function is not linear, as movies that have the same genre as a movie already picked have their marginal contribution to the solution that decreases by α. In these experiments, ADAPTIVE-SAMPLING uses only 100 samples at every round. In fact, we observe very similar performance for ADAPTIVE-SAMPLING whether it uses 10 or 10K samples per round. Thus, the sample complexity is not an issue for ADAPTIVE-SAMPLING in practice and can be much lower than the theoretical sample complexity needed for the approximation guarantee.

The role of curvature and homogeneity. Next, we analyze the performance of the algorithms as a function of curvature and homogeneity. Both functions have curvature $\kappa = 0$ when $\alpha = 0$ and $R = 0$ respectively, and the cuvature increase as α and the radius increase. The movie application has good homogeneity with μ close to 0 regardless of α since optimal movies all have similarly high predicted ratings and different genres. On the other hand, homogeneity gets worse as the radius increases for the taxi dispatch application since one neighborhood covers a larger number of neighborhoods as the radius increases.

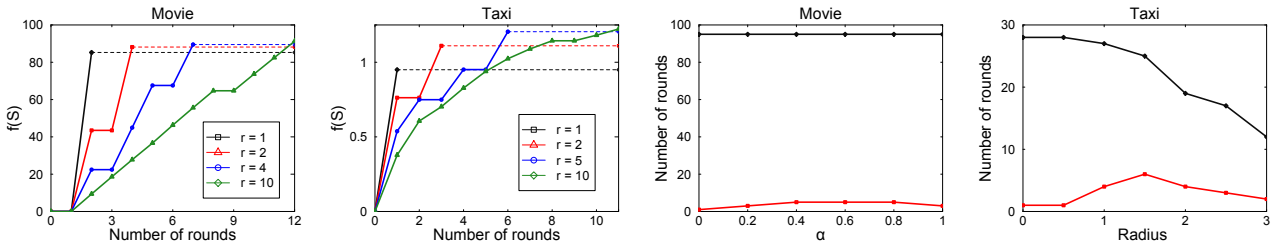

Figure 4. Figures 4(a) and 4(b) show the evolution of the value of the current solution of ADAPTIVE-SAMPLING at every round, for different values of the parameter r which controls the number of rounds of the algorithm. Figures 4(c) and 4(d) show how many rounds are needed by GREEDY (in black) and ADAPTIVE-SAMPLING (in red) to achieve 95 percent of the final value obtained by GREEDY.

Again, we observe in Figures 3(c) and 3(d) that ADAPTIVE-SAMPLING obtains a solution of value of very close to the value obtained by GREEDY, and significantly better than the two simple algorithms in general for any α and any radius. As it is implied by the theoretical bounds, ADAPTIVE-SAMPLING, TOPK, and GREEDY all perform arbitrarily close to the optimal solution when the curvature is small. The gap between ADAPTIVE-SAMPLING and GREEDY is the largest for mid-range values of α and R. This can be explained by the design of the functions, which become "easier" to optimize as α and R increase since any neighborhood covers a large fraction of the total value when R is large and since there is always a large number of movies that have a genre that is not yet in the current solution.

Figures 4(c) and 4(d) show how many rounds are needed by GREEDY and ADAPTIVE-SAMPLING to obtain 95 percent of the value of the solution of GREEDY. When the curvature is small, the k elements with largest contribution is a good solution so ADAPTIVE-SAMPLING only needs one round, whereas the value obtained by GREEDY grows linearly so it needs to be close to 95 percent of its k rounds. For the movie recommendation, since the value obtained by GREEDY always grows almost linearly, GREEDY always needs 95 rounds for $k = 100$. For the taxi dispatch, since a small number of elements can have very large value for large radius, the number of rounds needed by GREEDY decreases for large radius, as well as for ADAPTIVE-SAMPLING. Similarly as in the two previous figures with the approximation, we observe that the setting where ADAPTIVE-SAMPLING needs the most number of rounds is for mid-range radius.

Number of rounds r versus performance. There is a tradeoff between the number of rounds of ADAPTIVE-SAMPLING and its performance. This tradeoff is more apparent for the taxi application than for the movie recommender application where ADAPTIVE-SAMPLING obtains high value after 2 rounds (Figures 4(a) and 4(b)). Overall, ADAPTIVE-SAMPLING obtains a high value in a small number of rounds, but this value can be slightly improved by increasing the number of rounds of ADAPTIVE-SAMPLING.

6. Related Work

Map-Reduce. There is a long line of work on distributed submodular optimization in the Map-Reduce model (Kumar et al., 2015; Mirzasoleiman et al., 2013; Mirrokni & Zadimoghaddam, 2015; Mirzasoleiman et al., 2015; Barbosa et al., 2015; 2016; Epasto et al., 2017). Map-Reduce is designed to tackle issues related to massive data sets that are too large to either fit or be processed by a single machine. Instead of addressing distributed challenges, adaptivity addresses the issue of sequentiality, where query-evaluation time is the main runtime bottleneck and where these evaluations can be parallelized. The existing Map-Reduce algorithms for submodular optimization have adaptivity that is linear in n in the worst-case. This high adaptivity is caused by the distributed algorithms which are run on each machine, which are variants of the greedy algorithm and thus have adaptivity at least linear in k.

Parallel computing and depth. In the PRAM model, the notion of depth is closely related to the concept of adaptivity. The *depth* of a PRAM algorithm is the number of parallel steps of this algorithm on a shared memory machine with any number of processors, in other words, it is the longest chain of dependencies of the algorithm, including operations which are not necessarily queries. The problem of designing low-depth algorithms is well-studied , e.g. (Blelloch, 1996; Blelloch et al., 2011; Berger et al., 1989; Rajagopalan & Vazirani, 1998; Blelloch & Reid-Miller, 1998; Blelloch et al., 2012). Our positive results extend to the PRAM model with the adaptive sampling algorithm having $\tilde{\mathcal{O}}(\log^2 n \cdot d_f)$ depth, where d_f is the depth required to evaluate the function on a set. While the PRAM model assumes that the input is loaded in memory, we consider the value query model where the algorithm is given oracle access to a function of potentially exponential size.

More broadly, there has been recent interest in machine learning to scale submodular optimization algorithms for applications over large datasets (Jegelka et al., 2011; 2013; Wei et al., 2014; Nishihara et al., 2014; Pan et al., 2014).

Acknowledgements

This research was supported by a Google PhD Fellowship, NSF grant CAREER CCF-1452961, BSF grant 2014389, NSF USICCS proposal 1540428, Google research award, and a Facebook research award.

References

Agarwal, A., Agarwal, S., Assadi, S., and Khanna, S. Learning with limited rounds of adaptivity: Coin tossing, multi-armed bandits, and ranking from pairwise comparisons. In *COLT*, pp. 39–75, 2017.

Anari, N., Goel, G., and Nikzad, A. Mechanism design for crowdsourcing: An optimal 1-1/e competitive budget-feasible mechanism for large markets. In *FOCS*, pp. 266–275. IEEE, 2014.

Balkanski, E. and Hartline, J. D. Bayesian budget feasibility with posted pricing. In *WWW*, pp. 189–203, 2016.

Balkanski, E. and Singer, Y. The adaptive complexity of maximizing a submodular function. In *STOC*, 2018.

Balkanski, E., Rubinstein, A., and Singer, Y. The power of optimization from samples. In *NIPS*, pp. 4017–4025, 2016.

Barbosa, R., Ene, A., Nguyen, H., and Ward, J. The power of randomization: Distributed submodular maximization on massive datasets. In *ICML*, pp. 1236–1244, 2015.

Barbosa, R. d. P., Ene, A., Nguyen, H. L., and Ward, J. A new framework for distributed submodular maximization. In *FOCS*, pp. 645–654. Ieee, 2016.

Bei, X., Chen, N., Gravin, N., and Lu, P. Budget feasible mechanism design: from prior-free to bayesian. In *STOC*, pp. 449–458. ACM, 2012.

Berger, B., Rompel, J., and Shor, P. W. Efficient nc algorithms for set cover with applications to learning and geometry. In *FOCS*, pp. 54–59. IEEE, 1989.

Blelloch, G. E. Programming parallel algorithms. *Communications of the ACM*, 39(3):85–97, 1996.

Blelloch, G. E. and Reid-Miller, M. Fast set operations using treaps. In *SPAA*, pp. 16–26, 1998.

Blelloch, G. E., Peng, R., and Tangwongsan, K. Linear-work greedy parallel approximate set cover and variants. In *SPAA*, pp. 23–32, 2011.

Blelloch, G. E., Simhadri, H. V., and Tangwongsan, K. Parallel and i/o efficient set covering algorithms. In *SPAA*, pp. 82–90. ACM, 2012.

Braverman, M., Mao, J., and Weinberg, S. M. Parallel algorithms for select and partition with noisy comparisons. In *STOC*, pp. 851–862, 2016.

Buhrman, H., García-Soriano, D., Matsliah, A., and de Wolf, R. The non-adaptive query complexity of testing k-parities. *arXiv preprint arXiv:1209.3849*, 2012.

Canonne, C. and Gur, T. An adaptivity hierarchy theorem for property testing. *arXiv preprint arXiv:1702.05678*, 2017.

Chen, X., Servedio, R. A., Tan, L.-Y., Waingarten, E., and Xie, J. Settling the query complexity of non-adaptive junta testing. *arXiv preprint arXiv:1704.06314*, 2017.

Cole, R. Parallel merge sort. *SIAM Journal on Computing*, 17(4):770–785, 1988.

Conforti, M. and Cornuéjols, G. Submodular set functions, matroids and the greedy algorithm: tight worst-case bounds and some generalizations of the rado-edmonds theorem. *Discrete applied mathematics*, 7(3):251–274, 1984.

Duris, P., Galil, Z., and Schnitger, G. Lower bounds on communication complexity. In *STOC*, pp. 81–91, 1984.

Epasto, A., Mirrokni, V. S., and Zadimoghaddam, M. Bicriteria distributed submodular maximization in a few rounds. In *SPAA*, pp. 25–33, 2017.

Feldman, M., Harshaw, C., and Karbasi, A. Greed is good: Near-optimal submodular maximization via greedy optimization. *arXiv preprint arXiv:1704.01652*, 2017.

Harper, F. M. and Konstan., J. A. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4, Article 19 (December 2015), 19 pages.*, 2015. doi: http://dx.doi.org/10.1145/2827872.

Haupt, J., Nowak, R., and Castro, R. Adaptive sensing for sparse signal recovery. In *Digital Signal Processing Workshop and 5th IEEE Signal Processing Education Workshop*, pp. 702–707. IEEE, 2009a.

Haupt, J. D., Baraniuk, R. G., Castro, R. M., and Nowak, R. D. Compressive distilled sensing: Sparse recovery using adaptivity in compressive measurements. In *Signals, Systems and Computers, 2009 Conference Record of the Forty-Third Asilomar Conference on*, pp. 1551–1555. IEEE, 2009b.

Indyk, P., Price, E., and Woodruff, D. P. On the power of adaptivity in sparse recovery. In *FOCS*, pp. 285–294. IEEE, 2011.

Iyer, R. K. and Bilmes, J. A. Submodular optimization with submodular cover and submodular knapsack constraints. In *NIPS*, 2013.

Iyer, R. K., Jegelka, S., and Bilmes, J. A. Curvature and optimal algorithms for learning and minimizing submodular functions. In *NIPS*, 2013.

Jegelka, S., Lin, H., and Bilmes, J. A. On fast approximate submodular minimization. In *Advances in Neural Information Processing Systems*, pp. 460–468, 2011.

Jegelka, S., Bach, F., and Sra, S. Reflection methods for user-friendly submodular optimization. In *Advances in Neural Information Processing Systems*, pp. 1313–1321, 2013.

Kumar, R., Moseley, B., Vassilvitskii, S., and Vattani, A. Fast greedy algorithms in mapreduce and streaming. *ACM Transactions on Parallel Computing*, 2(3):14, 2015.

Lindgren, E. M., Wu, S., and Dimakis, A. G. Sparse and greedy: Sparsifying submodular facility location problems. In *NIPS Workshop on Optimization for Machine Learning*, 2015.

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

Mirzasoleiman, B., Karbasi, A., Sarkar, R., and Krause, A. Distributed submodular maximization: Identifying representative elements in massive data. In *NIPS*, pp. 2049–2057, 2013.

Mirzasoleiman, B., Karbasi, A., Badanidiyuru, A., and Krause, A. Distributed submodular cover: Succinctly summarizing massive data. In *NIPS*, pp. 2881–2889, 2015.

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

Mitrovic, S., Bogunovic, I., Norouzi-Fard, A., Tarnawski, J. M., and Cevher, V. Streaming robust submodular maximization: A partitioned thresholding approach. In *NIPS*, pp. 4560–4569, 2017.

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

Nemhauser, G. L., Wolsey, L. A., and Fisher, M. L. An analysis of approximations for maximizing submodular set functionsi. *Mathematical Programming*, 14(1):265–294, 1978.

Nisan, N. and Widgerson, A. Rounds in communication complexity revisited. In *STOC*, pp. 419–429, 1991.

Nishihara, R., Jegelka, S., and Jordan, M. I. On the convergence rate of decomposable submodular function minimization. In *Advances in Neural Information Processing Systems*, pp. 640–648, 2014.

NYC-Taxi-Limousine-Commission. New york city taxi and limousine commission trip record data. 2017. URL http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml.

Pan, X., Jegelka, S., Gonzalez, J. E., Bradley, J. K., and Jordan, M. I. Parallel double greedy submodular maximization. In *Advances in Neural Information Processing Systems*, pp. 118–126, 2014.

Papadimitriou, C. H. and Sipser, M. Communication complexity. *Journal of Computer and System Sciences*, 28(2): 260–269, 1984.

Rajagopalan, S. and Vazirani, V. V. Primal-dual rnc approximation algorithms for set cover and covering integer programs. *SIAM Journal on Computing*, 28(2):525–540, 1998.

Rubinsteyn, A. and Feldman, S. fancyimpute, matrix completion and feature imputation algorithms. 2017. URL https://github.com/hammerlab/fancyimpute.

Sviridenko, M., Vondrák, J., and Ward, J. Optimal approximation for submodular and supermodular optimization with bounded curvature. In *SODA*, 2015.

Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., and Altman, R. B. Missing value estimation methods for dna microarrays. *Bioinformatics*, 17(6):520–525, 2001.

Valiant, L. G. Parallelism in comparison problems. *SIAM Journal on Computing*, 4(3):348–355, 1975.

Vondrák, J. Submodularity and curvature: the optimal algorithm. *RIMS*, 2010.

Wei, K., Iyer, R., and Bilmes, J. Fast multi-stage submodular maximization. In *ICML*, pp. 1494–1502, 2014.