Guaranteed Non-convex Optimization: Submodular Maximization over Continuous Domains

Andrew An Bian ETH Zurich
Baharan Mirzasoleiman ETH Zurich
Joachim M. Buhmann ETH Zurich
Andreas Krause ETH Zurich

Abstract

Submodular continuous functions are a category of (generally) non-convex/non-concave functions with a wide spectrum of applications. We characterize these functions and demonstrate that they can be maximized efficiently with approximation guarantees. Specifically, i) We introduce the `weak DR` property that gives a unified characterization of submodularity for all set, integer-lattice and continuous functions; ii) for maximizing monotone DR-submodular continuous functions under general down-closed convex constraints, we propose a Frank-Wolfe variant with $(1-1/e)$ approximation guarantee, and sub-linear convergence rate; iii) for maximizing general *non-monotone* submodular continuous functions subject to box constraints, we propose a DoubleGreedy algorithm with 1/3 approximation guarantee. Submodular continuous functions naturally find applications in various real-world settings, including influence and revenue maximization with continuous assignments, sensor energy management, facility location, etc. Experimental results show that the proposed algorithms efficiently generate superior solutions compared to baseline algorithms.

1 Introduction

Non-convex optimization delineates the new frontier in machine learning, arising in numerous learning tasks from training deep neural networks to latent variable models [4]. Understanding, which classes of objectives can be tractably optimized remains a central challenge.

In this paper, we investigate a class of generally non-convex and non-concave functions–*submodular continuous functions*, and derive algorithms for approximately optimizing them with strong approximation guarantees.

Submodularity is a structural property usually associated with *set functions*, with important implications for optimization. Optimizing submodular set functions has found numerous applications in machine learning, including variable selection [34], dictionary learning [32, 12], sparsity inducing regularizers [6], summarization [39, 41] and variational inference [13]. Submodular set functions can be efficiently minimized [27], and there are strong guarantees for approximate maximization [42, 33].

Even though submodularity is most widely considered in the discrete realm, the notion can be generalized to arbitrary lattices [20]. Recently, [5] showed how results from *submodular set function minimization* can be lifted to the continuous domain. In this paper, we further pursue this line of investigation, and demonstrate that results from *submodular set function maximization* can be generalized as well. Note that the underlying concepts associated with submodular function minimization and maximization are quite distinct, and both require different algorithmic treatment and analysis techniques.

As motivation for our inquiry, we firstly give a thorough characterization of the class of submodular and DR-submodular[1] functions. We propose the `weak DR` property and prove that it is equivalent to submodularity for general functions. This resolves the question whether there exists a diminishing-return-style characterization that is equivalent to submodularity for all set, integer-lattice and continuous functions. We then present two guaranteed algorithms for maximizing submodular continuous functions. The first approach, based on the Frank-Wolfe algorithm [19] and the continuous greedy algorithm [54], applies to monotone

Proceedings of the 20^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2017, Fort Lauderdale, Florida, USA. JMLR: W&CP volume 54. Copyright 2017 by the author(s).

[1] A DR-submodular function is a function with the diminishing returns property, which will be formally defined in Section 2.

DR-submodular functions. It provides a $(1-1/e)$ approximation guarantee under *general* down-closed convex constraints. We also provide a second, coordinate-ascent-style algorithm, which applies to arbitrary submodular continuous function maximization under box constraints, and provides a 1/3 approximation guarantee. This algorithm is based on the double greedy algorithm [9] from submodular set function maximization. Due to space limit, we defer further details on background and related work to Appendix A.[2]

Notation. We assume $E = \{e_1, e_2, \cdots, e_n\}$ is the ground set of n elements, and $\boldsymbol{\chi}_i \in \mathbb{R}^n$ is the characteristic vector for element e_i. We use boldface letters $\boldsymbol{x} \in \mathbb{R}^E$ and $\boldsymbol{x} \in \mathbb{R}^n$ interchanglebly to indicate a n-dimensional vector, where x_i is the i-th entry of $\boldsymbol{x}$. We use a boldface captial letter $\mathbf{A} \in \mathbb{R}^{m\times n}$ to denote a matrix. For two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^E$, $\boldsymbol{x} \leq \boldsymbol{y}$ means $x_i \leq y_i$ for every element i in E. Finally, $\boldsymbol{x}|x_i \leftarrow k$ is the operation of setting the i-th element of $\boldsymbol{x}$ to k, while keeping all other elements unchanged.

2 Characterizations of submodular continuous functions

Submodular continuous functions are defined on subsets of $\mathbb{R}^n$: $\mathcal{X} = \prod_{i=1}^n \mathcal{X}_i$, where each $\mathcal{X}_i$ is a compact subset of $\mathbb{R}$ [53, 5]. A function $f : \mathcal{X} \to \mathbb{R}$ is submodular *iff* for all $(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{X} \times \mathcal{X}$,

$$f(\boldsymbol{x})+f(\boldsymbol{y}) \geq f(\boldsymbol{x}\vee\boldsymbol{y})+f(\boldsymbol{x}\wedge\boldsymbol{y}), \quad (\texttt{submodular}) \quad (1)$$

where $\wedge$ and $\vee$ are the coordinate-wise minimum and maximum operations, respectively. Specifically, $\mathcal{X}_i$ could be a finite set, such as $\{0, 1\}$ (in which case $f(\cdot)$ is called *set* function), or $\{0, \cdots, k_i - 1\}$ (called *integer-lattice* function), where the notion of continuity is vacuous; $\mathcal{X}_i$ can also be an interval, which is referred to as a continuous domain. In this work, we consider the interval by default, but it is worth noting that the properties introduced in this section can be applied to $\mathcal{X}_i$ being a general compact subset of $\mathbb{R}$.

When twice-differentiable, $f(\cdot)$ is submodular iff all off-diagonal entries of its Hessian are non-positive[3] [5],

$$\forall \boldsymbol{x} \in \mathcal{X}, \quad \frac{\partial^2 f(\boldsymbol{x})}{\partial x_i \partial x_j} \leq 0, \quad \forall i \neq j. \quad (2)$$

The class of submodular continuous functions contains a subset of both convex and concave functions, and shares some useful properties with them (illustrated in Figure 1). Examples include submodular and convex functions of the form $\phi_{ij}(x_i - x_j)$ for ϕ_{ij} convex; submodular and concave functions of the form $\boldsymbol{x} \mapsto g(\sum_{i=1}^n \lambda_i x_i)$ for g concave and λ_i non-negative (see Section 5 for example applications).

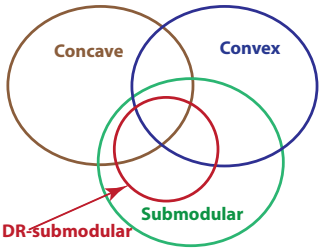


Figure 1: Concavity, convexity, submodularity and DR-submodularity.

Lastly, indefinite quadratic functions of the form $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \mathbf{H}\boldsymbol{x} + \boldsymbol{h}^\top \boldsymbol{x}$ with all off-diagonal entries of $\mathbf{H}$ non-positive are examples of submodular but non-convex/non-concave functions. Continuous submodularity is preserved under various operations, e.g., the sum of two submodular continuous functions is submodular, a submodular continuous function multiplied by a positive scalar is still submodular. Interestingly, characterizations of submodular continuous functions are in correspondence to those of convex functions, which are summarized in Table 1.

In the remainder of this section, we introduce useful properties of submodular continuous functions. First of all, we generalize the DR property (which was introduced when studying set and integer-lattice functions) to general functions defined over $\mathcal{X}$. It will soon be clear that the DR property defines a subclass of submodular functions.

Definition 1 (DR property)**.** *A function $f(\cdot)$ defined over $\mathcal{X}$ satisfies the diminishing returns (DR) property if $\forall \boldsymbol{a} \leq \boldsymbol{b} \in \mathcal{X}$, $\forall i \in E$, $\forall k \in \mathbb{R}_+$ s.t. $(k\boldsymbol{\chi}_i + \boldsymbol{a})$ and $(k\boldsymbol{\chi}_i + \boldsymbol{b})$ are still in $\mathcal{X}$, it holds,*

$$f(k\boldsymbol{\chi}_i + \boldsymbol{a}) - f(\boldsymbol{a}) \geq f(k\boldsymbol{\chi}_i + \boldsymbol{b}) - f(\boldsymbol{b}).$$

$f(\cdot)$ is called a DR-submodular[4] function.

One immediate observation is that for a differentiable DR-submodular function $f(\cdot)$, we have that $\forall \boldsymbol{a} \leq \boldsymbol{b} \in \mathcal{X}$, $\nabla f(\boldsymbol{a}) \geq \nabla f(\boldsymbol{b})$, i.e., the gradient $\nabla f(\cdot)$ is an *antitone* mapping from $\mathbb{R}^n$ to $\mathbb{R}^n$. Recently, the DR property is explored by [15] to achieve the worst-case competitive ratio for an online concave maximization problem. DR is also closely related to a sufficient condition on a concave function $g(\cdot)$ [8, Section 5.2], to ensure submodularity of the corresponding set function generated by giving $g(\cdot)$ boolean input vectors.

It is well known that for set functions, DR is equivalent to submodularity, while for integer-lattice functions, submodularity does not in general imply DR

[2] A full version of this work is in [7].

[3] Notice that an equilavent definition of (1) is that $\forall \boldsymbol{x} \in \mathcal{X}$, $\forall i \neq j$ and $a_i, a_j \geq 0$ s.t. $x_i + a_i \in \mathcal{X}_i, x_j + a_j \in \mathcal{X}_j$, it holds $f(\boldsymbol{x}+a_i\boldsymbol{\chi}_i)+f(\boldsymbol{x}+a_j\boldsymbol{\chi}_j) \geq f(\boldsymbol{x})+f(\boldsymbol{x}+a_i\boldsymbol{\chi}_i+a_j\boldsymbol{\chi}_j)$. With a_i and a_j approaching zero, one get (2).

[4] Note that DR property implies submodularity and thus the name "DR-submodular" contains redundant information about submodularity of a function, but we keep this terminology to be consistent with previous literature on submodular integer-lattice functions.

Properties	Submodular continuous function $f(\cdot)$	Convex function $g(\cdot)$, $\forall\lambda\in[0,1]$
0^{th} order	$f(\boldsymbol{x})+f(\boldsymbol{y})\geq f(\boldsymbol{x}\vee\boldsymbol{y})+f(\boldsymbol{x}\wedge\boldsymbol{y})$	$\lambda g(\boldsymbol{x})+(1-\lambda)g(\boldsymbol{y})\geq g(\lambda\boldsymbol{x}+(1-\lambda)\boldsymbol{y})$
1^{st} order	`weak DR` (this work, Definition 2)	$g(\boldsymbol{y})-g(\boldsymbol{x})\geq\langle\nabla g(\boldsymbol{x}),\boldsymbol{y}-\boldsymbol{x}\rangle$
2^{nd} order	$\frac{\partial^2 f(\boldsymbol{x})}{\partial x_i\partial x_j}\leq 0,\forall i\neq j$	$\nabla^2 g(\boldsymbol{x})\succeq 0$ (positive semidefinite)

Table 1: Comparison of properties of submodular and convex continuous functions

[48, 49, 50]. However, it was unclear whether there exists a diminishing-return-style characterization that is equivalent to submodularity of integer-lattice functions. We give a positive answer to this open problem by proposing the *weak diminishing returns* (`weak DR`) property for general functions defined over $\mathcal{X}$, and prove that `weak DR` gives a sufficient and necessary condition for a general function to be submodular.

Definition 2 (`weak DR` property)**.** *A function $f(\cdot)$ defined over $\mathcal{X}$ has the weak diminishing returns property (`weak DR`) if $\forall\boldsymbol{a}\leq\boldsymbol{b}\in\mathcal{X}$, $\forall i\in E$ s.t. $a_i=b_i,\forall k\in\mathbb{R}_+$ s.t. $(k\boldsymbol{\chi}_i+\boldsymbol{a})$ and $(k\boldsymbol{\chi}_i+\boldsymbol{b})$ are still in $\mathcal{X}$, it holds,*

$$f(k\boldsymbol{\chi}_i+\boldsymbol{a})-f(\boldsymbol{a})\geq f(k\boldsymbol{\chi}_i+\boldsymbol{b})-f(\boldsymbol{b}). \tag{3}$$

The following proposition shows that for all set functions, as well as integer-lattice and continuous functions, submodularity is equivalent to `weak DR`.

Proposition 1 (`submodularity`) $\Leftrightarrow$ (`weak DR`)**.** *A function $f(\cdot)$ defined over $\mathcal{X}$ is submodular iff it satisfies the weak DR property.*

All of the proofs can be found in Appendix B. Given Proposition 1, one can treat `weak DR` as the first order condition of submodularity: Notice that for a differentiable function $f(\cdot)$ with `weak DR`, we have $\forall\boldsymbol{a}\leq\boldsymbol{b}\in\mathcal{X}$, $\forall i\in E$ s.t. $a_i=b_i$, it holds $\nabla_i f(\boldsymbol{a})\geq\nabla_i f(\boldsymbol{b})$, i.e., $\nabla f(\cdot)$ is a weak antitone mapping. Now we show that `DR` is stronger than `weak DR` , and the class of DR-submodular functions is a proper subset of that of submodular functions, as indicated by Figure 1.

Proposition 2 (`submodular/weak DR`) + (`coordinate-wise concave`) $\Leftrightarrow$ (`DR`)**.** *A function $f(\cdot)$ defined over $\mathcal{X}$ satisfies `DR` iff $f(\cdot)$ is submodular and coordinate-wise concave, where the `coordinate-wise concave` property is defined as: $\forall\boldsymbol{x}\in\mathcal{X}$, $\forall i\in E$, $\forall k,l\in\mathbb{R}_+$ s.t. $(k\boldsymbol{\chi}_i+\boldsymbol{x}),(l\boldsymbol{\chi}_i+\boldsymbol{x}),((k+l)\boldsymbol{\chi}_i+\boldsymbol{x})$ are still in $\mathcal{X}$, it holds,*

$$f(k\boldsymbol{\chi}_i+\boldsymbol{x})-f(\boldsymbol{x})\geq f((k+l)\boldsymbol{\chi}_i+\boldsymbol{x})-f(l\boldsymbol{\chi}_i+\boldsymbol{x}),$$

equivalently (if twice differentiable) $\frac{\partial^2 f(\boldsymbol{x})}{\partial x_i^2}\leq 0,\forall i\in E$.

Proposition 2 shows that a twice differentiable function $f(\cdot)$ is DR-submodular iff $\forall\boldsymbol{x}\in\mathcal{X}$, $\frac{\partial^2 f(\boldsymbol{x})}{\partial x_i\partial x_j}\leq 0,\forall i,j\in E$, which does not necessarily imply the concavity of $f(\cdot)$. Given Proposition 2, we also have the characterizations of DR-submodular continuous functions, which are summarized in Table 2.

3 Maximizing monotone DR-submodular continuous functions

In this section, we present an algorithm for maximizing a monotone DR-submodular continuous function subject to a general down-closed convex constraint, i.e., $\max_{\boldsymbol{x}\in\mathcal{P}_{\underline{\boldsymbol{u}}}} f(\boldsymbol{x})$. A down-closed convex set $\mathcal{P}_{\underline{\boldsymbol{u}}}$ is a convex set $\mathcal{P}$ associated with a lower bound $\underline{\boldsymbol{u}}\in\mathcal{P}$, such that 1) $\forall\boldsymbol{y}\in\mathcal{P}$, $\underline{\boldsymbol{u}}\leq\boldsymbol{y}$; and 2) $\forall\boldsymbol{y}\in\mathcal{P}$, $\boldsymbol{x}\in\mathbb{R}^n$, $\underline{\boldsymbol{u}}\leq\boldsymbol{x}\leq\boldsymbol{y}$ implies $\boldsymbol{x}\in\mathcal{P}$. Without loss of generality, we assume $\mathcal{P}$ lies in the postitive orthant and has the lower bound 0, since otherwise we can always define a new set $\mathcal{P}'=\{\boldsymbol{x}\mid\boldsymbol{x}=\boldsymbol{y}-\underline{\boldsymbol{u}},\boldsymbol{y}\in\mathcal{P}\}$ in the positive orthant, and a corresponding monotone DR-submdular function $f'(\boldsymbol{x}):=f(\boldsymbol{x}+\underline{\boldsymbol{u}})$.

Maximizing a monotone DR-submodular function over a down-closed convex constraint has many real-world applications, e.g., influence maximization with continuous assignments and sensor energy management. In particular, for influence maximization (see Section 5), the constraint is a down-closed polytope in the positive orthant: $\mathcal{P}=\{\boldsymbol{x}\in\mathbb{R}^n\mid 0\leq\boldsymbol{x}\leq\bar{\boldsymbol{u}},\mathbf{A}\boldsymbol{x}\leq\boldsymbol{b},\bar{\boldsymbol{u}}\in\mathbb{R}^n_+,\mathbf{A}\in\mathbb{R}^{m\times n}_+,\boldsymbol{b}\in\mathbb{R}^m_+\}$. We start with the following hardness result:

Proposition 3. *The problem of maximizing a monotone DR-submodular continuous function subject to a general down-closed* polytope *constraint is NP-hard. For any $\epsilon>0$, it cannot be approximated in polynomial time within a ratio of $(1-1/e+\epsilon)$ (up to low-order terms), unless RP = NP.*

Due to the NP-hardness of converging to the global optimum, in the following by "convergence" we mean

Properties	DR-submodular $f(\cdot)$, $\forall\boldsymbol{x},\boldsymbol{y}\in\mathcal{X}$
0^{th} order	$f(\boldsymbol{x})+f(\boldsymbol{y})\geq f(\boldsymbol{x}\vee\boldsymbol{y})+f(\boldsymbol{x}\wedge\boldsymbol{y})$, and $f(\cdot)$ is coordinate-wise concave
1^{st} order	the `DR` property (Definition 1)
2^{nd} order	$\frac{\partial^2 f(\boldsymbol{x})}{\partial x_i\partial x_j}\leq 0,\forall i,j\in E$

Table 2: Properties of DR-submodular functions

Algorithm 1: FRANK-WOLFE variant for monotone DR-submodular function maximization

Input: $\max_{\boldsymbol{x}\in\mathcal{P}} f(\boldsymbol{x})$, $\mathcal{P}$ is a down-closed convex set in the positive orthant with lower bound 0, prespecified stepsize $\gamma \in (0, 1]$

1 $\boldsymbol{x}^0 \leftarrow 0$, $t \leftarrow 0$, $k \leftarrow 0$; *//k : iteration index*

2 **while** $t < 1$ **do**

3 find $\boldsymbol{v}^k$ s.t. $\langle \boldsymbol{v}^k, \nabla f(\boldsymbol{x}^k)\rangle \geq \alpha \max_{\boldsymbol{v}\in\mathcal{P}} \langle \boldsymbol{v}, \nabla f(\boldsymbol{x}^k)\rangle - \frac{1}{2}\delta L$; *//$L > 0$ is the Lipschitz parameter, $\alpha \in (0, 1]$ is the mulplicative error level, $\delta \in [0, \bar{\delta}]$ is the additive error level*

4 find stepsize $\gamma_k \in (0, 1]$, e.g., $\gamma_k \leftarrow \gamma$; and set $\gamma_k \leftarrow \min\{\gamma_k, 1 - t\}$;

5 $\boldsymbol{x}^{k+1} \leftarrow \boldsymbol{x}^k + \gamma_k \boldsymbol{v}^k$, $t \leftarrow t + \gamma_k$, $k \leftarrow k + 1$;

6 Return $\boldsymbol{x}^K$; *//assuming there are K iterations in total*

converging to a point near the global optimum. The algorithm is a generalization of the continuous greedy algorithm of [54] for maximizing a smooth submodular function, and related to the convex Frank-Wolfe algorithm [19, 28] for minimizing a convex function. We summarize the FRANK-WOLFE variant in Algorithm 1. In each iteration k, the algorithm uses the linearization of $f(\cdot)$ as a surrogate, and moves in the direction of the maximizer of this surrogate function, i.e. $\boldsymbol{v}^k = \arg\max_{\boldsymbol{v}\in\mathcal{P}} \langle \boldsymbol{v}, \nabla f(\boldsymbol{x}^k)\rangle$. Intuitively, we search for the direction in which we can maximize the improvement in the function value and still remain feasible. Finding such a direction requires maximizing a linear objective at each iteration. Meanwhile, it eliminates the need for projecting back to the feasible set in each iteration, which is an essential step for methods such as projected gradient ascent. The algorithm uses stepsize γ_k to update the solution in each iteration, which can be simply a prespecified value γ. Note that the FRANK-WOLFE variant can tolerate both multiplicative error α and additive error δ when solving the subproblem (Step 3 of Algorithm 1). Setting $\alpha = 1$ and $\delta = 0$, we recover the error-free case.

Notice that the FRANK-WOLFE variant in Algorithm 1 is different from the convex Frank-Wolfe algorithm mainly in the update direction being used: For Algorithm 1, the update direction (in Step 5) is $\boldsymbol{v}^k$, while for convex Frank-Wolfe it is $\boldsymbol{v}^k - \boldsymbol{x}^k$, i.e., $\boldsymbol{x}^{k+1} \leftarrow \boldsymbol{x}^k + \gamma_k(\boldsymbol{v}^k - \boldsymbol{x}^k)$. The reason of this difference will soon be clear by exploring the property of DR-submodular functions. Specifically, DR-submodular functions are non-convex/non-concave in general, however, there is certain connection between DR-submodularity and concavity.

Proposition 4. *A DR-submodular continuous function $f(\cdot)$ is concave along any non-negative direction, and any non-positive direction.*

Proposition 4 implies that the univariate *auxiliary* function $g_{\boldsymbol{x},\boldsymbol{v}}(\xi) := f(\boldsymbol{x} + \xi\boldsymbol{v}), \xi \in \mathbb{R}_+, \boldsymbol{v} \in \mathbb{R}_+^E$ is concave. As a result, the FRANK-WOLFE variant can follow a concave direction at each step, which is the main reason it uses $\boldsymbol{v}^k$ as the update direction (notice that $\boldsymbol{v}^k$ is a non-negative direction).

To derive the approximation guarantee, we need assumptions on the non-linearity of $f(\cdot)$ over the domain $\mathcal{P}$, which closely corresponds to a Lipschitz assumption on the derivative of $g_{\boldsymbol{x},\boldsymbol{v}}(\cdot)$. For a $g_{\boldsymbol{x},\boldsymbol{v}}(\cdot)$ with L-Lipschitz continuous derivative in $[0, 1]$ ($L > 0$), we have,

$$-\frac{L}{2}\xi^2 \leq g_{\boldsymbol{x},\boldsymbol{v}}(\xi) - g_{\boldsymbol{x},\boldsymbol{v}}(0) - \xi\nabla g_{\boldsymbol{x},\boldsymbol{v}}(0) \quad (4)$$
$$= f(\boldsymbol{x} + \xi\boldsymbol{v}) - f(\boldsymbol{x}) - \langle \xi\boldsymbol{v}, \nabla f(\boldsymbol{x})\rangle, \forall \xi \in [0, 1].$$

To prove the approximation guarantee, we first derive the following lemma.

Lemma 1. *$\boldsymbol{x}^K \in \mathcal{P}$. Assuming $\boldsymbol{x}^*$ to be the optimal solution, one has, $\forall k = 0, \cdots, K - 1$,*

$$\langle \boldsymbol{v}^k, \nabla f(\boldsymbol{x}^k)\rangle \geq \alpha[f(\boldsymbol{x}^*) - f(\boldsymbol{x}^k)] - \frac{1}{2}\delta L. \quad (5)$$

Theorem 1 (Approximation guarantee)**.** *For error levels $\alpha \in (0, 1], \delta \in [0, \bar{\delta}]$, with K iterations, Algorithm 1 outputs $\boldsymbol{x}^K \in \mathcal{P}$ s.t.,*

$$f(\boldsymbol{x}^K) \geq (1-e^{-\alpha})f(\boldsymbol{x}^*) - \frac{L}{2}\sum_{k=0}^{K-1}\gamma_k^2 - \frac{L\delta}{2} + e^{-\alpha}f(0).$$

Theorem 1 gives the approximation guarantee for arbitrary chosen stepsize γ_k. By observing that $\sum_{k=0}^{K-1}\gamma_k = 1$ and $\sum_{k=0}^{K-1}\gamma_k^2 \geq K^{-1}$ (see the proof in Appendix C.5), with constant stepsize, we obtain the following "tightest" approximation bound,

Corollary 1. *For a fixed number of iterations K, and constant stepsize $\gamma_k = \gamma = K^{-1}$, Algorithm1 provides the following approximation guarantee:*

$$f(\boldsymbol{x}^K) \geq (1 - e^{-\alpha})f(\boldsymbol{x}^*) - \frac{L}{2K} - \frac{L\delta}{2} + e^{-\alpha}f(0).$$

Corollary 1 implies that with a constant stepsize γ, 1) when $\gamma \to 0$ ($K \to \infty$), Algorithm1 will output the solution with the worst-case guarantee $(1 - 1/e)f(\boldsymbol{x}^*)$ in the error-free case if $f(0) = 0$; and 2) The FRANK-WOLFE variant has a sub-linear convergence rate for monotone DR-submodular maximization over a down-closed convex constraint.

Time complexity. It can be seen that when using a constant stepsize, Algorithm 1 needs $O(\frac{1}{\epsilon})$ iterations to get ϵ-close to the worst-case guarantee $(1 - e^{-1})f(\boldsymbol{x}^*)$ in the error-free case. When $\mathcal{P}$ is a polytope in the positive orthant, one iteration of Algorithm 1 costs approximately the same as solving a positive LP, for which a nearly-linear time solver exists [3].

Algorithm 2: DOUBLEGREEDY algorithm for maximizing non-monotone submodular continuous functions

Input: $\max_{\boldsymbol{x}\in[\underline{\boldsymbol{u}},\bar{\boldsymbol{u}}]} f(\boldsymbol{x})$, f is generally non-monotone, $f(\underline{\boldsymbol{u}}) + f(\bar{\boldsymbol{u}}) \geq 0$

1 $\boldsymbol{x}^0 \leftarrow \underline{\boldsymbol{u}}$, $\boldsymbol{y}^0 \leftarrow \bar{\boldsymbol{u}}$;

2 **for** $k = 1 \rightarrow n$ **do**

3 find $\hat{u}_a$ s.t. $f(\boldsymbol{x}^{k-1}|x^{k-1}_{e_k} \leftarrow \hat{u}_a) \geq \max_{u_a \in [\underline{u}_{e_k}, \bar{u}_{e_k}]} f(\boldsymbol{x}^{k-1}|x^{k-1}_{e_k} \leftarrow u_a) - \delta$, $\delta_a \leftarrow f(\boldsymbol{x}^{k-1}|x^{k-1}_{e_k} \leftarrow \hat{u}_a) - f(\boldsymbol{x}^{k-1})$; *//$\delta \in [0, \bar{\delta}]$ is the additive error level*

4 find $\hat{u}_b$ s.t. $f(\boldsymbol{y}^{k-1}|y^{k-1}_{e_k} \leftarrow \hat{u}_b) \geq \max_{u_b \in [\underline{u}_{e_k}, \bar{u}_{e_k}]} f(\boldsymbol{y}^{k-1}|y^{k-1}_{e_k} \leftarrow u_b) - \delta$, $\delta_b \leftarrow f(\boldsymbol{y}^{k-1}|y^{k-1}_{e_k} \leftarrow \hat{u}_b) - f(\boldsymbol{y}^{k-1})$;

5 **If** $\delta_a \geq \delta_b$: $\boldsymbol{x}^k \leftarrow (\boldsymbol{x}^{k-1}|x^{k-1}_{e_k} \leftarrow \hat{u}_a)$, $\boldsymbol{y}^k \leftarrow (\boldsymbol{y}^{k-1}|y^{k-1}_{e_k} \leftarrow \hat{u}_a)$;

6 **Else**: $\boldsymbol{y}^k \leftarrow (\boldsymbol{y}^{k-1}|y^{k-1}_{e_k} \leftarrow \hat{u}_b)$, $\boldsymbol{x}^k \leftarrow (\boldsymbol{x}^{k-1}|x^{k-1}_{e_k} \leftarrow \hat{u}_b)$;

7 Return $\boldsymbol{x}^n$ (or $\boldsymbol{y}^n$); *//note that $\boldsymbol{x}^n = \boldsymbol{y}^n$*

4 Maximizing non-monotone submodular continuous functions

The problem of maximizing a general non-monotone submodular continuous function under box constraints[5], i.e., $\max_{\boldsymbol{x}\in[\underline{\boldsymbol{u}},\bar{\boldsymbol{u}}]\subseteq\mathcal{X}} f(\boldsymbol{x})$, has various real-world applications, including revenue maximization with continuous assignments, multi-resolution summarization, etc, as discussed in Section 5. The following proposition shows the NP-hardness of the problem.

Proposition 5. *The problem of maximizing a generally non-monotone submodular continuous function subject to box constraints is NP-hard. Furthermore, there is no $(1/2+\epsilon)$-approximation $\forall \epsilon > 0$, unless RP = NP.*

We now describe our algorithm for maximizing a non-monotone submodular continuous function subject to box constraints. It provides a 1/3-approximation, is inspired by the double greedy algorithm of [9] and [23], and can be viewed as a procedure performing coordinate-ascent on *two* solutions.

We view the process as two particles starting from $\boldsymbol{x}^0 = \underline{\boldsymbol{u}}$ and $\boldsymbol{y}^0 = \bar{\boldsymbol{u}}$, and following a certain "flow" toward each other. The pseudo-code is given in Algorithm 2. We proceed in n rounds that correspond to some arbitrary order of the coordinates. At iteration k, we consider solving a one-dimensional (1-D) subproblem over coordinate e_k for each particle, and moving the particles based on the calculated local gains toward each other. Formally, for a given coordinate e_k, we solve a 1-D subproblem to find the value of the first solution $\boldsymbol{x}$ along coordinate e_k that maximizes f, i.e., $\hat{u}_a = \arg\max_{u_a} f(\boldsymbol{x}^{k-1}|x^{k-1}_{e_k} \leftarrow u_a) - f(\boldsymbol{x}^{k-1})$, and calculate its marginal gain δ_a. We then solve another 1-D subproblem to find the value of the second solution $\boldsymbol{y}$ along coordinate e_k that maximizes f, i.e., $\hat{u}_b = \arg\max_{u_b} f(\boldsymbol{y}^{k-1}|\boldsymbol{y}^{k-1}_{e_k} \leftarrow u_b) - f(\boldsymbol{y}^{k-1})$, and calculate the second marginal gain δ_b. We decide by comparing the two marginal gains. If changing x_{e_k} to be $\hat{u}_a$ has a larger local benefit, we change *both* x_{e_k} and y_{e_k} to be $\hat{u}_a$. Otherwise, we change *both* of them to be $\hat{u}_b$. After n iterations the particles should meet at point $\boldsymbol{x}^n = \boldsymbol{y}^n$, which is the final solution. Note that Algorithm 2 can tolerate additive error δ in solving each 1-D subproblem (Steps 3, 4).

We would like to emphasize that the assumptions required by DOUBLEGREEDY are submodularity of f, $f(\underline{\boldsymbol{u}}) + f(\bar{\boldsymbol{u}}) \geq 0$ and the (approximate) solvability of the 1-D subproblem. For proving the approximation guarantee, the idea is to bound the loss in the objective value from the assumed optimal objective value between every two consecutive steps, which is then used to bound the maximum loss after n iterations.

Theorem 2. *Assuming the optimal solution to be $\boldsymbol{x}^*$, the output of Algorithm 2 has function value no less than $\frac{1}{3}f(\boldsymbol{x}^*) - \frac{4n}{3}\delta$, where $\delta \in [0, \bar{\delta}]$ is the additive error level for each 1-D subproblem.*

Time complexity. It can be seen that the time complexity of Algorithm 2 is $O(n * \texttt{cost\_1D})$, where `cost_1D` is the cost of solving the 1-D subproblem. Solving a 1-D subproblem is usually very cheap. For non-convex/non-concave quadratic programming it has a closed form solution.

5 Examples of submodular continuous objective functions

In this part, we discuss several concrete problem instances with their corresponding submodular continuous objective functions.

Extensions of submodular set functions. The multilinear extension [10] and softmax extension [21] are special cases of DR-submodular functions, that are extensively used for submodular set function maximization. The Lovász extension [40] used for submodular set function minimization is both submodular and convex (see Appendix A in [5]).

Non-convex/non-concave quadratic programming (NQP). NQP problem of the form $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \mathbf{H}\boldsymbol{x} + \boldsymbol{h}^\top \boldsymbol{x} + c$ under linear constraints naturally

[5] It is also called "unconstrained" maximization in the combinatorial optimization community, since the domain $\mathcal{X}$ itself is also a box. Note that the box can be in the negative orthant here.

arises in many applications, including scheduling [47], inventory theory, and free boundary problems. A special class of NQP is the submodular NQP (the minimization of which was studied in [30]), in which all off-diagonal entries of $\mathbf{H}$ are required to be non-positive. In this work, we mainly use submodular NQPs as synthetic functions for both monotone DR-submodular maximization and non-monotone submodular maximization.

Optimal budget allocation with continuous assignments. Optimal budget allocation is a special case of the influence maximization problem. It can be modeled as a bipartite graph $(S, T; W)$, where S and T are collections of advertising channels and customers, respectively. The edge weight, $p_{st} \in W$, represents the influence probability of channel s to customer t. The goal is to distribute the budget (e.g., time for a TV advertisement, or space of an inline ad) among the source nodes, and to maximize the expected influence on the potential customers [48, 25]. The total influence of customer t from all channels can be modeled by a proper monotone DR-submodular function $I_t(\boldsymbol{x})$, e.g., $I_t(\boldsymbol{x}) = 1 - \prod_{(s,t)\in W} (1 - p_{st})^{x_s}$ where $\boldsymbol{x} \in \mathbb{R}_+^S$ is the budget assignment among the advertising channels. For a set of k advertisers, let $\boldsymbol{x}^i \in \mathbb{R}_+^S$ to be the budget assignment for advertiser i, and $\boldsymbol{x} := [\boldsymbol{x}^1, \cdots, \boldsymbol{x}^k]$ denote the assignments for all the advertisers. The overall objective is,

$$g(\boldsymbol{x}) = \sum\nolimits_{i=1}^{k} \alpha_i f(\boldsymbol{x}^i) \text{ with } \quad f(\boldsymbol{x}^i) := \sum\nolimits_{t\in T} I_t(\boldsymbol{x}^i), \\ 0 \le \boldsymbol{x}^i \le \bar{\boldsymbol{u}}^i, \forall i = 1, \cdots, k$$

which is monotone DR-submodular. A concrete application is for search marketing advertiser bidding, in which vendors bid for the right to appear alongside the results of different search keywords. Here, x_s^i is the volume of advertising space allocated to the advertiser i to show his ad alongside query keyword s. The search engine company needs to distribute the budget (advertising space) to all vendors to maximize their influence on the customers, while respecting various constraints. For example, each vendor has a specified budget limit for advertising, and the ad space associated with each search keyword can not be too large. All such constraints can be formulated as a down-closed polytope $\mathcal{P}$, hence the Frank-Wolfe variant can be used to find an approximate solution for the problem $\max_{\boldsymbol{x}\in\mathcal{P}} g(\boldsymbol{x})$. Note that one can flexibly add regularizers in designing $I_t(\boldsymbol{x}^i)$ as long as it remains monotone DR-submodular. For example, adding separable regularizers of the form $\sum_s \phi(x_s^i)$ does not change the off-diagonal entries of the Hessian, and hence maintains submodularity. Alternatively, bounding the second-order derivative of $\phi(x_s^i)$ ensures DR-submodularity.

Revenue maximization with continuous assignments. In viral marketing, sellers choose a small subset of buyers to give them some product for free, to trigger a cascade of further adoptions through "word-of-mouth" effects, in order to maximize the total revenue [24]. For some products (e.g., software), the seller usually gives away the product in the form of a trial, to be used for free for a limited time period. In this task, except for deciding whether to choose a user or not, the sellers also need to decide how much the free assignment should be, in which the assignments should be modeled as continuous variables. We call this problem *revenue maximization with continuous assignments*. Assume there are q products and n buyers/users, let $\boldsymbol{x}^i \in \mathbb{R}_+^n$ to be the assignments of product i to the n users, let $\boldsymbol{x} := [\boldsymbol{x}^1, \cdots, \boldsymbol{x}^q]$ denote the assignments for the q products. The revenue can be modelled as $g(\boldsymbol{x}) = \sum_{i=1}^q f(\boldsymbol{x}^i)$ with

$$f(\boldsymbol{x}^i) := \alpha_i \sum\nolimits_{s:x_s^i=0} R_s(\boldsymbol{x}^i) + \beta_i \sum\nolimits_{t:x_t^i\neq 0} \phi(x_t^i) \\ + \gamma_i \sum\nolimits_{t:x_t^i\neq 0} \bar{R}_t(\boldsymbol{x}^i), \quad 0 \le \boldsymbol{x}^i \le \bar{\boldsymbol{u}}^i, \tag{6}$$

where x_t^i is the assignment of product i to user t for free, e.g., the amount of free trial time or the amount of the product itself. $R_s(\boldsymbol{x}^i)$ models revenue gain from user s who did not receive the free assignment. It can be some non-negative, non-decreasing submodular function. $\phi(x_t^i)$ models revenue gain from user t who received the free assignment, since the more one user tries the product, the more likely he/she will buy it after the trial period. $\bar{R}_t(\boldsymbol{x}^i)$ models the revenue loss from user t (in the free trial time period the seller cannot get profits), which can be some non-positive, non-increasing submodular function. With β=γ=0, we recover the classical model of [24]. For products with continuous assignments, usually the cost of the product does not increase with its amount, e.g., the product as a software, so we only have the box constraint on each assignment. The objective in Eq. 6 is generally *non-concave/non-convex*, and non-monotone submodular (see Appendix E for more details), thus can be approximately maximized by the proposed Double-Greedy algorithm.

Lemma 2. *If $R_s(\boldsymbol{x}^i)$ is non-decreasing submodular and $\bar{R}_t(\boldsymbol{x}^i)$ is non-increasing submodular, then $f(\boldsymbol{x}^i)$ in Eq. 6 is submodular.*

Sensor energy management. For cost-sensitive outbreak detection in sensor networks [36], one needs to place sensors in a subset of locations selected from all the possible locations E, to quickly detect a set of contamination events V, while respecting the cost constraints of the sensors. For each location $e \in E$ and each event $v \in V$, a value $t(e, v)$ is provided as the time it takes for the placed sensor in e to detect event v. [49] considered the sensors with discrete energy lev-

els. It is also natural to model the energy levels of sensors to be a *continuous* variable $\boldsymbol{x} \in \mathbb{R}_+^E$. For a sensor with energy level x_e, the success probability it detects the event is $1-(1-p)^{x_e}$, which models that by spending one unit of energy one has an extra chance of detecting the event with probability p. In this model, beyond deciding whether to place a sensor or not, one also needs to decide the optimal energy levels. Let $t_\infty = \max_{e\in E, v\in V} t(e,v)$, let e_v be the first sensor that detects event v (e_v is a random variable). One can define the objective as the expected detection time that could be *saved*,

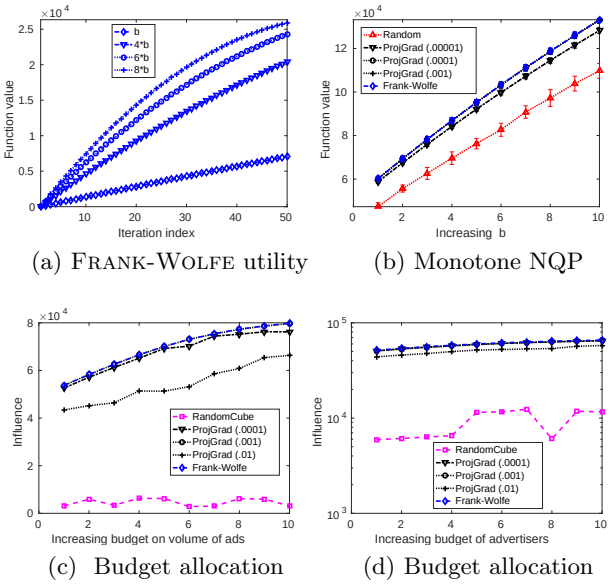
$$f(\boldsymbol{x}) := \mathbb{E}_{v\in V}\mathbb{E}_{e_v}[t_\infty - t(e_v, v)], \tag{7}$$

which is monotone DR-submodular. Maximizing $f(\boldsymbol{x})$ w.r.t. the cost constraints pursues the goal of finding the optimal energy levels of the sensors, to maximize the expected detection time that could be saved.

Other applications. More applications with submodular continuous objectives exist, e.g., multi-resolution summarization, facility location with continuous opening scales, maximum coverage with confidence level and the problem of text summarization with submodular objectives [38]. We defer details to Appendix F.

6 Experimental results

We compare the performance of our proposed algorithms, the Frank-Wolfe variant and DoubleGreedy, with the following baselines: a) Random: uniformly sample k_s solutions from the constraint set using the hit-and-run sampler [35], and select the best one. For the constraint set as a very high-dimensional polytope, this approach is computationally very expensive. To accelerate sampling from a high-dimensional polytope, we also use b) RandomCube: randomly sample k_s solutions from the hypercube, and decrease their elements until they are inside the polytope. In addition we consider c) ProjGrad: projected gradient ascent with an empirically tuned step size; and d) SingleGreedy: for non-monotone submodular functions maximization over a box constraint, we greedily increase each coordinate, as long as it remains feasible. This approach is similar to the coordinate ascent method. In all of the experiments, we use random order of coordinates for DoubleGreedy. We use constant step size for the Frank-Wolfe variant since it gives the tightest approximation guarantee (see Corollary 1). The performance of the methods are evaluated for the following tasks.



(a) Frank-Wolfe utility (b) Monotone NQP

(c) Budget allocation (d) Budget allocation

Figure 2: (Both Frank-Wolfe and ProjGrad were ran for 50 iterations): a) Frank-Wolfe function value for 4 instances with different $\boldsymbol{b}$; b) NQP function value returned w.r.t. different $\boldsymbol{b}$; c, d) Influence returned w.r.t. different budgets on volume of ads and different budgets of advertisers, respectively;

6.1 Results for monotone maximization

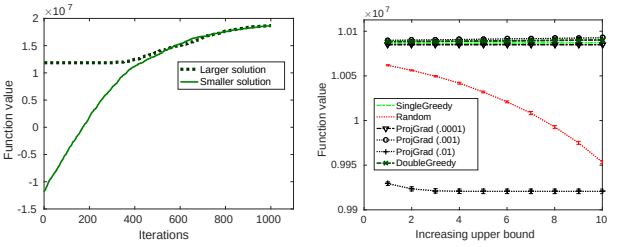
Monotone DR-submodular NQP. We randomly generated monotone DR-submodular NQP functions of the form $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \mathbf{H}\boldsymbol{x} + \boldsymbol{h}^\top \boldsymbol{x}$, where $\mathbf{H} \in \mathbb{R}^{n\times n}$ is a random matrix with *uniformly* distributed non-positive entries in $[-100, 0]$, $n = 100$. We further generated a set of $m = 50$ linear constraints to construct the positive polytope $\mathcal{P} = \{\boldsymbol{x} \in \mathbb{R}^n | \mathbf{A}\boldsymbol{x} \leq \boldsymbol{b}, 0 \leq \boldsymbol{x} \leq \bar{\boldsymbol{u}}\}$, where $\mathbf{A}$ has uniformly distributed entries in $[0, 1]$, $\boldsymbol{b} = \mathbf{1}, \bar{\boldsymbol{u}} = \mathbf{1}$. To make the gradient non-negative, we set $\boldsymbol{h} = -\mathbf{H}^\top \bar{\boldsymbol{u}}$. We empirically tuned step size α_p for ProjGrad and ran all algorithms for 50 iterations. Figure 2a shows the utility obtained by the Frank-Wolfe variant v.s. the iteration index for 4 function instances with different values of $\boldsymbol{b}$. Figure 2b shows the average utility obtained by different algorithms with increasing values of $\boldsymbol{b}$, which is the average of 20 repeated experiments. For ProjGrad, we plotted the curves for three different values of α_p. One can see that the performance of ProjGrad fluctuates with different step sizes. With the best-tuned step size, ProjGrad performs close to the Frank-Wolfe variant.

Optimal budget allocation. As our real-world experiments, we used the Yahoo! Search Marketing Advertiser Bidding Data[6], which consists of 1,000 search

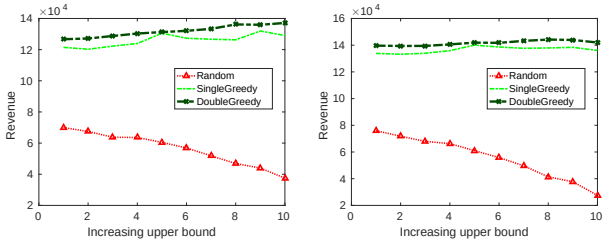
[6] https://webscope.sandbox.yahoo.com/catalog.php?datatype=a

keywords, 10,475 customers and 52,567 edges. We considered the frequency of (keyword, customer) pairs to estimate the influence probabilities, and used the average of the bidding prices to put a limit on the budget of each advertiser. Since the RANDOM sampling was too slow, we compared with the RANDOMCUBE method. Figures 2c and 2d show the value of the utility function (influence) when varying the budget on volume of ads and on budget of advertisers, respectively. Again, we observe that the FRANK-WOLFE variant outperforms the other baselines, and the performance of PROJGRAD highly depends on the choice of the step size.

6.2 Results for non-monotone maximization



(a) DOUBLEGREEDY utility (b) Non-monotone NQP



(c) $\alpha = \beta = \gamma = 10$ (d) $\alpha = 10, \beta = 5, \gamma = 10$

Figure 3: Non-monotone experiments. a) Function values of the two intermediate solutions of DOUBLEGREEDY in each iteration; b) Non-monotone NQP function value w.r.t. different upper bounds; c, d) Revenue returned with different upper bounds $\bar{\boldsymbol{u}}$ on the Youtube social network dataset.

Non-monotone submodular NQP. We randomly generated non-monotone submodular NQPs of the form $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \mathbf{H}\boldsymbol{x} + \boldsymbol{h}^\top \boldsymbol{x} + c$, where $\mathbf{H} \in \mathbb{R}^{n \times n}$ is a sparse matrix with *uniformly* distributed non-positive off-diagonal entries in $[-10, 0]$, $n = 1000$. We considered a matrix for which around 50% of the eigenvalues are positive and the other 50% are negative. We set $\boldsymbol{h} = -0.2 * \mathbf{H}^\top \bar{\boldsymbol{u}}$ to make $f(\boldsymbol{x})$ non-monotone. We then selected a value for c such that $f(0) + f(\bar{\boldsymbol{u}}) \geq 0$. PROJGRAD was executed for n iterations, with empirically tuned step sizes. For the RANDOM method we set $k_s = 1,000$. Figure 3a shows the utility of the two intermediate solutions maintained by DOUBLEGREEDY. One can observe that they both increase in each iteration. Figure 3b shows the values of the utility function for varying upper bound $\bar{\boldsymbol{u}}$. The result is the average over 20 repeated experiments. We can see that DOUBLEGREEDY has strong approximation performance, while PROJGRAD's results depend on the choice of the step size. With carefully hand-tuned step size, its performance is comparable to DOUBLEGREEDY.

Revenue maximization. W.l.o.g., we considered maximizing the revenue from selling one product (corresponding to $q = 1$, see Appendix E for more details on this model). Notice that the objective in Eq. 6 is generally non-smooth and *discontinuous* at any point $\boldsymbol{x}$ which contains the element of 0. Since the subdifferential can be empty, we cannot use the subgradient-based method and could not compare with PROJGRAD. We performed our experiments on the top 500 largest communities of the YouTube social network[7] consisting of 39,841 nodes and 224,235 edges. The edge weights were assigned according to a uniform distribution $U(0, 1)$. See Figure 3c, 3d for an illustration of revenue for varying upper bound ($\bar{\boldsymbol{u}}$) and different combinations of the parameters (α, β, γ) (see Eq. 6). For different values of the upper bound, DOUBLEGREEDY outperforms the other baselines, while SINGLEGREEDY maintaining only one intermediate solution obtained a lower utility than DOUBLEGREEDY.

7 Conclusion

We characterized submodular continuous functions, and proposed two approximation algorithms to efficiently maximize them. In particular, for maximizing monotone DR-submodular continuous functions s.t. general down-closed convex constraints, we proposed a $(1-1/e)$-approximation algorithm, and for maximizing non-monotone submodular continuous functions s.t. a box constraint, we proposed a 1/3-approximation algorithm. We demonstrate the effectiveness of our algorithms through a set of experiments on real-world applications, including budget allocation, revenue maximization, and submodular quadratic programming, and show that our proposed methods outperform the baselines in all the experiments. This work demonstrates that submodularity can ensure guaranteed optimization in the continuous setting for problems with non-convex/non-concave objectives.

Acknowledgements

The authors would like to thank Martin Jaggi for valuable discussions. This research was partially supported by ERC StG 307036 and the Max Planck ETH Center for Learning Systems.

[7] http://snap.stanford.edu/data/com-Youtube.html

References

[1] Alexander A Ageev and Maxim I Sviridenko. Pipage rounding: A new method of constructing algorithms with proven performance guarantee. *Journal of Combinatorial Optimization*, 8(3):307–328, 2004.

[2] Zeyuan Allen-Zhu and Elad Hazan. Variance reduction for faster non-convex optimization. *arXiv preprint arXiv:1603.05643*, 2016.

[3] Zeyuan Allen-Zhu and Lorenzo Orecchia. Nearly-linear time positive lp solver with faster convergence rate. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, pages 229–236. ACM, 2015.

[4] Animashree Anandkumar, Rong Ge, Daniel Hsu, Sham M Kakade, and Matus Telgarsky. Tensor decompositions for learning latent variable models. *JMLR*, 15(1):2773–2832, 2014.

[5] Francis Bach. Submodular functions: from discrete to continous domains. *arXiv:1511.00394*, 2015.

[6] Francis R Bach. Structured sparsity-inducing norms through submodular functions. In *NIPS*, pages 118–126, 2010.

[7] Andrew An Bian, Baharan Mirzasoleiman, Joachim M. Buhmann, and Andreas Krause. Guaranteed non-convex optimization: Submodular maximization over continuous domains. *arXiv preprint arXiv:1606.05615*, 2016.

[8] Jeffrey Bilmes and Wenruo Bai. Deep submodular functions. *arXiv preprint arXiv:1701.08939*, 2017.

[9] Niv Buchbinder, Moran Feldman, Joseph Seffi Naor, and Roy Schwartz. A tight linear time (1/2)-approximation for unconstrained submodular maximization. In *FOCS*, pages 649–658. IEEE, 2012.

[10] Gruia Călinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a submodular set function subject to a matroid constraint. In *Integer programming and combinatorial optimization*, pages 182–196. Springer, 2007.

[11] Gruia Călinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM J. Comput.*, 40(6):1740–1766, 2011.

[12] Abhimanyu Das and David Kempe. Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection. *arXiv preprint arXiv:1102.3975*, 2011.

[13] Josip Djolonga and Andreas Krause. From map to marginals: Variational inference in bayesian submodular models. In *NIPS*, pages 244–252, 2014.

[14] Shahar Dobzinski and Jan Vondrák. From query complexity to computational complexity. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 1107–1116. ACM, 2012.

[15] Reza Eghbali and Maryam Fazel. Designing smoothing functions for improved worst-case competitive ratio in online optimization. In *NIPS*, pages 3279–3287. 2016.

[16] Alina Ene and Huy L Nguyen. A reduction for optimizing lattice submodular functions with diminishing returns. *arXiv preprint arXiv:1606.08362*, 2016.

[17] Uriel Feige. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.

[18] Uriel Feige, Vahab S Mirrokni, and Jan Vondrak. Maximizing non-monotone submodular functions. *SIAM Journal on Computing*, 40(4):1133–1153, 2011.

[19] Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2):95–110, 1956.

[20] Satoru Fujishige. *Submodular functions and optimization*, volume 58. Elsevier, 2005.

[21] Jennifer Gillenwater, Alex Kulesza, and Ben Taskar. Near-optimal map inference for determinantal point processes. In *Advances in Neural Information Processing Systems*, pages 2735–2743, 2012.

[22] Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, pages 427–486, 2011.

[23] Corinna Gottschalk and Britta Peis. Submodular function maximization on the bounded integer lattice. In *Approximation and Online Algorithms*, pages 133–144. Springer, 2015.

[24] Jason Hartline, Vahab Mirrokni, and Mukund Sundararajan. Optimal marketing strategies over social networks. In *Proceedings of the 17th international conference on World Wide Web*, pages 189–198. ACM, 2008.

[25] Daisuke Hatano, Takuro Fukunaga, Takanori Maehara, and Ken-ichi Kawarabayashi. Lagrangian decomposition algorithm for allocating marketing channels. In *AAAI*, pages 1144–1150, 2015.

[26] Elad Hazan, Kfir Y Levy, and Shai Shalev-Swartz. On graduated optimization for stochastic non-convex problems. *arXiv preprint arXiv:1503.03712*, 2015.

[27] Satoru Iwata, Lisa Fleischer, and Satoru Fujishige. A combinatorial strongly polynomial algorithm for minimizing submodular functions. *Journal of the ACM*, 48(4):761–777, 2001.

[28] Martin Jaggi. Revisiting frank-wolfe: Projection-free sparse convex optimization. In *ICML 2013*, pages 427–435, 2013.

[29] Majid Janzamin, Hanie Sedghi, and Anima Anandkumar. Beating the perils of non-convexity: Guaranteed training of neural networks using tensor methods. *CoRR abs/1506.08473*, 2015.

[30] Sunyoung Kim and Masakazu Kojima. Exact solutions of some nonconvex quadratic optimization problems via sdp and socp relaxations. *Computational Optimization and Applications*, 26(2):143–154, 2003.

[31] Vladimir Kolmogorov. Submodularity on a tree: Unifying $l^\natural$-convex and bisubmodular functions. In *Mathematical Foundations of Computer Science*, pages 400–411. Springer, 2011.

[32] Andreas Krause and Volkan Cevher. Submodular dictionary selection for sparse representation. In *ICML*, pages 567–574, 2010.

[33] Andreas Krause and Daniel Golovin. Submodular function maximization. *Tractability: Practical Approaches to Hard Problems*, 3:19, 2012.

[34] Andreas Krause and Carlos Guestrin. Near-optimal nonmyopic value of information in graphical models. In *UAI*, pages 324–331, 2005.

[35] Dirk P Kroese, Thomas Taimre, and Zdravko I Botev. *Handbook of Monte Carlo Methods*, volume 706. John Wiley & Sons, 2013.

[36] Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne VanBriesen, and Natalie Glance. Cost-effective outbreak detection in networks. In *ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 420–429, 2007.

[37] Huan Li and Zhouchen Lin. Accelerated proximal gradient methods for nonconvex programming. In *NIPS*, pages 379–387, 2015.

[38] Hui Lin and Jeff Bilmes. Multi-document summarization via budgeted maximization of submodular functions. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 912–920, 2010.

[39] Hui Lin and Jeff Bilmes. A class of submodular functions for document summarization. In *HLT*, 2011.

[40] László Lovász. Submodular functions and convexity. In *Mathematical Programming The State of the Art*, pages 235–257. Springer, 1983.

[41] Baharan Mirzasoleiman, Amin Karbasi, Rik Sarkar, and Andreas Krause. Distributed submodular maximization: Identifying representative elements in massive data. In *NIPS*, pages 2049–2057, 2013.

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

[43] Yurii Nesterov and Boris T Polyak. Cubic regularization of newton method and its global performance. *Mathematical Programming*, 108(1):177–205, 2006.

[44] BT Polyak. Gradient methods for the minimisation of functionals. *USSR Computational Mathematics and Mathematical Physics*, 3(4):864–878, 1963.

[45] Sashank J Reddi, Suvrit Sra, Barnabas Poczos, and Alex Smola. Fast stochastic methods for nonsmooth nonconvex optimization. *arXiv preprint arXiv:1605.06900*, 2016.

[46] Ajit P Singh, Andrew Guillory, and Jeff Bilmes. On bisubmodular maximization. In *International Conference on Artificial Intelligence and Statistics*, pages 1055–1063, 2012.

[47] Martin Skutella. Convex quadratic and semidefinite programming relaxations in scheduling. *J. ACM*, 2001.

[48] Tasuku Soma, Naonori Kakimura, Kazuhiro Inaba, and Ken-ichi Kawarabayashi. Optimal budget allocation: Theoretical guarantee and efficient algorithm. In *ICML*, pages 351–359, 2014.

[49] Tasuku Soma and Yuichi Yoshida. A generalization of submodular cover via the diminishing return property on the integer lattice. In *NIPS*, pages 847–855, 2015.

[50] Tasuku Soma and Yuichi Yoshida. Maximizing submodular functions with the diminishing return property over the integer lattice. *arXiv preprint arXiv:1503.01218*, 2015.

[51] Suvrit Sra. Scalable nonconvex inexact proximal splitting. In *NIPS*, pages 530–538, 2012.

[52] Maxim Sviridenko. A note on maximizing a submodular set function subject to a knapsack constraint. *Operations Research Letters*, 32(1):41–43, 2004.

[53] Donald M Topkis. Minimizing a submodular function on a lattice. *Operations research*, 26(2):305–321, 1978.

[54] Jan Vondrák. Optimal approximation for the submodular welfare problem in the value oracle model. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pages 67–74, 2008.

[55] Justin Ward and Stanislav Zivny. Maximizing bisubmodular and k-submodular functions. In *SODA 2014*, pages 1468–1481, 2014.

[56] Laurence A. Wolsey. Maximising real-valued submodular functions: Primal and dual heuristics for location problems. *Math. Oper. Res.*, 7(3):410–425, 1982.

Appendix

A More on background and related work

Notions of submodularity. Submodularity is often viewed as a discrete analogue of convexity, and provides computationally effective structure so that many discrete problems with this property are efficiently solvable or approximable. Of particular interest is a $(1-1/e)$-approximation for maximizing a monotone submodular set function subject to a cardinality, a matroid, or a knapsack constraint [42, 54, 52]. Another result relevant to this work is unconstrained maximization of non-monotone submodular set functions, for which [9] propose the deterministic double greedy algorithm with 1/3 approximation guarantee, and the randomized double greedy algorithm which achieves the tight 1/2 approximation guarantee.

Although most commonly associated with set functions, in many practical scenarios, it is natural to consider generalizations of submodular set functions. [22] introduce the notion of adaptive submodularity to generalize submodular set functions to adaptive policies. [31] studies tree-submodular functions and presents a polynomial algorithm for minimizing them. For distributive lattices, it is well-known that the combinatorial polynomial-time algorithms for minimizing a submodular set function can be adopted to minimize a submodular function over a bounded integer lattice [20]. Recently, maximizing a submodular function over integer lattices has attracted considerable attention. In particular, [48] develop a $(1-1/e)$-approximation algorithm for maximizing a monotone DR-submodular integer-lattice function under a knapsack constraint. For non-monotone submodular functions over the bounded integer lattice, [23] provide a 1/3-approximation. Approximation algorithms for maximizing bisubmodular functions and k-submodular functions have also been proposed by [46, 55].

[56] considers maximizing a special class of submodular continuous functions subject to one knapsack constraint, in the context of solving location problems. That class of functions are additionally required to be monotone, piecewise linear and concave. [10, 54] discuss a subclass of submodular continuous functions, which is termed smooth submodular functions[8], to describe the multilinear extension of a submodular set function. They propose the continuous greedy algorithm, which has a $(1-1/e)$ approximation guarantee on maximizing a smooth submodular functions under a down-monotone polytope constraint. Recently, [5] considers the minimization of a submodular continuous function, and proves that efficient techniques from convex optimization may be used for minimization. Very recently, [16] provide a reduction from a integer-lattice DR-submodular function maximization problem to a submodular set function maximization problem, which suggests a way to optimize submodular continuous functions over *simple* continuous constriants: Discretize the continuous function and constraint to be an integer-lattice instance, and then optimize it using the reduction. However, for monotone DR-submodular functions maximization, this method can not handle the general continuous constraints discussed in this work, i.e., arbitrary down-closed convex sets. And for general submodular function maximization, this method cannot be applied, since the reduction needs the additional diminishing returns property. Therefore we focus on continuous methods in this work.

Non-convex optimization. Optimizing non-convex continuous functions has received renewed interest in the last decades. Recently, tensor methods have been used in various non-convex problems, e.g., learning latent variable models [4] and training neural networks [29]. A fundamental problem in non-convex optimization is to reach a stationary point assuming the smoothness of the objective [51, 37, 45, 2]. With extra assumptions, certain global convergence results can be obtained. For example, for functions with Lipschitz continuous Hessians, the regularized Newton scheme of [43] achieves global convergence results for functions with an additional star-convexity property or with an additional gradient-dominance property [44]. [26] introduce the family of σ-nice functions and propose a graduated optimization-based algorithm, that provably converges to a global optimum for this family of (generally) non-convex functions. However, it is typically difficult to verify whether these assumptions hold in real-world problems.

To the best of our knowledge, this work is the *first* to address the general problem of monotone and non-monotone submodular maximization over continuous domains. It is also the first to propose a sufficient and necessary diminishing-return-style characterization of submodularity for general functions. We propose efficient algorithms with strong approximation guarantees. We further show that continuous submodularity is a common

[8] A function $f:[0,1]^n \to \mathbb{R}$ is smooth submodular if it has second partial derivatives everywhere and all entries of its Hessian matrix are non-positive.

property of many well-known objectives and finds various real-world applications.

B Proofs of properties of submodular continuous functions

Since $\mathcal{X}_i$ is a compact subset of $\mathbb{R}$, we denote its lower bound and upper bound to be $\underline{u}_i$ and $\bar{u}_i$, respectively in this section.

B.1 Alternative formulation of the `weak DR` property

First of all, we will prove that `weak DR` has the following alternative formulation, which will be used to prove Proposition 1.

Lemma 3 (Alternative formulation of `weak DR`)**.** *The `weak DR` property (Eq. 3, denoted as `Formulation I`) has the following equilvalent formulation (Eq. 8, denoted as `Formulation II`):* $\forall \boldsymbol{a} \leq \boldsymbol{b} \in \mathcal{X}$, $\forall i \in \{i' | a_{i'} = b_{i'} = \underline{u}_{i'}\}, \forall k' \geq l' \geq 0$ *s.t.* $(k'\boldsymbol{\chi}_i + \boldsymbol{a})$, $(l'\boldsymbol{\chi}_i + \boldsymbol{a})$, $(k'\boldsymbol{\chi}_i + \boldsymbol{b})$ *and* $(l'\boldsymbol{\chi}_i + \boldsymbol{b})$ *are still in* $\mathcal{X}$*, the following inequality is satisfied,*

$$f(k'\boldsymbol{\chi}_i + \boldsymbol{a}) - f(l'\boldsymbol{\chi}_i + \boldsymbol{a}) \geq f(k'\boldsymbol{\chi}_i + \boldsymbol{b}) - f(l'\boldsymbol{\chi}_i + \boldsymbol{b}) \quad \textit{(Formulation II)} \tag{8}$$

Proof. Let $D_1 = \{i | a_i = b_i = \underline{u}_i\}$, $D_2 = \{i | \underline{u}_i < a_i = b_i < \bar{u}_i\}$, and $D_3 = \{i | a_i = b_i = \bar{u}_i\}$.

1) `Formulation II` $\Rightarrow$ `Formulation I`

When $i \in D_1$, set $l' = 0$ in `Formulation II` one can get $f(k'\boldsymbol{\chi}_i + \boldsymbol{a}) - f(\boldsymbol{a}) \geq f(k'\boldsymbol{\chi}_i + \boldsymbol{b}) - f(\boldsymbol{b})$.

When $i \in D_2$, $\forall k \geq 0$, let $l' = a_i - \underline{u}_i = b_i - \underline{u}_i > 0$, $k' = k + l' = k + (a_i - \underline{u}_i)$, and let $\bar{\boldsymbol{a}} = (\boldsymbol{a} | a_i \leftarrow \underline{u}_i)$, $\bar{\boldsymbol{b}} = (\boldsymbol{b} | b_i \leftarrow \underline{u}_i)$. It is easy to see that $\bar{\boldsymbol{a}} \leq \bar{\boldsymbol{b}}$, and $\bar{a}_i = \bar{b}_i = \underline{u}_i$. Then from `Formulation II`,

$$\begin{aligned} &f(k'\boldsymbol{\chi}_i + \bar{\boldsymbol{a}}) - f(l'\boldsymbol{\chi}_i + \bar{\boldsymbol{a}}) = f(k\boldsymbol{\chi}_i + \boldsymbol{a}) - f(\boldsymbol{a}) \\ &\geq f(k'\boldsymbol{\chi}_i + \bar{\boldsymbol{b}}) - f(l'\boldsymbol{\chi}_i + \bar{\boldsymbol{b}}) = f(k\boldsymbol{\chi}_i + \boldsymbol{b}) - f(\boldsymbol{b}). \end{aligned}$$

When $i \in D_3$, Eq. 3 holds trivially.

The above three situations proves the `Formulation I`.

2) `Formulation II` $\Leftarrow$ `Formulation I`

$\forall \boldsymbol{a} \leq \boldsymbol{b}$, $\forall i \in D_1$, one has $a_i = b_i = \underline{u}_i$. $\forall k' \geq l' \geq 0$, let $\hat{\boldsymbol{a}} = l'\boldsymbol{\chi}_i + \boldsymbol{a}, \hat{\boldsymbol{b}} = l'\boldsymbol{\chi}_i + \boldsymbol{b}$, let $k = k' - l' \geq 0$, it can be verified that $\hat{\boldsymbol{a}} \leq \hat{\boldsymbol{b}}$ and $\hat{a}_i = \hat{b}_i$, from `Formulation I`,

$$\begin{aligned} &f(k\boldsymbol{\chi}_i + \hat{\boldsymbol{a}}) - f(\hat{\boldsymbol{a}}) = f(k'\boldsymbol{\chi}_i + \boldsymbol{a}) - f(l'\boldsymbol{\chi}_i + \boldsymbol{a}) \\ \geq &f(k\boldsymbol{\chi}_i + \hat{\boldsymbol{b}}) - f(\hat{\boldsymbol{b}}) = f(k'\boldsymbol{\chi}_i + \boldsymbol{b}) - f(l'\boldsymbol{\chi}_i + \boldsymbol{b}) \end{aligned}$$

which proves `Formulation II`. □

B.2 Proof of Proposition 1

Proof. 1) `submodularity` $\Rightarrow$ `weak DR`:

Let us prove the `Formulation II` (Eq. 8) of `weak DR`, which is,

$\forall \boldsymbol{a} \leq \boldsymbol{b} \in \mathcal{X}$, $\forall i \in \{i' | a_{i'} = b_{i'} = \underline{u}_{i'}\}, \forall k' \geq l' \geq 0$, the following inequality holds,

$$f(k'\boldsymbol{\chi}_i + \boldsymbol{a}) - f(l'\boldsymbol{\chi}_i + \boldsymbol{a}) \geq f(k'\boldsymbol{\chi}_i + \boldsymbol{b}) - f(l'\boldsymbol{\chi}_i + \boldsymbol{b}).$$

And f is a submodular function iff $\forall \boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}$, $f(\boldsymbol{x}) + f(\boldsymbol{y}) \geq f(\boldsymbol{x} \vee \boldsymbol{y}) + f(\boldsymbol{x} \wedge \boldsymbol{y})$, so $f(\boldsymbol{y}) - f(\boldsymbol{x} \wedge \boldsymbol{y}) \geq f(\boldsymbol{x} \vee \boldsymbol{y}) - f(\boldsymbol{x})$.

Now $\forall \boldsymbol{a} \leq \boldsymbol{b} \in \mathcal{X}$, one can set $\boldsymbol{x} = l'\boldsymbol{\chi}_i + \boldsymbol{b}$ and $\boldsymbol{y} = k'\boldsymbol{\chi}_i + \boldsymbol{a}$. It can be easily verified that $\boldsymbol{x} \wedge \boldsymbol{y} = l'\boldsymbol{\chi}_i + \boldsymbol{a}$ and $\boldsymbol{x} \vee \boldsymbol{y} = k'\boldsymbol{\chi}_i + \boldsymbol{b}$. Substituting all the above equalities into $f(\boldsymbol{y}) - f(\boldsymbol{x} \wedge \boldsymbol{y}) \geq f(\boldsymbol{x} \vee \boldsymbol{y}) - f(\boldsymbol{x})$ one can get $f(k'\boldsymbol{\chi}_i + \boldsymbol{a}) - f(l'\boldsymbol{\chi}_i + \boldsymbol{a}) \geq f(k'\boldsymbol{\chi}_i + \boldsymbol{b}) - f(l'\boldsymbol{\chi}_i + \boldsymbol{b})$.

2) `submodularity` $\Leftarrow$ `weak DR`:

Let us use `Formulation I` (Eq. 3) of `weak DR` to prove the `submodularity` property.

$\forall \boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}$, let $D := \{e_1, \cdots, e_d\}$ to be the set of elements for which $y_e > x_e$, let $k_{e_i} := y_{e_i} - x_{e_i}$. Now set $\boldsymbol{a}^0 := \boldsymbol{x} \wedge \boldsymbol{y}, \boldsymbol{b}^0 := \boldsymbol{x}$ and $\boldsymbol{a}^i = (\boldsymbol{a}^{i-1} | a^{i-1}_{e_i} \leftarrow y_{e_i}) = k_{e_i}\boldsymbol{\chi}_i + \boldsymbol{a}^{i-1}, \boldsymbol{b}^i = (\boldsymbol{b}^{i-1} | b^{i-1}_{e_i} \leftarrow y_{e_i}) = k_{e_i}\boldsymbol{\chi}_i + \boldsymbol{b}^{i-1}$, for $i = 1, \cdots, d$. One can verify that $\boldsymbol{a}^i \leq \boldsymbol{b}^i, a^i_{e_{i'}} = b^i_{e_{i'}}$ for all $i' \in D, i = 0, \cdots, d$, and that $\boldsymbol{a}^d = \boldsymbol{y}, \boldsymbol{b}^d = \boldsymbol{x} \vee \boldsymbol{y}$.

Applying Eq. 3 of the `weak DR` property for $i = 1, \cdots, d$ one can get

$$
\begin{aligned}
&f(k_{e_1}\boldsymbol{\chi}_{e_1} + \boldsymbol{a}^0) - f(\boldsymbol{a}^0) \geq f(k_{e_1}\boldsymbol{\chi}_{e_1} + \boldsymbol{b}^0) - f(\boldsymbol{b}^0) \\
&f(k_{e_2}\boldsymbol{\chi}_{e_2} + \boldsymbol{a}^1) - f(\boldsymbol{a}^1) \geq f(k_{e_2}\boldsymbol{\chi}_{e_2} + \boldsymbol{b}^1) - f(\boldsymbol{b}^1) \\
&\cdots \\
&f(k_{e_d}\boldsymbol{\chi}_{e_d} + \boldsymbol{a}^{d-1}) - f(\boldsymbol{a}^{d-1}) \geq f(k_{e_d}\boldsymbol{\chi}_{e_d} + \boldsymbol{b}^{d-1}) - f(\boldsymbol{b}^{d-1}).
\end{aligned}
$$

Taking a sum over all the above d inequalities, one can get

$$
\begin{aligned}
&f(k_{e_d}\boldsymbol{\chi}_{e_d} + \boldsymbol{a}^{d-1}) - f(\boldsymbol{a}^0) \geq f(k_{e_d}\boldsymbol{\chi}_{e_d} + \boldsymbol{b}^{d-1}) - f(\boldsymbol{b}^0) \Leftrightarrow \\
&f(\boldsymbol{y}) - f(\boldsymbol{x} \wedge \boldsymbol{y}) \geq f(\boldsymbol{x} \vee \boldsymbol{y}) - f(\boldsymbol{x}) \Leftrightarrow \\
&f(\boldsymbol{x}) + f(\boldsymbol{y}) \geq f(\boldsymbol{x} \vee \boldsymbol{y}) + f(\boldsymbol{x} \wedge \boldsymbol{y}),
\end{aligned}
$$

which proves the submodularity. □

B.3 Proof of Proposition 2

Proof. 1) `submodular` + `coordinate-wise concave` ⇒ `DR`:

From coordinate-wise concavity we have $f(\boldsymbol{a}+k\boldsymbol{\chi}_i) - f(\boldsymbol{a}) \geq f(\boldsymbol{a}+(b_i-a_i+k)\boldsymbol{\chi}_i) - f(\boldsymbol{a}+(b_i-a_i)\boldsymbol{\chi}_i)$. Therefore, to prove DR it suffices to show that

$$
f(\boldsymbol{a} + (b_i - a_i + k)\boldsymbol{\chi}_i) - f(\boldsymbol{a} + (b_i - a_i)\boldsymbol{\chi}_i) \geq f(\boldsymbol{b} + k\boldsymbol{\chi}_i) - f(\boldsymbol{b}). \tag{9}
$$

Let $\boldsymbol{x} := \boldsymbol{b}, \boldsymbol{y} := (\boldsymbol{a} + (b_i - a_i + k)\boldsymbol{\chi}_i)$, so $\boldsymbol{x} \wedge \boldsymbol{y} = (\boldsymbol{a} + (b_i - a_i)\boldsymbol{\chi}_i), \boldsymbol{x} \vee \boldsymbol{y} = (\boldsymbol{b} + k\boldsymbol{\chi}_i)$. From submodularity, one can see that inequality 9 holds.

2) `submodular` + `coordinate-wise concave` ⇐ `DR`:

From `DR` property, the `weak DR` (Eq. 3) property is implied, which equivalently proves the *submodularity* property.

To prove *coordinate-wise concavity*, one just need to set $\boldsymbol{b} := \boldsymbol{a} + l\boldsymbol{\chi}_i$, then it reads $f(\boldsymbol{a} + k\boldsymbol{\chi}_i) - f(\boldsymbol{a}) \geq f(\boldsymbol{a} + (k+l)\boldsymbol{\chi}_i) - f(\boldsymbol{a} + l\boldsymbol{\chi}_i)$. □

C Proofs for the monotone DR-submodular continuous functions maximization

C.1 Proof of Proposition 3

Proof. On a high level, the proof idea follows from the reduction from the problem of maximizing a monotone submodular set function subject to cardinality constraints.

Let us denote Π_1 as the problem of maximizing a monotone submodular set function subject to cardinality constraints, and Π_2 as the problem of maximizing a monotone DR-submodular continuous function under general down-closed polytope constraints. Following [11], there exist an algorithm $\mathscr{A}$ for Π_1 that consists of a polynomial time computation in addition to polynomial number of subroutine calls to an algorithm for Π_2. For details on $\mathscr{A}$ see the following.

First of all, the multilinear extension [10] of a monotone submodular set function is a monotone submodular continuous function, and it is coordinate-wise linear, thus falls into a special case of monotone DR-submodular continuous functions.

So the algorithm $\mathscr{A}$ can be: 1) Maximize the multilinear extension of the submodular set function over the matroid polytope associated with the cardinality constraint, which can be achieved by solving an instance of Π_2. We call the solution obtained the fractional solution; 2) Round the fractional solution to a feasible integeral

solution using polynomial time rounding technique in [1, 10] (called the pipage rounding). Thus we prove the reduction from Π_1 to Π_2.

Our reduction algorithm $\mathscr{A}$ implies the NP-hardness and inapproximability of problem Π_2.

For the NP-hardness, because Π_1 is well known to be NP-hard [10, 17], so Π_2 is NP-hard as well.

For the inapproximability: Assume there exists a polynomial algorithm $\mathscr{B}$ that can solve Π_2 better than $1-1/e$, then we can use $\mathscr{B}$ as the subroutine algorithm in the reduction, which implies that one can solve Π_1 better than $1-1/e$. Now we slightly adapt the proof of inapproximability on max-k-cover in [17], since max-k-cover is a special case of Π_1. According to Theorem 5.3 in [17] and our reduction $\mathscr{A}$, we have a reduction from approximating 3SAT-5 to problem Π_2. Using the rest proof of Theorem 5.3 in [17], we reach the result that one cannot solve Π_2 better than $1-1/e$, unless RP = NP. □

C.2 Proof of Proposition 4

Proof. Consider a function $g(\xi) := f(\boldsymbol{x}+\xi\boldsymbol{v}^*), \xi \geq 0, \boldsymbol{v}^* \geq 0$. $\frac{dg(\xi)}{d\xi} = \langle \boldsymbol{v}^*, \nabla f(\boldsymbol{x}+\xi\boldsymbol{v}^*)\rangle$.

$g(\xi)$ is concave $\Leftrightarrow$

$$\frac{d^2 g(\xi)}{d\xi^2} = (\boldsymbol{v}^*)^\top \nabla^2 f(\boldsymbol{x}+\xi\boldsymbol{v}^*)\boldsymbol{v}^* = \sum_{i\neq j} v_i^* v_j^* \nabla^2_{ij} f + \sum_i (v_i^*)^2 \nabla^2_{ii} f \leq 0$$

The non-positiveness of $\nabla^2_{ij} f$ is ensured by submodularity of $f(\cdot)$, and the non-positiveness of $\nabla^2_{ii} f$ results from the coordinate-wise concavity of $f(\cdot)$.

The proof of concavity along any non-positive direction is similar, which is omitted here. □

C.3 Proof of Lemma 1

Proof. It is easy to see that $\boldsymbol{x}^K$ is a convex linear combination of points in $\mathcal{P}$, so $\boldsymbol{x}^K \in \mathcal{P}$.

Consider the point $\boldsymbol{v}^* := (\boldsymbol{x}^* \vee \boldsymbol{x}) - \boldsymbol{x} = (\boldsymbol{x}^* - \boldsymbol{x}) \vee 0 \geq 0$. Because $\boldsymbol{v}^* \leq \boldsymbol{x}^*$ and $\mathcal{P}$ is down-closed, we get $\boldsymbol{v}^* \in \mathcal{P}$. By monotonicity, $f(\boldsymbol{x}+\boldsymbol{v}^*) = f(\boldsymbol{x}^* \vee \boldsymbol{x}) \geq f(\boldsymbol{x}^*)$.

Consider the function $g(\xi) := f(\boldsymbol{x}+\xi\boldsymbol{v}^*), \xi \geq 0$. $\frac{dg(\xi)}{d\xi} = \langle \boldsymbol{v}^*, \nabla f(\boldsymbol{x}+\xi\boldsymbol{v}^*)\rangle$. From Proposition 4, $g(\xi)$ is concave, hence

$$g(1) - g(0) = f(\boldsymbol{x}+\boldsymbol{v}^*) - f(\boldsymbol{x}) \leq \frac{dg(\xi)}{d\xi}\Big|_{\xi=0} \times 1 = \langle \boldsymbol{v}^*, \nabla f(\boldsymbol{x})\rangle.$$

Then one can get

$$\begin{aligned}
&\langle \boldsymbol{v}, \nabla f(\boldsymbol{x})\rangle \overset{(a)}{\geq} \alpha \langle \boldsymbol{v}^*, \nabla f(\boldsymbol{x})\rangle - \frac{1}{2}\delta L \geq \\
&\alpha(f(\boldsymbol{x}+\boldsymbol{v}^*) - f(\boldsymbol{x})) - \frac{1}{2}\delta L \geq \alpha(f(\boldsymbol{x}^*) - f(\boldsymbol{x})) - \frac{1}{2}\delta L
\end{aligned}$$

where (a) is from the selection rule in Step 3 of Algorithm1. □

C.4 Proof of Theorem 1

Proof. From the Lipschitz continuous derivative assumption of $g(\cdot)$ (Eq. 4):

$$\begin{aligned}
f(\boldsymbol{x}^{k+1}) - f(\boldsymbol{x}^k) &= f(\boldsymbol{x}^k + \gamma_k \boldsymbol{v}^k) - f(\boldsymbol{x}^k) \\
&= g(\gamma_k) - g(0) \\
&\geq \gamma_k \langle \boldsymbol{v}^k, \nabla f(\boldsymbol{x}^k)\rangle - \frac{L}{2}\gamma_k^2 \quad \text{(Lipschitz assumption in Eq. 4)} \\
&\geq \gamma_k \alpha [f(\boldsymbol{x}^*) - f(\boldsymbol{x}^k)] - \frac{1}{2}\gamma_k \delta L - \frac{L}{2}\gamma_k^2 \quad \text{(Lemma 1)}
\end{aligned}$$

After rearrangement,

$$f(\boldsymbol{x}^{k+1}) - f(\boldsymbol{x}^*) \geq (1 - \alpha\gamma_k)[f(\boldsymbol{x}^k) - f(\boldsymbol{x}^*)] - \frac{1}{2}\gamma_k \delta L - \frac{L}{2}\gamma_k^2$$

Therefore,

$$f(\boldsymbol{x}^K) - f(\boldsymbol{x}^*) \geq \prod_{k=0}^{K-1}(1 - \alpha\gamma_k)[f(0) - f(\boldsymbol{x}^*)] - \frac{\delta L}{2}\sum_{k=0}^{K-1}\gamma_k - \frac{L}{2}\sum_{k=0}^{K-1}\gamma_k^2.$$

One can observe that $\sum_{k=0}^{K-1}\gamma_k = 1$, and since $1 - y \leq e^{-y}$ when $y \geq 0$,

$$\begin{aligned} f(\boldsymbol{x}^*) - f(\boldsymbol{x}^K) &\leq [f(\boldsymbol{x}^*) - f(0)]e^{-\alpha\sum_{k=0}^{K-1}\gamma_k} + \frac{\delta L}{2} + \frac{L}{2}\sum_{k=0}^{K-1}\gamma_k^2 \\ &= [f(\boldsymbol{x}^*) - f(0)]e^{-\alpha} + \frac{\delta L}{2} + \frac{L}{2}\sum_{k=0}^{K-1}\gamma_k^2. \end{aligned}$$

After rearrangement, we get,

$$f(\boldsymbol{x}^K) \geq (1 - 1/e^{\alpha})f(\boldsymbol{x}^*) - \frac{L}{2}\sum_{k=0}^{K-1}\gamma_k^2 - \frac{L\delta}{2} + e^{-\alpha}f(0).$$

□

C.5 Proof of Corollary 1

Proof. Fixing K, to reach the tightest bound in Theorem 1 amounts to solving the following problem:

$$\begin{aligned} &\min \sum_{k=0}^{K-1}\gamma_k^2 \\ &\text{s.t.} \sum_{k=0}^{K-1}\gamma_k = 1, \gamma_k \geq 0. \end{aligned}$$

Using Lagrangian method, let λ be the Lagrangian multiplier, then

$$L(\gamma_0, \cdots, \gamma_{K-1}, \lambda) = \sum_{k=0}^{K-1}\gamma_k^2 + \lambda\left[\sum_{k=0}^{K-1}\gamma_k - 1\right].$$

It can be easily verified that when $\gamma_0 = \cdots = \gamma_{K-1} = K^{-1}$, $\sum_{k=0}^{K-1}\gamma_k^2$ reaches the minimum (which is K^{-1}). Therefore we obtain the tightest worst-case bound in Corollary 1. □

D Proofs for the non-monotone submodular continuous functions maximization

D.1 Proof of Proposition 5

Proof. The main proof follows from the reduction from the problem of maximizing an unconstrained non-monotone submodular set function.

Let us denote Π_1 as the problem of maximizing an unconstrained non-monotone submodular set function, and Π_2 as the problem of maximizing a box constrained non-monotone submodular continuous function. Following the Appendix A of [9], there exist an algorithm $\mathscr{A}$ for Π_1 that consists of a polynomial time computation in addition to polynomial number of subroutine calls to an algorithm for Π_2. For details see the following.

Given a submodular set function $F : 2^E \to \mathbb{R}_+$, its multilinear extension [10] is a function $f : [0,1]^E \to \mathbb{R}_+$, whose value at a point $\boldsymbol{x} \in [0,1]^E$ is the expected value of F over a random subset $R(\boldsymbol{x}) \subseteq E$, where $R(\boldsymbol{x})$ contains each

element $e \in E$ independently with probability x_e. Formally, $f(\boldsymbol{x}) := \mathbb{E}[R(\boldsymbol{x})] = \sum_{S \subseteq E} F(S) \prod_{e \in S} x_e \prod_{e' \notin S} (1 - x_{e'})$. It can be easily seen that $f(\boldsymbol{x})$ is a non-monotone submodular continuous function.

Then the algorithm $\mathscr{A}$ can be: 1) Maximize the multilinear extension $f(\boldsymbol{x})$ over the box constraint $[0,1]^E$, which can be achieved by solving an instance of Π_2. Obtain the fractional solution $\hat{\boldsymbol{x}} \in [0,1]^n$; 2) Return the random set $R(\hat{\boldsymbol{x}})$. According to the definition of multilinear extension, the expected value of $F(R(\hat{\boldsymbol{x}}))$ is $f(\hat{\boldsymbol{x}})$. Thus proving the reduction from Π_1 to Π_2.

Given the reduction, the hardness result follows from the hardness of unconstrained non-monotone submodular set function maximization.

The inapproximability result comes from that of the unconstrained non-monotone submodular set function maximization in [18] and [14]. □

D.2 Proof of Theorem 2

To better illustrate the proof, we reformulate Algorithm2 into its equivalent form in Algorithm3, where we split the update into two steps: when $\delta_a \geq \delta_b$, update $\boldsymbol{x}$ first while keeping $\boldsymbol{y}$ fixed and then update $\boldsymbol{y}$ first while keeping $\boldsymbol{x}$ fixed ($\boldsymbol{x}^i \leftarrow (\boldsymbol{x}^{i-1} | x^{i-1}_{e_i} \leftarrow \hat{u}_a)$, $\boldsymbol{y}^i \leftarrow \boldsymbol{y}^{i-1}$; $\boldsymbol{x}^{i+1} \leftarrow \boldsymbol{x}^i$, $\boldsymbol{y}^{i+1} \leftarrow (\boldsymbol{y}^i | y^i_{e_i} \leftarrow \hat{u}_a)$), when $\delta_a < \delta_b$, update $\boldsymbol{y}$ first. This iteration index change is only used to ease the analysis.

To prove the theorem, we first prove the following Lemmas.

Algorithm 3: DoubleGreedy algorithm reformulation (for analysis only)

Input: $\max f(\boldsymbol{x})$, $\boldsymbol{x} \in [\underline{\boldsymbol{u}}, \bar{\boldsymbol{u}}]$, f is generally non-monotone, $f(\underline{\boldsymbol{u}}) + f(\bar{\boldsymbol{u}}) \geq 0$

$\boldsymbol{x}^0 \leftarrow \underline{\boldsymbol{u}}$, $\boldsymbol{y}^0 \leftarrow \bar{\boldsymbol{u}}$;

for $i = 1, 3, 5, \cdots, 2n-1$ **do**

  find $\hat{u}_a$ s.t. $f(\boldsymbol{x}^{i-1} | x^{i-1}_{e_i} \leftarrow \hat{u}_a) \geq \max_{u_a \in [\underline{u}_{e_i}, \bar{u}_{e_i}]} f(\boldsymbol{x}^{i-1} | x^{i-1}_{e_i} \leftarrow u_a) - \delta$, $\delta_a \leftarrow f(\boldsymbol{x}^{i-1} | x^{i-1}_{e_i} \leftarrow \hat{u}_a) - f(\boldsymbol{x}^{i-1})$;

  //$\delta \in [0, \bar{\delta}]$ is the additive error level.

  find $\hat{u}_b$ s.t. $f(\boldsymbol{y}^{i-1} | y^{i-1}_{e_i} \leftarrow \hat{u}_b) \geq \max_{u_b \in [\underline{u}_{e_i}, \bar{u}_{e_i}]} f(\boldsymbol{y}^{i-1} | y^{i-1}_{e_i} \leftarrow u_b) - \delta$, $\delta_b \leftarrow f(\boldsymbol{y}^{i-1} | y^{i-1}_{e_i} \leftarrow \hat{u}_b) - f(\boldsymbol{y}^{i-1})$;

  if $\delta_a \geq \delta_b$ **then**

    $\boldsymbol{x}^i \leftarrow (\boldsymbol{x}^{i-1} | x^{i-1}_{e_i} \leftarrow \hat{u}_a)$, $\boldsymbol{y}^i \leftarrow \boldsymbol{y}^{i-1}$;

    $\boldsymbol{x}^{i+1} \leftarrow \boldsymbol{x}^i$, $\boldsymbol{y}^{i+1} \leftarrow (\boldsymbol{y}^i | y^i_{e_i} \leftarrow \hat{u}_a)$;

  else

    $\boldsymbol{y}^i \leftarrow (\boldsymbol{y}^{i-1} | y^{i-1}_{e_i} \leftarrow \hat{u}_b)$, $\boldsymbol{x}^i \leftarrow \boldsymbol{x}^{i-1}$;

    $\boldsymbol{y}^{i+1} \leftarrow \boldsymbol{y}^i$, $\boldsymbol{x}^{i+1} \leftarrow (\boldsymbol{x}^i | x^i_{e_i} \leftarrow \hat{u}_b)$;

Return $\boldsymbol{x}^{2n}$ (or $\boldsymbol{y}^{2n}$) ; *//note that $\boldsymbol{x}^{2n} = \boldsymbol{y}^{2n}$*

Lemma 4 is used to demonstrate that the objective value of each intermediate solution is non-decreasing,

Lemma 4. *$\forall i = 1, 2, \cdots, 2n$, one has,*

$$f(\boldsymbol{x}^i) \geq f(\boldsymbol{x}^{i-1}) - \delta, \quad f(\boldsymbol{y}^i) \geq f(\boldsymbol{y}^{i-1}) - \delta. \tag{10}$$

Proof of Lemma 4. Let $j := e_i$ be the coordinate that is going to be changed. From submodularity,

$$f(\boldsymbol{x}^{i-1} | x^{i-1}_j \leftarrow \bar{u}_j) + f(\boldsymbol{y}^{i-1} | y^{i-1}_j \leftarrow \underline{u}_j) \geq f(\boldsymbol{x}^{i-1}) + f(\boldsymbol{y}^{i-1})$$

So one can verify that $\delta_a + \delta_b \geq -2\delta$. Let us consider the following two situations:

1) If $\delta_a \geq \delta_b$, $\boldsymbol{x}$ is changed first.

We can see that the Lemma holds for the first change (where $\boldsymbol{x}^{i-1} \to \boldsymbol{x}^i, \boldsymbol{y}^i = \boldsymbol{y}^{i-1}$). For the second change, we are left to prove $f(\boldsymbol{y}^{i+1}) \geq f(\boldsymbol{y}^i) - \delta$. From submodularity:

$$f(\boldsymbol{y}^{i-1} | y^{i-1}_j \leftarrow \hat{u}_a) + f(\boldsymbol{x}^{i-1} | x^{i-1}_j \leftarrow \bar{u}_j) \geq f(\boldsymbol{x}^{i-1} | x^{i-1}_j \leftarrow \hat{u}_a) + f(\boldsymbol{y}^{i-1}) \tag{11}$$

Therefore, $f(\boldsymbol{y}^{i+1}) - f(\boldsymbol{y}^i) \geq f(\boldsymbol{x}^{i-1} | x^{i-1}_j \leftarrow \hat{u}_a) - f(\boldsymbol{x}^{i-1} | x^{i-1}_j \leftarrow \bar{u}_j) \geq -\delta$, the last inequality comes from the selection rule of δ_a in the algorithm.

2) Otherwise, $\delta_a < \delta_b$, $\boldsymbol{y}$ is changed first.

The Lemma holds for the first change ($\boldsymbol{y}^{i-1} \to \boldsymbol{y}^i, \boldsymbol{x}^i = \boldsymbol{x}^{i-1}$). For the second change, we are left to prove $f(\boldsymbol{x}^{i+1}) \geq f(\boldsymbol{x}^i) - \delta$. From submodularity,

$$f(\boldsymbol{x}^{i-1}|x_j^{i-1}\leftarrow\hat{u}_b) + f(\boldsymbol{y}^{i-1}|y_j^{i-1}\leftarrow\underline{u}_j) \geq f(\boldsymbol{y}^{i-1}|y_j^{i-1}\leftarrow\hat{u}_b) + f(\boldsymbol{x}^{i-1}), \tag{12}$$

So $f(\boldsymbol{x}^{i+1}) - f(\boldsymbol{x}^i) \geq f(\boldsymbol{y}^{i-1}|y_j^{i-1}\leftarrow\hat{u}_b) - f(\boldsymbol{y}^{i-1}|y_j^{i-1}\leftarrow\underline{u}_j) \geq -\delta$, the last inequality also comes from the selection rule of δ_b. □

Let $OPT^i := (\boldsymbol{x}^* \vee \boldsymbol{x}^i) \wedge \boldsymbol{y}^i$, it is easy to observe that $OPT^0 = \boldsymbol{x}^*$ and $OPT^{2n} = \boldsymbol{x}^{2n} = \boldsymbol{y}^{2n}$.

Lemma 5. *$\forall i = 1, 2, \cdots, 2n$, it holds,*

$$f(OPT^{i-1}) - f(OPT^i) \leq f(\boldsymbol{x}^i) - f(\boldsymbol{x}^{i-1}) + f(\boldsymbol{y}^i) - f(\boldsymbol{y}^{i-1}) + 2\delta. \tag{13}$$

Before proving Lemma 5, let us get some intuition about it. We can see that when changing i from 0 to $2n$, the objective value changes from the optimal value $f(\boldsymbol{x}^*)$ to the value returned by the algorithm: $f(\boldsymbol{x}^{2n})$. Lemma 5 is then used to bound the objective loss from the assumed optimal objective in each iteration.

Proof. Let $j := e_i$ be the coordinate that will be changed.

First of all, let us assume $\boldsymbol{x}$ is changed, $\boldsymbol{y}$ is kept unchanged ($\boldsymbol{x}^i \neq \boldsymbol{x}^{i-1}, \boldsymbol{y}^i = \boldsymbol{y}^{i-1}$), this could happen in four situations: 1.1) $x_j^i \leq x_j^*$ and $\delta_a \geq \delta_b$; 1.2) $x_j^i \leq x_j^*$ and $\delta_a < \delta_b$; 2.1) $x_j^i > x_j^*$ and $\delta_a \geq \delta_b$; 2.2) $x_j^i > x_j^*$ and $\delta_a < \delta_b$. Let us prove the four situations one by one.

If $x_j^i \leq x_j^*$, the Lemma holds in the following two situations:

1.1) When $\delta_a \geq \delta_b$, it happens in the first change: $x_j^i = \hat{u}_a \leq x_j^*$, so $OPT^i = OPT^{i-1}$; According to Lemma 4, $\delta_a + \delta_b \geq -2\delta$, so $f(\boldsymbol{x}^i) - f(\boldsymbol{x}^{i-1}) + f(\boldsymbol{y}^i) - f(\boldsymbol{y}^{i-1}) + 2\delta \geq 0$, so the Lemma holds;

1.2) When $\delta_a < \delta_b$, it happens in the second change: $x_j^i = \hat{u}_b \leq x_j^*, y_j^i = y_j^{i-1} = \hat{u}_b$, and since $OPT^{i-1} = (\boldsymbol{x}^* \vee \boldsymbol{x}^{i-1}) \wedge \boldsymbol{y}^{i-1}$, so $OPT_j^{i-1} = \hat{u}_b$ and $OPT_j^i = \hat{u}_b$, so one still has $OPT^i = OPT^{i-1}$. So it amouts to prove that $\delta_a + \delta_b \geq -2\delta$, which is true according to Lemma 4.

Else if $x_j^i > x_j^*$, it holds that $OPT_j^i = x_j^i$, all other coordinates of OPT^{i-1} remain unchanged. The Lemma holds in the following two situations:

2.1) When $\delta_a \geq \delta_b$, it happens in the first change. One has $OPT_j^i = x_j^i = \hat{u}_a$, $x_j^{i-1} = \underline{u}_j$, so $OPT_j^{i-1} = x_j^*$. And $x_j^i = \hat{u}_a > x_j^*, y_j^{i-1} = \bar{u}_j$. From submodularity,

$$f(OPT^i) + f(\boldsymbol{y}^{i-1}|y_j^{i-1}\leftarrow x_j^*) \geq f(OPT^{i-1}) + f(\boldsymbol{y}^{i-1}|y_j^{i-1}\leftarrow\hat{u}_a) \tag{14}$$

Suppose by virtue of contradiction that,

$$f(OPT^{i-1}) - f(OPT^i) > f(\boldsymbol{x}^i) - f(\boldsymbol{x}^{i-1}) + 2\delta \tag{15}$$

Summing Eq. 14 and 15 we get:

$$0 > f(\boldsymbol{x}^i) - f(\boldsymbol{x}^{i-1}) + \delta + f(\boldsymbol{y}^{i-1}|y_j^{i-1}\leftarrow\hat{u}_a) - f(\boldsymbol{y}^{i-1}|y_j^{i-1}\leftarrow x_j^*) + \delta \tag{16}$$

Because $\delta_a \geq \delta_b$ then from the selection rule of δ_b,

$$\delta_a = f(\boldsymbol{x}^i) - f(\boldsymbol{x}^{i-1}) \geq \delta_b \geq f(\boldsymbol{y}^{i-1}|y_j^{i-1}\leftarrow c) - f(\boldsymbol{y}^{i-1}) - \delta, \forall \underline{u}_j \leq c \leq \bar{u}_j. \tag{17}$$

Setting $c = x_j^*$ and substitite (17) into (16), one can get,

$$0 > f(\boldsymbol{y}^{i-1}|y_j^{i-1}\leftarrow\hat{u}_a) - f(y^{i-1}) + \delta = f(\boldsymbol{y}^{i+1}) - f(\boldsymbol{y}^i) + \delta, \tag{18}$$

which contradicts with Lemma 4.

2.2) When $\delta_a < \delta_b$, it happens in the second change. $y_j^{i-1} = \hat{u}_b, x_j^i = \hat{u}_b > x_j^*, OPT_j^i = \hat{u}_b, OPT_j^{i-1} = x_j^*$. From submodularity,

$$f(OPT^i) + f(\boldsymbol{y}^{i-1}|y_j^{i-1}\leftarrow x_j^*) \geq f(OPT^{i-1}) + f(\boldsymbol{y}^{i-1}|y_j^{i-1}\leftarrow \hat{u}_b) \tag{19}$$

Suppose by virtue of contradiction that,

$$f(OPT^{i-1}) - f(OPT^i) > f(\boldsymbol{x}^i) - f(\boldsymbol{x}^{i-1}) + 2\delta. \tag{20}$$

Summing Equations 19 and 20 we get:

$$0 > f(\boldsymbol{x}^i) - f(\boldsymbol{x}^{i-1}) + \delta + f(\boldsymbol{y}^{i-1}|y_j^{i-1}\leftarrow \hat{u}_b) - f(\boldsymbol{y}^{i-1}|y_j^{i-1}\leftarrow x_j^*) + \delta. \tag{21}$$

From Lemma 4 we have $f(\boldsymbol{x}^i) - f(\boldsymbol{x}^{i-1}) + \delta \geq 0$, so $0 > f(\boldsymbol{y}^{i-1}|y_j^{i-1} \leftarrow \hat{u}_b) - f(\boldsymbol{y}^{i-1}|y_j^{i-1} \leftarrow x_j^*) + \delta$, which contradicts with the selection rule of δ_b.

The case when $\boldsymbol{y}$ is changed, $\boldsymbol{x}$ is kept unchanged is similar, the proof of which is omitted here. □

With Lemma 5 at hand, one can prove Theorem 2: Taking a sum over i from 1 to $2n$, one can get,

$$\begin{aligned} f(OPT^0) - f(OPT^{2n}) &\leq f(\boldsymbol{x}^{2n}) - f(\boldsymbol{x}^0) + f(\boldsymbol{y}^{2n}) - f(\boldsymbol{y}^0) + 4n\delta \\ &= f(\boldsymbol{x}^{2n}) + f(\boldsymbol{y}^{2n}) - (f(\underline{\boldsymbol{u}}) + f(\bar{\boldsymbol{u}})) + 4n\delta \\ &\leq f(\boldsymbol{x}^{2n}) + f(\boldsymbol{y}^{2n}) + 4n\delta \end{aligned}$$

Then it is easy to see that $f(\boldsymbol{x}^{2n}) = f(\boldsymbol{y}^{2n}) \geq \frac{1}{3} f(\boldsymbol{x}^*) - \frac{4n}{3}\delta$.

E Details of revenue maximization with continuous assignments

E.1 More details about the model

From the discussion in the main text, $R_s(\boldsymbol{x}^i)$ should be some non-negative, non-decreasing, submodular function, we set $R_s(\boldsymbol{x}^i) := \sqrt{\sum_{t:x_t^i \neq 0} x_t^i w_{st}}$, where w_{st} is the weight of edge connecting users s and t. The first part in R.H.S. of Eq. 6 models the revenue from users who have not received free assignments, while the second and third parts model the revenue from users who have gotten the free assignments. We use w_{tt} to denote the "self-activation rate" of user t: Given certain amount of free trail to user t, how probable is it that he/she will buy after the trial. The intuition of modeling the second part in R.H.S. of Eq. 6 is: Given the users more free assignments, they are more likely to buy the product after using it. Therefore, we model the expected revenue in this part by $\phi(x_t^i) = w_{tt} x_t^i$; The intuition of modeling the third part in R.H.S. of Eq. 6 is: Giving the users more free assignments, the revenue could decrease, since the users use the product for free for a longer period. As a simple example, the decrease in the revenue can be modeled as $\gamma \sum_{t:x_t^i \neq 0} -x_t^i$.

E.2 Proof of Lemma 2

Proof. First of all, we prove that $g(\boldsymbol{x}) := \sum_{s:x_s=0} R_s(\boldsymbol{x})$ is a non-negative submodular function.

It is easy to see that $g(\boldsymbol{x})$ is non-negative. To prove that $g(\boldsymbol{x})$ is submodular, one just need,

$$g(\boldsymbol{a}) + g(\boldsymbol{b}) \geq g(\boldsymbol{a} \vee \boldsymbol{b}) + g(\boldsymbol{a} \wedge \boldsymbol{b}), \quad \forall \boldsymbol{a}, \boldsymbol{b} \in [0, \bar{\boldsymbol{u}}]. \tag{22}$$

Let $A := \text{supp}(\boldsymbol{a}), B := \text{supp}(\boldsymbol{b})$, where $\text{supp}(\boldsymbol{x}) := \{i | x_i \neq 0\}$ is the support of the vector $\boldsymbol{x}$. First of all, because $R_s(\boldsymbol{x})$ is non-decreasing, and $\boldsymbol{b} \geq \boldsymbol{a} \wedge \boldsymbol{b}$, $\boldsymbol{a} \geq \boldsymbol{a} \wedge \boldsymbol{b}$,

$$\sum_{s \in A \setminus B} R_s(\boldsymbol{b}) + \sum_{s \in B \setminus A} R_s(\boldsymbol{a}) \geq \sum_{s \in A \setminus B} R_s(\boldsymbol{a} \wedge \boldsymbol{b}) + \sum_{s \in B \setminus A} R_s(\boldsymbol{a} \wedge \boldsymbol{b}). \tag{23}$$

By submodularity of $R_s(\boldsymbol{x})$, and summing over $s \in E \setminus (A \cup B)$,

$$\sum_{s \in E \setminus (A \cup B)} R_s(\boldsymbol{a}) + \sum_{s \in E \setminus (A \cup B)} R_s(\boldsymbol{b}) \geq \sum_{s \in E \setminus (A \cup B)} R_s(\boldsymbol{a} \vee \boldsymbol{b}) + \sum_{s \in E \setminus (A \cup B)} R_s(\boldsymbol{a} \wedge \boldsymbol{b}). \tag{24}$$

Summing Equations 23 and 24 one can get

$$\sum_{s\in E\setminus A} R_s(\boldsymbol{a}) + \sum_{s\in E\setminus B} R_s(\boldsymbol{b}) \geq \sum_{s\in E\setminus(A\cup B)} R_s(\boldsymbol{a}\vee\boldsymbol{b}) + \sum_{s\in E\setminus(A\cap B)} R_s(\boldsymbol{a}\wedge\boldsymbol{b})$$

which is equivalent to Eq. 22.

Then we prove that $h(\boldsymbol{x}) := \sum_{t:x_t\neq 0} \bar{R}_t(\boldsymbol{x})$ is submodular. Because $\bar{R}_t(\boldsymbol{x})$ is non-increasing, and $\boldsymbol{a} \leq \boldsymbol{a}\vee\boldsymbol{b}$, $\boldsymbol{b} \leq \boldsymbol{a}\vee\boldsymbol{b}$,

$$\sum_{t\in A\setminus B} \bar{R}_t(\boldsymbol{a}) + \sum_{t\in B\setminus A} \bar{R}_t(\boldsymbol{b}) \geq \sum_{t\in A\setminus B} \bar{R}_t(\boldsymbol{a}\vee\boldsymbol{b}) + \sum_{t\in B\setminus A} \bar{R}_t(\boldsymbol{a}\vee\boldsymbol{b}). \tag{25}$$

By submodularity of $\bar{R}_t(\boldsymbol{x})$, and summing over $t \in A\cap B$,

$$\sum_{t\in A\cap B} \bar{R}_t(\boldsymbol{a}) + \sum_{t\in A\cap B} \bar{R}_t(\boldsymbol{b}) \geq \sum_{t\in A\cap B} \bar{R}_t(\boldsymbol{a}\vee\boldsymbol{b}) + \sum_{t\in A\cap B} \bar{R}_t(\boldsymbol{a}\wedge\boldsymbol{b}). \tag{26}$$

Summing Equations 25, 26 we get,

$$\sum_{t\in A} \bar{R}_t(\boldsymbol{a}) + \sum_{t\in B} \bar{R}_t(\boldsymbol{b}) \geq \sum_{t\in A\cup B} \bar{R}_t(\boldsymbol{a}\vee\boldsymbol{b}) + \sum_{t\in A\cap B} \bar{R}_t(\boldsymbol{a}\wedge\boldsymbol{b})$$

which is equivalent to $h(\boldsymbol{a}) + h(\boldsymbol{b}) \geq h(\boldsymbol{a}\vee\boldsymbol{b}) + h(\boldsymbol{a}\wedge\boldsymbol{b})$, $\forall \boldsymbol{a}, \boldsymbol{b} \in [0, \bar{\boldsymbol{u}}]$, thus proving the submodularity of $h(\boldsymbol{x})$.

Finally, because $f(\boldsymbol{x})$ is the sum of two submodular functions and one modular function, so it is submodular. □

E.3 Solving the 1-D subproblem when applying the DoubleGreedy algorithm

Suppose we are varying $x_j \in [0, \bar{u}_j]$ to maximize $f(\boldsymbol{x})$, notice that this 1-D subproblem is non-smooth and discontinuous at point 0. First of all, let us leave $x_j = 0$ out, one can see that $f(\boldsymbol{x})$ is concave and smooth along $\boldsymbol{\chi}_j$ when $x_j \in (0, \bar{u}_j]$,

$$\frac{\partial f(\boldsymbol{x})}{\partial x_j} = \alpha \sum_{s\neq j: x_s=0} \frac{w_{sj}}{2\sqrt{\sum_{t:x_t\neq 0} x_t w_{st}}} - \gamma + \beta w_{jj}$$

$$\frac{\partial^2 f(\boldsymbol{x})}{\partial x_j^2} = -\frac{1}{4}\alpha \sum_{s\neq j: x_s=0} \frac{w_{sj}^2}{\left(\sqrt{\sum_{t:x_t\neq 0} x_t w_{st}}\right)^3} \leq 0.$$

Let $\bar{f}(z)$ be the univariate function when $x_j \in (0, \bar{u}_j]$, then we extend the domain of $\bar{f}(z)$ to be $z \in [0, \bar{u}_j]$ as,

$$\bar{f}(z) = \bar{f}(x_j) := \alpha \sum_{s\neq j: x_s=0} R_s(\boldsymbol{x}) + \beta \sum_{t\neq j: x_t\neq 0} \phi(x_t) + \gamma \sum_{t\neq j: x_t\neq 0} \bar{R}_t(\boldsymbol{x}) + \beta\phi(x_j) + \gamma \bar{R}_j(\boldsymbol{x}).$$

One can see that $\bar{f}(z)$ is concave and smooth. Now to solve the 1-D subproblem, we can first of all solve the smooth concave 1-D maximization problem[9]: $z^* := \arg\max_{z\in[0,\bar{u}_j]} \bar{f}(z)$, then compare $\bar{f}(z^*)$ with the function value at the discontinuous point 0: $f(\boldsymbol{x}|x_j\leftarrow 0)$, and return the point with the larger function value.

F More applications

Multi-resolution summarization. Suppose we have a collection of items, e.g., images $E = \{e_1, \cdots, e_n\}$. Our goal is to extract a representative summary, where representativeness is defined w.r.t. a submodular set function $F : 2^E \to \mathbb{R}$. However, instead of returning a single set, our goal is to obtain summaries at multiple levels of detail or resolution. One way to achieve this goal is to assign each item e_i a nonnegative score x_i. Given a user-tunable threshold τ, the resulting summary $S_\tau = \{e_i | x_i \geq \tau\}$ is the set of items with scores exceeding τ. Thus, instead of solving the discrete problem of selecting a fixed set S, we pursue the goal to optimize over the scores, e.g., to use the following submodular continuous function,

$$f(\boldsymbol{x}) = \sum\nolimits_{i\in E} \sum\nolimits_{j\in E} \phi(x_j) s_{i,j} - \sum\nolimits_{i\in E} \sum\nolimits_{j\in E} x_i x_j s_{i,j}, \tag{27}$$

where $s_{i,j} \geq 0$ is the similarity between items i, j, and $\phi(\cdot)$ is a non-decreasing concave function.

[9] It can be efficienlty solved by various methods, e.g., the bisection method or Newton method.

Facility location. The classical discrete facility location problem can be naturally generalized to the continuous case where the scale of a facility is determined by a continuous value in interval $[0, \bar{\boldsymbol{u}}]$. For a set of facilities E, let $\boldsymbol{x} \in \mathbb{R}_+^E$ be the scale of all facilities. The goal is to decide how large each facility should be in order to optimally serve a set T of customers. For a facility s of scale x_s, let $p_{st}(x_s)$ be the value of service it can provide to customer $t \in T$, where $p_{st}(x_s)$ is a normalized monotone function ($p_{st}(0) = 0$). Assuming each customer chooses the facility with highest value, the total service provided to all customers is $f(\boldsymbol{x}) = \sum_{t \in T} \max_{s \in E} p_{st}(x_s)$. It can be shown that f is monotone submodular.

Maximum coverage. In the maximum coverage problem, there are n subsets $C_1, \cdots, C_n$ from the ground set V. One subset C_i can be chosen with "confidence" level $x_i \in [0, 1]$, the set of covered elements when choosing subset C_i with confidence x_i can be modeled with the following monotone normalized covering function: $p_i : \mathbb{R}_+ \to 2^V, i = 1, \cdots, n$. The target is to choose subsets from $C_1, \cdots, C_n$ with confidence level to maximize the number of covered elements $|\cup_{i=1}^n p_i(x_i)|$, at the same time respecting the budget constraint $\sum_i c_i x_i \leq b$ (where c_i is the cost of choosing subset C_i). This problem generalizes the classical maximum coverage problem. It is easy to see that the objective function is monotone submodular, and the constraint is a down-closed polytope.

Text summarization. Submodularity-based objective functions for text summarization perform well in practice [38]. Let C to be the set of all concepts, and E to be the set of all sentences. As a typical example, the concept-based summarization aims to find a subset S of the sentences to maximize the total credit of concepts covered by S. [48] discussed extending the submodular text summarization model to the one that incorporates "confidence" of a sentence, which has discrete value, and modeling the objective to be a monotone submodular lattice function. It is also natural to model the confidence level of sentence i to be a continuous value $x_i \in [0, 1]$. Let us use $p_i(x_i)$ to denote the set of covered concepts when selecting sentence i with confidence level x_i, it can be a monotone covering function $p_i : \mathbb{R}_+ \to 2^C, \forall i \in E$. Then the objective function of the extended model is $f(\boldsymbol{x}) = \sum_{j \in \cup_i p_i(x_i)} c_j$, where $c_j \in \mathbb{R}_+$ is the credit of concept j. It can be verified that this objective is a monotone submodular continuous function.