Submodular Maximization beyond Non-negativity: Guarantees, Fast Algorithms, and Applications

Christopher Harshaw [1] **Moran Feldman** [2] **Justin Ward** [3] **Amin Karbasi** [4]

Abstract

It is generally believed that submodular functions—and the more general class of γ-weakly submodular functions—may only be optimized under the non-negativity assumption $f(S) \geq 0$. In this paper, we show that once the function is expressed as the difference $f = g - c$, where g is monotone, non-negative, and γ-weakly submodular and c is non-negative modular, then strong approximation guarantees may be obtained. We present an algorithm for maximizing $g - c$ under a k-cardinality constraint which produces a random feasible set S such that $\mathbb{E}\left[g(S) - c(S)\right] \geq (1 - e^{-\gamma} - \epsilon)g(OPT) - c(OPT)$, whose running time is $O(\frac{n}{\epsilon}\log^2\frac{1}{\epsilon})$, independent of k. We extend these results to the unconstrained setting by describing an algorithm with the same approximation guarantees and faster $O(\frac{n}{\epsilon}\log\frac{1}{\epsilon})$ runtime. The main techniques underlying our algorithms are two-fold: the use of a surrogate objective which varies the relative importance between g and c throughout the algorithm, and a geometric sweep over possible γ values. Our algorithmic guarantees are complemented by a hardness result showing that no polynomial-time algorithm which accesses g through a value oracle can do better. We empirically demonstrate the success of our algorithms by applying them to experimental design on the Boston Housing dataset and directed vertex cover on the Email EU dataset.

[1]Department of Computer Science, Yale University, New Haven, USA [2]Department of Mathematics and Computer Science, Open University of Israel, Raanana, Israel [3]School of Mathematical Sciences, Queen Mary University of London, London, UK [4]Department of Electrical Engineering, Yale University, New Haven, USA. Correspondence to: Christopher Harshaw <christopher.harshaw@yale.edu>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

1. Introduction

From summarization and recommendation to clustering and inference, many machine learning tasks are inherently discrete. Submodularity is an attractive property when designing discrete objective functions, as it encodes a natural diminishing returns condition and also comes with an extensive literature on optimization techniques. Submodular optimization techniques have been successfully applied in a wide variety of machine learning tasks, including sensor placement (Krause & Guestrin, 2005), document summarization (Lin & Bilmes, 2011), speech subset selection (Wei et al., 2013) influence maximization in social networks (Kempe et al., 2003), information gathering (Golovin & Krause, 2011), and graph-cut based image segmentation (Boykov et al., 2001; Jegelka & Bilmes, 2011), to name a few. However, in instances when the objective function is not submodular, existing techniques for submodular optimization many perform arbitrarily poorly, motivating the need to study broader function classes. While several notions of approximate submodularity have been studied, the class of γ-weakly submodular functions have (arguably) enjoyed the most practical success. For example, γ-weakly submodular optimization techniques have been used in feature selection (Das & Kempe, 2011; Khanna et al., 2017), anytime linear prediction (Hu et al., 2016), interpretation of deep neural networks (Elenberg et al., 2017), and high dimensional sparse regression (Elenberg et al., 2018).

Here, we study the constrained maximization problem

$$\max_{|S| \leq k} g(S) - c(S) \ , \tag{1}$$

where g is a non-negative monotone γ-weakly submodular function and c is a non-negative modular function. Problem (1) has various interpretations which may extend the current submodular framework to apply to more tasks in machine learning. For instance, the modular cost c may be added as a penalty to existing submodular maximization problems to encode a cost for each element. Such a penalty term may play the role of a regularizer or soft constraint in a model. When g models the revenue of some collection of products S and c models the cost of each item, then (1) corresponds to maximizing profits.

While Problem 1 has promising modeling potential, existing optimization techniques fail to provide nontrivial approximation guarantees. The main reason is that most existing techniques require the objective function to take only non-negative values, while $g(S) - c(S)$ may take both positive and negative values. Moreover, $g(S) - c(S)$ might be non-monotone, and thus, the definition of γ-weak submodularity does not even apply to it when $\gamma < 1$.

Our Contributions We provide several fast algorithms for solving Problem (1) as well as a matching hardness result and experimental validation of our methods. In particular,

1. **Algorithms.** In the case where γ is known, we provide a deterministic algorithm which uses $O(nk)$ function evaluations and returns a set S such that $g(S) - c(S) \geq (1 - e^{-\gamma})g(OPT) - c(OPT)$. If g is regarded as revenue and c as a cost, then this guarantee intuitively states that the algorithm will return a solution whose total profit is at least as much as would be obtained by paying the same cost as the optimal solution while gaining at least $(1 - e^{-\gamma})$ as much revenue. We extend this to a randomized variant which uses $O(n \log \frac{1}{\epsilon})$ function evaluations and has a similar approximation guarantee in expectation, but with an ϵ additive loss in the approximation factor. We also provide a randomized algorithm for the unconstrained setting (when $k = n$) which achieves the same $1 - e^{-\gamma}$ approximation factor in expectation using only $O(n)$ function evaluations. When γ is unknown, we give a meta-algorithm for guessing γ that loses a δ additive factor in the approximation ratio and increases the run time by a multiplicative $O(\frac{1}{\delta} \log \frac{1}{\delta})$ factor.

2. **Hardness of Approximation.** To complement our algorithms, we provide a matching hardness result which shows that no algorithm which makes polynomially many queries in the value oracle model may do better. To the best of our knowledge, this is the first hardness result of this kind for γ-weakly submodular functions.

3. **Experimental Evaluation.** We demonstrate the effectiveness of our algorithm on experimental design on the Boston Housing dataset and directed vertex cover on the Email EU dataset, both with costs.

Prior Work The celebrated result of Nemhauser et al. (1978) showed that the greedy algorithm achieves a $(1-1/e)$ approximation for maximizing a nonnegative monotone submodular function subject to a cardinality constraint. Das & Kempe (2011) showed the more general result that the greedy algorithm achieves a $(1 - e^{-\gamma})$ when g is γ-weakly submodular. At the same time, an extensive line of research has lead to the development of algorithms to handle non-monotone submodular objectives under more complicated constraints (see, e.g., (Buchbinder & Feldman, 2016; Chekuri et al., 2014; Ene & Nguyen, 2016; Feldman et al., 2017; Lee et al., 2010; Sviridenko, 2004)). The $(1 - 1/e)$ approximation was shown to be optimal in the value oracle model (Nemhauser & Wolsey, 1978), but until this work, no stronger hardness result was known for constrained γ-weakly submodular maximization. The problem of maximizing $g + \ell$ for non-negative monotone submodular g and an (arbitrary) modular function ℓ under cardinality constraints was first considered in (Sviridenko et al., 2017), who gave a randomized polynomial time algorithm which outputs a set S such that $g(S) + \ell(S) \geq (1 - 1/e)g(OPT) + \ell(OPT)$ where OPT is the optimal set. This approximation was shown to be optimal in the value oracle model via a reduction from submodular maximization with bounded curvature. However, the algorithm of Sviridenko et al. (2017) is of mainly theoretical interest, as it requires continuous optimization of the multilinear extension and an expensive routine to guess the contribution of OPT to the modular term, yielding it practically intractable. Feldman (2018) suggested using a surrogate objective that varies with time, and showed that this removes the need for the guessing step. However, the algorithm of (Feldman, 2018) still requires expensive sampling as it is based on the multilinear extension. Moreover, neither of these approaches can currently handle γ-weakly submodular functions, as optimization routines that go through their multilinear extensions have not yet been developed.

Organization The remainder of the paper is organized as follows. Preliminary definitions are given in Section 2. The algorithms we present for solving Problem (1) are presented in Section 3. The hardness result is stated in Section 4. Applications, experimental set-up, and experimental results are discussed in Section 5. Finally, we conclude with a discussion in Section 6. Due to space considerations, most of the proofs have been omitted from the main paper and may be found in the supplementary material.

2. Preliminaries

Let Ω be a ground set of size n. For a real-valued set function $g : 2^{\Omega} \to \mathbb{R}$, we write the marginal gain of adding an element e to a set A as $g(e \mid S) \triangleq g(S \cup \{e\}) - g(S)$. We say that g is *monotone* if $g(A) \leq g(B)$ for all $A \subseteq B$. We say that g is *submodular* if for all sets $A \subseteq B \subseteq \Omega$ and element $e \notin B$,

$$g(e \mid A) \geq g(e \mid B) \ . \tag{2}$$

When g is interpreted as a utility function, (2) encodes a natural diminishing returns condition in the sense that the marginal gain of adding an element decreases as the current set grows larger. An equivalent definition is that $\sum_{e \in B} g(e \mid A) \geq g(A \cup B) - g(A)$, which allows for the

following natural extension. A monotone set function g is γ-*weakly submodular* for $\gamma \in (0, 1]$ if

$$\sum_{e \in B \setminus A} g(e \mid A) \geq \gamma \left(g(A \cup B) - g(A) \right) \tag{3}$$

holds for all $A \subseteq B$. Here, γ is referred to as the *submodularity ratio*. Intuitively, such a function g may not have strictly diminishing returns, but the increase in the returns is bounded by the marginals. Note that g is submodular if and only if it is γ-weakly submodular with $\gamma = 1$. A real-valued set function $c : 2^{\Omega} \to \mathbb{R}$ is *modular* if (2) holds with equality. A modular function may always be written in terms of coefficients as $c(S) = \sum_{e \in S} c_e$ and is non-negative if and only if all of its coefficients are non-negative.

Our algorithms are specified in the *value oracle model*, namely under the assumption that there is an oracle that, given a set $S \subseteq \Omega$, returns the value $g(S)$. As is standard, we analyze the run time complexity of these algorithms in terms of the number of function evaluations they require.

3. Algorithms

In this section, we present a suite of fast algorithms for solving Problem 1. The main idea behind each of these algorithms is to optimize a surrogate objective, which changes throughout the algorithm, preventing us from getting stuck in poor local optima. Further computational speed ups are obtained by randomized sub-sampling of the ground set.[1] The first algorithms we present assume knowledge of the weak submodularity parameter γ. However, γ is rarely known in practice (unless it is equal to 1), and thus, we show in Section 3.4 how to adapt these algorithms for the case of unknown γ.

To motivate the distorted objective we use, let us describe a way in which the greedy algorithm may fail. Suppose there is a "bad element" $b \in \Omega$ which has the highest overall gain, $g(b) - c_b$ and so is added to the solution set; however, once added, the marginal gain of all remaining elements drops below the corresponding costs, and so the greedy algorithm terminates. This outcome is suboptimal when there are other elements e that, although their overall marginal gain $g(e \mid S) - c_e$ is lower, have much higher ratio between the marginal utility $g(e \mid S)$ and the cost c_e (see Appendix A for an explicit construction).

To avoid this type of situation, we design a distorted objective which initially places higher relative importance on the modular cost term c, and gradually increases the relative importance of the utility g as the algorithm progresses. Our analysis relies on two functions: Φ, the distorted objective, and Ψ, an important quantity in analyzing the trajectory of Φ. Let k denote the cardinality constraint, then for any iteration $i = 0, \ldots, k-1$ of our algorithm and any set T, we define

$$\Phi_i(T) \triangleq \left(1 - \frac{\gamma}{k}\right)^{k-i} g(T) - c(T) \ .$$

Additionally, for any $i = 0, \ldots, k$, a set $T \subseteq \Omega$, and an element $e \in \Omega$, let

$$\Psi_i(T, e) \triangleq \max \left\{ 0, \left(1 - \frac{\gamma}{k}\right)^{k-(i+1)} g(e \mid T) - c_e \right\} \ .$$

Most proofs in this section are deferred to Appendix B.

Algorithm 1 DISTORTED GREEDY

Input: utility g, weak γ, cost c, cardinality k
Initialize $S_0 \leftarrow \varnothing$
for $i = 0$ **to** $k - 1$ **do**
  $e_i \leftarrow \arg\max_{e \in \Omega} \left\{ \left(1 - \frac{\gamma}{k}\right)^{k-(i+1)} g(e \mid S_i) - c_e \right\}$
  if $\left(1 - \frac{\gamma}{k}\right)^{k-(i+1)} g(e_i \mid S_i) - c_{e_i} > 0$ **then**
    $S_{i+1} \leftarrow S_i \cup \{e_i\}$
  end if
end for

3.1. Distorted Greedy

Our first algorithm, DISTORTED GREEDY, is presented as Algorithm 1. At each iteration, this algorithm chooses an element e_i maximizing the increase in the distorted objective. The algorithm then only accepts e_i if it positively contributes to the distorted objective. The analysis consists mainly of two lemmas. First, Lemma 1 shows that the marginal gain in the distorted objective is lower bounded by a term involving Ψ. This fact relies on the non-negativity of c and the rejection step in the algorithm.

Lemma 1. *In each iteration of* DISTORTED GREEDY,

$$\Phi_{i+1}(S_{i+1}) - \Phi_i(S_i) = \Psi_i(S_i, e_i) + \frac{\gamma}{k}\left(1 - \frac{\gamma}{k}\right)^{k-(i+1)} g(S_i) \ .$$

The second lemma shows that the marginal gain in the distorted objective is sufficiently large to ensure the desired approximation guarantees. This fact relies on the monotonicity and γ-weak submodularity of g.

Lemma 2. *In each iteration of* DISTORTED GREEDY,

$$\Psi_i(S_i, e_i) \geq \frac{\gamma}{k}\left(1 - \frac{\gamma}{k}\right)^{k-(i+1)} [g(OPT) - g(S_i)] - \frac{1}{k} c(OPT) \ .$$

Using these two lemmas, we present an approximation guarantee for DISTORTED GREEDY.

[1] We note that these two techniques can be traced back to the works of (Feldman, 2018) and (Mirzasoleiman et al., 2015), respectively.

Theorem 3. DISTORTED GREEDY *makes $O(nk)$ evaluations of g and returns a set R of size at most k with*

$$g(R) - c(R) \geq \left(1 - e^{-\gamma}\right) g(OPT) - c(OPT) \ .$$

Proof. Since c is modular and g is non-negative, the definition of Φ gives

$$\Phi_0(S_0) = \left(1 - \frac{\gamma}{k}\right)^k g(\varnothing) - c(\varnothing) \geq 0$$

and

$$\Phi_k\left(S_k\right) = \left(1 - \frac{\gamma}{k}\right)^0 g(S_k) - c(S_k) = g(S_k) - c(S_k) \ .$$

Using this and the fact that the returned set R is in fact S_k, we get

$$\begin{aligned} g(R) - c(R) &\geq \Phi_k(S_k) - \Phi_0(S_0) \qquad (4) \\ &= \sum_{i=0}^{k-1} \Phi_{i+1}(S_{i+1}) - \Phi_i(S_i) \ . \end{aligned}$$

Applying Lemmas 1 and 2, respectively, we have

$$\begin{aligned} &\Phi_{i+1}(S_{i+1}) - \Phi_i(S_i) \\ &= \Psi_i(S_i, e_i) + \frac{\gamma}{k}\left(1 - \frac{\gamma}{k}\right)^{k-(i+1)} g(S_i) \\ &\geq \frac{\gamma}{k}\left(1 - \frac{\gamma}{k}\right)^{k-(i+1)} \left[g(OPT) - g(S_i)\right] \\ &\quad - \frac{1}{k} c(OPT) + \frac{\gamma}{k}\left(1 - \frac{\gamma}{k}\right)^{k-(i+1)} g(S_i) \\ &= \frac{\gamma}{k}\left(1 - \frac{\gamma}{k}\right)^{k-(i+1)} g(OPT) - \frac{1}{k} c(OPT) \ . \end{aligned}$$

Finally, plugging this bound into (4) yields

$$\begin{aligned} &g(R) - c(R) \\ &\geq \sum_{i=0}^{k-1} \left[\frac{\gamma}{k}\left(1 - \frac{\gamma}{k}\right)^{k-(i+1)} g(OPT) - \frac{1}{k} c(OPT)\right] \\ &= \left[\frac{\gamma}{k} \sum_{i=0}^{k-1} \left(1 - \frac{\gamma}{k}\right)^i\right] g(OPT) - c(OPT) \\ &= \left(1 - \left(1 - \frac{\gamma}{k}\right)^k\right) g(OPT) - c(OPT) \\ &\geq \left(1 - e^{-\gamma}\right) g(OPT) - c(OPT) \ . \end{aligned}$$

□

3.2. Stochastic Distorted Greedy

Our second algorithm, STOCHASTIC DISTORTED GREEDY, is presented as Algorithm 2. It uses the same distorted objective as DISTORTED GREEDY, but enjoys an asymptotically faster run time due to sampling techniques of (Mirzasoleiman et al., 2015). Instead of optimizing over the entire ground set at each iteration, STOCHASTIC DISTORTED GREEDY optimizes over a random sample $B_i \subseteq \Omega$ of size $O\left(\frac{n}{k}\log\frac{1}{\epsilon}\right)$. This sampling procedure ensures that sufficient potential gain occurs *in expectation*, which is true for the following reason. If the sample size is sufficiently large, then B_i contains at least one element of OPT with high probability. Conditioned on this (high probability) event, choosing the element with the maximum potential gain is at least as good as choosing an average element from OPT.

Algorithm 2 STOCHASTIC DISTORTED GREEDY

Input: utility g, weak γ, cost c, cardinality k, error ϵ
Initialize $S_0 \leftarrow \varnothing$, $s \leftarrow \lceil \frac{n}{k}\log(\frac{1}{\epsilon})\rceil$
for $i = 0$ **to** $k-1$ **do**
  $B_i \leftarrow$ sample s elements uniformly & ind. from Ω
  $e_i \leftarrow \arg\max_{e \in B_i} \left\{\left(1 - \frac{\gamma}{k}\right)^{k-(i+1)} g(e \mid S_i) - c_e\right\}$
  if $\left(1 - \frac{\gamma}{k}\right)^{k-(i+1)} g(e_i \mid S_i) - c_{e_i} > 0$ **then**
    $S_{i+1} \leftarrow S_i \cup \{e_i\}$
  end if
end for

Lemma 4. *In each step of* STOCHASTIC DISTORTED GREEDY*,*

$$\begin{aligned} \mathbb{E}\left[\Psi_i(S_i, e_i)\right] \geq (1-\epsilon)\Bigg(\frac{\gamma}{k}\left(1 - \frac{\gamma}{k}\right)^{k-(i+1)} \big[g(OPT) \\ - \mathbb{E}\left[g(S_i)\right]\big] - \frac{1}{k} c(OPT)\Bigg) \ . \end{aligned}$$

Theorem 5. STOCHASTIC DISTORTED GREEDY *uses $O(n\log\frac{1}{\epsilon})$ evaluations of g and returns a set R with*

$$\mathbb{E}\left[g(R) - c(R)\right] \geq \left(1 - e^{-\gamma} - \epsilon\right) g(OPT) - c(OPT) \ .$$

3.3. Unconstrained Distorted Greedy

In this section, we present UNCONSTRAINED DISTORTED GREEDY, an algorithm for the unconstrained setting (i.e., $k = n$), listed as Algorithm 3. UNCONSTRAINED DISTORTED GREEDY samples a *single* random element at each iteration, adding it to the current solution if the potential gain is sufficiently large. Note that this algorithm is faster than the previous two, as it requires only $O(n)$ evaluations of g. As before, the algorithm relies on the distorted objective and the heart of the analysis is showing that the potential increase is sufficiently large in each iteration.

Lemma 6. *In each step of* UNCONSTRAINED DISTORTED GREEDY*,*

$$\begin{aligned} \mathbb{E}\left[\Psi_i(S_i, e_i)\right] \geq \frac{\gamma}{n}\left(1 - \frac{\gamma}{n}\right)^{n-(i+1)} \big[g(OPT) \\ - \mathbb{E}\left[g(S_i)\right]\big] - \frac{1}{n} c(OPT) \ . \end{aligned}$$

In the same way that Theorem 3 follows from Lemma 2,

Algorithm 3 UNCONSTRAINED DISTORTED GREEDY

Input: utility g, weak γ, cost c, cardinality k
Initialize $S_0 \leftarrow \varnothing$
for $i = 0$ **to** $n-1$ **do**
  $e_i \leftarrow$ sample uniformly from Ω
  if $\left(1-\frac{\gamma}{n}\right)^{n-(i+1)} g(e_i \mid S_i) - c_{e_i} > 0$ **then**
    $S_{i+1} \leftarrow S_i \cup \{e_i\}$
  end if
end for

the next theorem follows from Lemma 6 (and therefore, we omit its proof also from the appendix).

Theorem 7. UNCONSTRAINED DISTORTED GREEDY *requires $O(n)$ function evaluations and outputs a set R such that*

$$\mathbb{E}\left[g(R) - c(R)\right] \geq (1-e^{-\gamma})g(OPT) - c(OPT) \ .$$

3.4. Guessing Gamma: A Geometric Sweep

The previously described algorithms required knowledge of the submodularity ratio γ. However, it is very rare that the precise value of γ is known in practice—unless g is submodular, in which case $\gamma = 1$. Oftentimes, γ is data dependent and only a crude lower bound $L \leq \gamma$ is known. In this section, we describe a meta algorithm that "guesses" the value of γ. γ-SWEEP, listed as Algorithm 4, runs a maximization algorithm $\mathcal{A}$ as a subroutine with a geometrically decreasing sequence of "guesses" $\gamma^{(k)}$ for $k = 0, 1, \ldots, \lceil \frac{1}{\delta} \log \frac{1}{\delta} \rceil$. The best set obtained by this procedure is guaranteed to have nearly as good approximation guarantees as when the true submodularity ratio γ is known exactly. Moreover, fewer guesses are required if a good lower bound $L \leq \gamma$ is known, which is true for several problems of interest. In the following theorem, we assume that $\mathcal{A}(g, \gamma, c, k, \epsilon)$ is an algorithm which returns a set S with $|S| \leq k$ and $\mathbb{E}\left[g(S) - c(S)\right] \geq (1 - e^{-\gamma} - \epsilon)\, g(OPT) - c(OPT)$ when g is γ-weakly submodular, and $L \leq \gamma$ is known (one may always use $L = 0$).

Algorithm 4 γ-SWEEP

Input: utility g, cost c, alg. $\mathcal{A}$, lower bound L, $\delta \in (0,1)$
$S_{-1} \leftarrow \varnothing$, $T \leftarrow \left\lceil \frac{1}{\delta} \ln\left(\frac{1}{\max\{\delta, L\}}\right) \right\rceil$
for $r = 0$ **to** T **do**
  $\gamma_r \leftarrow (1-\delta)^r$
  $S_r \leftarrow \mathcal{A}(g, \gamma_r, c, k, \delta)$
end for
$R \leftarrow \arg\max_{r=-1,\ldots,T} \{g(S_r) - c(S_r)\}$

Theorem 8. γ-SWEEP *requires at most $O\left(\frac{1}{\delta}\log\frac{1}{\delta}\right)$ calls to $\mathcal{A}$ and returns a set R with*

$$\mathbb{E}\left[g(R) - c(R)\right] \geq \left(1 - e^{-\gamma} - O(\delta)\right) g(OPT) - c(OPT) \ .$$

In our experiments, we see that STOCHASTIC DISTORTED GREEDY combined with the γ-SWEEP outperforms the DISTORTED GREEDY with γ-SWEEP, especially for larger values of k. Here, we provide some experimental evidence and explanation for why this may be occurring. Figure 1 shows the objective value of the sets $\{S_r\}_{r=0}^{T}$ produced by STOCHASTIC DISTORTED GREEDY and DISTORTED GREEDY during the γ-SWEEP for cardinality constraints $k = 5$, 10, and 20. Both subroutines return the highest objective value for similar ranges of γ. However, the STOCHASTIC DISTORTED GREEDY subroutine appears to be better in two ways. First, the average objective value is usually larger, meaning that an individual run of STOCHASTIC DISTORTED GREEDY is returning a higher quality set than DISTORTED GREEDY. This is likely due to the sub-sampling of the ground set, which might help avoiding the picking of a single "bad element", if one exists. Second, the variation in STOCHASTIC DISTORTED GREEDY leads to a higher chance of producing a good solution. For many values of γ, the DISTORTED GREEDY subroutine returns a set of the same value; thus, the extra guesses of γ are not particularly helpful. On the other hand, the variation within STOCHASTIC DISTORTED GREEDY subroutine means that these extra guesses are not wasted; in fact, they allow a higher chance of producing a set with good value. Figure 1 also shows that the objective function throughout the sweep is fairly well-behaved, suggesting the possibility of early stopping heuristics. However, that is outside the scope of this paper.

4. Hardness Result

In this section, we give a hardness result which complements our algorithmic guarantees. The hardness result shows that—in the case where $c = 0$—no algorithm making polynomially many queries to g can achieve a better approximation ratio than $1 - e^{-\gamma}$. Although this was known in the case when $\gamma = 1$ (i.e., g is submodular), the more general result for $\gamma < 1$ was unknown until this work.

Theorem 9. *For every two constants $\varepsilon > 0$ and $\gamma \in (0, 1]$, no polynomial time algorithm achieves $(1 - e^{-\gamma} + \varepsilon)$-approximation for the problem of maximizing a non-negative monotone γ-weakly submodular function subject to a cardinality constraint in the value oracle model.*

As is usual in hardness proofs for submodular functions, the proof is based on constructing a family of γ-weakly submodular functions on which any deterministic algorithm

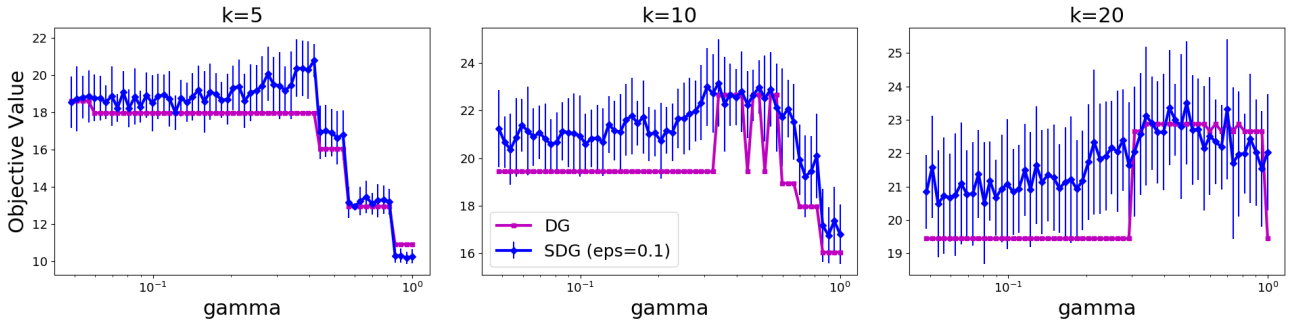


Figure 1: Results of the γ-SWEEP with DISTORTED GREEDY (DG) and STOCHASTIC DISTORTED GREEDY (SDG) as subroutines. For STOCHASTIC DISTORTED GREEDY, mean values with standard deviation bars are reported over 20 trials.

will perform poorly in expectation, and then applying Yao's principle. We defer details to Appendix C.

5. Experiments

To demonstrate the effectiveness of our proposed algorithms, we run experiments on two applications: Bayesian A-optimal design with costs and directed vertex cover with costs. The code was written using the `Julia` programming language, version 1.0.2. Experiments were run on a 2015 MacBook Pro with 3.1 GHz Intel Core i7 and 8 GB DDR3 SDRAM and the timing was reported using the `@timed` feature in `Julia`. [2]

5.1. Bayesian A-Optimal Design

We first describe the problem of Bayesian A-Optimal design. Suppose that $\theta \in \mathbb{R}^d$ is an unknown parameter vector that we wish to estimate from noisy linear measurements using least squares regression. Our goal is to choose a set S of linear measurements (the so-called experiments) which have low cost and also maximally reduce the variance of our estimate $\hat{\theta}$. More precisely, let $x_1, x_2, \dots x_n \in \mathbb{R}^d$ be a fixed set of measurement vectors, and let $X = [x_1, x_2, \dots x_n]$ be the corresponding $d \times n$ matrix. Given a set of measurement vectors $S \subseteq [n]$, we may run an experiment and obtain the noisy linear observation, $y_S = X_S^T \theta + \zeta_S$, where ζ_S is normal i.i.d. noise, i.e., $\zeta_1, \dots, \zeta_n \sim N(0, \sigma^2)$. We estimate θ using the least squares estimator $\hat{\theta} = (X_S X_S^T)^{-1} X_S^T y_S$. Assuming a normal Bayesian prior distribution on the unknown parameter, $\theta \sim N(0, \Sigma)$, the sum of the variance of the coefficients given the measurement set S is $r(S) = Tr\left(\Sigma^{-1} + \frac{1}{\sigma^2} X_S X_S^T\right)^{-1}$. We define $g(S) = r(\varnothing) - r(S)$ to be the *reduction in variance* produced by experiment set S. Bian et al. (2017) showed that g is γ-weakly submodular, providing a lower bound for γ in the case where $\Sigma = \beta I$. However, their bound relies rather unfavorably on the spectral norm of X, and does not extend to general Σ. Chamon & Ribeiro (2017) showed that g satisfies the stronger condition of γ-weak DR (Definition C.1), but their bound on the submodularity ratio γ depends on the cardinality of the sets. We give a tighter bound here, and the proof appears in Appendix D.

Claim 10. *g is a non-negative, monotone and γ-weakly submodular function with*

$$\gamma \geq \left(1 + \frac{s^2}{\sigma^2} \lambda_{\max}(\Sigma)\right)^{-1} ,$$

where $s = \max_{i \in [n]} \|x_i\|_2$.

Suppose that each experiment x_i has an associated non-negative cost c_i. In this application, we seek to maximize the "revenue" of the experiment,

$$g(S) - c(S) = Tr\left(\Sigma\right) - Tr\left(\Sigma^{-1} + \frac{1}{\sigma^2} X_S X_S^T\right)^{-1} - c(S) ,$$

which trades off the utility of the experiments (i.e., the variance reduction in the estimator) and their overall cost.

Unlike submodular functions, lazy evaluations (Minoux, 1978) of γ-weakly submodular g are generally not possible, as the marginal gains vary unpredictably. However, for specific functions, one can possibly speed up the greedy search. For the utility g considered here, we implemented a faster greedy search using the matrix inversion lemma. The naive approach of computing $g(e \mid S)$ by constructing $\Sigma^{-1} + X_S X_S^T$, explicitly computing its inverse, and summing the diagonal elements is not only expensive—inversion alone costs $O(d^3)$ arithmetic operations—but also memory-inefficient. Instead, one can use the matrix inversion lemma to show that

$$g(e \mid S) = \frac{\|z_e\|^2}{\sigma^2 + \langle x_e, z_e \rangle} ,$$

where $z_e = M_S^{-1} x_e$ and $M_S = \Sigma^{-1} + X_S X_S^T$. Moreover, M_S^{-1} may be *stored and updated directly* without any matrix

[2] Source code available at https://github.com/crharshaw/submodular-minus-linear

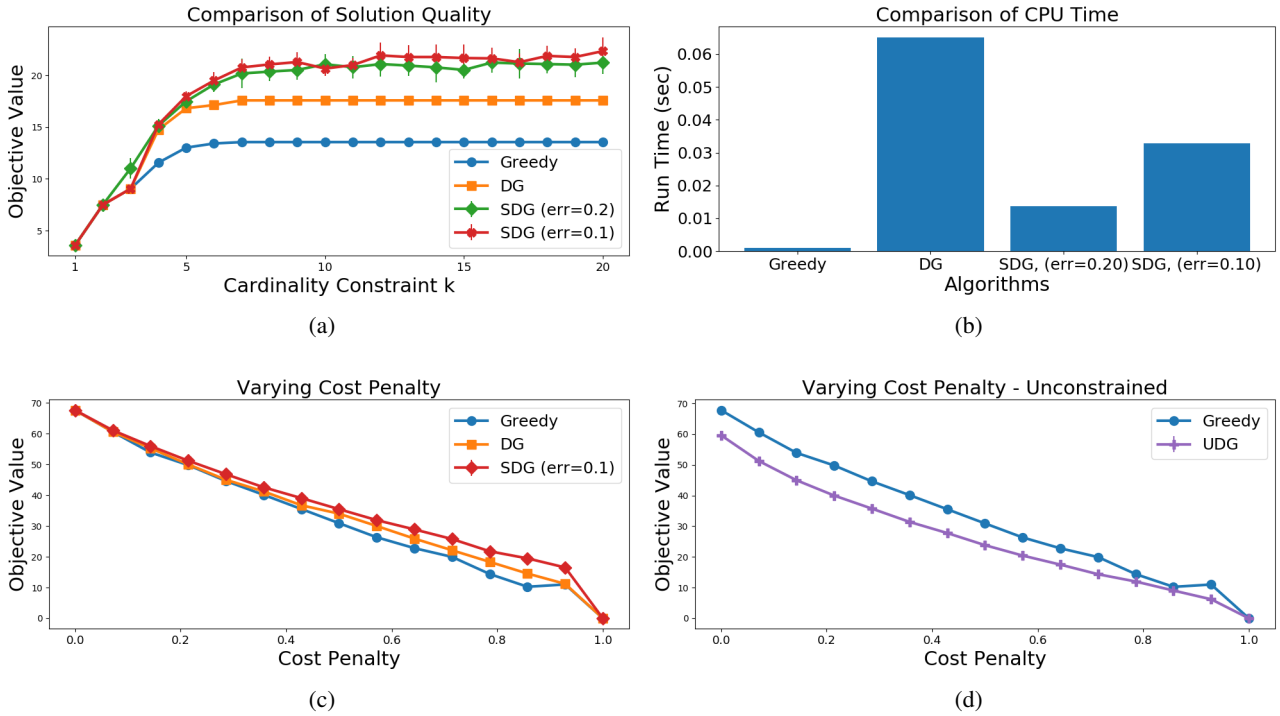


Figure 2: An algorithmic performance comparison for Bayesian A-Optimal design on the Boston Housing dataset. We report values for stochastic algorithms with mean and standard deviation bars, over 20 trials. (2a) objective values, varying the cardinality k, for a fixed cost penalty $\alpha = 0.8$. (2b) runtime for a fixed cardinality $k = 15$. (2c) objective values, varying the cost penalty α for a fixed cardinality $k = 15$. (2d) objective values, varying the cost penalty α in an unconstrained setting.

inversion. In this way, marginal gains $g(e \mid S)$ may be queried using only matrix-vector multiplication with a fixed M_S^{-1} and inner product computations, which requires $O(d^2)$ arithmetic operations and is more memory efficient. More details are given in Appendix D.

For this experiment, we used the Boston Housing dataset (Jr. & Rubenfield, 1978), a standard benchmark dataset containing $d = 14$ attributes of $n = 506$ Boston homes, including average number of rooms per dwelling, proximity to the Charles River, and crime rate per capita. We preprocessed the data by normalizing the features to have a zero mean and a standard deviation of 1. As there is no specified cost per measurement, we assigned costs proportionally to initial marginal gains in utility; that is, $c_e = \alpha g(e)$ for some $\alpha \in [0, 1]$. We set $\sigma = 1/\sqrt{d}$, and randomly generated a normal prior with covariance $\Sigma = ADA^T$, where A is randomly chosen as $A_{i,j} \sim N(0, 1)$ and D is diagonal with $D_{i,i} = (i/d)^2$. We choose not to use $\Sigma = \beta I$, as we found this causes g to be nearly modular along solution paths, yielding it an easy problem instance for all algorithms and not a suitable benchmark.

In our first experiment, we fixed the cost penalty $\alpha = 0.8$, and ran the algorithms for varying cardinality constraints from $k = 1$ to $k = 15$. We ran the greedy algorithm, DISTORTED GREEDY with γ-SWEEP (setting $\delta = 0.1$), and two instances of STOCHASTIC DISTORTED GREEDY with γ-SWEEP (with $\delta = \epsilon = 0.1$ and $\delta = \epsilon = 0.05$). All γ-SWEEP runs used $L = 0$. In Figure 2a, one can observe that the marginal gain obtained by the greedy algorithm is not non-increasing (at least for the first few elements), which is a result of the fact that g is weakly submodular with $\gamma < 1$. For small values of k, all algorithms produce comparable solutions; however, the greedy algorithm gets stuck in a local maximum of size $k = 7$, while our algorithms are able to produce larger solutions with higher objective value. Moreover, γ-SWEEP with STOCHASTIC DISTORTED GREEDY performs better than γ-SWEEP with DISTORTED GREEDY for larger values of k, for reasons discussed in Section 3.4. Figure 2b shows CPU times of each algorithm run with the single cardinality constraint $k = 20$. We see that the greedy algorithm runs faster than our algorithms. This difference in the runtime is a result of both the added complexity of the γ-SWEEP procedure, and that greedy terminates early, when a local maximum is reached. Figure 2b also shows that the sub-sampling step in STOCHASTIC DISTORTED GREEDY results in a faster runtime than DISTORTED GREEDY, as predicted by the theory. We did not display the number of function evaluations, as it exhibits nearly identical trends to the actual CPU run time. In our next experiment, we fixed

the cardinality $k = 15$ and varied the cost penalty $\alpha \in [0, 1]$. Figure 2c shows that all algorithm return similar solutions for $\alpha = 0$ and $\alpha = 1$, which are the cases in which either $c = 0$ or the function $g - c$ is non-positive, respectively. For all other values of α, our algorithms yield improvements over greedy. In our final experiment, we varied the cost penalty $\alpha \in [0, 1]$, comparing the output of greedy and γ-SWEEP with UNCONSTRAINED DISTORTED GREEDY for the unconstrained setting. Figure 2d shows that greedy outperforms our algorithm in this instance, which can occur, especially in the absence of "bad elements" discussed in Section 3.

5.2. Directed Vertex Cover with Costs

The second experiment is directed vertex cover with costs. Let $G = (V, E)$ be a directed graph and let $w : V \to \mathbb{R}$ be a weight function on the vertices. For a vertex set $S \subseteq V$, let $N(S)$ denote the set of vertices which are pointed to by S, $N(S) \triangleq \{v \in V \mid (u, v) \in E \text{ for some } u \in S\}$. The weighted directed vertex cover function is $g(S) = \sum_{u \in N(S) \cup S} w_u$. We also assume that each vertex $v \in V$ has an associated nonnegative cost c_v. Our goal is to maximize the resulting revenue,

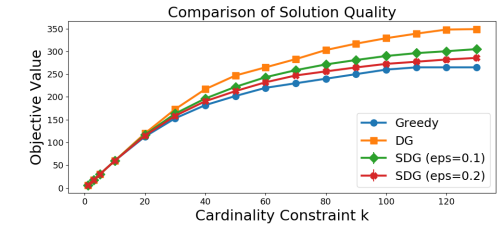
$$g(S) - c(S) = \sum_{u \in N(S) \cup S} w_u - \sum_{u \in S} c_u \ .$$

Because g is submodular, we can forgo the γ-SWEEP routine and apply our algorithms directly with $\gamma = 1$. Moreover, we implement lazy evaluations of g in our code.

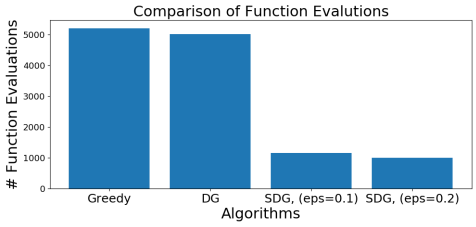
For our experiments, we use the EU Email Core network, a directed graph generated using email data from a large European research institution (Yin et al., 2017; Leskovec et al., 2007). The graph has 1k nodes and 25k directed edges, where nodes represent people and a directed edge from u to v means that an email was sent from u to v. We assign each node a weight of 1. Additionally, as there are no costs in the dataset, we assign costs in the following manner. For a fixed q, we set $c(v) = 1 + \max\{d(v) - q, 0\}$, where $d(v)$ is the out-degree of v. Thus, all vertices with out-degree larger than q have the same initial marginal gain $g(v) - c(v) = q$.

In our first experiment, we fixed the cost factor $q = 6$, and ran the algorithms for varying cardinality constraints from $k = 1$ to $k = 130$. We see in Figure 3a that our methods outperform greedy. DISTORTED GREEDY achieves the highest objective value for each cardinality constraint, while STOCHASTIC DISTORTED GREEDY achieves higher objective values as the accuracy parameter ϵ is decreased. Figure 3b shows the number of function evaluations made by the algorithms when $k = 130$. We observe that STOCHASTIC DISTORTED GREEDY requires much fewer function evaluations, even when lazy evaluations are implemented.[3]

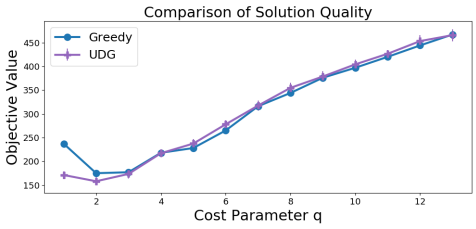
[3]We do not report CPU time, which does not reflect function evaluations here. This is due to the lazy evaluation implementation.



(a)



(b)



(c)

Figure 3: A performance comparison for directed vertex cover on the EU Email Core network. We report values for stochastic algorithms with mean and standard deviation bars, over 20 trials. (3a) objective values, varying the cardinality k, for a fixed cost factor $q = 6$. (3b) g evaluations for a fixed cardinality $k = 130$. (3c) objective values, varying the cost factor q in an unconstrained setting.

Finally, we ran greedy and UNCONSTRAINED DISTORTED GREEDY while varying the cost factor q from 1 to 12, and we note that in this setting (as can be seen in Figure 3c) our algorithm performs similarly to the greedy algorithm.

6. Conclusion

We presented a suite of fast algorithms for maximizing the difference between a non-negative monotone γ-weakly submodular g and a non-negative modular c in both the cardinality constrained and unconstrained settings. Moreover, we gave a matching hardness result showing that no algorithm can do better with only polynomially many oracle queries to g. Finally, we experimentally validated our algorithms on Bayesian A-Optimality and directed vertex cover with costs, demonstrating that they outperform the greedy heuristic.

Acknowledgments

Christopher Harshaw was supported in part by National Science Foundation GRFP (DGE1122492) and by ONR Award N00014-16-1-2374. Moran Feldman was supported in part by Israel Science Foundation (ISF) grant number 1357/16. Amin Karbasi was supported by AFOSR Young Investigator Award (FA9550-18-1-0160) and Grainger Award (PO 2000008083 2016012).

References

Bian, A. A., Buhmann, J. M., Krause, A., and Tschiatschek, S. Guarantees for greedy maximization of non-submodular functions with applications. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pp. 498–507, 2017.

Boykov, Y., Veksler, O., and Zabih, R. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11): 1222–1239, 2001.

Buchbinder, N. and Feldman, M. Constrained submodular maximization via a non-symmetric technique. *CoRR*, abs/1611.03253, 2016.

Chamon, L. F. O. and Ribeiro, A. Approximate supermodularity bounds for experimental design. In *Advances in Neural Information Processing Systems*, 2017.

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

Chvátal, V. The tail of the hypergeometric distribution. *Discrete Mathematics*, 25(3):285–287, 1979.

Das, A. and Kempe, D. Submodular meets Spectral: Greedy Algorithms for Subset Selection, Sparse Approximation and Dictionary Selection. In *International Conference on Machine Learning*, pp. 1057–1064, 2011.

Elenberg, E. R., Dimakis, A. G., Feldman, M., and Karbasi, A. Streaming weak submodularity: Interpreting neural networks on the fly. In *Advances in Neural Information Processing Systems*, 2017.

Elenberg, E. R., Khanna, R., Dimakis, A. G., and Negahban, S. Restricted strong convexity implies weak submodularity. *Annals of Statistics*, 46, 2018.

Ene, A. and Nguyen, H. L. Constrained submodular maximization: Beyond 1/e. In *FOCS*, pp. 248–257, 2016.

Feldman, M. Guess free maximization of submodular and linear sums. Technical Report arXiv:1810.03813v1 [cs.DS], ArXiV, October 2018.

Feldman, M., Harshaw, C., and Karbasi, A. Greed is good: Near-optimal submodular maximization via greedy optimization. In *COLT*, pp. 758–784, 2017.

Golovin, D. and Krause, A. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42:427–486, 2011.

Hoeffding, W. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 1963.

Hu, H., Grubb, A., Bagnell, J. A., and Hebert, M. Efficient feature group sequencing for anytime linear prediction. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, 2016.

Jegelka, S. and Bilmes, J. Submodularity beyond submodular energies: coupling edges in graph cuts. In *Computer Vision and Pattern Recognition (CVPR)*, pp. 1897–1904. IEEE, 2011.

Jr., D. H. and Rubenfield, D. L. Hedonic housing prices and the demand for clean air. *J. of Environmental Economics and Management*, 5(1):81–102, 1978.

Kempe, D., Kleinberg, J., and Tardos, E. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 137–146. ACM, 2003.

Khanna, R., Elenberg, E. R., Dimakis, A. G., Negahban, S., and Ghosh, J. Scalable Greedy Feature Selection via Weak Submodularity. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 1560–1568, 2017.

Krause, A. and Guestrin, C. Near-optimal Nonmyopic Value of Information in Graphical Models. In *Uncertainty in Artificial Intelligence (UAI)*, pp. 324–331, 2005.

Kuhnle, A., Smith, J. D., Crawford, V. G., and Thai, M. T. Fast maximization of non-submodular, monotonic functions on the integer lattice. In *ICML*, pp. 2791–2800, 2018.

Lee, J., Sviridenko, M., and Vondrák, J. Submodular maximization over multiple matroids via generalized exchange properties. *Math. Oper. Res.*, 35(4):795–806, 2010.

Leskovec, J., Kleinberg, J., and Faloutsos, C. Graph evolution: Densification and shrinking diameters. *ACM Trans. Knowl. Discov. Data*, 1(1), 2007.

Lin, H. and Bilmes, J. A class of submodular functions for document summarization. In *Proceedings of the 49th*

Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1, pp. 510–520. Association for Computational Linguistics, 2011.

Minoux, M. Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization Techniques*, pp. 234–243, 1978.

Mirzasoleiman, B., Badanidiyuru, A., Karbasi, A., Vondrák, J., and Krause, A. Lazier than lazy greedy. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pp. 1812–1818, 2015.

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 functions–I. *Mathematical Programming*, 14(1):265–294, 1978.

Skala, M. Hypergeometric tail inequalities: ending the insanity. *CoRR*, abs/1311.5939, 2013.

Sviridenko, M. A note on maximizing a submodular set function subject to a knapsack constraint. *Oper. Res. Lett.*, 32(1):41–43, 2004.

Sviridenko, M., Vondrák, J., and Ward, J. Optimal approximation for submodular and supermodular optimization with bounded curvature. *Math. Oper. Res.*, 42(4):1197–1218, 2017.

Wei, K., Liu, Y., Kirchhoff, K., and Bilmes, J. Using Document Summarization Techniques for Speech Data Subset Selection. In *Proceedings of NAACL-HLT 2013*, pp. 721726, 2013.

Yin, H., Benson, A. R., Leskovec, J., and Gleich, D. F. Local higher-order graph clustering. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2017.