RIMS Kôkyûroku Bessatsu
Bx (200x), 000–000

Submodularity and curvature: the optimal algorithm

By

Jan Vondrák[*]

Abstract

Let $(X, \mathcal{I})$ be a matroid and let $f : 2^X \to \mathcal{R}_+$ be a monotone submodular function. The curvature of f is a parameter $c \in [0,1]$ such that for any $S \subset X$ and $j \in X \setminus S$, $f(S \cup \{j\}) - f(S) \geq (1-c) f(\{j\})$. We consider the optimization problem $\max\{f(S) : S \in \mathcal{I}\}$.

It is known that the greedy algorithm yields a 1/2-approximation for this problem [11], and $\frac{1}{1+c}$-approximation when f has curvature c [4]. For the uniform matroid, it was known that the greedy algorithm yields an improved $\frac{1}{c}(1-e^{-c})$-approximation [4].

In this paper, we analyze the continuous greedy algorithm [23] and prove that it gives a $\frac{1}{c}(1-e^{-c})$-approximation for any matroid. Moreover, we show that this holds for a relaxed notion of curvature, *curvature with respect to the optimum*, and we prove that any better approximation under these conditions would require an exponential number of value queries.

§ 1. Introduction

In this paper, we consider the following optimization problem:

$$\max\{f(S) : S \in \mathcal{I}\}$$

where $f : 2^X \to \mathcal{R}_+$ is a monotone submodular function, and $\mathcal{I} \subset 2^X$ is the collection of independent sets in a matroid. A function f is *monotone* if $f(A) \leq f(A')$ for any $A \subset A'$, and f is *submodular* if $f(A \cup B) + f(A \cap B) \leq f(A) + f(B)$ for all A, B. For the definition of a matriod, see Section 2. For computational purposes we will assume that f and $\mathcal{I}$ are specified by value/membership oracles, which can be queried polynomially many times. We call this framework the *value oracle model.*

The problem of maximizing a submodular set function subject to independence constraints has been studied extensively [14, 5, 16, 19, 11, 20, 24, 25, 4]. A number

Received April 20, 200x. Revised September 11, 200x.
2000 Mathematics Subject Classification(s): 68Q25
Key Words: *submodular function, curvature, matroid, approximation algorithms*
[*]Princeton University, Princeton, NJ 08544.
e-mail: `jvondrak@gmail.com`

of interesting and useful combinatorial optimization problems are special cases. Some classical examples are Maximum-Weight Independent Set in a matroid, Matroid Intersection, and Max-k-cover. A more recent application is the Submodular Welfare Problem [17, 6, 9, 23] which arises in combinatorial auctions.

The Greedy Algorithm. The "greedy algorithm" incrementally builds a solution (without backtracking) starting with the empty set. In each iteration it adds an element that most improves the current solution (according to f) while maintaining the condition $S \in \mathcal{I}$.

In classical work, Nemhauser, Wolsey and Fisher proved that this algorithm yields a $(1-1/e)$-approximation for the problem $\max\{f(S) : S \in \mathcal{I}\}$ when f is monotone submodular and $(X, \mathcal{I})$ is the uniform matroid, $\mathcal{I} = \{S \subseteq X : |S| \leq k\}$ [19]. This approximation factor is optimal in the value oracle model [20] and it is optimal even for the explicit special case of Max-k-cover, unless $P = NP$ [8]. In the case of a general matroid $(X, \mathcal{I})$, however, the greedy algorithm gives only a $1/2$-approximation [11]. The optimal $(1-1/e)$-approximation for any matroid has been achieved recently using a *continuous greedy algorithm* and *pipage rounding* [3, 23].

Curvature. Submodular functions can be defined alternatively using the notion of *marginal values.* We denote by $f_S(j) = f(S \cup \{j\}) - f(S)$ the marginal value of j with respect to S. It is known that submodularity is equivalent to $f_T(j) \leq f_S(j)$ for any $j \notin T \supset S$.

The notion of curvature reflects how much the marginal values $f_S(j)$ can decrease as a function of S. The *total curvature* of f is defined as

$$c = 1 - \min_{S, j \notin S} \frac{f_S(j)}{f_\emptyset(j)}.$$

Note that $c \in [0,1]$, and if $c = 0$ then the marginal values are independent of S (i.e. f is additive). In this case, it is known that the greedy algorithm returns the optimal solution to $\max\{f(S) : S \in \mathcal{I}\}$. An analysis of the greedy algorithm depending on c was given by Conforti and Cornuéjols [4]. The greedy algorithm gives a $\frac{1}{1+c}$-approximation to $\max\{f(S) : S \in \mathcal{I}\}$ when f has total curvature c. In case of the uniform matroid $\mathcal{I} = \{S : |S| \leq k\}$, the approximation factor is $\frac{1}{c}(1 - e^{-c})$ [4].

Our results. In this paper, we analyze the *continuous greedy algorithm* introduced in [23]. We show that given total curvature c, the continuous greedy algorithm yields a $\frac{1}{c}(1 - e^{-c})$-approximation for the problem $\max\{f(S) : S \in \mathcal{I}\}$, for any matroid. This matches the previously known result for uniform matroids.

In fact, we show that the same approximation guarantee holds under a weaker assumption, namely when f has curvature c *with respect to the optimum.* We define this notion formally below; roughly, this means that for (some) optimum solution S^*, the value of any set T added on top of S^* is still at least $(1-c)f(T)$. This condition is implied by total curvature c.

Finally, we show that given curvature c with respect to the optimum, it is impossible to achieve an approximation better than $\frac{1}{c}(1-e^{-c})$ using a polynomial number of value queries. Thus, the continuous greedy algorithm is optimal in the value oracle model, for any fixed curvature $c \in [0,1]$ with respect to the optimum. The hardness result follows along the lines of [18] where it is proved that the factor $1-1/e$ is optimal for Submodular Welfare in the value oracle model.

Submodular Welfare. Our results have a natural interpretation for the Submodular Welfare Problem, which is a special case of this framework [17, 23]. In this problem, we are given n players and m items. Each player i has a submodular utility function $w_i : 2^{[m]} \to \mathcal{R}_+$. The goal is to allocate items to maximize the total utility $\sum_{i=1}^{n} w_i(S_i)$. An optimal $(1-1/e)$-approximation for this problem (in the value oracle model) has been shown recently [23].

Essentially, our results show that the approximation factor $1-1/e$ is tight only in a special situation, where any optimal allocation makes the players completely satisfied. If, however, items still retain significant value on top of some optimal allocation, the continuous greedy algorithm achieves a better approximation factor. Assuming that marginal values of sets with respect to the optimal allocation decrease by a factor of at most $1-c$, we achieve an improved approximation factor $\frac{1}{c}(1-e^{-c})$, and this is optimal.

§ 2. Preliminaries

Submodular functions. A function $f : 2^X \to \mathcal{R}$ is submodular if for all $A, B \subseteq X$,

$$f(A \cup B) + f(A \cap B) \leq f(A) + f(B).$$

In the following, we assume that $f(\emptyset) = 0$ and f is monotone. Given $A \subset X$, the function f_A defined by $f_A(S) = f(S \cup A) - f(A)$ is also submodular. By $f_A(i)$, we denote the "marginal value" $f(A \cup \{i\}) - f(A)$. We also use $f(i)$ to denote $f_\emptyset(i) = f(\{i\})$. For monotone functions, submodularity is equivalent to $f_A(i)$ being non-increasing as a function of A for every fixed i.

Curvature. The total curvature of a monotone submodular function f is

$$c = 1 - \min_{S, j \notin S} \frac{f_S(j)}{f_\emptyset(j)}.$$

In this paper, we also define curvature with respect to a set S as follows. For another set T, we consider the multiset $S+T$ and define its value by

$$f(S+T) = f(S \cup T) + \sum_{j \in S \cap T} f_{S \cup T \setminus \{j\}}(j).$$

I.e., for the duplicate elements in $S+T$, we add their marginal values with respect to $S \cup T$.

Then, we say that f has *curvature c with respect to S*, if $c \in [0,1]$ is the smallest value such that for every T,

$$f(S+T) - f(S) \geq (1-c) f(T).$$

It is easy to see that if f has total curvature c, then it has curvature at most c with respect to any S:

$$f(S+T)-f(S) = (f(S\cup T)-f(S)) + \sum_{j \in S \cap T} f_{S \cup T \setminus \{j\}}(j) \geq \sum_{j \in T} f_{S \cup T \setminus \{j\}}(j) \geq (1-c) \sum_{j \in T} f(j).$$

We will work mostly with the weaker notion of curvature c with respect to S; nevertheless, all our positive results also hold for total curvature c.

Smooth submodular functions. For a monotone submodular set function $f : 2^X \to \mathcal{R}_+$, a canonical extension to a smooth function $F : [0,1]^X \to \mathcal{R}_+$ can be obtained as follows [3]:

For $y \in [0,1]^X$, let $\hat{y}$ denote a random vector in $\{0,1\}^X$ where each coordinate is independently rounded to 1 with probability y_j or 0 otherwise. Let R be the associated random subset of X, such that $\hat{y} = \mathbf{1}_R$. Then, define

$$F(y) = \mathbf{E}[f(R)] = \sum_{R \subseteq X} f(R) \prod_{i \in R} y_i \prod_{j \notin R} (1-y_j).$$

This is a multilinear polynomial with the following properties (see [23]):

- $\frac{\partial F}{\partial y_j} = \mathbf{E}[f(R \cup \{j\}) - f(R \setminus \{j\})] \geq 0$ (monotonicity).
- $\frac{\partial^2 F}{\partial y_i \partial y_j} = \mathbf{E}[f(R \cup \{i,j\}) - f(R \cup \{j\} \setminus \{i\}) - f(R \cup \{i\} \setminus \{j\}) + f(R \setminus \{i,j\})] \leq 0$ (submodularity).
- $\frac{\partial^2 F}{\partial {y_j}^2} = 0$ (multilinearity).

Thus the *gradient* $\nabla F = (\frac{\partial F}{\partial y_1}, \ldots, \frac{\partial F}{\partial y_n})$ is a nonnegative vector. The submodularity condition $\frac{\partial^2 F}{\partial y_i \partial y_j} \leq 0$ means that the coordinates of ∇F are non-increasing with respect to any y_j.

Matroids. A matroid is a pair $\mathcal{M} = (X, \mathcal{I})$ where $\mathcal{I} \subseteq 2^X$ and

1. $\forall B \in \mathcal{I}, A \subset B \Rightarrow A \in \mathcal{I}$.
2. $\forall A, B \in \mathcal{I}; |A| < |B| \Rightarrow \exists x \in B \setminus A; A + x \in \mathcal{I}$.

Matroid polytopes. For a matroid $\mathcal{M} = (X, \mathcal{I})$, the matroid polytope is defined as

$$P(\mathcal{M}) = \text{conv } \{\mathbf{1}_I : I \in \mathcal{I}\}.$$

As shown by Edmonds [7], an equivalent description is

$$P(\mathcal{M}) = \{x \geq 0 : \forall S \subseteq X; \sum_{j \in S} x_j \leq r_M(S)\}.$$

Here, $r_M(S) = \max\{|I| : I \subseteq S \ \& \ I \in \mathcal{I}\}$ is the *rank function* of matroid $\mathcal{M}$.

An important property for us is that it is easy to maximize linear functions over $P(\mathcal{M})$. This boils down to finding the maximum-weight independent set in a matroid, which can be done by the greedy algorithm.

§ 3. The continuous greedy algorithm

The continuous greedy algorithm [23] works with a nonlinear relaxation of the problem,

$$\max\{F(y) : y \in P(\mathcal{M})\},$$

and relies on the fact that any solution $y \in P(\mathcal{M})$ can be rounded to a discrete solution $S \in \mathcal{I}$ such that $f(S) \geq F(y)$ (*pipage rounding*, see [3]). The continuous greedy algorithm can be viewed as a particle starting at $y(0) = 0$ and following a certain flow over a unit time interval:

$$\frac{dy}{dt} = v(y),$$

where $v(y)$ is defined as

$$v(y) = \text{argmax}_{v \in P(\mathcal{M})}(v \cdot \nabla F(y)).$$

The following algorithm is a discrete version of the continuous greedy process [23].

Algorithm **ContinuousGreedy**$(f, \mathcal{M})$:

1. Fix a small $\delta > 0$, and start with $t = 0$, $y(0) = \mathbf{0}$.
2. Let $R(t)$ contain each j independently with probability $y_j(t)$.
   For each $j \in X$, estimate by sampling

$$\omega_j = \frac{\partial F}{\partial y_j} = \mathbf{E}[f(R(t) \cup \{j\}) - f(R(t) \setminus \{j\})].$$

3. Let $v(t) = \text{argmax}_{v \in P(\mathcal{M})}(v \cdot \nabla F)$. We can find this by the greedy algorithm for max-weight independent set with weights ω_j. Set
$$y(t+\delta) = y(t) + \delta v(t).$$

4. Increment $t := t + \delta$; if $t < 1$, go back to Step 2. Otherwise, return $y(1)$.

The fractional solution found by the continuous greedy algorithm can be converted into a discrete one using pipage rounding. Details can be found in [3]. The crucial lemma that allows us to analyze the performance of the algorithm is the following.

Lemma 3.1. *Let $OPT = \max_{S \in \mathcal{I}} f(S)$, and S^* a respective optimal solution. Assume that f has curvature c with respect to S^*. Consider any $y \in [0,1]^X$. Then*
$$\max_{v \in P(\mathcal{M})} (v \cdot \nabla F) \geq OPT - c\ F(y).$$

Proof. We prove the assertion by exhibiting a particular vector $v^* = \mathbf{1}_{S^*}$. We get
$$\begin{aligned} v^* \cdot \nabla F = \sum_{j \in S^*} \frac{\partial F}{\partial y_j} &= \sum_{j \in S^*} \mathbf{E}[f(R \cup \{j\}) - f(R \setminus \{j\})] \\ &= \sum_{j \in S^* \setminus R} \mathbf{E}[f_R(j)] + \sum_{j \in S^* \cap R} \mathbf{E}[f_{R \setminus \{j\}}(j)] \\ &\geq \mathbf{E}[f_R(S^* \setminus R)] + \sum_{j \in S^* \cap R} \mathbf{E}[f_{S^* \cup R \setminus \{j\}}(j)] \\ &= \mathbf{E}[f(S^* \cup R)] - \mathbf{E}[f(R)] + \sum_{j \in S^* \cap R} \mathbf{E}[f_{S^* \cup R \setminus \{j\}}(j)] \\ &= \mathbf{E}[f(S^* + R)] - \mathbf{E}[f(R)]. \end{aligned}$$

Now using the definition of curvature with respect to S^*,
$$\begin{aligned} v^* \cdot \nabla F &\geq \mathbf{E}[f(S^* + R)] - \mathbf{E}[f(R)] \geq \mathbf{E}[f(S^*) + (1-c)f(R)] - \mathbf{E}[f(R)] \\ &= OPT - c\ \mathbf{E}[f(R)] = OPT - c\ F(y). \end{aligned}$$

□

Given this lemma, the behavior of the continuous greedy process is easy to predict. We obtain the following differential equation:
$$\frac{dF}{dt} = v(t) \cdot \nabla F = \max_{v \in P(\mathcal{M})} v \cdot \nabla F \geq OPT - c\ F(y(t)).$$

The solution is $F(y(t)) \geq \frac{1}{c}(1 - e^{-ct})OPT$. The process stops at time $t = 1$, when we obtain $F(y(1)) \geq \frac{1}{c}(1 - e^{-c})OPT$. More rigorously, we prove the following result.

Theorem 3.2. *Let $OPT = \max_{S \in \mathcal{I}} f(S)$, and S^* an optimal solution. Assume that f has curvature $c \in (0,1)$ with respect to S^*. Then* **ContinuousGreedy** *with $\delta = 1/n^2$ finds a fractional solution of value*

$$F(y(1)) \geq \frac{1}{c}(1 - e^{-c} - o(1))OPT.$$

Proof. We start with $F(y(0)) = 0$. Our goal is to estimate how much $F(y(t))$ increases during one step of the algorithm. Recall that $F(y(t+\delta)) = F(y(t) + \delta v)$. By the main theorem of analysis,

$$F(y(t) + \delta v) - F(y(t)) = \int_0^\delta v \cdot \nabla F(y + \xi v) d\xi.$$

We want to argue that $\nabla F(y + \xi v)$ cannot be far away from $\nabla F(y)$, which is what we would really like to have. For this purpose, we estimate

$$\begin{aligned} \left|\frac{\partial^2 F}{\partial y_i \partial y_j}\right| &= \Big|\mathbf{E}[f(R \cup \{i,j\}) - f(R \cup \{i\} \setminus \{j\}) - f(R \cup \{j\} \setminus \{i\}) + f(R \setminus \{i,j\})]\Big| \\ &\leq \min\{f(i), f(j)\} \leq OPT. \end{aligned}$$

(We assume that each singleton is a feasible solution, otherwise we can remove it from the instance.) Therefore,

$$\frac{\partial F}{\partial y_j}\Big|_{y+\xi v} \geq \frac{\partial F}{\partial y_j}\Big|_{y} - n\xi \; OPT$$

and

$$F(y(t+\delta)) - F(y(t)) = \int_0^\delta v \cdot \nabla F(y + \xi v) d\xi \geq \delta v \cdot \nabla F(y) - \frac{1}{2} n \delta^2 OPT.$$

The way we chose the vector v was to maximize $v \cdot \nabla F$ over $v \in P(\mathcal{M})$. Recall that our estimate of ∇F might not have been exact, but we can make it accurate enough so that we find the maximum within an error of $\frac{1}{2} n\delta \; OPT$. Hence, by Lemma 3.1, $v \cdot \nabla F \geq OPT - cF(y) - \frac{1}{2} n\delta OPT$ and

$$F(y(t+\delta)) - F(y(t)) \geq \delta \; OPT - c\delta F(y(t)) - n\delta^2 OPT = c\delta(\tilde{OPT} - F(y(t)))$$

where $\tilde{OPT} = \frac{1}{c}(1 - n\delta)OPT$. Then we can rewrite the inequality as a recurrence:

$$\tilde{OPT} - F(y(t+\delta)) \leq (1 - c\delta)(\tilde{OPT} - F(y(t)))$$

which leads to

$$\tilde{OPT} - F(y(k\delta)) \leq (1 - c\delta)^k \tilde{OPT}.$$

Setting $k = 1/\delta$, we get

$$F(y(1)) \geq (1 - (1 - c/k)^k)\tilde{OPT} \geq (1 - e^{-c})\tilde{OPT}.$$

For $\delta = 1/n^2$, we get $\tilde{OPT} = \frac{1}{c}(1 - o(1))OPT$ and the desired result. $\square$

§ 4. Proof of optimality

In this section, we prove that the continuous greedy algorithm is optimal for any fixed curvature c with respect to the optimum.

Theorem 4.1. *For any fixed $c \in (0, 1)$ and $\beta > 0$, any $\frac{1+\beta}{c}(1-e^{-c})$-approximation algorithm for maximizing a submodular function subject to a matriod constraint, on the class of functions of curvature c with respect to the optimum, would require exponentially many value queries.*

The proof is inspired by the proof of optimality of the $1 - 1/e$ approximation factor for Submodular Welfare which appeared in [18]. This proof is also valid for maximizing a submodular function subject to a uniform matroid, and the same holds for our proof. Since some of the technicalities are the same as in [18], we give a more cursory description here.

Proof. We consider a ground set of $n = k\ell$ elements, divided into k blocks $X_1, X_2, \ldots, X_k$ of size ℓ. We work with continuous functions $\phi : [0, 1]^k \to \mathcal{R}_+$, which are associated with discrete functions $f : 2^X \to \mathcal{R}_+$ in the following way:

$$f(S) = \phi\left(\frac{|S \cap X_1|}{|X_1|}, \frac{|S \cap X_2|}{|X_2|}, \ldots, \frac{|S \cap X_k|}{|X_k|}\right).$$

If the partial derivatives of ϕ satisfy $\frac{\partial\phi}{\partial y_j} \geq 0$ and $\frac{\partial^2\phi}{\partial y_i \partial y_j} \leq 0$ for all i, j, then the associated f is monotone and submodular. We use this fact and work in the continuous world which is more convenient than the discrete one.

The basis of our proof lies in two instances which are hard to distinguish. Consider the following two functions:

- $f(x_1, \ldots, x_k) = 1 - \prod_{i=1}^{k}(1 - x_i)$.
- $g(x_1, \ldots, x_k) = 1 - (1 - \bar{x})^k$ where $\bar{x} = \frac{1}{k}\sum_{i=1}^{k} x_i$.

Let $D = \{x \in [0, 1]^k : x_1 = x_2 = \ldots = x_k\}$. The important fact is that the two functions are not only equal for any $x \in D$, but moreover $\nabla f = \nabla g$ on D. Therefore, it is possible to modify f slightly so that it is *exactly equal* to g in the close vicinity of D, and does not differ significantly from the original f anywhere. The details of this construction can be found in [18]; we summarize it here as follows.

Construction. Let $h(x) = f(x) - g(x) = (1-\bar{x})^k - \prod_{i=1}^{k}(1-x_i)$. Fix $\epsilon > 0$, $\epsilon_1 = k\epsilon$, $\epsilon_2 = \sqrt{\epsilon_1}$ and $\alpha = 2/\ln(1/\epsilon_1)$. We define

$$\hat{f}(x) = f(x_1, \dots, x_n) - \phi(h(x)) + a\, g(x)$$

where $a = 2\alpha k^6$ and ϕ is defined as follows:

- For $t \in [0, \epsilon_1]$, $\phi(t) = t$.
- For $t \in [\epsilon_1, \epsilon_2]$, $\phi'(t) = 1 - \alpha \ln(t/\epsilon_1)$.
- For $t \in [\epsilon_2, \infty)$, $\phi(t) = \phi(\epsilon_2)$.

The function $\hat{f}$ has the following properties which follow from [18]. The parameters in the lemma can be chosen as follows: ϵ as above, $a = 2\alpha k^6$ as above, and $\delta = \phi(\epsilon_2)$.

Lemma 4.2. *Let $f(x_1, \dots, x_k) = 1 - \prod_{i=1}^{k}(1-x_i)$, $g(x_1, \dots, x_k) = 1 - (1-\bar{x})^k$. Fix any $a > 0$. Then there is $0 < \delta < a$, $\epsilon > 0$ and $\hat{f} : [0,1]^k \to \mathcal{R}_+$ such that*

1. $\frac{\partial \hat{f}}{\partial x_i} \geq 0$, $\frac{\partial^2 \hat{f}}{\partial x_i \partial x_j} \leq 0$.
2. *For any $x \in [0,1]^k$, $f(x) + ag(x) - \delta \leq \hat{f}(x) \leq f(x) + ag(x)$.*
3. *Whenever $\forall i, j; |x_i - x_j| \leq \epsilon$, then $\hat{f}(x) = (1+a)g(x)$.*
4. *Whenever $\exists i, j; |x_i - x_j| \geq c/2$, then $\hat{f}(x) = f(x) + ag(x) - \delta$.*

The instance. Now we define an instance of submodular maximization derived from $\hat{f}$. Fix $c \in [0,1]$ and a large ℓ, and assume for convenience that $c\ell$ is an integer. Let $|X_1| = \ldots = |X_k| = \ell$. With a slight abuse of notation, we define $\hat{f}(S) = \hat{f}(|S \cap X_1|/|X_1|, \dots, |S \cap X_k|/|X_k|)$. This is a monotone submodular function, due to the properties of $\hat{f}$. We consider the problem[5]

$$\max\{\hat{f}(S) : |S| \leq c\ell\}.$$

An optimal solution is for example $S^* \subseteq X_1$, $|S^*| = c\ell$ which gives $\hat{f}(S^*) = c + ag(c, 0, \dots, 0) - \delta$.

Curvature. A new property that we need to prove is that $\hat{f}$ has curvature c with respect to S^*. The analysis would be quite clean if we had to deal with f instead of $\hat{f}$, but unfortunately $\hat{f}$ contains (small) additional terms which makes the analysis more messy. We claim a slightly weaker statement, that the curvature with respect to S^* is

[5]Equivalently, we could consider an instance of the Submodular Welfare Problem with k/c players, where each player has utility function $\hat{f}$.

at most $c' = c + \gamma$ for some $\gamma > 0$ which can be made arbitrarily small as $\epsilon \to 0$. This is sufficient to prove Theorem 4.1, where $\beta > 0$ is fixed and hence $\gamma > 0$ can be chosen appropriately smaller than β.

For any T where $|T \cap X_i| = t_i \ell$, we identify T with the vector $t = (t_1, t_2, \dots, t_k)$, and we denote $\bar{t} = \frac{1}{k} \sum_{i=1}^{k} t_i$. We have

$$\hat{f}(T) \leq f(t) + ag(t) = 1 - (1 - t_1)(1 - t_2) \cdots (1 - t_k) + a(1 - (1 - \bar{t})^k).$$

Now we estimate $\hat{f}(S^* + T) = \hat{f}(S^* \cup T) + \sum_{j \in S^* \cap T} \hat{f}_{S^* \cup T \setminus \{j\}}(j)$. First, observe that S^* and T can overlap only on X_1, and by submodularity and the equivalence of elements of X_1, $\hat{f}(S^* + T)$ is minimized if S^* and T overlap as little as possible. I.e., they are disjoint if $c + t_1 \leq 1$, and otherwise $S^* \cup (T \cap X_1) = X_1$.

Let $x = (\min\{c + t_1, 1\}, t_2, \dots, t_k)$. We distinguish two cases. Recall that $h(x) = (1 - \bar{x})^k - \prod_{i=1}^{k} (1 - x_i)$. First, assume that $h(x) > \epsilon_2$. Then $\phi(h(x)) = \delta$ is a constant, and we are in the range where $\hat{f}(x) = f(x) + ag(x) - \delta$. If $c + t_1 \leq 1$, then S^* and T are disjoint and

$$\begin{aligned}
\hat{f}(S^* + T) &= f(c + t_1, t_2, \dots, t_k) + ag(c + t_1, t_2, \dots, t_k) - \delta \\
&\geq f(c + t_1, t_2, \dots, t_k) + ag(c, 0, \dots, 0) - \delta \\
&= 1 - (1 - c - t_1)(1 - t_2) \cdots (1 - t_k) + ag(c, 0, \dots, 0) - \delta.
\end{aligned}$$

Similarly, if $c + t_1 > 1$, we have $(c + t_1 - 1)\ell$ elements in $S^* \cap T$, which contribute marginal values corresponding to the partial derivatives at $x = (1, t_2, \dots, t_k)$. Again, this is in the range where $\hat{f}(x) = f(x) + ag(x) - \delta$. Since the partial derivatives of g are nonnegative,

$$\frac{\partial \hat{f}}{\partial x_1}\Big|_{(1, t_2, \dots, t_k)} \geq \frac{\partial f}{\partial x_1}\Big|_{(1, t_2, \dots, t_k)} = (1 - t_2) \cdots (1 - t_k).$$

This brings us to the same estimate as above,

$$\begin{aligned}
\hat{f}(S^* + T) &\geq f(1, t_2, \dots, t_k) + (c + t_1 - 1) \frac{\partial f}{\partial x_1}\Big|_{(1, t_2, \dots, t_k)} + ag(c, 0, \dots, 0) - \delta \\
&\geq 1 - (1 - c - t_1)(1 - t_2) \cdots (1 - t_k) + ag(c, 0, \dots, 0) - \delta.
\end{aligned}$$

We continue:

$$\begin{aligned}
\hat{f}(S^* + T) &\geq 1 - (1 - c)(1 - t_1) \cdots (1 - t_k) + ag(c, 0, \dots, 0) - \delta \\
&= (1 - c)(1 - (1 - t_1) \cdots (1 - t_k)) + c + ag(c, 0, \dots, 0) - \delta \\
&= (1 - c) f(t_1, t_2, \dots, t_k) + \hat{f}(c, 0, \dots, 0) = (1 - c) f(T) + \hat{f}(S^*).
\end{aligned}$$

Since we always have $\hat{f}(T) \leq f(T) + ag(T) \leq (1 + a) f(T)$, we can conclude that

$$\hat{f}(S^* + T) \geq \frac{1 - c}{1 + a} \hat{f}(T) + \hat{f}(S^*).$$

As a second case, we assume that $h(x) \leq \epsilon_2$. (Recall that $x = (\min\{c+t_1, 1\}, t_2, \ldots, t_k)$.) This means that $\hat{f}(T)$ cannot be too small: for example, if $\hat{f}(T) < c/2$, then $\forall i; t_i \leq c/2$ and $\min\{c + t_1, 1\}$ is larger than all other coordinates by at least $c/2$. By Lemma 4.2, we would be in the first case. Therefore, we can assume $\hat{f}(T) \geq c/2$.

If $c + t_1 \leq 1$, we have

$$\begin{aligned}
\hat{f}(S^* + T) &= \hat{f}(c + t_1, \ldots, t_k) \geq f(c + t_1, \ldots, t_k) - \delta \\
&= 1 - (1 - c - t_1)(1 - t_2) \cdots (1 - t_k) - \delta \\
&\geq c + (1 - c)f(t_1, \ldots, t_2) - \delta \\
&\geq (f(S^*) - ac) + (1 - c)(\hat{f}(T) - a) - \delta \\
&\geq f(S^*) + (1 - c - 2(a + \delta)/c)\hat{f}(T)
\end{aligned}$$

using $\hat{f}(T) \geq c/2$.

If $c + t_1 > 1$, we have $x = (1, t_2, \ldots, t_k)$, $\bar{x} = \frac{1}{k}(1 + t_2 + \ldots + t_k)$, and $h(x) = f(x) - g(x) = (1 - \bar{x})^k \leq \epsilon_2$. This means $\bar{x} \geq 1 - \epsilon_2^{1/k}$ and therefore there is $i \geq 2$ such that $t_i \geq 1 - \frac{k}{k-1}\epsilon_2^{1/k} \geq 1 - 2\epsilon_2^{1/k}$. In this case, we conclude easily that

$$\hat{f}(S^* + T) \geq \hat{f}(T) \geq 1 - \prod_{i=1}^{k}(1 - t_i) - \delta \geq 1 - 2\epsilon_2^{1/k} - \delta.$$

Observe that k is fixed and ϵ_2 can be made arbitrarily small, so we can push this arbitrarily close to 1. On the other hand, $\hat{f}(S^*) \leq (1 + a)c$, and so we get

$$\hat{f}(S^*) \leq \frac{(1 + a)c}{1 - 2\epsilon_2^{1/k} - \delta}\hat{f}(T) \leq \frac{c}{1 - a - 2\epsilon_2^{1/k} - \delta}\hat{f}(T),$$

$$\hat{f}(S^* + T) - \hat{f}(S^*) \geq \hat{f}(T) - \hat{f}(S^*) \geq \left(1 - \frac{c}{1 - a - 2\epsilon_2^{1/k} - \delta}\right)\hat{f}(T).$$

As $\epsilon \to 0$, we also have $a \to 0$, $\epsilon_2 \to 0$ and $\delta \to 0$. Hence, in all the cases, curvature with respect to S^* is bounded by $c + \gamma$ where $\gamma \to 0$.

Analysis. The rest of the analysis is the same as in [18]. Suppose that the labeling of the elements (and hence the partition $(X_1, X_2, \ldots, X_k)$) is uniformly random on the input. Any query Q that an algorithm issues is going to partitioned roughly equally among $X_1, \ldots, X_k$, with high probability. More precisely, for any fixed $\delta > 0$, the corresponding coordinates $q_i = |Q \cap X_i|/|X_i|$ will satisfy $|q_i - q_j| < \delta$ with high probability. (By Chernoff bounds, the probability that $|q_i - q_j| \geq \delta$ for some i, j is exponentially small in n, the total number of elements.) Let's call a query *balanced* if $|q_i - q_j| < \delta$ for all i, j.

Assume for now that the algorithm is deterministic. If it's randomized, condition on its random bits. By the definition of $\hat{f}$, $\hat{f}(Q) = (1+a)(1-(1-\bar{q})^k)$ for any balanced query, and this value does not depend on the input partition in any way. Let $\mathcal{P}$ denote the path of computation that the algorithm would follow assuming that the answer to every query Q is $\hat{f}(Q) = (1+a)(1-(1-\bar{q})^k)$. For each value query, this happens with probability $1-e^{-\Omega(n)}$. However, assuming that the computation path $\mathcal{P}$ is polynomially long in n, with high probability all the value queries on $\mathcal{P}$ are balanced, the answer to each of these queries is $\hat{f}(Q) = (1+a)(1-(1-\bar{q})^k)$ and then the algorithm indeed follows path $\mathcal{P}$.

As a consequence, the algorithm will return the same answer with high probability, independent of $(X_1, \ldots, X_k)$. By unconditioning on the random bits of the algorithm, this holds even for randomized algorithms. The answer A is a set of size at most $c\ell$, hence the corresponding vector $a_i = |A \cap X_i|/|X_i|$ satisfies $\sum a_i \leq c$. With high probability, all the coordinates a_i are the same within δ, and hence

$$\hat{f}(a) \leq \hat{f}(c/k+\delta, \ldots, c/k+\delta) = (1+a)(1-(1-c/k-\delta)^k)$$

which can be made arbitrarily close to $1-e^{-c}$, as $a \to 0$, $\delta \to 0$, and $k \to \infty$. The optimum, on the other hand, is $\hat{f}(c, 0, \ldots, 0) \geq c - \delta$, which tends to c. Thus the approximation factor can be made arbitrarily close to $\frac{1}{c}(1-e^{-c})$. We proved that any algorithm will achieve this approximation factor at best, with high probability with respect to the random input. Hence, the same holds with respect to the worst case input as well. $\square$

References

[1] A. Ageev and M. Sviridenko. Pipage rounding: a new method of constructing algorithms with proven performance guarantee. *J. of Combinatorial Optimization*, 8:307–328, 2004.

[2] C. Chekuri and A. Kumar. Maximum coverage problem with group budget constraints and applications. *Proc. of APPROX*, Springer LNCS, 72–83, 2004.

[3] G. Calinescu, C. Chekuri, M. Pál and J. Vondrák. Maximizing a submodular set function subject to a matroid constraint. *Proc. of* 12th *IPCO*, 182–196, 2007.

[4] M. Conforti and G. Cornuéjols. Submodular set functions, matroids and the greedy algorithm: tight worst-case bounds and some generalizations of the Rado-Edmonds theorem. *Discrete Applied Math*, 7(3):251–274, 1984.

[5] G. Cornuéjols, M. Fisher and G. Nemhauser. Location of bank accounts to optimize float: an analytic study of exact and approximate algorithms. *Management Science*, 23: 789–810, 1977.

[6] S. Dobzinski and M. Schapira. An improved approximation algorithm for combinatorial auctions with submodular bidders. *Proc. of ACM-SIAM SODA*, 1064–1073, 2006.

[7] J. Edmonds. Matroids, submodular functions and certain polyhedra. Combinatorial Structures and Their Applications, 69–87, 1970.

[8] U. Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.

[9] U. Feige and J. Vondrák. Approximation algorithms for allocation problems: Improving the Factor of $1 - 1/e$. *Proc. of IEEE FOCS*, 667–676, 2006.

[10] U. Feige, V. Mirrokni and J. Vondrák. Maximizing a non-monotone submodular function. *Proc. of IEEE FOCS*, 461–471, 2007.

[11] M. L. Fisher, G. L. Nemhauser and L. A. Wolsey. An analysis of approximations for maximizing submodular set functions - II. *Math. Prog. Study*, 8:73–87, 1978.

[12] L. Fleischer, S. Fujishige and S. Iwata. A combinatorial, strongly polynomial-time algorithm for minimizing submodular functions. *Journal of the ACM* 48:4, 761–777, 2001.

[13] P. Goundan and A. Schulz. Revisiting the Greedy Approach to Submodular Set Function Maximization. Manuscript, July 2007.

[14] T. A. Jenkyns. The efficiency of the "greedy" algorithm. *Proc. of 7th South Eastern Conference on Combinatorics, Graph Theory and Computing*, 341–350, 1976.

[15] S. Khot, R. Lipton, E. Markakis and A. Mehta. Inapproximability results for combinatorial auctions with submodular utility functions. *Proc. of WINE 2005.*

[16] B. Korte and D. Hausmann. An analysis of the greedy heuristic for independence systems. *Annals of Discrete Math.*, 2:65–74, 1978.

[17] B. Lehmann, D. J. Lehmann and N. Nisan. Combinatorial auctions with decreasing marginal utilities. *Games and Economic Behavior* 55:270–296, 2006.

[18] V. Mirrokni, M. Schapira and J. Vondrák. Tight information-theoretic lower bounds for welfare maximization in combinatorial auctions. In EC 2008.

[19] G. L. Nemhauser, L. A. Wolsey and M. L. Fisher. An analysis of approximations for maximizing submodular set functions - I. *Math. Prog.*, 14:265–294, 1978.

[20] G. L. Nemhauser and L. A. Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Math. Oper. Research*, 3(3):177–188, 1978.

[21] A. Schrijver. Combinatorial optimization - polyhedra and efficiency. Springer, 2003.

[22] A. Schrijver. A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *Journal of Combinatorial Theory, Series B* 80, 346–355, 2000.

[23] J. Vondrák. Optimal approximation for the Submodular Welfare Problem in the value oracle model. *Proc. of ACM STOC*, 67–74, 2008.

[24] L. Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2:385–393, 1982.

[25] L. Wolsey. Maximizing real-valued submodular functions: Primal and dual heuristics for location Problems. *Math. of Operations Research*, 7:410–425, 1982.